ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TESIS PREVIO A LA OBTENCIÓN DEL TíTULO DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTOR: ANDRADE GUANOQUIZA, FERNANDA LUCÍA

TEMA: EXTRACCIÓN INTERACTIVA DE OBJETOS EN RESTAURACIÓN DIGITAL DE IMÁGENES

DIRECTOR: DR. VINICIO CARRERA

CODIRECTOR: ING. JULIO LARCO

SANGOLQUÍ

2015

Certificado de tutoría

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICADO

Ing. Vinicio Carrera, PhD.

Ing. Julio Larco, MSc.

CERTIFICAN

Que el trabajo titulado "EXTRACCIÓN INTERACTIVA DE OBJETOS EN RESTAURACIÓN DIGITAL DE IMÁGENES", realizado por Fernanda Lucía Andrade Guanoquiza, ha sido guiado y revisado periódicamente y cumple normas estatutarias establecidas por la Universidad de las Fuerzas Armadas - ESPE en su reglamento.

Debido a que se trata de un trabajo de investigación, recomiendan su publicación.

Sangolquí, 21 de agosto de 2015.

Ing. Vinicio Carrera, PhD.	Ing. Julio Larco, MSc.
DIRECTOR	CODIRECTOR

Declaración de responsabilidad

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

FERNANDA LUCÍA ANDRADE GUANOQUIZA

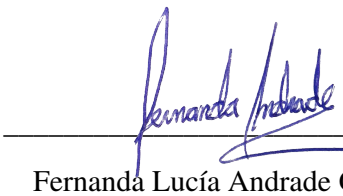
DECLARO QUE:

El proyecto de grado denominado "EXTRACCIÓN INTERACTIVA DE OBJETOS EN RESTAURACIÓN DIGITAL DE IMÁGENES" ha sido desarrollado en base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan al pie, de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, 21 de agosto de 2015.



Fernanda Lucía Andrade Guanoquiza

Autorización de publicación

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

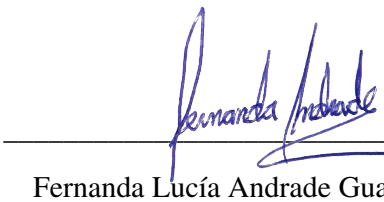
INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

FERNANDA LUCÍA ANDRADE GUANOQUIZA

Autorizo a la Universidad de las Fuerzas Armadas - ESPE la publicación en la biblioteca virtual de la Institución del trabajo "EXTRACCIÓN INTERACTIVA DE OBJETOS EN RESTAURACIÓN DIGITAL DE IMÁGENES", cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría.

Sangolquí, 21 de agosto de 2015.



Fernanda Lucía Andrade Guanoquiza

DEDICATORIA

A mis padres y mi tía Guadalupe.

AGRADECIMIENTO

Este trabajo es la culminación de un periodo excepcional de aprendizaje y debe su existencia a un grupo de personas que han cooperado de diversas maneras, y a quienes doy las gracias por haber estado a mi lado cuando me han hecho falta.

Agradezco al doctor Vinicio Carrera. Su afán por el trabajo minucioso y su mirada rigurosa han contribuido a agudizar mi capacidad de análisis y constituyen, además, la principal referencia en mi formación académica. Las incontables discusiones y reflexiones sobre este trabajo no solo me han guiado en su realización, sino que me han permitido descubrir el entusiasmo que esconde desentrañar nuevos conocimientos.

El ingeniero Julio Larco ha sido también un importante apoyo para la elaboración de este proyecto. Sus observaciones y correcciones han enriquecido, sin duda, la calidad de mi trabajo.

No estaría aquí, por su puesto, sin mi familia. Su guía impulsó la curiosidad y respaldó la libertad intelectual para seguir aprendiendo y reflexionando. Mi primeros pasos, gracias a ellos, fueron en un lugar donde la ética, el conocimiento y el cariño iban de la mano. Mi padre me enseñó el discernimiento; mi madre, la generosidad y mi tía, el fervor. Gracias, Sebastián, por la camaradería y las elucubraciones.

Este agradecimiento no estaría completo sin la mención a mis amigos, cuyos lazos de amistad han iluminado este sinuoso pero radiante camino. Todas esas noches sin dormir fueron tolerables gracias a sus bromas, ocurrencias y sonrisas. Gracias, Dennys, Chalo, Nico, Dan, Santiago, Eve, Nataly, Clau, Carlitos, Katty, Roberto, Rayhra, Many, Mancheche y toda la juerga de TM3 y la mansión Rodríguez.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La eliminación de objetos es una tarea comúnmente realizada en la edición de imágenes. Por consiguiente, en este trabajo se exploran los métodos requeridos para esta tarea, la cual requiere el trabajo conjunto de los proceso de segmentación y restauración. En el proceso de segmentación, el usuario señala el objeto y la imagen es dividida en dos regiones, de manera que los pixeles pertenecientes al objeto son separados de los demás para su eliminación. Posteriormente, el proceso de restauración repara la región de donde se extrajo este objeto utilizando la información adyacente a esta zona. En consecuencia, aquí se evalúan diversos algoritmos de segmentación interactiva, especialmente los métodos basados en *Graph Cuts* que destacan por su alto desempeño y eficiencia computacional. Para esta evaluación se ha recopilado una base de datos con imágenes, referencias y semillas; lo cual no sólo permite analizar el influjo de la intervención del usuario, sino que también posibilita la reproducibilidad de los resultados presentados. De manera similar, también se analiza la efectividad de distintos algoritmos de restauración de imágenes. Los resultados muestran que *GrabCut* y el método de Criminisi son los algoritmos más robustos para aplicaciones de extracción interactiva de objetos en imágenes.

Palabras clave:

- PROCESAMIENTO DIGITAL DE IMÁGENES
- SEGMENTACIÓN INTERACTIVA
- RESTAURACIÓN DE IMÁGENES
- GRAFOS
- GRAPH CUTS

ABSTRACT

Object removal is a task usually performed in image editing applications. For this reason, required methods for this two-stage task are explored: segmentation and inpainting. In the segmentation stage, the image is divided into two regiones using user guidance. Here, the pixels of the object are separated from the others in order to extract the object from the image. Next, the inpainting stage repairs the empty region using adjacent information. Therefore, this work evaluates and compares several interactive segmentation algorithms, particularly graph cut based segmentation methods, which have been extensively explored because they report high performance and computational efficiency. Consequently, a novel seed-based user input dataset is introduced in the aim of presenting reproducible results. Similarly, effectiveness of a number of inpainting algorithms are analyzed. Results show that GrabCut and Criminisi are the most robust methods among all the evaluated algorithms for interactive object removal.

Keywords:

- DIGITAL IMAGE PROCESSING
- INTERACTIVE SEGMENTATION
- INPAINTING
- GRAPHS
- GRAPH CUTS

CAPITULO 1

EXTRACCIÓN INTERACTIVA DE OBJETOS EN RESTAURACIÓN DIGITAL DE IMÁGENES

Bertalmio *et al.* [1] acuñaron la expresión *image inpainting* para describir el proceso de restauración digital de imágenes. Esta técnica, también conocida como *image completion* en inglés, se inspira en la restauración de arte, tarea fundamental para la recuperación y preservación del patrimonio cultural. Los restauradores especializados retocan y reparan áreas deterioradas de obras de arte con la intención de conservar la apariencia original de las mismas.

De manera similar, los algoritmos de restauración digital de imágenes reconstruyen la información perdida en regiones afectadas de una imagen utilizando la información espacial adyacente a estas regiones. Como resultado de este proceso, un observador casual no podría notar los daños anteriores de la imagen. Por este motivo, esta técnica es utilizada en diversas tareas, como restauración o eliminación de objetos de fotografías, vídeos o pinturas.

A diferencia de las técnicas de eliminación de ruido en procesamiento digital de señales, la información perdida de las imágenes no puede ser recuperada completamente por ningún método. La restauración digital, por tanto, puede ser considerada como manipulación de las imágenes porque crea una imagen nueva con el reemplazo de la información perdida. En suma, los algoritmos de restauración digital buscan crear

una imagen similar a la original utilizando únicamente la información disponible en la imagen [1].

En la actualidad existen principalmente dos clases de algoritmos para la restauración digital de imágenes según su enfoque: ecuaciones diferenciales parciales y parches [2]. Los algoritmos basados en ecuaciones diferenciales parciales rellenan la imagen a través de difusión de la información de los bordes. Por otro lado, los algoritmos basados en parches rellenan la región deteriorada copiando los pixeles de una región de características similares. Esta última aproximación es una forma eficiente de restaurar imágenes con amplias regiones afectadas [3].

Sin embargo, pese a que la eliminación de objetos es una de las aplicaciones más extendidas de la restauración de imágenes, la mayoría de los trabajos de investigación no se enfocan en la etapa de segmentación. Esto sucede a pesar de que en esta se extraen los objetos definidos por el usuario. Ciertamente, las investigaciones presentan resultados donde los objetos han sido eliminados manualmente de la imagen [2].

Por estas razones, este trabajo se enfoca en el proceso de eliminación de objetos de imágenes, el cual requiere la combinación de las tareas de segmentación para la extracción de objetos e *inpainting* para el relleno. Este proceso demanda que la extracción del objeto sea tan precisa como el proceso de relleno de la imagen.

El proceso de segmentación de imágenes divide una imagen en diferentes regiones que pueden tener diferente color, intensidad o textura. Típicamente, los métodos de segmentación se pueden dividir en cinco categorías: segmentación basada en umbral, en bordes, en regiones, en cuencas y en minimización de energía [4]. En esta última categoría, la segmentación es formulada en términos de una función de energía, construida según la información de contornos y regiones. Posteriormente, métodos como, por ejemplo, *Graph Cuts* son utilizados para obtener la configuración correspondiente a la mínima energía, logrando resultados óptimos globales [5]. Estos métodos basados en *Graph Cuts* han sido ampliamente explorados en los últimos años por su alto desem-

peño y eficiencia computacional, y porque permiten una formulación más flexible del problema de segmentación [6].

También es necesario considerar que la tarea de eliminación de objetos requiere la guía del usuario para identificarlos. Por estas razones, en este trabajo se exploran los algoritmos basados en *Graph Cuts*, particularmente aquellos algoritmos de segmentación interactiva que permiten la intervención del usuario para mejorar los resultados. En general, los usuarios marcan el fondo y el objeto de interés a través de cuadros delimitadores (*bounding boxes*) o semillas, también conocidos como *scribbles*.

Consecuentemente, aquí se evalúan los principales algoritmos de segmentación interactiva basados en *Graph Cuts* que, además, utilizan *scribbles* para la extracción de objetos en imágenes. Para esta evaluación se ha recopilado una nueva base de datos de imágenes con *scribbles*. Esto permite la reproducibilidad y repetitividad de los resultados presentados.

El análisis incluye un estudio cualitativo de diferentes métricas utilizadas en el estado del arte para la evaluación de algoritmos de segmentación. Es importante verificar la aptitud de estas métricas para mostrar el desempeño de los algoritmos de segmentación de imágenes.

De manera similar, en este trabajo se analiza el desempeño de diferentes algoritmos de restauración de imágenes como etapa de relleno de la región de donde el objeto fue extraído. No obstante, la calidad de una imagen restaurada se fundamenta en las opiniones de los observadores. Por lo cual, en este análisis se utilizan criterios cualitativos y cuantitativos para la evaluación de los algoritmos. Este enfoque es ventajoso porque permite constatar la efectividad de las métricas cuantitativas para reflejar la calidad percibida por los observadores.

CAPITULO 2

PROCESAMIENTO DIGITAL DE IMÁGENES

La visión es primordial en la percepción del ser humano. Las personas tienen la capacidad de adquirir e interpretar la información visual de su entorno a través de imágenes. Por lo tanto, no es extraño que el procesamiento digital de imágenes sea un área de investigación ampliamente explorada, puesto que las diferentes técnicas que se han desarrollado en este campo son fundamentales para emular la visión humana [7].

2.1. Procesamiento digital de imágenes

Las imágenes digitales son la representación en dos dimensiones de una escena física tridimensional. Estas imágenes son utilizadas a diario en un sinfín de aplicaciones en entretenimiento, arte, astronomía y medicina. Por consiguiente, el procesamiento digital de imágenes que surgió para resolver problemas relacionados a estas prácticas. Sin embargo, como Gonzalez y Woods [8] señalan, los problemas tratados por las técnicas de procesamiento digital de imágenes no tienen un límite claramente generalizado que los diferencie de otros campos como visión artificial y, para aclararlo, divide a las técnicas de manipulación de imágenes en tres categorías:

1. **Procesos de nivel bajo (*low-level vision*):** Operaciones básicas como preprocesamiento de imágenes para reducir ruido o mejorar contraste. Se caracterizan porque la entrada y salida de estos procesos son imágenes [8].

2. **Procesos de nivel medio (*mid-level vision*):** Tareas como segmentación para dividir una imagen en regiones u objetos, y reconocimiento de estos para clasificación. Las entradas de estos procesos también son imágenes, pero las salidas son atributos extraídos de las imágenes, como contornos o bordes [8].

3. **Procesos de nivel alto (*high-level vision*):** Procesos que utilizan los atributos de las imágenes para la entrada y la salida. Estos procesos buscan interpretar la imagen para intentar emular las funciones cognitivas asociadas a la visión [8].

El procesamiento digital de imágenes incluye técnicas de procesos de nivel bajo y medio [8]. Estas técnicas no sólo tienen como objetivo mejorar la imagen, sino que también procesan la imagen y extraen atributos que podrían ser utilizados en tareas de visión artificial.

2.2. Representación de la imagen digital

Una imagen es una función bidimensional continua $F(x,y)$ donde las variables x,y representan coordenadas espaciales en un plano, y la amplitud F es la intensidad en un punto x,y de la imagen como se muestra en la Figura 1. La imagen digital es el resultado del proceso de digitalización de la imagen $F(x,y)$.

En esta sección se explica la representación de una imagen digital en escala de grises. Este tipo de imágenes están compuestas por un canal que representa la intensidad, brillo o densidad de la imagen. Como se mencionó previamente, al final del proceso de digitalización, se obtiene una matriz de número reales que tiene M filas y N columnas, y donde los valores de las coordenadas (x,y) ahora son valores discretos [8].

Una imagen digital es, por tanto, una matriz donde las filas y columnas son utilizadas para representar un punto en la imagen, y el valor del elemento en ese punto corresponde al valor de intensidad de la imagen en ese punto como se muestra en la Figura 2. Cada elemento en esta matriz es llamado pixel [9].

Figura 1: Representación de una imagen.

La imagen digital es el resultado del proceso de digitalización de la imagen $F(x,y)$. Al final de este proceso se obtiene una matriz de número reales que tiene M filas y N columnas, y donde los valores de las coordenadas (x,y) ahora son valores discretos [8]. Una imagen digital es, por tanto, una matriz donde las filas y columnas son utilizadas para representar un punto en la imagen, y el valor del elemento en ese punto corresponde al valor de intensidad de la imagen en ese punto como se muestra en la Figura 2. Cada elemento en esta matriz es llamado pixel [9].

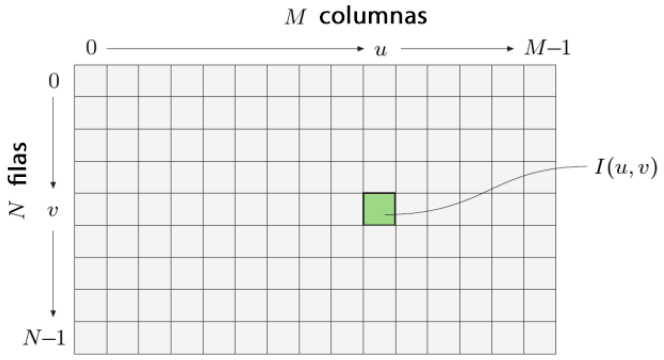


Figura 2: Representación de una imagen digital en una matriz $M \times N$.

Fuente: Adaptado de Burger y Burge [10].

Estas definiciones permiten representar matemáticamente la imagen digital $M \times N$ como la matriz de la ecuación 2.1.

$$f(x,y) = \begin{pmatrix} f_{0,0} & f_{0,1} & \cdots & f_{0,N-1} \\ f_{1,0} & f_{1,1} & \cdots & f_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ f_{M-1,0} & f_{M-1,1} & \cdots & f_{M-1,N-} \end{pmatrix} \tag{2.1}$$

Dado que las imágenes que se perciben están formadas por la cantidad de luz que se refleja en los objetos del entorno [9], la función $f(x,y)$ tiene dos componentes:

1. La cantidad de luz incidente en la escena.

2. La cantidad de luz reflejada por el objeto en la escena.

La luz incidente es la iluminación y se denota como $i(x,y)$, y, por otro lado, la luz reflejada se denota como $r(x,y)$. La función de la imagen $f(x,y)$ es el producto de estas dos componente como se expresa en la ecuación 2.2.

$$f(x,y) = i(x,y) \times r(x,y) \tag{2.2}$$

El valor de $i(x,y)$ está definido por la fuente de luz y el valor de $r(x,y)$ está determinado por las características del objeto en la escena. Por ejemplo, en un día iluminado, el valor de $i(x,y)$ en la superficie de la tierra es de alrededor de 90000 lm, mientras que, en un día nublado, la iluminación es menor a 10000 lm [9].

2.3. Imagen en colores

Las imágenes en color se representan utilizando una combinación de varios canales para formar un color. La mayoría de estas imágenes utilizan el sistema de color RGB y codifican los colores primarios, rojo, verde y azul, a través de tres canales [10]. De

igual manera que las imágenes en escala de grises, cada canal representa la intensidad de un componente de color y, en caso de un sistema de 8 bits por pixel, los valores tienen el rango de $[0 \ldots 255]$. En la Figura 3 se muestran los valores de intensidad de los tres canales RGB de una imagen en color.

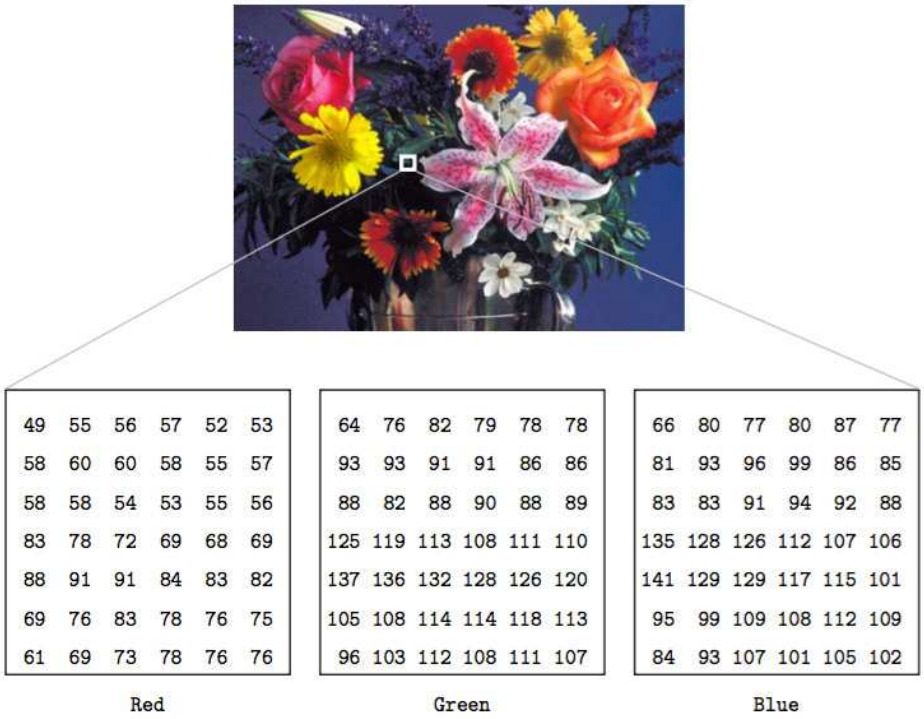


Figura 3: Ejemplo de imagen en color.

Fuente: McAndrew [11].

2.3.1. Procesamiento de imágenes en color

Las imágenes en color contienen más información que las imágenes en escala de grises, por lo que el procesamiento de estas imágenes son importantes para diversas aplicaciones. Para entender el procesamiento de las imágenes en color es necesario examinar diversos aspectos relacionados con el estudio de color: las propiedades físicas de la luz, la forma en que el ojo humano detecta el color y los mecanismos del cerebro para identificar los colores a través de la información del objeto [11].

La luz visible es parte del espectro electromagnético. El ojo humano es sensible a la radiación de longitud de onda entre 400 nm (violeta) a 770 nm (rojo) [12]. Este rango responde a una fracción del espectro como se observa en la Figura 4. Existen, sin embargo, sistemas que responden a rangos más amplios: 780 nm a 1400 nm, 1400 nm a 3300 nm, 3 3 µm a 10 µm en el rango infrarrojo y de 100 nm a 380nm en el rango de luz ultravioleta. Esto ha sido posible debido a los sensores especiales que existen para este tipo de señales [12].

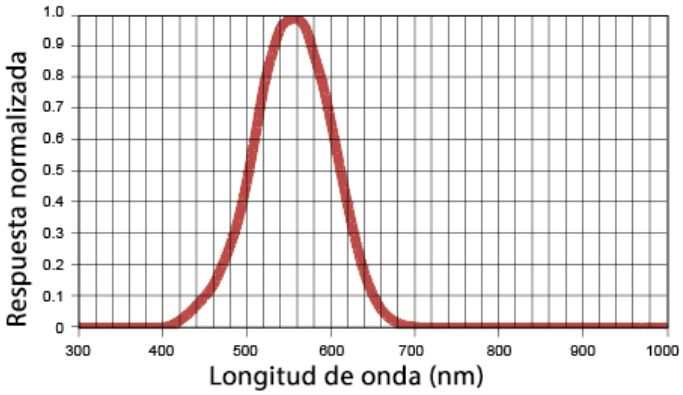


Figura 4: Rango espectral visible al ojo humano.

Fuente: Adaptado de Mehtar [13].

La visión humana tiende a percibir el color como la variación de diversas cantidades de rojo, verde y azul. Esto se debe a que el ojo humano es más sensible a estos colores llamados primarios. La cantidad de rojo, verde y azul percibidos en un color fue determinado por un experimento realizado la Comisión Internacional de Iluminación (CIE por su nombre en francés) [11].

2.3.2. Espacios de color

La representación de color ha sido uno de los problemas que se presentan en las aplicaciones de procesamiento de imágenes. La reducción del número de colores puede reducir la eficiencia de estas aplicaciones [12]. Un sistema de representación de color

debe ser adecuado para almacenar, mostrar y procesar imágenes a color. De hecho, esta representación debe estar de acuerdo a las demandas matemáticas de los algoritmos de procesamiento, a las condiciones técnicas de dispositivos como cámaras, monitores e impresoras. Sin embargo, no se puede cumplir con estos requerimiento simultáneamente y, por tal motivo, se han presentado diversos modelos de representación del color de acuerdo al objetivo de uso [14].

Los espacios de color definen sistemas de coordenadas de color donde cada punto representa un color específico [12].

2.3.2.1. RGB

En el espacio de color RGB, cada color se representa en función de tres componentes primarios: rojo, verde y azul. Este es el espacio de color que se utiliza para mostrar imágenes en pantallas de televisores o monitores de computadoras [8].

