UNIVERSIDAD MICHOACANA
DE SAN NICOLÁS DE HIDALGO

FACULTAD DE INGENIERÍA ELÉCTRICA

"RECONOCIMIENTO INTELIGENTE DE FLUJO VEHICULAR BASADO EN EL PROCESAMIENTO DE IMÁGENES"

TESIS

Que para obtener el Título de:
INGENIERO EN ELECTRÓNICA

Presenta:
VICENTE BLAS CABELLO HERNÁNDEZ

Asesor de Tesis:
DR. JUAN ANZUREZ MARÍN

Morelia, Michoacán, Enero 2010

Agradecimientos

A la Universidad Michoacana de San Nicolás de Hidalgo, en especial a la Facultad de Ingeniería Eléctrica, por los conocimientos adquiridos dentro de sus aulas.

A la mesa de sinodales conformada por: M.I. Isidro I. Lázaro Castillo, M.I. Salvador Ramírez Zavala, Dr. Antonio Ramos Paz, Dra. Elisa Espinosa Juárez, por sus valiosos comentarios y tiempo dedicado a la revisión de este documento, con lo cual se mejoró la presentación y redacción del mismo.

Muy especialmente al Dr. Juan Anzurez Marín por todo el apoyo brindado durante la elaboración de este proyecto de tesis, gracias por su humildad, dedicación y paciencia

A mi hijo que con su ternura y amor me ha enseñado a descubrir el amor de Padre.

Por supuesto a mis padres los señores Vicente Cabello Rayas, La Sra. Italia Hernández Avilés, A mis hermanos Beto, Pepe, Lupe, Toña y Wendy por apoyarme en todos los momentos dándome la fortaleza para seguir adelante, a mi cuñado Pablo Pineda por su apoyo incondicional.

A ti Isela, porque contigo conocí el amor, por los momentos que hemos pasado juntos que han sido incomparables y por haberme dado un hijo maravilloso.

A la casa del estudiante Isaac Arriaga de la Universidad Michoacana De San Nicolás de Hidalgo, por abrirme las puertas y permitir con ello seguir con mis estudios.

Al Programa de Fortalecimiento al Programa de Beca (BecaNet Superior), por haberme aceptado y con ello obtener la Beca de Titulación, puesto que esta beca sido un aliciente para la realización de este proyecto de tesis. Y que tiene la dirección de Internet http//:www.becanetsuperior.sep.gob.mx.

Dedicatoria

A MI PADRE Y A MI MADRE los señores Vicente Cabello Rayas e Italia Hernández Avilés, puesto que es por ellos que sigo de pie, por el amor que como padres me han dado el mejor de los ejemplos, y es que, aun en las más grandes adversidades siempre están unidos dándome toda su comprensión y apoyo.

CON TODO MI AMOR A MI HIJO el regalo más grande que Dios trajo a mi vida despertando ese amor tan incomparable y a la vez inexplicable de ser padre, tú eres el motivo más importante de mi vida **Raciel**.

Resumen

Reconocimiento inteligente de flujo vehicular basado en el procesamiento de imágenes.

El objetivo del presente trabajo de tesis es, sentar las bases para determinar la densidad de flujo vehicular en las diferentes vías automovilísticas de la ciudad de Morelia Michoacán, México, mediante el uso de visión artificial y procesamiento de imágenes, con la finalidad de que en una siguiente etapa se llegue a la integración de la información necesaria para el control de semáforos inteligentes, que tomen decisiones automáticas en relación a los tiempos de luz verde, basado en la densidad de flujo vehicular de acuerdo a las imágenes tomadas de las cámaras.

Para lograr el conteo de flujo vehicular se registran videos de aproximadamente 5 minutos cada uno, de tal forma que mediante la selección de imágenes consecutivas se pueda estimar el fondo del lugar monitoreado. Una vez estimado el fondo, se compara con una nueva imagen, la cual presenta autos a contabilizar del mismo lugar, con el fin de eliminar toda la parte estática y analizar la parte dinámica (autos), que es el interés del presente trabajo.

Mediante el uso de herramientas de probabilidad, erosión, dilatación y conectividad, es posible determinar la cantidad de autos presentes en la imagen. La aportación del presente trabajo de tesis es precisamente la selección óptima de dichas herramientas.

Contenido

Página

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

El tráfico vehicular en las grandes urbes, es un problema que día con día se forma más complejo y origina un gran número de problemas que abarcan desde pérdida de tiempo, gasto innecesario de combustible, entre otros. Esta problemática conduce a buscar posibles soluciones que den un mayor control en cuanto al tráfico vehicular se refiere, por lo que se han propuestos sistemas de monitoreo mediante lazos inductivos que agilicen la congestión vehicular como lo es el caso del Sistema Integral de Tránsito Metropolitano (Sintram), ubicado en la Ciudad de Monterrey Nuevo León México [1]. Sin embargo, se ha visto conveniente la creación de un sistema que permita realizar esta función en forma automática y con un alto nivel de confiabilidad mediante el análisis de imágenes.

Dicho sistema, presenta una solución a los problemas de alta congestión vehicular que se registran en cruces donde generalmente se envía un agente de tránsito para realizar las funciones de semáforo, idealmente con un mejor criterio en cuanto a tiempos se refiere, quedando todo este tiempo ignorados por completo los semáforos, lo cual es un indicativo de que es ineficiente el sistema actual.

En la presente tesis se propone la creación de un programa implementado en MatLab, que por medio de la obtención de imágenes y el procesado de las mismas nos genere información de flujo vehicular que se pueda utilizar en etapas posteriores para la toma de decisiones en semáforos respecto a tiempos

de la señal de *siga* para los autos, dependiendo esta acción del flujo vehicular registrado en cada uno de los carriles monitoreados.

El sistema propuesto estaría integrado por una cámara para la obtención de las imágenes y software para el procesado de las mismas y de esta forma generar información para una etapa posterior de control de semáforos.

1.1 Antecedentes

A diferencia del estudio de los mecanismos de la visión humana, el procesamiento y análisis de imágenes digitales nace en el momento en que se dispone de recursos tecnológicos para captar y manipular grandes cantidades de información espacial en forma de matrices de valores. Esta distinción sitúa al procesamiento y análisis de imágenes digitales como una tecnología asociada a las *Ciencias de la Computación* y por lo tanto es posible definirla como una proyección del término *Visión Artificial* dentro del ámbito de la *Inteligencia Artificial* [2], [3].

Así, aparecen los conceptos de *técnicas para el procesamiento de imágenes digitales* como el conjunto de todas aquellas técnicas asociadas a la captura, codificación y representación de las imágenes que no introducen sobre las mismas ningún tipo de interpretación, y*, técnicas de visión por computadora o visión mediante robot* como acepciones que se refieren a aquellas técnicas que tratan de extraer la información presente en la imagen con el fin de hacer una interpretación de la escena representada por dicha imagen [3].

Durante los años ochenta las técnicas de análisis de imágenes se desarrollan de forma vertiginosa como consecuencia de la gran cantidad de aplicaciones que aparecen y la madurez alcanzada en el diseño de arquitecturas de computadoras. Los desarrollos más teóricos han seguido en gran medida las pautas marcadas por **Courtnay David Marr**, habiendo sido la línea marcada por el **Instituto Tecnológico de Massachussets** la que más influencia ha tenido. Las mayores contribuciones se han centrado en el desarrollo de algoritmos para la detección de características (bordes, líneas, texturas) que ayudan a definir lo que Marr llamo el *esbozo primitivo*, así como en el

desarrollo de técnicas globales de segmentación de una imagen en regiones. Por lo anterior cabe destacar aquellas aproximaciones que introdujeron la información de contexto en los procesos de clasificación y segmentación [3], [4], [5].

Debido al desarrollo de las técnicas y métodos matemáticos, también se desarrollaron diferentes arquitecturas de computadoras específicas para el procesamiento de datos de imágenes digitales, generando mayor avance en cuanto a algoritmos de reconocimiento de patrones en imágenes se refiere. Como consecuencia de esto, el término *Visión Artificial* dentro del campo de la Inteligencia Artificial puede considerarse como el *conjunto de todas aquellas técnicas y modelos que nos permitan el procesamiento, análisis y explicación de cualquier tipo de información espacial obtenida a través de imágenes digitales* [3].

Han tenido un significado especial la gran cantidad de trabajos que han usado técnicas de representación del reconocimiento para los problemas de interpretación de imágenes, en relación con aplicaciones de ambiente industrial, se han desarrollado fuertemente híbridos entre las técnicas de la Inteligencia Artificial para la aplicación del conocimiento y los técnicas de interpretación de escenas a partir de imágenes digitales [1], [3], [6].

En la actualidad los trabajos que se están efectuando en relación a la visión artificial y procesamiento de imágenes son variados, por lo general se enfocan a una misma finalidad, la identificación de objetos dentro de una imagen digital, las técnicas son numerosas y variadas, por lo que van desde descriptores de Fourier, comparaciones respecto a máscaras, conectividad, etc.

Una aplicación importante y de suma utilidad es el control de flujo vehicular debido a la gran congestión vehicular presente en las grandes ciudades, a pesar de que se han empleado diferentes técnicas de identificación el costo computacional es una limitante que se ha presentado al hacer uso de análisis de imágenes y son tantas las variantes que intervienen en una imagen que es muy difícil obtener un reconocimiento del 100% de la identificación a realizar. Por lo que la tendencia es ir creando diferentes técnicas que influyan

de manera que al integrarlas y trabajando en un mismo reconocimiento si falla alguna, la otra no omita la falla, es decir si una técnica no logra determinar con certeza la identificación la otra si lo haga y de esta manera obtener más confiabilidad en el reconocimiento.

1.2 Objetivo

Desarrollar un programa en MatLab de procesamiento de imágenes, que mediante el análisis y procesado píxel a píxel de las mismas, nos ayude a detectar el alto o bajo flujo vehicular en zonas de alto índice automovilístico.

1.3 Justificación

Debido a la gran congestión vehicular que actualmente enfrenta la ciudad de Morelia Michoacán, México, en las diferentes vías que tienen semáforos con tiempos definidos, y que, para evitar lo antes mencionado se envía un agente de tránsito para hacer las funciones del semáforo que con su experiencia agilice el flujo vehicular, sin embargo con esto descuida otros puntos de la ciudad que podría estar atendiendo. Cabe mencionar que también el poco avance de los vehículos hace que las personas se estresen y debido a su desesperación existan mayores accidentes en su intento por avanzar, aunado a esto el tiempo perdido se pudiera utilizar en algo más productivo. Por otra parte hay que destacar que son muchos los puntos dentro de la ciudad que cuentan con este sistema de semáforos, por lo que tomando en cuenta el gran desarrollo que actualmente ha tenido la visión artificial y suponiendo que una solución a esta problemática es el conteo de vehículos mediante el procesamiento digital de imágenes y con esto poder abrir el panorama para el diseño *hardware y software* necesario que nos conduzca a crear el sistema completo de control de semáforos actuales con algunas modificaciones, y de esta manera poder definir los tiempos de *siga* en los semáforos de los diferentes carriles de un cruce vehicular mediante visión artificial. Ayudando con esto a reducir la congestión vehicular así como la problemática que trae consigo, como es mayor consumo de combustible,

desesperación en los automovilistas, mejora en los tiempos de espera, entre otros.

1.4 Metodología

Para lograr el objetivo fundamental de este proyecto de tesis, se propone la siguiente metodología.

1. Revisión bibliográfica.
2. Seleccionar la técnica de procesamiento a utilizar.
3. Aplicar el procesamiento de imagen a las imágenes capturadas como es:
    a. Obtención de imágenes de un mismo lugar en diferentes tiempos para la extracción del fondo.
    b. Comparación del fondo obtenido, contra una imagen del mismo lugar con autos, para eliminar todo lo estático y obtener una imagen binaria.
    c. Aplicación de la técnica de erosión y dilatación a la imagen binaria para crear una silueta homogénea del auto.
    d. Por análisis de conectividad en la imagen binaria determinar la cantidad de autos en la imagen.
4. Mostrar resultados y conclusiones.

Capítulo 2

Conceptos fundamentales del procesamiento de imágenes

En el procesamiento digital de imágenes, es muy importante tener bien definido como se encuentran estructuradas y con qué herramientas o técnicas contamos para lograr trabajar con ellas. A continuación se detallan conceptos fundamentales y técnicas para el procesado de imágenes utilizadas en este proyecto.

2.1 Visión por computadora

La visión es uno de los mecanismos sensoriales de percepción más importantes que tiene el ser humano, y la mayoría de los organismos biológicos la utilizamos para interactuar eficientemente dentro del ambiente que nos rodea y detectar los objetos que son de nuestro interés por medio de su forma, color, relieve, dimensiones, distancia a la que se encuentra, etc.

En el caso de la visión artificial, la podemos definir como la capacidad de la máquina para ver el mundo que le rodea, para deducir la estructura y las propiedades del mundo tridimensional a partir de una o más imágenes bidimensionales. El establecimiento en una máquina, de habilidades como la de detectar y determinar la identidad de los objetos, no sólo liberan al hombre de tareas tediosas y peligrosas, sino también permite la realización de algunas otras tareas imposibles de ejecutar para el ser humano [7].

2.2 Píxel

Es el elemento básico de una imagen. Un pixel o píxel (acrónimo del inglés *picture element*, "elemento de imagen") es la menor unidad homogénea en color, que forma parte de una imagen digital, ya sea de una fotografía, fotograma de video o un gráfico [8], [9].

Ampliando lo suficiente una imagen digital (zoom), por ejemplo en la pantalla de una computadora, se puede observar los píxeles que componen la imagen. Los píxeles aparecen como pequeños cuadrados o rectángulos en color, en blanco o en negro, o en tono de grises.

Las imágenes se forman como una matriz rectangular de píxeles, donde cada píxel forma un área relativamente pequeña respecto a la imagen total. En la Figura 2.1, se muestra un ejemplo de ampliación a una imagen, un píxel ahora representa un nuevo conjunto de píxeles como consecuencia de la aplicación del zoom digital.

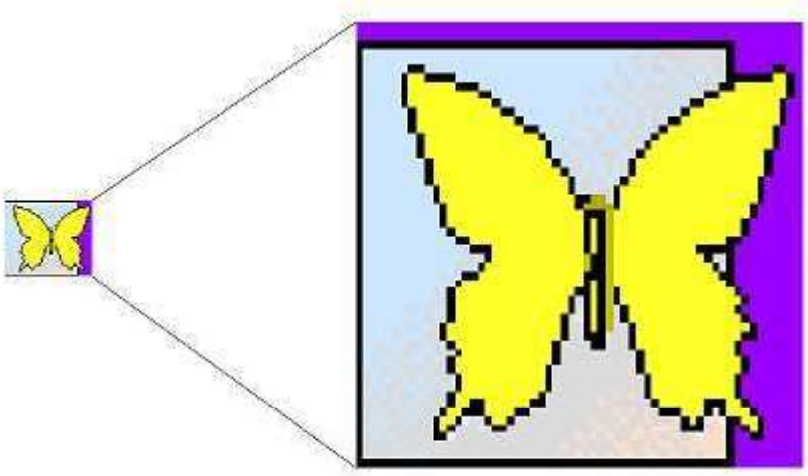


Figura 2.1 Ampliación de una imagen.

2.3 Imagen

Una imagen es un arreglo bidimensional de píxeles con diferente intensidad luminosa (escala de grises). La Figura 2.2, muestra un ejemplo de imagen con

16 píxeles en escala de grises y su distribución en el plano (x,y).

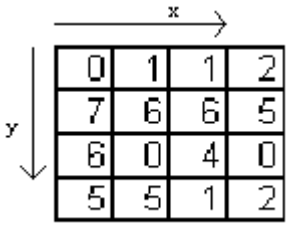


Figura 2.2 Imagen de 16 píxeles.

Si la intensidad luminosa de cada píxel se representa por *n* bits, entonces existirán 2^n escalas de grises diferentes. Una imagen se representa por:

$$r = f(x, y) \tag{2.1}$$

Donde *r* es la intensidad luminosa del píxel cuyas coordenadas son *(x,y)*. Matemáticamente, un sistema para procesar imágenes se representa como:

$$g(x, y) = T[f(x, y)] \tag{2.2}$$

Donde *T* representa la transformación aplicada al píxel con coordenadas función de (*x,y*), y g(*x,y*) el nuevo valor de intensidad adquirido [8].

2.4 Formato JPG (JPEG)

El formato JPEG ofrece los imprescindibles 16 millones de colores (*truecolor*), unido a una compresión realmente asombrosa (valores superiores a 20:1 son habitúales). Sólo tiene una limitación: para obtener esos valores de compresión modifica sutilmente la imagen, descartándose su uso en aplicaciones en las que se desea mantener una calidad bit a bit. El diseño de este formato está pensado para almacenar imágenes del "mundo real", también llamadas imágenes de tono continuo, como digitalizaciones de alta calidad. Si

se intenta almacenar imágenes de tipo vectorial o dibujos sencillos no reales, se observará como la compresión disminuye enormemente, y las modificaciones hechas sobre la imagen original por el algoritmo de compresión se observan a simple vista. La abreviación JPG viene de las iniciales de ***Joint Photographic Experts Group.*** Se trata del grupo de expertos que definieron las bases de este formato. El formato JPEG sólo puede almacenar imágenes de 24 bits (*truecolor*), utilizando tres canales para su almacenamiento o de escala de grises. La compresión JPEG consiste en una serie de complejas operaciones matemáticas, tales como: conversión del formato del color, transformación separada de coseno (DCT), cuantizaciones y codificaciones entrópica. JPEG, junto con GIF, son los formatos de imágenes usados en Internet.

