UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE

SISTEMAS INFORMÁTICOS

Máster Universitario en Ciencias y Tecnologías de la Computación

Proyecto Fin de Máster

Utilización de métodos de visión artificial para PC como apoyo en la automoción.

Autor:

Luis Rodrigo Barba Guamán

Director:

José Eugenio Naranjo

Doctor Ingeniero en Informática

Codirector

Liliana Enciso Quispe

Doctor en Informática

Julio 2015

Resumen

La detección de los bordes de líneas en la carretera es una parte muy importante en los sistemas inteligentes de transportación, así como la detección de objetos tal como vehículos, con la finalidad de informar o prevenir a través de una alerta al conductor o al sistema informático.

De aquí nace el interés de analizar algunos métodos de **visión artificial** (VA) que es un subcampo de la inteligencia artificial, cuyo propósito es programar un computador y que este “entienda” una escena o imagen, algunos de los métodos más comunes en la detección de líneas y vehículos (considerados objetos en nuestra investigación) son la transformada de Hough, el método de Canny, clasificador Haar Cascade, filtros de Fourier, etc. Se desarrollará una aplicación de escritorio o PC (Personal Computer) para el reconocimiento de vehículos y las líneas de bordes, el lenguaje de programación utilizado será Python y la biblioteca OpenCV que contiene más de 500 funciones en el campo de visión por computador

La validación del reconocimiento de objetos se la realizará con una prueba de campo. Este resultado apoyará a la automoción (máquina que se desplaza por acción de un motor como el vehículo) con datos que luego pueden ser procesados.

Abstract

The detection of the edges and lines on the road is a very important part in the intelligent vehicle systems. As well as, the object detection such as automobiles, which will help to inform or prevent the driver or computer system from running over the objects through an alert.

There is an interest to analyze some methods of **artificial vision** (AV), which is a subfield of artificial intelligence. Its purpose is to program a computer, which can "understand" a scene or image. Some of the most common methods in Line detection and automobiles are the Hough transform, the Canny method, Cascade Haar, Fourier filters, etc. A desktop application or PC (Personal Computer) for vehicle recognition of edges and lines will be developed. The programming language used is Python and OpenCV library, which contains over 500 functions that span to many areas in computer vision; including factory product inspection medical imaging, security, user interface, camera calibration, stereo vision and robotics.

The validation of the object recognition will be verified with a field test. This result will support the automotive, (that machine is moved by the action of an engine similar to the car) with data that can be processed.

Agradecimiento

El presente trabajo de tesis primeramente me gustaría agradecerte a ti Señor, por bendecirme y guiarme en todo momento, llegar hasta donde tu lo has decidido, porque hiciste realidad este sueño anhelado.

A mi muy amada familia, en primer lugar a mi querida esposa Jamile quién me apoyó en todo momento, a mis hijos Cristian Rodrigo y Sara Valentina quienes son una bendición y la alegría de nuestro hogar, a mis padres por sus consejos y su apoyo, en fin a toda mi familia.

A la UNIVERSIDAD POLITECNICA DE MADRID por darme la oportunidad de continuar estudiando. A mi director de tesis, Dr. Eugenio Naranjo por su esfuerzo y dedicación, quien con sus conocimientos, su experiencia, su paciencia y su motivación ha logrado en mí que pueda terminar mis estudios con éxito.

También me gustaría agradecer a mis profesores durante toda mi carrera profesional porque todos han aportado con un granito de arena a mi formación, y en especial a mi codirectora Dra. Liliana Enciso por sus consejos, su enseñanza y más que todo por su amistad.

Quiero agradecer de manera muy especial, a la UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA, por su apoyo incondicional para continuar con mi preparación en la labor docente e investigación, y más que eso formarme como un profesional con valores para servir a la sociedad a través de la ciencia.

INDICE

CAPÍTULO 1

INTRODUCCIÓN

En la actualidad los fabricantes de vehículos tratan de brindar el mejor servicio y protección al cliente, esto se puede evidenciar en el diseño de sus modelos, la potencia o seguridad que ofrecen. La seguridad no significa solamente el material con el que es fabricado o los dispositivos de airbags instalados en el vehículo, ahora se habla de los "sistemas de ayuda al conductor" (***Advanced Driver Assistence Systems - ADAS***, por sus siglas en ingles), cuyo objetivo es la reducción de los accidentes y evitar en lo posible impactos por falta de atención del conductor.

Hace algunos años los sistemas ADAS eran exclusivos solo de algunas marcas y se instalaban en sus modelos de gama media y alta, pero esto ha cambiado y se puede encontrar en la mayoría de vehículos, existiendo una mejor democratización en el uso de este tipo de sistemas.

Existen muchos tipos de sistemas de ayuda al conductor entre ellos tenemos los mas sencillos:

- Sistema de ayuda en navegación GPS
- Sistema de ayuda en aparcamiento.
- Sistema Xenon LED para días con poca visibilidad.
- Sistema de velocidad de crucero adaptativo para mantener una velocidad fija.

Pero existen sistemas más complejos tal como:

- Reparto electrónico de frenada, sucesor del sistema ABS (Antiblockiersystem).
- El sistema pre-colisión, pensada en bajar el daño posible a los pasajeros al momento de un impacto.
- Sistemas de detección de peatones, señales de tránsito, líneas en la vías, detección de fatiga del conductor, entro otros que usan métodos de visión artificial para el procesamiento de imágenes.

Este trabajo se orienta a la utilización de métodos de visión por computador que es una sub área de la Inteligencia Artificial (IA) como proyecto de fin de máster; el cual esta basado en la detección de líneas en la carretera que ha sido investigado por mas de tres décadas (Collado, Hilario, De La Escalera, & Armingol, 2005) además, la detección de vehículos y su velocidad a través del método de clasificación “Haar Cascade” (Patel & Patel, 2013) (P Viola & Jones, 2011), todo esto con la finalidad de apoyar el proceso de conducción.

1.1 Objetivos

Dentro de los objetivos que se quiere alcanzar en este trabajo de fin de máster es la utilización de métodos de visión artificial para el reconocimiento de objetos como son las líneas en la carretera, la aproximación de la velocidad de los vehículos y detección de los mismos, dado que se esta trabajando con referencia en los sistemas de ayuda al conductor se pretende que la aplicación creada pueda reconocer los objetos mencionados anteriormente con la mayor eficiencia, es decir, por poco que sean los objetos reconocidos, que lo pueda realizar con calidad y servir de ayuda al conductor.

Entre los objetivos secundarios mencionamos los siguientes:

- Analizar los métodos en el campo de visión artificial para el reconocimiento de líneas de borde y vehículos en la carretera.
- Determinar los métodos de procesamiento de imágenes y análisis de movimiento a través de librerías que manejan procesos de visión artificial.
- Desarrollar una aplicación para PC (personal computer), donde se pueda evaluar los métodos del trabajo de investigación.

1.2. Motivación

Ayudar al conductor a mejorar la capacidad de conducción a través del reconocimiento de objetos (líneas de borde de la carretera y vehículos) usando

métodos de visión artificial la librería de código abierto OpenCV para el procesamiento de imágenes y el lenguaje de programación Python para desarrollar la aplicación informática.

