Universidad Autónoma de Chihuahua
Facultad de Ingeniería
Secretaría de Investigación y Posgrado

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

Reconstrucción de imágenes por medio de autómatas celulares

POR:
Ing. Eliana López Ortega

TESIS PRESENTADA COMO REQUISITO PARA OBTENER EL GRADO DE
Maestría en Ciencias Básicas

Director:
Dr. José Luis Herrera Aguilar
M. en C. Ana Virginia Contreras García

CHIHUAHUA, CHIH. MÉXICO
2019

UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

Reconstrucción de imágenes por medio de autómatas celulares. Tesis presentada por *Ing. Eliana López Ortega* como requisito parcial para obtener el grado de *Maestría en Ciencias Básicas*, ha sido aprobada y aceptada por:

M. I. Javier González Cantú
Director de la Facultad de Ingeniería

Dr. Alejandro Villalobos Aragón
Secretario de Investigación y posgrado

M. I. Luis Carlos Quiñónez Baca
Coordinador Académico

Dr. José Luis Herrera Aguilar
Director de tesis

Febrero del 2019

Comité:
Dr. José Luis Herrera Aguilar
M. en C. Ana Virginia Contreras
Dra. Haydey Álvarez Allende
Dr. Octavio Raúl Hinojosa de la Garza

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

11 de Marzo de 2019

ING. ELIANA LÓPEZ ORTEGA

Presente

En atención a su solicitud relativa a la Tesis para obtener el grado de Maestría en Ciencias Básicas, nos es grato transcribirle el tema aprobado por esta Dirección, propuesto y dirigido por el director **Dr. José Luis Herrera Aguilar** para que lo desarrolle Tesis, con el título: **"RECONSTRUCCIÓN DE IMÁGENES POR MEDIO DE AUTÓMATAS CELULARES".**

ÍNDICE

FACULTAD DE INGENIERÍA
Circuito No.1, Campus Universitario 2
Chihuahua, Chih., México. C.P. 31125
Tel. (614) 442-95-00
www.fing.uach.mx

UNIVERSIDAD AUTÓNOMA DE
CHIHUAHUA

Capítulo 7. Conclusiones

Anexos

Referencias

Curriculum Vitae

Solicitamos a Usted tomar nota de que el título del trabajo se imprima en lugar visible de los ejemplares del estudio de caso.

A T E N T A M E N T E
"Naturam subiecit aliis"

EL DIRECTOR

M.I. JAVIER GONZÁLEZ CANTÚ

FACULTAD DE INGENIERÍA U.A.CH. DIRECCIÓN

EL SECRETARIO DE INVESTIGACIÓN Y POSGRADO

DR. ALEJANDRO VILLALOBOS ARAGÓN

FACULTAD DE INGENIERÍA
Circuito No.1, Campus Universitario 2
Chihuahua, Chih., México. C.P. 31125
Tel. (614) 442-95-00
www.fing.uach.mx

Dedicatoria

A mis padres.

Agradecimientos

Agradezco a Dios por su infinita misericordia al darme la oportunidad de llegar a este punto de mi formación profesional, por la vida y por las personas con las que he tenido el placer de trabajar y convivir a lo largo de este proyecto de investigación.

Agradezco al Dr. José Luis Herrera Aguilar y M. en C. Ana V. Contreras Gracía por el apoyo brindado, su generosidad, compromiso y paciencia, y sobre todo por confiar en mi y mis habilidades al permitirme formar parte de su equipo de trabajo, les estoy eternamente agradecida, sin ustedes esto no sería posible.

Con todo el amor agradezco a mis padres y hermanas por apoyarme en cada uno de mis proyectos, ustedes son la principal motivación que me empuja a cumplir mis propósitos.

A mi esposo, Iván Ugarte por infundirme aliento y darme su apoyo.

Resumen

En este proyecto se propone un nuevo método para redimensionar imágenes que se encuentran tanto en escala de grises como en color, ya que muchos de los métodos existentes, como las interpolaciones, solo comparan los pixeles adyacentes; 4 píxeles en el caso de la bilineal, 16 en la interpolación bicúbica y en el caso de la interpolación spline bilineal o spline bicúbica realiza el procesamiento de forma similar tomando los pixeles vecinos al punto de interpolación deseado, solo que usa funciones para suavizarlo y definirlo; también podemos mencionar el área de interpolación que solo se basa en interpolar al vecino más cercano y el método orientado a los bordes que solo busca mejorar las áreas que no son uniformes, pero para estos métodos el tiempo de procesado y la calidad de las imágenes no son siempre las deseadas.

El método aqui presentado puede ser útil cuando el tiempo y calidad sean factores importantes en el procesado de la imagen, ya que se basa en la preservación de los bordes, haciendo particiones de imágenes digitales. Para ello emplea autómatas celulares que dividen la imagen permitiendo trabajar en áreas pequeñas lo cual preserva la calidad de los bordes de un pixel desconocido en base a la dirección de los bordes de sus vecinos, e ir uniendo cada partición hasta obtener la imagen completa mejorada. Debido a las técnicas empleadas para este método los resultados de procesar una imagen hace posible ampliar la imagen sin que se pierda calidad, además el tiempo de procesado es corto.

Índice general

Índice de figuras

CAPÍTULO: 1

Introducción

La reconstrucción de imágenes es comúnmente usada para el mejoramiento de la calidad de fotografías o imágenes en particular y para ello se usan una gran cantidad de métodos, sin embargo, no siempre se obtienen los resultados deseados. Esto se debe a que los métodos existentes no satisfacen las características para el redimensionado de imágenes, como conservar la calidad y nítidez,además, el proceso toma un largo tiempo, es decir, un alto costo computacional.

Entre los métodos más usados está la interpolación, anteriormente solo se usaba la interpolación lineal en la cuál se tomaba en cuenta solo cuatro áreas adyacentes al área a mejorar. Si bien se conseguía mejorar la imagen con este método, aún seguía con una resolución no tan buena, así que se podía buscar otro método más eficiente usando la interpolación. Se buscó hacer una interpolación "adaptativa" de imágenes, basado en características del gradiente local para obtener mejores resultados. Para ello dentro de la imagen se tomó en cuenta la posición de los pixeles y se hizó subdivisiones y se adaptó la interpolación bilineal común a una interpolación bilineal adaptativa, es decir, tomando en cuenta el gradiente local del área a mejorar.

Puesto que al usar el método de interpolación bilinear adaptativa obtuvó mejores resultados, se adaptó para la interpolación bicúbica. En la interpolación bicúbica se toman 16 áreas adyacentes al área a mejorar, si bien la interpolación bicúbica obtiene mejores resultados que la interpolación bilinear, se buscó mejorar el proceso basándose en el gradiente local, de esta forma se obtuvó la

interpolación bicúbica adaptativa. Entonces esta al ser una mejora de la interpolación bicúbica regular, obtuvó mejores resultados. Jung Woo Hwang y Hwang Soo Lee en el artículo " Adaptative Image Interpolation based on local gradient Features" (Hwang y Lee, 2004) explican y determinan la función para 2D de la interpolación bicúbica adaptativa.

De acuerdo a los métodos anteriormente mencionados se notó que se obtienen mejores resultados realizando más divisiones a la imagen y aún mejores cuando se toma en cuenta los gradientes de las áreas seleccionadas, entonces se pensó en otro método que cumpliera con ambas características y se propuso usar **autómatas celulares**.

Para este nuevo método basado en autómatas celulares se requiere dividir en pequeños cuadros (áreas) toda la imagen e ir analizando cuadro por cuadro. El gradiente del área indica si el cuadro o subsección seleccionada es un trozo que tiene cambios o bien no es uniforme, a esto le llamaremos área de borde; si el área seleccionada es uniforme le llamaremos área homogénea. Si el gradiente del área seleccionada es mayor que el área de la subsección entonces será un área de borde, si el gradiente es menor entonces será un área homogénea.

El método propuesto consiste en localizar todas las áreas de borde, que serán las áreas a trabajar. En cada área de borde se tendrá que aplicar nuevamente, es decir, dividir en cuadros más pequeños y verificar cuales de estos son áreas de borde o áreas homogéneas. Las áreas de borde son las que se busca mejorar o dicho de otra forma, se buscan convertir en áreas homogéneas, para ello se basa en sus áreas adyacentes, quienes determinarán como será esa nueva área homogénea, por lo tanto es importante que las áreas de borde seleccionadas sean subsecciones tan pequeñas como sea posible para obtener mejores resultados. Cuando se obtiene la subárea mínima de borde se compara con todos sus adyacentes y se toma en cuenta la cantidad de área mayor igual a uno de sus adyacentes y se convierte así en un área homogénea.

El proceso subdivide las áreas al mínimo, comienza a convertir las áreas de borde a áreas homogéneas y se va alejando y comparando nuevamente la imagen, dando como resultado una imagen de alta resolución.

CAPÍTULO: 2

Planteamiento del problema

Actualmente el uso de las cámaras es exhaustivo y en algunos casos de vital importancia, como lo es para las cámaras de seguridad de establecimientos en bancos ó negocios que se encuentran en funcionamiento las 24 horas del día, y cuyas imágenes son de baja resolución, es decir, mala calidad y en ocaciones poco legibles. Otro tipo de cámaras que generan imágenes con mala resolución son las satelitales, estas cámaras además de estar orbitando el planeta deben estar hechas de materiales resistentes a las condiciones de la exosfera, los cuales en ocasiones pueden interferir con la resolución de la cámara; algunas otras profesiones, como los fotógrafos quienes usan por largos periodos sus cámaras, en estos casos es indispensable contar con una cámara que pueda captar imágenes de buena calidad y que no se pierdan o deformen al ampliarlas.

El solo hecho de ampliar una imagen con los métodos tradicionales implica una perdida de información en la imagen ampliada, ya que se añaden píxeles nuevos que originalmente no existían en la imagen y se rellenan con un color base ya sea blanco o negro, debido a que se introduce un nuevo color hace que la imagen pierda su calidad, se obscurezca o se ilumine; o bien en lugar de añadir nuevos píxeles aumenta el tamaño de cada píxel en particular, esto no introduce nuevos colores pero hace notar los píxeles en la imagen, es decir, no es una imagen suave, lo que comúnmente conocemos como imagen pixeleada, o con efecto cuadrícula, en ambos casos existe un aumento en el tamaño de la imagen que ocasiona una perdida de calidad, debido a esto es necesario la

modificación de los métodos existentes para obtener imágenes nítidas y que sea posible ampliarlas sin perder detalles en la imagen, esto además puede reducir el costo del cambio de cámaras de mayor resolución, en el caso de los mapas poder delimitar entre edificios y calles, además de poder obtener una fotografía fiel de los delincuentes que buscan en bancos y establecimientos en base a una fotografía de baja resolución, ya que por lo general no se puede apreciar de forma clara el rostro de la persona.

Existen algunos otros procesos para mejorar imágenes, pero además de no dar los resultados deseados, conllevan un largo procesamiento en la máquina usada, es decir, toman un largo tiempo de procesado y un alto costo computacional. El método propuesto está basado en Autómatas Celulares (A.C.), permite hacer particiones y trabajar en pequeñas partes, lo anterior tiene un menor costo de procesamiento y consecuentemente un menor tiempo de computo, dando como resultado la obtinción de imágenes de alta calidad.

CAPÍTULO: 3

Hipótesis

Por medio de algoritmos de reconstrución y procesamiento de imágenes a través de Autómatas Celulares, realizar el redimensionado de imágenes sin perder la calidad de la misma, además de obtener imágenes de alta resolución a partir de imágenes de baja resolución; optimizar el tiempo de procesamiento y tal vez sea posible disminuir el espacio ocupado en la máquina.

CAPÍTULO: 4

Objetivos

4.1. Objetivo General

Obtener imágenes nítidas a partir de imágenes de baja resolución usando reconstrucción de imágenes por medio de autómatas celulares.

4.2. Objetivo Particular

Implementar al menos un método de reconstrucción de imágenes por medio de autómatas celulares que genere imágenes de calidad.

CAPÍTULO: 5

Marco Teórico

A pesar de existir algoritmos de procesamiento de imágenes, muchos de estos procesos no son usados debido a que sus resultados son pobres en calidad o de muy poca mejoría en comparación con la imagen original o bien los procesos para lograr una ampliación de una fotografía (permitiendo que esta conserve su calidad) son tardados (Ioannidis, Andreadis, y Sirakoulis, 2012) , por lo tanto nos dimos a la tarea de investigar un algoritmo de reconstrucción de imágenes, que por medio de autómatas celulares pretende realizar ampliaciones de fotografías sin que estas pierdan su calidad o de ser posible que se mejore la resolución.

Para crear el algoritmo de reconstrucción de imagenes por medio de autómatas celulares nos basamos en un artículo titulado *An Edge Preserving Image Resizing Method Based on Cellular Automata* (Ioannidis y cols., 2012) , en el cuál describe el proceso de imágenes para obtener ampliaciones conservando la calidad.

Existen algoritmos para el procesamiento de imágenes de los cuales, los más usados para procesamiento de imágenes son las interpolaciones, de ellas existen diferentes métodos (Hwang y Lee, 2004):

- Interpolación bilineal.
- Interpolación bicúbica.
- Áreas de interpolación.

- Orientado a los bordes.

Para darnos una idea de lo que se pretende realizar, daremos una explicación de qué es lo que hacen cada uno de estos métodos.

La interpolación bilineal (Hwang y Lee, 2004) al igual que la mayoría de los métodos se basa inicialmente en los píxeles conocidos o bien los píxeles originales de la imagen tomando un cuadro de 2x2 píxeles más cercanos, en base a esos 4 píxeles el algoritmo determinará cuál será el color del nuevo píxel, tomando el promedio de los 4 píxeles originales (cuadro de 2x2) y se calcula su valor interpolado, los resultados en cuanto resolución no son tan buenos, a menos que sea una imágen con pocos colores ya que solo toma, para generar el nuevo color, cuatro de los píxeles adyacentes, además de necesitar más tiempo de procesado.

La interpolación bicúbica (Keys, 1981) es muy similar a la interpolación bilineal, realiza casi el mismo proceso, toma un cuadro de 4x4 píxeles originales más cercanos, es decir, en lugar de 4 píxeles hace el promedio de 16 píxeles más cercanos, sin embargo, como no todos los píxeles están a la misma distancia, se les dará mayor valor o peso a aquellos que estén más cercanos, el resultado es una imágen más nítida, su relación entre calidad y tiempo de procesado es considerable. En esté tipo de interpolación, pese a que su tiempo de procesado es solo un poco mayor que en la interpolación bilineal, su calidad es mucho mejor; también los resultados en cuanto a calidad son mejores con este método que con el método de vecinos más cercanos (Ioannidis y cols., 2012), el cuál se explica a continuación.

El método de áreas de interpolaciones también es conocido como el método del vecino más cercano. Debido a que es un método muy sencillo requiere poco tiempo de procesado, sin embargo los resultados obtenidos no son de una calidad deseada (Ioannidis y cols., 2012). Lo que hace este método es que solo toma en cuenta un único píxel, que será el que esté más cercano al punto de interpolado, básicamente lo que hace este algoritmo es incrementar el tamaño del píxel por eso su tiempo de procesado es corto, sin embargo, se obtiene una versión pixelada de la imagen. En la figura 5.1 podemos ver cómo la imagen ampliada con el algoritmo resultó tener un efecto cuadricula.

De acuerdo con el artículo (Ioannidis, Andreadis, y Sirakoulis, s.f.) el cuál propone realizar un método que además de aplicar los autómatas celulares también hace uso de el método de detección de bordes, se pretende que el resultado sea una ampliación de buena calidad.