El espacio de color RGB se basa en el sistema de coordenadas cartesianas como se muestra en la Figura 5. Los colores primarios están en tres esquinas y los colores cian, magenta y amarillo están en los otras tres esquinas; el color negro está en el origen y el color blanco está en la esquina más distante del origen. Los colores son puntos dentro de este cubo y se definen por vectores que parten del origen del sistema cartesiano.

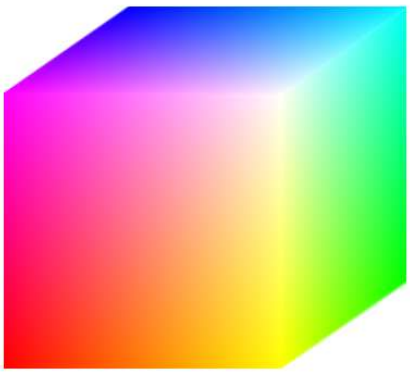


Figura 5: Representación del espacio RGB.

Fuente: Gonzalez y Woods[8].

2.3.2.2. CIELAB

El modelo de color RGB es muy útil para representar el color en dispositivos, pero no es adecuado para el modelo de la visión humana porque no es lineal. En un espacio de color uniforme la distancia euclidiana entre dos colores corresponde a la diferencia perceptiva de los dos colores en el sistema de visión humana definida [12].

El espacio de color CIELAB fue adoptado por CIE como un estándar internacional en 1970. La uniformidad de este espacio es ventajoso para análisis de similitud de colores, y ha sido utilizado exitosamente para tareas de agrupamiento de color [12].

La transformación de RGB a CIELAB requiere un paso intermedio [15]. En primer lugar, RGB se transforma a XYZ según las ecuaciones definidas en 2.3.

$$\begin{aligned} X &= 0{,}412453R + 0{,}357580G + 0{,}180423B \\ Y &= 0{,}212671R + 0{,}715160G + 0{,}072169B \\ Z &= 0{,}019334R + 0{,}119193G + 0{,}950227B \end{aligned} \tag{2.3}$$

Según estos valores XYZ, los componente $L*$, $a*$, $b*$ son

$$\begin{aligned} L* &= 116 \cdot (\frac{Y}{Y_n}) - 16 \\ a* &= 500 \cdot [f(\frac{X}{X_n}) - f(\frac{Y}{Y_n})] \\ b* &= 200 \cdot [f(\frac{Y}{Y_n}) - f(\frac{Z}{Z_n})] \end{aligned} \tag{2.4}$$

En las cuales la función f se define como:

$$f(x) = \begin{cases} q^{1/3} & \text{si } q > 0{,}008856 \\ 7{,}7871 + 16/116 & \end{cases} \tag{2.5}$$

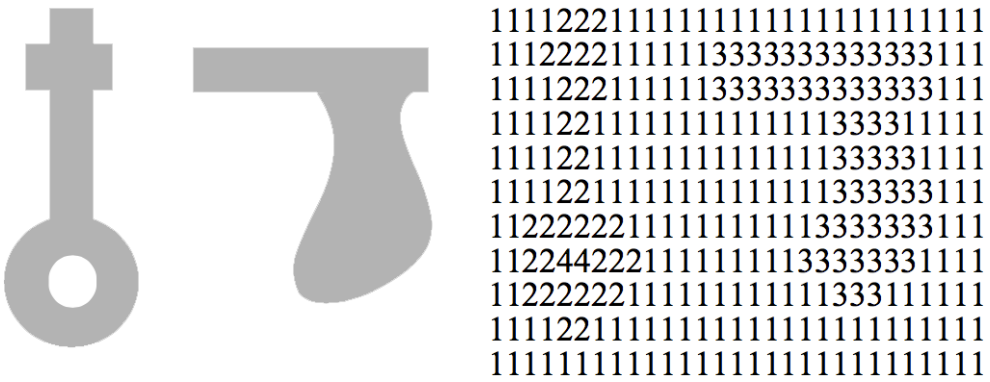
Donde X_n, Y_n y Z_n se calculan para un punto blanco de referencia de depende de la iluminación. Esta iluminación es usualmente definida por la temperatura de la fuente. Por ejemplo, para el estándar D65 (6500 K), R=G=B=100 en la ecuación 2.4 [15].

2.4. Segmentación de imágenes

Segmentación es un proceso en el cual una imagen es dividida en regiones distintivas. Haralick y Shapiro [16] señalan que las regiones de una imagen segmentada deben ser uniformes y homogéneas respecto a ciertas características, como tonos o textura.

Estas regiones pueden ser identificadas como conjuntos de pixeles que etiquetan a los objetos de una imagen. El proceso de segmentación, en definitiva, busca simplificar el análisis de la imagen original a través de esta representación fraccionada de sus objetos. En la Figura 2.6(b) se muestra los tres conjuntos de pixeles que identifican las tres regiones principales de la imagen de la Figura 2.6(a).

La segmentación es una etapa fundamental en sistemas de procesamiento de imágenes para obtener información sobre los objetos de una imagen [17].



(a) Imagen con tres regiones (b) Segmentación y etiquetado de imagen

Figura 6: Ejemplo de segmentación y etiquetado de una imagen con tres regiones. Fuente: Snyder y Qi [18].

Típicamente, lo métodos de segmentación se pueden distribuir en cinco categorías según el enfoque utilizado. La primera categoría corresponde a la segmentación basada en un umbral, donde la imagen es dividida en fondo (*background*) y primer plano (*foreground*). Cuando la intensidad de un pixel es mayor o menor a la definida por el valor del umbral, este es clasificado como fondo o primer plano [4].

La segunda categoría corresponde a los métodos basados en regiones que trabajan con un conjunto de nodos que deben ser identificados mediante un criterio definido por la similitud de intensidad, color o textura La tercera categoría es la de segmentación basada en bordes. Este método asume que el valor de los pixeles que conectan el fondo y el primer plano son distintos [4].

La cuarta categoría es la segmentación basada en cuencas. Estos métodos tienen problemas de segmentación debido al ruido o irregularidades. Finalmente, la quinta categoría corresponde a los métodos de segmentación basados en energía. Estos necesitan establecer una función que alcanzará su mínimo valor cuando la imagen es segmentada según lo esperado [4].

2.4.1. Segmentación basada en un umbral

La segmentación basada en un umbral es usada en sistemas de procesamiento donde los objetos de interés contrastan con el fondo. Este técnica divide a la imagen en dos regiones según un valor de intensidad. Para esto, cada pixel es comparado con un valor umbral: si el pixel es mayor al valor umbral, se etiqueta como primer plano (*foreground*) y si es menor, como fondo (*background*) [19].

Esta técnica crea una imagen binaria $b(x,y)$ a partir de una imagen $i(x,y)$ conforme el criterio expresado en 2.6 [20].

$$b(x) = \begin{cases} 1 & \text{si } i(x,y) \geq T \\ 0 & \text{si } i(x,y) < T \end{cases} \tag{2.6}$$

Donde T es el valor umbral de intensidad. La correcta elección de este valor es indispensable para una buena segmentación. Por ejemplo, en la Figura 7 se muestra un histograma bimodal de una imagen en escala de grises donde se identifica claramente los valores distintivos de intensidad de las dos regiones de la imagen. El valor del umbral puede tomar los valores entre T_1 y T_2, situados entre los dos picos de las

dos regiones. La región *A*, cuyos valores de intensidad están sobre el valor umbral, corresponde a la región de primer plano [19]

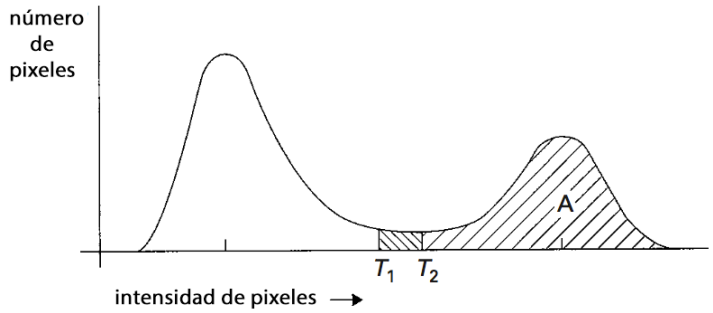


Figura 7: Valor de umbral en el histograma de una imagen en escala de grises.

Fuente: Adaptado de Dougherty [19].

Varios métodos han sido propuestos para la selección del valor umbral. El algoritmo *Isodata*, por ejemplo, selecciona el valor del umbral en función de los valores medios de intensidad de las dos regiones distintivas [21]. Por otro lado, el algoritmo *Triangle* es utilizado para imágenes en las cuales la región de primer plano no produce un pico muy alto en el histograma, y en donde el valor es umbral se calcula a través de la comparación de distancias entre los picos de las regiones [22].

Sin embargo, la segmentación basada en umbral tiene inconvenientes [20]:

- No considera relación espacial de los pixeles.
- Es sensible a las variaciones de iluminación de la escena.
- Sólo es aplicable a casos en que la imagen puede ser dividida entre objetos de interés de similar intensidad y un fondo de diferente intensidad.

2.4.2. Segmentación basada en regiones

Las segmentación basada en borde es un proceso que divide una imagen en diferentes regiones siguiendo un criterio. Si *R* representa la región de la imagen completa,

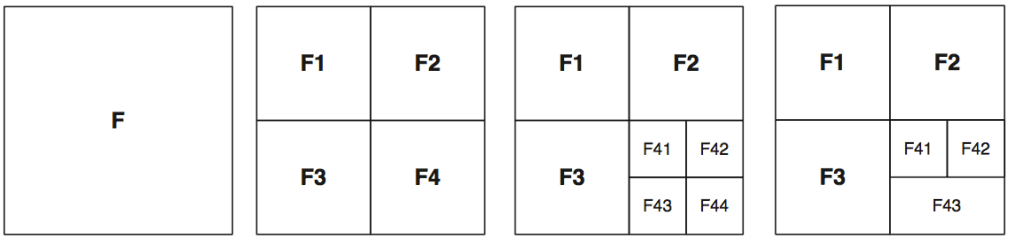
entonces la segmentación divide R en n subregiones $R_1, R_2, ..., R_n$ tal que [8]:

- Todo pixel debe pertenecer a una región.
- Los pixeles de una región deben conectarse acorde a un criterio de conectividad.
- Las regiones deben estar separadas.
- Los pixeles de una región deben tener una propiedad en común como, por ejemplo, un mismo nivel de intensidad.
- Las regiones deben tener propiedades diferentes entre ellas.

Las técnicas de *Region Growing* agrupan pixeles o subregiones en regiones más grandes a través de un proceso iterativo [15]:

1. Selección de semillas o *seeds* que representan las diferentes regiones. Estas semillas suelen ser elegidas por un usuario.
2. Selección de las propiedades que reflejan la pertenencia de los pixeles a cada región. Estas propiedades pueden incluir niveles de gris, textura y color.
3. Expansión iterativa de las regiones a través de la evaluación de la similitud de los pixeles adyacentes a las semillas.
4. Se comprueba de que no exista una region con un tamaño inferior al tamaño umbral definido por el usuario.

A diferencia de la operación con enfoque de abajo arriba de *Region Growing*, las técnicas de *Region Splitting* operan de arriba abajo. Estas inician con una región inicial, después evalúan la región para decidir si los pixeles satisfacen un criterio de homogeneidad. Si este criterio no se cumple, la región se divide en subregiones, las cuales serán nuevamente evaluadas a través de un proceso iterativo (Figura 8).



(a) Imagen completa (b) Primera división (c) Segunda divisón (d) Tercera división

Figura 8: Ejemplo de segmentación de una imagen utilizando técnicas de *Region Growing*. Fuente: Wu *et al.* [23].

Uno de los problemas de las técnicas de segmentación basada en regiones es que no consideran información de bordes, por lo que las regiones de los resultados pueden tener bordes irregulares [24].

2.4.3. Segmentación basada en bordes

Las técnicas de segmentación basadas en bordes se enfocan en la búsqueda de diferencias entre pixeles en lugar de similitudes. El objetivo es determinar los bordes de los objetos, lo cual permite diferenciar las diferentes regiones de la imagen. Para esto se sigue un proceso de detección e identificación de de bordes. [19].

Los bordes de una imagen son detectados a través operadores como Robert, Sobel, Prewitt, Canny o Krisch. El operador Robert, por ejemplo, es un operador gradiente de 2x2, mientras que el operador Sobel utiliza un operador gradiente de 3x3. [7]. Por esta razón estas técnicas son sencillas de implementar y de rápida ejecución ya que se realiza una operación de convolución de las imágenes con los filtros. Además, a diferencia de las técnicas basadas en regiones, no requiere información previa sobre la información de la imagen [24].

Existen casos de imágenes con ruido en que los bordes no aparecen completos después de la detección y, por tal motivo, se necesita una operación de unión de bordes adyacentes si tienen propiedades similares. Finalmente, después de la unión de bordes,

se procede a analizar los bordes para identificar bordes pertenecientes a un región.

La segmentación basada en bordes tiene tres problemas importantes [24]:

- Es muy sensible al ruido.
- Requiere la selección de un valor umbral para el borde.
- No genera bordes completos de los objetos cuando están solapados.

2.4.4. Segmentación basada en cuencas

En el procesamiento digital de imágenes, las cuencas se conocen en inglés como *watershed*. La segmentación toma este nombre porque se basa en conceptos de hidrología y topografía. Una imagen en escala de grises es considerada una superficie con diferentes alturas donde los pixeles con valores altos de intensidad son crestas y los pixeles con valores bajos de intensidad son valles [25].

Si una corriente de agua cayera desde algún punto alto, se movería hacia una superficie más baja hasta alcanzar una altitud mínima local. Esta acumulación de agua en un mínimo local es llamada cuenca hidrográfica o *catchment basin*. Todos los puntos que desembocan en la misma cuenca hidrográfica pertenecen a la misma fuente o *watershed*. Un valle es una región que es rodeada de crestas y, por otro lado, una cresta es el lugar de gradiente máxima de una superficie alta [25].

Inundación (*flooding*) es el principal método para el cálculo de una cuenca o *watershed* de una imagen [25]. En los algoritmos de inundación se realizan agujeros conceptuales en cada mínimo local y el agua procede a llenar cada cuenca. Si el agua está a punto de desbordar, se coloca una presa en la cresta más cercana para evitar el desbordamiento como se observa en la Figura 9 [25]. Una vez que la superficie se inunda, estas presas son las cuencas o *watershed* que marcan los bordes de las regiones de segmentación [19].

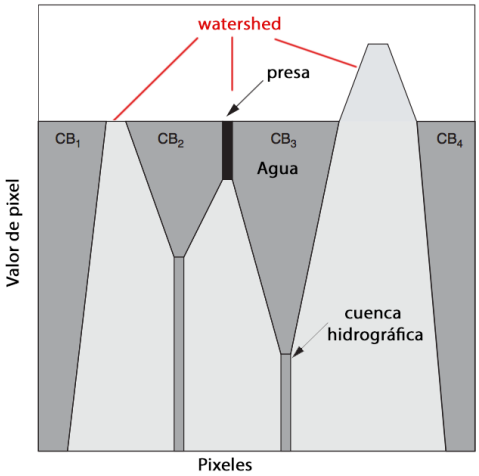


Figura 9: Ilustración de segmentación *watershed* por inundación.

Fuente: Adaptado de Pratt [25].

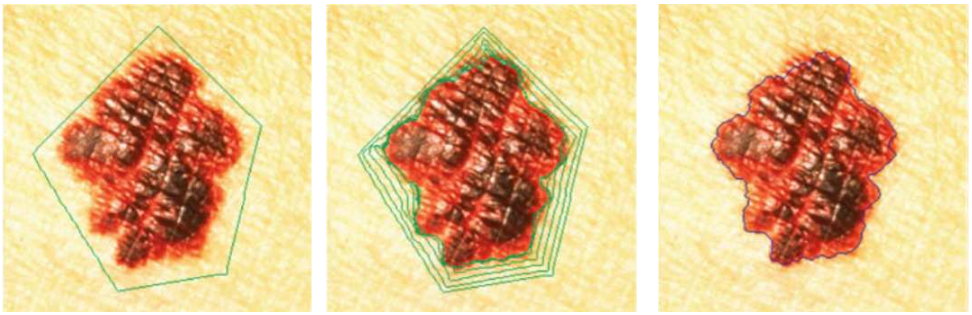
Las técnicas de segmentación basadas en cuencas tienen problemas de sobre segmentación [25]. Esto sucede cuando una imagen es dividida en más regiones de las requeridas por la aplicación.

2.4.5. Segmentación basada en energía

En la segmentación basada en energía se establece una función de energía que alcanzará un valor mínimo cuando la imagen es segmentada según lo esperado. Las más conocidas son *Active Contour*, *Live Wire* y grafos [4].

2.4.5.1. Active Contour

El algoritmo *Active Contour* o Contorno Activo, también conocido como Serpientes (*Snakes* en inglés), ha sido utilizado para segmentación en aplicaciones médicas. En el campo médico, los objetos de interés de una imagen son irregulares y, a menudo, poco distintivos, por lo que la tarea de segmentación se dificulta. En estos casos se utiliza *Active Contour*, donde el contorno está formado por varios puntos que se ciñen gradualmente al objeto de interés como se observa en la Figura 10 [19].



(a) Contorno inicial (b) Contorno intermedio (c) Contorno final

Figura 10: Ejemplo de segmentación de una imagen utilizando *Active Contour*. Fuente: Fernandez-Maloigne [26].

Este contorno se controla a través de una función de energía que se define como la suma de la energía interna y externa de la ecuación 2.7. El contorno envolverá paulatinamente al objeto de interés conforme la energía es minimizada.

$$E_{contorno} = E_{interna} + E_{externa} \tag{2.7}$$

La energía interna depende de propiedades intrínsecas del contorno como su longitud y curvatura. Por otro lado, la energía externa caracteriza las propiedades de la imagen en los puntos donde el contorno se ubica [18].

La principal desventaja de esta técnica, además de que sólo utiliza información de bordes, es su sensibilidad a la curva inicial. Este proceso ocasiona que el usuario no sepa el resultado final cuando inicializa el contorno y si este no es satisfactorio, todo el proceso debe ser repetido desde el principio [27].

2.4.5.2. Live Wire

Live Wire es similar a *Active Contour*. Los dos algoritmos son interactivos y utilizan características de los bordes y funciones de costo para encontrar bordes óptimos. Sin embargo, difieren en ciertos aspectos como [27]:

- *Live Wire* selecciona el límite óptimo de todas las posibles rutas de costo mínimo,

mientras que *Active Contour* calcula el borde óptimo a través del refinamiento de una sola aproximación inicial del borde.

- *Active Contour* se enfoca en un resultado óptimo global para todo el contorno. *Live Wire*, por el contrario, encuentra un balance entre las características globales y locales del borde.

- *Live Wire* utiliza un término de suavizado basado en varios aspectos de la imagen, mientras que *Active Contour* sólo utiliza datos internos basados en la geometría y posición del contorno.

Pese a estas diferencia, *Live Wire* y *Active Contour* no garantizan un resultado óptimo global ya que estos convergen en mínimos locales [4].

2.4.6. Segmentación basada en grafos

La segmentación basada en *Graph Cuts* no solo construye una función de energía utilizando información de bordes y contornos, sino que obtiene resultados óptimos globales [4]. Estos algoritmos de segmentación se basan en el trabajo de Boykov y Jolly [5], quienes propusieron la segmentación de imágenes como un problema de minimización de energía y para su resolución se utiliza la teoría de grafos [5].

En esta sección se explora los fundamentos de grafos para segmentación de imágenes en dos regiones: objeto de interés y fondo.

2.4.6.1. Grafos

Un **grafo** o *graph* es un conjunto de vértices V y arcos E. Este puede ser descrito como un grafo no dirigido $G = \langle E, V \rangle$ en el cual los arcos conectan los vértices.

Existen dos clases de **vértices** o nodos V. La primera clase son nodos vecinos y corresponden a los pixeles de la imagen, y la segunda clase son nodos terminales que, a su vez, pueden ser fuente s (*s-node*) o sumidero t (*t-node*). Estos grafos también son

llamados grafos $s-t$ en donde el objeto de interés de la imagen es representado por los nodos s y el fondo, por los nodos t [4].

Existen también dos tipos de **arcos** E: los arcos *t-links* conectan los pixels vecinos de la imagen con los nodos terminales, mientras que los arcos *n-links* conectan los pixeles vecinos de la imagen a través de un criterio de proximidad. En estos grafos, a cada arco se le asigna un peso no negativo llamado costo y que se denota w_e [4].

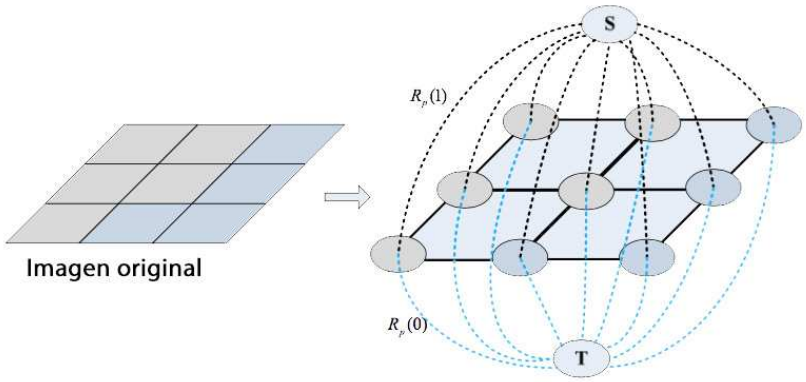


Figura 11: Ilustración de un grafo *s-t*.

Fuente: Adaptado de Yi y Moon [4].

En la Figura 11 se presenta un grafo *s-t*. Los pixeles de la imagen corresponden a los vértices vecinos en el gráfico. Las líneas continuas del grafo son *n-links* y las líneas punteadas son *t-links*.

Un **corte** o *cut* C es un subconjunto de arcos E que se expresa como $C \subset E$. El costo de un corte $|C|$ es la suma de los pesos de los arcos de C, lo cual se expresa en la ecuación 2.8 [4].

$$|C| = \sum_{e \in C} w_e \tag{2.8}$$

Un **corte mínimo** o *minimum cut* es el corte que tiene el mínimo costo y al que se llama *min-cut*. Encontrar este corte *min-cut* es equivalente a calcular el flujo máximo o *maximum flow* (*max-flow*) desde la fuente s al sumidero t. El flujo máximo es la

máxima cantidad de "agua" que se puede enviar desde la fuente al sumidero si se considera que los arcos son tubos con diferentes capacidades, equivalentes a los costos de los arcos [28].

El algoritmo *max-flow/min-cut*, presentado por Boykov y Kolmogorov [29], se utiliza para obtener el corte mínimo del grafo *s-t*. Entonces, los nodos son divididos en dos subconjuntos S y T donde $s \in S$, $t \in T$ y $S \cup T = V$. Estos subconjuntos corresponden al objeto de interés y al fondo de la imagen segmentada [29].

2.4.6.2. Segmentación con Graph Cuts

La segmentación de imágenes se puede considerar como una problema de etiquetado de pixeles: los pixeles del objeto de interés (*s-node*) se etiquetan como 1, mientras que los pixeles del fondo (*t-node*) se etiquetan como 0 [4].