Mi aporte en este trabajo de investigación plantear una mejora para la detección de las líneas en la carretera a través del mejor ajuste a la recta encontrada con la técnica de mínimos cuadrados.

1.3 Estructura del documento

El presente trabajo de fin de máster está dividido en cinco capítulos.
En el capítulo 1 “Introducción” se enmarca el tema del trabajo, se expone de forma global los antecedentes y la importancia del tema, objetivos y motivación.

En el capítulo 2 “Estado del arte” se presenta de forma resumida los diferentes aspectos que se abordarán en el transcurso de la investigación tal como, revisión de literatura, revisión de trabajos relacionados que hacen referencia a la segmentación de la imagen, métodos para detectar bordes como el algoritmo de Canny, y el método de la transformada de Houhg usada para la detección de líneas.

En el capítulo 3 “Metodología“ se describe los métodos utilizados en el trabajo de investigación como el método de los mínimos cuadrados para realizar el ajuste de la líneas.

En el capítulo 4 “Análisis de resultados“ se presenta los resultados obtenidos del proceso de análisis de la información.

En el capítulo 5 “Conclusiones y trabajo futuro” se comenta las principales conclusiones a las que se ha llegado y la línea futura de trabajo que ha surgido. Por último se presentan bibliografía consultada y citada en el documento.

CAPÍTULO 2

ESTADO DEL ARTE

2.1 Antecedentes

Uno de los medios de transporte más utilizado a nivel mundial es el vehículo que ofrece comodidad, autonomía y ventajas sobre otros medios de transporte. En la actualidad el mercado global que existe y especialmente el sector de la automoción se ve afectado de forma seria y en ocasiones preocupante, es decir, la complejidad que existe en la tecnología, precios y plazos hacen que las empresas se vean, cada día, enfrentadas a nuevos retos, retos que la sociedad demanda del vehículo de las cuales se puede destacar: mayor seguridad y menor impacto al ambiente. Estas necesidades han generado que las empresas inviertan una cantidad de recurso humano y material en el proceso de investigar, desarrollar e innovar nuevos productos, así también mejorar los procesos para lograr esta meta.

El incremento de vehículos ha generado que los organismos de control y fabricantes busquen soluciones a los problemas que mencionamos anteriormente, en nuestro caso se pondrá énfasis en la seguridad. De este tema se puede mencionar que hace algunos años se ha querido dotar de inteligencia al vehículo, de aquí se ha venido mejorando gracias a los avances en el campo de la electrónica y la computación.

El vehículo generalmente suele ser el factor menos influente al momento de un accidente debido a la importante avance en materia de seguridad que incorpora, aunque se sigue investigando, en fin la mayor cantidad de accidentes de tránsito son producto del error humano, pero se esta lejos de hacer que los vehículos puedan circular de manera autónoma con la única finalidad que ayude al conductor a prevenir un accidente. Recordemos que la seguridad vial es un problema social, económico y tecnológico de gran importancia en las modernas sociedades motorizadas.

En el campo de la seguridad vial, hoy en día las empresas que fabrican vehículos se han enfocado en los **Sistemas Avanzados de Asistencia a la conducción** (***Advanced Driver Assistence Systems***) (*ADAS*) (Barcelona, 2015), este tipo de sistemas se basan en la utilización de sensores, radares o cámaras de vídeo que permiten monitorear el ambiente donde se encuentra el vehículo e informar al conductor, todo este proceso se realiza para que la conducción sea cómoda o tomar alguna decisión en caso de que la situación se presente como crítica.(Mendieta, 2013)

Por mencionar algunos de los sistemas ADAS que actualmente existen son:

- **Sistemas de navegación a bordo**. GPS es el acrónimo de “Global positioning system”, que traducimos por **Sistema de posicionamiento global**, que ofrece información del posicionamiento en la carretera e información del tráfico.
- **Control de Crucero Adaptativo.** Un sistema que, mediante radares, permite al conductor programar una velocidad deseada de crucero pero también una distancia de seguridad con el vehículo que nos precede.

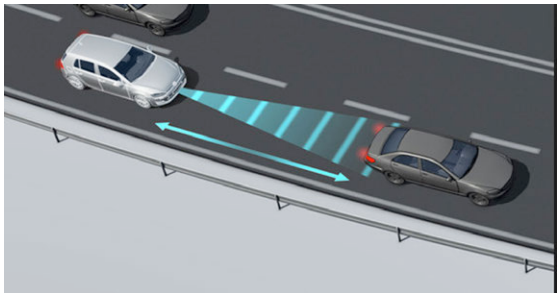


***Figura 2.1**: Sistema de Control de Crucero*

***Figura 2.2** Sistema GPS*

- **Sistema anticolisión.** Es un sistema que usa un radar frontal para detectar obstáculos móviles o estáticos en la dirección de avance del vehículo; ayudando a prevenir choques, igualmente auto carga los frenos preparándolos para entrar en acción cuando sea necesario.

- **Sistema de adaptación de velocidad inteligente.** Este sistema sabe con precisión la velocidad máxima que hay en cada tramo. Esto se puede hacer mediante una **base de datos** de límites de velocidad asociada a un mapa y la posición **GPS** del auto, o también se puede hacer mediante una **cámara de reconocimiento de señales** de tráfico que lea la señal de límite de velocidad máxima en cada momento. Incluso se pueden combinar los dos.

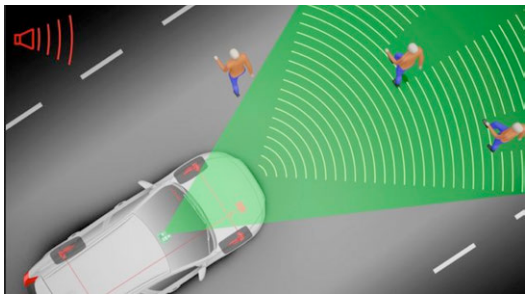


***Figura 2.3**: Sistema anticolisión*

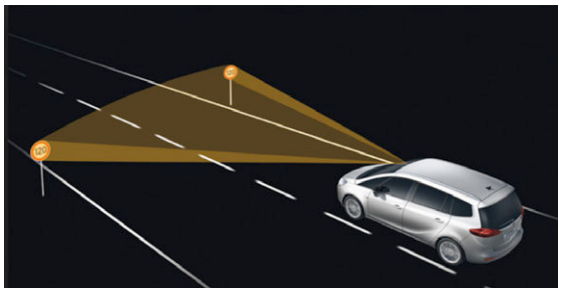


***Figura 2.4:** Sistema adaptación de velocidad inteligente*

- **Sistemas de reconocimiento de señales de tránsito**. Son sistemas que usan una cámara instalada en el lado interior del parabrisas registra las señales de tráfico que limitan la velocidad situadas, por ejemplo, en el margen de la calzada, en puentes o en zonas de obras.

- **Sistemas de ayuda de asistencia en aparcamiento.** Este tipo de sistema de ayuda al aparcamiento **con cámara de visión trasera** y **sensores de proximidad**, que indican a través de señales la cercanía de obstáculos.