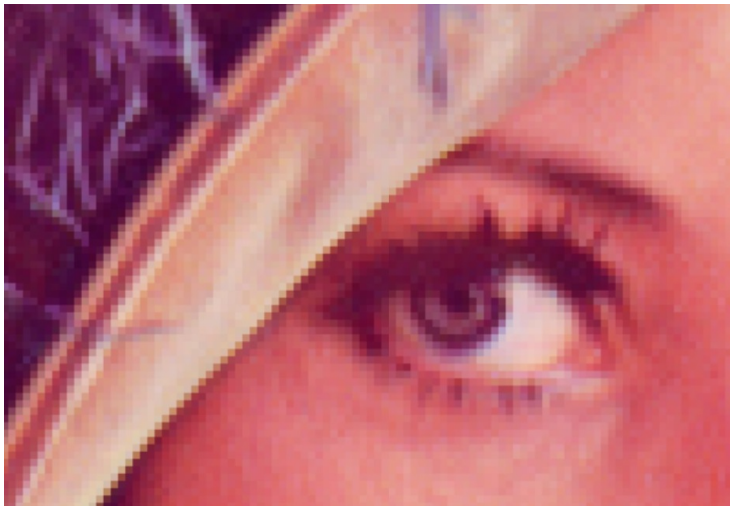

Figura 5.1: Vecino más cercano.
Esta figura muestra una imagen ampliada por el método de áreas de interpolación también conocido como el método del vecino más cercano; se puede apreciar la imagen con efecto cuadrícula ya que este método toma solo un vecino, es decir, solo hace más grande el píxel original

Se considera que al realizar la implementación de estos algoritmos combinados de reconstrucción y procesamiento de imágenes, se mejore la calidad de la imagen procesada, que nos permitirá realizar el redimensionado de imágenes sin perder la calidad o de ser posible mejorar la resolución. Debido a que los autómatas celulares trabajan tomando un problema (en este caso nuestra imagen a tratar) en el cual, dividen en pequeñas partes en nuestro caso a píxeles y resuelven el problema en cada parte para luego nuevamente unir la imagen y obtener una mejora, una de las características principales de los autómatas es que reducen el tiempo de procesado, por lo tanto, se considera que al unir los métodos además de reducir el tiempo de procesamiento se obtendrá una buena imagen ampliada sin perder colores o calidad.

Como definición un Autómata se refiere a una máquina, dispositivo, instrumento o mecanismo que obedece un grupo de reglas que sigue un mecanismo automático de operaciones para procesar información (Sayama, 2015). En este estudio se hace el análisis usando Autómatas Celulares, los cuales realizan el procesamiento de datos de entrada de forma celular, es decir, divide el problema en su fracción mínima para realizar una concatenación de los datos de forma

continua y realizar las operaciones necesarias aptas para el procesado de datos, en nuestro caso particular imágenes.

Debido a que la parte más pequeña de las imágenes es el píxel, este será la célula a trabajar por el autómata, tomando estos como los datos de entrada para lograr como resultado el redimensionado de una imagen completa. La vecindad de la célula está dada por los píxeles vecinos y la frontera está determinada por el tamaño de la imagen, es decir, por la cantidad total de píxeles.

Antes de detallar cada uno de los pasos de este proceso de reconstrucción se debe preparar la imagen para su procesado, se requiere asegurar que nuestra imagen cumpla con ciertos requisitos, esto es para que cualquier tipo de imagen pueda ser procesada por medio de este algoritmo. Para el procesado de la imagen se requiere que la estructura del color de cada píxel esté representada en formato **RGB**, que determina el color en base a la intensidad de cada color primario de la luz, rojo, verde, azul, (RGB, por sus siglas en inglés: Red=Rojo, Green=Verde, Blue=Azul) 5.2, por lo tanto cada píxel tiene un vector de longitud 3, uno para cada color cuyos valores representan la intensidad de dicho color.

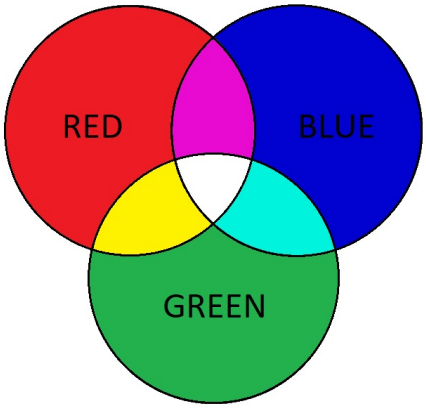

Figura 5.2: Composición de colores en RGB.
En la figura se muestra la composición de colores con base en los tres colores primarios de la luz: Rojo (Red), Verde (Green) y Azul (blue); RGB por sus siglas en inglés.

En la figura 5.3 seleccionamos un píxel cuyo color está representado por el vector RGB, donde el rojo tiene una intensidad de 91, el verde de 58 y el azul de 41; la combinación de la intensidad de colores nos da da el color resultante.

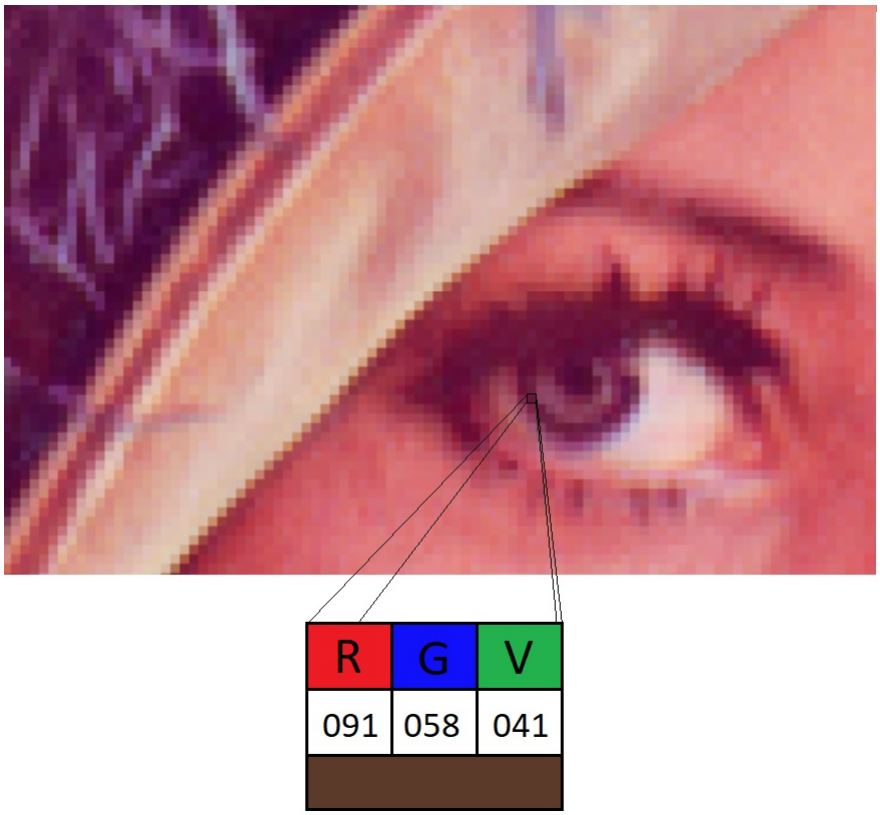

Figura 5.3: Vector RGB.

Cada píxel de la imagen está conformado por un vector de longitud 3 y este asigna el color del píxel, donde el vector corresponde al modelo de color RGB y cada uno corresponde a la intensidad que tendrá de cada color, siendo el primer componente del vector el color rojo, el segundo corresponde al color verde y el último al color azul.

Es importante destacar que en cada color su intensidad puede ir desde 0 (cero) hasta 255, donde el 0 es la ausencia del color en cuestión y 255 la máxima intensidad o saturación del color. En la imagen 5.4 se da un ejemplo de distintos colores con su representación en RGB y además su equivalente en hexadécimal.

255, 255, 255 #FFFFFF	245, 228, 228 #F5E4E4	251, 243, 230 #FBF3E6	255, 251, 214 #FFFBD6	224, 239, 215 #E0EFD7	217, 240, 246 #D9F0F6	249, 231, 241 #F9E7F1
210, 211, 213 #D2D3D5	239, 195, 195 #EFC3C3	255, 235, 205 #FFEBCD	245, 244, 193 #F5F4C1	212, 232, 195 #D4E8C3	201, 233, 242 #C9E0F2	243, 211, 229 #F3D3E5
189, 191, 193 #BDBFC1	242, 179, 179 #F2B3B3	250, 255, 184 #FAE1B8	250, 246, 190 #FAF6BE	197, 224, 170 #C5E0AA	186, 228, 241 #BAE4F1	238, 195, 219 #EEC3DB
169, 171, 174 #A9ABAE	241, 147, 147 #F19393	255, 220, 168 #FCDCA8	244, 241, 158 #F4F19E	180, 216, 147 #B4D893	164, 221, 238 #A4DDEE	225, 166, 203 #E1A6CB
150, 152, 154 #96989A	237, 116, 118 #ED7476	253, 211, 146 #FDD392	245, 239, 138 #F5EF8A	166, 209, 117 #A6D175	153, 217, 235 #99D9EB	220, 147, 191 #DC93BF
132, 134, 136 #848688	237, 98, 98 #ED6262	254, 204, 124 #FECC7C	241, 236, 114 #F1EC72	156, 204, 95 #9CCC5F	133, 211, 232 #85D3E8	211, 128, 180 #D380B4
114, 115, 118 #727376	230, 73, 73 #E64949	251, 191, 95 #FBBF5F	248, 238, 101 #F8EE65	143, 199, 74 #8FC74A	120, 208, 232 #78D0E8	205, 108, 170 #CD6CAA
96, 96, 98 #606062	237, 50, 55 #EB3237	251, 179, 63 #FBB33F	248, 237, 80 #F8ED50	123, 194, 77 #7BC24D	106, 204, 230 #6ACCE6	199, 91, 161 #C75BA1
75, 75, 77 #4B4B4D	235, 51, 55 #EB3337	250, 169, 49 #FAA931	249, 237, 59 #F9ED3B	118, 192, 78 #76C04E	96, 202, 230 #60CAE6	196, 79, 156 #C44F9C
0, 0, 0 #000000	237, 50, 55 #ED3237	248, 156, 50 #F89C32	248, 236, 34 #F8EC22	111, 190, 79 #6FBE4F	12, 180, 214 #0CB4D6	188, 67, 151 #BC4397

Figura 5.4: Colores con su etiqueta RGB y Hexadecimal.
La imagen muestra algunos colores con su valor en RGB y su equivalente en Hexadecimal.

5.1. Detector de Bordes de Canny

El detector de bordes de Canny lo que hace es diferenciar las áreas homogéneas de aquellas que no lo son. Las áreas homogéneas son aquellas que sus vecinos no presentan cambios de color o bien son muy similares; las áreas de borde o no homogéneas serán aquellas cuyos vecinos sean de un color muy diferente, los bordes los podemos interpretar como la delimitación de los objetos. En la imagen 5.5 se representan los píxeles ampliados en una imagen, los círculos negros representan los colores que son similares entre sí u homogénos, los círculos blancos representan los píxeles donde hay cambios significativos de color, píxeles o áreas de borde y el último círculo nos muestra los bordes definidos ampliados.

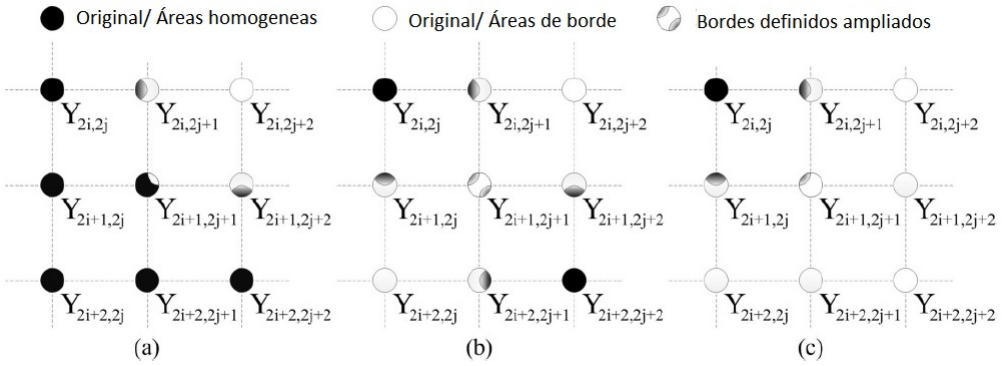

Figura 5.5: Detección de Bordes.

La imagen representa los píxeles ampliados en una imagen, los círculos negros representan los colores que son similares entre si o homogénos, los círculos blancos representan los píxeles donde hay cambios significativos de color, píxeles o áreas de borde y el último círculo nos muestra los bordes definidos ampliados.

El detector de bordes de Canny es usado por sus características especificas, ya que asegura una correcta detección, es decir, promete encontrar cada borde si es que existe en la imagen original; localización precisa, los bordes detectados están donde hay cambios de intensidades en la imagen o bien muy cerca de ellos y una respuesta mínima de bordes, esto quiere decir que no genera bordes adicionales a los que existen además de marcar cada borde sólo una vez.

El detector comienza filtrando la imagen mediante un filtro Gaussiano de 2D con desviación estandar predefinida en 0.35, esto con el fin de eliminar el ruido. Para el filtrado de la imagen se usa la siguiente máscara de 3x3:

$$x = \begin{vmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{vmatrix} \qquad y = \begin{vmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{vmatrix}$$

Esta es una máscara predefinida para obtener los bordes de una imagen. Para aplicar la máscara se realiza la convolución de la máscara con la imagen original, la convolución de matrices en imágenes es para aplicar filtros, como se muestra en la imagen 5.6, donde la matriz de 3x3 representa nuestra máscara y la matriz de la derecha la imagen a tratar, se aplica la convolución con nuestra matriz x de 3x3 a la imagen original y de la misma forma con la matriz y de 3x3.

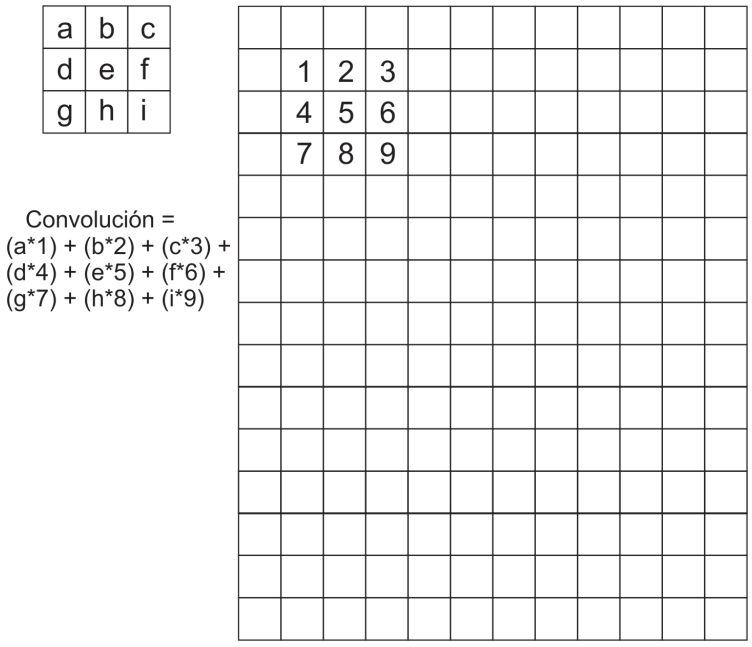

Figura 5.6: Matriz de Convolución.

La matriz de 3x3 representa la máscara a aplicar y la matriz de la derecha representa la imagen a tratar; para realizar la convolución de las matrices se determina que porcentaje de color de sus vecinos se le debe asignar al píxel tratado, desarrollando las multiplicaciones para obtener el porcentaje y sumando estos para obtener el color.

Es importante recordar que cada píxel de la imagen es un vector de longitud 3, y para poder realizar la convolución necesitamos convertir este valor a decimal, dado que la máscara esta en valores decimales para poder tratar con ella la imagen deberá estar en decimales, como se muestra en la imagen 5.7

Figura 5.7: Conversión REG a Decimal.
Para obtener el color en decimal, el color en RGB debe convertirse como se puestra en la figura, la suma de estos será el mismo color representado en decimal.

Una vez realizada la convolución, se aplicará la siguiente función, con la cuál se determina por medio de un condicional si es un borde o no.

$$g(x, y) = \sqrt{x^2 + y^2}$$

De esta función se obtiene un resultado en decimal, para poder imprimir nuevamente la imagen ya con bordes debemos tomar el valor entero que corresponde al color del píxel del decimal generado y este convertirlo a su equivalente en RGB (véase las tablas 5.1 y 5.2). Una vez que se tiene cada píxel nuevamente en RGB, se aplica la condicional, si el valor es mayor a 135 el color será negro, de lo contrario el color debe ser blanco, dando como resultado una imagen con sus bordes en color blanco y los colores homogénos en color negro.