Las etiquetas se definen como $L = \{l_1, l_2, l_3, ..., l_i, ..., l_p\}$ donde p es el número de pixeles de la imagen y $l_i \in \{0, 1\}$. Las restricciones que se imponen a las propiedades de las regiones y bordes de L son descritas por la función de energía expresada en la ecuación 2.9. Esta es la función de energía que será minimizada por el corte mínimo (*min-cut*) en el grafo *s-t* [5].

$$E(L) = \alpha R(L) + B(L) \tag{2.9}$$

Donde $R(L)$ se denomina término regional pues incorpora información de las regiones de la imagen y $B(L)$ es el término límite que introduce información de bordes. El coeficiente α es el factor de importancia relativa entre el término regional y límite. Por ejemplo, si $\alpha = 0$, sólo se considera la información de bordes [4].

El término regional $R(L)$ se define en la ecuación 2.10.

$$R(L) = \sum_{p \in P} R_p(l_p) \tag{2.10}$$

Donde $R_p(l_p)$ es la penalización por asignar la etiqueta l_p al pixel p. El peso de $R_p(l_p)$ se puede obtener a través de la comparación de la intensidad del pixel p con el

histograma de intensidad del objeto de interés y del fondo. Los pesos de los *t-link* se definen según las ecuaciones 2.11 y 2.12.

$$R_p(1) = -lnPr(I_p|'objeto') \tag{2.11}$$
$$R_p(0) = -lnPr(I_p|'fondo') \tag{2.12}$$

Se puede observar en las ecuaciones 2.11 y 2.12 que si $Pr(I_p|'objeto')$ es mayor que $Pr(I_p|'fondo')$, entonces $R_p(1)$ será menor que $R_p(0)$. Es decir, la penalización por agrupar un pixel en un objeto debería ser menor cuando la probabilidad de que el pixel sea un objeto es mayor. Entonces el término regional se minimiza cuando todos los pixeles son asignados correctamente en los dos subconjuntos.

El término límite $B(L)$ se define en la ecuación 2.13.

$$B(L) = \sum_{\{p,q\} \in N} B_{<p,q>} \cdot \delta(l_p, l_q) \tag{2.13}$$

Donde p, q son los pixeles vecinos y $\delta(l_p, l_q)$ es definido en la ecuación 2.14.

$$\delta(l_p, l_q) = \begin{cases} 1 & \text{si } l_p = l_q \\ 0 & \text{si } l_p \neq l_q \end{cases} \tag{2.14}$$

Esto se interpreta como la asignación de las etiquetas l_p, l_q a los pixeles vecinos. Si los pixeles vecinos tienen la misma etiqueta, entonces la penalización es 0; lo cual significa que que el término regional sólo sumará la penalización en el borde de la segmentación. El término $B_{<p,q>}$ de la ecuación 2.13 se define como una función no incremental de $|I_p - I_q|$ que se describe en la ecuación 2.15.

$$B_{<p,q>} \propto exp(-\frac{(I_p - I_q)^2}{2\sigma^2}) \tag{2.15}$$

Donde σ se interpreta como el ruido de la cámara. Si la intensidad de los dos pixeles vecinos es similar, la penalización es alta. Por tanto, es más probable que la energía adquiera su valor más bajo en el borde del objeto de interés [4].

Boykov y Jolly [5] mostraron que la la minimización de energía puede ser calculada por un corte mínimo (*min-cut*) a través de un flujo máximo (*max-flow*). Para esto, la asignación de los pesos en el grafo *s-t* es muy importante y, de acuerdo a la propuesta, la asignación de pesos se muestra en la ecuación 2.16

$$Pesos = \begin{cases} B_{<p,q>} & \{p,q\} \in \text{pixel vecino} \\ \alpha \cdot R_p(0) & \text{para arco } \{p,S\} \\ \alpha \cdot R_p(1) & \text{para arco } \{p,T\} \end{cases} \tag{2.16}$$

Si es más probable que la intensidad del pixel pertenezca al objeto, entonces el peso entre este pixel y el *s-node* será mayor que entre este pixel y el *t-node*. Para los pixeles vecinos, cuando su intensidad es similar, el peso será de valor alto y, por tanto, no serán separados por el corte mínimo. En resumen, el corte mínimo se ubicará cerca del borde del objeto [4].

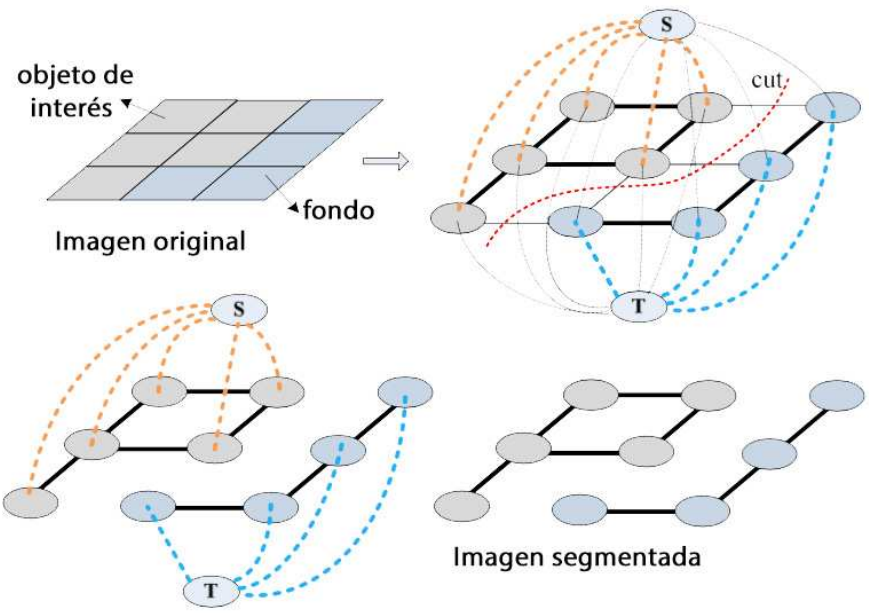


Figura 12: Ilustración del proceso de segmentación con *Graph Cuts*.

Fuente: Adaptado de Yi y Moon [4].

La Figura 12 ilustra el proceso de segmentación previamente descrito. La imagen es segmentada a través de un mínimo corte que corresponde a la mínima energía de la función de la ecuación 2.9. El espesor de las líneas indica el peso de los arcos.

2.4.6.3. Segmentación interactiva con Graph Cuts

La segmentación interactiva utiliza la guía del usuario para mejorar los resultados. Para esto, el usuario señala el objeto de interés y el fondo de una imagen a través de cajas delimitadoras (*bounding boxes*) o *scribbles* (semillas o *seeds* en inglés) [6].

En la tabla 1 se muestran los costos asignados a los pixeles que se han sido marcados como pertenecientes al objeto de interés o al fondo. Para esta asignación se considera que el grafo se denota $G = \langle E, V \rangle$ donde $V = P \cup \{S, T\}$ [4].

Tabla 1:

Pesos de arcos en segmentación interactiva con *Graph Cuts*.

Arco	Peso	Condición
$< p, q >$	$B_{<p,q>}$	$\{p, q\} \in N$
{p, S}	$\alpha \cdot R_p(0)$	$p \in P$ (desconocido)
	K	$p \in$ objeto
	0	$p \in$ fondo
{p, T}	$\alpha \cdot R_p(1)$	$p \in P$ (desconocido)
	0	$p \in$ objeto
	K	$p \in$ fondo

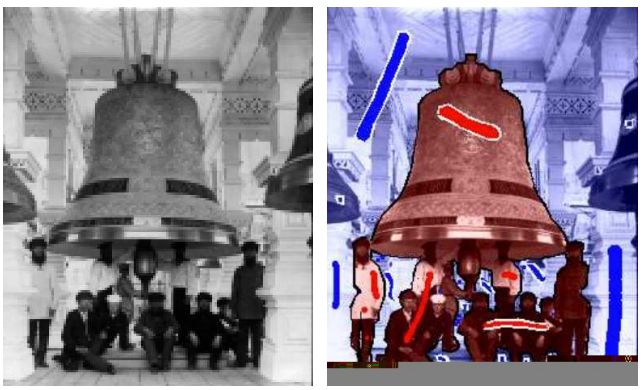
Fuente: Adaptado de Yi y Moon [4].

El valor K de la tabla 1 se define en la ecuación 2.17.

$$K = 1 + \max_{p \in P} \sum_{q:\{p,q\} \in N} B_{<p,q>} \tag{2.17}$$

Si un pixel es señalado como objeto, el peso entre este pixel y el *s-node* será alto, mientras que el peso entre este pixel y el *t-node* será 0. De manera similar, si un pixel es marcado como fondo, el peso entre este pixel y el *t-node* será alto, y el el peso entre este pixel y el *s-node* será 0. Esta estrategia permite que los pixeles marcados por el usuario sean clasificados correctamente en el grafo. Los pesos se actualizan si el usuario marca más pixeles [4] posteriormente.

En la Figura 13 se muestra un ejemplo de segmentación interactiva utilizando *Graph Cuts*. Para la segmentación se ha empleado *scribbles* para marcar pixeles correspondientes al fondo y al objeto de interés.



(a) Imagen original (b) Segmentación con *scribbles*

Figura 13: Ejemplo de segmentación interactiva a través de *Graph Cuts*
Fuente: Solomon y Leckon [20].

2.4.6.4. Solución al problema de min-cut/max-flow

En la sección 2.4.6.2 se explicó la asignación de los costos o pesos a los arcos del grafo *s-t*. En esta sección se analiza con mayor detalle la solución al problema de corte mínimo - flujo máximo. En 2004, Boykov y Kolmogorov [29] presentaron un nuevo algoritmo para encontrar el corte mínimo a través del flujo máximo. Este se

fundamenta en el método Ford-Fulkerson para la búsqueda de rutas en las cuales el flujo es máximo.

En este nuevo algoritmo hay dos árboles de búsqueda que no se superponen: S y T con sus raíces en la fuente y el sumidero, respectivamente. Los nodos que no pertenecen a S o T son nodos libres [29].

Los nodos en los árboles S o T pueden ser activos o pasivo. Los nodos activos están en el borde de cada árbol de tal manera que estos permiten el crecimiento del árbol, ya que estos nodos adquieren nuevos hijos de un conjunto de nodos libres. Los nodos pasivos están en la parte interior del árbol y no permiten el crecimiento del árbol.

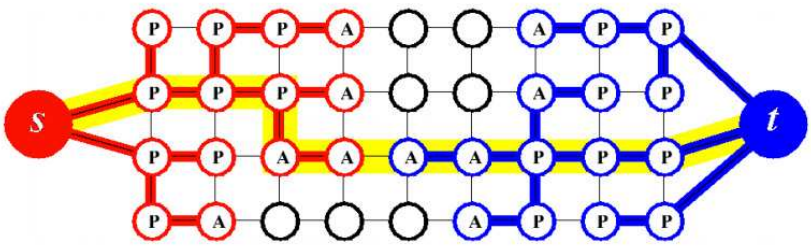


Figura 14: Ilustración de proceso de algoritmo de Boykov y Kolmogorov [29].

Fuente: Boykov y Kolmogorov [29].

Este algoritmo es ilustrado en la Figura 14. Los nodos rojos corresponden al árbol S y los nodos azules, al árbol T. Los nodos activos y pasivos se han etiquetado con las letras A y P, mientras que los nodos de color negro corresponden a los nodos libres. Este algoritmo tiene tres etapas:

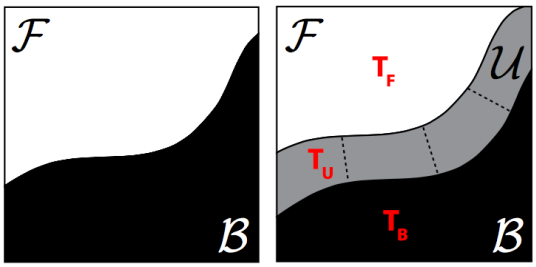
1. **Etapa de crecimiento:** Los nodos activos exploran nodos adyacentes no saturados y adquieren nuevos hijos de un conjunto de nodos libres. Estos nodos adquiridos se convierten en nodos activos. Los nodos se convierten en pasivos cuando se exploran todos sus nodos adyacentes. Esta etapa finaliza cuando un nodo activo de uno de los árboles detecta un nodo vecino que pertenece al otro árbol y, por tanto, cuando se ha formado una ruta entre fuente y sumidero.

2. **Etapa de aumento:** Se selecciona el camino con el máximo flujo entre la fuente y el sumidero. Para esto se saturan los arcos cuyos pesos son iguales al flujo máximo. Los nodos desconectados del *s-node* o *t-node* son llamados nodos huérfanos, los cuales pasan a ser las raíces de otros árboles de búsqueda.

3. **Etapa de adopción:** En esta etapa se intenta formar sólo un árbol compuesto por un subconjunto S y T. Por tanto, se busca un nuevo padre para cada nodo huérfano. Si ningún nodo califica como padre, entonces el nodo huérfano se convierte en un nodo libre y sus hijos son declarados huérfanos. Esta etapa termina cuando no quedan nodos huérfanos.

Después de la etapa de adopción, el algoritmo regresa a la etapa de crecimiento. El proceso concluye cuando los árboles de búsqueda de la fuente y el sumidero están separados por arcos saturados y no pueden crecer [29].

2.4.6.5. GrabCut



(a) Segmentación (b) *Trimap* creado

Figura 15: Ilustración del proceso de segmentación con *GrabCut*.
Fuente: Adaptación de Radke [30].

El objetivo de *GrabCut* es calcular una segmentación de dos regiones a través de *Graph Cut* como se muestra en la Figura 2.15(a) y, posteriormente, dilatar el borde de esta segmentación para crear un *trimap*. El *trimap* de la Figura 2.15(b) está comprendido por tres subregiones: pixeles de fondo (T_B), pixeles del objeto de interés (T_F) y

pixeles no etiquetados (T_U). Esta consideración permite que la segmentación suavice la transición entre el fondo y el objeto de interés, mejorando así el borde [30].

Para empezar, Rother *et al.* [31] definen la segmentación con *Graph Cuts* en función de las características de color de la imagen. La imagen se expresa como un arreglo de valores grises $z = (z_1, ..., z_n, ..., z_N)$ y la segmentación de la imagen se expresa como el arreglo $\alpha = (\alpha_1, ..., \alpha_n, ..., \alpha_N)$, en donde cada pixel tiene un valor de 0 en caso de pertenecer al fondo y 1 si pertenece al objeto de interés. El parámetro θ representa los dos histogramas de intensidad del fondo y del objeto de interés como se expresa en la ecuación 2.18. Estos histogramas normalizados utilizan los pixeles etiquetados de las regiones T_B y T_F [31].

$$\theta = \{h(z : a), \alpha = 0, 1\} \tag{2.18}$$

La segmentación se convierte en la tarea de dilucidar la opacidad desconocida de las variables α utilizando la información del modelo θ. La función de energía de la ecuación 2.9 se expresa entonces en la ecuación 2.19.

$$E(\alpha, \theta, z) = U(\alpha, \theta, z) + V(\alpha, z) \tag{2.19}$$

El término de datos U evalúa si la distribución de opacidad α pertenece al histograma del modelo θ. El término de suavidad V se enfoca en la uniformidad de las regiones con tonos de grises similares. De igual manera que el algoritmo de *Graph Cuts*, *GrabCut* resuelve este problema a través de un corte mínimo en el grafo *s-t*.

GrabCut es una versión mejorada del algoritmo de *Graph Cuts*. Estas mejoras son indicadas a continuación:

- **Modelamiento de color:** A diferencia del algoritmo con *Graph Cuts*, *GrabCut* reemplaza el modelado de la imagen en escala de grises y utiliza un modelado en color RGB. Para esto, utiliza un modelo gaussiano llamado *Gaussian Mixture Model* (GMM) para el fondo y para el objeto de interés. Lo cual introduce un

vector $k = \{k_1, ..., k_n, ..., k_N\}$ que asigna a cada pixel un componente GMM perteneciente al fondo o al objeto de interés [31]. La función de energía se expresa ahora en la ecuación 2.20.

$$E(\alpha, k, \theta, z) = U(\alpha, k, \theta, z) + V(\alpha, z) \tag{2.20}$$

El término $U(\alpha, k, \theta, z)$ ahora toma en cuenta el modelo GMM para la definición de pertenencia de un pixel al fondo o al objeto de interés. U se expresa en la ecuación 2.21.

$$U(\alpha, k, \theta, z) = \sum_n D(\alpha_n, k_n, \theta, z_n) \tag{2.21}$$

Donde $D(\alpha_n, k_n, \theta, z_n) = -\log p(z_n|\alpha_n, k_n, \theta) - \log \pi(\alpha_n, k_n)$. La distribución $p(\cdot)$ es una distribución de probabilidad gaussiana y $\pi(\cdot)$ son coeficientes de peso mixtos. Por otro lado, al igual que en el algoritmo con *Graph Cuts*, el término $V(\alpha, z)$ se enfoca en la similitud entre pixeles vecinos que pertenecen a una misma región, pero utiliza la distancia euclidiana en un espacio RGB [31].

- **Segmentación por proceso iterativo de minimización de energía:** La minimización de energía en GrabCut es iterativa. Esto permite refinar la opacidad de θ conforme nuevos pixeles de la region inicial T_U son etiquetados. Entonces los parámetros de color de GMM cambian y, por tanto, se puede mejorar el modelado de color del fondo y del objeto de interés. El proceso se muestra a continuación:

  1. Se asigna componentes GMM a los pixeles sin etiqueta. Para cada n en T_U como se expresa en la ecuación 2.22.

$$k_n := \arg\min_{k_n} D(\alpha_n, k_n, \theta, z_n) \tag{2.22}$$

  2. Se estiman nuevos parámetros GMM a partir de la intensidad de los pixeles de la imagen, incluyendo los pixeles recientemente etiquetados, como se

expresa en la ecuación 2.23. Para cada componente GMM de k, se calcula la media $\mu(\alpha, k)$ y la covarianza $\Sigma(\alpha, k)$ utilizando el subconjunto de pixeles correspondientes al fondo o al objeto de interés [31].

$$\theta := \arg\min_{\theta} U(\alpha, k, \theta, z) \tag{2.23}$$

3. Se construye el grafo *s-t* y se encuentra el corte mínimo a través del algoritmo de flujo máximo descrito en la sección 2.4.6.4. Este proceso se expresa en la ecuación 2.24 como la minimización de la función de energía.

$$\min_{\{\alpha_n : n \in T_U\}} E(\alpha, k, \theta, z) \tag{2.24}$$

- **Suavizado del borde:** Una banda de ancho fijo alrededor del borde T_U es definida como se muestra en la Figura 2.15(b). En esta región, los valores de α pueden ser fraccionarios. Por tanto, en esta banda no se define claramente el límite ente el fondo y el objeto de interés porque α toma valores entre 0 y 1. Esto produce una imagen con bordes suavizados [30].

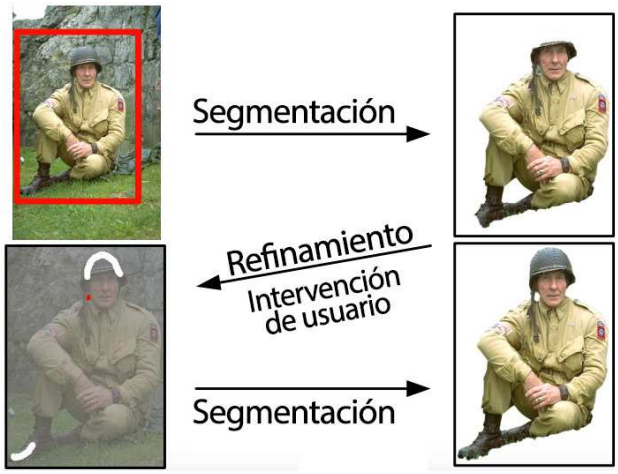


Figura 16: Ejemplo de segmentación interactiva con *GrabCut*.

Fuente: Adaptado de Rother *et al.* [31].

La propuesta de Rother *et al.* [31] incluye el uso de cajas delimitadoras o *bounding boxes* para marcar inicialmente el objeto de interés y *scribbles* para refinar los resultados como se muestra en la Figura 16. Para esta inicialización, *GrabCut* utiliza *trimaps* incompletos que permiten el uso de etiquetas provisionales.

El algoritmo utiliza etiquetas provisionales T_F para todos los pixeles dentro de la caja delimitadora, con los cuales se calcula el modelo GMM del objeto de interés. No obstante, para el modelado GMM del fondo, se asignan etiquetas fijas T_B a todos los pixeles que están afuera de la caja delimitadora. Los pixeles pueden cambiar de etiqueta provisional después de varias iteraciones del algoritmo o si el usuario añade *scribbles* para refinar el resultado.

2.4.6.6. OneCut

OneCut propone un nuevo término de energía que mide la distancia $L1$ entre los modelos de apariencia del objeto y del fondo. Tang *et al.* [32] demuestran que este nuevo término permite que la función de energía sea minimizada con un solo corte, lo cual reemplaza el proceso iterativo propuesto en *GrabCut*.

Para esto, Tang *et al.* [32] redefinen la función de energía de la ecuación 2.9 y la expresan en 2.25.

$$E(S) = |S| \cdot H(\theta^S) + |\overline{S}| \cdot H(\theta^{\overline{S}}) + |\partial S| \tag{2.25}$$

Donde S es un segmento de la imagen $S \subset z$, θ^S y $\theta^{\overline{S}}$ son los histogramas correspondientes al objeto S y al fondo $\overline{S}$. $H(\cdot)$ es la entropía para distribuciones de probabilidad.

Esta forma de energía se obtiene al reemplazar la suma de pixeles por la suma de *bins* de colores. Se observa también que el mínimo global de la función de energía no depende de los modelos de color proporcionados por el usuario. Por tanto, la iteración del algoritmo *GrabCut* se debe a que su solución es un mínimo local de la ecuación 2.25 [32].

Para expresar el término $L1$ en la ecuación 2.26 se considera que n_k es el número de pixeles de la imagen que pertenecen al *bin* k. Por lo que n_k^S y $n_k^{\overline{S}}$ corresponden al número de pixeles en el bin k del objeto de interés y fondo, respectivamente. El término de superposición propuesto penaliza la intersección entre los bins del objeto y el fondo a través de la incorporación del término L_1 en la función de energía [32].

$$E_{L_1}(\theta^S, \theta^{\overline{S}}) = -||\theta^S - \theta^{\overline{S}}||_{L_1} \tag{2.26}$$

En la ecuación 2.27 se expresa la función de energía con el término L_1 para segmentación interactiva a través de *scribbles*.

$$E(S) = -\beta||\theta^S - \theta^{\overline{S}}||_{L_1} + \lambda|\partial S| \tag{2.27}$$

2.4.6.7. Random Walks

Estes algoritmos, también llamados Caminante Aleatorio, utilizan *scribbles* para la segmentación interactiva. *Random Walks* formula el problema de segmentación a través del grafo *s-t* definido en la sección 2.4.6.1. Sin embargo, el método de resolución difiere de los algoritmos basados en *Graph Cuts*.