2.5 Imágenes binarias

En una imagen binaria, cada píxel asume un valor discreto; esencialmente dichos valores corresponden a uno ó cero, encendido ó apagado. Por lo tanto una imagen binaria se almacena en un arreglo de píxeles de ceros y unos. La Figura 2.3, muestra una imagen binaria donde se proyecta una porción para observar sus valores.

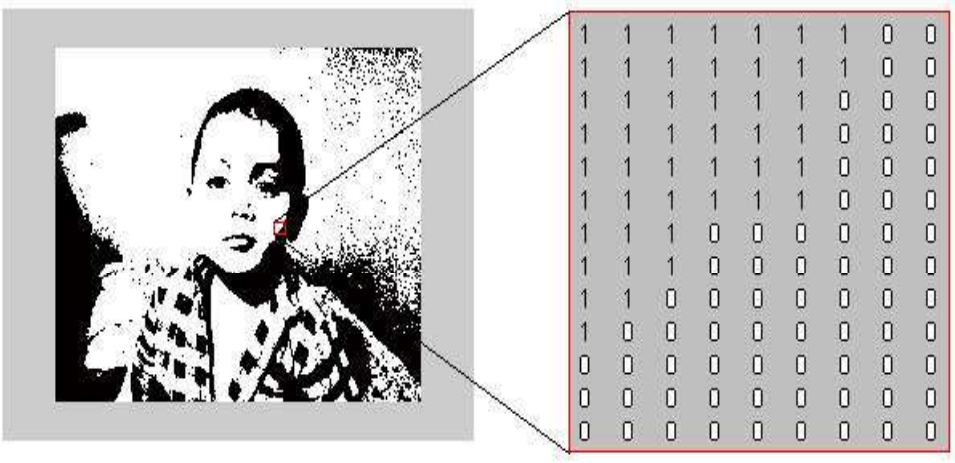


Figura 2.3 Imagen binaria.

2.6 Conectividad

La conectividad, es un concepto utilizado para establecer los límites de objetos en regiones dentro de una imagen. Para determinar si dos píxeles están conectados, se investiga si son adyacentes en algún sentido y si sus niveles de grises satisfacen un criterio de similaridad (por ejemplo si son iguales). En una imagen binaria con valores de unos y ceros, dos píxeles pueden ser vecinos, pero se dice están conectados sólo cuando tienen el mismo valor. En la Figura 2.4, se presenta una imagen binaria ubicada en el plano x, y, donde el píxel con coordenadas (x,y) se encuentra conectado con sus vecinos de coordenadas (x+1,y-1), (x-1,y) y (x,y+1), ya que satisfacen la condición de ser vecinos y poseer el mismo valor.

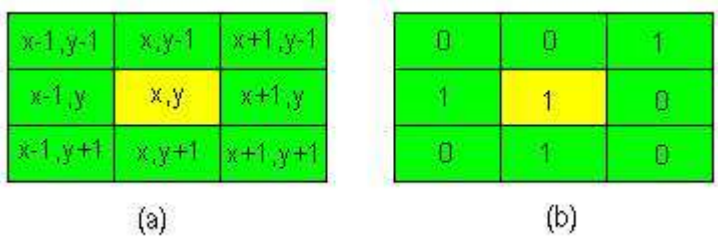


Figura 2.4 (a) Imagen ubicada en el plano de coordenadas (x,y), (b) Imagen binaria con píxeles conectados.

El análisis de conectividad en imágenes binarias, generalmente se utiliza para etiquetar píxeles conectados entre sí y determinar la cantidad de objetos contenidos en una imagen.

Al fijar el sentido y cantidad de píxeles adyacentes al momento de realizar un análisis de conectividad, determina la cantidad de objetos conectados, ya que no se generara la misma información de salida, si exclusivamente se toma en cuenta el píxel siguiente como conectado a que se tomen en cuenta todos los presentes a su alrededor.

2.6.1 Dilatación

La *dilatación* $A \oplus B$, es el conjunto de puntos de todas las posibles sumas de pares de elementos ceros de cada conjunto A y B, de acuerdo a la siguiente ecuación que define la dilatación de una imagen A en relación a un elemento estructural B:

$$A \oplus B = \bigcup_{\beta \in B} (A)_\beta \tag{2.3}$$

La Figura 2.5 muestra un ejemplo de la aplicación de la técnica de dilatación a una imagen binaria.

Figura 2.5. Dilatación de una imagen binaria con un elemento estructural de 1x2.

Por ejemplo dados A y B en forma de vectores para los valores correspondientes a los índices de los ceros de la imagen de la Figura 2.5, cuyas coordenadas se definen con respecto a (0,0), y $A \oplus B$ nos precisa en forma de vector el resultado de aplicar la suma vectorial de A más B, obtenemos los siguientes resultados.

$$A = \quad 0,1 \ , \ 1,2 \ , \ 2,2 \ , \ 3,2 \ , \ 4,2 \ , \ 4,3 \ , \ 4,4 \ , \ 5,1 \ , \ 6,1 \ , \ 7,1$$

$$B = \quad (0,0),(0,1)$$

$$A \oplus B = \begin{Bmatrix} (0,1),(1,2),(2,2),(3,2),(4,2),(4,3),(4,4),(5,1),(6,1),(7,1) \\ (0,2),(1,3),(2,3),(3,3),(4,3),(4,4),(4,5),(5,2),(6,2),(7,2) \end{Bmatrix}$$

En los conjuntos *A* y *B*, de la Figura 2.5, se considera que *A* es la imagen y *B* es el elemento *estructural*. El elemento *estructural* es en morfología matemática lo que la máscara (o núcleo) de convolución lo es en los filtros lineales. Los elementos estructurales más comunes son los conjuntos que están 4-conectados (N_4), y 8-conectados (N_8), sin embargo el elemento *estructural* puede variar de acuerdo a las necesidades que se presenten. La Figura 2.6, ilustra a dos de los elementos estructurales más comunes.

Figura 2.6 A) Elemento estructural N_4 , B) Elemento estructural N_8 .

La operación de *dilatación* hace que los objetos dentro de una imagen se expandan y la cantidad o la forma en la que crezcan dependen del elemento estructural que se elija [7]. Esta operación también se puede aplicar varias veces para obtener mayor expansión en los objetos, hasta obtener los resultados deseados.

La Figura 2.7, muestra una imagen representativa de ceros y unos, la cual se expande al aplicar la técnica de *dilatación,* produciendo tendencia a la unión de las dos siluetas contenidas en la imagen, aplicando un elemento estructural N_4 .

Al igual que en la Figura 2.5, en la Figura 2.7, el conjunto *A* es la imagen y *B* es considerado el elemento estructural que en este caso es N_4 , se detectan las coordenadas de la imagen *A* que contienen valor igual a uno y de acuerdo al

elemento estructural utilizado, se les proporciona el mismo valor a todos los píxeles que se encuentren adyacentes al píxel analizado, generando con ello expansión en las siluetas dentro de la imagen.

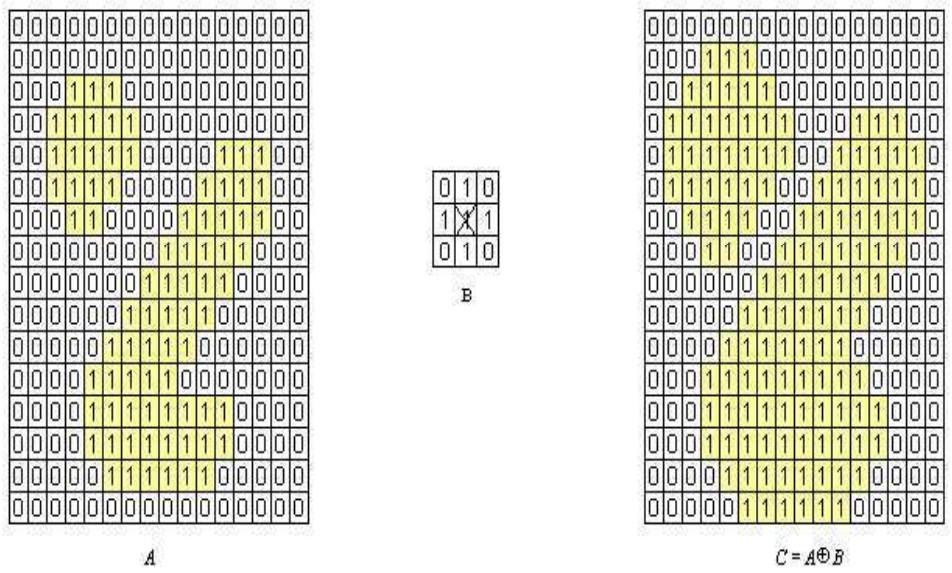


Figura 2.7 Dilatación: A) Imagen original, B) Elemento estructural N_4, C) Imagen dilatada.

Es relevante mencionar que una imagen que ya ha sido procesada mediante el método de dilatación, es posible volver a dilatar, es decir se puede dilatar una imagen *n* veces, con la limitante respecto a el tamaño de la imagen y el objeto dilatado, ya que llegará el momento en que toda la silueta o siluetas quedaran esparcidas en toda la imagen.

Por ejemplo en la Figura 2.7, si deseamos que se unan las dos siluetas, se puede utilizar un elemento estructural más grande o bien erosionar varias veces la imagen.

La Figura 2.8, muestra la expansión de la imagen mostrada en la Figura 2.7 (A), aplicando ahora un elemento estructural N_8, se puede notar que el

resultado presenta mayor esparcimiento de unos, logrando con ello se unan las dos siluetas formando una sola.

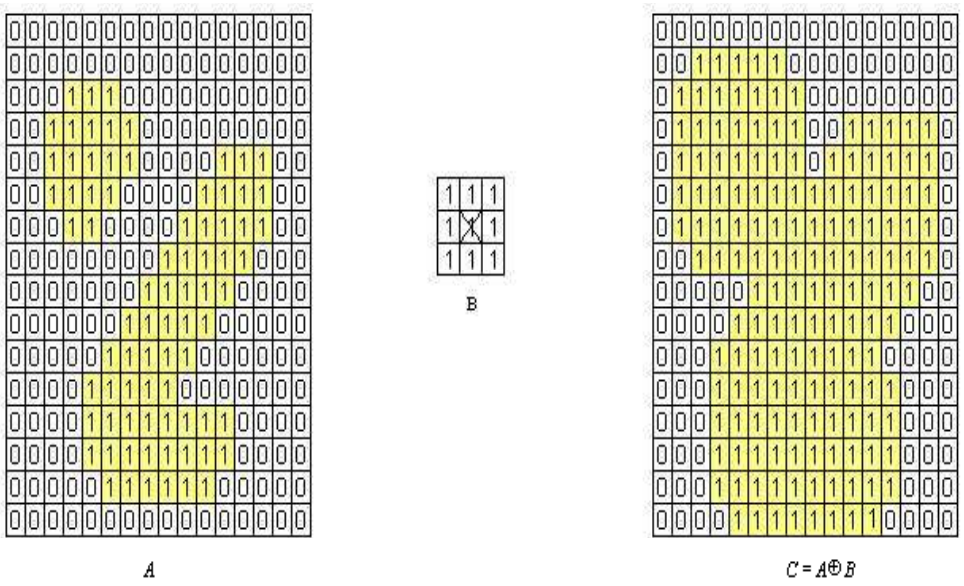


Figura 2.8 Dilatación: A) Imagen original, B) Elemento estructural N_8, C) Imagen dilatada.

2.6.2 Erosión

La *erosión* combina dos conjuntos usando la resta vectorial y es el dual de la dilatación.

La *erosión* se define con la siguiente ecuación y es útil para minar siluetas dentro de una imagen binaria.

$$A \ominus B = \bigcap_{b \in B} (A - b) \tag{2.4}$$

Donde *A* se refiere a la imagen a procesar y *B* es el elemento estructural.

En otras palabras, son los puntos *A* para los cuales todas las posibles translaciones definidas por *B* también están en *A* [7]. La Figura 2.9, muestra la *erosión* de una imagen y debido a que esta técnica es el dual de la *dilatación* los

valores de los índices de los vectores a analizar son ocupados en nuestro caso por los unos, que forman el entorno de las siluetas.

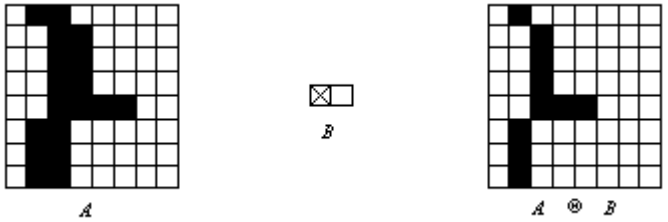


Figura 2.9 Erosión de una imagen utilizando un elemento estructural de 1x2.

Si las imágenes de la Figura 2.9, las representamos en forma de vectores, cuyas coordenadas se definen respecto a (0,0), donde los vectores contengan los índices de las coordenadas de los valores uno, podemos operar sobre ellos para definir los nuevos valores de los índices que contengan los nuevos valores uno, que debe de contener la imagen erosionada y $A \ominus B$ nos precisa en forma de vector el resultado de aplicar la resta vectorial de los índices de A menos B, obteniendo los siguientes vectores.

$$A = \left\{ \begin{array}{l} (0,0),(0,3),(0,4),(0,5),(0,6),(0,7),(1,0),(1,1),(1,4),(1,5) \\ (1,6),(1,7),(2,0),(2,1),(2,4),(2,5),(2,6),(2,7),(3,0),(3,1) \\ (3,4),(3,5),(3,6),(3,7),(4,0),(4,1),(4,6),(4,7),(5,0),(5,3) \\ (5,4),(5,5),(5,6),(5,7),(6,0),(6,3),(6,4),(6,5),(6,6),(6,7) \\ (7,0),(7,3),(7,4),(7,5),(7,6),(7,7) \end{array} \right\}$$

$$B = (0,0),(0,1)$$

$$A\Theta B = \left\{ \begin{array}{l} (0,0),(0,3),(0,4),(0,5),(0,6),(0,7),(1,0),(1,1),(1,4),(1,5) \\ (1,6),(1,7),(2,0),(2,1),(2,4),(2,5),(2,6),(2,7),(3,0),(3,1) \\ (3,4),(3,5),(3,6),(3,7),(4,0),(4,1),(4,6),(4,7),(5,0),(5,3) \\ (5,4),(5,5),(5,6),(5,7),(6,0),(6,3),(6,4),(6,5),(6,6),(6,7) \\ (7,0),(7,3),(7,4),(7,5),(7,6),(7,7) \\ (0,0),(0,2),(0,3),(0,4),(0,5),(0,6),(1,0),(1,0),(1,3),(1,4) \\ (1,5),(1,6),(2,0),(2,0),(2,3),(2,4),(2,5),(2,6),(3,0),(3,0) \\ (3,3),(3,4),(3,5),(3,6),(4,0),(4,0),(4,5),(4,6),(5,0),(5,2) \\ (5,3),(5,4),(5,5),(5,6),(6,0),(6,2),(6,3),(6,4),(6,5),(6,6) \\ (7,0),(7,2),(7,3),(7,4),(7,5),(7,6) \end{array} \right\}$$

Y eliminado las coordenadas que se repiten en el resultado $A\Theta B$.

$$A\Theta B = \left\{ \begin{array}{l} (0,0),(0,3),(0,4),(0,5),(0,6),(0,7),(1,0),(1,1),(1,4),(1,5) \\ (1,6),(1,7),(2,0),(2,1),(2,4),(2,5),(2,6),(2,7),(3,0),(3,1) \\ (3,4),(3,5),(3,6),(3,7),(4,0),(4,1),(4,6),(4,7),(5,0),(5,3) \\ (5,4),(5,5),(5,6),(5,7),(6,0),(6,3),(6,4),(6,5),(6,6),(6,7) \\ (7,0),(7,3),(7,4),(7,5),(7,6),(7,7) \\ (0,2),(1,3),(2,3),(3,3),(4,5),(5,2),(6,2),(7,2) \end{array} \right\}$$

Resultando el vector anterior con coordenadas de la ubicación de los píxeles de la imagen *erosionada* con valor uno de la Figura 2.9.

La Figura 2.10 muestra el resultado de aplicar erosión a la Figura 2.7 (A) con un elemento estructural N_8.

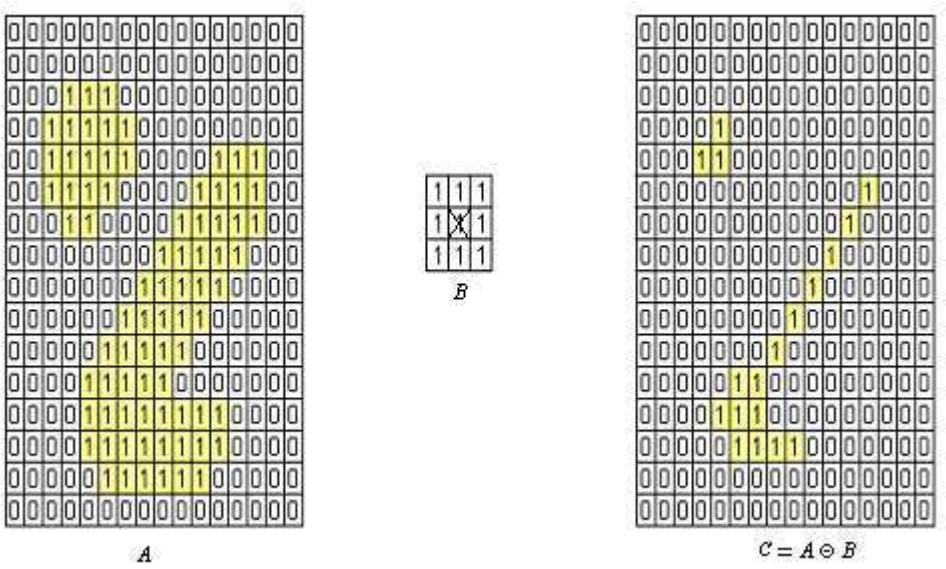


Figura 2.10 A) Imagen original, B) Elemento estructural de 3x3, C) Imagen erosionada.