Decimal a Hexadecimal							
Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex
0	0	32	20	54	40	96	60
1	1	33	21	65	41	97	61
2	2	34	22	66	42	98	62
3	3	35	23	67	43	99	63
4	4	36	24	68	44	100	64
5	5	37	25	69	45	101	65
6	6	38	26	70	46	102	66
7	7	39	27	71	47	103	67
8	8	40	28	72	48	104	68
9	9	41	29	73	49	105	69
10	A	42	2A	74	4A	106	6A
11	B	43	2B	75	4B	107	6B
12	C	44	2C	76	4C	108	6C
13	D	45	2D	77	4D	109	6D
14	E	46	2E	78	4E	110	6E
15	F	47	2F	79	4F	111	6F
16	10	48	30	80	50	112	70
17	11	49	31	81	51	113	71
18	12	50	32	82	52	114	72
19	13	51	33	83	53	115	73
20	14	52	34	84	54	116	74
21	15	53	35	85	55	117	75
22	16	54	36	86	56	118	76
23	17	55	37	87	57	119	77
24	18	56	38	88	58	120	78
25	19	57	39	89	59	121	79
26	1A	58	3A	90	5A	122	7A
27	1B	59	3B	91	5B	123	7B
28	1C	60	3C	92	5C	124	7C
29	1D	61	3D	93	5D	125	7D
30	1E	62	3E	94	5E	126	7E
31	1F	63	3F	95	5F	127	7F

Cuadro 5.1: Equivalente de RGB a Hexadecimal. Parte 1.

Esta tabla contiene todos los posibles valores de los componentes en color RGB (de 0 a 255) y cual es su correspondencia en Hexadecimal.

Decimal a Hexadecimal							
Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex
128	80	160	A0	192	C0	224	E0
129	81	161	A1	193	C1	225	E1
130	82	162	A2	194	C2	226	E2
131	83	163	A3	195	C3	227	E3
132	84	164	A4	196	C4	228	E4
133	85	165	A5	197	C5	229	E5
134	86	166	A6	198	C6	230	E6
135	87	167	A7	199	C7	231	E7
136	88	168	A8	200	C8	232	E8
137	89	169	A9	201	C9	233	E9
138	8A	170	AA	202	CA	234	EA
139	8B	171	AB	203	CB	235	EB
140	8C	172	AC	204	CC	236	EC
141	8D	173	AD	205	CD	237	ED
142	8E	174	AE	206	CE	238	EE
143	8F	175	AF	207	CF	239	EF
144	90	176	B0	208	D0	240	F0
145	91	177	B1	209	D1	241	F1
146	92	178	B2	210	D2	242	F2
147	93	179	B3	211	D3	243	F3
148	94	180	B4	212	D4	244	F4
149	95	181	B5	213	D5	245	F5
150	96	182	B6	214	D6	246	F6
151	97	183	B7	215	D7	247	F7
152	98	184	B8	216	D8	248	F8
153	99	185	B9	217	D9	249	F9
154	9A	186	BA	218	DA	250	FA
155	9B	187	BB	219	DB	251	FB
156	9C	188	BC	220	DC	252	FC
157	9D	189	BD	221	DD	253	FD
158	9E	190	BE	222	DE	254	FE
159	9F	191	BF	223	DF	255	FF

Cuadro 5.2: Equivalente de RGB a Hexadecimal. Parte 2.

Esta tabla contiene todos los posibles valores de los componentes en color RGB (de 0 a 255) y cual es su correspondencia en Hexadecimal.

CAPÍTULO: 6

Metodología

Debido a que nuestro objetivo es el redimensionado de imágenes sin perder la calidad de la misma, es decir, obtener una imagen de mayor tamaño y a su vez de más pixelaje, en el cuál los nuevos píxeles contendrán colores que estén dados en base a los colores respectivos de la imagen original y de tal forma que sea una imagen suavizada para que no sean visibles los cambios de color de píxel a píxel a menos que éste sea un píxel de borde. Para ello, debemos conocer en la imagen cuales píxeles son de borde y cuales no. Los píxeles de borde son aquellos cuyos vecinos son sumamente distintos a ellos, es decir, debe haber un cambio notable de color entre píxeles vecinos. A los píxeles que no son de borde les llamaremos píxeles homogéneos, que son aquellos cuyos vecinos o bien son iguales, o de colores similares.

Para las pruebas en el redimensionado de imágenes se usó principalmente la imagen de una joven sueca de nombre Lena Söderberg, cuya fotografía llamada *Lena* o *Lenna* es del contenido de un artículo de 1972 de la revista Playboy. Esta fotografía es relevante ya que ha sido usada para el procesamiento de imágenes, debido a sus texturas, detalles y colores.

6.1. Pasos para el redimensionado de imágenes por medio de autómatas celulares.

Antes de describir el desarrollo de forma particular de cada uno de los pasos, damos un listado de los pasos a seguir para procesamiento de imágenes usando autómatas celulares.

1. Verificar formato de color en la imagen a tratar.
2. Convertir a Blanco-Negro (Detección de bordes).
3. Ampliación.
4. Rellenar imagen.
5. Colorear.

Previo a asignar nuevamente el color a la imagen, debemos asegurarnos que en la imagen original cada píxel tenga un vector de longitud 3 que corresponde al color en RGB. Cada valor define la cantidad de rojo, verde y azul que define a ese píxel, pero algunas imágenes además de contener los tres valores contienen información adicional la cual no es útil y debemos eliminar para no afectar el color original al momento de reasignar colores en la imagen escalada, para ello usamos el programa *rgb.py*, el cuál arroja la imagen *rgb.png* que es una copia de la original en la cual nos aseguramos que cada píxel tenga un vector con únicamente tres elementos, los cuales corresponden al color en RGB, asegurándonos que no tenga información adicional que modifique el color original y nos permita procesar la imagen. La imagen obtenida de este proceso será necesaria para colorear la nueva imagen aumentada de acuerdo a los bordes, y garantiza que se conservan los colores originales en los píxeles originales.

Para que la asignación de colores sea más eficaz, buscaremos los bordes en la imágen, por medio de un algoritmo al que llamaremos *bordes.py*, el cuál obtiene los bordes de la imagen deseada basado en los píxeles homogéneos y diferenciándolos de los que no lo son, obteniendo una imagen del mismo tamaño que la imagen a original pero en blanco y negro. El color negro representa los píxeles homogéneos y el color blanco estará en aquellos píxeles que sean detectados como píxeles de borde, es decir, píxeles que no tienen un color definido o están

posicionados donde existe un cambio de color en la imagen. Para poder utilizar este programa se necesita la imagen original *lena.png*. Dando como resultado la imagen *bordes.png*.

Un vez detectados los bordes de la imagen se procede a hacer la ampliación de la imagen, esto se realiza en la imagen donde se tienen los bordes, para ello usamos un sencillo programa al que llamamos *resizing.py*, éste hace un aumento de la imagen obtenida del programa de *bordes.py*, añadiendo filas y columnas adicionales entre cada par añadido de píxeles original de la imagen con bordes, obteniendo un mallado de doble tamaño, donde todos los píxeles con coordenadas par-par contenga los puntos de la imagen obtenida de la imagen *bordes.png* (imagen obtenida del programa *bordes.py*) y los píxeles restantes son píxeles vacíos, este procesamiento arroja una imagen de doble tamaño y la guarda en *resizing.png*, la cual también es una imagen a blanco y negro, que muestra la imagen con bordes con huecos en cada fila y columna impar.

Hasta este punto, se tiene una imagen del mismo tamaño que nuestra imagen original solo que esta nos muestra los bordes de la imagen original, sobre esta imagen procederemos a realizar la ampliación de la imagen, para ello se introduce una nueva columna de píxeles entre cada columna de píxeles originales existentes, y también una nueva fila de píxeles entre cada fila original existente de píxeles. En 6.1 podemos ver una ampliación donde se muestran 4 píxeles originales de color negro y en color blanco se muestran las filas y columnas de los nuevos píxeles añadidos.

De este paso se obtiene una imagen del doble de tamaño, es decir, si la imagen original es de 512x512 píxeles, la nueva imagen será de 1024x1024 píxeles. Además de ser de doble tamaño, esta imagen también tenemos los bordes, solo que en cada columna y fila añadida tenemos un píxel vacío o lo que es lo mismo, un píxel sin un color.

En este paso lo que hacemos es rellenar la imagen, definir cada nuevo píxel añadido en base a los píxeles vecinos originales. Para los píxeles con coordenadas par-impar se toma a sus vecinos horizontales, del vecino izquierdo al nuevo píxel se le asignará el 25 % y de su vecino derecho tomará el 75 %. Para los píxeles con coordenadas impar-par se toma a sus vecinos verticales, es decir, del vecino que está encima del nuevo píxel tomará el 25 % y de su vecino inferior tomará el 75 %. Para los píxeles con coordenadas impar-impar se toma a sus cuatro vecinos más cercanos que son sus vecinos diagonales, de los cuales tomará el 30 % de su vecinos inferiores y derechos y el 10 % de su vecino que se posiciona a la

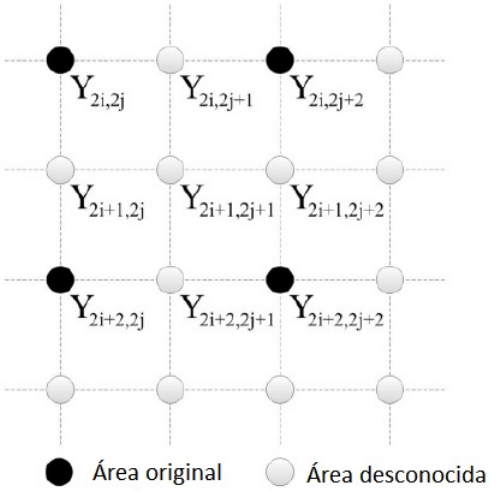

Figura 6.1: Píxeles añadidos en Resizing.
La imagen muestra en color negro a los píxeles originales y de color blanco los píxeles nuevos de las columnas y filas añadidas.

izquierda y arriba.

Para reasignar el color usamos el programa *color.py* que asigna los colores de la imagen *rem.png* que obtuvimos del programa *rgb.py*, la cual es una copia de la imagen original pero solo contiene en cada píxel el vector de color RGB correspondiente, a la nueva imagen escalada con bordes definidos llamada *resizing.png* obtenida del programa *resizing.py*, pasando cada color de la imagen original a su correspondiente posición en la nueva imagen, que básicamente coloca los colores de la imagen original en los píxeles con coordenada par-par de la imagen del *resizing.png*, obteniendo una copia de la imagen original ampliada con un mallado negro a rellenar de acuerdo a los colores originales. Este programa guarda la imagen *colorear.png*.

En base a la imagen aumentada al doble de su tamaño, con colores de la imagen original en los píxeles par-par y con los píxeles de las columnas y filas impar vacíos, el programa *remapping.py* rellena los píxeles vacíos en base a sus vecinos tomándolos de la siguientes manera:

Para los píxeles con coordenadas par-impar se toma a sus vecinos horizontales,

es decir, su vecino izquierdo y derecho. Al nuevo pixel se le asignará el 25 % del vecino izquierdo y de su vecino derecho tomará el 75 %.

Para los píxeles con coordenadas impar-par se toma a sus vecinos verticales, es decir, su vecino de arriba y abajo, del vecino que está encima del nuevo pixel tomará el 25 % y de su vecino inferior tomará el 75 %.

Para los píxeles con coordenadas impar-impar se toma a sus cuatro vecinos más cercanos que son sus vecinos diagonales, de los cuales tomará el 30 % de su vecinos inferiores y derechos y el 10 % de su vecino que se posiciona a la izquierda y arriba. De este programa obtenemos la imagen más definida y a todos los píxeles se le ha asignado un color, esta imagen es del doble tamaño de la imagen original.

CAPÍTULO: 7

Resultados

En este capítulo se describen los resultados del algoritmo de reconstrucción de imágenes, el cuál propone que por medio del uso de automátas celulares dismuinuirá considerablemente el tiempo de procesado en relación a la calidad de la imagen resultante procesada, tomando en cuenta que la resolución de la imagen no se perderá al aumentar su tamaño cualquier cantidad de veces, en comparación con otro tipo de algoritmos similares de reconstrucción de imágenes (Ioannidis y cols., 2012) cuyos resultados son imágenes de baja calidad.

Resultados del detector de bordes de Canny

Además de diferenciar áreas homogéneas de aquellas que no lo son, uno de los resultados adicionales al filtrar nuestra imagen fue que pudimos encontrar tanto imágenes que han sido cortadas y editadas, o bien aquellas que se realizaron por capas, además de ser sensible a los cambios de intensidad graduales, delimitaciones de objetos con fondos de colores similares y sombras, en la imagen 7.1 podemos ver un ejemplo de lo que realiza el detector de bordes, y nos muestra lo sensible que es, se pueden apreciar los bordes de las hojas, se aprecia el reflejo de la luz sobre la mesa. A pesar de que es un cambio drástico de color no lo detecta como borde, puesto que este se va difuminando del centro a las orillas, por lo tanto en cada par de vecinos no existe un cambio violento de color si no que es gradual, también se puede observar que el borde del cuaderno se pierde en

el reflejo de la luz, puesto que el algoritmo los toma como colores similares por lo tanto no detecta el borde. Otro ejemplo es la sombra de las hojas sobre otras hojas marcan como si existiera otra hoja adicional ya que el color de la sombra y la hoja son distintos, éste marca el borde.

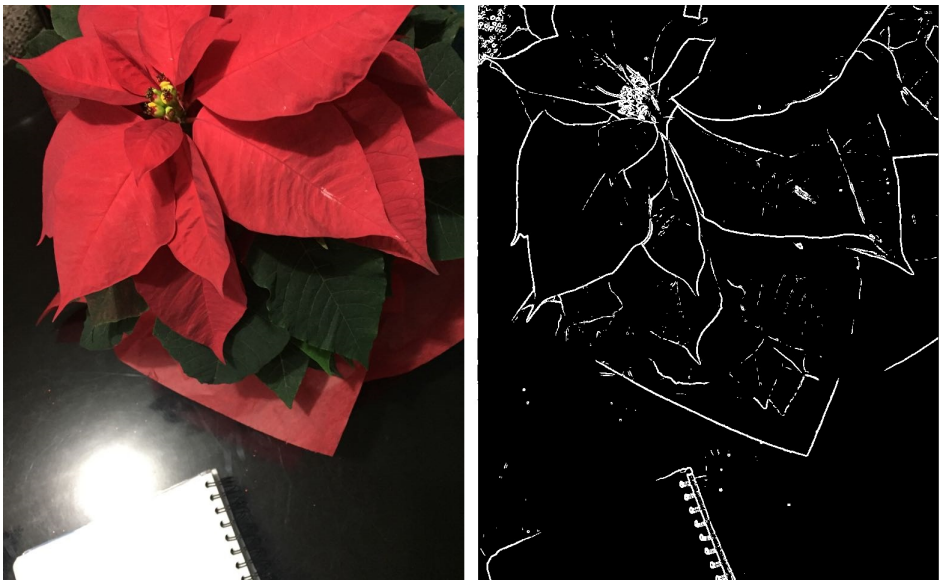

Figura 7.1: Bordes en una Imagen.
La imagen muestra los resultados del detector de bordes de Canny, mostrando delimitaciones de objetos y bordes en las sombras

Para probar nuestro detector de bordes en nuestras prácticas descargamos dos logotipos de la facultad de ingeniería uno de ellos desde la página oficial de la facultad. En la figura 7.2 tenemos el logotipo descargado para el uso de documentos de la facultad; al pasarlo por el detector de bordes podemos observar que la imagen es una sola pieza y detecta todos los bordes de forma correcta y precisa.

En la figura 7.3 podemos ver el logotipo de la página de ingeniería que se ve igual al anterior, pero al pasar el detector de bordes pudimos encontrar como si la imagen estuvierá trunca. Esto se debe a que la imagen es una transparencia y está realizada por capas, por lo cuál el detector de bordes solo nos muestra la primera capa que hace notar el recorte de las demás capas.

Figura 7.2: Logotipo Ingeniería y bordes del logotipo.
La figura muestra del lado izquierdo la imagen original del logotipo de la Facultad de Ingeniería de la U.A.Ch. y del lado derecho se muestra la imagen del detector de bordes, donde claramente muestra los limites entre un color y otro.