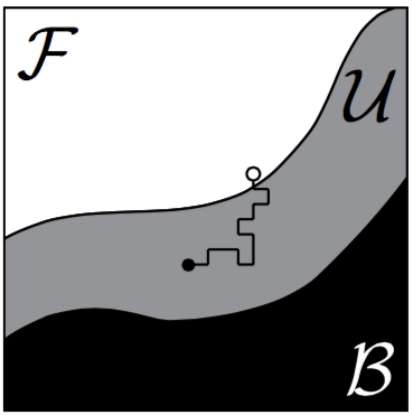


Figura 17: Ilustración del proceso de segmentación con *Random Walks*.

Fuente: Radke [30].

En *Random Walks*, cada arco $e_{pq} \in E$ tiene un peso no negativo w_{pq}. La formulación de w_{pq} varía según el algoritmo, pero en general, el valor w_{pq} es alto para pixeles similares y casi cero para pixeles diferentes. El algoritmo estima la etiqueta l_p en el pixel p de la región desconocida U como la probabilidad de que un *random walker* inicie en p y que, eligiendo arcos de acuerdo al peso w_{pq}, encuentre antes un pixel del objeto F que del fondo B como se muestra en la Figura 17 [30].

Grady [33] utiliza la expresión de la ecuación 2.28 para asignar pesos a los arcos del grafo, donde g_p es la intensidad de la imagen en el pixel p y β es un parámetro libre.

$$w_{pq} = exp(-\beta(g_p - g_q)^2) \tag{2.28}$$

Se debe notar que los algoritmos *Random Walks* no evalúan las ruta más corta o más probable, sino las probabilidades de la ruta que tome un *random walker*. Para esto, se demostró que la minimización del problema de Dirichlet de la ecuación 2.29 soluciona el problema de las probabilidades de *Random Walker* [4].

$$D[x] = \frac{1}{2} \sum_{e_{p,q} \in E} w_{p,q}(x_p - x_q)^2 \tag{2.29}$$

Minimizar $D[x]$ equivale a resolver la función armónica que satisface la condición de bordes. Por tal motivo, *Random Walker* es sensible a la posición y cantidad de los *scribbles* que funcionan como semilla para este algoritmo [4].

2.4.7. Otros métodos

2.4.7.1. SIOX: Simple Interactive Object Extraction in Still Images

SIOX es el algoritmo de segmentación interactiva que ha sido implementado en GIMP. Esta técnica está conformada por cuatro etapas:

1. **Entrada:** El usuario especifica tres regiones de la imagen a través de un *trimap*: fondo conocido, objeto conocido y región desconocida. Los valores del *trimap*

se asignan a una matriz de confianza, donde cada elemento corresponde a un pixel de la imagen. Estos valores están dentro de un rango $[0, 1]$ según la región del pixel: 0 para pixeles del fondo conocido, 1 para pixeles del objeto conocido y 0.5 para la región desconocida [34].

2. **Conversión a espacio CIELAB:** En la sección 2.3.2.2 se señaló que este espacio de color uniforme es útil para calcular la similitud entre colores a través de la distancia euclidiana. Por esta razón, SIOX convierte la imagen al espacio de color CIELAB utilizando el estándar D65 [34].

3. **Segmentación por color:** SIOX crea una modelo de color del fondo conocido y lo utiliza como referencia para clasificar los pixeles de la imagen. La muestra del fondo conocido suministrado por el usuario es agrupado en *clusters* del mismo tamaño. Esto es porque especificar el tamaño de un *cluster* es especificar una cierta precisión [34].

4. **Refinamiento:** En la última etapa de SIOX se eliminan los colores del fondo que aparecen en el objeto de interés. Se realiza también una búsqueda en anchura en la matriz de confianza para identificar las regiones espacialmente conectadas que han sido identificadas como objeto. Entonces se asume que las regiones más amplias pertenecen al objeto de interés y se eliminan las regiones más pequeñas. Un factor de suavizado es definido para este proceso: un factor alto es adecuado para reducir errores de clasificación, mientras que un factor bajo es adecuado para objetos que precisan una segmentación más precisa como, por ejemplo, cabello o nubes [34].

   Los valores finales de la matriz de confianza son utilizados como factores de transparencia para los pixeles. Si el pixel pertenece al fondo, entonces el factor de transparencia es 0 y, por tanto, el pixel no es visible; pero si el pixel pertenece al objeto, el objeto es totalmente visible porque el factor de transparencia es 1.

2.5. Restauración de imágenes

Figura 18: Detalle de restauración de La Purificación de la Virgen de Pedro de Campaña.

Fuente: Museo Nacional del Prado [35].

Bertalmio *et al.* [1] acuñaron la expresión *image inpainting* para describir el proceso de restauración digital de imágenes. Esta técnica, también conocida como *image completion* en inglés, se inspira en la restauración de arte, donde se retocan y reparan áreas deterioradas de obras de arte con la intención de conservar la apariencia original de las obras [1], tal y como como se muestra en la Figura 18.

Los algoritmos de restauración digital de imágenes reconstruyen la información perdida de las regiones deterioradas utilizando la información espacial adyacente a estas regiones. Por este motivo, estos algoritmos se utilizan en diversas aplicaciones, como restauración de películas y corrección de ojos rojos en fotografías [2].

En esta sección se revisan dos clases de métodos característicos de *inpainting* según su enfoque: métodos basados en ecuaciones diferenciales y métodos basados en parches. La primera clase, correspondiente a métodos basados en ecuaciones diferenciales parciales, se utilizan para áreas deterioradas de tamaño reducido. Por otro lado, los métodos basados en parches se utilizan para rellenar regiones más amplias [2].

2.5.1. Métodos basados en ecuaciones diferenciales parciales

Esta clase de métodos parten de la primera propuesta de Bertalmio *et al.* [1] para *inpainting*. Por tanto, aquí se describe el método introducido en este trabajo [1].

Si la región Ω es el área deteriorada, entonces los métodos basados en ecuaciones diferenciales rellenan iterativamente la región Ω desde el borde $\partial\Omega$ hacia el interior. Estos métodos inicializan el proceso con las condiciones de la ecuación 2.30 [30].

$$I_0(x,y) = \begin{cases} I(x,y) & (x,y) \notin \Omega \\ negro & (x,y) \in \Omega \end{cases} \tag{2.30}$$

En cada paso del proceso de *inpainting* se crea una imagen $I_{n+1}(x,y)$ basada en $I_n(x,y)$ hasta que el área deteriorada es rellenada. En este caso, los algoritmos propagan los colores y estructura de los pixeles adyacentes al área Ω.

Para esto, se propagan los bordes acentuados a lo largo de las direcciones isotópicas para producir un resultado más natural. Estas direcciones corresponden a la orientación donde bordes y contornos tienen un cambio mínimo de intensidad [30].

Considerando que la gradiente en un pixel indica la dirección de cambio máximo, entonces la dirección isotópica se puede calcular como el vector unitario a la gradiente, tal y como se expresa en la ecuación 2.31 [30].

$$\nabla_{\perp} I(x,y) = unitario \left(\begin{bmatrix} I(x,y) - I(x,y+1) \\ I(x+1,y) - I(x,y) \end{bmatrix} \right) \tag{2.31}$$

La intensidad de los pixeles que rellenan el área deteriorada debe satisfacer la ecuación diferencial parcial de la ecuación 2.32.

$$\nabla(\nabla^2 I) \cdot \nabla^{\perp} I = 0 \tag{2.32}$$

Acorde a esta expresión, el cambio en el Laplaciano $\nabla(\nabla^2 I)$ debe ser cero en la dirección del isótopo $\nabla_{\perp} I$. Idealmente, la ecuación 2.32 se puede resolver utilizando

una imagen que cambia en función del tiempo, tal y como se expresa en la ecuación 2.33 [30].

$$\frac{\partial I}{\partial t} = \nabla(\nabla^2 I) \cdot \nabla^{\perp} I \tag{2.33}$$

Entonces, la ecuación 2.32 se satisface cuando $\frac{\partial I}{\partial t} = 0$. Esto sucede cuando la imagen ya no cambia. Sin embargo, en la práctica, la ecuación 2.33 se aproxima con la expresión de la ecuación 2.34, donde t denota los pasos discretos del tiempo [1]. Para esto, se utilizan también las aproximaciones usuales de la gradiente y el Laplaciano.

$$I_{n+1}(x,y) = I_n(x,y) + (\nabla t) U_n(x,y), \forall (x,y) \in \Omega \tag{2.34}$$

Este proceso se alterna con otro llamado proceso de difusión anisotrópica [1] para reducir el ruido y lograr una actualización correcta de la dirección de la difusión durante el proceso de relleno del área deteriorada; de manera que no se pierda la definición de los bordes [30]. Por ejemplo, Bertalmio *et al.* [1] proponen un procedimiento donde se aplican 15 pasos de *inpainting* y 2 pasos de difusión a una velocidad $\triangle t = 0{,}1$ [1]. También es importante considerar que el proceso de difusión anisotrópica se aplica a toda la imagen al principio para reducir el error del cálculo de la dirección isotópica.

(a) Imagen original (b) Imagen restaurada

Figura 19: Ejemplo de restauración basada en ecuaciones diferenciales parciales. Fuente: Bertalmio *et al.* [1].

Este método es especialmente adecuado para casos en los que se retocan áreas pequeñas de una imagen, como se muestra en la Figura 19. Sin embargo, como aquí se

explicó, el algoritmo no puede reproducir información de texturas porque el interior de la zona restaurada es más suave y borroso que las regiones adyacentes como se ilustra en la Figura 20.

(a) Imagen original (b) Imagen restaurada

Figura 20: Ejemplo de restauración fallida con algoritmo de Bertalmio.
Fuente: Bertalmio *et al.* [1].

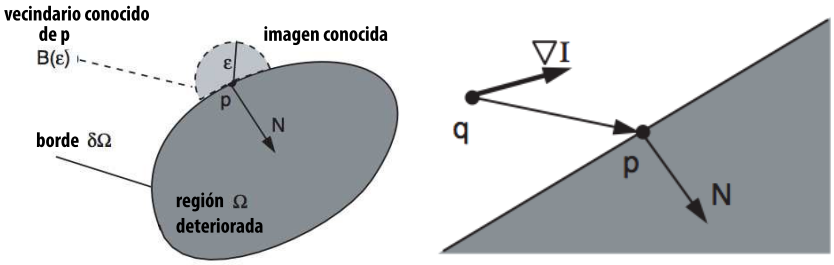
2.5.1.1. Navier-Stokes para *inpainting*

En un trabajo posterior, Bertalmio *et al.* [36] utilizaron la teoría de dinámica de fluidos y la ecuación de Navier-Stokes para reformular el problema de *inpainting*. Para esto, se estableció que la intensidad de imagen es equivalente al a la corriente del fluido donde las líneas de nivel de la imagen definen la corriente de líneas del flujo. La dirección isotópioca, por otro lado, es equivalente a la velocidad del fluido y la suavidad de la imagen puede ser considerada equivalente a la ecuación de transporte de vorticidad.

Entonces, utilizando este enfoque, se puede emplear la ecuación de vorticidad —en lugar de la expresión de la ecuación 2.34— para resolver el problema de restauración de imágenes. En consecuencia, esto amplía las opciones disponibles para la resolución del problema porque se puede utilizar la teoría de dinámica de fluidos que ha sido estudiada extensivamente [37].

2.5.1.2. Fast Marching Method

Telea [38] propuso un métodos de restauración basado en el trabajo de Bertalmio *et al.* [1], donde la suavidad de la imagen se estima como el promedio de los pixeles vecinos del área a rellenar. Para esto, la región deteriorada se trata como ajustes de nivel y se utiliza el método FMM (*Fast Marching Method*) para propagar la información [38]. Este método tiene la ventaja de ser más rápido porque no utiliza un proceso iterativo: el valor de cada pixel del área deteriorada sólo se modifica una vez.



(a) Definiciones de componentes (b) Representación de ecuación 2.35.

Figura 21: Proceso de restauración con Telea.
Fuente: Adaptado de Telea [38].

El pixel perteneciente al área deteriorada que será rellenado se denota p. El área adyacente que contiene los pixeles conocidos alrededor de p se llama *vecindario de p* y se denota B_ε, donde ε es el radio de $B(\varepsilon)$ como se muestra en la Figura 2.21(a). El valor ε es pequeño y debe ser configurado por el usuario. Posteriormente, el algoritmo considera una aproximación de la imagen $I_q(p)$ en el punto p según la expresión de la ecuación 2.35 [37].

$$I_q(p) = I(q) + \nabla I(q)(p-q) \tag{2.35}$$

Donde q es un pixel conocido del vecindario de p y $\nabla I(q)$ es la gradiente. Esta ecuación es representada en la Figura 2.21(b). Después, el siguiente paso es rellenar el pixel. Entonces el valor del pixel p corresponde a la suma de los estimados $I_q(p)$ en

B_ε, que se pondera en función de $w(p,q)$. Entonces el valor del pixel a ser rellenado $I(p)$ se calcula según la ecuación 2.36 [37].

$$Ip = \frac{\sum_{q \in B_\varepsilon(p)} w(p,q)[I(q) + \nabla I(q)(p-q)]}{B_\varepsilon(p) w(p,q)} \tag{2.36}$$

La función de ponderación $w(p,q)$ fue propuesta de tal manera que el relleno de p propague de igual manera los valores de gris y los detalles nítidos de la imagen en $B_\varepsilon(p)$. Esta función se define en la ecuación 2.37 [38].

$$w(p,q) = dir(p,q) \cdot dist(p,q) \cdot lev(p,q) \tag{2.37}$$

En las ecuaciones de la expresión 2.38 se definen dir, dst y lev:

$$\begin{aligned} dir(p,q) &= \frac{(p-q)}{\|p-q\|} \cdot N(p) \\ dst(p,q) &= \frac{d_0^2}{\|p-q\|^2} \\ lev(p,q) &= \frac{T_0}{1+|T(p)-T(q)|} \end{aligned} \tag{2.38}$$

En donde $dir(o,q)$ denota la componente direccional diseñada para garantizar una contribución mayor de los pixeles situados a la dirección normal N de la gradiente. Por otro lado, $dst(p,q)$ es la componente de distancia geométrica que asegura una contribución mayor para los pixeles que están espacialmente cerca del pixel p. Finalmente, $lev(p,q)$ es el componente de ajustes de nivel que asegura que los pixeles más cercanos a la línea isotópica, que pasa a través del pixel p, tengan una mayor contribución que el resto de pixeles [37].

Por consiguiente, los pixeles de la region deteriorada Ω son rellenados utilizado la ecuación 2.36. Como la técnicas de Bertalmio *et al.* [1], este proceso de relleno inicia desde el borde $\partial\Omega$, de tal manera que los pixeles del área Ω más cercanos a la región conocida son rellenados primero. Para este proceso, sin embargo, Telea [38] utiliza el método FMM (*First Marching Method*) que garantiza el orden del relleno en base a la distancia hacia el borde Ω. Este método numérico es usado para resolver la expresión de la ecuación 2.39 [37].

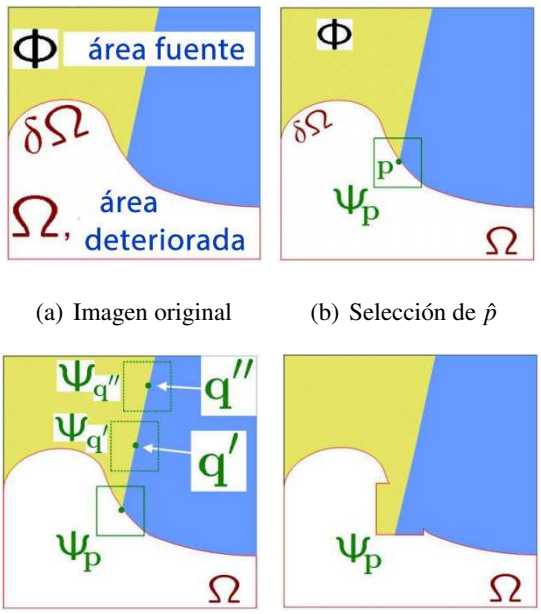
$$|\nabla T| = 1 \quad \text{en} \quad \Omega, \qquad \text{con} \quad T = 0 \quad \text{en} \quad \partial\Omega \tag{2.39}$$

Donde T es la distancia de los pixeles de Ω al borde $\partial\Omega$.

Sin embargo, pese a que Telea [38] propone un algoritmo más eficiente, tiene la misma limitación que el algoritmo de Bertalmio *et al.* [1]: su incapacidad de replicar la información de texturas de la imagen.

2.5.2. Métodos basados en parches

Criminisi *et al.* [3] propusieron una técnica basada en parches para restauración de imágenes que, a diferencia de los métodos basados en ecuaciones diferenciales parciales, replica texturas. Para esto, se copian los pixeles de una región fuente $\Phi = I - \Omega$ en la región deteriorada Ω.



(a) Imagen original (b) Selección de $\hat{p}$

(c) Búsqueda de parche $\Psi_{\hat{q} \subset Phi}$ (d) Reemplazo de pixeles.

Figura 22: Proceso de restauración basada en parches.
Fuente: Criminisi *et al.* [3].

En la Figura 22 se muestra el proceso iterativo que estos métodos siguen para restaurar el área deteriorada. Este proceso se resume a continuación [30]:

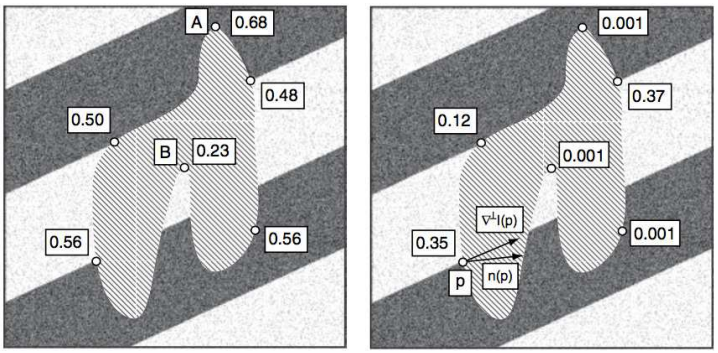
1. Determinar la prioridad $P(p)$ para cada pixel $p \in \partial\Omega$.

2. Seleccionar el pixel $\hat{p}$ de mayor prioridad y señalar el parche $\Psi_{\hat{p}}$ que se ubican alrededor de este pixel. La dimensión del parche se denota $W \times W$. Esta selección se muestra en la Figura 2.22(b).

3. Encontrar el parche $\Psi_{\hat{q} \subset Phi}$ que mejor coincida con $\Psi_{\hat{p}}$. Esto se muestra en la Figura 2.22(c).

4. Reemplazar los pixeles de $\Psi_{\hat{p} \cap Omega}$ con los pixeles de $\Psi_{\hat{q}}$. Este paso reduce la región deteriorada Ω como se muestra en la Figura 2.22(d).

La prioridad $P(p)$ es el producto del término de confianza $C(p)$ y el término de información $D(p)$. El término de confianza de un pixel es alto cuando los pixeles que le rodean tienen intensidades conocidas. Por ejemplo, el pixel A de la Figura 2.23(a) es un buen candidato porque la mayoría de los pixeles que lo rodean ya tienen una intensidad conocida, mientras que el pixel B de la Figura 2.23(b) es un mal candidato porque la mayoría de los pixeles que lo rodean pertenecen a la zona deteriorada [30].



(a) Término de confianza (b) Término de información

Figura 23: Término de confianza e información en Criminisi.
Fuente: Bertalmio *et al.* [36].

El término de confianza se inicializa según la expresión de la ecuación 2.40.

$$C(p) = \begin{cases} 1 & \text{si } p \in \Omega \\ 0 & \text{si } p \notin \Omega \end{cases} \tag{2.40}$$

Después, $C(p)$ se calcula como el promedio de los pixeles en Ψ_p como se expresa en la ecuación 2.41.

$$C(p) = \frac{1}{w^2} \sum_{q \in \Omega_p} C(q) \tag{2.41}$$

El propósito del término de información es similar al propuesto en el método basado en ecuaciones diferenciales parciales. Este término busca propagar las intensidades en la región deteriorada a lo largo de las direcciones isotópicas. Este término se calcula según la expresión de la ecuación 2.42 [30].

$$D(p) = \|\nabla I(p)\| |\nabla^{\perp} I(p) \cdot n(p)| \tag{2.42}$$

Donde $n(p)$ es un vector unitario ortogonal a $\partial\Omega$ en p y $\nabla^{\perp} I(p)$ es el vector unitario perpendicular a la gradiente definida en la ecuación 2.31. Por consiguiente, el término de información en un pixel aumenta acorde a la gradiente de la imagen. En la Figura 2.23(b) se muestra que el término es alto cuando un borde intenso es perpendicular a $\partial\Omega$ y es menor en regiones de igual intensidad.

Después de la selección del pixel $\hat{p}$ de mayor prioridad, se crea un parche alrededor a este pixel. Entonces se busca un parche similar en la región fuente a través de la distancia euclidiana. Posteriormente, los pixeles son copiados de la región fuente en la región deteriorada $\Psi_{\hat{p}} \cap \Omega$ [30].

Finalmente, los valores de $C(\hat{p})$ son asignados al término de confianza de los pixeles copiados. Esto permite que el término disminuya de valor conforme se va rellenando la imagen.

(a) Imagen original (b) Imagen removida

Figura 24: Ejemplo de objetos removidos con Criminisi.
Fuente: Criminisi *et al.* [3].

En la Figura 24 se observa que el algoritmo basado en parches, de Criminisi *et al.* [3], puede replicar las texturas de la imagen. Evidentemente, los métodos basados en parches son adecuado para remover objetos de un imagen.

CAPITULO 3

DISEÑO DEL EXPERIMENTO

3.1. Extracción de objetos

Este trabajo aborda la extracción interactiva de objetos. Para esto, esta tarea elimina el objeto a través del proceso de segmentación y rellena la zona utilizando el proceso de restauración de imágenes. Entonces, el diseño del experimento presentado en este capítulo se enfoca independientemente en estas dos etapas.

La primera etapa, segmentación, busca dividir la imagen en dos regiones. Para este cometido, la guía del usuario es primordial para identificar los objetos que serán eliminados de las imágenes. En consecuencia, el experimento evalúa algoritmos de segmentación interactiva considerando la intervención del usuario.

La segunda etapa, correspondiente a la restauración de la imagen, rellena la zona de la imagen de donde se ha extraído el objeto. Los algoritmos utilizan información de las zonas adyacentes para reconstruir estas imágenes. Esta característica, sin embargo, plantea un inconveniente en la evaluación porque no se cuenta con una referencia. Por consiguiente, el experimento propuesto involucra una evaluación cualitativa de los resultados.

Pese a que esta dos etapas trabajan independientemente, su mutua dependencia es evidente para extraer objetos de imágenes. Tanto es así que la restauración no funciona si la segmentación falla.

3.2. Segmentación de imágenes

3.2.1. Objetivo de la fase experimental

La fase experimental busca identificar el algoritmo de segmentación más adecuado para la tarea de extracción de objetos. Con esta finalidad, el diseño del experimento parte de la formulación los siguientes planteamientos:

- ¿Cuál es el algoritmo más eficiente?
- ¿Cuál es el algoritmo más eficaz?
- ¿Cuál es el algoritmo que requiere menos esfuerzo por parte del usuario?