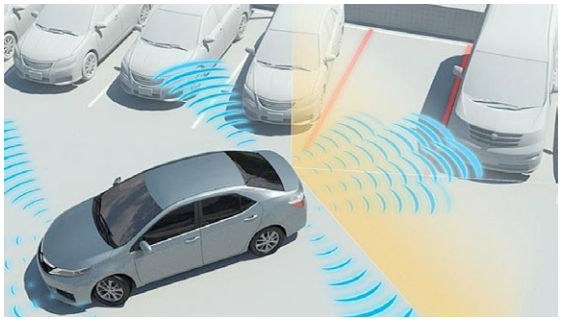


***Figura 2.5:** Sistema de asistencia en aparcamiento*

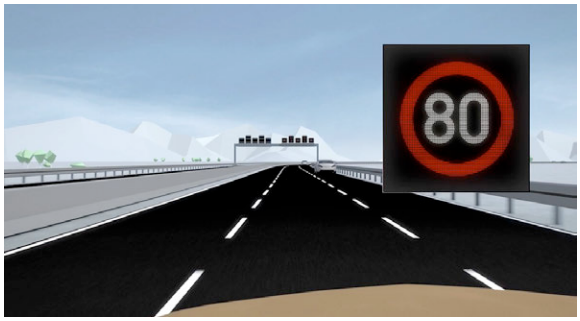


***Figura 2.6:** Sistema de señales de tránsito*

- **Control de descenso**. Permite un descenso de pendientes suave y controlado en terrenos irregulares sin que el conductor tenga que pisar el freno.

- **Detección de somnolencia en el conductor**. Dispone de una cámara que monitoriza la cara del conductor y detecta cuando éste está somnoliento.

- **Sistemas de comunicación Vehicular**. Son nodos que se sitúan en las carreteras y que se comunican con los vehículos transmitiéndoles información.

- **Sistema de control de luces adaptativo**. Actúa sobre las luces del vehículo en respuesta a las condiciones de visibilidad, dirección, suspensión, velocidad del vehículo, o presencia de otros vehículos.

- **Sistema de visión nocturna**. Dispone de una pantalla donde se monitoriza la carretera en visión nocturna. Sirve para mejorar la percepción del conductor en la oscuridad.

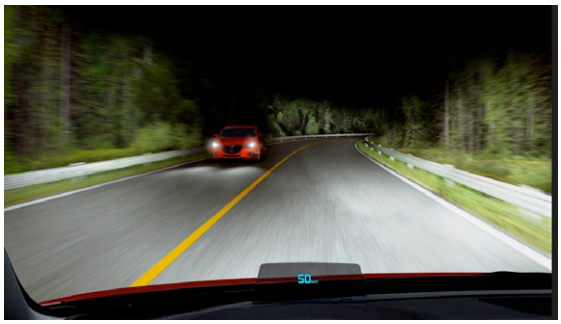


***Figura 2.7**: Sistema de control de luces adaptativa*

***Figura 2.8:** Sistema de visión nocturna.*

Como se puede apreciar son muchos sistemas, pero bastante relacionados unos con otros. Lo que se trata de incluir en este trabajo es la detección de líneas en la carretera, detección de vehículos y su velocidad, para ello utilizaremos una vídeo cámara y un computador donde se podrá apreciar la aplicación creada.

2.2 Visión por computador

La visión por computador o artificial es un campo de las ciencias de la computación y una sub área de la Inteligencia Artificial (IA), que realiza la extracción de información del mundo físico a partir de una o más imágenes, utilizando sus propiedades como formas, iluminación y distribución del color. (Szeliski, 2010)

Hoy en día la visión artificial es usada en una gran variedad de aplicaciones, de las cuales podemos incluir:

- **Reconocimiento óptico de caracteres (OCR).** Emular la capacidad del ojo humano para reconocer objetos como reconocimiento de placas de vehículos, números de las tarjetas postales, etc.
- **Inspección y control de calidad**. Inspección visual y automática a un producto como partes de vehículos, circuitos, transistores donde se busca algún defecto.
- **Imágenes médicas**. Procesamiento de imágenes, a menudo orientadas hacia el diagnóstico de dolencias o enfermedades, entre ellas; radiografías, resonancias magnéticas, tomografías.
- **Reconocimiento y clasificación**. Reconocimiento de objetos por su tamaño y establecer propiedades entre estos objetos.
- **Seguridad vehicular**. Detección de objetos como por ejemplo: peatones en la calle, vehículos, señales de tránsito, bajo ciertas condiciones donde se utilizan técnicas de visión artificial usando radares o cámaras de vídeo.
- **Vigilancia**. Monitoreo de intrusos, análisis de tráfico en autopistas, monitoreo en piscinas para posibles victimas de ahogamiento.
- **Reconocimiento dactilar y biométrico**. Procesos para la autentificación automática y aplicaciones forenses.
- **Construcción de modelos 3D**. Construcción automática de modelos en 3D para uso de fotografías aéreas como los mapas de Bing.

La visión artificial inicia con el proceso de detectar un objeto, el cual es captado por un sistema de cámaras de diversos tipos. En el interior de estas cámaras es donde se forma la imagen que, posteriormente, será pre procesada para mejorar su calidad en caso de ser necesario.

Una vez adquirida la imagen se procede a su análisis mediante la extracción de las características más relevantes o, simplemente, aquellas que necesitemos para reconocer y localizar el objeto captado.

Con la información obtenida se interpreta la imagen y, finalmente, se lleva a cabo la toma de decisiones.

Cabe destacar que la extracción de características es una de las fases críticas en el proceso, pues de ella depende la correcta o incorrecta actuación final. Este paso ha sido estudiado por gran cantidad de investigadores(W. Zhao et al., 2003), llegándose a desarrollar diversos métodos (Szeliski, 2010) (Jafri & Arabnia, 2009), pero, habitualmente, todos ellos constan de una detección de contornos seguida de una división de la imagen en varias regiones u objetos, proceso conocido como segmentación.

2.3 Trabajos relacionados

Existe diversos trabajos relacionados con respecto a la detección de objetos de interés en la carretera, tal como la detección de vehículos (Uke, 2013) y detección de líneas en la carretera (Collado et al., 2005) (Y. Zhao, Pan, Du, & Zheng, 2015), así también existen trabajos que se encargan de segmentar la imagen y localizar elementos u objetos de interés en carreteras o autopistas (Nieto, Arróspide Laborda, & Salgado, 2011) (Daigavane & Bajaj, 2010).

Es necesario indicar que las investigaciones que se han encontrado sobre el tema, realizan el proceso de segmentar la carretera o autopista, con el objetivo de buscar elementos u objetos de interés, en nuestro caso vamos identificar las líneas de la carretera (Chen, He, & Zhang, 2011) (Daigavane & Bajaj, 2010)

(Felisa & Zani, 2010); y la detección de vehículos (Nieto et al., 2011) (Prati, Mikic, Trivedi, & Cucchiara, 2003) (Shukla, 2013).