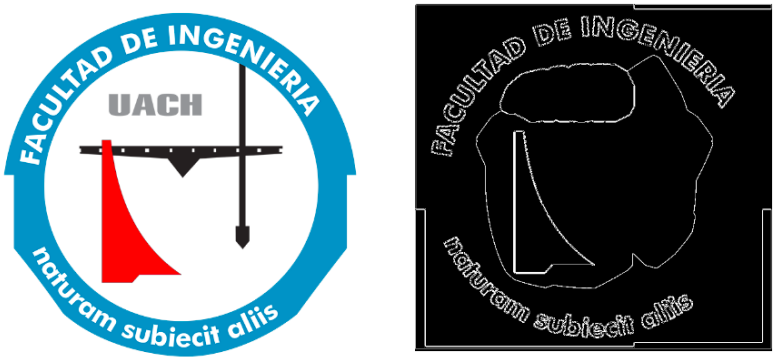

Figura 7.3: Logotipo Ingeniería escalable y borde del logotipo.
La figura de la izquierda se muestra el logotipo oficial para documentos, descargable desde la página de la Facultad de Ingeniería de la U.A.Ch., a la derecha la imagen procesada por el detector de bordes de Canny. Esta imagen muestra como la imagen original ha sido truncada o editada, esto es debido a que la imagen original es una transparencia y el detector se corre sobre la primer capa de la imagen tomando las demás capas como un corte o edición.

7.1. Ordenamiento de los algoritmos

Puesto que en los algoritmos que trabajan con autómatas celulares es importante el orden (Sayama, 2015), cambiamos el ordenamiento del algoritmo para ver los resultados obtenidos. Basados en el artículo "An Edge Preserving Image Resizing Method Based on Cellular Automata" (Ioannidis y cols., s.f.) el ordenamiento de procesamiento debe ser el siguiente:

- Filtrado de la imagen.
- Obtención de bordes.
- Resizing.
- Remmaping.
- Color.

En la figura 7.4 se presenta el diagrama de flujo para este ordenamiento.

La imagen a procesar se muestra en la figura 7.5, siendo esta la imagen original de tamaño 512x512 píxeles. Basándose en el ordenamiento anterior descrito (Ioannidis y cols., 2012) los resultados obtenidos son los que se presentan a continuación. En la imagen 7.6 tenemos el primer filtro del procesado de la imagen, una imagen del mismo tamaño que la original, 512x512 píxeles, donde se muestran los bordes de la imagen, es decir, donde existen áreas no homogeneas.

Una vez obtenidos los bordes siguiendo este ordenamiento, el algoritmo realiza el Resizing, que es en sí el cambio de tamaño de la imagen, para este caso es la ampliación al doble de su dimensión original. Este proceso lo que hace es crear una fila de columnas y filas adicionales entre las filas y columnas de píxeles originales de la imagen. En la figura 7.7 tenemos el resultado del Resizing, donde la imagen tiene doble tamaño y con filas y columnas vacías, para poder apreciar mejor en la imagen se muestra un aumentado de 400 % de una parte de la imagen.

De la figura 7.7 se tiene una imagen del doble de tamaño que la imagen original, pero esta imagen solo se presenta los bordes y las áreas homogéneas con columnas y filas vacías, ahora se tiene que rellenar todos los nuevos píxeles vacíos, para ella usamos el proceso de Remapping, donde se rellenan los nuevos píxeles, para los píxeles con coordenadas par-impar se toma a sus vecinos horizontales

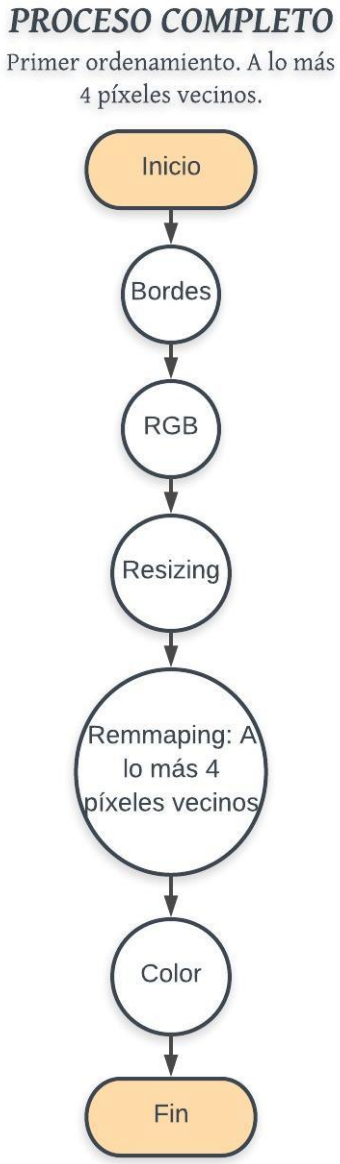

Figura 7.4: Diagrama de flujo. Proceso completo, primer ordenamiento. A lo más cuatro píxeles vecinos.

Proceso completo para realizar el redimensionado de la imagen usando el primer ordenamiento con el método de a lo más cuatro píxeles vecinos.

Figura 7.5: Imagen Lena Original

Imagen de prueba Lena a procesar en su estado original de 512x512 píxeles.

Figura 7.6: Bordes Imagen Lena.

Resultados obtenidos una vez procesada la imagen por el detector de bordes de Canny de tamaño 512x512

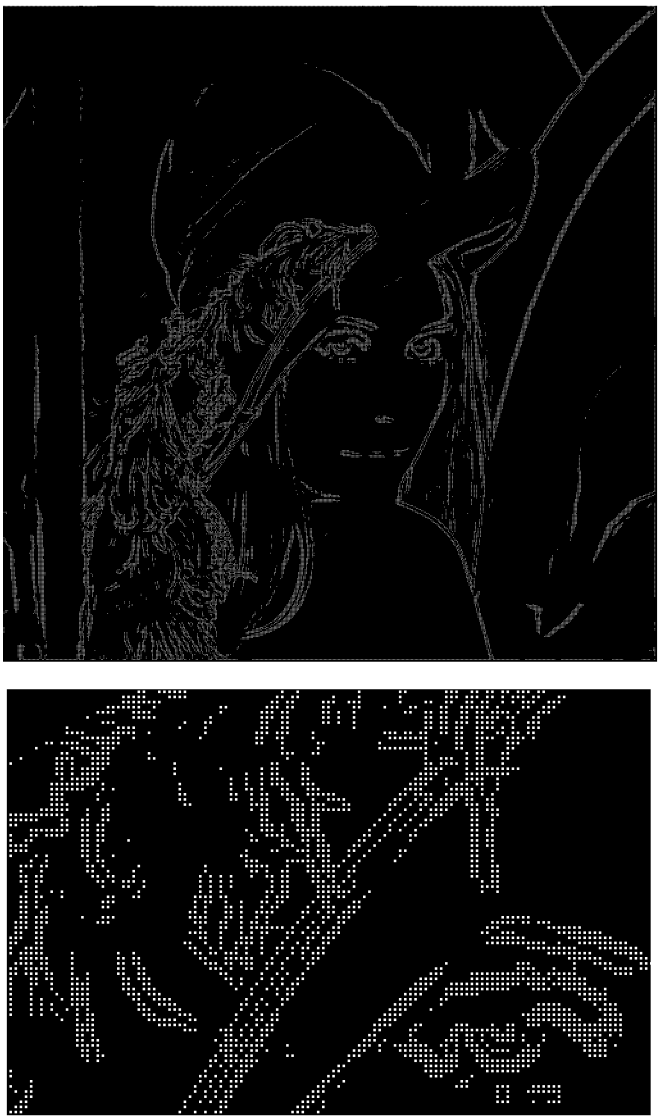

Figura 7.7: Resizing.

Esta imagen es del doble tamaño que la imagen donde se obtienen los bordes siendo de 1024x1024, donde también se muestra la imagen con bordes a blanco y negro, añadido filas y columnas nuevas entres cada fila y columna original, estos nuevos pixeles añadidos se verán de color negro, ya que no tienen un color especifico asignado. En la parte inferior se presenta una parte de la imagen al 400 % donde se pueden apreciar los huecos de color negro entre cad para de píxeles de borde (píxeles de color blanco).

asignando el 25 % del vecino izquierdo y de su vecino derecho tomará el 75 %; para los píxeles con coordenadas impar-par se toma a sus vecinos verticales, es decir su vecino de arriba y abajo, del vecino que está encima del nuevo pixel tomará el 25 % y de su vecino inferior tomará el 75 %; para los píxeles con coordenadas impar-impar se toma a sus cuatro vecinos más cercanos que son sus vecinos diagonales asignando el 10 % de su vecino superior izquierdo y 30 % de cada uno de los restantes.

Una vez realizado el Remapping se asignan los colores a la imagen ampliada en base a los colores originales. Para todos los píxeles originales se les asigna su color y en base al color de los píxeles originales obtendremos el color de los nuevos píxeles añadidos.

En la figura 7.8 tenemos el resultado final del proceso, donde la imagen no pierde colores pero sí se enfoca a un punto específico se puede apreciar que la imagen no está suavizada y presenta efecto cuadrícula, como se muestra en la parte inferior de la figura, se puede distinguir de forma más clara entre el área del sombrero y el ojo .

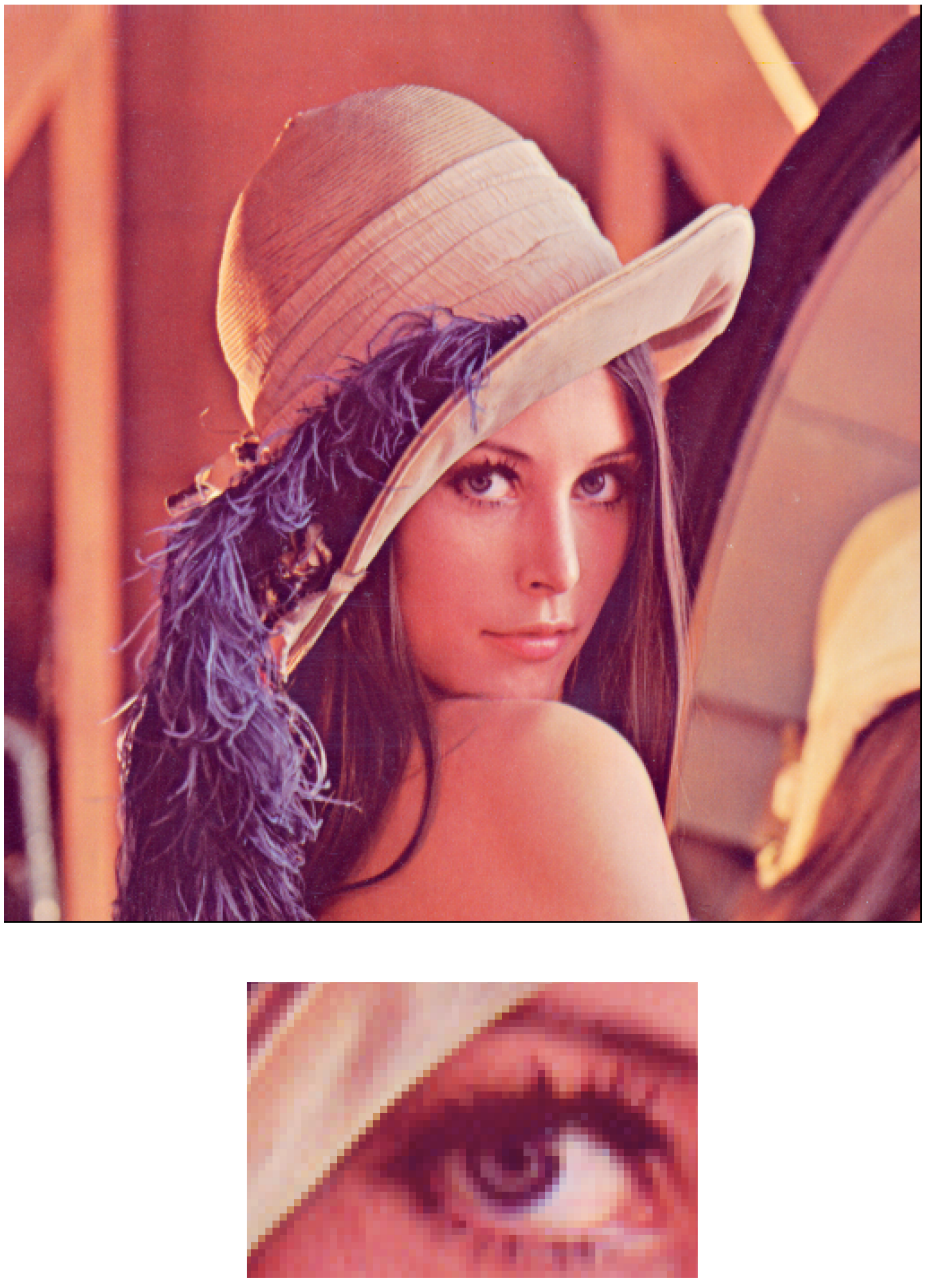

Figura 7.8: Imagen Lena ampliada.

La imagen superior es de tamaño 1024x1024, procesada siguiendo el orden de procesado: 1. Filtrado de la imagen. 2. Obtención de bordes. 3. Resizing. 4. Remmaping. 5. Color. En la parte inferior tenemos una porción ampliada al 200 % donde se puede apreciar los detalles y efecto cuadrícula de la imagen.

Dados los resultados obtenidos, las mejores imágenes en cuanto a calidad, difuminación, menor efecto cuadrícula se obtienen con el siguiente orden:

- Filtrado de la imagen.
- Obtención de bordes.
- Resizing.
- Color.
- Remmaping.

En la figura 7.9 podemos ver el diagrama de flujo de este procesamiento.

Una vez ejecutado el Resizing, se le asignan los colores a los píxeles originales dándonos como resultado una imagen con color, la cuál tiene filas y columnas de píxeles sin un color asignado entre cada píxel original, en la imagen 7.10 podemos apreciar la imagen del doble de tamaño con colores en los píxeles originales pero con filas y columnas sin color asignado.

Una vez asignados los colores a los píxeles originales se procede a realizar el Remapping, de la misma forma que anteriormente, asignando a los píxeles vacíos con coordenadas par-impar sus vecinos horizontales, el 25 % del vecino izquierdo y el 75 % de su vecino derecho, a los píxeles vacíos con coordenadas impar-par sus vecinos verticales, 25 % del vecino superior y 75 % de su vecino inferior. A los píxeles vacíos con coordenadas impar-impar con base en sus cuatro vecinos más cercanos, un 10 % de su vecino superior izquierdo y 30 % de cada uno de los restantes.

De lo anterior obtenemos una imagen del doble de tamaño, de 1024x1024 píxeles sin perder la calidad y los colores de la imagen original 7.11. Si realizamos un enfoque a la imagen podemos notar que el efecto cuadrícula se ha suavizado en general en la imagen, como se muestra en imagen inferior de la figura 7.11.

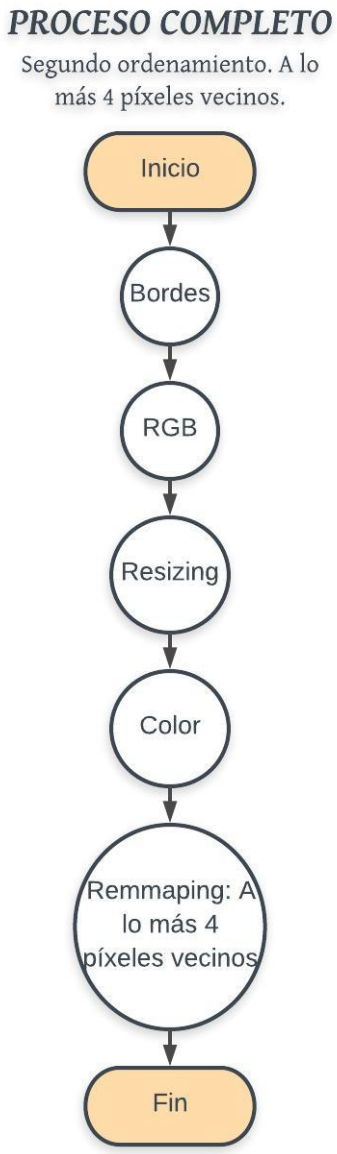

Figura 7.9: Diagrama de flujo. Proceso completo, segundo ordenamiento. A lo más cuatro píxeles vecinos.

Proceso completo para realizar el redimensionado de la imagen usando el segundo ordenamiento con el método de a lo más cuatro píxeles vecinos.