En este trabajo se pretende responder a estos planteamientos de manera cuantitativa con datos experimentales que puedan ser utilizados como referencia para futuros trabajos de investigación. Por tal motivo, este trabajo emplea tres criterios de evaluación utilizados en trabajos relacionados [4, 39, 40]: eficiencia, exactitud y esfuerzo.

En el diagrama de la Figura 25 se muestra el proceso del experimento propuesto para la evaluación en base a estos criterios.

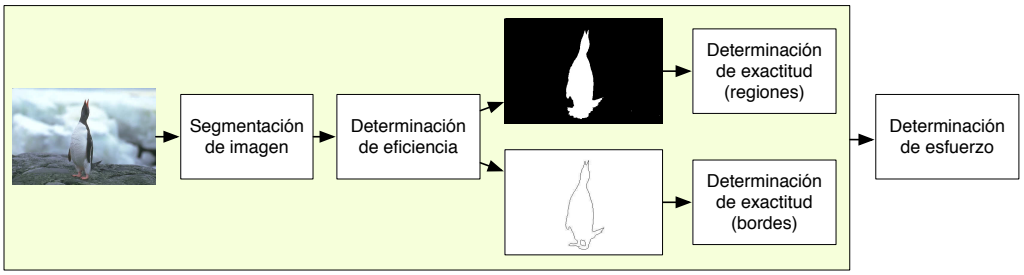


Figura 25: Diagrama del proceso de evaluación de algoritmos de segmentación.

En general, el propósito es diseñar un experimento que cumpla con los principios de repetibilidad y reproducibilidad de los resultados. En consecuencia, para esta evaluación se han seleccionado algoritmos que no sólo son relevantes en el estado del arte,

sino que cuentan con implementaciones públicas. De igual manera, en este trabajo se ha recopilado una base de datos para evaluar algoritmos de segmentación de imágenes en extracción interactiva de objetos.

En este experimento, el primer paso es la determinación de la eficiencia de los algoritmo de segmentación. Para esto, se debe medir el tiempo que los algoritmos tardan en segmentar cada imagen de la base de datos. Posteriormente, los resultados de estos algoritmos son utilizados para determinar la exactitud de la segmentación. Esta medición tiene dos enfoques:

- **Regiones:** Las máscaras, que definen las regiones pertenecientes al objeto y fondo, se comparan con una segmentación de referencia. La similitud se mide utilizando el coefficiente Jaccard.

- **Bordes:** La máscaras se convierten en mapas de bordes y estas, a su vez, se comparan con los bordes de la referencia. Entonces se evalúan los valores de precisión y exhaustividad para graficar las curvas.

Se considera también que el desempeño de los algoritmos de esta segmentación interactiva depende de las estrategias computacionales y de la guía del usuario. Por esta razón, el tercer criterio evalúa el esfuerzo requerido por el usuario para conseguir una mejor segmentación.

3.2.2. Selección de los algoritmos

En la sección2.4.5 se identificaron y describieron los principales algoritmos de segmentación basados en energía. De estos métodos, los algoritmos basados en *Graph Cuts* han sido ampliamente explorados en los últimos años por su alto desempeño y eficiencia computacional, y porque permiten una formulación más flexible del problema de segmentación [6]. Por esta razón, la propuesta de este trabajo es la evaluación de algoritmos basados en *Graph Cuts*.

Esta propuesta también señala que estos algoritmos serán utilizados para extraer objetos de imágenes. Para lo cual se debe considerar que esta tarea requiere que los usuarios definan los objetos que desean extraer a través de un proceso interactivo entre el usuario y algoritmos.

Evidentemente, la evaluación de este trabajo debe enfocarse en los algoritmos de segmentación interactiva basados en *Graph Cuts*. Con este objetivo en mente, se han seleccionado cuatro métodos relevantes en el estado del arte que cuentan con implementaciones públicas.

GrabCut destaca por ser el primer trabajo en proponer el uso de algoritmos de segmentación basados en *Graph Cuts* para la extracción de objetos [31]. Por otro lado, *OneCut*, una versión mejorada del algoritmo *GrabCut*, propone mejorar la eficiencia computacional de este tipo de métodos, reemplazando el proceso iterativo por un solo ciclo o fase [32].

De manera similar, *Random Walks* es un algoritmo reconocido en la literatura por formular la segmentación de imágenes a través de un grafo [33]. Este algoritmo también requiere la intervención del usuario para definir el objeto de interés.

En contraste, *SIOX* no es un algoritmo basado en *Graph Cuts*, pero se ha incluido en la evaluación como referencia por ser un algoritmo interactivo que reporta un alto desempeño y que, además, ha sido implementado en el software de edición de imágenes GIMP [34].

3.2.3. Selección de la metodología de evaluación

La segmentación de imágenes es un campo activo en el procesamiento digital de imágenes. En los últimos años se han desarrollado innumerables técnicas de segmentación de imágenes y, en consonancia, innumerables métodos de evaluación han sido propuestos para su evaluación. Esto se debe a que la definición de una buena segmentación depende de un criterio subjetivo según la aplicación [41]. Pese a este inconve-

niente, ciertos métodos de evaluación han empezado a ser ampliamente aceptados en este campo porque reflejan la calidad de las imágenes segmentadas.

Las metodología de investigación pueden ser divididas, según Zhang *et al.* [41], en evaluaciones supervisadas y no supervisadas. Las evaluaciones supervisadas evalúan la calidad de segmentación de una imagen utilizando una referencia conocida como *ground truth*. La referencia es una imagen manualmente segmentada por una persona. Esto permite que se conserve el criterio cualitativo en el proceso de evaluación [42].

Por otro lado, las evaluaciones no supervisadas no utilizan el criterio de una persona como referencia. Estas evaluaciones miden la calidad de la segmentación utilizando la información disponible de la imagen [41]. Sin embargo, la formulación de los criterios de evaluación presenta la misma dificultad que al diseñar un algoritmo de segmentación.

Es evidente que una evaluación supervisada simplifica la definición de una buena segmentación, ya que se tiene como referencia la percepción de las personas. Además, esta evaluación se ha convertido en la metodología de evaluación estándar en trabajos de segmentación de imágenes [43, 40, 39, 44, 4]. Por estos motivos, este trabajo utiliza una evaluación supervisada.

Sin embargo, la segmentación interactiva requiere que un usuario defina el fondo y el objeto de interés de una imagen través de semillas o *seeds*. Por consiguiente, los métodos de evaluación deben considerar la intervención del usuario para permitir la reproducibilidad de los resultados, importante para el estudio y comparación de algoritmos de segmentación.

Pese a que la literatura referente a la evaluación de segmentación interactiva es minoritaria en comparación a la evaluación de segmentación automática [39], existen diversas propuestas para abordar este problema:

(a) Imágenes originales (b) Segmentación manual (c) Segmentación manual

Figura 26: Ejemplo de imágenes de la base de datos de Berkeley.
Fuente: Imágenes de Arbelaez *et al.* [43].

- **Bases de datos:** La base de datos de la Universidad de California en Berkeley [43] se ha convertido en la base de datos estándar para la evaluación de algoritmos de segmentación. Esta colección está conformada por cinco segmentaciones manuales como referencia para cada una de las 500 imágenes. Por ejemplo, en la Figura 26 se observan dos imágenes con dos segmentaciones diferentes. La principal desventaja de esta base de datos es que las imágenes están sementadas en diferentes regiones y no cuentan con una descripción semántica, por lo que no es posible distinguir entre objetos y fondo. Por tanto, esta base de datos no es adecuada para segmentación interactiva de imágenes donde el objetivo es extraer el objeto del fondo.

  Existen dos bases de datos que facilitan la evaluación de algoritmos de segmentación. La primera es la base de datos suministrada por el Instituto Weizmann [45]. La colección incluye 100 imágenes segmentadas es dos regiones, objeto y fondo, a manera de *ground truth*. No obstante, no cuenta con ningún tipo de señalización de usuarios como, por ejemplo, *scribbles* o cuadros delimitadores.

Por tal motivo, los resultados de las evaluaciones realizadas con esta bases de datos no son reproducibles.

Por otro lado, la base de datos GrabCut, incluida en el trabajo de Rother *et al.* [31], cuenta con 50 imágenes, *ground truth* y *trimaps* a manera de señalización de usuarios. Las imágenes fueron seleccionados de tal manera que el objeto de interés sea identificado sin ninguna ambigüedad. Por estas razones, esta base de datos es ampliamente utilizada en la evaluación de algoritmos de segmentación interactiva [46]. Sin embargo, como se puede observar en la Figura 3.27(c), estos *trimaps* ya etiquetan la mayor parte de los pixeles como pertenecientes a la región del fondo o del objeto. Además, los *trimaps* no son señalizaciones realistas porque a un usuario le tomaría mucho tiempo crearlos.

- **Evaluación con usuarios:** McGuinness *et al.* [39] propusieron una evaluación de algoritmos de segmentación interactiva a través de experimentos con usuarios. Para esto, los evaluadores mostraron a los usuarios imágenes con indicaciones sobre los objetos que debían extraer de las imágenes. Posteriormente, los usuarios marcaron el fondo y el objeto de interés utilizando diferentes algoritmos de segmentación. Es indiscutible que esta evaluación es prolongada y no permite la reproducibilidad de los resultados porque depende de los usuarios.

- **Evaluación automatizada:** En las evaluaciones automatizas [44, 39], los usuarios son reemplazados por usuarios robot que emulan su comportamiento. Estos usuarios robot emulan la interacción mediante la generación automática de *scribbles*. Esta evaluación permite evaluar el esfuerzo requerido por los algoritmos para segmentar una imagen. Sin embargo, los usuarios robot sitúan la secuencia de *scribbles* de acuerdo al resultado de la segmentación previa. Es decir, los *scribbles* generados son diferentes para cada algoritmo y, por tanto, no se cumple con la condición de repetitividad de resultados.

(a) Imágenes originales (b) *Ground truth* (c) *Trimaps*

Figura 27: Ejemplo de imágenes de la base de datos GrabCut.
Fuente: Imágenes de Rother *et al.* [31].

Es evidente que las evaluaciones presentadas no consideran la repetitividad y reproducibilidad de los resultados de segmentación, lo cual dificulta la comparación de desempeño de nuevos algoritmos. Además, los estudios realizados no consideran el efecto de las características de los *scribbles* en el desempeño de los algoritmos.

Ciertamente, no existe en la literatura ninguna evaluación que considere las características de los *scribbles* para la evaluación de los algoritmos de segmentación interactiva. De manera que en este trabajo se ha extendido la base de datos *GrabCut* con *scribbles*; lo cual no sólo permite analizar el influjo de las características de los *scrlibbles*, sino que también permite la repetitividad y reproducibilidad de los resultados presentados.

3.2.4. Selección de las métricas

Para responder a los planteamientos formulados en la sección 3.2.1, se utilizan tres criterios, empleados en evaluaciones anteriores [4, 39, 40], para la valoración del desempeño de los algoritmos de segmentación interactiva:

- **Eficiencia:** La eficiencia se calcula midiendo el tiempo requerido por cada algoritmo para completar la segmentación de las imágenes.

- **Esfuerzo:** El desempeño de un algoritmo de segmentación interactiva depende de las estrategias computacionales y de la guía del usuario. Ciertamente, un algoritmo que presenta una buena segmentación pudo haber requerido que el usuario señale el objeto de interés con más detalle en comparación a otros algoritmos. Por este motivo, es necesario caracterizar esta intervención y presentar un análisis cuantitativo del esfuerzo que requieren los algoritmos.

  En trabajos anteriores, el esfuerzo ha sido presentado como la cantidad de interacciones que un usuario realiza para alcanzar una exactitud determinada [4, 40, 39]. Sin embargo, esta medición no puede ser realizada en el experimento propuesto porque se utiliza una base de datos con *scribbles* predefinidos. En consecuencia, el esfuerzo se determina a través de la medición de efectividad de los algoritmos para los dos conjuntos de *scribbles*, los cuales señalan el objeto de interés con distintos niveles de detalle.

- **Exactitud:** La exactitud se mide calculando la similitud entre los resultados de la segmentación y la referencia *ground truth*. La información de semejanza entre los resultados se puede observar a través de las regiones y bordes de la segmentación. Por tal motivo, las métricas de segmentación se pueden enfocar en dos planteamientos: métricas basadas en regiones y métricas basadas en bordes.

3.2.4.1. Métricas de exactitud basadas en regiones

La segmentación interactiva divide una imagen en dos regiones: objeto y fondo. Por tanto, la segmentación interactiva se puede tratar como un problema de clasificación binaria. Entonces, estas dos regiones pueden ser clasificadas como regiones pertenecientes o no pertenecientes al objeto.

En este contexto se puede definir que los pixeles positivos son los pixeles pertenecientes al objeto, mientras que los pixeles negativos corresponden a los pixeles pertenecientes al fondo. Con esta consideración, una imagen es la unión de los pixeles negativos y positivos como se indica en la ecuación 3.1

$$I = P_M \cup N_M \tag{3.1}$$

Considerando la imagen segmentada (M) y la referencia *ground truth* (G). los pixeles positivos (P) y negativos (N) pueden ser subdivididos utilizando la teoría de clasificación binaria. Tuset [42] define los subconjuntos de esta manera:

- Positivos Verdaderos (TP): Pixeles que son detectados como pertenecientes al objeto y están etiquetados como tal en la referencia *ground truth*.

$$TP = P_M \cap P_G \tag{3.2}$$

- Positivos Falsos (FP): Pixeles que son detectados como pertenecientes al objeto, pero no están etiquetados como tal en la referencia *ground truth*.

$$FP = P_M \cap N_G \tag{3.3}$$

- Negativos Falsos (FN): Pixeles que son detectados como no pertenecientes al objeto, pero no están etiquetados como tal en la referencia *ground truth*.

$$FN = N_M \cap P_G \tag{3.4}$$

- Negativos Verdaderos (TN): Pixeles que son detectados como no pertenecientes al objeto y están etiquetados como tal en la referencia *ground truth*.

$$TN = N_M \cap N_G \tag{3.5}$$

Las definiciones de TP, FP, FN y TN se ilustran en el diagrama de Venn de la Figura 28. En el diagrama, M corresponde a la imagen segmentada por cualquier algoritmo de segmentación y G es la referencia *ground truth*.

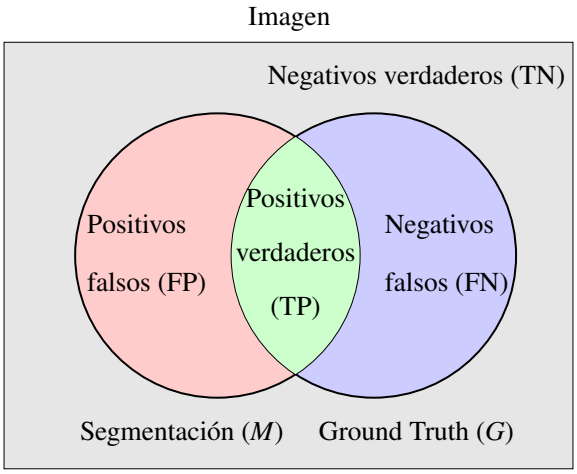


Figura 28: Diagrama de Venn para regiones de segmentación.

Por consiguiente, estas definiciones pueden ser aplicadas para medir la exactitud de los algoritmos de segmentación interactiva. Las métricas comúnmente utilizadas en la evaluación de estos algoritmos son las siguientes:

- **Exactitud de clasificación (ACC):** Es la proporción de pixeles positivos (TP y FP) del total de pixeles de la imagen como se expresa en la ecuación 3.6.

$$ACC = \frac{TP+TN}{TP+FP+TN+FN} \tag{3.6}$$

- **Sensibilidad (TPR):** También conocida como exhaustividad (*recall* en inglés). Esta medida presenta el porcentaje de pixeles positivos de la referencia *ground truth* que han sido detectados correctamente. Por tanto, esta ecuación se puede expresar como en la ecuación 3.7.

$$TPR = \frac{TP}{P_G} = \frac{TP}{TP+FN} \tag{3.7}$$

- **Precisión (*PPV*):** De manera análoga, precisión (*precision* en inglés) es el porcentaje de pixeles detectados que han sido correctamente identificados. Este proporción se expresa en la ecuación 3.8.

$$PPV = \frac{TP}{P_M} = \frac{TP}{TP + FP} \tag{3.8}$$

- **Valor F (F_1):** Idealmente, los algoritmos deberían presentar valores altos de precisión y sensibilidad. Sin embargo, esto no suele ocurrir y, en general, existe una compensación entre los dos valores [42]. Esta compensación se define como el valor F o medida F (*F-measure* en inglés), y corresponde a la media armónica de la precisión y sensibilidad, tal y como se expresa en la ecuación 3.9.

$$F_1 = (1 + \beta^2)\frac{PPV \cdot TRR}{\beta^2 \cdot PPV + TRR} \tag{3.9}$$

  En término de detecciones positivas o negativas, esta métrica puede ser expresada como la ecuación 3.10.

$$D = \frac{2TP}{2TP + FN + FP} \tag{3.10}$$

  Esta expresión concuerda con la definición del **Coeficiente Dice**. Por ejemplo, esta métrica ha sido utilizada por Alpert *et al.* [45] para reportar el desempeño de algoritmos de segmentación. Por este motivo y por convención, en este trabajó se llamará Coeficiente Dice (*Dice*) a la métrica enfocada en regiones.

- **Especificidad (*SPC*):** La especificidad es inversa a la sensibilidad [47]. Esta métrica es el porcentaje de pixeles negativos de la referencia *ground truth* que han sido detectados correctamente como se expresa en la ecuación 3.11.

$$SPC = \frac{TN}{N_G} = \frac{TN}{TN + FP} \tag{3.11}$$

- **Índice Jaccard (*Jaccard*):** El índice Jaccard es una métrica de superposición de regiones que es popularmente utilizado en diversos trabajos de segmentación y detección de objetos. Por ejemplo, McGuinness *et al.* [39, 40] utilizaron este índice para sus evaluaciones de algoritmos de segmentación interactiva. El índice Jaccard se utiliza para evaluar si un objeto ha sido correctamente segmentado a través de la expresión de la ecuación 3.12.

$$Jaccard = \frac{|G \cap M|}{|G \cup M|} = \frac{TP}{TP + FN + FP} \tag{3.12}$$

  Se puede observar que la expresión de la ecuación 3.12 es similar al coeficiente Dice, pero este coeficiente otorga doble peso al porcentaje de pixeles pertenecientes al objeto que fueron correctamente identificados como tal. Por consiguiente, como se expresa en la ecuación 3.13, el índice Jaccard y el coeficiente Dice están funcionalmente relacionados [42].

$$Jaccard = \frac{F_1}{2 - F_1} \tag{3.13}$$

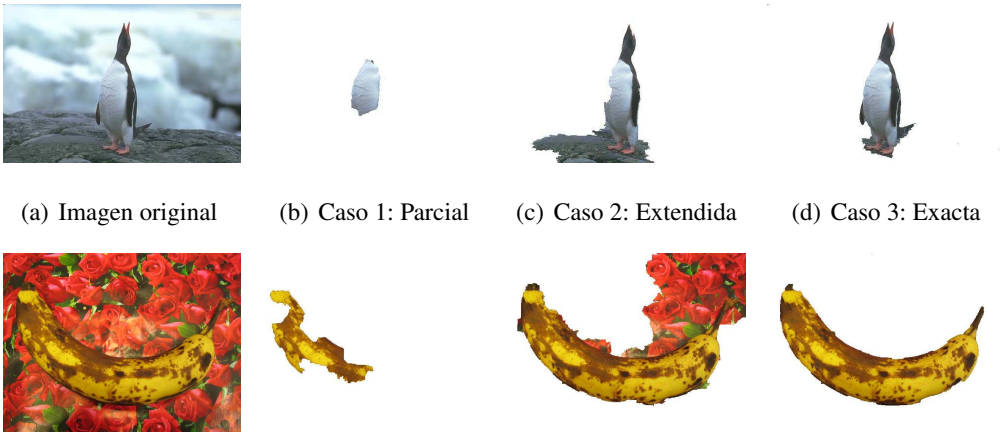
  Esto implica que el índice Jaccard y el coeficiente Dice son equivalentes en una evaluación. Por ejemplo, los mejores algoritmos identificado en una evaluación realizada con el índice Jaccard serían, de igual manera, los mejores algoritmos en una evaluación que emplea el coeficiente Dice.

3.2.4.2. Efectividad de las métricas basadas en regiones

La diversidad de métricas dificulta el establecimiento de un estándar en la evaluación de algoritmos de segmentación interactiva. Esta diversidad también establece la necesidad de verificar la efectividad de las métricas para mostrar la calidad de los algoritmos evaluados.

En la Figura 29 dos imágenes de la base de datos y seis casos de segmentación. La imágenes de las Figuras 3.29(b) y 3.29(f) corresponden al primer grupo de segmenta-

ción parcial. Estos casos presentan segmentaciones muy inexactas donde el objeto no ha sido extraído completamente del fondo.



(a) Imagen original (b) Caso 1: Parcial (c) Caso 2: Extendida (d) Caso 3: Exacta

(e) Imagen original (f) Caso 4: Parcial (g) Caso 5: Extendida (h) Caso 6: Exacta

Figura 29: Ejemplos de imágenes y distintos casos de segmentación.
Fuente: Imágenes originales de Rother *et al.* [31].

La imágenes de las Figuras 3.29(c) y 3.29(g) corresponden al segundo grupo de segmentación extendida. Estos casos presentan segmentaciones deficientes donde el objeto ha sido completamente extraído, pero una porción del fondo también ha sido identificada como objeto. Finalmente, las imágenes de las Figuras 3.29(d) y 3.29(h) corresponden al tercer grupo de segmentación exacta. En este grupo se muestran algunos de los casos más exactos de segmentación presentados en este experimento.

Como se observa en la tabla 2, es evidente que la sensibilidad (TPR) penaliza los pixeles del objeto que fueron marcados como fondo, pero es menos susceptible a los pixeles el fondo que fueron marcados incorrectamente como objeto. Por otro lado, el coeficiente Dice y el índice Jaccard son más susceptibles a las variaciones de pixeles erróneos que la especificidad (SPC). Ciertamente, mientras estas dos métricas muestran resultados drásticamente diferentes para los casos mostrados en la Figura 29, la especificidad (SPC) y exactitud de clasificación (ACC) sólo varían $\pm 26\,\%$.

Tabla 2:

Medidas de exactitud basadas en regiones.

Caso	ACC	TPR	SPC	Jaccard	Dice
1: Parcial	0.826	0.295	1.000	0.295	0.456
2: Extendida	0.797	0.975	0.738	0.543	0.704
3: Exacta	0.992	0.967	1.000	0.966	0.983
4: Parcial	0.940	0.324	1.000	0.324	0.490
5: Extendida	0.919	0.773	0.933	0.458	0.628
6: Exacta	0.986	0.909	0.994	0.854	0.921

El índice Jaccard y el coeficiente Dice reflejan con mayor detalle la calidad de la imagen segmentada porque proporcionan más información sobre la variación de exactitud. Estas métricas cambian significativamente de valor según el desempeño de los algoritmos varía. Además, Jaccard y Dice son las métricas más utilizadas en la evaluación de algoritmos de segmentación interactiva. Desde esta perspectiva, sería razonable utilizar ambas métricas para la evaluación de la exactitud basadas en regiones.