2.3.1 Detección de líneas en la carretera

La detección de líneas en la carretera tiene mas de dos décadas de investigación, algunos tienen resultados impresionantes con pruebas reales, especialmente bajo conducción automática (Collado et al., 2005), aunque hasta la presente se tiene una diversidad en ese tema (Bar Hillel, Lerner, Levi, & Raz, 2012).

Hoy en día los sistemas ADAS tienen incorporados sistemas que usan algoritmos de visión artificial para ayudar al conductor en la detección de objetos y poder informar o alertar al conductor (Barcelona, 2015), cabe recalcar que las líneas de la carretera son uno de los elementos fundamentales en este análisis. Diversas investigaciones (Talib, Rui, & Ghazali, 2013) (Collado et al., 2005), plantean una propuesta en el proceso de detectar líneas rectas a partir de la búsqueda de puntos de borde de la imagen, generalmente se usa el algoritmo de Canny (Szeliski, 2010) y la variedad de maneras de utilizar este método (Daigavane & Bajaj, 2010).

En la carretera por lo general las líneas tienen una característica especial que es ser rectas (exceptuando las curvas o sinusoidales), uno de los métodos utilizados para la detección de líneas rectas es la transformada de Hough (Talib et al., 2013) (Y. Zhao et al., 2015), que realiza la búsqueda de bordes en la imagen, implementa un filtro gaussiano lo que permite reducir el ruido (Szeliski, 2010). La transformada utiliza se basa en un sistema de votación, donde se verifica que existan rectas en la imagen como resultado de la unión de sus píxeles, el número más elevado en la votación suelen ser las líneas divisorias del carril cuando se aplica sobre una imagen (Alonso, 2013).

Además, al utilizar la transformada de Hough es posible aplicar diferentes filtros para mejorar el reconocimiento de líneas; esta alternativa permiten especificar la anchura de línea, lo cual es una buena técnica al momento de realizar la búsqueda de bordes ratificando una mejor exploración de la o las línea(s). Otra forma que se ha encontrado en trabajos científicos sobre este tema es el uso

de sistemas basados en gradientes (Chen et al., 2011) o donde se registran el mayor número de líneas en la carretera (Felisa & Zani, 2010).

2.3.2 Detección de vehículos

Otro de los temas de gran investigación es la detección de vehículos que podemos en ver en el trabajo “Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis” de Sayanan Sivaraman y Mohan Manubhai (Sivaraman & Trivedi, 2013), algunos de los métodos encontrados se basan en la utilización de filtros y descriptores (Chan, Huang, Fu, Hsiao, & Lo, 2012) (Yang, Xu, & Wang, 2012). En este estudio no se hará un análisis exhaustivo de la detección de vehículos, utilizaremos el modelo de clasificación “cascada” que fue propuesta por Paul viola y Michael Jones (P Viola & Jones, 2011) y luego mejorado por Rainer Lienhart y Jochen Maydt para que usen características diagonales (Lienhart & Maydt, 2002).

En primer lugar, un clasificador cascada se entrena con pocos cientos de ejemplos (imágenes) de un objeto particular (vehículos), estos son llamados ejemplos positivos, que están a escala en el mismo de tamaño (por ejemplo, 20x20 píxeles), y ejemplos negativos, luego este clasificador puede ser aplicado a una región de interés (del mismo tamaño como el durante el entrenamiento) en una imagen de entrada. El clasificador envía un "1" si la región es probable que muestre el objeto (vehículos), y "0" en caso contrario.

2.3.3 Modelo de región de interés (ROI)

Al momento de realizar el procesamiento de una imagen, es necesario determinar un área o región específica de trabajo, a esta área se la conoce como “región de interés” o ROI (Region of Interest). Es importante fijar este ROI con la finalidad de bajar la cantidad de píxeles de la imagen a ser procesada, en nuestro caso queremos preservar los carriles izquierdo y derecho en la carretera para luego ser usados en la detección de líneas.

A continuación se muestra una imagen normal y la misma imagen aplicando el ROI.

***Figura 2.9**: Imagen de la carretera*

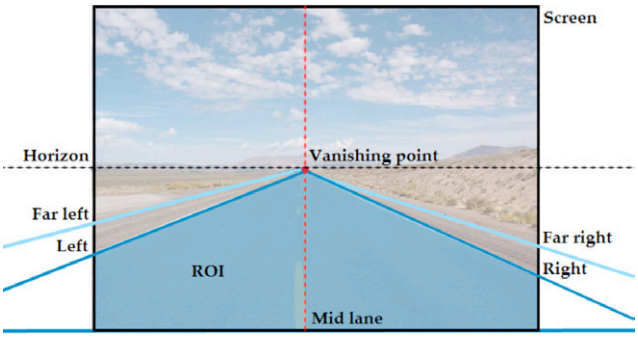


***Figura 2.10**: Imagen de la carretera aplicando ROI*

2.4 Métodos de visión artificial

Vamos a indicar la base teórica de los modelos de visión artificial utilizados en este trabajo de investigación con la finalidad de tener claro los conceptos a usar.

2.4.1 Clasificador Haar

2.4.1.1 Definición

Este apartado es muy importante ya que a través de este clasificador se puede detectar cualquier objeto en una imagen. La primera versión donde se utilizó este proceso o técnica tomo el nombre de *Haar wavelets* propuestas por Papageorgiou (Papageorgiou, Oren, & Poggio, 1998) para la detección de caras y peatones, pero su uso no fue muy significativo hasta unos años más tarde. Entonces Viola y Jones (Paul Viola & Jones, 2004) adaptaron la idea desarrollando una nueva versión a la que denominaron características Haar (Haar-Like Features). La idea que propone esta técnica consiste en identificar aquellos rasgos que van a definir a un objeto en base a la estructura de los niveles de intensidad que presentan sus píxeles en una imagen. Esta información será extraída aplicando sobre la imagen del objeto una serie de funciones implementadas bajo las denominadas características Haar. La ventaja de esta técnica es que permite detectar la estructura de los objetos aunque esta no sea uniforme.

Para entender mejor sobre Haar, se puede detallar como una ventana de píxeles, de tamaño y orientación variable, dividida en regiones rectangulares,

pudiendo ser cada una de estas regiones de dos tipos a las que se les llamará positivas o negativas (Paul Viola & Jones, 2004).

+	-
+	-

Figura 2.11: *Característica Haar.*

Esta ventana compuesta por píxeles recorre toda la imagen, sumando los píxeles positivos y negativos respectivamente, luego se realiza una diferencia de que se denomina valor de la característica. Este valor se puede representar a través de la siguiente ecuación:

$$H(x,y) = \sum_{p} I\,(x,y) - \sum_{n} I(x,y) \qquad (1)$$

Donde:

I (x, y) = Imagen a evaluar.
p y *n* = región positiva y negativa.
H (x, y) = valor de la característica Haar en el punto *x* y *y.*

2.4.1.2 Ejemplos de clasificador Haar.

Una característica de este método Haar es que está definida por parámetros tal como el tamaño, la orientación o la distribución de las regiones positivas y negativas, pudiéndose construir infinidad de tipos. A su vez, estos parámetros van a depender de los rasgos del objeto a detectar, y más en concreto de la distribución de intensidades de los píxeles que componen dicho rasgo. Así pues, el objetivo a la hora de construir una característica Haar es buscar que su estructura se asemeje a la del rasgo a detectar tal como bordes, líneas o contornos tal como en el reconocimiento de placa (Peng, Xu, Jin, Luo, & Zhao, 2011; Szeliski, 2010).