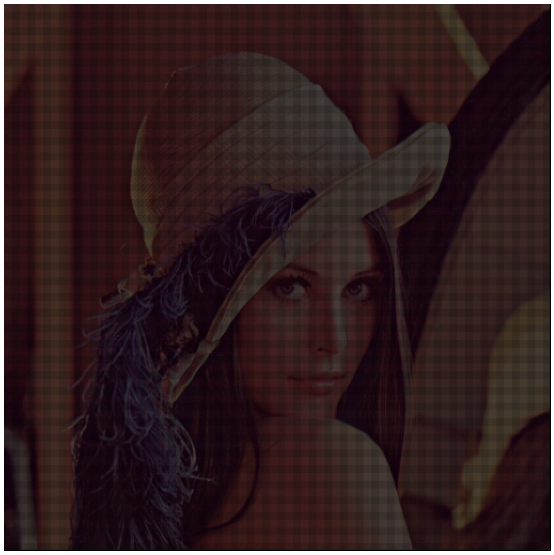

Figura 7.10: Imagen Lena ampliada con color asignado.
La imagen es de tamaño 1024x1024 donde se han asignado los colores originales a los píxeles originales, dando como resultado una imagen a color con cada fila y columna par vacía, es decir, sin un color asignado o negro.

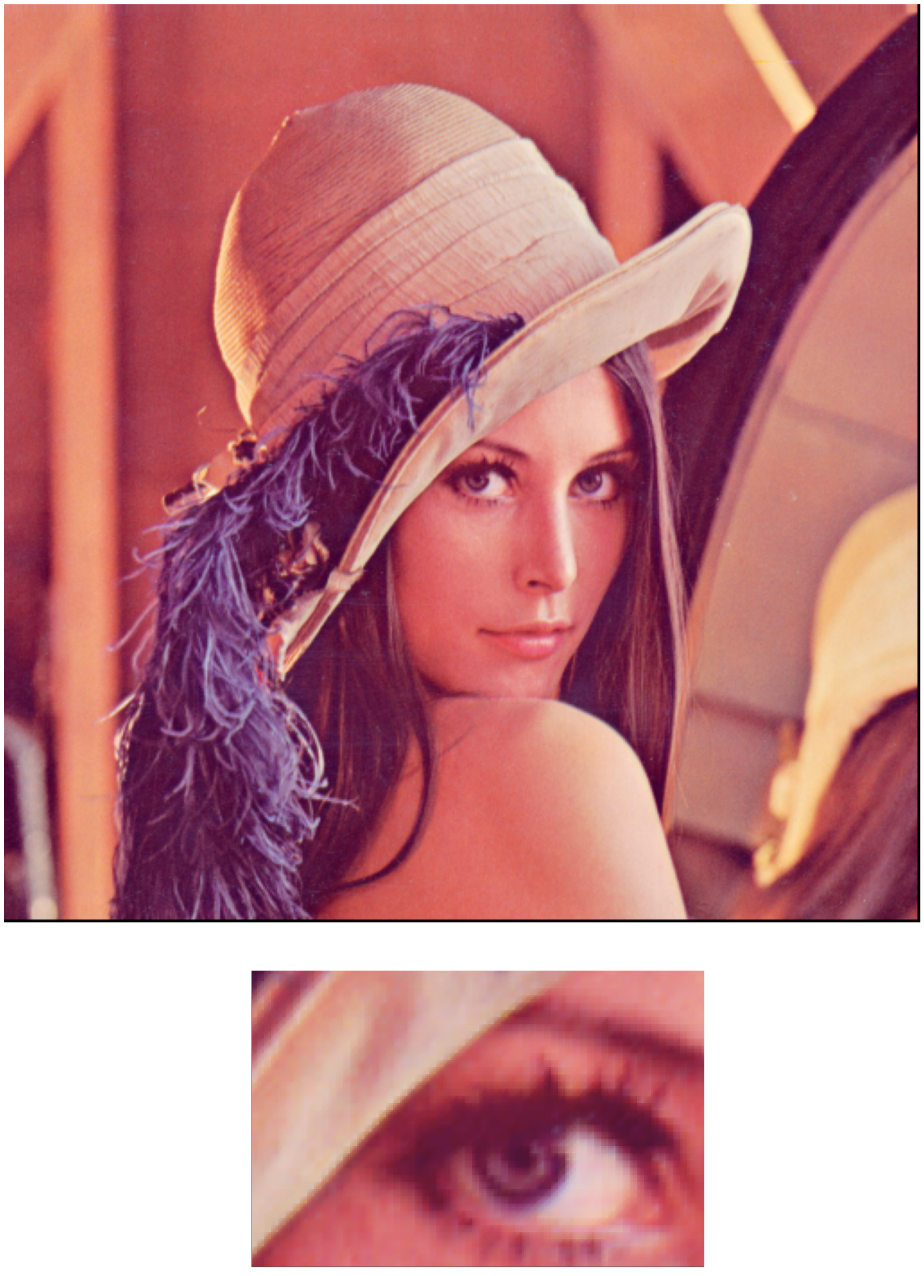

Figura 7.11: Imagen Lena Ampliada, segundo ordenamiento.
La imagen superior es una ampliación de la imagen original 7.5 de tamaño 1024x1024, siguiendo el orden de procesado: 1. Filtrado de la imagen. 2. Obtención de bordes. 3. Resizing. 4. Color. 5. Remmaping. La imagen inferior es una porción de la imagen con una ampliación al 200 % donde se puede apreciar que el efecto cuadrícula está más suavizado que con el ordenamiento anterior.

7.2. Mejoras

De acuerdo a los resultados anteriores, es posible percibir que al hacer enfoques la imagen presenta menor efecto cuadrícula con el segundo ordenamiento: 1. Filtrado de la imagen. 2. Obtención de bordes. 3. Resizing. 4. Color. 5. Remmaping; por lo tanto en base a este ordenamiento se ejecutan las mejoras, las cuales consisten principalmente en cambiar el orden del prosesamiento, realizar el Remmaping de distintas formas, añadiendo cantidad de píxeles vecinos a considerar y modificando los porcentajes de influencia de estos, algunos definidos y otros haciendo uso de proceso estocástico, aleatorizando una variable que defina el porcentaje de influencia a asignar.

Las mejoras presentadas se realizan dentro del Remmaping, que es la asignación de color en base a los colores de la imagen original una vez filtrada la imagen por los bordes. A continuación se presenta el ordenamiento de los algoritmos y 5 diferentes opciones de Remapping.

- Filtrado de la imagen.
- Obtención de bordes.
- Resizing.
- Color.
- Remmaping.
  - A lo más cuatro píxeles vecinos.
  - A lo más seis píxeles vecinos.
  - Proceso estocástico .
  - Proceso estocástico considerando el color de sus vecinos.
  - Proceso estocástico con dos variables.

7.2.1. Remapping: A lo más cuatro píxeles vecinos.

Este desarrollo es el presentado en el Remapping anterior, donde se asignan a los píxeles vacíos con coordenadas impar-par sus vecinos verticales, 25 % del

vecino superior y 75 % de su vecino inferior, de igual forma a los píxeles par-impar el 25 % del vecino izquierdo y el 75 % de su vecino derecho, y a los píxeles impar-impar los colores de sus cuatro vecinos más cercanos, un 10 % de su vecino superior izquierdo y 30 % de cada uno de los restantes, teniendo así a lo más 4 píxeles. Los resultados se pueden apreciar en la figura 7.11, anteriormente presentadas.

También se aplicó el procedimiento a mapas satelitales, para ello empleamos la imagen 7.12 de tamaño 700x489, el reto en mapas no solo es conservar, si no que al efocar podamos visualizar de manera clara las lineas de calles y delimitaciones de objetos. En la figura 7.13 se presenta la imagen ampliada de tamaño 1400x978 donde se hacen dos ampliaciones al 200 % y se puede ver que este método aunque no presenta efecto cuadrícula, difumina los colores lo que hace que se pierdan las líneas.

Figura 7.12: Imagen mapa.
Imagen satelital de tamaño 700x489.

Figura 7.13: Imagen mapa. A lo más cuatro vecinos.

La imagen superior es de tamaño 1400x978, resultado del procesamiento con el segundo ordenamiento considerando a lo más cuatro píxeles vecinos. En la parte inferior se muestran dos porciones ampliadas al 200 % tenemos la imagen esta muy difuminada, es decir, se pierden las líneas y también presenta efecto cuadrícula muy marcado.

7.2.2. Remapping: A lo más seis píxeles vecinos.

Este Remapping para los píxeles par-impar toma seis de sus vecinos más cercanos 7.14, asignando el mismo porcentaje de cada uno para nuestro nuevo píxel, de igual forma lo hace con los píxeles vacíos impar-par 7.15, tomando el mismo porcentaje de cada uno para el nuevo píxel. Para los píxeles vacíos impar-impar se realiza de la misma manera que en el Remmaping anterior.

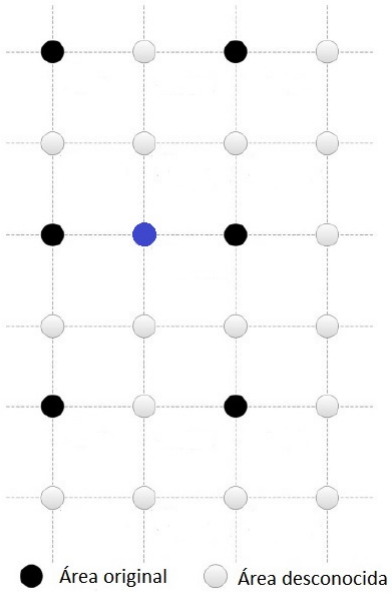

Figura 7.14: Píxel sin color asignado.
El píxel de color azul muestra el píxel que se rellenará en base a los seis píxeles originales mas cercanos (color negro), tomando el mismo porcentaje de cada uno

En la figura 7.16 se puede ver el diagrama de flujo que corresponde a este procesamiento.

Los resultados del procesamiento con a lo más seis píxeles es una imagen más suavizada, con menor efecto cuadrícula si se desea enfocar, pero con bordes menos definidos, es decir, los objetos no están bien delimitados. En la figura 7.17 podemos ver la imagen resultante de tamaño 1024x1024, en la imagen en la parte inferior podemos ver una ampliación al 200 % del resultado donde se

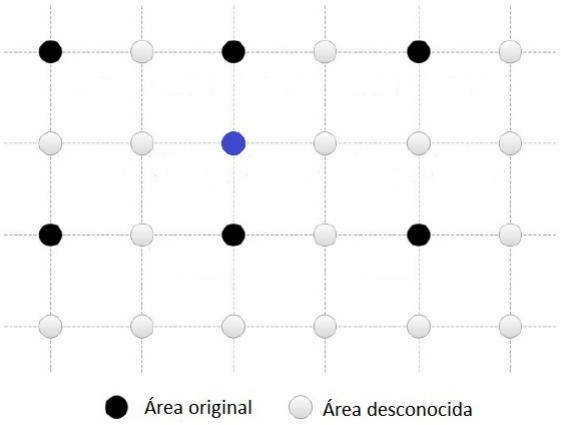

Figura 7.15: Píxel a asignar color 2.
El píxel de color azul muestra el píxel que se rellenará en base a los seis píxeles originales más cercanos (color negro), tomando el mismo porcentaje de cada uno

puede apreciar la imagen suavizada, con menor efecto cuadrícula y con menos definición.

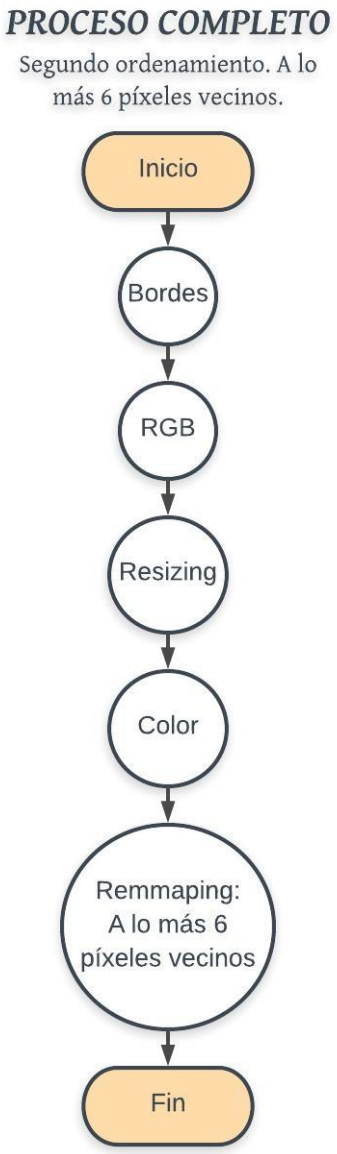

Figura 7.16: Diagrama de flujo. Proceso completo, segundo ordenamiento. A lo más seis píxeles vecinos.

Proceso completo para realizar el redimensionado de la imagen usando el segundo ordenamiento con el método de a lo más seis píxeles vecinos.

Figura 7.17: Imagen Lena ampliada. A lo más 6 píxeles vecinos.
La imagen superior es el resultado bajo el procesamiento de a lo más seis píxeles. En la imagen inferior tenemos una porción de la imagen ampliada al 200 % donde se muestra una imagen con menor efecto cuadrícula, más suavizada pero menos definida.

7.2.3. Proceso estocástico.

Este proceso es usando a lo más cuatro píxeles vecinos, la diferencia es que no se tiene definido que porcentaje de los píxeles vecinos originales se tomará, si no que se elige el porcentaje mediante un proceso estocástico. Para ello se genera la semilla con la que se trabajará, un número aleatorio (0, 1), dependiendo el valor obtenido será el porcentaje asignado.

Para los píxeles par-impar:

- Si está entre 0 y 0.25, elegir el píxel de la izquierda.
- Si está entre 0.25 y 0.5, elegir el píxel de la derecha.
- Si es igual o mayor o 0.5, hacer el promedio.

Para los píxeles impar-par:

- Si está entre 0 y 0.25, elegir el píxel superior.
- Si está entre 0.25 y 0.5, elegir el píxel inferior.
- Si es igual o mayor o 0.5, hacer el promedio.

Para los píxeles impar-impar:

- Si está entre 0 y 0.25, elegir el píxel de la superior izquierdo.
- Si está entre 0.25 y 0.5, elegir el píxel de la superior derecho.
- Si es igual o mayor o 0.5, hacer el promedio.

En la figura 7.18 se puede ver el diagrama de flujo correspondiente al procesamiento con proceso estocástico con una variable, con a lo más cuatro píxeles vecinos.

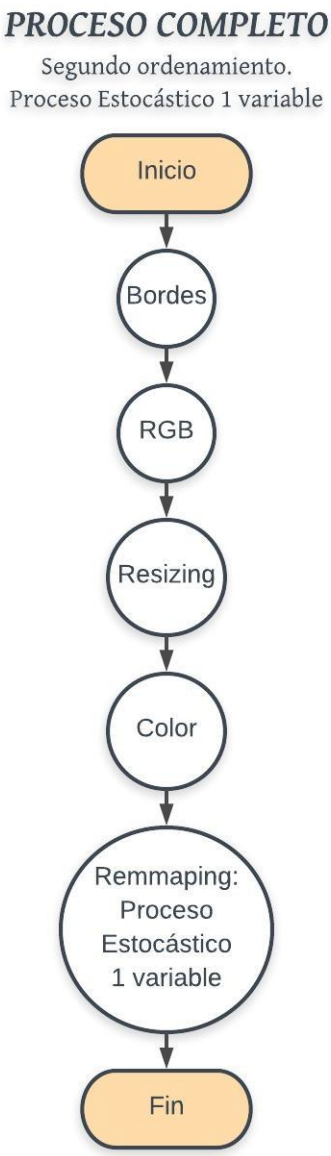

Figura 7.18: Diagrama de flujo. Proceso completo, segundo ordenamiento. Proceso Estocástico con 1 variable.

Proceso completo para realizar el redimensionado de la imagen usando el segundo ordenamiento con a lo más cuatro píxeles vecinos, con el método de Proceso Estocástico con 1 variable .

El resultado de esta técnica con a lo más cuatro píxeles vecinos eligiendo los porcentajes mediante un proceso estocástico es una imagen redimensionada al doble de su tamaño 1024x1024, donde se conservan los colores y no se pierde la definición de la imagen 7.19, y en la parte inferior de la imagen se enfoca en un área particular, podemos apreciar que la imagen está más suavizada, el efecto cuadrícula es mínimo y está bien definida.