2.6.3 Apertura

La *apertura* de *A* por *B* aplicada a imágenes, es la *erosión* de *A* menos *B* seguida de la *dilatación,* utilizando en ambas operaciones el mismo elemento estructural, definida por la siguiente ecuación.

$$A \circ B = (A \ominus B) \oplus B \tag{2.5}$$

La *apertura* en general, se utiliza para alisar el contorno de un objeto, rompe uniones pequeñas, uniones entre objetos y elimina componentes ruidosas (puntos blancos). En la Figura 2.11, se utiliza la *apertura* como un filtro, de esta manera se elimina objetos pequeños menores a un elemento estructural y también ruido contenido en la imagen, el elemento estructural que se utilizó es de 4x4 y se aplicó 3 veces.

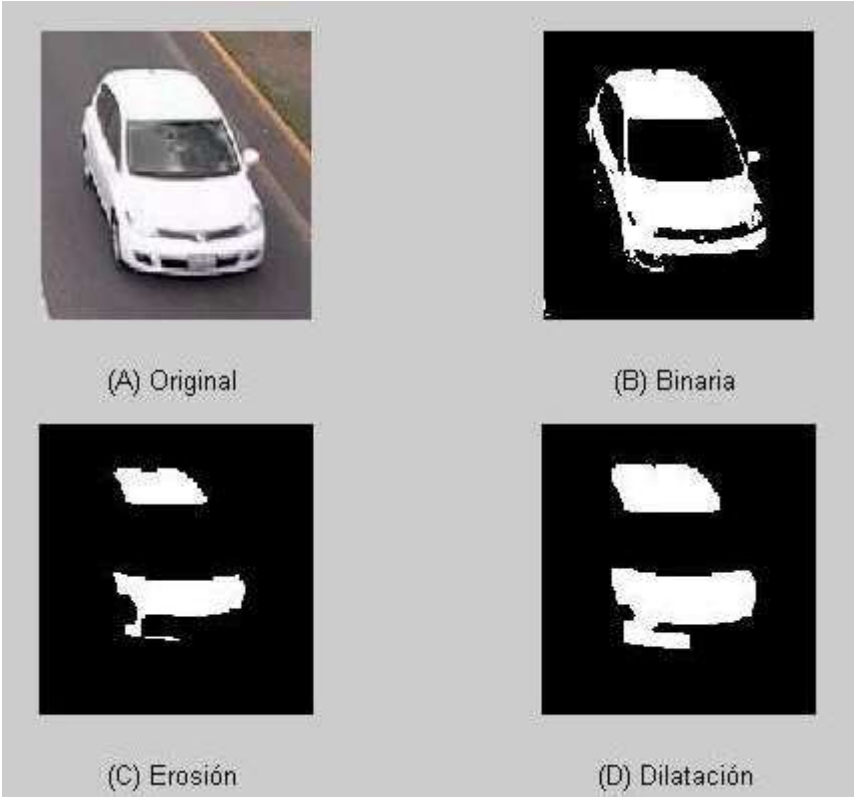


Figura 2.11 Apertura: A) Imagen original, B) Imagen binaria, C) Imagen resultado de la aplicación de la técnica erosión, D) Imagen resultado de la aplicación de la técnica dilatación.

2.6.4 Cerradura

La *cerradura* de *A* por *B* aplicada a imágenes, es la *dilatación* de *A* más *B* seguida de la *erosión*, utilizando en ambas operaciones el mismo elemento estructural. Y se define bajo la siguiente ecuación.

$$A \bullet B = (A \oplus B) \ominus B \tag{2.6}$$

La *cerradura* tiende a eliminar pequeños agujeros (puntos negros) del objeto, fusiona pequeños rompimientos y rellena pequeñas entradas al objeto [8], como se muestra en la Figura 2.12, donde se aplicó la técnica de cerradura a la imagen original de la Figura 2.11 (B) utilizando el mismo elemento estructural de 4x4 empleado 3 veces.

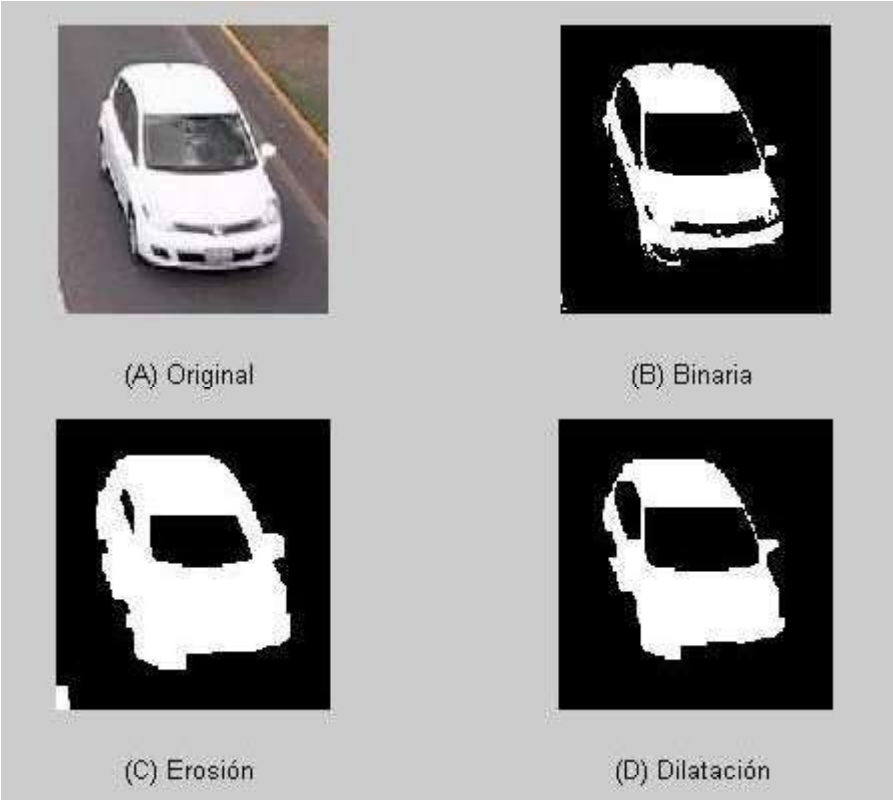


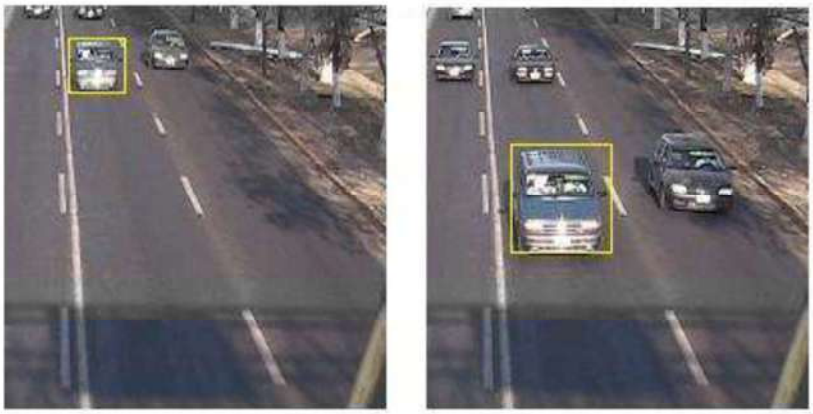
Figura 2.12 Cerradura: A) Imagen original, B) Imagen binaria, C) Imagen resultado de la aplicación de la técnica dilatación, D) Imagen resultado de la aplicación de la técnica erosión.

En general las técnicas de *apertura* y *cerradura* pueden ser aplicadas de acuerdo al interés de los resultados que se requieran, generando diferentes resultados en las imágenes procesadas a partir de una original, aun utilizando el mismo elemento estructural pero aplicado varias veces.

2.7 Distorsión de perspectiva en imágenes

La distorsión de perspectiva en imágenes, se presenta debido a la diferencia de profundidad de los objetos presentes en una imagen bidimensional, donde el tamaño de los objetos se encuentra relacionado con la ubicación espacial dentro de la imagen captada respecto a un observador, por ejemplo, la Figura 2.13, muestra como debido al fenómeno de distorsión de perspectiva, un automóvil

ubicado más cerca de la cámara como se muestra en la Figura 2.13 (B), obtiene un mayor número de píxeles respecto al mismo automóvil ubicado a superior distancia de la cámara tal como se muestra en la Figura 2.13 (A).



A B

Figura 2.13 Imágenes con distorsión de perspectiva.

Si deseamos hacer un procesado *espacial,* en las imágenes de la Figura 2.13, debemos de presentar especial atención a la situación, de que la cantidad de píxeles que representan al automóvil marcado con rectángulo amarillo de la imagen de la Figura 4.13 (A), es menor a la cantidad de píxeles que representan al mismo automóvil ubicado a menor distancia de la cámara como se presenta en la imagen de la Figura 4.13 (B).

Para hacer una rectificación de perspectiva se puede hacer uso de la técnica de *rectificación métrica,* de acuerdo con Domingo Mery [10]. Mediante esta técnica es posible modelar la distorsión geométrica que le ocurre a un plano que es visto por una cámara perspectiva. Con esta transformación algunas propiedades se conservan como la colinearidad (líneas rectas son vistas como líneas rectas) mientras que otras propiedades no (las líneas paralelas por lo general no son vistas como líneas paralelas), como se muestra en la Figura 3.14, donde se hace una corrección de perspectiva a la imagen.

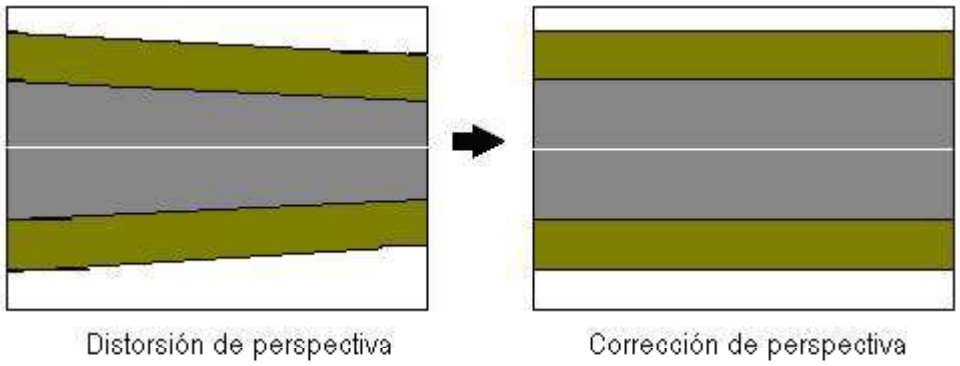


Figura 2.14 Corrección de perspectiva.

Capítulo 3

Procesamiento de imágenes usando Matlab®

MatLab (*MATrix LABoratory*), es un entorno interactivo basado en matrices para la realización de cálculo numérico y visualización de resultados. Permite la resolución de problemas sencillos sin escribir un programa y con facilidad de representación gráfica de los resultados. Además incorpora un lenguaje de programación que permite implementar programas complejos de modo relativamente simple.

Actualmente MatLab se usa tanto a nivel académico, dentro de la Universidad, como a nivel de investigación e industria para la resolución de complicados problemas científicos o de ingeniería. Es empleado para el desarrollo de cálculo numérico de propósito general, logrando manipular vectores y matrices tanto reales como complejas con funciones y fórmulas de variadas ramas de la matemática, y resolución de problemas con formulación matricial que aparecen en *Control, Estadística y Procesado de Señales*. MatLab aporta por medio del *toolbox,* funciones para resolver problemas específicos, como por ejemplo procesado de señales, diseño de sistemas de control, identificación de sistemas, simulación de sistemas dinámicos, optimización, redes neuronales, etc. [11].

3.1 Manejo de imágenes en MatLab®

La estructura básica de datos en MatLab es el arreglo, el cual se puede definir como un conjunto ordenado de datos reales o complejos. En el caso de las imágenes, estas pueden ser representadas por matrices formadas por conjuntos ordenados de valores reales que representan la intensidad de color o de niveles de grises.

MatLab almacena la mayoría de las imágenes como arreglos bidimensionales (matrices), en las cuales cada elemento de la matriz corresponde a la intensidad de un píxel de la imagen. Por ejemplo, una imagen de 200 renglones por 300 columnas se almacena en MatLab como una matriz de 200x300. Algunas imágenes, como las imágenes a color (*RGB*), requieren de un arreglo tridimensional, donde el primer plano representa la intensidad de rojo de los píxeles, el segundo plano representa la intensidad de verde de los píxeles y el tercer plano representa la intensidad de azul de los píxeles. La Figura 3.1, muestra como se encuentran distribuidos los colores del tipo *RGB* en el plano tridimensional.

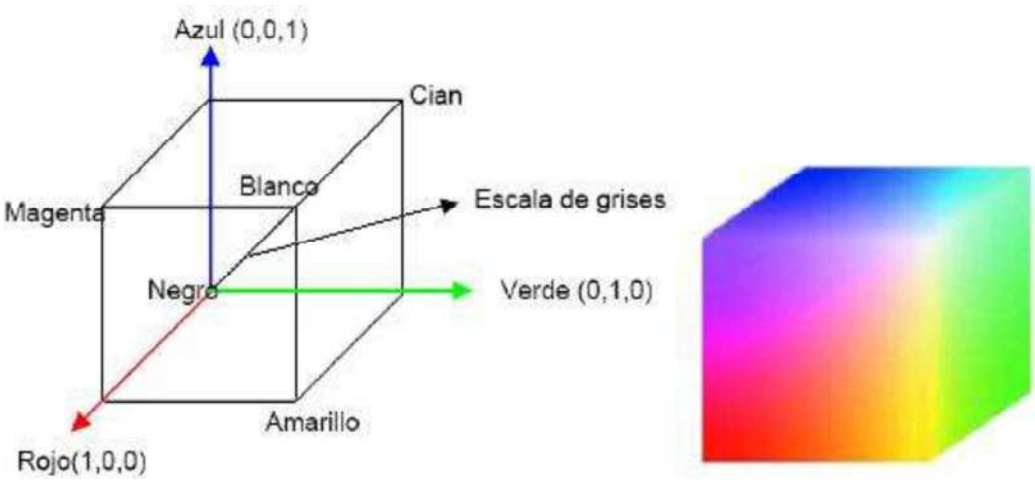


Figura 3.1 Distribución de colores *RGB*.

Esta convención hace que el trabajar con imágenes en MatLab sea similar a trabajar con matrices con datos de cualquier tipo. Por ejemplo, se puede seleccionar un sólo píxel de una imagen-matriz de la forma I(2,15), con lo cual MatLab regresa el valor del píxel de una imagen localizado en el renglón 2, columna 15 de la imagen-matriz I.

3.2 Clases de almacenamiento usadas en el *toolbox* de MabLab®

Por *default*, MatLab almacena la mayoría de los datos en la clase *double*. Los datos en estos arreglos se almacenan como datos de punto flotante de doble presición (64 bits).

En el caso de las imágenes, esta representación no es la ideal, debido a que en una imagen se tiene un número amplio de píxeles. Por ejemplo, si se tiene una imagen de 1000 x 1000 píxeles, se tiene un millón de píxeles y debido a que cada píxel se representa con al menos un elemento del arreglo, se requerirán aproximadamente 8MB de memoria para almacenarla.

Para reducir el espacio en memoria requerido para almacenar imágenes, MatLab almacena los datos en arreglos de 8 o 16 bits sin signo, clases *uint8 y uint16*, respectivamente. Estos arreglos requieren al menos la octava o cuarta parte de la memoria requerida por un arreglo tipo *double*.

3.3 Conversiones entre tipos de imágenes

Para ciertas operaciones es necesario convertir una imagen de su tipo original a otro tipo de imagen que facilite su procesamiento. Por ejemplo, si se desea filtrar una imagen a color almacenada como imagen indexada, primero se debe convertir la imagen a formato *RGB*. Esto es para que MatLab filtre los valores de intensidad de la imagen de forma apropiada. Si se intenta filtrar una imagen *indexada*, el filtro simplemente se aplica a los índices que se encuentran en la matriz *indexada* y los resultados no serán los deseados [8].

3.4 Lectura de imágenes en formato *JPG* mediante Matlab®

La lectura de imágenes en formato *JPG* en MatLab, se produce mediante el comando ***IMREAD***, obteniendo tres matrices bidimensionales con valores de los colores *RGB* respectivamente. Por ejemplo, para una imagen donde colocamos los valores del plano *G*=0, *B*=0, sin modificar el valor de los píxeles del plano *RED,* podemos visualizar los valores de la matriz que contiene el color rojo, de igual forma se pueden manipular los valores para los planos *GREEN* y *BLUE*. En la Figura 3.2, podemos observar como una imagen a color del tipo *RGB* se encuentra compuesta por tres matrices bidimensionales, que representan la intensidad de los colores *rojo, verde* y *azul*.