Sin embargo, utilizar el índice Jaccard y el coeficiente sería una labor redundante porque se mostró que estas dos medidas son equivalentes y que, por tanto, las conclusiones de una evaluación comparativa serían la mismas con las dos métricas. Se puede decir, incluso, que Jaccard y Dice muestran las mismas variaciones y características de los algoritmos de segmentación evaluados, pero con distintos valores. Por lo que es necesario identificar qué métrica arroja el valor más intuitivo para calificar la calidad de una imagen segmentada.

Para esto, se debe considerar que el coeficiente Dice otorga mayor peso al porcentaje de pixeles del objeto detectados correctamente. Entonces, se puede argumentar que Jaccard es más susceptible a errores que Dice. Por ejemplo, en la tabla 2, los resultados de Jaccard varían entre 0.295 y 0.966, mientras que los valores de Dice entre 0.456 - 0.983. Es evidente que Dice no penaliza con suficiente rigurosidad los casos de

segmentación parcial de las Figuras 3.29(b) y 3.29(f). En ambos casos, los pixeles del objeto han sido etiquetados incorrectamente en un porcentaje superior al 60 %, pero el coeficiente Dice valora estas segmentaciones con 0.456 y 0.49, respectivamente.

Por las razones mencionadas, en este trabajo se utiliza el índice Jaccard para evaluar la exactitud basadas en regiones de los algoritmos de segmentación interactiva.

3.2.4.3. Métricas de exactitud basadas en bordes

En la segmentación interactiva los bordes son el límite entre la región del objeto y la región del fondo. Por esta razón, la medición de exactitud basadas en bordes se pueden plantear como un problema de detección de bordes. Por otra parte, aplicando la analogía de clasificación binaria, se puede manifestar que una imagen está compuesta por pixeles que pertenecen o no pertenecen al borde [48].

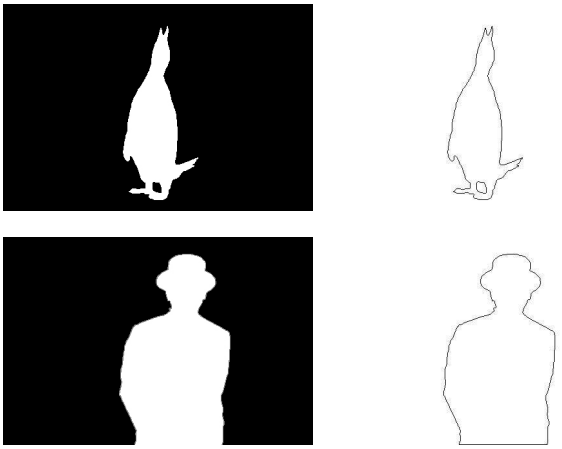
Esta aproximación de clasificación binaria permite utilizar las métricas utilizadas en la sección 3.2.4.1. Sin embargo, los pixeles negativos y positivos deben ser determinados de una manera más flexible. Si, por ejemplo, los pixeles positivos y negativo son determinados sólo en función del borde de la referencia *ground truth*, es inevitable que la mayoría de los pixeles del borde de la imagen segmentada sean etiquetados como incorrectos. Entonces, es necesario adoptar una metodología más tolerable a errores pequeños. De manera que las métricas no penalicen demasiado a los algoritmos que producen segmentaciones aceptables, pero con bordes ligeramente mal ubicados [49].

En los últimos años, la evaluación propuesta por Martin *et al.* [49], que utiliza curvas de precisión y exhaustividad, se ha convertido en un estándar para evaluaciones basadas en bordes. De hecho, estas curvas se han utilizado en el campo de recuperación de información para determinar qué porcentaje de señal se necesita para ser detectada (exhaustividad) y qué porcentaje de ruido es tolerado (precisión) [49].

Para resolver el problema de la tolerancia a errores, Martin *et al.* [49] estiman una correspondencia entre los bordes generados por el algoritmo y los bordes de la

referencia *ground truth* [49]. Es entonces cuando se procede a calcular las curvas de precisión y exhaustividad.

Estas curvas son empleadas para evaluar segmentación de imágenes porque muestran la compensación que existe entre la precisión y exhaustividad. Un punto específico en esta curva se define como valor f (F_1). Al igual que en las métricas basadas en regiones, esta métrica se expresa en la ecuación 3.9, donde $\beta = 0{,}5$ [49]. El valor F (F_1) se utiliza para caracterizar el desempeño de los algoritmos [49].



(a) *Ground truth* (b) Mapa de bordes

Figura 30: Mapas de bordes de la referencia *ground truth*.
Fuente: Imágenes originales de Arbelaez *et al.* [43].

En este trabajo se utiliza la herramienta de evaluación SEISM, proporcionada por Tuset y Marques [48], para la evaluación de la exactitud basada en bordes de los algoritmos de segmentación interactiva. Esta herramienta utiliza los mapas de bordes de las imágenes segmentadas de la base de datos de Berkeley [43]. Por consiguiente, las referencias *ground truth* de la base de datos generada para este trabajo deben ser transformadas en mapas de bordes, como se muestra en la Figura 3.30(b). De igual manera, se debe utilizar mapas de bordes de los resultados de los algoritmos de segmentación.

3.2.5. Descripción del experimento

3.2.5.1. Condiciones experimentales

Los experimentos fueron realizados en una MacBook Pro con procesador Intel Core i7 2.9 GHz, 8 Gb de RAM y sistema operativo Ubuntu 14.04 (64 bits). Se utilizó la implementación de *GrabCut* en OpenCV C++. Además, se utilizó la implementación en C++ de *OneCut*, la implementación en MATLAB de *Random Walks* y la implementación en Java de *SIOX* que fueron proporcionadas por los autores.

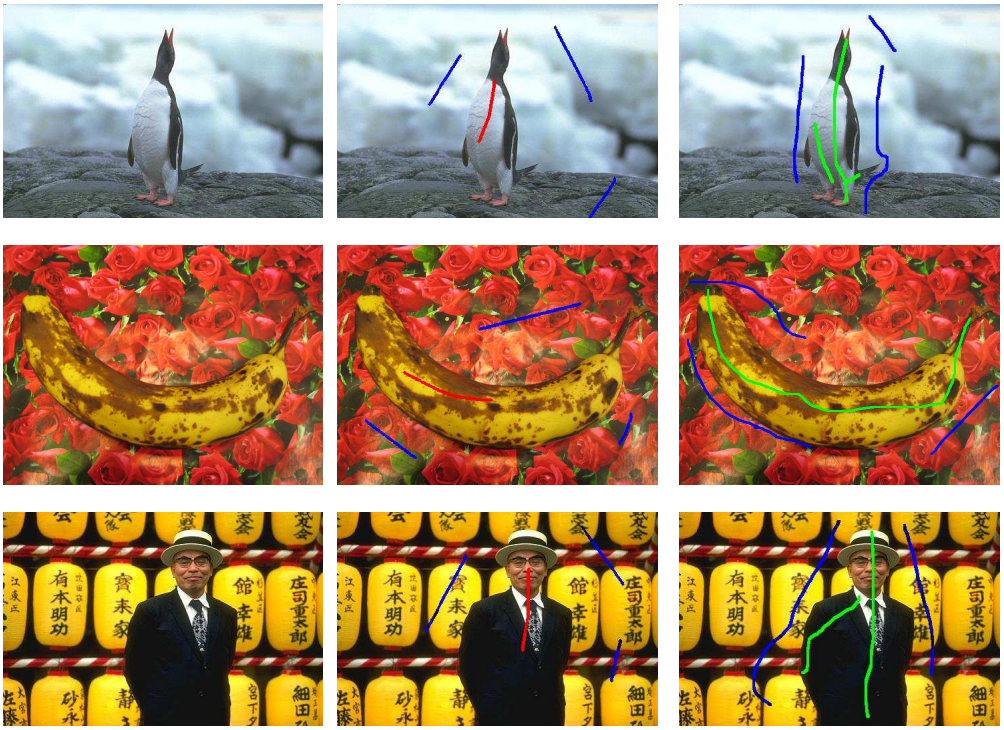
Para garantizar la reproducibilidad de los resultados de esta evaluación, se emplearon los dos conjuntos de *scribbles* como señalización de usuarios. Se aclara que en estos experimentos, el algoritmo *GrabCut* no fue inicializado con cajas delimitadoras, propuestas originalmente por Gulshan *et al.* [44], sino que sólo se utilizó *scribbles*.

Finalmente, no sólo se emplearon los valores sugeridos por los autores para los parámetros de los algoritmos, sino que también se utilizaron distintos valores para evaluar el desempeño de los algoritmos de segmentación interactiva.

3.2.5.2. Base de datos

La base de datos generada para este trabajo está compuesta por 50 imágenes, referencias *ground-truth* y dos conjuntos de *scribbles* o semillas. Las imágenes contienen objetos de interés claramente definidos que pueden ser extraídos por los usuarios sin ambigüedad alguna.

Para esta recopilación se utilizaron las imágenes y *ground truth* de la base de datos GrabCut [31]. Se debe notar que las regiones de *ground truth* incluyen una región de pixeles no clasificados porque existen casos en los que no es posible identificar si un pixel pertenece a la región del fondo o del objeto. Estos pixeles no son considerados en las mediciones.



(a) Imágenes originales (b) Primer conjunto (c) Segundo conjunto

Figura 31: Ejemplo de imágenes y conjuntos de *scribbles* de la base de datos.
Fuente: Imágenes originales de Rother *et al.* [31].

Los dos conjuntos de *scribbles* corresponden a la señalización de pixeles pertenecientes al fondo y al objeto de interés. Para el primer conjunto de *scribbles*, se utilizaron las semillas de los usuarios robot de la base de datos de Gulshan *et al.* [44] para evaluación de algoritmos de segmentación. Este conjunto utiliza en promedio cuatro *scribbles* en cada imagen, pero señalan una región pequeña del objeto de interés como se muestra en las imágenes de la Figura 3.31(b). En este conjunto, el promedio de pixeles etiquetados como objetos es de 634.

Para extender esta base de datos, se generó manualmente un segundo conjunto de *scribbles*. Como se aprecia en las imágenes de la Figura 3.31(c), estos señalan con mayor detalle el objeto de interés. Para este conjunto se han etiquetado en promedio 2330 pixeles como objeto.

Según lo mencionado previamente, es claro que los conjuntos de *scribbles* reflejan dos grados de esfuerzo del usuario: el segundo conjunto marca más regiones del objeto en comparación al primer conjunto. En el segundo conjunto, el número de pixeles etiquetados como objeto es 2.67 veces el número de pixeles del primer conjunto de *scribbles*.

3.3. Restauración de imágenes

3.3.1. Objetivo de la fase experimental

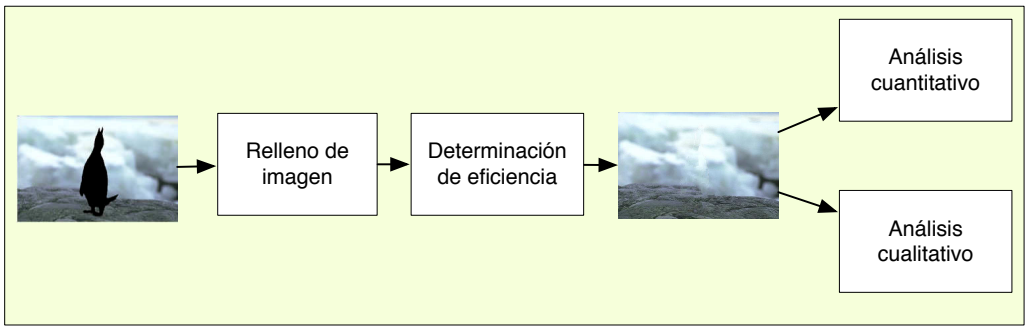


Figura 32: Diagrama del proceso de evaluación de algoritmos de restauración.

El objetivo de la fase experimental es identificar el algoritmo de restauración de imágenes (*inpainting*) más adecuado para la tarea de extracción de objetos. Para esto, es necesario diseñar un experimento que permita determinar qué algoritmo es el más eficiente y eficaz.

Bertalmio *et al.* [1] mencionaron en su trabajo que una eficiente restauración digital de imágenes implica que un observador no se percate de que este proceso ha sido realizado. No obstante, a diferencia de la segmentación de imágenes, el criterio para definir un buen resultado no está generalizado y, por tanto, dos resultados completamente diferentes pueden ser calificados como acertados.

Por esta razón, incluso la recopilación de una base de datos se convierte en una tarea compleja porque el criterio para definir una referencia tampoco está claro. Si por ejemplo, un objeto es removido, ¿es necesario comparar la imagen restaurada con un fondo real?

En este trabajo, por ende, no se realiza una evaluación íntegramente cuantitativa. Como se muestra en el diagrama de la Figura 32, el proceso del experimento propuesto incluye una evaluación cualitativa.

En este experimento, el primer paso es la determinación de la eficiencia de los algoritmos de restauración. Para esto, se debe medir el tiempo que los algoritmos tardan en rellenar áreas deterioradas. Posteriormente, los resultados de estos algoritmos son utilizados para determinar la eficacia de la segmentación. Esta medición, como se mencionó previamente, tiene dos enfoques:

- **Cuantitativo:** La eficacia es determinada aplicando medidas de la calidad de imágenes. Este enfoque ha sido ampliamente utilizado en el estado del arte [50].
- **Cualitativo:** Un grupo de personas determina la eficiencia calificando la calidad de las imagenes restauradas con los algoritmos evaluados.

Estos dos criterios permiten examinar la aptitud de las métricas basadas en la calidad de una imagen para representar el desempeño de los algoritmos evaluados.

3.3.2. Selección de los algoritmos

En la sección 2.5 se identificaron y describieron los principales métodos de restauración de imágenes. Se observa que, a diferencia del campo de segmentación de imágenes, los algoritmos de restauración digital de imágenes comenzaron a proponerse desde hace unos años [1]. Por tal motivo, la literatura relacionada con esta tarea es menor.

En esta evaluación, por tanto, se han elegido tres métodos relevantes en el estado del arte que cuentan con implementaciones públicas. La identificación de cada algoritmo se la ha realizado utilizando el nombre del autor principal.

Los algoritmos de Bertalmio [36] y Telea [38] pertenecen a los métodos de restauración basada en ecuaciones diferenciales parciales. Estos algoritmos son utilizados para rellenar zonas pequeñas de una imagen porque no replican texturas.

El método de Bertalmio utiliza un proceso iterativo basado en Navier-Stokes para la restauración de las imágenes. De manera similar, el método Telea se basa en la propuesta original de Bertalmio *et al.* [1], pero es más eficiente porque no utiliza un proceso iterativo.

En contraste, el algoritmo de Criminisi es un método de restauración basada en parches. Este algoritmo es muy útil en áreas deterioradas más amplias porque puede replicar información de intensidades y texturas. Con este propósito, el algoritmo copia la información de los pixeles adyacentes a la región deteriorada.

3.3.3. Selección de la metodología de evaluación

Los algoritmos de restauración digital buscan rellenar áreas deterioradas de una imagen de manera que un observador no perciba la intervención [1]. Es evidente que este objetivo es esencialmente subjetivo porque requiere la valoración del observador.

Por consiguiente, la principal dificultad para la evaluación surge al trasladar las valoraciones de los observadores a criterios cuantitativos. La primera opción sopesada para abordar esta dificultad es el uso de una base de datos con referencias, tal y como se utilizó en la evaluación de segmentación interactiva. Sin embargo, a diferencia de los algoritmos de segmentación, en la restauración no existe un consenso generalizado sobre el aspecto que debe tener una segmentación eficaz en relación a una referencia. De hecho, el área restaurada de una imagen puede ser diferente a la imagen de referencia y ser, al mismo tiempo, una restauración aceptada por un observador.

En consecuencia, es necesario abordar otro enfoque para efectuar una evaluación cuantitativa de los algoritmos. Para esto se debe describir las características de una restauración adecuada de tal forma que pueda ser medible. Entonces, siguiendo las premisas de los algoritmos presentados en la sección 2.5, se puede establecer que una restauración es adecuada cuando:

- Se ha replicado la textura de las regiones adyacentes al área deteriorada.
- Se han replicado los valores de intensidad de las regiones adyacentes al área deteriorada.
- No hay artificios en la región restaurada. Esto significa que no hay interrupciones en la continuidad de texturas e intensidades.

Con esta finalidad en mente, la evaluación de los algoritmos de restauración digital deben reflejar si las imágenes restauradas cumplen con estos condiciones. Ciertamente, en este campo se han empleado diversas métricas de calidad para analizar los métodos propuestos. No obstante, pocos trabajos dedican un análisis a la capacidad de las métricas utilizadas para mostrar la correlación entre estas métricas y la calidad percibida por los observadores [37].

Por este motivo, en este trabajo se propone una evaluación cualitativa de los algoritmos de restauración digital de imágenes. Esto no sólo permite identificar el algoritmo más eficaz, sino que también se convierte en una referencia para analizar la efectividad de las métricas para reflejar el desempeño de los algoritmos.

3.3.4. Evaluación cuantitativa

3.3.4.1. Condiciones experimentales

Los experimentos fueron realizados en una MacBook Pro con procesador Intel Core i7 2.9 GHz, 8 Gb de RAM y sistema operativo Ubuntu 14.04 (64 bits). Se utilizaron las implementaciones de los algoritmos de Bertalmio y Telea en OpenCV C++.

Además, se utilizó la implementación en MATLAB del algoritmo de Criminisi proporcionada por Sooraj Bhat [51].

Debido a que este trabajo se enfoca en la tarea de extracción de objetos, en este experimento se utilizó la base de datos proporcionada en la sección 3.2.5.2 para la evaluación de algoritmos de segmentación. Esta disposición posibilita la verificación del desempeño de los algoritmos de *inpainting* para restaurar las regiones correspondientes a los objetos de la base de datos *GrabCut*.

Se utilizaron, además, las máscaras de referencia de la base de datos para señalar las regiones deterioradas. No se utilizaron las máscaras generadas por los algoritmos de segmentación porque en este trabajo se pretende evaluar el desempeño de la restauración de manera independiente. Además, el desempeño de los algoritmos de restauración digital no variará si se cambian las máscaras.

En esta evaluación no sólo se emplearon los valores sugeridos por los autores para los parámetros de los algoritmos, sino que también se utilizaron distintos valores para evaluar el desempeño de los algoritmos de restauración digital.

3.3.4.2. Selección de las métricas

Para responder a los planteamientos formulados en la sección 3.3.3, se utilizan dos criterios para la valoración del desempeño de los algoritmos de restauración digital de imágenes:

- **Eficiencia:** La eficiencia se calcula midiendo el tiempo requerido por cada algoritmo para completar la restauración de las imágenes.
- **Eficacia:** En el campo de la restauración, las medidas de calidad de una imagen han sido ampliamente utilizadas para evaluar la eficacia de los algoritmos [37].

En esta sección se presentan las dos métricas más utilizadas en el estado del arte. Para esto se considera la imagen original I_O y la imagen restaurada I_R:

- **MSE:** El error cuadrático medio calcula el promedio de los errores cuadráticos entre dos imágenes. Esta métrica está definida por la expresión de la ecuación 3.14 [37].

$$MSE = \frac{1}{mn} \sum_{y=0}^{m-1} \sum_{x=0}^{n-1} = [I_O(x,y) - I_R(x,y)]^2 \tag{3.14}$$

  En esta ecuación, x, y corresponden a la localización de los pixeles de la imagen. Por otro lado, m es el número de filas y n es el número de columnas de las imágenes. Con esta medida, una imagen es de mejor calidad cuando el valor de MSE es menor.

  De igual manera que en el trabajo de Richard *et al.* [52], en esta evaluación se calcula el MSE de cada uno de los canales de color RGB.

- **PSNR:** La relación señal a ruido de pico estima la proporción entre la señal y ruido de dos imágenes. Por tanto, este valor será mayor para imágenes de mejor calidad. El valor de PSNR se calcula con la expresión de la ecuación 3.15 [37].

$$PSNR = 10 \log_{10} \frac{R^2}{MSE} \tag{3.15}$$

  En donde R es el máximo valor que un pixel puede tomar. En este trabajo, el máximo valor que puede tomar un pixel es $R = 255$. Además, el resultado de esta métrica está dado en decibelios.

3.3.5. Evaluación cualitativa

El experimento aquí descrito evalúa cualitativamente los algoritmos de restauración digital en base a las calificaciones otorgadas por un grupo de personas.

3.3.5.1. Condiciones experimentales

En este experimento se utilizaron las 50 imágenes pertenecientes a la base de datos *Grabcut*. Las imágenes fueron restauradas con los algoritmos de Bertalmio, Telea y Criminisi. Sin embargo, debido a que una evaluación cualitativa requiere tiempo, no se evaluaron todos los resultados obtenidos para cada variación de los parámetros de los algoritmos. En su lugar, se seleccionaron los siguientes casos:

- **Bertalmio.** Radio del vecindario: 40
- **Telea.** Radio del vecindario: 40
- **Criminisi.** Tamaño del parche: 9

El grupo de participantes que evaluó la calidad de las imágenes restauradas estuvo conformado por 21 estudiantes la Universidad de las Fuerzas Armadas - ESPE.

3.3.5.2. Metodología de evaluación

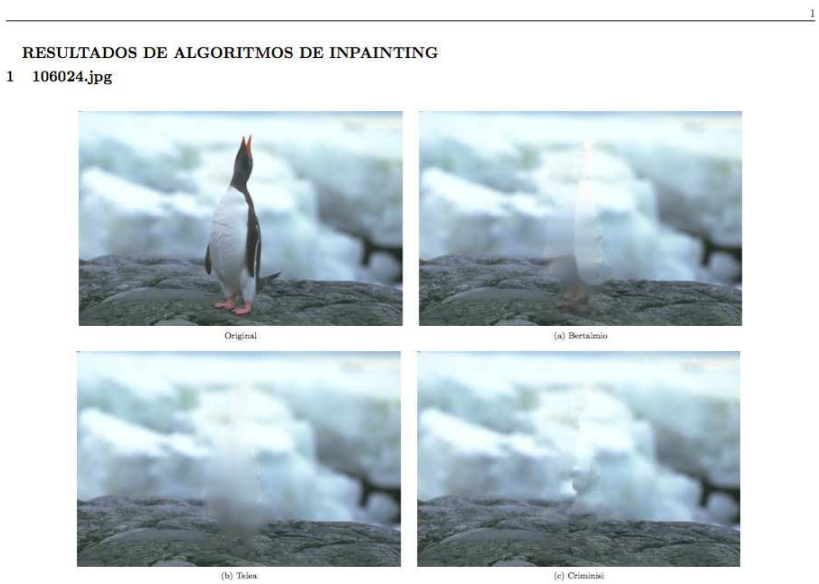


Figura 33: Ejemplo de casos mostrados a participantes.

En este evaluación se utilizó una metodología similar a la metodología empleada en las pruebas para obtener la opinión media de un servicio de voz (*Mean Opinion Score* en inglés). Este tipo de evaluaciones se han utilizado en el campo de procesamiento digital de imágenes para evaluar la calidad de las imágenes [37, 53].