Figura 7.19: Imagen Lena Ampliada. Proceso Estocástico con 4 píxeles vecinos.
La imagen superior de tamaño 1024x1024 es resultado del procesamiento por medio de proceso estocástico. En la parte inferior se tiene una porción con ampliación del 200 % donde se puede apreciar una imagen más definida, mucho más suavizada y con efecto cuadrícula casi nulo.

Bajo este procedimiento también se aplicó a la imagen 7.12 de tamaño 700x489 de mapa satelital, en la figura 7.20 se presenta la imagen ampliada de tamaño 1400x978 donde se hacen dos ampliaciones al 200 % y se puede ver que este método presenta efecto cuadrícula muy marcado, lo cuál en mapas permite conservar las lineas, es decir, tener definidas las calles y delimitados los objetos.

Figura 7.20: Imagen mapa. Proceso Estocástico con cuatro píxeles vecinos. La imagen superior es de tamaño 1400x978, resultado del procesamiento de proceso estocástico con una variable con a lo más cuatro píxeles vecinos. En la parte inferior se muestran dos porciones ampliadas al 200 % tenemos la imagen donde se puede apreciar que la imagen presenta un efecto cuadrícula marcada, sin embargo en este caso el efecto cuadrícula permite la delimitación de calles, ya que mantiene las líneas y bordes.

7.2.4. Proceso estocástico considerando el color de sus cuatro vecinos.

En este procedimiento, es igual al anterior utilizando a lo más cuatro píxeles, y de la misma manera que el anterior no se tiene definido que porcentaje de los píxeles vecinos originales se asignará al nuevo píxel, se elige el porcentaje mediante un proceso estocástico. Como anteriormente se genera la semilla con la que se trabajará, se da un número aleatorio (0, 1), dependiendo el valor obtenido será el porcentaje asignado, pero solo se usa cuando los vecinos son distintos entre sí, para ello se realiza una comparación de los colores, si son similares o iguales se toman como iguales, si la diferencia es grande se toman como diferentes.

- Si se tienen cuatro colores diferentes, hacer el promedio de los cuatro píxeles.
- Si tienes colores iguales y uno diferente, elegir el que tiene más colores iguales.
- Si se tienen dos colores:
  - Si está entre 0 y 0.25, elegir el píxel de la izquierda.
  - Si está entre 0.25 y 0.5, elegir el píxel de la derecha.
  - Si es igual o mayor o 0.5, hacer el promedio.

En la figura 7.21 se puede ver el diagrama de flujo correspondiente al procesamiento de proceso estocástico considerando el color de sus vecinos, con a lo más cuatro píxeles vecinos.

De este proceso obtenemos una imagen del doble de tamaño, 1024x1024, bien definida y de colores claros 7.22, si hacemos un enfoque en la imagen podemos notar que este proceso genera un efecto cuadrícula más pequeño.

Bajo este procedimiento igualmente se aplicó a la imagen 7.12 de tamaño 700x489 de mapa satelital, en la figura 7.23 se presenta la imagen ampliada de tamaño 1400x978 donde se hacen dos ampliaciones al 200 % y se puede ver que este método difumina, alargando los píxeles, lo que nos da una imagen difuminada pero a asimismo con efecto cuadrícula marcado, lo que nos resulta una mala imagen ya que se pierden las líneas.

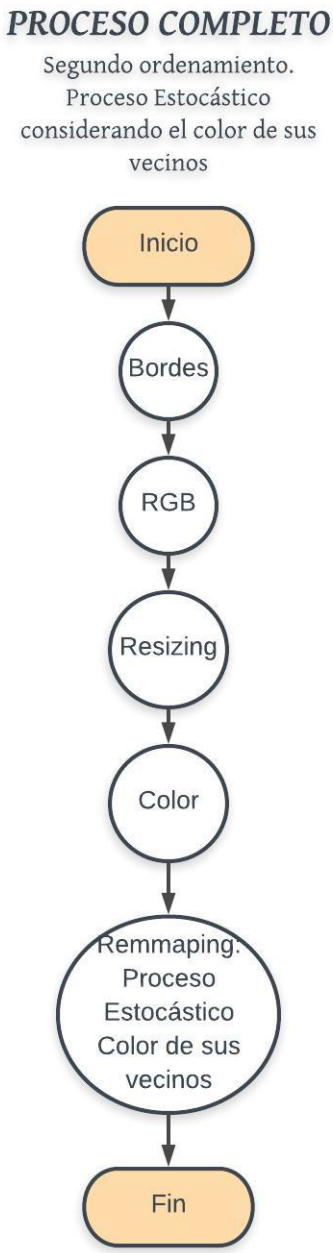

Figura 7.21: Diagrama de flujo. Proceso completo, segundo ordenamiento. Proceso Estocástico considerando el color de sus píxeles vecinos.

Proceso completo para realizar el redimensionado de la imagen usando el segundo ordenamiento con a lo más cuatro píxeles vecinos, con el método de Proceso Estocástico considerando el color de sus píxeles vecinos.

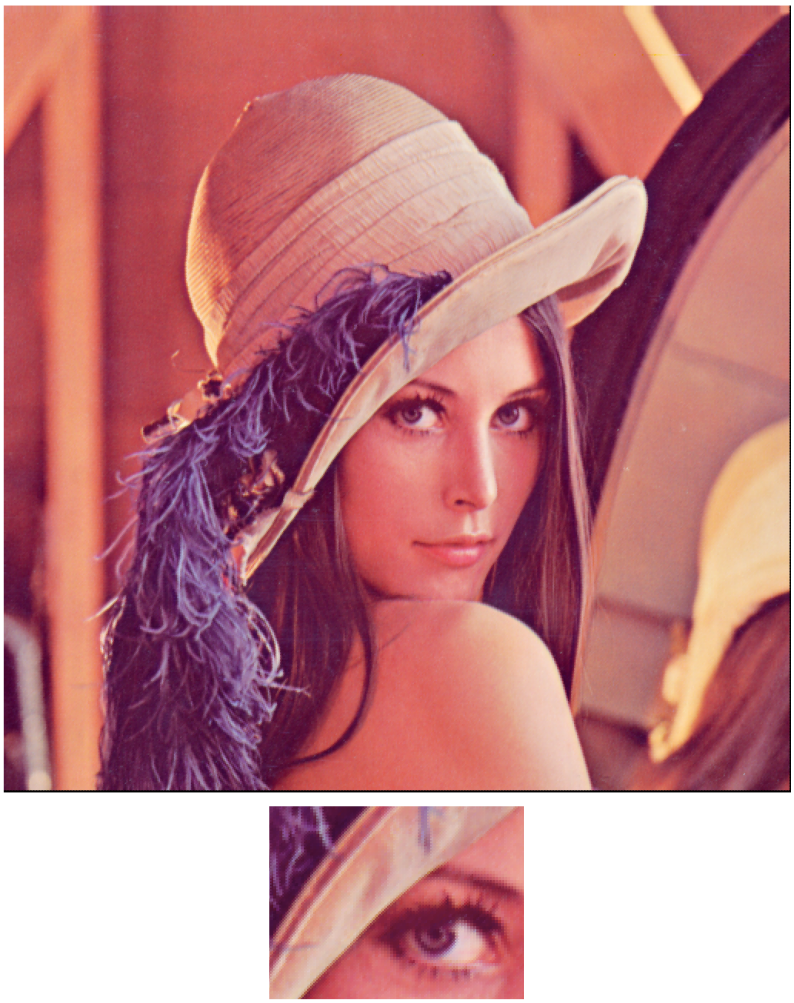

Figura 7.22: Imagen Lena Ampliada. Proceso Estocástico considerando el color de sus cuatro vecinos.

La imagen superior es de tamaño 1024x1024, resultado del procesamiento por medio de proceso estocástico. En la parte inferior se tiene una porción ampliada al 200 % donde se puede apreciar una imagen más definida, presenta efecto cuadrícula pero más pequeño

Figura 7.23: Imagen mapa. Proceso Estocástico considerando el color de sus cuatro vecinos.

La imagen superior es de tamaño 1400x978, resultado del procesamiento por medio de proceso estocástico. En la parte inferior se muestran dos porciones ampliadas al 200 % tenemos la imagen esta muy difuminada, es decir, se pierden las lineas y también presenta efecto cuadrícula muy marcado.

7.2.5. Proceso estocástico con dos variables.

Al igual que los dos anteriores este algoritmo usa a lo más cuatro píxeles, y tampoco se ha predefinido el porcentaje que se asigna a los nuevos píxeles de acuerdo a sus vecinos; para ello usaremos dos variables en el intervalo (0, 1), la primera es para los píxeles par-impar y impar-par, puesto que estos solo tienen dos vecinos, derecha e izquierda, y superior e inferior, respectivamente. Con la primera variable, para par-impar:

- Si está entre 0 y 0.25, elegir el píxel de la izquierda.
- Si está entre 0.25 y 0.5, elegir el píxel de la derecha.
- Si es igual o mayor o 0.5, hacer el promedio

Con la primera variable, para impar-par:

- Si está entre 0 y 0.25, elegir el píxel superior.
- Si está entre 0.25 y 0.5, elegir el píxel inferior.
- Si es igual o mayor o 0.5, hacer el promedio

La segunda variable se usa para el píxel que tiene cuatro vecinos, impar-impar, eligiendo el porcentaje de color de cada un de la siguiente manera:

- Si está entre 0 y 0.2, elegir el píxel superior izquierdo.
- Si está entre 0.2 y 0.4, elegir el píxel superior derecho.
- Si está entre 0.4 y 0.6, elegir el píxel inferior izquierdo.
- Si está entre 0.6 y 0.8, elegir el píxel inferior derecho.
- Si es igual o mayor o 0.8, hacer el promedio

En la imagen 7.24 se puede ver el diagrama de flujo correspondiente al proceso estocástico usando dos variables, con a lo más cuatro píxeles vecinos.

De este redimensionado obtenemos una imagen del doble de tamaño, 1024x1024, con colores claros y bien definida (figura 7.25), si hacemos un enfoque podemos ver que esta imagen presenta un efecto cuadrícula bastante marco, aunque conserva delimitados los bordes de la imagen.

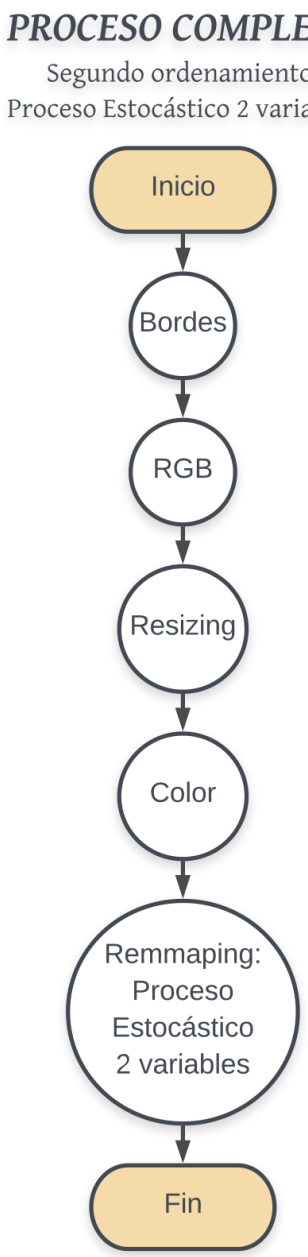

Figura 7.24: Diagrama de flujo. Proceso completo, segundo ordenamiento. Proceso Estocástico con dos variables.

Proceso completo para realizar el redimensionado de la imagen usando el segundo ordenamiento con a lo más cuatro píxeles vecinos, con el método de Proceso Estocástico con dos variables aleatorias.

Figura 7.25: Imagen Lena Ampliada. Proceso Estocástico con dos variables.
La imagen superior es de tamaño 1024x1024, resultado del procesamiento por medio de proceso estocástico con dos variables. En la imagen inferior se tiene una porción ampliada al 200 % donde se puede apreciar que la imagen presenta efecto cuadrícula más marcado

CAPÍTULO: 8

Conclusiones

Los resultados esperados eran poder realizar la ampliación de una imagen, y que esta conservará su nitidez y calidad, preservando los colores originales de la imagen y no se obscureciera, o bien, iluminara.

La primera implemetación es el cambio de orden del procesamiento de los algoritmos, donde en primer lugar se realizó el siguiente ordenamiento:

- Filtrado de la imagen.
- Obtención de bordes.
- Resizing.
- Remmaping.
- Color.

Donde efectivamente se pueden realizar ampliaciones de una imagen conservando colores y definición, pero en ella era notable un efecto cuadrícula ligero, por lo que se cambio de orden, realizandose primeramente el coloreo de la imagen y despues de ello se realizÃ$^3 el Remmaping$.

- Filtrado de la imagen.

- Obtención de bordes.
- Resizing.
- Color.
- Remmaping.

Bajo este ordenamiento el redimensionado de la imagen mantuvo los colores originales, pero además presenta los píxeles suavizados, es decir, el efecto cuadrícula presentado en el método anterior dismuye.

Considerando los resultados anteriormente mencionados, asegurando que el mejor ordenamiento para el procesamiento es realizar primero el Color y despues el Remmaping, las siguientes mejoras se realizan bajo este orden. Además de realizar el Remmaping con a lo más cuatro vecinos, se implemento cuatro mejoras adicionales:

1. A lo más cuatro píxeles vecinos.
2. A lo más seis píxeles vecinos.
3. Proceso estocástico .
4. Proceso estocástico considerando el color de sus vecinos.
5. Proceso estocástico con dos variables.

De los 5 procesamientos realizados bajo el ordenamiento, Filtrado de la imagen, Obtención de Bordes, Resizing, Color, Remmaping. Es notable, que la imagen es más suavizada cuando se incrementa el número de vecinos, como lo tenemos con el método de a lo más seis píxeles vecinos, pero incrementar el número de vecinos hace considerar colores más lejanos al píxel lo que provoca generar mas colores, por lo tanto, se obtiene mejor definición con cuatro píxeles vecinos.

Asimismo en el tratamiento de la imagen de mapa satelital bajo este ordenamiento se obtuvó una buena calidad de imagen, el inconventiente para el uso de mapas es que difumina la imagen y desaparece el efecto cuadrícula lo que provoca la perdida de nítidez en lineas o delimitaciones de calles, como lo fue con el método de a lo más cuatro vecinos bajo el segundo ordenamiento, porque

es mejor el uso de este procesamiento en fotografías donde se deseé conservar calidad y no delimitar objetos, como en el caso de mapas.

En cambio el procesamiento de Proceso Estocástico usando una variable con a lo más cuatro píxeles vecinos, aunque para el uso de imagenes y fotografías los resultados no los deseados ya que presenta mucho efecto efecto cuadrícula y poca calidad en la imagen, en el tratamiento de mapas satelitales se considera el mejor método, pese a que presenta efecto cuadrícula se realiza de tal forma que permite conservar las lineas, es decir, no se pierde la figura o delimitación de calles y objetos al realizar enfoques muy amplios.

Dentro del área de resultados se presenta primeramente un resultado no esperado dentro del procesamiento de imágenes, el resultado adicional es el encontrar imagenes que han sido truncadas, editadas o realizadas en forma de transparencias (por capas). El detector de bordes aplicado para los tratamientos de la imagen, además de obtener los bordes correctos en las imagenes a procesar, permite verificar la autenticidad de una imagen, al pasar la imagen por el detector de bordes mostrará si la imagen ha sido modificada anteriormente, lo cuál es una gran herramienta para descubrir si la imagen está editada de alguna forma, ya sea por medio de recortes, o por el uso de imagenes por capas.

Debido a los resultados obtenidos, donde tenemos métodos que difuminan correctamente pero no delimitan lineas (para el caso de mapas), o bien presentan gran efecto cuadrícula y se pierde la imagen, se pueden realizar implementaciones donde se cambien la topología de la célula, que en nuestro caso usamos el píxel con topología de forma cuadrada, lo cuál en algunos caso no permite una correcta delimitación. Otras topologías sugeridas es tomar nuestras celulas de forma triangular, para lograr delimitaciones más marcadas donde se tengan lineas diagonales, lo cuál no es posible de manera eficiente con los métodos presentados. Igualmente se puede aplicar una topología hexagonal, para tomar diferentes píxeles vecinos, lo cuál pretende lograr un mejor difuminado y eliminación de efecto cuadrícula con resultados esperados de una buena delimitación de las líneas diagonales.

CAPÍTULO: A

Anexos

A.1. Diagramas de flujo.

A.1.1. Color de la imagen en RGB.