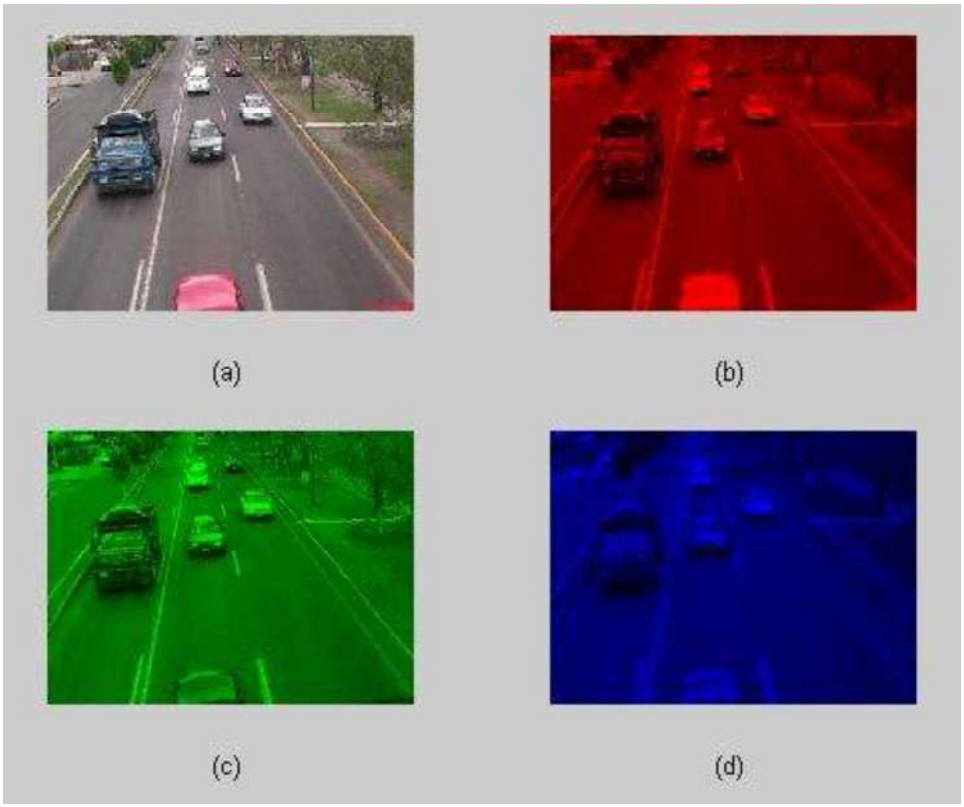


Figura 3.2 (a) Imagen original, (b) Componente rojo, (c) Componente verde, (d) Componente azul.

Es importante destacar, que en las imágenes de la Figura 3.2, podemos observar el color de un sólo plano debido a que se conservan las tres componentes, pero algunas de ellas con valor cero, ya que si se presenta la imagen como una matriz bidimensional es decir únicamente tomando una componente sin alterar sus valores, se podría observar la imagen en tono de grises como se muestra en la Figura 3.3 donde se extrajo exclusivamente una de las matrices *RGB* sin mostrar alguna otra de sus componentes.

Figura 3.3 (a) Imagen original, (b) Componente rojo, (c) Componente verde, (d) Componente azul.

En la imagen de la Figura 3.3 (b), podemos observar que el automóvil de color rojo presente en la imagen original de Figura 3.3 (a), muestra mayor valor visual, debido a que esta imagen representa en escala de grises la componente

de color rojo y por lo tanto los objetos de color rojo tienen mayor valor numérico en los píxeles de la imagen.

Debido al gran costo computacional al momento de procesar imágenes por la gran cantidad de píxeles en forma de datos obtenidos, se han buscado opciones para obtener información de las tres componentes que se presentan en una imagen *RGB* sin dejar de tomar en cuenta los datos numéricos de los tres planos, por lo que una opción se refiere al cálculo de la intensidad de la imagen para obtener una matriz bidimensional, la cual se puede calcular mediante la siguiente ecuación.

$$I = \frac{R + G + B}{3} \tag{3.1}$$

Donde:

R = Plano bidimensional *RED*.

G = Plano bidimensional *GREEN*.

B = Plano bidimensional *BLUE*.

I = Nueva imagen bidimensional.

Aun cuando cada plano contiene información específica de los tres colores de una imagen, al determinar la *intensidad* o también llamada *brillo de la imagen*, delimitamos el procesamiento a únicamente una matriz bidimensional que contiene características de los tres planos [7].

3.5 Procesamiento de imágenes binarias en Matlab®

Matlab versión R12 cuenta con un comando para convertir imágenes del tipo *RGB* a binarias como lo es ***IM2BW***, sin embargo al producir la conversión necesariamente se tiene que definir un valor de umbral para determinar los píxeles que se han de convertir en cero o en unos, dependiendo esto del nivel que se elija, sin embargo este valor crea la diferencia entre perder información útil o no. En este proyecto, para evitar la pérdida de datos valiosos en la

identificación de flujo vehicular, se determina primeramente el fondo de el lugar monitoreado, de esta manera al compararse con una imagen que contenga autos que no formen parte de el fondo, se excluyan todos aquellos píxeles que se parezcan mucho en su valor, impidiendo la eliminación de autos con color que genere datos en los píxeles por debajo de un nivel de umbral seleccionado para la conversión a imagen binaria. Sin embargo un inconveniente generado por este método utilizado, es la eliminación de autos con color totalmente parecido al asfalto del lugar monitoreado.

3.5.1 Erosión y dilatación de imágenes en Matlab®

La erosión y la dilatación, como se menciono en el Capítulo 2, son dos técnicas usuales en el procesamiento de imágenes, Matlab R12 las integra en el *toolbox* mediante los comandos ***ERODE*** y ***DILATE***, con la finalidad de poder aplicarlas a cualquier matriz binaria *n* veces y obtener *erosión* o *dilatación* respectivamente, así como sus combinaciones de *cerradura* y *apertura* para cualquier matriz en general, o bien hacer uso del comando **BWMORPH** especializado en imágenes mediante la sintaxis:

```
BW2 = BWMORPH(BW1,OPERATION,N);
```

Donde:

BW1 = Imagen a procesar.

OPERATION = Operación a realizar sobre BW1.

N = Número de veces a aplicar la operación.

BW2 = Imagen resultado.

Por ejemplo, la Figura 3.4 muestra el resultado de *erosionar* y *dilatar* una imagen procesada a binaria utilizando los comandos ***ERODE*** y ***DILATE*** de Matlab R12.

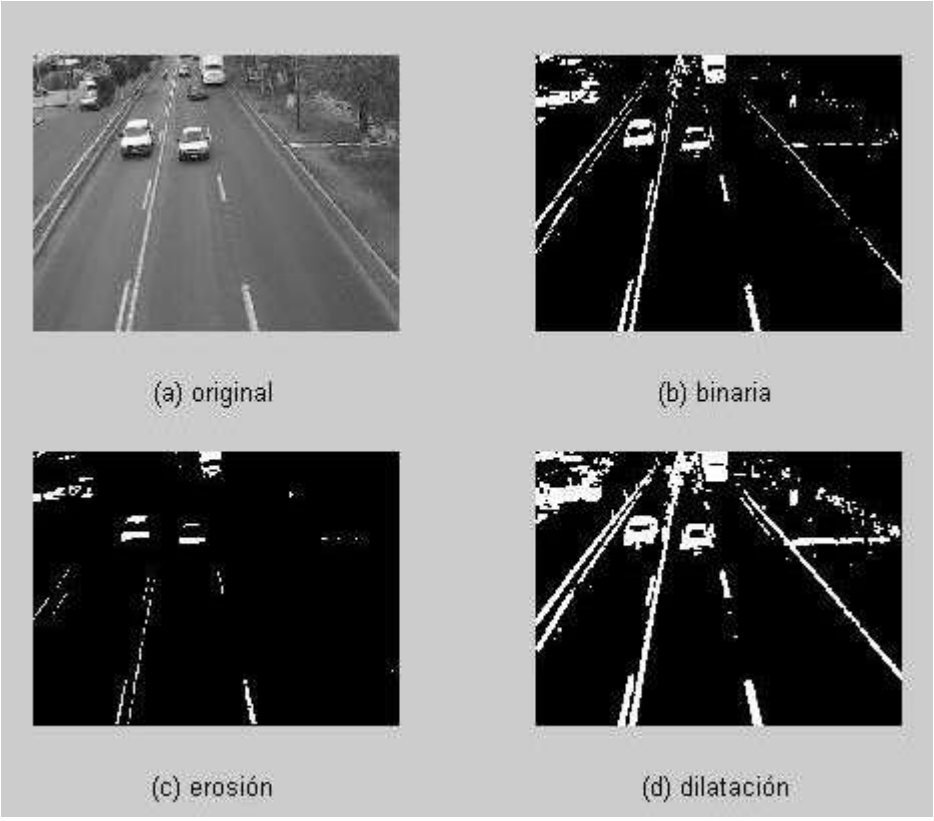


Figura 3.4 Erosión y dilatación de una imagen procesada a binaria.

3.5.2 Homogenización de imágenes binarias en Matlab®

En imágenes binarias, es común encontrarnos con siluetas que no se encuentran bien definidas dentro de la imagen, y a pesar de que no se localicen unidas a otra silueta o bien no contengan rompimientos, estas presentan espacios vacíos dentro de ellas, lo cual no es conveniente si se desea aplicar erosión o dilatación ya que se podrían generar uniones no deseadas o rompimientos que no sean convenientes en el intento de definir siluetas. Para estos casos MatLab dispone del comando ***BWFILL*** el cual logra generar en base a una imagen binaria de entrada, una imagen de salida sin orificios en las siluetas. Por ejemplo la Figura 3.5 muestra la homogenización de las siluetas que presentan espacios vacíos en la imagen original.

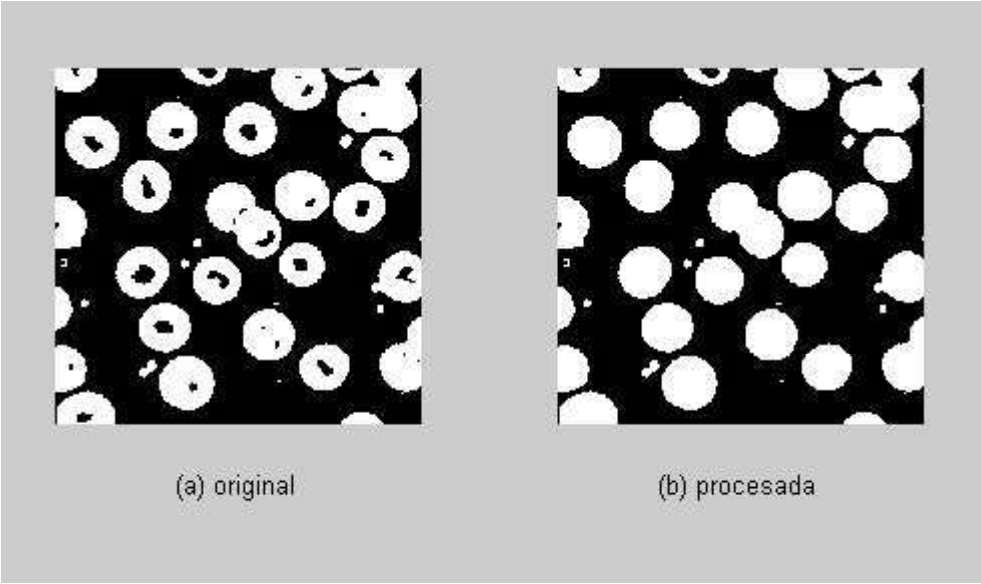


Figura 3.5 (a) Imagen original, (b) Imagen resultado de aplicar bwfill.

3.5.3 Conteo de objetos conectados en una imagen binaria utilizando Matlab®

Si contamos con una imagen binaria con siluetas de cuerpos bien definidos, es posible determinar la cantidad de objetos presentes dentro de la imagen, etiquetando los píxeles que se encuentren conectados, de esta manera el objeto etiquetado con el valor mayor, determina el número de objetos contenidos en la imagen. En MatLab es posible llevar el conteo de objetos en una imagen binaria mediante el comando ***BWLABEL*** el cual recibe una imagen binaria de entrada y regresa una matriz del mismo tamaño pero con el número de objetos etiquetados, además el valor máximo de los píxeles marcados que corresponde al número de objetos en la imagen.

La Figura 3.6, muestra el etiquetado de los píxeles con valor uno de la imagen-matriz mediante un análisis de conectividad, con un elemento estructural N_8, que determina si son iguales y si están conectados.

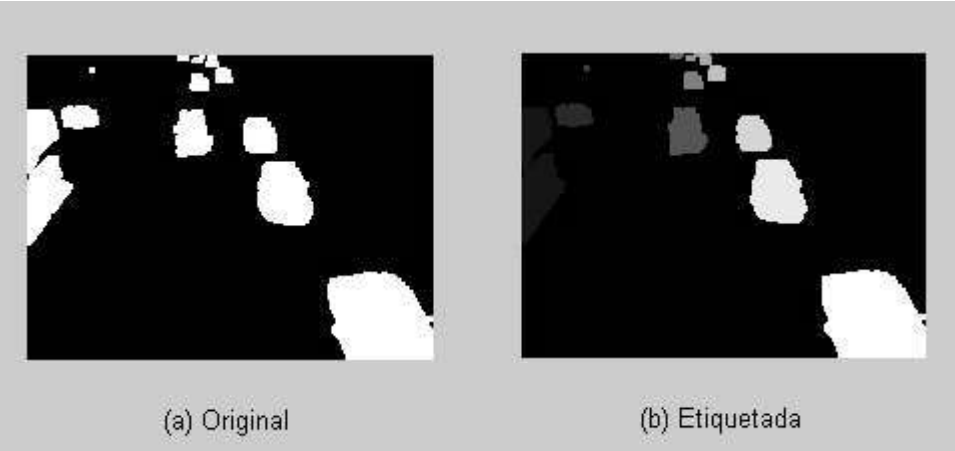


Figura 3.6 Etiquetado de una imagen binaria, (a) Imagen original, (b) Imagen etiquetada.

Al realizar el barrido de una imagen para realizar un análisis de conectividad, el etiquetado empieza por el primer píxel encontrado con las especificaciones deseadas, en nuestro caso con valor igual a uno y se determina que otros píxeles se encuentra conectado para asentarlos con valor igual a uno, de esta misma manera al encontrar otro píxel que no se encuentre conectado al primer píxel que se ubicó (o bien a los que se encuentren conectados con dicho pixel), se le asigna el valor de dos y se ubican todos los píxeles conectados con este nuevo pixel y entre ellos para asignarles el valor de dos, de esta manera se sigue con el barrido de la imagen hasta recorrer todos los píxeles contenidos en la imagen.

En el caso de la Figura 3.6 (b), para poder mostrar el resultado visualmente, se dividió el valor del pixel etiquetado entre el número de siluetas presente en la imagen, donde este valor corresponde al mayor valor de los píxeles etiquetados es decir 12, ya que los valores de una imagen con valores del tipo *double* oscilan entre cero y uno, por lo tanto el primer objeto encontrada en la Figura 3.6 (b) contiene el valor de 1/12 en sus píxeles y el ultimo objeto encontrado 12/12=1. Lo anterior se ejecutó con la finalidad de mostrar visualmente el resultado de etiquetar una imagen binaria. Una imagen etiquetada en realidad parecería binaria si se muestra en niveles de grises en el rango de cero y uno, ya que los píxeles etiquetados emprenden del valor uno en adelante.

En la imagen Figura 3.7, se observan porciones de las siluetas de los autos identificados utilizando el comando ***IMCROP,*** el cual se puede utilizar para seleccionar porciones de una imagen-matriz y mostrar sus valores.

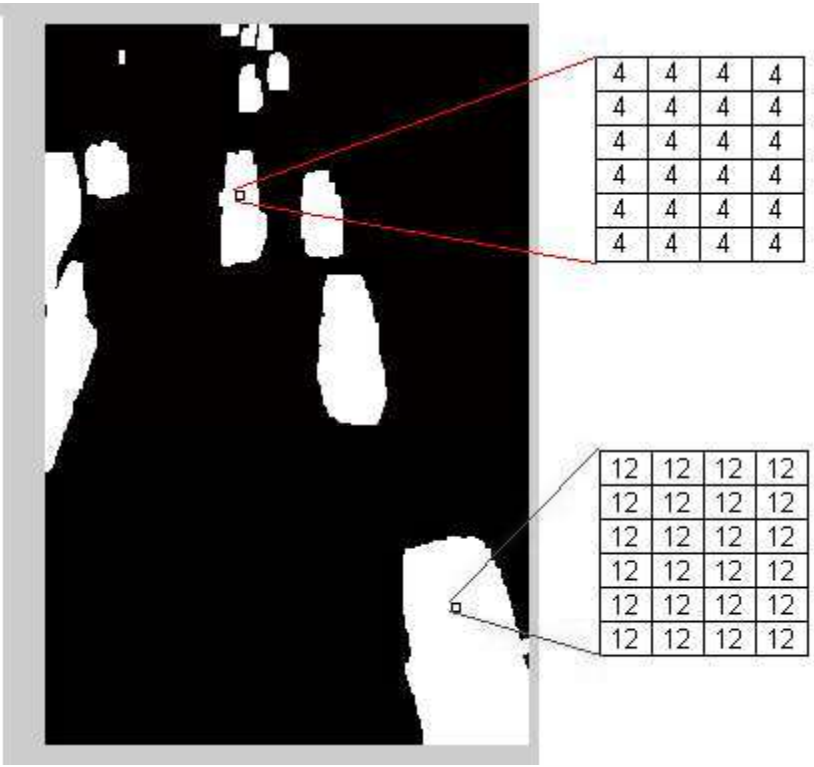


Figura 3.7 Imagen etiquetada y muestras proyectando valores.