Por consiguiente, siguiendo esta metodología, a cada participante se le mostró una serie de 50 casos. Cada ejemplo estaba conformado por la imagen original y tres imágenes restauradas con los algoritmos de Bertalmio, Telea y Criminisi. En la Figura 32 se observa la disposición de uno de los casos mostrados.

Posteriormente, se pidió a los participantes calificar la calidad de cada una de las imágenes restauradas a través de la siguiente pregunta:

En una calificación de 1 a 5, ¿cómo calificaría usted la calidad de las imágenes presentadas?

La escala utilizada corresponde a la escala de cinco puntos puntos de la ITU-R [37] que evalúa la calidad en términos de: excelente, muy buena, buena, mala, muy mala. Acorde a esta escala, se dice que una restauración es excelente cuando su puntaje es 5 y, por otro lado, una restauración es mala cuando su puntaje es 1. Esta equivalencia se muestra a continuación:

1. Mala
2. Regular
3. Buena
4. Muy buena
5. Excelente

Se estima, entonces, la opinión media de cada algoritmo utilizando los datos recolectados. La opinión media de cada algoritmo se calcula con la expresión de la ecuación 3.16.

$$MOS = \frac{1}{n}\sum_{i=1}^{n} S_i \tag{3.16}$$

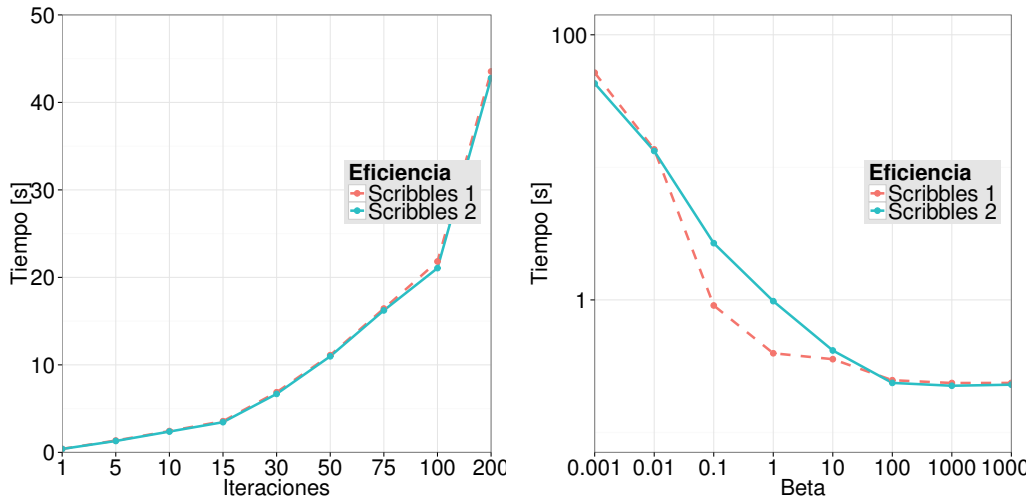
Donde n es el número de participantes y S_i corresponde al puntaje otorgado por el participante i.

CAPITULO 4

ANÁLISIS DE RESULTADOS

4.1. Segmentación de imágenes

4.1.1. Eficiencia



(a) GrabCut (b) OneCut

Figura 34: Tiempo promedio de ejecución de los algoritmos *GrabCut* y *OneCut*.

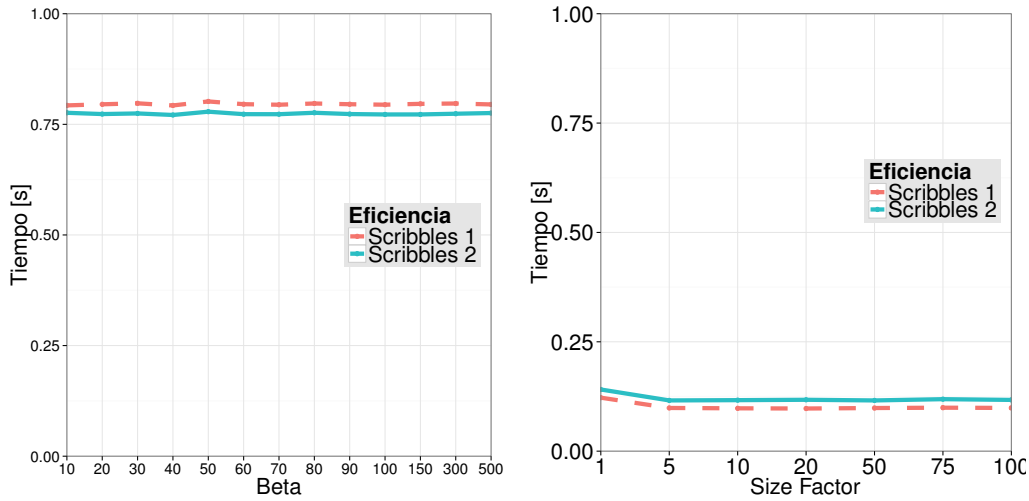
El algoritmo *GrabCut* utiliza un proceso iterativo para minimizar la energía a través de un corte mínimo [31]. El gráfico de la Figura 4.34(a) muestra que el tiempo de ejecución promedio incrementa exponencialmente conforme aumenta el número de

iteraciones realizadas, alcanzando un tiempo de ejecución superior a 40 segundos para 200 iteraciones.

En el experimento se observó que la calidad de los *scribbles* no afecta significativamente la eficiencia del algoritmo. De hecho, el tiempo promedio registrado para el segundo conjunto de *scribbles* disminuyo sólo 2 % en comparación al primer conjunto.

De manera similar, el algoritmo *OneCut* registró tiempos de ejecución superiores a 50 segundos de acuerdo a lo mostrado en la Figura 4.34(b). Para esto, se utilizó 64 bins de color conforme lo recomendado por los autores [32] y se varió el valor del parámetro β de la ecuación 2.27. Los resultados muestran que el tiempo de ejecución aumenta drásticamente cuando β es inferior a 0.1: 15 veces para el primer conjunto de *scribbles* y 5 veces para el segundo conjunto.

OneCut, a diferencia del algoritmo *GrabCut*, presenta una diferencia del 10 % entre los promedios de ejecución del primer y segundo conjunto de *scribbles*. Es evidente que este método es más sensible al nivel de detalle de los *scribbles* y, por tanto, el algoritmo es más eficiente cuando el número de pixeles marcados es mayor.



(a) Random Walks (b) SIOX

Figura 35: Tiempo promedio de ejecución de los algoritmos *Random Walks* y *SIOX*.

Por otro lado, *Random Walks* y *SIOX* son los algoritmos más eficientes en esta evaluación como se muestra en la Figura 35. En el caso de *Random Walks*, se utilizaron distintos valores para el parámetro libre de la ecuación 2.28. Los tiempos registrados de ejecución promedio se sitúan entre 0.77 y 0.88 segundos como se muestra en la Figura 4.35(b). Además, la calidad de los *scribbles* no afectan sustancialmente la eficiencia. Tan sólo se registró una disminución del 2 % en el tiempo promedio de ejecución del algoritmo al utilizar el segundo conjunto de *scribbles*.

La Figura 4.35(b) muestra claramente que *SIOX* es el algoritmo de segmentación interactiva más eficiente. La revisión del procedimiento utilizado por es algoritmo (sección 2.4.7.1) ratifica que este proceso es más eficiente porque es matemáticamente menos complejo que los algoritmos basados en *Graph Cuts*.

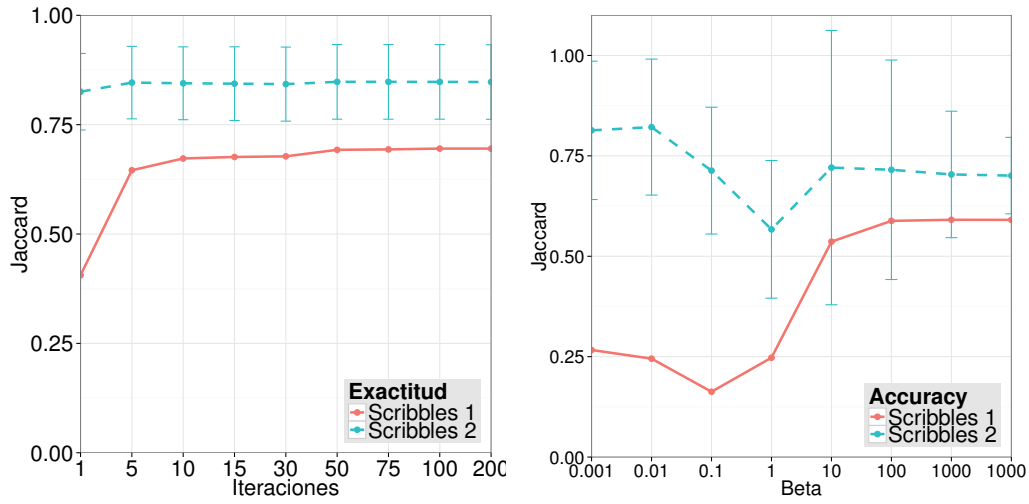
Los tiempos de ejecución promedio varían entre 0.1 y 0.14 segundos para *SIOX*. Asimismo, la eficiencia de este algoritmo tampoco es afectado por la calidad de los *scribbles* ya que sólo se registra que el tiempo de ejecución promedio disminuye 2 % en el segundo conjunto de *scribbles*.

Finalmente, se debe notar que *SIOX* y *Random Walks* son los algoritmos más eficientes de esta evaluación, a pesar de que fueron implementados en Java y MATLAB, respectivamente.

4.1.2. Exactitud

4.1.2.1. Métricas basadas en regiones

4.1.2.1.1. GrabCut: El esquema de minimización de energía del algoritmo *GrabCut* funciona iterativamente, de forma que la energía disminuye en cada paso hasta su convergencia [31]. En la Figura 4.36(a) se muestran los valores promedio del coeficiente Jaccard para cada iteración. Es evidente que la energía converge en 10 iteraciones para los dos conjuntos de *scribbles*.



(a) GrabCut (b) OneCut

Figura 36: Coeficiente Jaccard promedio de *GrabCut* y *OneCut*.

Sin embargo, pese a que la exactitud no mejora significativamente después de las 10 iteraciones del algoritmo, el tiempo de ejecución aumenta exponencialmente según lo mencionado para la Figura 4.34(a).

En general, *GrabCut* es el algoritmo más eficaz porque registra el mejor coeficiente Jaccard de esta evaluación. Los mejores valores promedios del coeficiente Jaccard corresponden a 0.69 y 0.85 para el primer y segundo conjunto de *scribbles*, respectivamente.

Se puede observar que la eficiencia del algoritmo *GrabCut* mejora considerablemente cuando los *scribbles* marcan el objeto y el fondo con mayor detalle. De hecho, de acuerdo al valor promedio del coeficiente Jaccard en 10 iteraciones, la exactitud del algoritmo para el segundo conjunto de *scribbles* mejora 20 % en comparación al primer conjunto.

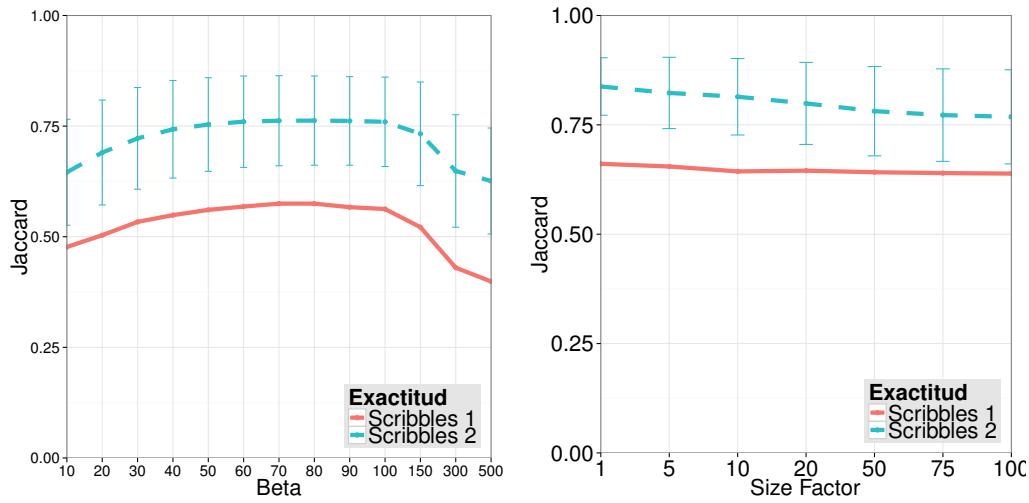
4.1.2.1.2. OneCut: En contraste, los valores promedio de Jaccard de la Figura 4.36(b) señalan que *OneCut* es el algoritmo más sensible a las características de los *scribbles*. El mejor valor del coeficiente Jaccard del segundo conjunto mejora en 40 %

al mejor valor del coeficiente Jaccard del primer conjunto de *scribbles*.

Por otro lado, esta sensibilidad también se registra cuando se cambiar el valor del parámetro β. Por ejemplo, en el primer conjunto de *scribbles*, el valor promedio del coeficiente Jaccard aumenta drásticamente después de $\beta = 1$. Sin embargo, este comportamiento no se registra en el segundo conjunto. Aquí, el menor valor promedio de Jaccard es registrado cuando $\beta = 1$, pero este valor mejora conforme β cambia. Por este motivo, un mayor tiempo de ejecución de *OneCut* no garantiza mejores resultados de acuerdo a la Figura 4.34(b).

Asimismo, la desviación estándar del coeficiente Jaccard tiene la mayor variación registrada, por lo que se puede argumentar que la eficiencia de *OneCut* no solo depende de los cambios del parámetro β y características de los *scribbles*, sino que también depende de las características de las imágenes de la base de datos.

Estas fluctuaciones en la exactitud hacen del algoritmo *OneCut* la opción menos viable para un sistema de extracción de objetos, pese a que los mejores valores promedio Jaccard son 0.59 y 0.82 para el primer y segundo conjunto de *scribbles*, respectivamente.



(a) Random Walks (b) SIOX

Figura 37: Coeficiente Jaccard promedio de *Random Walks* y *SIOX*.

4.1.2.1.3. Random Walks: Para el algoritmo de *Random Walks*, en este experimento se utilizaron valores de β entre 10 y 500, pese a que el autor utiliza un valor $\beta = 900$ en su trabajo [33]. Esto se debe a que en la pruebas realizadas, se encontró que la exactitud es mejor para valores de $\beta < 100$.

Random Walks es evidentemente el algoritmo menos exacto de esta evaluación, según lo mostrado en la Figura 4.37(a). Los mejores valores promedio del coeficiente Jaccard registrados son 0.57 y 0.76 para el primer y segundo conjunto de *scribbles*. Es decir, la exactitud mejora 34 % cuando se utiliza el segundo conjunto de *scribbles*. Para ambos conjuntos, los mejores promedios del coeficiente Jaccard se registran cuando β se sitúa entre 70 y 90.

Es importante notar que, de acuerdo a la evaluación presentada, *Random Walks* es uno de los algoritmos más eficientes, pero sin embargo es el menos eficaz. En efecto, la falta de un proceso de modelamiento de color [33] pone en desventaja a este algoritmo.

4.1.2.1.4. SIOX: Finalmente, en la Figura 4.37(b) se muestran los valores promedio del coeficiente Jaccard para *SIOX*. Este algoritmo utiliza un factor de tamaño, llamado *Size Factor*, para remover áreas desconectadas del objeto de interés. En este experimento se emplearon diferentes valores de *Size Factor*.

Los resultados muestran que *SIOX* puede ser tan exacto como *GrabCut*. Los mejores valores promedio de Jaccard se registran cuando *Size Factor* es 1: 0.66 y 0.84 para el primer y segundo conjunto de *scribbles*, respectivamente. En otras palabras, el coeficiente Jaccard mejora en aproximadamente 27 % cuando se utiliza el segundo conjunto de *scribbles*.

Estas características convierten a *SIOX* en uno de los algoritmos más eficaces que, a su vez, es el más eficiente. Ciertamente, *SIOX* no es tan robusto como *GrabCut*, pero es adecuado para aplicaciones que priorizan la ejecución en tiempo real.

4.1.2.2. Métricas basadas en bordes

En la Figura 38 se muestran las curvas de precisión y exhaustividad de los algoritmos de segmentación interactiva para los dos conjuntos de *scribbles*. Estas curvas ratifican los resultados obtenidos en el análisis de exactitud basado en regiones.

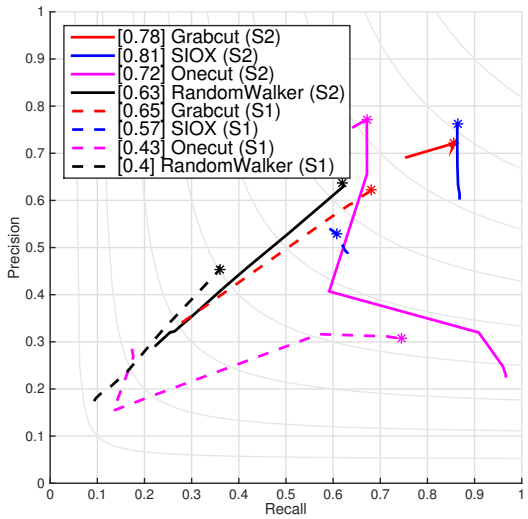


Figura 38: Curvas de precisión y exhaustividad de los algoritmos de segmentación interactiva.

4.1.2.2.1. GrabCut: Las curvas de precisión y exhaustividad muestran que *Grabcut* es el mejor método con un valor F de $F_1 = 0{,}65$ para el segundo conjunto de *scribbles*. Este desempeño mejora linealmente para el segundo conjunto. No obstante, *Grabcut* se ubica aquí en segundo lugar con $F_1 = 0{,}78$.

Pese a esto, el algoritmo de *GrabCut* muestra una compensación proporcional entre precisión y exhaustividad. Es decir, cuando la tasa de precision disminuye, la tasa de exhaustividad aumenta y viceversa.

4.1.2.2.2. OneCut: Las curvas de precisión y exhaustividad muestran la misma fluctuación del desempeño del algoritmo *OneCut* que en el análisis basado en regiones.

Se registra valores de $F_1 = 0{,}43$ y $F_1 = 0{,}72$ para el primer y segundo conjunto de *scribbles*, respectivamente. Estos valoras colocan a *OneCut* en el tercer lugar de

eficacia basada en regiones.

Estas curvas muestran que *OneCut* tiene una alta tasa de exhaustividad cuando la tasa de precisión es baja. Sin embargo, hay casos en los que la eficacia es tan deficiente como en *Random Walks*

4.1.2.2.3. Random Walks: Las curvas de precisión y exhaustividad también muestran que *Random Walks* es el algoritmo menos eficaz. Se registran valores de $F_1 = 0{,}4$ y $F_1 = 0{,}6$ para el primer y segundo conjunto de *scribbles*, respectivamente.

Random Walks muestra la misma compensación proporcional que *GrabCut*: si la tasa de precisión aumenta, entonces la tasa de exhaustividad disminuye.

4.1.2.2.4. SIOX: El algoritmo *SIOX* es casi tan exacto como el algoritmo *GrabCut*. Incluso se puede argumentar que, de acuerdo a las curvas de precisión y exhaustividad, SIOX es más exacto al delimitar los bordes del objeto cuando se utiliza el segundo conjunto de *scribbles*.

Por otro lado, el comportamiento de este algoritmo es diferente al comportamiento del algoritmo *OneCut*. *SIOX* muestra una tendencia de tasas constantes de exhaustividad. Es decir, para una misma tasa de exhaustividad, se puede observar diferentes tasas de precisión. Esta característica es beneficiosa para refinar la calidad de los bordes de los resultados porque permite mejorar su precisión sin perder exhaustividad.

4.1.3. Discusión

Los resultados muestran que la exactitud de los algoritmos incremente significativamente cuando se marca el objeto de interés con mayor detalle. En comparación, no existe un cambio relevante entre los tiempos de ejecución de los algoritmos para los dos conjuntos de *scribbles*.

Esta evaluación indica que *GrabCut* es el método más exacto y robusto. El modelado de color (GMM) utilizado por este algoritmo lo hace menos sensible a las ca-

racterísticas particulares de los *scribbles* que los demás algoritmos. En este sentido, *OneCut* es el algoritmo más sensible no solo a las características de los *scribbles*, sino también a las características de las imágenes y los cambios de parámetros. Ciertamente, *OneCut* consigue resultados tan exactos como los de *GrabCut*, pero la fluctuación de su desempeño lo convierte en un algoritmo poco confiable.

En contraste, *Random Walks* es el algoritmo menos exacto, pero destaca por ser, junto a *SIOX* , uno de los algoritmos más eficientes. En efecto, *SIOX* no solo es el algoritmo más eficiente, sino que logra resultados casi tan exactos como los de *GrabCut*, lo que lo convierte una alternativa efectiva para aplicaciones en las cuales se prioriza el desempeño en tiempo real.

4.2. Restauración de imágenes

4.2.1. Evaluación cualitativa

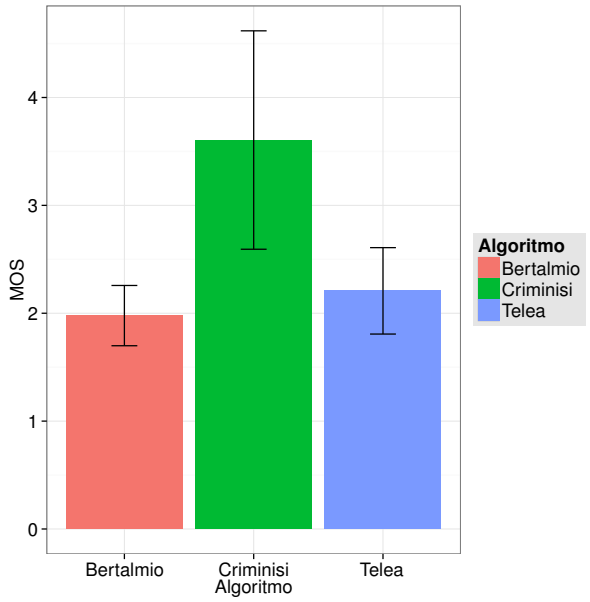


Figura 39: MOS de algoritmos de restauración de imágenes.

Los resultados de la evaluación cualitativa son mostrados en la Figura 39. En la Figura se indican los valores MOS obtenidos para los algoritmos de Bertalmio, Telea y Criminisi.

El algoritmo de restauración propuesto por Criminisi *et al.* [3] recibió las calificaciones más altas de esta evaluación. El MOS de valor 3.61 indica que los participantes perciben las imágenes restauradas con una calidad cercana a *muy buena*. No obstante, la desviación estándar muestra que existieron casos que recibieron calificaciones cercanas a *excelente* y, asimismo, casos que recibieron calificaciones cercanas a calidad *buena*.

El segundo mejor algoritmo, de acuerdo a las calificaciones recibidas, es el algoritmo propuesto por Telea [38]. Sin embargo, en este caso, los participantes califican los resultados como cercano a *regular* ($MOS = 2{,}20$). De igual manera, los resultados del algoritmo propuesto por Telea [38] fueron calificados como *regulares* ($MOS = 1{,}98$) de acuerdo a la percepción de los participantes.