El diagrama de flujo **RGB** describe el proceso de verificación de los componentes del vector de cada píxel de la imagen y nos asegura que cada uno sea de longitud 3, que corresponde al color en RGB.

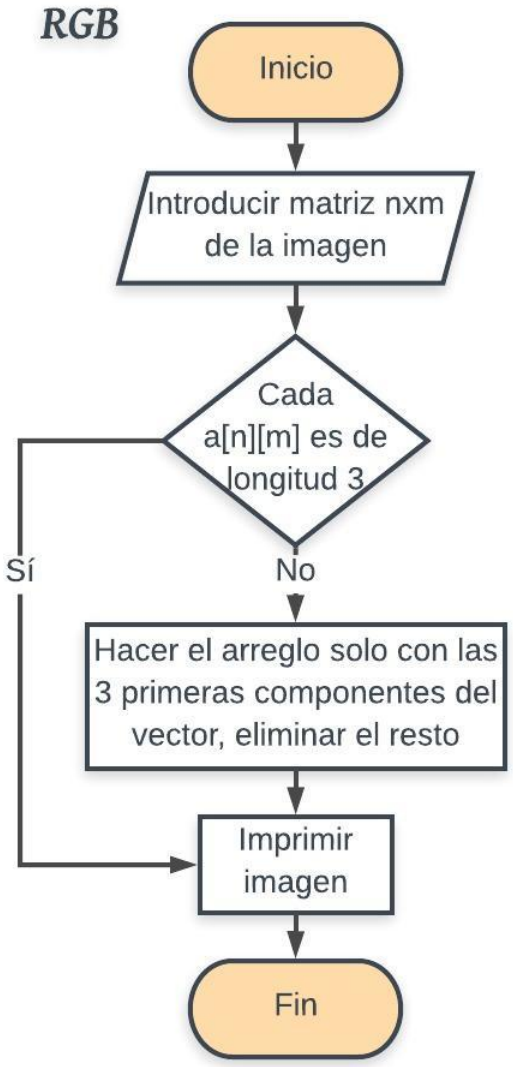

Figura A.1: Diagrama de flujo RGB

Proceso para verificar que la imagen tenga un vector de longitud 3, correspondiente al color en RGB en cada píxel.

A.1.2. Obtención de bordes.

El diagrama de flujo **BORDES** describe el proceso del filtrado de la imagen para obtener la diferenciación de las áreas homogeneas de aquellas que no lo son, es decir, los bordes de la imagen.

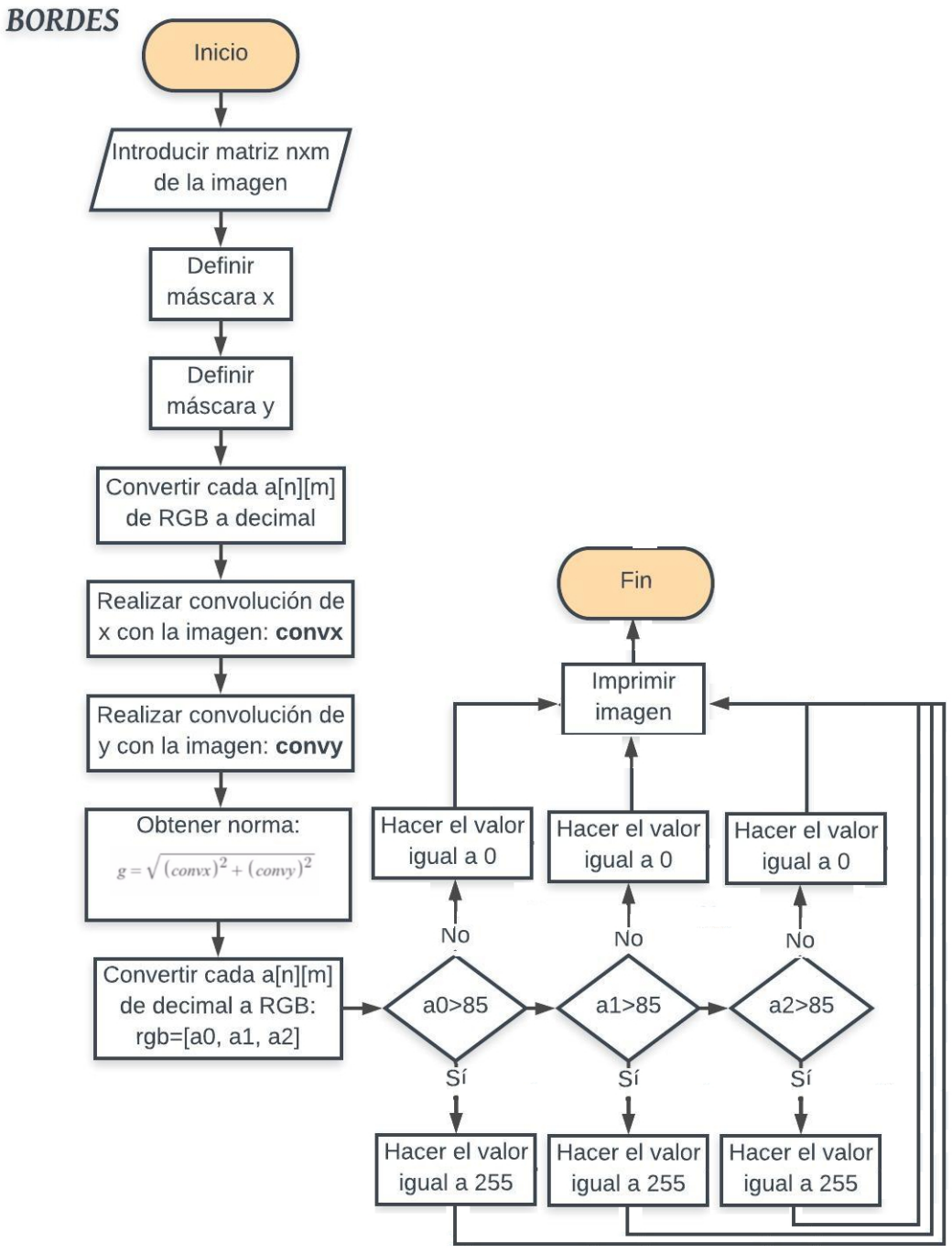

Figura A.2: Diagrama de flujo BORDES.
Proceso para filtrar la imagen y obtener sus bordes.

A.1.3. Resizing

El diagrama de flujo **RESIZING** describe el proceso para aumentar el tamaño de la imagen obtenida de la imagen de bordes, al doble de tamaño.

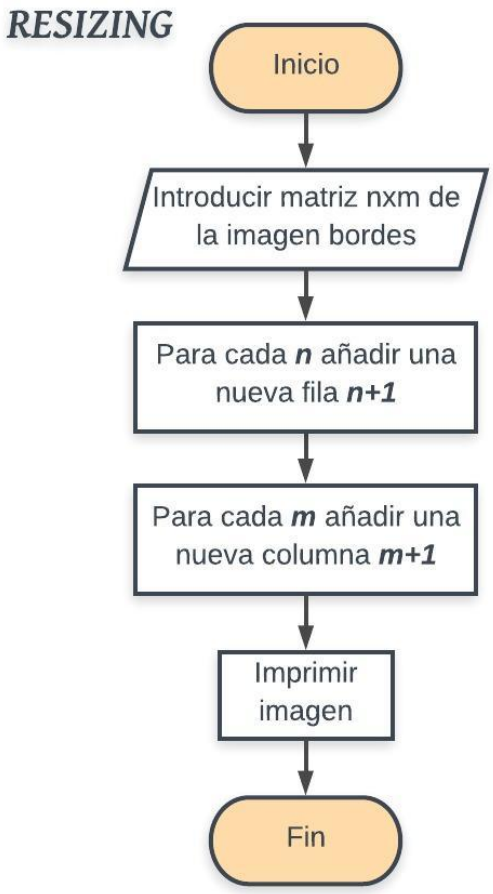

Figura A.3: Diagrama de flujo RESIZING.
Proceso para redimensionar la imagen.

A.1.4. Color

El diagrama de flujo **COLOR** describe el proceso para asignar colores a la imagen procesada en base a los colores de la imagen original.

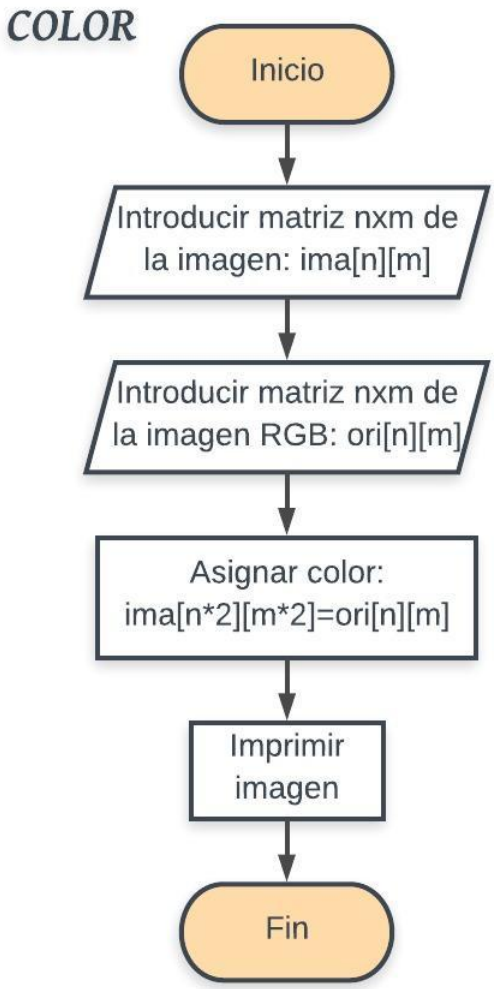

Figura A.4: Diagrama de flujo COLOR.
Proceso de asignación de color.

A.1.5. Remmaping, a lo más 4 píxeles

El diagrama de flujo **REMAPPING A lo más 4 píxeles** describe el proceso para asignar colores a la imagen con base en sus cuatro píxeles vecinos.

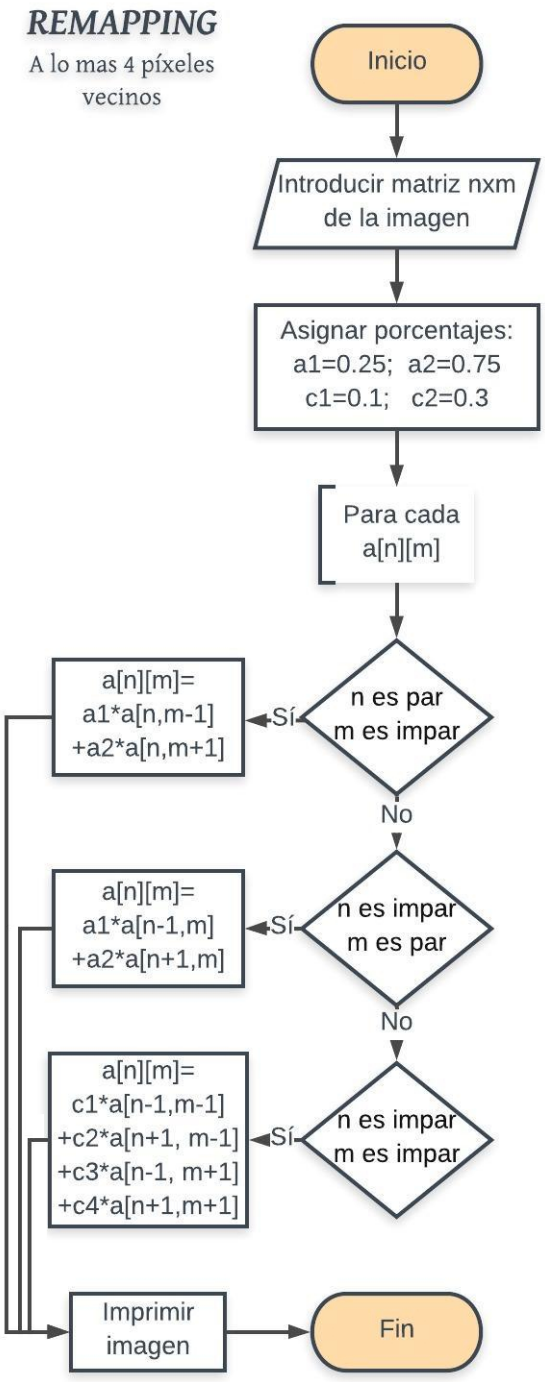

Figura A.5: Diagrama de flujo REMAPPING, A lo más 4 píxeles vecinos.
Proceso de asignación de color a todos los píxeles de la imagen considerando a lo más cuatro vecinos.

A.1.6. Remmaping, a lo más 6 píxeles

El diagrama de flujo **REMAPPING A lo más 6 píxeles** describe el proceso para asignar colores a la imagen considerando sus seis píxeles vecinos.

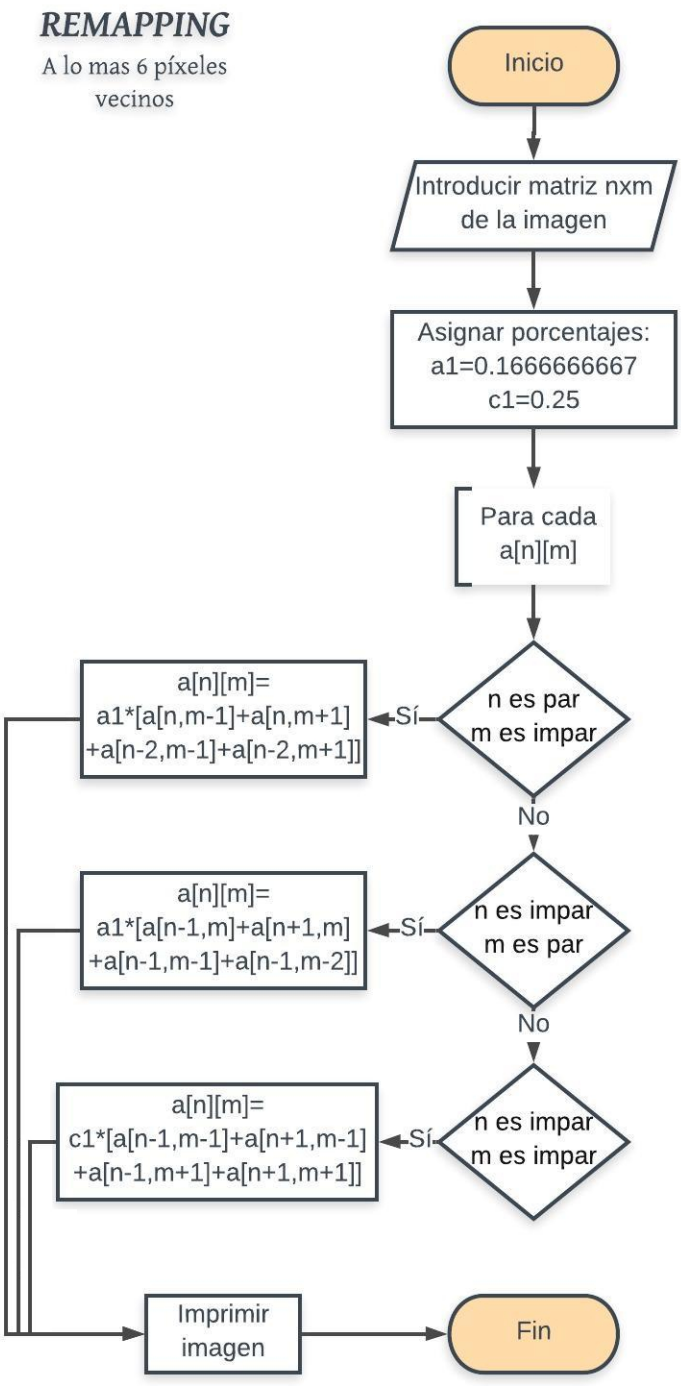

Figura A.6: Diagrama de flujo REMAPPING, A lo más 4 píxeles vecinos.
Proceso de asignación de color a todos los píxeles de la imagen considerando a lo más seis vecinos.

A.1.7. Remmaping, Proceso estocástico con 1 variable.

El diagrama de flujo **REMAPPING Proceso estocástico con 1 variable** describe el proceso para asignar colores a la imagen considerando cuatro píxeles vecinos, usando una variable aleatoria que determina el porcentaje de color que será asignado.