3.5.3.1 Selección de regiones en una imagen binaria utilizando Matlab®

Si deseamos ubicar las regiones de los objetos conectados en una imagen binaria a la par de llevar a cabo un etiquetado, podemos también hacer uso del comando ***BWLABEL*** mediante la sintaxis:

[L,NUM] = BWLABEL(BW,N) ;

Donde BW es una imagen binaria que contiene siluetas y N el elemento estructural a utilizar, L contiene una matriz de las mismas dimensiones de BW con objetos etiquetados y NUM la cantidad de objetos conectados. A partir de esta información es posible aplicar el comando ***FIND***, este recibe un vector o matriz de los valores a ubicar y regresa los índices de los valores diferentes de cero. En la Figura 3.8, a la imagen (a) se le aplicó un análisis de conectividad con un elemento estructural N_4 y se etiquetaron todos los píxeles, obteniendo la imagen (b).

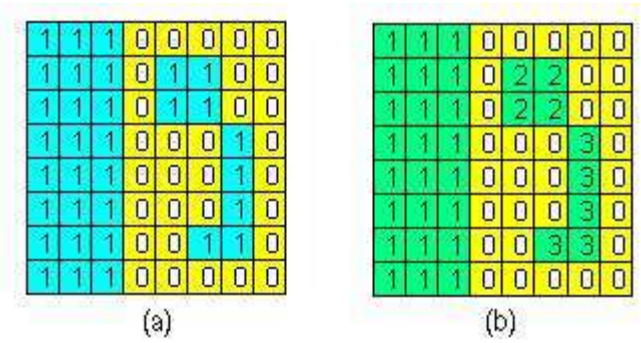


Figura 3.8 (a) Imagen original, (b) Imagen etiquetada.

Si ya tenemos píxeles etiquetados o bien ya tenemos bien definido cuales queremos ubicar, la sintaxis utilizada para el comando FIND en MatLab es:

```
[I ,J ,V] = FIND(L);
```

Donde:

L = Vector ó matriz a analizar.

V = Vector columna con valor del píxel ubicado en las coordenadas (I,J).

J = Vector columna que contiene la ubicación en valor, de la columna de los píxeles con valor diferentes de cero.

I = Vector columna que contiene la ubicación en valor del renglón de los píxeles con valor diferentes de cero.

También se puede especificar los valores a ubicar dentro de la matriz. Por ejemplo en la Figura 3.8 (b), si deseamos ubicar los índices de los píxeles con valor igual a tres, podemos utilizar en Matlab R12 la siguiente sintaxis.

[R,C]=find(L= =3)

Donde los valores de los índices obtenidos en R y C son:

R = [7; 4; 5; 6; 7]
C = [6; 7; 7; 7; 7]

De esta manera, obtuvimos la ubicación de los píxeles con el valor que deseamos ubicar, en la Figura 3.9, se marcan las coordenadas de los dos primeros píxeles ubicados de color rojo y azul respectivamente, para señalar los píxeles ubicados por el comando ***find*** en la Figura 3.8

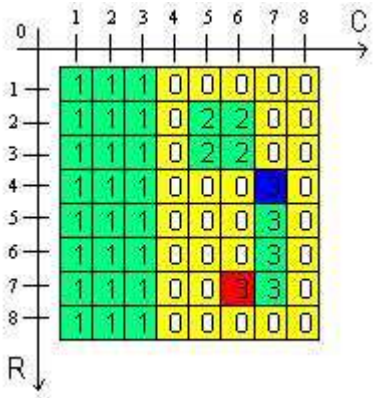


Figura 3.9 Píxeles ubicados en coordenadas (R,C).

Capítulo 4

Procedimiento para el registro de flujo vehicular y resultados

Las cámaras de video actualmente nos proveen de visión artificial, la cual puede ser almacenada y procesada en grandes cantidades, nos da la oportunidad de crear sistemas autónomos que en base a información proporcionada por el ser humano y la adquirida por ellos, tomen decisiones que se parezcan cada vez más a la de los seres humanos.

En base al gran avance tecnológico tanto de técnicas de procesado de imágenes y de hardware para el procesamiento de datos, en esta ocasión se pretende sentar las bases para la creación de un sistema que sin ninguna intervención humana realice conteo de flujo vehicular.

4.1 Obtención de imágenes

El proceso de captura de imágenes se realizó en Morelia Michoacán México, en el cruce de Periférico Paseo de la República Sector República con la Avenida Huaniqueo, su ubicación se muestra en el Apéndice D; sobre un puente peatonal ubicado a un costado del cruce vehicular, considerando una altura similar a los semáforos.

El dispositivo de video utilizado para la captura de video, fue una videocámara Canon® modelo ZR70. Posterior a la captura de videos se llevó a cabo la digitalización de imágenes con la misma, ya que cuenta con la función

de digitalizar imágenes de video. La primera fase del proceso consiste en digitalizar imágenes a partir de un video de 5 minutos de duración aproximadamente, en esta parte se extraen 6 imágenes consecutivas como se muestra en la Figura 4.1. Recordemos que una secuencia de video se trata en realidad de una secuencia de cuadros fijos [1].

Figura 4.1 Secuencia de imágenes tomadas en el cruce Periférico Paseo de la República Sector República, con la Avenida Huaniqueo en Morelia Michoacán.

4.2 Procesado de imágenes

Para el procesado de imágenes se siguieron los siguientes pasos:

1. **Obtención del fondo.-** Para determinar los píxeles que pertenecen al fondo en una secuencia de imágenes del mismo lugar, se hace la comparación píxel a píxel para determinar qué valores sufrieron cambio considerable en su valor respecto a los píxeles de las imágenes contiguas a lo largo de la secuencia, aquellos valores que no sufrieron cambio significativo se pueden tomar como estáticos ó fondo del lugar monitoreado.

2. **Comparación del fondo obtenido con una nueva imagen**.- A partir del fondo estimado, se puede hacer la comparación con una nueva imagen del lugar monitoreado, de esta manera los píxeles de la nueva imagen en donde se encuentran los autos no coincidirán en sus valores con los del fondo y serán excluidos.

3. **Binarización de la imagen comparada.-** A todos los valores que se discriminaron, debido al poco parecido de su valor entre ellos, se le asigna el valor de uno, mientras que a los restantes el valor de cero, de esta manera se crea una imagen binaria con siluetas de objetos no estáticos contenidos en la imagen.

4. **Eliminación de ruido.-** Debido a la presencia de ruido en la imagen obtenida, se aplica la técnica de la erosión seguida de la dilatación para la eliminación de ruido**.**

5. **Determinación de la cantidad de autos en la imagen binaria.-** La cantidad de autos presente en la imagen, se determinó utilizando el comando ***BWLABEL*** de MatLab R12 mediante análisis de conectividad.

4.2.1 Estimación del fondo

En general el fondo de una imagen lo podemos definir, para nuestro caso particular, como todo aquello que se encuentra estático en una secuencia de imágenes tomadas del mismo lugar en diferentes tiempos relativamente cortos. Existen diferentes técnicas para su estimación debido a que es un paso muy importante en los diferentes tipos de reconocimiento, de esta manera se puede conocer que no pertenece al fondo y aislar los objetos ó partes de la imagen de nuestro interés.

La propuesta en esta tesis para la estimación del fondo, se refiere a considerar conceptos estadísticos, analizando los píxeles que cuentan con el mismo valor en una secuencia de imágenes del mismo lugar y tomándolos como parte del fondo, en esta parte se tomaron seis imágenes de aproximadamente diez segundos entre una y otra, procesándolas al mismo tiempo y calculando la *mediana* para cada uno de los píxeles y sus similares de coordenadas (x,y), considerando que las imágenes obtenidas se procesaron inicialmente para convertirlas a escalas de grises y no perder tiempo en analizar los tres planos RGB por separado[12].

Por ejemplo, dada la secuencia de imágenes presentes en la Figura 4.1 podemos determinar cuál es el fondo común presente en cada una de ellas y eliminar todo lo dinámico, generando una nueva imagen con píxeles que no cambiaron considerablemente en su valor. La Figura 4.2 muestra el fondo obtenido a partir de la secuencia de imágenes de la Figura 4.1.

El cálculo se realizó en MATLAB® (Versión 6.0.0.88 (R12) *on* PCWIN) comparando cada uno de los píxeles $I_1(1,1),\ldots..I_n(i,j)$, donde n es el número de imágenes, en este caso n=6, y donde i, j los renglones y columnas respectivamente de la imágenes I_n.

Figura 4.2 Fondo estimado a partir de la secuencia de imágenes de la Figura 4.1.

4.2.1.1 Robustez en la estimación del fondo

Extraer continuamente el fondo del sitio monitoreado, conduce a obtener mejor certeza de que los cambios *climáticos* no afecten los valores de los píxeles debido a mayor o menor iluminación del lugar, detectando cambios la cámara en los valores de sus píxeles.

Otro aspecto a considerar en el proceso de extracción del fondo, consiste en establecer los autos que se encuentran en movimiento. Como se describió anteriormente, el proceso es dinámico en caso contrario el programa lo tomaría como parte del fondo, eliminando toda posibilidad de contar un auto más de una vez si este no se mueve, en la Figura 4.3 se muestran fallas que el programa toma como parte del fondo, las cuales se marcan con rectángulos rojos.

Figura 4.3 Fallas marcadas por incidencia automovilística tomadas por el programa como parte del fondo.

Al momento de calcular el fondo de una secuencia de imágenes, se pueden crear fallas debido a la incidencia de objetos, ya que por ejemplo, en nuestro caso en particular, en la secuencia de seis imágenes como se muestra en la Figura 4.1, varios automóviles aparecen en el mismo lugar, si tienen el mismo color serán tomados como parte del fondo ya que no habría cambios significativos en los valores de los píxeles o por lo menos en algunos de ellos.

4.2.2 Extracción del fondo y binarización de la imagen

El proceso de extracción del fondo y binarización de autos consiste en la comparación de los píxeles del fondo obtenido contra los valores de los píxeles de una nueva imagen del mismo lugar con autos presentes, y consiste en hacer las siguientes consideraciones:

a).- La resta de los píxeles de una imagen contra otra que contenga el fondo sin elementos adicionales, en condiciones ideales debe de ser cero. Si dos

imágenes se restan y estas tienen el mismo valor en sus píxeles el resultado debe de ser una imagen totalmente negra, es decir todos sus píxeles tienen valor cero.

b).- La resta de los píxeles de una imagen contra otra que contenga el fondo con objetos adicionales debe ser diferente de cero en condiciones ideales. Si se restan dos imágenes con diferentes valores en sus píxeles el resultado debe de ser diferente de cero ya sea positivo o negativo.

En la Figura 4.4, se ilustra a grandes rasgos como a una imagen con autos se extrae el fondo, para obtener como resultado una imagen con fondo negro, además de autos y ruido en color blanco.

Figura 4.4 Comparación de una imagen con su fondo para la eliminación de todo lo estático.

En la Figura 4.5, se amplía la imagen binaria, que se presenta como resultado de restar a la imagen original el fondo calculado, con la intención de poder observar con mejor detalle la presencia de ruido.

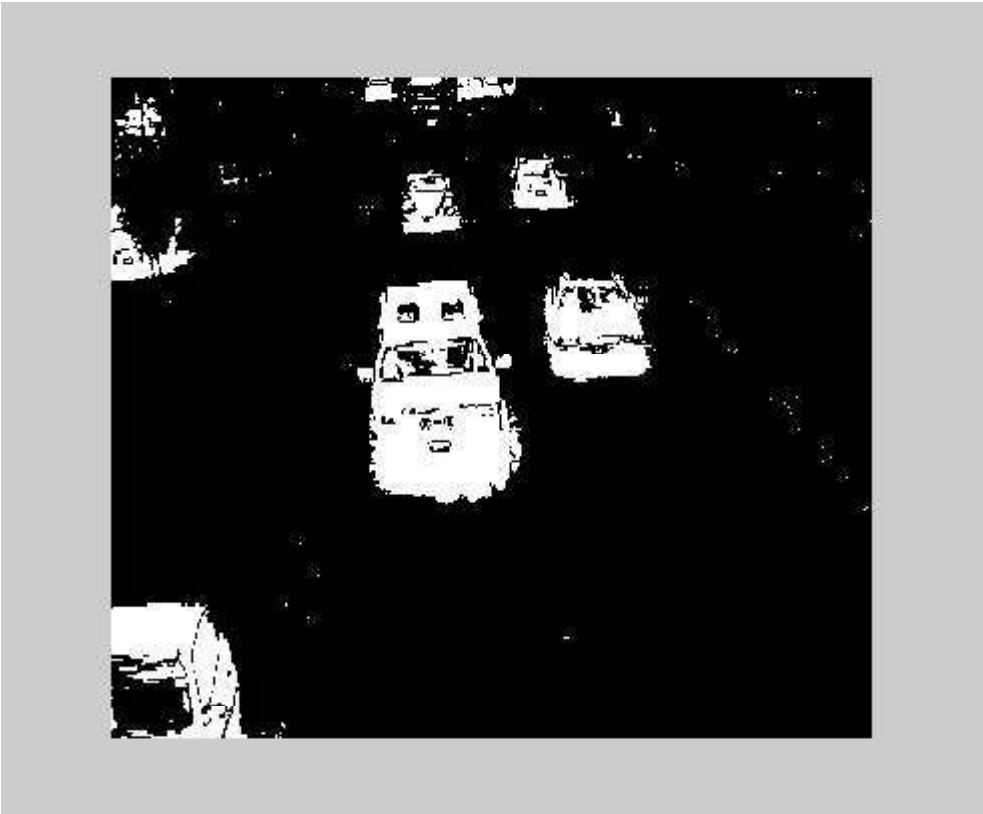


Figura 4.5 Imagen binaria con mayor detalle.

Podemos notar que la imagen de la Figura 4.5, no sólo contiene autos blancos con fondo negro sino que debido a pequeños cambios en el ambiente, movimiento ó bien debido a fallas en la extracción del fondo, se genero ruido en la imagen, lo cual no es deseable pero que podemos eliminar llevando a cabo la técnica de erosión seguida de la dilatación.

Cabe destacar que el lugar que se está monitoreando además de autos contiene siluetas de árboles que debido al movimiento de la cámara o de los árboles por el viento presente, genera cierto ruido en la imagen que no se tendría si sólo se monitorea asfalto con automóviles. En el caso de la imagen de la Figura 4.5, se puede notar que el movimiento fue en la cámara ya que no exclusivamente aparece ruido por parte de las ramas de los árboles si no que también en las líneas divisorias de los carriles, las cuales son estáticas, y debido a que también generaron ruido en la imagen, nos da un indicativo de que toda la

imagen tuvo cierto desplazamiento, sin embargo es innegable que entre menos movimiento de la cámara es factible obtener mejores resultados.

4.2.3 Proceso de homogenización

Los colores similares en los autos con respecto al asfalto originan que se generan espacios vacíos dentro de las siluetas de los autos, y tienen que ser procesados para lograr homogenizar la imagen. Para realizar esta operación se utiliza el comando ***BWFILL*** de Matlab, procesando la Figura 4.5, se logra una imagen de salida como se muestra a en la Figura 4.6.

Figura 4.6 Imagen binaria sin orificios en las siluetas.

4.2.4 Erosión de la imagen binaria

Aun cuando se han cubierto los espacios vacíos detectados dentro de las

siluetas de los autos, siguen presentes puntos que no son de interés, conocidos como ruido, el cual debe ser eliminado para no generar problemas con el conteo de objetos en movimiento. La técnica utilizada para eliminar esta información no deseada es la *erosión*. La Figura 4.7, muestra la *erosión* de la Figura 4.6, utilizando una matriz estructural 8-conectados (N_8 conectados) aplicada cinco veces.

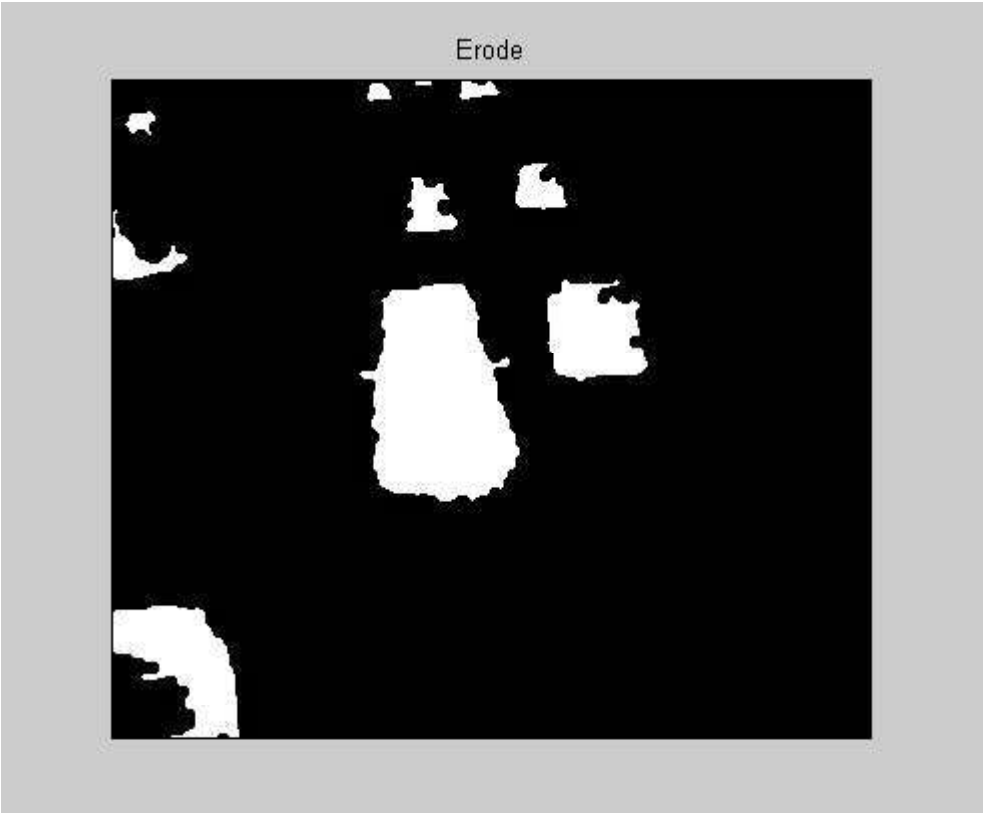


Figura 4.7 Imagen binaria erosionada.