La cascada utilizada por Viola y Jones (Paul Viola & Jones, 2004) se compone por el número de regiones para un determinado uso, aquí se explica esta clasificación:

Numero de regiones	Descripción	Imágenes
Dos	Detectar bordes según la orientación: Borde Horizontal: 0° Borde Vertical: 90°	
Tres	Detectar lineas según su orientación: Linea horizontal: 0° Línea vertical: 90°	
Cuatro	Detectar lineas diagonales	

Figura 2.12: *Características Haar utilizadas por Viola y Jones* (Paul Viola & Jones, 2004).

Es fue una gran idea para determinar características usando sumas y restas de los niveles de intensidad de la imagen, Lienhar y Maydt en su trabajo (Lienhart & Maydt, 2002) añadieron Haar inclinadas para mejorar la detección de objetos en las imágenes, así con este nuevo conjunto consiguieron detectar bordes y líneas de 45°.

Numero de regiones	Descripción	Imágenes.
Dos	Detectar bordes según la orientación: Borde diagonal: 45°	
Tres	Detectar lineas según su orientación: Linea diagonal: 45° y 135°	
Dos	Detectar centros-contornos	

Figura 2.13: *Características Haar utilizadas por Lienhart y Maydt* (Lienhart & Maydt, 2002).

En el siguiente ejemplo se puede ver el reconocimiento facial utilizando Haar Cascade:

***Figura 2.14**: Detección de rostro a través de Haar Cascade.*

2.4.1.3 Definición de imagen integral

Como se ha podido comprobar, las características Haar son muy sencillas de calcular, pero aún así, suponen un coste computacional ligeramente alto. Hay que pensar que en una detección cada característica Haar utilizada tiene que evaluar toda la imagen píxel por píxel y, además, para diferentes escalas de la imagen. En definitiva esto es un una limitación para su uso en aplicaciones en tiempo real. Con esta idea de mejorar la velocidad de procesamiento de las características Haar, Viola y Jones introdujeron por primera vez el concepto de imagen integral (Paul Viola & Jones, 2004). La imagen integral es una versión transformada de la imagen original que se crea previamente para su evaluación con las características Haar.

Esta transformación consiste en crear una matriz, llamada *imagen integral*, que contiene en cada elemento la suma de todos los píxeles de la imagen original que queden por encima y a la izquierda de dicho punto, inclusive:

$$II(x,y) = \sum_{i \leq x} \sum_{j \leq y} I\,(i,j) \qquad \textbf{(2)}$$

Donde:

II (x, y) = Imagen integral
x y *y* = Elementos de la imagen.
I (x, y) = Imagen original.

Usando la imagen integral, cualquier suma rectangular de píxeles puede ser calculada rápidamente con cuatro referencias. Esto se puede comprobar con un ejemplo. En la Figura 2.15 se tiene una imagen de la cual se quiere calcular

la suma de los píxeles de la región D. El valor de la imagen integral en la referencia 1 es la suma de los píxeles de la región A. El valor en la referencia 2 será A+B, en la referencia 3 será A+C y en la referencia 4 será A+B+C+D. De esta manera la suma de los píxeles de la **región D** se puede calcular como 4+1-(2+3) sobre la imagen integral:

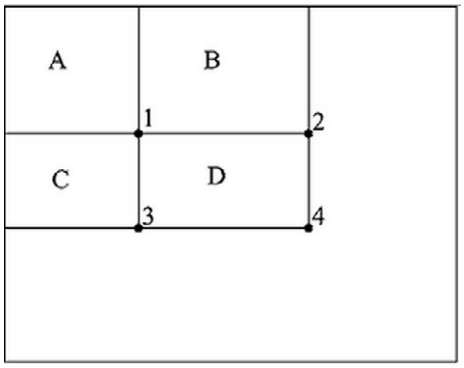


***Figura 2.15**: Cálculo de una región a partir de una imagen integral.*

Para encontrar la imagen integral se utiliza las ecuaciones 3 y 4:

$$S(x,y) = S(x,y-1) + I(x,y) \quad (3)$$

$$II(x,y) = II(x-1,y) + S(x,y) \quad (4)$$

Donde:
S(x,y) = es la suma de los píxeles en *x* y *y*
II (x, y) = Imagen integral

El objetivo de Viola & Jones era la detección de rostros, pero las cascadas Haar no sólo detectan caras, también funcionan bastante bien con objetos “rígidos”, es decir, que tienen vistas muy distintas. Esta es una característica muy importante , ya que las diferentes vistas de un vehículo son muy distintas entre sí.

En la figura 2.13 donde el número de regiones son tres, el ángulo de inclinación es de 45° y 135° (Barreto, Menezes, & Dias, 2004), la imagen integral estará acotada por la curva x−|y −y′|, por lo que la imagen integral II(x, y) está dada por:

$$rII(x,y) = \sum_{x' \le x, x' \le x \,|\, y - y'|} i\,(x', y') \quad (5)$$

Donde:

rII (x,y): Es la rotación de la imagen integral en los puntos x e y.
i (x,y): es la imagen integral

De la ecuación 5 podemos calcular en un solo paso los píxeles a través de la siguiente ecuación:

$$rII(x,y) = rII(x,y-1) + rII(x-1,y) + i(x,y) - rII(x-1,y-1) \quad (6)$$

Basado en las ecuaciones 5 y 6, la suma de todas las rectas (RecSum) viene dada por la ecuación 7:

$$RecSum(r) = rII(x+w,y+w) + rII(x-h,y+h) - rII - II(x+w-h,y+w+h) \quad (7)$$

Donde:

RecSum(r)= Es la suma de todos los rectángulos de la rotación de la imagen.
w, h = Es el ancho y altura del píxel, respectivamente.
x,y = coordenadas *x* y *y* de los píxeles en el plano cartesiano.

Esta características extendidas propuesta por Lienhart (Lienhart, Kuranov, & Pisarevsky, 2003) probó las nuevas características contra el método de Viola Jones, donde encontró una mejora del 10%.