La base de datos utilizada está conformada por imágenes en donde el objeto de interés ocupa un alto porcentaje de la imagen. En consecuencia, es evidente que el algoritmo de Criminisi es el más eficaz para restaurar imágenes en la tarea de extraer estos objetos.

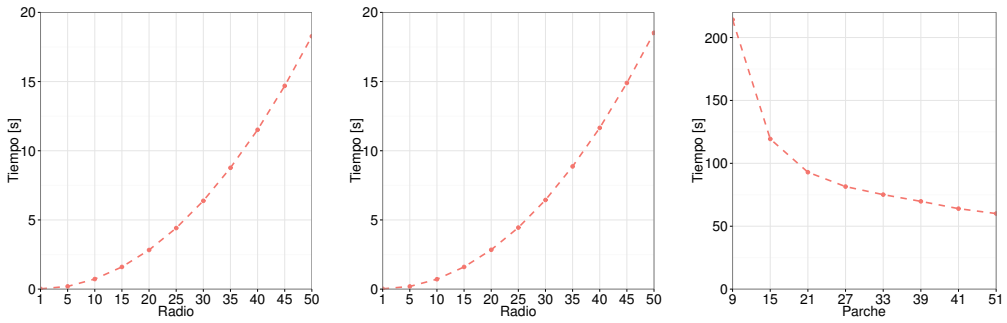
4.2.2. Evaluación cuantitativa

4.2.2.1. Eficiencia

Los algoritmos de Telea y Bertalmio tienen un comportamiento similar, según los gráficos mostrados en las Figuras 4.40(a) y 4.40(b). En ambos casos, el tiempo promedio de ejecución de los algoritmos aumenta conforme se incrementa el radio del vecindario. Este comportamiento se debe a que un radio de vecindad mayor acarrea un mayor número de operaciones porque existen más pixeles.

No obstante, los algoritmos de Telea y Bertalmio son los algoritmos más eficientes

pues el tiempo máximo de ejecución registrado es de 20 segundos. Por otro lado, el algorimo de Criminisi alcanza tiempos promedio de ejecución superiores a los 200 segundos.



(a) Bertalmio (b) Telea (c) Criminisi

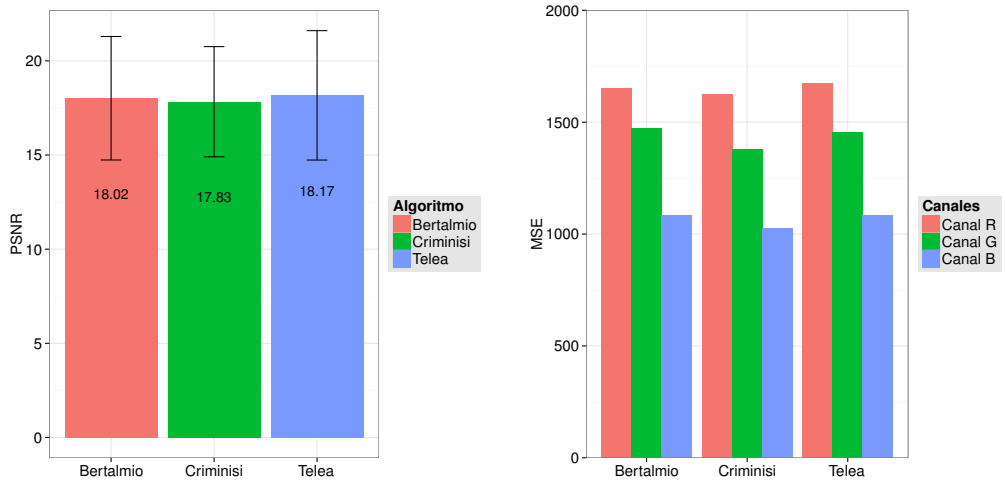
Figura 40: Tiempo promedio de ejecución de los algoritmos de restauración de imágenes.

En contraste, el algoritmo de Criminisi disminuye el tiempo de ejecución según disminuye el tamaño del parche. Esto se debe a que un tamaño de parche menor implica que, al dividir la imagen con ese tamaño, existen más parches como opciones de búsqueda. Por tanto, el algoritmo debe buscar un parche similar entre más parches de igual tamaño.

4.2.2.2. Eficacia

En la Figura 41 se muestran los resultados de la evaluación de algoritmos de restauración utilizando las métricas de calidad de imágenes PSNR y MSE. Para poder comparar con los resultados cualitativos, en la Figura 41 se utilizaron los datos para los siguientes casos:

- **Bertalmio.** Radio del vecindario: 40
- **Telea.** Radio del vecindario: 40
- **Criminisi.** Tamaño del parche: 9



(a) PSNR (b) MSE

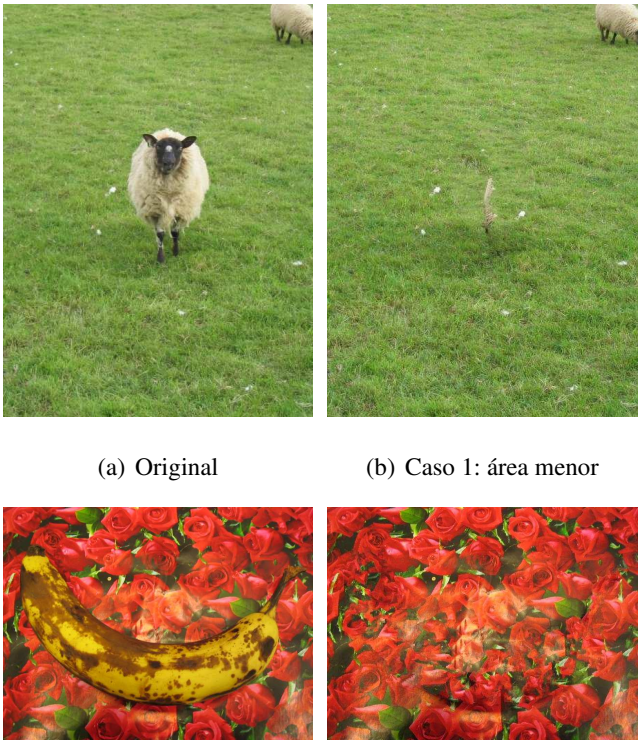
Figura 41: Valores PSR y MSE promedio de imágenes restauradas.

En la Figura 4.41(b) se muestran los valores promedio de MSE para cada canal del modelo de color RGB. Los datos de MSE reportados indican la misma clasificación otorgada por los observadores. Sin embargo, en la evaluación cualitativa, el algoritmo de Criminisi es significativamente superior a Telea. Tanto es así que el MOS de Criminisi supera en un 60 % a Telea. Ciertamente, este comportamiento no es mostrado en MSE pues, según estos valores, Criminisi sólo supera en un 6 % a Telea.

Por este motivo, se puede argumentar que los valores MSE no son muy adecuados para hacer una evaluación comparativa entre algoritmos de restauración de imágenes.

Por otro lado, de acuerdo a los datos de PSNR de la Figura 4.41(a), Criminisi es el algoritmo menos eficaz y, por otro lado, Telea es el algoritmo con mejor calidad de restauración. Por lo tanto, la métrica PSNR no es capaz de reflejar la percepción de los observadores, quienes reportaron que el algoritmo de Criminisi es el más eficaz.

En la Figura 42 se observa el caso de dos imágenes restauradas con el algoritmo de Criminisi. Según la evaluación cuantitativa, la imagen de la Figura 4.42(b) obtuvo 4.524 de MOS promedio y la imagen de la Figura 4.42(d), 4.476. Es decir, para los participantes de la evaluación, las dos imágenes tienen una calidad cercana a excelente.



(a) Original (b) Caso 1: área menor

(c) Original (d) Caso 2: área mayor

Figura 42: Ejemplos de imágenes restauradas con Criminisi ($W = 9$).
Fuente: Imágenes originales de Rother *et al.* [31].

Sin embargo, según las métricas de PSNR, la imagen de la Figura 4.42(b) tiene 24.086 dB y, por otro lado, la imagen de la Figura 4.42(d) tiene 18.163 dB. Es evidente que esta métrica no presenta datos coherentes con los resultados obtenidos en la evaluación cualitativa. Esto sucede porque PSNR utiliza los valores de los pixeles de la imagen restaurada para la comparación, y estos pixeles han cambiado después del proceso de restauración. En definitiva, PSNR no es una métrica apta para clasificar algoritmos de restauración. De hecho, es más sensible al tamaño del objeto que al desempeño de los algoritmos: el valor de PSNR será mayor para las figuras con menor área de restauración.

4.2.3. Discusión

Los resultados de la evaluación muestran que el algoritmo de Criminisi tiene el mejor desempeño. Ciertamente, según los participantes del experimento realizado, la calidad de las imágenes restauradas superan en un 60 % a Telea. En contraste, la eficiencia del algoritmo de Criminisi es baja comparada con los algoritmos de Bertalmio y Telea.

Sin embargo, se debe tener en cuenta que este trabajo utiliza una implementación de Criminisi en MATLAB y, por tanto, el tiempo de ejecución es menor en comparación a una implementación en C/C++. Por ejemplo, existen trabajos de evaluación que reportan el desempeño del algoritmo utilizando implementaciones particulares en C++ [37, 54]; en donde los tiempos de ejecución reportados son menores a 20 segundos.

CAPITULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones y recomendaciones

En este trabajo se ha presentado un estudio comparativo de algoritmos de segmentación y de restauración de imágenes para tareas de eliminación de objetos. Con esta finalidad, en la primera parte se ha evaluado el desempeño de cuatro algoritmos de segmentación interactiva: *GrabCut*, *OneCut*, *Random Walks* y *SIOX*.

Los algoritmos de segmentación interactiva requieren de la guía del usuario para identificar los objetos que serán eliminados de la imagen. Esta intervención mejora el desempeño de los algoritmos, pero dificulta la formulación de una evaluación subjetiva con resultados reproducibles. Por este motivo, en este trabajo se ha recopilado una base de datos para emular la intervención del usuario a través de dos conjuntos de *scribbles* proporcionados para cada imagen.

Esta base de datos incluye una referencia de segmentación, llamada *ground truth*, para evaluar los algoritmos. La exactitud, por tanto, se mide en función de la similitud entre la referencia y los resultados de los algoritmos. En esta evaluación se ha determinado la similitud comparando los bordes y las regiones de manera independiente. Para esto, se determinó que el índice Jaccard refleja más exhaustivamente la exactitud basada en regiones que otras métricas utilizadas en el estado del arte, como el coeficiente Dice o la sensibilidad.

La evaluación basada en bordes y basada en regiones han mostrado que la exactitud de los algoritmos incrementa cuando se marca el objeto de interés con mayor detalle. En comparación, no se ha registrado un cambio relevante entre los tiempos de ejecución de los algoritmos para los dos conjuntos de *scribbles*. Particularmente, en este aspecto, *Random Walks* es el algoritmos más eficiente, pero el menos exacto.

Desde el punto de vista de la eficacia, *GrabCut* se ha erigido como el algoritmo más exacto, seguido de cerca por *SIOX*. De hecho, *SIOX* registra mejor exactitud de bordes cuando se utiliza el segundo conjunto de *scribbles*. Sin embargo, esta exactitud es significativamente inferior en el primer conjunto. Esto muestra que *SIOX* es más sensible a la guía del usuario y, por tanto, menos robusto.

Ciertamente, el objetivo de los dos conjuntos de *scribbles* es definir dos grados de minuciosidad en el señalamiento del objeto de interés: el segundo conjunto señala el objeto con mayor detalle que el primer conjunto de *scribbles*. Los resultados obtenidos han permitido identificar a *GrabCut* como el algoritmo más robusto de esta evaluación porque su desempeño es menos sensible a las características de los *scribbles*. Al contrario, *Random Walks* y *OneCut* no sólo son los algoritmos menos exactos, sino que también los menos robustos, puesto que su rendimiento aumenta significativamente cuando se utiliza el segundo conjunto. En otras palabras, *Random Walks* y *OneCut* demandan una mejor señalización del objeto para alcanzar resultados con segmentaciones aceptables.

En definitiva, *GrabCut* reúne los mejores resultados de desempeño de la evaluación de algoritmos de segmentación interactiva. Este método es el más exacto y el menos sensible a las características de los *scribbles*. Por lo tanto, el algoritmo alcanza la segmentación más exacta sin demandar más esfuerzo del usuario.

En la segunda parte de este trabajo se ha evaluado el desempeño de tres métodos de restauración: los algoritmos de Bertalmio, Telea y Criminisi. Esta evaluación, no obstante, presenta un mayor inconveniente para aplicar criterios objetivos. Esto sucede

porque la definición de una buena restauración depende de la percepción de un observador, quien debe indicar si existen anomalías que evidencien el proceso de retoque de la imagen.

Pese a este inconveniente, diversos trabajo de investigación han utilizado métricas cuantitativas, como MSE y PSNR, para evaluar la calidad de las imágenes restauradas. Sin embargo, la efectividad de estas métricas no han sido validadas por observadores. Por esta razón, en este trabajo se ha incorporado una evaluación cualitativa de los algoritmos.

La evaluación cualitativa muestra que la calidad de los resultados del algoritmo de Criminisi superan significativamente a los resultados de Bertalmio y Telea. Sin embargo, este desempeño no se registra en las métricas cuantitativas. De hecho, sólo MSE muestra que el método de Criminisi es el mejor, pero no es lo suficientemente efectivo para mostrar la significativa superioridad de Criminisi frente a los demás algoritmos de restauración de imágenes.

En contraste, los algoritmos basados en ecuaciones diferenciales parciales, de Bertalmio y Telea, son los más eficientes de esta evaluación. El algoritmo de Criminisi, por su parte, registra tiempos de ejecución significativamente mayores que los otros dos. Pese a esto, hay registros de trabajos relacionados en los cuales se mejora la eficiencia al optimizar la implementación del algoritmo introducido por Criminisi.

Por estas razones, el algoritmo de Criminisi es el más adecuado para complementar el algoritmo *GrabCut* en tareas de eliminación de objetos. En ambos casos, los algoritmos no sólo logran los resultados más precisos, sino que también son los algoritmos más robustos, siendo los menos sensibles a características particulares de las imágenes.

5.2. Trabajos futuros

La evaluación de los algoritmos de segmentación interactiva puede ser extendida a través de un estudio más exhaustivo de las condiciones de segmentación. Es posible modificar el experimento, aquí presentado, para añadir ruido a las imágenes segmentadas. Esta modificación permitiría identificar el desempeño de los algoritmos en función del ruido de las imágenes.

Por otro lado, es evidente que no hay un criterio generalizado para realizar una evaluación cuantitativa de los algoritmos de restauración. Además, las evaluaciones cuantitativas realizadas en trabajos anteriores no tienen la capacidad de reflejar de manera exhaustiva el desempeño de los algoritmos.

Por estos motivos, es necesario reformular los criterios de evaluación utilizados en trabajos anteriores. Es posible que sea necesario incorporar los mismos criterios utilizados en los algoritmos de restauración. Por consiguiente, trabajos futuros deben explorar las posibles alternativas a métricas basadas en la calidad de la imagen. Es importante, además, que estas alternativas ratifiquen su validez a través de una evaluación cualitativa.

BIBLIOGRAFÍA

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 417–424.

[2] K. s Mahajan and M. Vaidya, "Image in painting techniques: A survey," *IOSR Journal of Computer Engineering (IOSRJCE) ISSN: 2278-0661, ISBN: 2278-8727 Vol*, vol. 5, pp. 45–49, 2012.

[3] A. Criminisi, P. Perez, and K. Toyama, "Object removal by exemplar-based inpainting," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 2. IEEE, 2003, pp. II–721.

[4] F. Yi and I. Moon, "Image segmentation: A survey of graph-cut methods," in *Systems and Informatics (ICSAI), 2012 International Conference on*. IEEE, 2012, pp. 1936–1941.

[5] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 105–112.

[6] B. Peng, L. Zhang, and D. Zhang, "A survey of graph theoretical approaches to image segmentation," *Pattern Recognition*, vol. 46, no. 3, pp. 1020–1038, 2013.

[7] T. Acharya and A. Ray, *Image Processing: Principles and Applications*. Wiley, 2005. [Online]. Available: https://books.google.com.ec/books?id=smBw4-xvfrIC

[8] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. New Jersey: Prentice Hall, 2002.

[9] S. Annadurai, *Fundamentals of Digital Image Processing*. Pearson, 2007. [Online]. Available: https://books.google.com.ec/books?id=wM8QRLJXJFwC

[10] W. Burger and M. Burge, *Digital Image Processing: An Algorithmic Introduction Using Java*, ser. Texts in Computer Science. Springer London, 2009. [Online]. Available: https://books.google.com.ec/books?id=4gBUz_IkkSsC

[11] A. McAndrew, "An introduction to digital image processing with matlab notes for scm2511iimage processing," *School of Computer Science and Mathematics, Victoria University of Technology*, pp. 1–264, 2004.

[12] T. Acharya and A. K. Ray, *Image Processing: Principles and Applications*. John Wiley & Sons, 2005.

[13] A. Mehtar, "Ambient-light sensing optimizes visibility and battery life of portable displays," Maxim Integrated, Tech. Rep., 06 2011. [Online]. Available: http://www.maximintegrated.com/en/app-notes/index.mvp/id/5051

[14] A. Koschan and M. Abidi, *Digital Color Image Processing*. John Wiley & Sons, 2008.

[15] J. C. Russ, *The Image Processing Handbook*, 6th ed. Boca Raton, Florida: Taylor & Francis Group, 2011.

[16] R. M. Haralick and L. G. Shapiro, "Image segmentation techniques," in *1985*

Technical Symposium East. International Society for Optics and Photonics, 1985, pp. 2–9.

[17] C. Demant, C. Demant, and B. Streicher-Abel, *Industrial Image Processing*. Springer, 1999.

[18] W. E. Snyder and H. Qi, *Machine Vision*. Cambridge University Press, 2010.

[19] G. Dougherty, *Digital Image Processing for Medical Applications*. Cambridge University Press, 2009.

[20] C. Solomon and T. Breckon, *Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab*. John Wiley & Sons, 2011.

[21] T. Ridler and S. Calvard, "Picture thresholding using an iterative selection method," *IEEE transactions on Systems, Man and Cybernetics*, vol. 8, no. 8, pp. 630–632, 1978.

[22] G. Zack, W. Rogers, and S. Latt, "Automatic measurement of sister chromatid exchange frequency." *Journal of Histochemistry & Cytochemistry*, vol. 25, no. 7, pp. 741–753, 1977.

[23] Q. Wu, F. Merchant, and K. Castleman, *Microscope Image Processing*. Academic Press, 2010.

[24] T. R. Reed, *Digital Image Sequence Processing, Compression, and Analysis*. CRC Press, 2004.

[25] W. K. Pratt, *Digital Image Processing*. John Wiley & Sons, 2007.

[26] C. Fernandez-Maloigne, *Advanced Color Image Processing and Analysis*. Springer Science & Business Media, 2012.

[27] W. A. Barrett and E. N. Mortensen, "Interactive live-wire boundary extraction," *Medical image analysis*, vol. 1, no. 4, pp. 331–341, 1997.

[28] S. Chen, H. Tong, and C. Cattani, "Markov models for image labeling," *Mathematical Problems in Engineering*, vol. 2012, 2011.

[29] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 9, pp. 1124–1137, 2004.

[30] R. Radke, *Computer Vision for Visual Effects*, ser. Computer Vision for Visual Effects. Cambridge University Press, 2012.

[31] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004.

[32] M. Tang, L. Gorelick, O. Veksler, and Y. Boykov, "Grabcut in one cut," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1769–1776.

[33] L. Grady, "Random walks for image segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 11, pp. 1768–1783, 2006.

[34] G. Friedland, K. Jantz, and R. Rojas, "Siox: Simple interactive object extraction in still images," in *Multimedia, Seventh IEEE International Symposium on*. IEEE, 2005, pp. 7–pp.

[35] Museo Nacional del Prado, "The restoration of The Purification of the Virgin in the Temple by Pedro de Campaña," 2010. [Online]. Available: https://www.museodelprado.es/en/research/restoration-and-research/the-restoration-of-emthe-purification-of-the-virgin-in-the-templeem/

[36] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–355.

[37] A. I. Oncu, "Digital inpainting for artwork restoration: Algorithms and evaluation," Master's thesis, Gjøvik University College, Gjøvik, Norway, 2012.

[38] A. Telea, "An image inpainting technique based on the fast marching method," *Journal of graphics tools*, vol. 9, no. 1, pp. 23–34, 2004.

[39] K. McGuinness and N. E. O'Connor, "A comparative evaluation of interactive segmentation algorithms," *Pattern Recognition*, vol. 43, no. 2, pp. 434–444, 2010.

[40] ——, "Toward automated evaluation of interactive segmentation," *Computer Vision and Image Understanding*, vol. 115, no. 6, pp. 868–884, 2011.

[41] H. Zhang, J. E. Fritts, and S. A. Goldman, "Image segmentation evaluation: A survey of unsupervised methods," *computer vision and image understanding*, vol. 110, no. 2, pp. 260–280, 2008.

[42] J. Pont Tuset, "Image segmentation evaluation and its application to object detection," Ph.D. dissertation, Universitat Politècnica de Catalunya, Barcelona, España, 2014.

[43] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, may 2011.

[44] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman, "Geodesic star convexity for interactive image segmentation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3129–3136.

[45] S. Alpert, M. Galun, A. Brandt, and R. Basri, "Image segmentation by probabilistic bottom-up aggregation and cue integration," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 2, pp. 315–327, 2012.

[46] J. Wu, Y. Zhao, J.-Y. Zhu, S. Luo, and Z. Tu, "Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 256–263.

[47] D. M. Powers, "Evaluation: From precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[48] J. Pont-Tuset and F. Marques, "Measures and meta-measures for the supervised evaluation of image segmentation," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 2131–2138.

[49] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 5, pp. 530–549, 2004.

[50] A. I. Oncu, F. Deger, and J. Y. Hardeberg, "Evaluation of digital inpainting quality in the context of artwork restoration," in *Computer Vision–ECCV 2012. Workshops and Demonstrations*. Springer, 2012, pp. 561–570.

[51] Vision Imaging Science and Technology Lab, "Object removal," 2013. [Online]. Available: http://white.stanford.edu/teach/index.php/ObjectRemoval

[52] M. M. O. B. B. Richard and M. Y.-S. Chang, "Fast digital image inpainting," in *Appeared in the Proceedings of the International Conference on Visualization, Imaging and Image Processing (VIIP 2001), Marbella, Spain*, 2001, pp. 106–107.

[53] A. Ninassi, O. Le Meur, P. Le Callet, and D. Barba, "On the performance of human visual system based image quality assessment metric using wavelet domain,"

in *SPIE Conference Human Vision and Electronic Imaging XIII*, vol. 6806, 2008, pp. 680 610–1.

[54] M. Daisy, D. Tschumperlé, and O. Lézoray, "A fast spatial patch blending algorithm for artefact reduction in pattern-based image inpainting," in *SIGGRAPH Asia 2013 Technical Briefs*. ACM, 2013, p. 8.