Como se detalló en el Capítulo 2, la *erosión* se puede aplicar cuantas veces se desee sin embargo si se aplica repetidas ocasiones se puede erosionar demasiado la imagen y desaparecer por completo las siluetas presentes en la misma. También puede ocurrir que la erosión sea menor que no elimine el ruido por completo, de esta forma la aplicación de esta técnica estaría limitada por la

cantidad de ruido presente en la imagen, idealmente se desea que la imagen no contenga ruido, aun cuando esto casi es imposible una forma de evitar su presencia es la buena fijación de la cámara para que esta sea lo más estática posible.

4.2.5 Dilatación de la imagen binaria

La *dilatación* se aplica en general para enlazar rompimientos, sin embargo se debe poseer cuidado al igual que en la erosión, ya que se puede aplicar varias veces, pero se pueden dilatar demasiado las siluetas de la imagen que el resultado puede ser una imagen totalmente blanca, por otro lado si el proceso de dilatación se aplica un mínimo de veces se corre el riesgo de no entrelazar rompimientos, lo que ocasiona que un solo auto se cuente dos veces o más dependiendo de las partes que se generen a partir del auto analizado. La Figura 4.8, muestra la dilatación de la imagen de la Figura 4.7.

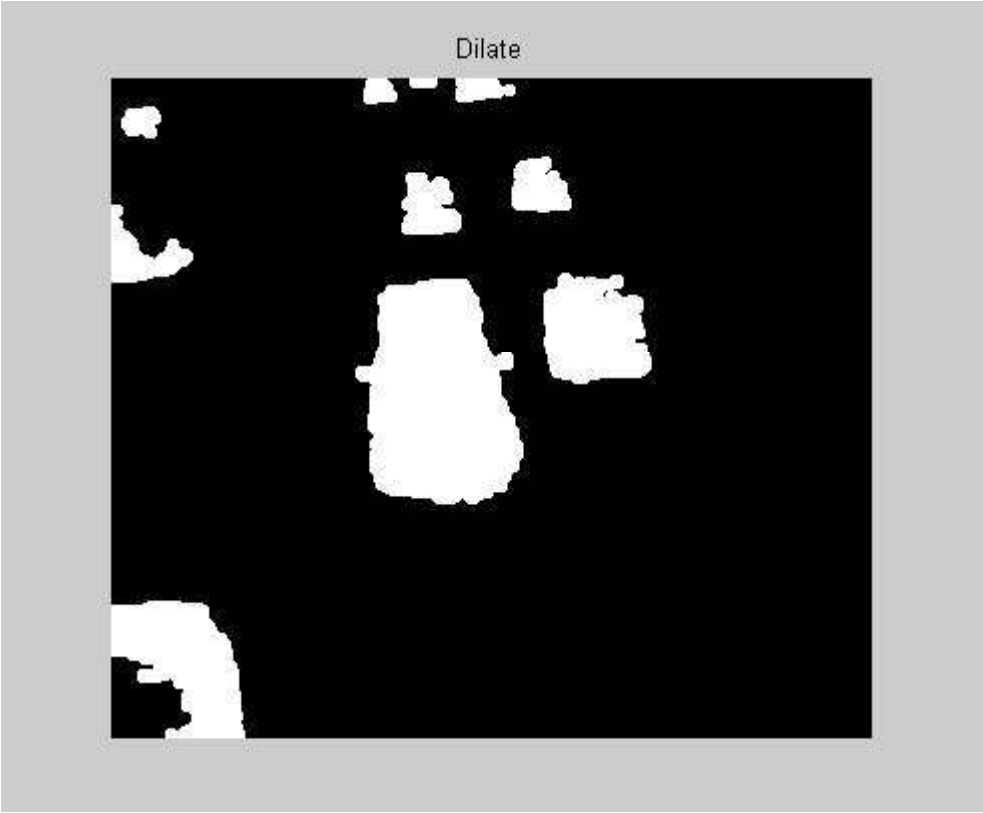


Figura 4.8 Imagen dilatada.

Incluso cuando el resultado en este caso en particular sea el mismo para efecto de conteo vehicular si se aplica dilatación o no, se debe hacer uso de la técnica, ya que es común que se generen rompimientos en las imágenes de los autos al aplicar *erosión* debido al parecido del color de algunas partes de los autos, con el asfalto monitoreado.

4.3 Resultados obtenidos

La imagen resultado presente en la Figura 4.8, al aplicarle análisis de conectividad mediante el comando ***BWLABEL,*** se obtuvo la cantidad de siluetas de autos presentes, donde el resultado que arroja MatLab es de diez autos detectados, si observamos la Figura 4.9 donde se presenta la imagen que se analizó, podemos apreciar que efectivamente son diez los automóviles presentes en la imagen los cuales están marcados con rectángulos amarillos.

Figura 4.9 Imagen analizada para conteo vehicular.

Podemos observar también, que aun cuando en la imagen mostrada en la

Figura 4.9, existen autos incompletos, se toman en cuenta, ya que los programas implementados en el Apéndice A, B y C no identifican autos, sino intensidad de flujo vehicular es decir cantidad de automóviles en movimiento.

También como paso adicional, se produjo la extracción de los autos que se detectaron mediante el programa, utilizando el comando ***FIND*** de Matlab ya que como se mencionó en el Capítulo 3, es posible extraer los índices de los píxeles de los autos detectados como se muestra en la Figura 4.10.

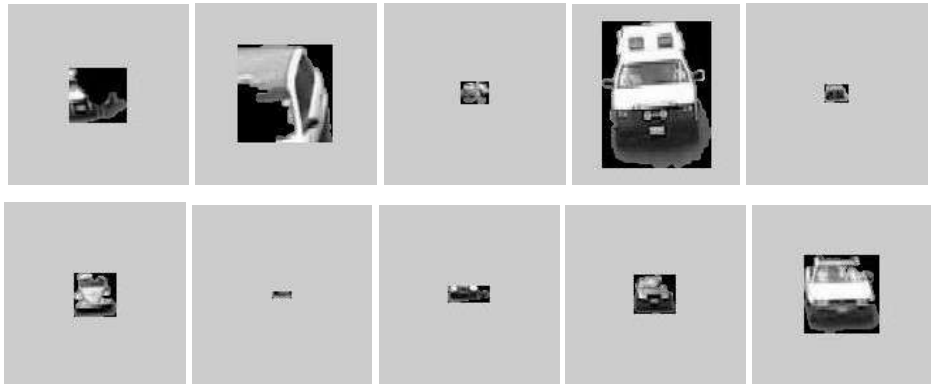


Figura 4.10 Autos detectados y separados de la imagen de la Figura 4.9 mediante aplicación del comando ***FIND*** de MatLab.

Hasta el momento, mediante el comando ***FIND*** se verifican los autos detectados para comprobar cuales fueron mostrados y aislados, sin embargo también nos podría servir para un estudio posterior de características de autos y obtener un resultado más fiable y no sólo llevar acabo conteo de flujo vehicular sino además un reconocimiento total del automóvil.

En la Tabla 4.1, se muestra los resultados de analizar veinte imágenes escogidas al azar para determinar que tan fiable es determinar el fondo en base a analizar el promedio de los tres planos *RGB,* para la comparación con el promedio de los tres planos de la imagen a contabilizar.

Tabla 4.1 Resultados de analizar veinte imágenes mediante el análisis del promedio de los tres planos que componen una imagen *RGB*.

Imagen	Conteo visual	Conteo del programa	Tiempo de procesado en segundos
1	8	7	67.1680
2	5	5	65.1329
3	0	0	65.0330
4	5	4	68.1680
5	4	4	65.9759
6	5	5	68.1680
7	4	4	65.7740
8	1	1	67.2970
9	2	2	65.7740
10	6	8	66.5060
11	7	8	67.2770
12	5	10	68.6190
13	0	0	68.6190
14	4	4	66.6860
15	6	9	68.1780
16	3	3	67.2760
17	5	6	189.5320
18	1	2	64.9644
19	3	4	65.4540
20	7	7	66.9270

Básicamente los errores de conteo por el programa marcados con verde en la Tabla 4.1, se debe al conteo de autos en más de una vez debido al parecido de su color con el asfalto, ó bien debido al traslape entre autos.

4.3.1 Balance para el registro de flujo vehicular mediante el promedio de los tres planos y los tres planos de las imágenes

En el Capítulo 3, se mencionó el uso de la Ecuación 3.1 para obtener una imagen bidimensional a partir de una imagen de tres dimensiones, esto con la finalidad de disminuir el tiempo de análisis de las imágenes, sin embargo, podemos pensar que es posible perder información si únicamente se analiza el promedio de una imagen del tipo *RGB,* a diferencia de tomar en cuenta los tres

planos por separado. En el Apéndice A se muestra el programa que analiza exclusivamente el promedio de una imagen del tipo *RGB* para la extracción del fondo del lugar monitoreado, mientras que en el Apéndice B, se presenta el programa que determina el fondo para cada plano *RGB* que se compara contra los tres planos respectivamente de la imagen que se va analizar, para la eliminación de la parte estática, por lo que determinaremos si es necesario decretar el fondo para los tres planos, ó si basta con analizar el promedio de los tres planos. Además se expresaran los tiempos de ejecución del programa para observar en cuanto aumenta el tiempo al analizar los tres planos. En la Figura 4.11, se muestran dos imágenes, donde la primera imagen (a); para determinar la cantidad de autos presentes, se analizó sólo un plano bidimensional, el cual es el promedio de los tres planos *RGB*, y en el caso de la segunda imagen (b) se analizaron por separado los tres planos para determinar la cantidad de autos presentes en la imagen original. La imagen original analizada poseía las dimensiones de: 380x330x3.

(a) (b)

Figura 4.11 (a) Imagen resultado de analizar un plano, (b) Imagen resultado de analizar los tres planos.

Al comparar visualmente las dos imágenes de la Figura 4.11, detectamos que existen pequeñas variaciones entre una y otra, sin embargo, el resultado que arrojo el programa es de *siete* automóviles detectados en ambos casos, y el

tiempo de procesado para la imagen (a) fue de 68.2280 segundos, mientras que para la imagen (b), el tiempo de procesado fue de 196.9330 segundos, por lo que podemos notar que efectivamente al procesar los tres planos *RGB* por separado, el procesado tarda prácticamente el triple. En la Figura 4.12, se muestra la imagen analizada donde efectivamente se detectan visualmente *siete* automóviles, los cuales son marcados con rectángulos color amarillo.

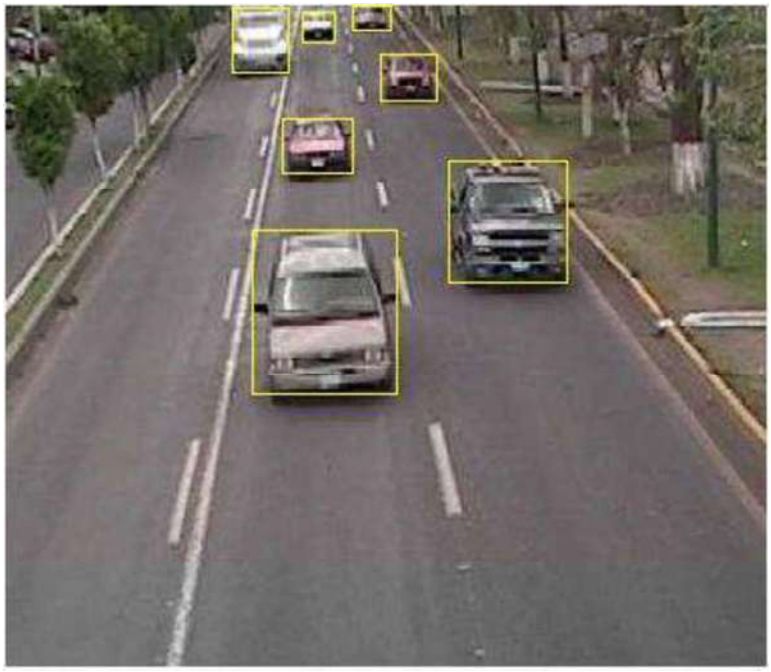
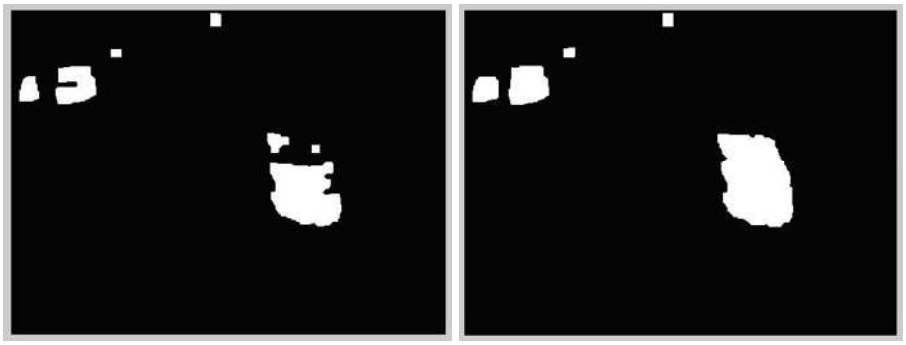


Figura 4.12 Imagen analizada.

Cabe mencionar que se utilizó para analizar las imágenes, una computadora portátil marca *DELL®* que cuenta con *Sistema Operativo Microsoft® Windows® Profesional XP versión 2002,* y el equipo cuenta con un *Procesador Intel Pentium III* a *701 MegaHertz* y *256 MegaBytes* de *RAM,* por lo que se puede mejorar el tiempo de procesado con equipos más actualizados.

Aun cuando se detectaron correctamente la cantidad de automóviles presentes en la imagen de la Figura 4.12, se determinaron más pruebas con diferentes imágenes escogidas al azar, encontrando que el programa que analiza

los tres planos de las imágenes ubicado en el Apéndice B, presenta mejor obtención de resultados. En la Figura 4.13 se presentan imágenes con diferencias de resultados, ya que la imagen (a) fue analizada como imagen bidimensional y la imagen (b) tomando en cuenta los tres planos que la definen como imagen a color. El tamaño de la imagen analizada es de 640x480 para el caso del resultado de la imagen (a) y para el caso de la imagen (b) es de 640x480x3.

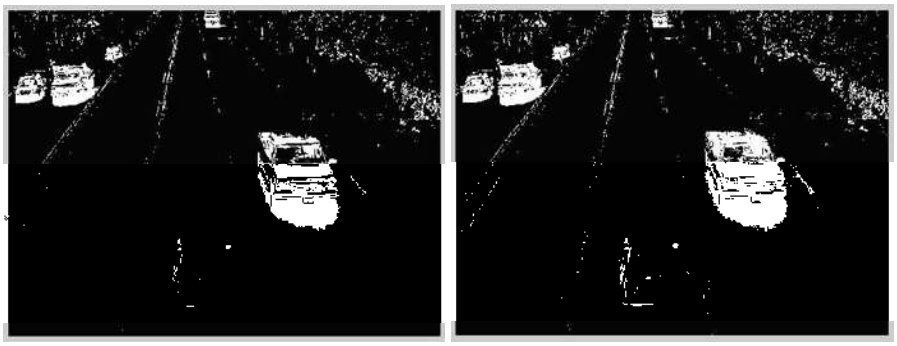


(a) (b)

Figura 4.13 (a) Imagen resultado de analizar un plano bidimensional, (b) Imagen resultado de analizar los tres planos de la imagen *RGB*.

En las imágenes de la Figura 4.13, el programa detecto para el caso de la imagen (a) siete carros, mientras que para la imagen (b) cinco carros, y el tiempo de procesado 368.0900 segundos y 130.3770 segundos respectivamente.

En la Figura 4.14, se presentan las imágenes resultado del paso inmediato a la extracción del fondo y binarización, donde se puede apreciar que en algunas partes de los automóviles debido al color, el valor de los píxeles es mucho mayor o mucho menor en sólo algún plano, y esto marca la diferencia entre crear imágenes binarias con las características completas de los tres planos, ***por lo que es necesario para disminuir las fallas en el conteo de flujo vehicular tomar en cuenta los tres planos***.



(a) (b)

Figura 4.14 (a) Imagen binaria tomando en cuenta el promedio de los tres planos, (b) Imagen binaria tomando en cuenta los tres planos por separado.

En la Figura 4.14, se puede notar que el automóvil ubicado al frente de la imagen (a), presenta diferencias en el parabrisas respecto al parabrisas del mismo automóvil de la imagen (b). Es decir la imagen (b) presenta en general características más completas de los automóviles detectados.

En la Figura 4.15, se observa la imagen analizada para las imágenes de las Figuras 4.13 y 4.14. Donde se marcaron los automóviles detectados con rectángulos amarillos.