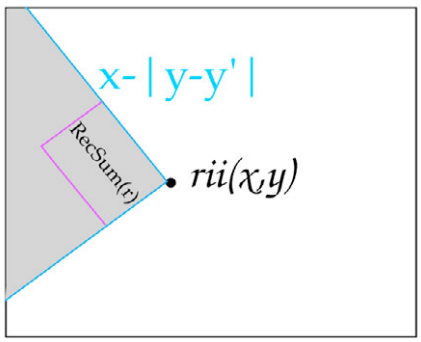


***Figura 2.16**: Cálculo de imagen integral y rectágulo rotado 135° (Lienhart & Maydt, 2002).*

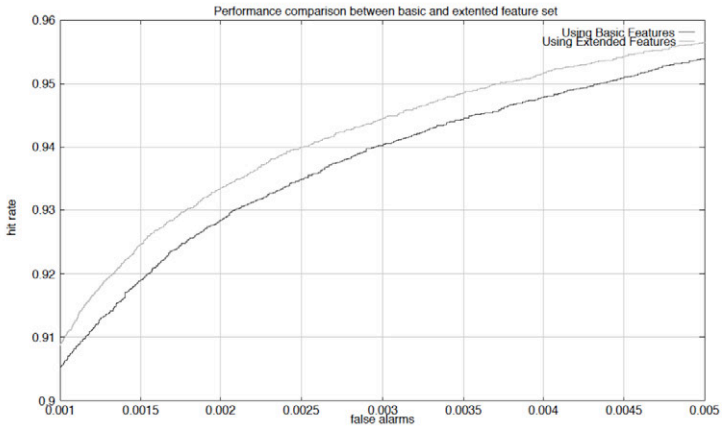


***Figura 2.17**: Curvas ROC usando característica básicas vs características extendidas.*

2.4.2 Transformada de Hough

La transformada de Hough es una poderosa técnica que puede ser utilizada para detectar características de una forma particular dentro de una imagen. Lleva el nombre de Paul Hough que patentó el método en 1962 (Hart, 2009) .El estándar transformada de Hough requiere que las características de la forma deseada puede ser descrita por una ecuación paramétrica. Por lo tanto, se utiliza más comúnmente para una detección de líneas, círculos, elipses, etc. La principal ventaja de la transformada de Hough es que es tolerante de lagunas en los bordes de las formas hasta cierto punto y es relativamente no es afectado por el ruido de una imagen y la iluminación desigual (Szeliski, 2010).

En nuestro estudio vamos a utilizar la transformada de Hough para la detección de líneas, por lo que las líneas pueden ser representadas por las siguientes ecuaciones para un plano cartesiano como el sistema de coordenadas polares:

a) Sistema de coordenadas cartesiano.

La ecuación de la recta es

$$y = mx + b \tag{8}$$

Donde:

x, y = coordenadas del punto en la línea.
m = es la pendiente de la recta.
b = es el punto de intersección en el eje *y*

Gráficamente se representa así:

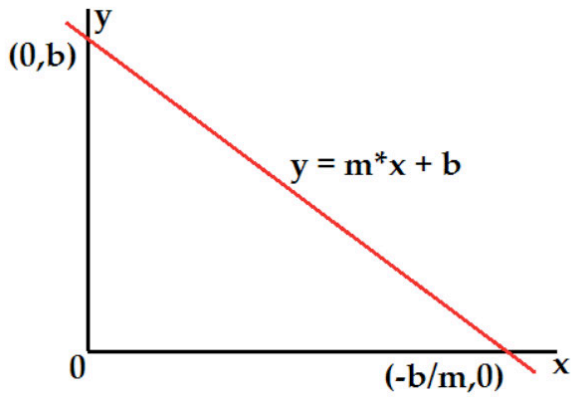


***Figura 2.18**: Línea recta en el sistema cartesiano*

La ecuación 5 tiene un inconveniente y es que cuando la pendiente tiende al infinito no podemos describir líneas verticales.

b) Sistema de coordenadas polares.

La ecuación de la recta es:

$$y = x * \left(-\frac{\cos\theta}{sen\ \theta}\right) + \left(\frac{r}{sen\ \theta}\right) \quad (9)$$

Donde:

x, y = coordenadas del punto en la línea.
θ = es el ángulo del vector directo de la recta perpendicular a la recta original.
r = es la distancia entre el origen y el punto (x,y).

Gráficamente se representa así:

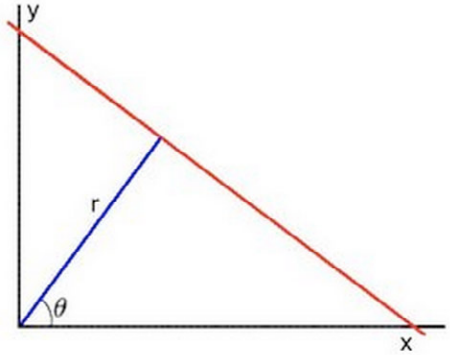


Figura 2.19**: Línea recta expresada en términos de **r

También se puede expresar en términos del parámetro r:

$$r = x\cos\theta + y\ sen\ \theta \quad (10)$$

El ángulo θ se limita en el siguiente rango 0 ≤ θ ≤ 2π. Para cualquier línea única, los valores de ρ y θ son constantes. Se puede decir que esta línea también logra ser considerada en un espacio (ρ, θ) en lugar del plano o espacio cartesiano (x, y). En el espacio (ρ, θ), un punto representa completamente una línea en el espacio de (x, y). Este espacio (ρ, θ) se cuantifica en una matriz con un número especifico de celdas por dimensión y cada celda representa una figura cuyos parámetros se pueden obtener a partir de la posición de la misma; esta cuantificación va desde el valor 1 a la distancia máxima entre el origen de la imagen y cualquier píxel de la imagen. A esta cuantificación se llama *matriz acumulador*.

Para construir la matriz acumulador es necesario separar los parámetros que describen la figura y se los realiza sobre los intervalos (r_{min}, r_{max}) y (θ_{min}, θ_{max}). Por cada punto en la imagen, se buscan todas las posibles figuras a las que puede pertenecer ese punto. Esto se logra buscando todas las posibles combinaciones de valores para parámetros que describen el objeto (los posibles valores se obtienen a partir del acumulador). Si es así, se calculan los parámetros de ese objeto, después se busca la posición en el acumulador correspondiente al objeto definido, y se incrementa el valor que hay en esa posición.

Las figuras se pueden detectar buscando las posiciones del acumulador con mayor valor (máximos locales en el espacio del acumulador). La forma más sencilla de encontrar estos **picos** es aplicando alguna forma de **umbral**, pero distintas técnicas podrían dar mejores resultados en distintas circunstancias, determinando donde se encuentran las figuras y cuantas hay.

Cada punto que es encontrado (figura 2.20) se representa en el espacio de Hough (Szeliski, 2010), y la salida es una línea sinusoide (figura 2.21), a continuación se muestra la figura en el plano cartesiano y el espacio de Hough:

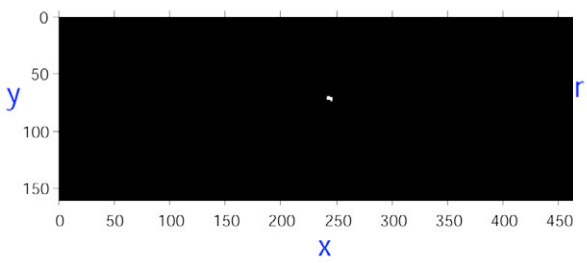


***Figura 2.20**: Representación de un punto en sistema de coordenas cartesiano*.

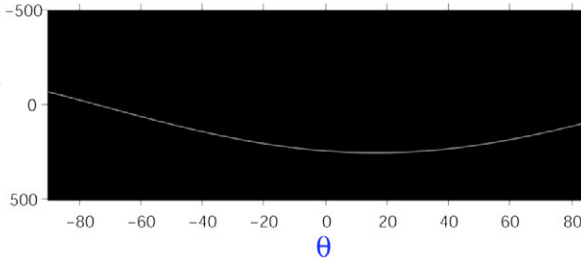


***Figura 2.21:** Representación de un punto en el espacio de Hough.*

En el caso de realizar la detección de una línea recta que se encuentran en las en sistema cartesiano (figura 2.22), se puede observar su correspondiente salida de líneas sinusoides en el espacio de Hough, todas ellas se intersectan en un punto común (figura 2.23).