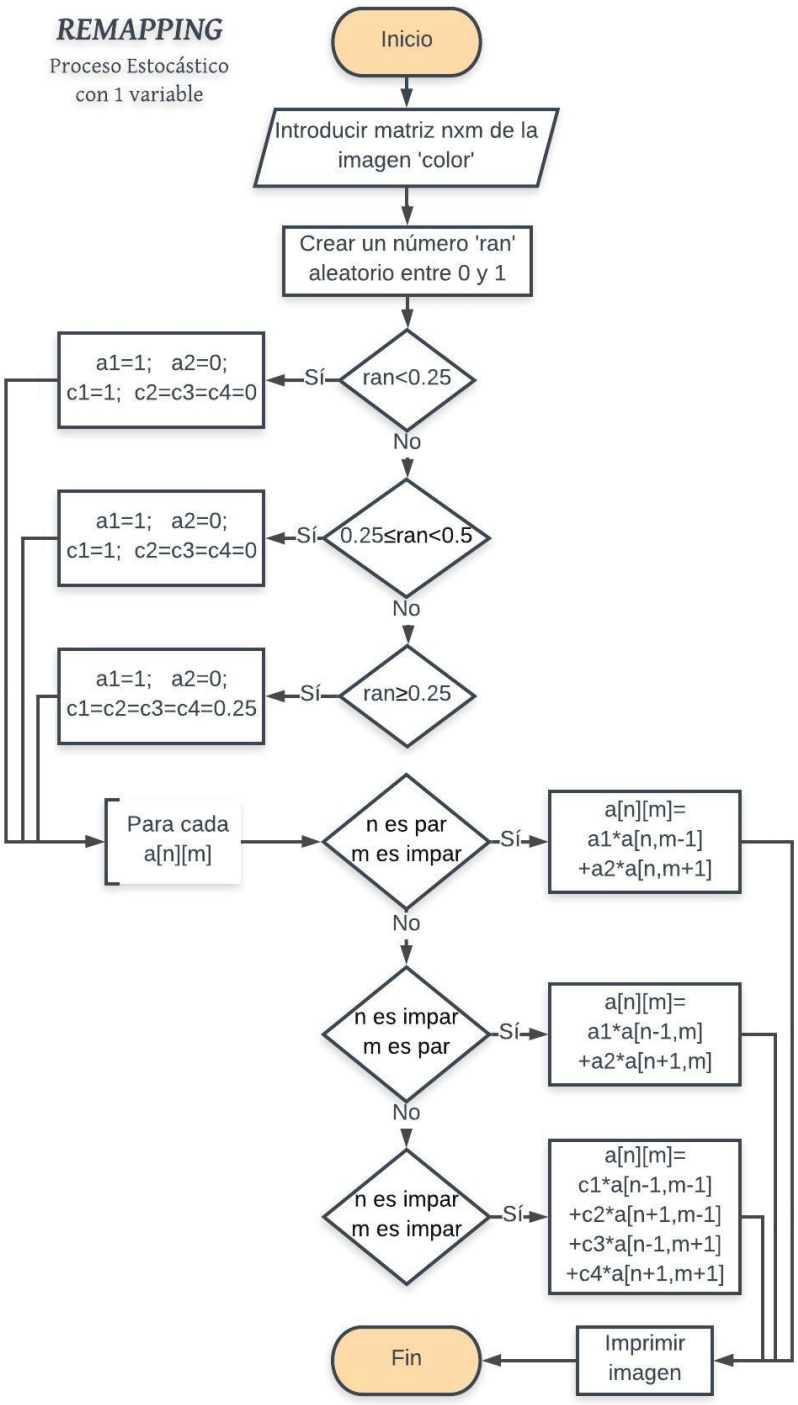

Figura A.7: Diagrama de flujo REMAPPING, Proceso estocástico con 1 variable.
Proceso de asignación de color a todos los píxeles de la imagen considerando a lo más cuatro vecinos, usando una variable aleatoria para asignar porcentajes.

A.1.8. Remmaping, Proceso estocástico con 2 variables.

El diagrama de flujo **REMAPPING Proceso estocástico con 2 variables** describe el proceso para asignar colores a la imagen considerando cuatro píxeles vecinos, usando una variable aleatoria que determina el porcentaje de color que será asignado para aquellos píxeles que solo tiene dos vecinos y una segunda variable aleatoria para aquellos píxeles que tengan 4 píxeles vecinos.

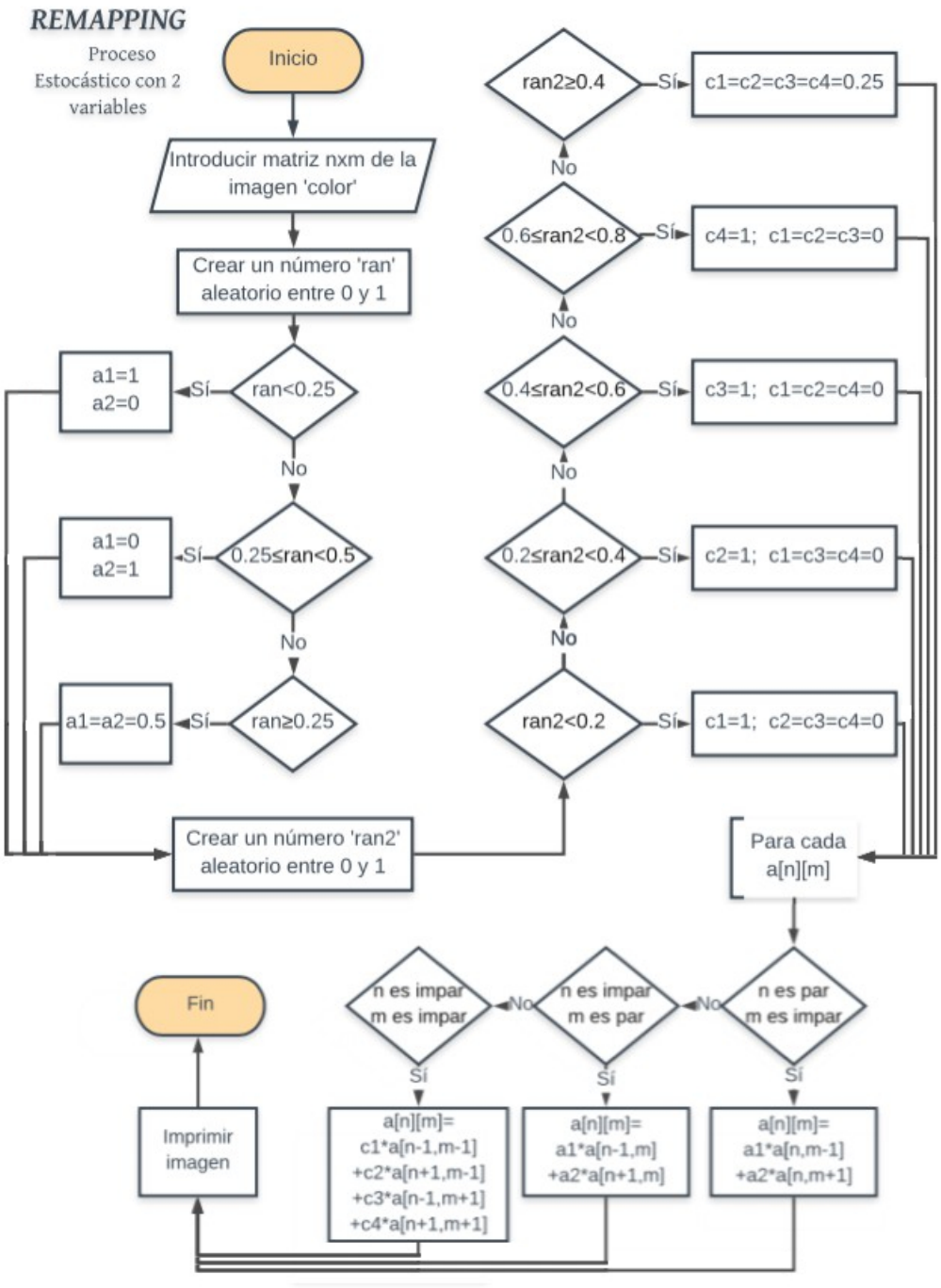

Figura A.8: Diagrama de flujo REMAPPING, Proceso estocástico con 2 variables.
Proceso de asignación de color a todos los píxeles de la imagen considerando a lo más cuatro vecinos, usando dos variables aleatoria para asignar porcentajes.

A.1.9. Remmaping, Proceso estocástico connsiderando el color de sus vecinos.

El diagrama de flujo **REMAPPING Proceso estocástico considerando el color de sus vecinos** describe el proceso para asignar colores a la imagen considerando cuatro píxeles vecinos, usando una variable aleatoria que determina el porcentaje de color que será asignado cuando se tengan colores diferentes entre los vecinos.

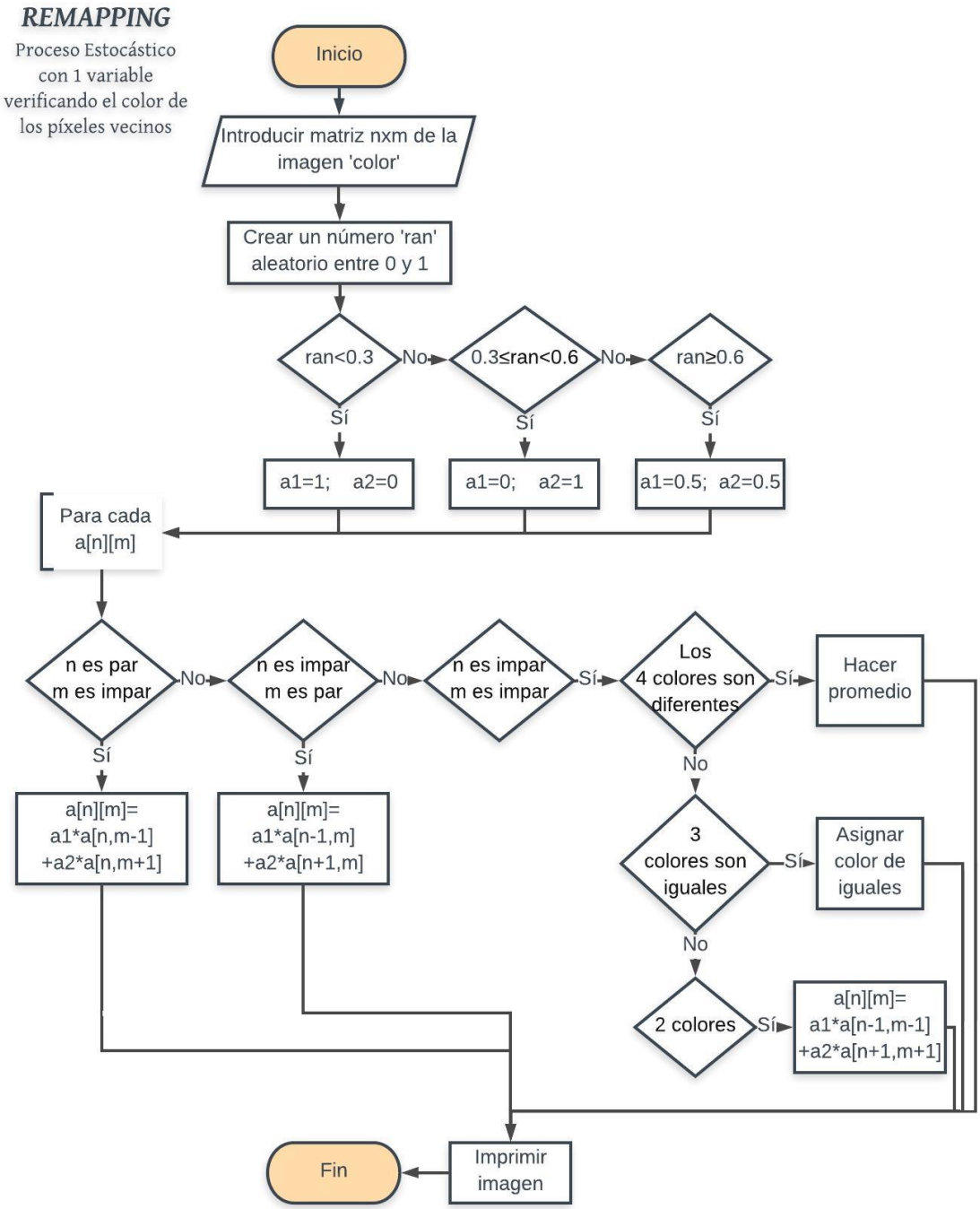

Figura A.9: Diagrama de flujo REMAPPING, Proceso estocástico considerando el color de sus vecinos.

Proceso de asignación de color a todos los píxeles de la imagen considerando a lo más cuatro vecinos, usando una variable aleatoria para asignar porcentajes y considerando el color de sus vecinos.

Referencias

Gray, L. (2003). A mathematician looks at wolfram's new kind of science. *Notices-American Mathematical Society*, *50*(2), 200–211.

Hwang, J. W., y Lee, H. S. (2004). Adaptive image interpolation based on local gradient features. *IEEE Signal Processing Letters*, *11*(3), 359–362.

Ioannidis, K., Andreadis, I., y Sirakoulis, G. C. (s.f.). An edge preserving image resizing method based on cellular automata.

Ioannidis, K., Andreadis, I., y Sirakoulis, G. C. (2012). An edge preserving image resizing method based on cellular automata. En *International conference on cellular automata* (pp. 375–384).

Karafyllidis, I., y Thanailakis, A. (1997). A model for predicting forest fire spreading using cellular automata. *Ecological Modelling*, *99*, 87–97.

Keys, R. (1981). Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, *29*(6), 1153–1160.

Macedo, Y. P. A., y Chaimowicz, L. (2017). Improving procedural 2d map generation based on multi-layered cellular automata and hilbert curves. En *2017 16th brazilian symposium on computer games and digital entertainment (sbgames)*.

Peer, M., Qadir, F., y Khan, K. (2012). Investigations of cellular automata game of life rules for noise filtering and edge detection. *International Journal of Information Engineering and Electronic Business*, *4*(2), 22.

Qadir, F., Peer, M., y Khan, K. (2013). Digital image scrambling based on two dimensional cellular automata. *International Journal of Computer Network and Information Security*, *5*(2), 36.

Saha, S., Nadiga, S., Thiaw, C., Wang, J., Wang, W., Zhang, Q., ... others (2006). The ncep climate forecast system. *Journal of Climate*, *19*(15), 3483–3517.

Sarkar, P. (2000). A brief history of cellular automata. *Acm computing surveys (csur)*, *32*(1), 80–107.

Sayama, H. (2015). *Introduction to the modeling and analysis of complex systems*. Open SUNY Textbooks.

Sevcenco, A.-M., y Lu, W.-S. (2007). Combined adaptive and averaging strategies for jpeg-based low bit-rate image coding. En *Electrical and computer engineering, 2007. ccece 2007. canadian conference on* (pp. 168–171).

Sirakoulis, G. C., y Bandini, S. (2012). *Cellular automata: 10th international conference on cellular automata for research and industry, acri 2012, santorini island, greece, september 24-27, 2012. proceedings* (Vol. 7495). Springer.

Xiong, R., Ding, W., Ma, S., y Gao, W. (2010). Improved autoregressive image model estimation for directional image interpolation. En *Picture coding symposium (pcs), 2010* (pp. 442–445).

Zhao, Y., Guo, H., y Billings, S. A. (2012). Application of totalistic cellular automata for noise filtering in image processing. *J. Cellular Automata*, *7*(3), 207–221.

Curriculum Vitae

Formación Académica

Universidad Autónoma de Chihuahua. Facultad de Ingeniería (2010-2014)
Título: Ingeniero Matemático. Cédula Profesional 9733059

Centro de Bachillerato Tecnológico Agropecuario 147 (2007-2010)
Título: Técnico en Informática. Cédula 7488589

Publicaciones

Montes, W. E., López, O. E., Contreras, G. A. V., Herrera, A. J. L. (Junio - Agosto 2018), La conexión de las elecciones, FINGUACH, Año 5, Núm. 16, págs. 3-5.

López, O. E., Hinojosa, G. O. R., Herrera, A. J. L. (Diciembre 2017- Enero 2018), Música y matemáticas, FINGUACH, Año 4, Núm. 14, págs. 6-7.

Domicilio: Calle Walter Scott # 912, Fracc. Alamedas III, Chihuahua, Chih. C.P. 31136.

Esta tesis fue mecanografiada por *Eliana López Ortega*