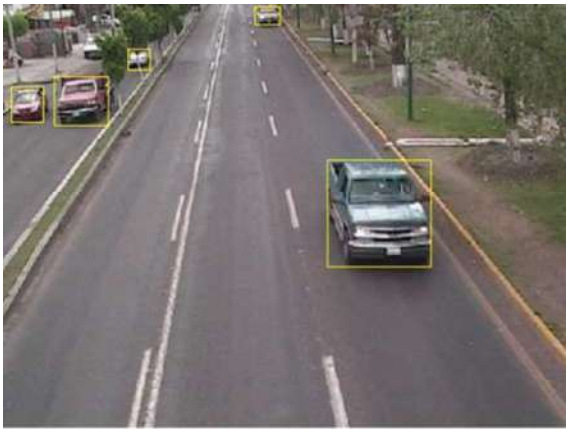


Figura 4.15 Imagen analizada de tipo *RGB*.

En la Tabla 4.2, se muestran los resultados de considerar veinte imágenes, analizando y comparando los tres planos por separado para comparar desempeño de los programas de los Apéndices A y B, se puede observar en los valores marcados con verde, que presenta mejores resultados el programa del Apéndice B en relación al programa del Apéndice A, donde el programa del Apéndice B, extrae el fondo para cada uno de los espacios *RGB* del lugar monitoreado y compara cada uno de ellos contra los tres planos respectivamente de la imagen a realizar el conteo de flujo vehicular.

Tabla 4.2 Resultados de analizar veinte imágenes mediante el análisis de los tres planos *RGB* por separado para la extracción del fondo y comparación de la imagen.

Imagen	Conteo visual	Conteo del programa	Tiempo de procesado en segundos
1	8	7	198.1290
2	5	5	193.4500
3	0	0	195.1100
4	5	5	187.4500
5	4	4	189.3320
6	5	5	186.6740
7	4	4	189.3320
8	1	1	186.4880
9	2	2	188.2300
10	6	6	189.3320
11	7	7	192.0160
12	5	7	189.8830
13	0	0	188.2710
14	4	5	189.8730
15	6	9	184.7050
16	3	3	197.9640
17	5	5	189.5320
18	1	2	185.9670
19	3	4	189.0610
20	7	8	190.6140

4.3.2 Balance para el registro de flujo vehicular mediante el análisis de la imagen por partes

Como se menciona en el Capítulo 2, el fenómeno de distorsión de perspectiva, se presenta en objetos con diferencia de profundidad dentro de una imagen. Debido al uso de las técnicas propuestas en este proyecto de tesis, que trabajan de manera *espacial* en las imágenes, es necesario tomar en cuenta la posibilidad de pérdida de datos, ya que las técnicas *erosión* y *dilatación* se aplican de igual manera a toda la imagen. Sin embargo se detecto visualmente que en la mayoría de las imágenes analizadas, el ruido con mayor tamaño en las imágenes, se encontraba precisamente en el fondo de la misma como se muestra en la Figura 4.16, donde se muestran cuatro imágenes que se les aplicó un mejoramiento para la eliminación del ruido, el cual es un paso esencial en la identificación de flujo vehicular.

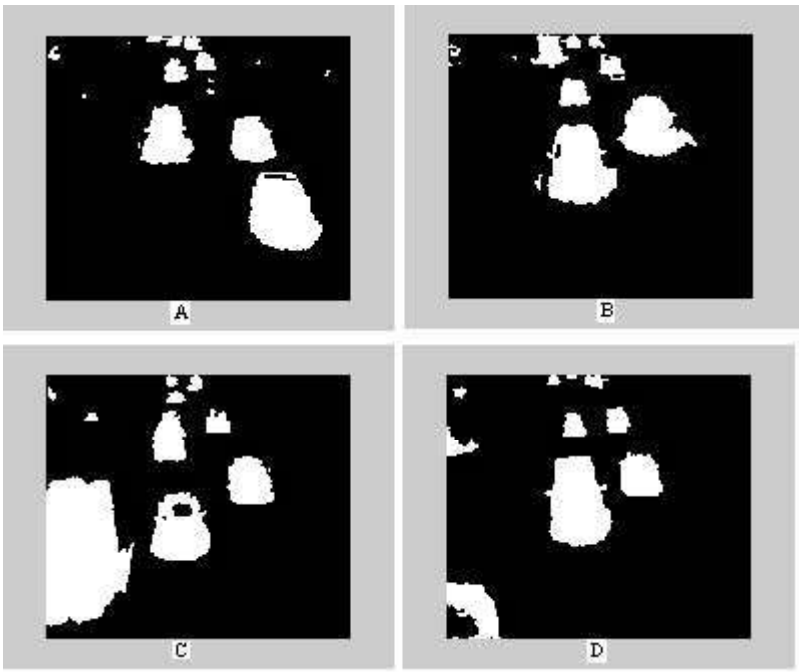


Figura 4.16 Imágenes con ruido.

Si observamos las imágenes presentes en la Figura 4.16, podemos apreciar

que el ruido presente, esencialmente se muestra al fondo de la imagen lo cual es contrario a lo que pudiéramos deducir en relación a especular que debido a la distorsión de perspectiva, el ruido presente en una imagen se mostraría con mayor número de píxeles al frente de la imagen. En la Figura 4.17, se muestra la imagen D de la Figura 4.16, un paso anterior del mejoramiento de la imagen binaria y se puede notar que el ruido presente en la imagen, es en general del mismo tamaño en toda la imagen.

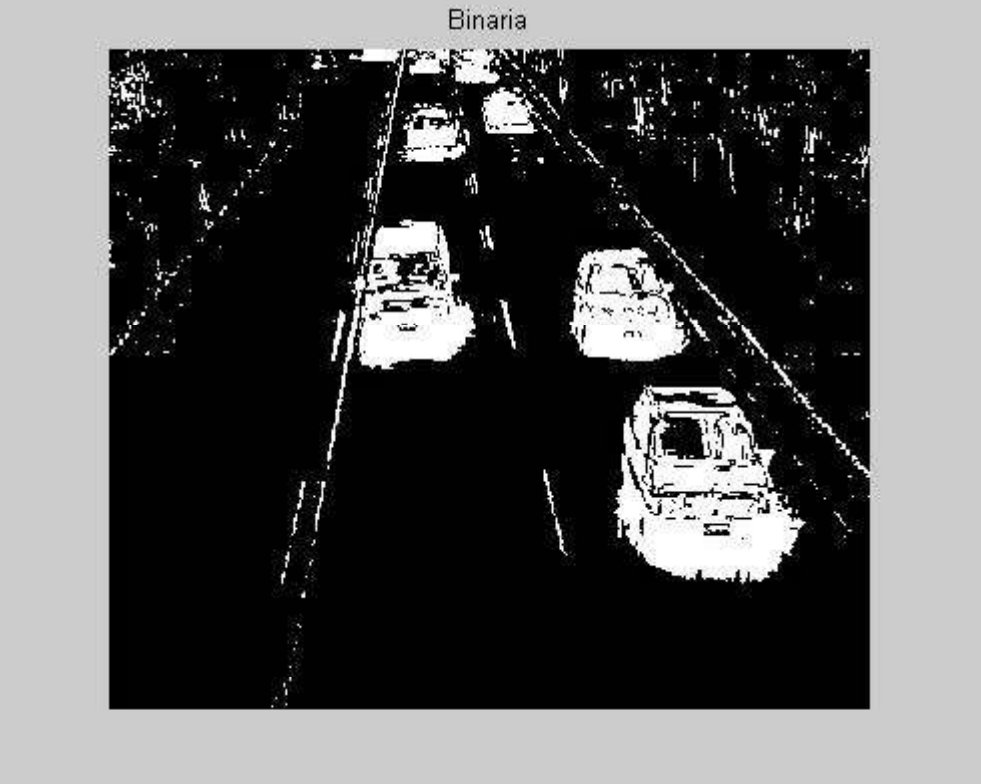


Figura 4.17 Imagen con ruido distribuido de manera regular.

Para determinar de manera más fiable si es necesario analizar el ruido presente en las imágenes mostradas se debe de comparar los resultados presentes en la Tabla 4.2 con los de la Tabla 4.3. El programa para analizar las imágenes con ruido distribuido de manera irregular se muestra en el Apéndice

C. Además para no llevar a cabo *corrección métrica* completa para cada una de las imágenes a examinar debido al gran costo computacional, se opto por analizar mediante *cerradura* por partes, es decir los objetos que se ubican al fondo, se les aplicaría una *cerradura* menor que los objetos que se encontraran al frente de la imagen, dividiendo la imagen en cinco partes, ya que se observo en las imágenes, que un auto completo ubicado al final de la imagen era cinco veces menor en tamaño aproximadamente que un auto ubicado al frente según la perspectiva de nuestras imágenes, tomando en cuenta que para definir en cuantas veces se ha de dividir la imagen se debe de considerar que los autos que se comparan tanto del fondo como el de frente de la imagen deben de ser de tamaño similar. En la Figura 4.18, se muestra los resultados de aplicar *cerradura* sin dividir la imagen para su estudio de ruido y su similar dividiendo la imagen para su análisis tomando en cuenta la distorsión de perspectiva.

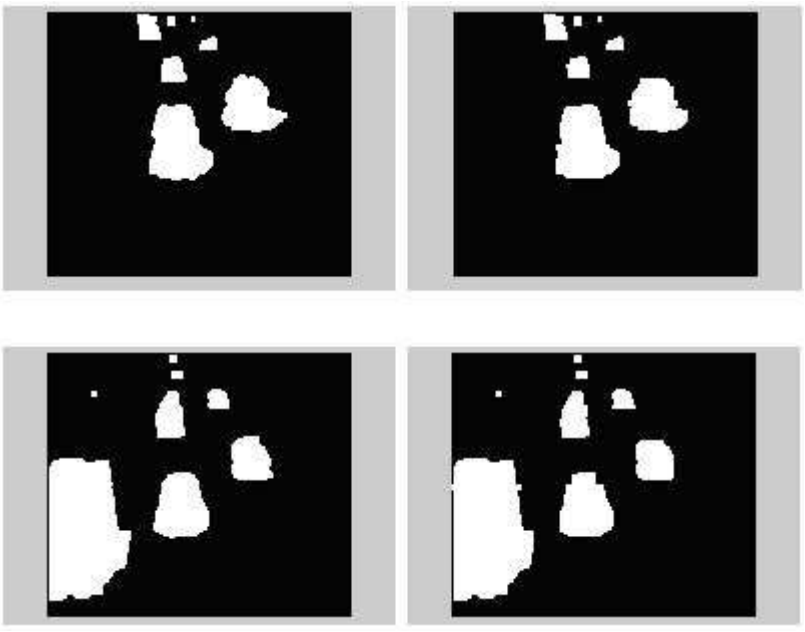


Figura 4.18 Imágenes aplicando cerradura a toda la imagen por igual a la izquierda, y a la derecha aplicando la cerradura por partes.

Los tiempos de procesado de las imágenes de la izquierda de la Figura 4.18, es de 140.0610 segundos, mientras que para las de la derecha fue de 140.2410 segundos, y el tamaño de las imágenes es de 330x380x3, los resultados de autos presentes en las imágenes fue de siete para las dos primeras imágenes y de 8 para las dos siguiente, y se puede notar que los resultados de tiempo de ejecución es similar, sin embargo se puede observar que las siluetas de los autos detectados en las imágenes analizadas por partes presentan mayor homogeneidad, en la Tabla 4.3, se muestran los resultados de analizar las mismas imágenes analizadas en la Tabla 4.1, para observar si hubo cambios significativos en los resultados al analizar la imagen binaria por partes y analizando los tres planos *RGB*.

Tabla 4.3 Resultados de analizar veinte imágenes mediante el análisis de los tres planos de la imagen y aplicando cerradura por partes.

Imagen	Conteo visual	Conteo del programa	Tiempo de procesado en segundos
1	8	8	189.8030
2	5	5	194.4500
3	0	0	191.6960
4	5	5	189.1020
5	4	4	190.0130
6	5	5	190.6740
7	4	4	188.1710
8	1	1	191.8860
9	2	2	192.8750
10	6	6	222.8010
11	7	7	200.5680
12	5	5	200.1080
13	0	0	199.7770
14	4	4	201.4000
15	6	7	206.0160
16	3	3	203.7130
17	5	4	189.7130
18	1	1	189.2220
19	3	3	187.9100
20	7	7	190.5540

De manera que se determina mediante los resultados anteriores, que para obtener siluetas más *homogéneas* y mejores resultados *se debe analizar las imágenes con cerradura por partes, es decir, presenta mejores resultados la aplicación de cerradura por partes aplicando mayor cerradura al ruido presente al frente de la imagen, que al ruido del fondo de la imagen.*

Otro aspecto detectado y que debemos de considerar, es que la aplicación de *cerradura por partes* presenta mejor desempeño para imágenes que han sufrido desplazamiento debido al movimiento de la cámara, además de que el tiempo de procesado es similar ya que básicamente se analiza la imagen simplemente por partes.

Capítulo 5

Conclusiones y trabajos futuros

La estimación de flujo vehicular en las grandes urbes, se ha convertido en una necesidad que no puede pasar desapercibida, debido al gran aumento de vehículos que en los últimos años se ha presentado en las grandes ciudades y como consecuencia de este gran aumento existe saturación de vehículos en intersecciones que cuentan con semáforos con tiempos definidos.

5.1 Conclusiones generales

Con la implementación de la extracción de fondo en una secuencia de imágenes adquiridas del mismo lugar en diferentes tiempos y de la utilización de técnicas existentes como lo son la *erosión*, la *dilatación* y el suavizado de siluetas con huecos rodeados de píxeles blancos, se **logró** la identificación de objetos que no forman parte del fondo pero que en un momento determinado estaban dentro del área monitoreada, por lo que, si además consideramos que en el lugar monitoreado sólo circulan autos, podemos determinar que todos los objetos identificados son autos, además al aplicar la *erosión* seguida de la *dilatación* se pueden eliminar objetos con tamaño menor a un auto y esto se puede lograr ajustando la cantidad de veces que se debe aplicar la técnica de *cerradura* filtrando en la imagen sólo objetos del tamaño de un auto.

En general este trabajo se puede extender a varias etapas posteriores que puede abarcar desde un estudio detallado de cuál es la mejor posición, ángulo, o distancia del área a monitorear para la identificación de flujo vehicular o bien a

la creación de *hardware* y *software* que utilice redes neuronales o lógica difusa para el control de semáforos que dadas las circunstancias de flujo, se tomen decisiones que ayuden a agilizar el tránsito vehicular. Además, como la propuesta de este proyecto de tesis presenta la extracción continua del fondo del lugar monitoreado fortalece la identificación de flujo vehicular para cualquier cambio en el ambiente, ya sea de iluminación, climáticos, o de lugar además se puede utilizar el ultimo fondo calculado para el nuevo cálculo del fondo estimado.

5.2 Conclusiones particulares

El objetivo fundamental se cumplió, sentando las bases para una etapa posterior en la creación de un sistema completo de monitoreo de flujo vehicular, toma de decisiones y control de semáforos que ayude a agilizar el flujo vehicular en intersecciones.

5.3 Trabajos futuros

El trabajo futuro inmediato es la creación de *hardware* que ayude a la implementación de semáforos inteligentes sobre la instalaciones existente para agilizar el flujo vehicular en intersecciones, además se podrían mejorar los programas propuestos ya que con el gran avance que se está obteniendo en el aumento de la velocidad del procesamiento de datos, se podría hacer la corrección de perspectiva de la imagen para determinar el tipo de vehículo detectado, y la creación de una base de datos para guardar la cantidad de autos detectados en determinadas horas y días, para poder establecer para horas pico la cantidad de autos que por lo general circulan a esa hora y días determinados.

Conjuntamente como se ha logrado la extracción y separación de los vehículos localizados, al existir vehículos aislados libres de características del entorno que podrían ser tomados como partes de un automóvil, se puede mejorar el programa buscando descriptores en los autos para poder identificarlos de manera más fiable y no confundirlos con motocicletas, animales o simplemente con peatones.

En general el hecho de que este programa extraiga el fondo de un lugar, origina sostén a aislar objetos que no se encuentren estáticos y por lo tanto la oportunidad procesar sólo objetos de interés que no se confundan con el entorno.

Referencias

[1] M. J. Maldonado, "SISTEMA AUTOMÁTICO DE CONTEO Y CLASIFICACIÓN DE FLUJO VEHICULAR BASADO EN SECUENCIAS DE VIDEO Y REDES NEURONALES ARTIFICIALES", Tesis de Maestría. Universidad Autónoma de Nuevo León, San Nicolás de los Garza N.L. México 2006, 234 p.

[2] R. Molina, "DEL PROCESAMIENTO A LA VISIÓN ARTIFICIAL" Programa Del Departamento Ciencias de la Computación e I.A., Universidad de Granada, 43 p.

[3] "MODELOS DE VISIÓN COMPUTACIONAL", 25/09/09 http://ftp.es.vim.org/vision/intro/node6.html

[4] http://es.wikipedia.org/wiki/InstitutoTecnol%C3%B3gicodeMassachusetts

[5] http://en.wikipedia.org/wiki/David_Marr_(neuroscientist)

[6] http://www.etsimo.uniovi.es/vision/intro/node6.html

[7] E. García, "DETECCIÓN Y CLASIFICACIÓN DE OBJETOS DENTRO DE UN SALON DE CLASES EMPLEANDO TÉCNICAS DE PROCESAMIENTO DIGITAL DE IMÁGENES", Reporte de actividades del proyecto de investigaciones, computación I, UNIVERSIDAD AUTONOMA METROPOLITANA, 27 De Noviembre del 2006, México Distrito Federal, 99 p.

[8] J. J. Esqueda, "FUNDAMENTOS DE PROCESAMIENTO DE IMÁGENES", Evento CONATEC 2002, Sede: Instituto Tecnológico de

Ciudad Madero, Noviembre 2002, Ciudad Madero Tamaulipas México, 50 p.