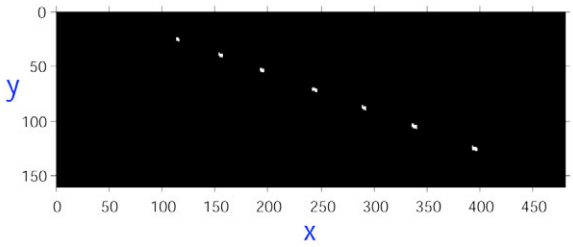


***Figura 2.22**: Representación de muchos puntos en sistema de coordenas cartesiano.*

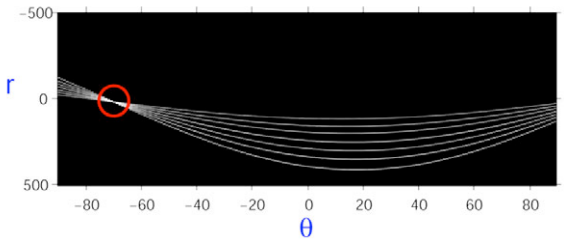


***Figura 2.23:** Representación de muchos puntos en el espacio de Hough.*

Todas estas líneas sinusoides que se han intersectado en un punto común en el espacio de Hough indican la presencia de líneas rectas, la tarea aquí es buscar los máximos locales y extraer estos puntos de intersección para regresarlos al plano o espacio cartesiano y superponer esta imagen en la imagen original, así tenemos la detección de la líneas, tal como indica la figura 2.24:

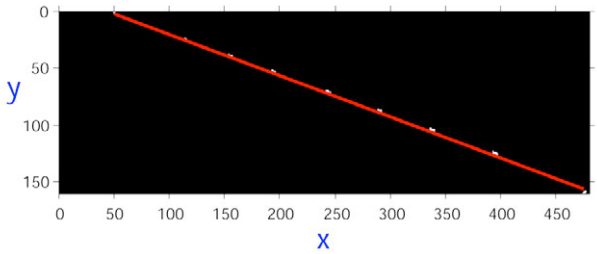


***Figura 2.24**: Linea detectada.*

2.4.3 Segmentación

La segmentación de una imagen es la búsqueda de píxeles que tienen algo en común, seguidamente se los trata de agrupar de acuerdo a esta característica, con la única finalidad que sea más fácil analizarlo (Mart, 2004).

El tema de segmentar una imagen fue investigado ya en 1970, por Brice y Fennema, ellos hablaron de una técnica para segmentar una imagen en regiones atómicas de intensidad uniforme, y luego utilizan la heurística para unirse a regiones similares entre sí (Brice & Fennema, 1970).

Existen muchos métodos de Segmentación que han surgido durante este tiempo (Szeliski, 2010) (Zuva, Olugbara, Ojo, & Ngwira, 2011), es por eso que ha sido necesario clasificarlos de acuerdo a sus propiedades.

Según el artículo “Image segmentation, available techniques, developments and open issues” hay tres tipos básicos de segmentación de imágenes (Zuva et al., 2011):

- Segmentación basada en el umbral.
- Segmentación basada en el borde.
- Segmentación basada en la región.

2.4.3.1 Segmentación basado en un umbral.- Es el más simple de los métodos. Una función o proceso se realiza en cada píxel de la imagen y el valor de salida se compara con un valor umbral establecido. Al píxel se le asigna una clase basada en qué lado del umbral cae su valor. Esto se convierte en una imagen de salida que contiene dos segmentos, uno que pasa el umbral y otro que no lo hace. Este método se esfuerza por identificar más de dos clases de regiones en una imagen.

Los umbrales actúan como separadores que permitirán decidir que conjunto de tonos de gris pertenece a una determinada región. Estas técnicas son aplicadas sobre una imagen completa, y también pueden combinarse con otras durante el pre-procesamiento o post-procesamiento de la imagen, de manera que se obtengan mejores resultados.

Una forma de separar los objetos del fondo consiste en seleccionar un umbral T que separe estas regiones. Entonces, cualquier punto (x, y) para el que se cumpla que:

$$I(x,y) > T \tag{11}$$

Es decir a una región se la etiqueta como objeto y la otra como fondo, a través del uso del histograma de grises se puede ver esta separación en una imagen *I(x,y).*

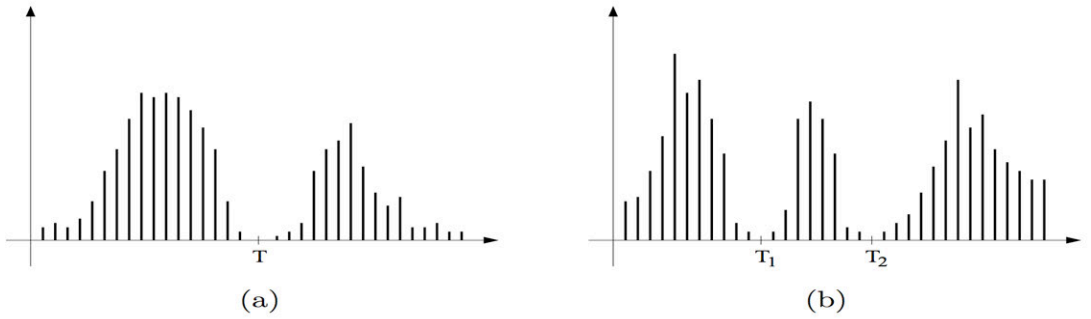


***Figura 2.25**: Histogramas de niveles de grises que se pueden segmentar con a) un umbral y b) multiples umbrales.*

El tipo de clasificación con varios umbrales es menos viable, ya que es más difícil determinar esos umbrales que separen de forma efectiva las regiones, especialmente cuando en el histograma podemos observar varios picos y valles.

El método de umbral se puede ver como una operación en la que se hace un test de cada píxel con respecto a una función T de la forma:

$$T = T(x, y, p(x, y), I(x, y)) \quad \textbf{(12)}$$

Donde:

I(x,y) es el nivel de gris del punto *(x,y).*
p(x,y) propiedad local en ese punto.