[9] http://es.wikipedia.org/wiki/p%C3%ADxel

[10] D. Mery, "VISIÓN POR COMPUTADOR", Departamento de Ciencia de la Computación, Universidad Católica de Chile, 17 de agosto de 2004, Santiago de chile.

[11] Y. A. Cerquera, "CURSO BÁSICO DE MATLAB", Programa De Ingeniería Electrónica, Universidad Surcolombiana, Enero 2007, Colombia, 62 p.

[12] Murray R. Spiegel, "PROBABILIDAD Y ESTADÍSTICA", México D.F.: McGRAW-HILL, 1992.

APÉNDICE A

CÓDIGO FUENTE

En este apartado, se presenta el código fuente del reconocimiento inteligente de flujo vehicular basado en el procesamiento de imágenes analizando el promedio de los planos *RGB*. Se muestra el programa principal y se incluyen las rutinas complementarias.

Programa: flujovehicular1planos.m

Descripción: Contiene el desarrollo del proceso de detección y conteo vehicular obteniendo el promedio de los tres planos *RGB* para su procesado. Se apoya en la rutina complementaria: convbin2.m. y aislar_autos.m

Entradas: Lee 7 imágenes en formato *jpg*. Contenidas en la carpeta *work*.

Salidas: Variable denominada *carros* con el valor de la cantidad de autos contenidos en la imagen a contabilizar sus autos. Así como autos aislados si se desactiva el comentario para la rutina complementaria aislar_autos.m.

Código fuente:

```
clear all; close all;  tic
%Se leen las imágenes a procesar para extracción de fondo
%Imagenes 1 y 2
```

```
I1=imread('foto36.jpg');                I2=imread('foto37.jpg');
I1=im2double(I1);                       I2=im2double(I2);
I1=(I1(:,:,1)+I1(:,:,2)+I1(:,:,3))/3;   I2=(I2(:,:,1)+I2(:,:,2)+I2(:,:,3))/3;
%Imagenes 3 y 4
I3=imread('foto38.jpg');                I4=imread('foto39.jpg');
I3=im2double(I3);                       I4=im2double(I4);
I3=(I3(:,:,1)+I3(:,:,2)+I3(:,:,3))/3;   I4=(I4(:,:,1)+I4(:,:,2)+I4(:,:,3))/3;
%Imagenes 5 y 6
I5=imread('foto40.jpg');                I6=imread('foto41.jpg');
I5=im2double(I5);                       I6=im2double(I6);
I5=(I5(:,:,1)+I5(:,:,2)+I5(:,:,3))/3;   I6=(I6(:,:,1)+I6(:,:,2)+I6(:,:,3))/3;
%Imagen a contabilizar sus autos
Im=imread('foto42.jpg');
Im=im2double(Im);
Im=(Im(:,:,1)+Im(:,:,2)+Im(:,:,3))/3;
%tamaño de las imagenes
[r,c]=size(I1);
%se calcula el fondo de las imágenes
for r1=1:r
 for c1=1:c
  vector=[I1(r1,c1),I2(r1,c1),I3(r1,c1),I4(r1,c1),I5(r1,c1),I6(r1,c1)];
  fondo(r1,c1)=median(vector);
 end
end
figure; imshow(fondo)
%Se comparan el fondo con la imagen con autos
binarizada=convbin2(Im,fondo);
figure; imshow(binarizada);title('Binaria');
%Se rellenan huecos
sinceros= bwfill(binarizada,'holes');
figure; imshow(sinceros);%imagen sin huecos
```

```
%Se mejora la imagen
contraria= bwmorph(sinceros,'majority',30);
figure; imshow(contraria);
contraria= bwfill(contraria,'holes');
contraria= bwmorph(contraria,'shrink');
figure; imshow(contraria);
binarizada= bwfill(contraria,'holes');
figure; imshow(binarizada);
%Se elimina ruido
cerradura= bwmorph(binarizada,'erode',3);
figure; imshow(cerradura);title('Erode')
cerradura= bwmorph(cerradura,'dilate',3);
figure; imshow(cerradura);title('Dilate 1 plano')
%Objetos conectados
[L,carros] = BWLABEL(cerradura);
I10=im2double(cerradura);
aislados(:,:,1)=Im.*I10;
figure; imshow(aislados)%autos detectados
carros  %cantidad de objetos conectados
%Se separan los autos detectados
%aislar_autos(carros,aislados,L)
toc
```

Rutina: convbin2.m

Descripción: Esta rutina compara una imagen que contiene el fondo estimado, con otra imagen del mismo lugar para eliminar todo lo que pertenece al fondo, haciendo 0 a los valores que se parezcan en su valor y 1 a aquellos que no pertenecen al fondo.

Entradas: Recibe dos matrices de datos del mismo tamaño, donde una previamente a sido estimada a partir de 6 imágenes es decir el fondo y la otra es la imagen a contabilizar sus vehículos.

Salidas: Retorna una matriz con datos binarios.

Código fuente:

```
function [imbinaria] = convbin2(imagen,fondo)
imbinaria=abs(imagen-fondo);
imbinaria=im2bw(imbinaria,.1);
```

Rutina: aislar_autos.m

Descripción: Esta rutina presenta por separado los autos detectados.

Entradas: Recibe la cantidad de autos detectados, la imagen con autos a contabilizar y la matriz etiquetada de las siluetas de los autos.

Salidas: Presenta los autos detectados por separado visualmente.

Código fuente:

```
function aislar_autos(carros,aislados,L)
for a=1:carros;
 [r,c] = find(L == a);
 tamano=length(r);
  for b=1:tamano;
   Iselec(r(b)-(min(r)-1),c(b)-(min(c)-1))=aislados(r(b),c(b),1);
  end
  figure; imshow(Iselec);
  Iselec=0;
end
```

APÉNDICE B

CÓDIGO FUENTE

En este apartado se presenta el código fuente del reconocimiento inteligente de flujo vehicular basado en el procesamiento de imágenes analizando los tres planos *RGB* por separado. Se muestra el programa principal y se utilizan las rutinas complementarias del Apéndice A.

Programa: flujovehicular3planos.m

Descripción: Contiene el desarrollo del proceso de detección y conteo vehicular obteniendo el promedio de los tres planos *RGB* para su procesado. Se apoya en las rutinas complementarias presentadas en el Apéndice A: convbin2.m. y aislar_autos.m

Entradas: Lee 7 imágenes de formato *jpg*. Contenidas en la carpeta *work*.

Salidas: Variable denominada *carros* con el valor de la cantidad de autos contenidos en la imagen a contabilizar sus autos. Así como autos aislados si se desactiva el comentario para la rutina complementaria aislar_autos.m.

Código fuente:

```
clear all; close all;
tic
```

```
%Se leen las imagenes a procesar para extraccion de fondo
%imagenes 1 y 2
I1=imread('foto13.jpg');        I2=imread('foto14.jpg');
I1=im2double(I1);               I2=im2double(I2);
%imagenes 3 y 4
I3=imread('foto15.jpg');        I4=imread('foto16.jpg');
I3=im2double(I3);               I4=im2double(I4);
%imagenes 5 y 6
I5=imread('foto17.jpg');        I6=imread('foto18.jpg');
I5=im2double(I5);               I6=im2double(I6);
%Imagen a contabilizar sus autos
Im=imread('foto19.jpg');
Im=im2double(Im);
%Tamaño de las imagenes
[r,c,d]=size(I1);
%Se calcula el fondo del lugar monitoreado
for d1=1:d
for r1=1:r
 for c1=1:c

vector=[I1(r1,c1,d1),I2(r1,c1,d1),I3(r1,c1,d1),I4(r1,c1,d1),I5(r1,c1,d1),I6(r1,c1,
d1)];
  fondo(r1,c1,d1)=median(vector);
 end
end
end
figure; imshow(fondo)
%Se binariza imagen con la funcion convbin2
binarizada1=convbin2(Im(:,:,1),fondo(:,:,1));
binarizada2=convbin2(Im(:,:,2),fondo(:,:,2));
binarizada3=convbin2(Im(:,:,3),fondo(:,:,3));
```

```
%se conbinan las tres imagenes binarias
binaria=or(binarizada1,binarizada2);
binaria=or(binaria,binarizada3);
figure; imshow(binaria);title('Binaria');
sinceros= bwfill(binaria,'holes');
figure; imshow(sinceros);%imagen sin huecos
%SE mejora la imagen binaria
contraria= bwmorph(sinceros,'majority',30);
figure; imshow(contraria);
contraria= bwfill(contraria,'holes');
contraria= bwmorph(contraria,'shrink');
figure; imshow(contraria);
binarizada= bwfill(contraria,'holes');
figure; imshow(binarizada);
%Se aplica cerradura a la imagen binaria
cerradura= bwmorph(binarizada,'erode',3);
figure; imshow(cerradura);title('Erode')
cerradura= bwmorph(cerradura,'dilate',3);
figure; imshow(cerradura);title('Dilate 3 planos aplicado tres veces')
%Objetos conectados
[L,carros] = BWLABEL(cerradura);
I10=im2double(cerradura);
aislados(:,:,1)=Im(:,:,1).*I10;
aislados(:,:,2)=Im(:,:,2).*I10;
aislados(:,:,3)=Im(:,:,3).*I10;
figure; imshow(aislados)%autos detectados
carros  %cantidad de objetos conectados
%Autos detectados separados
%aislar_autos(carros,aislados,L)
toc
```

APÉNDICE C

CÓDIGO FUENTE

En este apartado se presenta el código fuente del reconocimiento inteligente de flujo vehicular basado en el procesamiento de imágenes analizando los tres planos *RGB* por separado y aplicando *cerradura por partes*. Se muestra el programa principal y se utilizan las rutinas complementarias del Apéndice A.

Programa: flujovehicular3planospartes.m

Descripción: Contiene el desarrollo del proceso de detección y conteo vehicular, obteniendo el promedio de los tres planos *RGB* para su procesado. Se apoya en las rutinas complementarias presentadas en el Apéndice A: convbin2.m. y aislar_autos.m

Entradas: Lee 7 imágenes de formato *jpg*. Contenidas en la carpeta *work*.

Salidas: Variable denominada *carros* con el valor de la cantidad de autos contenidos en la imagen a contabilizar sus autos. Así como autos aislados si se desactiva el comentario para la rutina complementaria aislar_autos.m.

Código fuente:

```
clear all; close all; tic
%Se leen las imagenes a procesar para extraccion de fondo
```

```
%imagenes 1 y 2
I1=imread('foto45.jpg');        I2=imread('foto46.jpg');
I1=im2double(I1);               I2=im2double(I2);
%imagenes 3 y 4
I3=imread('foto47.jpg');        I4=imread('foto48.jpg');
I3=im2double(I3);               I4=im2double(I4);
%imagenes 5 y 6
I5=imread('foto49.jpg');        I6=imread('foto50.jpg');
I5=im2double(I5);               I6=im2double(I6);
%Imagen a contabilizar sus autos
Im=imread('foto51.jpg');
Im=im2double(Im);
%tamaño de las imagenes
[r,c,d]=size(I1);
%se calcula el valor que mas aparece
for d1=1:d
for r1=1:r
 for c1=1:c

vector=[I1(r1,c1,d1),I2(r1,c1,d1),I3(r1,c1,d1),I4(r1,c1,d1),I5(r1,c1,d1),I6(r1,c1,
d1)];
  fondo(r1,c1,d1)=median(vector);%se calcula la moda
 end
end
end
figure; imshow(fondo)
%se binariza imagen con la funcion convbin
binarizada1=convbin2(Im(:,:,1),fondo(:,:,1));
binarizada2=convbin2(Im(:,:,2),fondo(:,:,2));
binarizada3=convbin2(Im(:,:,3),fondo(:,:,3));
%se rellenan huecos
```

```
binaria=or(binarizada1,binarizada2);
binaria=or(binaria,binarizada3);
figure; imshow(binaria);title('Binaria');
sinceros= bwfill(binaria,'holes');
figure; imshow(sinceros);%imagen sin huecos

contraria= bwmorph(sinceros,'majority',30);
figure; imshow(contraria);
contraria= bwfill(contraria,'holes');
contraria= bwmorph(contraria,'shrink');
figure; imshow(contraria);
binarizada= bwfill(contraria,'holes');
figure; imshow(binarizada);

cerradura(1:r/5,1:c)= bwmorph(binarizada(1:r/5,1:c),'erode',3);
cerradura(r/5:r/4,1:c)= bwmorph(binarizada(r/5:r/4,1:c),'erode',4);
cerradura(r/4:r/3,1:c)= bwmorph(binarizada(r/4:r/3,1:c),'erode',5);
cerradura(r/3:r/2,1:c)= bwmorph(binarizada(r/3:r/2,1:c),'erode',6);
cerradura(r/2:r,1:c)= bwmorph(binarizada(r/2:r,1:c),'erode',7);
figure; imshow(cerradura);title('Erode')
cerradura(1:r/5,1:c)= bwmorph(cerradura(1:r/5,1:c),'dilate',3);
cerradura(r/5:r/4,1:c)= bwmorph(cerradura(r/5:r/4,1:c),'dilate',4);
cerradura(r/4:r/3,1:c)= bwmorph(cerradura(r/4:r/3,1:c),'dilate',5);
cerradura(r/3:r/2,1:c)= bwmorph(cerradura(r/3:r/2,1:c),'dilate',6);
cerradura(r/2:r,1:c)= bwmorph(cerradura(r/2:r,1:c),'dilate',7);
figure; imshow(cerradura);title('Dilate 3 planos')

%Objetos conectados
[L,carros] = BWLABEL(cerradura);
I10=im2double(cerradura);
aislados(:,:,1)=Im(:,:,1).*I10;
```

```
aislados(:,:,2)=Im(:,:,2).*I10;
aislados(:,:,3)=Im(:,:,3).*I10;
figure; imshow(aislados)%autos detectados
carros  %cantidad de objetos conectados
%Se ahislan los autos detectados
%aislar_autos(carros,aislados,L)
toc
```

APÉNDICE D

Mapa de Morelia Michoacán

En este apartado se presenta el mapa de Morelia Michoacán, México, donde señala la ubicación del cruce Periférico Paseo de la República Sector República, con la Avenida Huaniqueo donde se llevó a cabo la captura de imágenes analizadas.

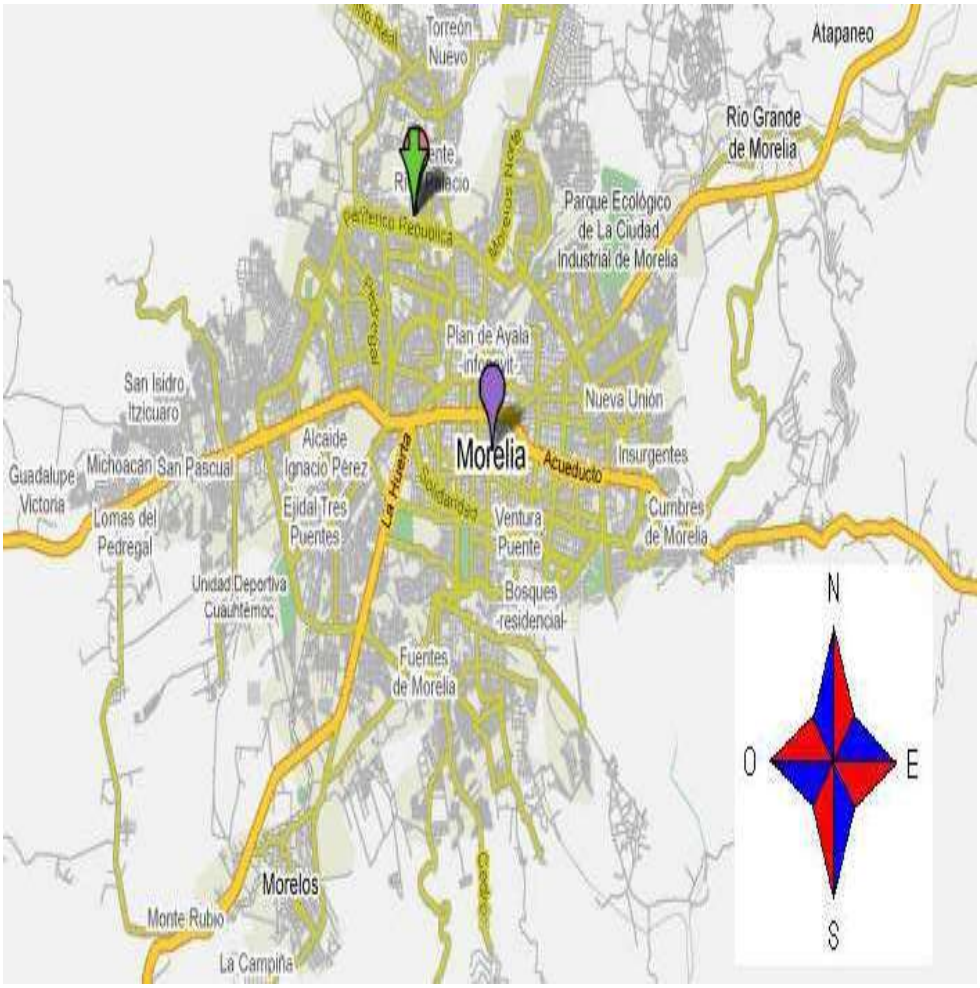


Figura Mapa de Morelia. Ubicación de Periférico Paseo de la República respecto al centro de Morelia Michoacán.

La siguiente Figura muestra un acercamiento a la ubicación del Cruce Periférico Paseo de la República con la Avenida Huaniqueo.

Figura Intersección. Confluencia del Periférico Paseo de la República con la Avenida Huaniqueo.