El método de umbral dará lugar a otra imagen *g* de los valores de grises de los píxeles v_{ij} comparados con respecto a un solo valor umbral *t*, definido como:

$$g(x, y) = \begin{cases} 1 & Si\ v(x, y) < t \\ 0 & Si\ v(x, y) \geq t \end{cases} \quad \textbf{(13)}$$

Debido a que contamos con un único umbral, la imagen final estará binarizada, es decir, obtendremos píxeles con dos tonos de gris, que constituyen las regiones. Si nuestro objetivo es separar los objetos del fondo, con este proceso será suficiente, pero si necesitamos realizar otro tipo de tareas es necesario trabajar con un conjunto de umbrales; esto se conoce como la técnica de multiumbrales. Una forma de escoger los umbrales adecuados es utilizando el histograma de la imagen, donde se observan picos y valles. A cada valle del

histograma se asocia un valor umbral, entonces tendremos un conjunto de umbrales (Ver figura 2.25 (b)). Todos los píxeles con un valor menor al de un umbral t_i son asignados al segmento s_i. Formalmente queda expresado así:

$$g(x,y) = \begin{cases} 0 & \text{Si } v(x,y) < t_1, \\ 1 & \text{Si } t_1 \leq v(x,y) < t_2, \\ 2 & \text{Si } t_2 \leq v(x,y) < t_3, \\ \vdots & \vdots \\ n & \text{Si } t_n \leq v(x,y). \end{cases} \quad \textbf{(14)}$$

La técnica de multiumbrales, es una de las más frecuentemente usadas pero a la vez es un proceso complejo, además no toma en cuenta aspectos espaciales de los objetos que componen la imagen (Szeliski, 2010).

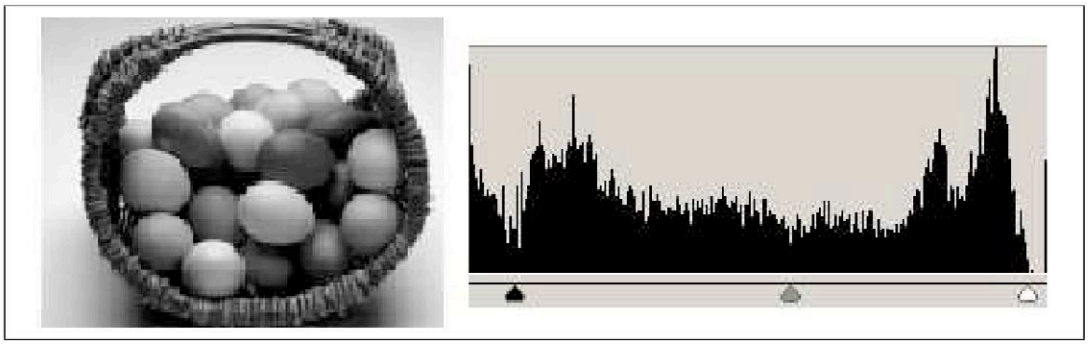


Figura 2.26: *Imagen con su histrograma se observa tres valles como posibles umbrales.*

2.4.3.2. Segmentación basada en el borde.- El borde de una imagen se puede definir como una discontinuidad en el número de píxeles en una dirección colineal (los puntos se encuentran en la misma recta), al encontrar los bordes, estos determinan los límites de cada segmento en la imagen y así poder identificar un objeto. Para encontrar estos bordes se tiene algunos métodos, incluyendo el Canny, Sobel y el algoritmo Laplaciano de Gauss. Este método puede fallar si la imagen contiene objetos con límites graduales, o en imágenes muy ruidosos (Szeliski, 2010).

- *Borde Canny*.- Este trabajo fue desarrollado por J.F. Canny (Canny, 1986). El objetivo principal de un algoritmo de detección de bordes es que sea optimo en lo siguientes puntos:

Criterio	**Descripción**
a) Detección	Detectar solo los bordes existentes.
b) Localización	Los bordes detectados versus los bordes reales deben ser tan cercanos, es decir, se debe minimizar al máximo su distancia.
c) Número de respuestas	Responder solamente al borde.

El algoritmo básico de Canny se puede resumir en 4 pasos (Canny, 1986):

1) Reducción de ruido de imagen con una convolución del kernel de Gauss.
2) El cálculo de los gradientes de la imagen.
3) Supresión de bordes que no son los máximos.
4) Borde de marca de histéresis.

En base a estos criterios, el detector de bordes de Canny suaviza primero la imagen para eliminar y ruido. Luego se busca el gradiente de la imagen para destacar regiones con derivadas espaciales altas, seguidamente en estas regiones se suprime cualquier píxel que no está en el máximo (supresión no máxima). La matriz del gradiente se ha reducido aún más por histéresis (conservación de sus propiedades en ausencia de un estímulo). La histéresis se utiliza para rastrear a lo largo de los píxeles restantes que no han sido suprimidos, con este proceso se pretende reducir la posibilidad de aparición de contornos falsos (Szeliski, 2010).

El proceso consiste en tomar la imagen obtenida del paso anterior, tomar la orientación de los puntos de borde de la imagen y tomar dos umbrales, el primero más pequeño que el segundo. Para cada punto de la imagen se debe localizar el siguiente punto de borde no explorado que sea mayor al segundo umbral. A partir de dicho punto seguir las cadenas de máximos locales conectados en ambas direcciones perpendiculares a la normal del borde siempre que sean mayores al primer umbral. Así se marcan todos los puntos

explorados y se almacena la lista de todos los puntos en el contorno conectado (Brown, 2014). Es así como en este paso se logra eliminar las uniones en forma de *Y* de los segmentos que confluyan en un punto.

Una típica salida usando el detector de Canny es la siguiente:

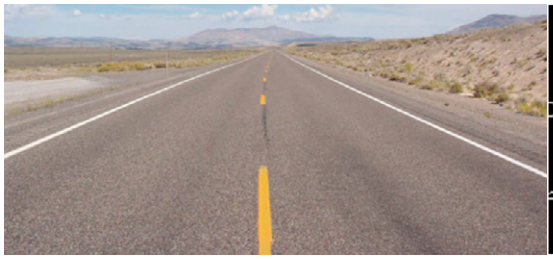


***Figura 2.27**: Entrada en el detector de Canny*

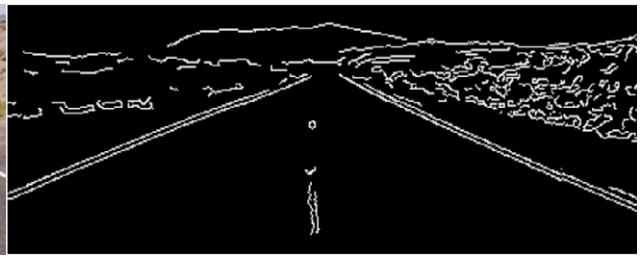


***Figura 2.28**: Salida del detector de Canny.*

La idea básica detrás de cualquier detector de bordes es el cálculo de un operador local de derivación. La figura 2.29 permite ver el concepto. En la parte derecha se puede ver una imagen de una banda clara sobre un fondo oscuro, el perfil a lo largo de una línea horizontal y la primera y segunda derivada de dicho perfil. Se puede observar que el perfil del borde se ha modelado como una discontinuidad suave, esto se debe a que los bordes en las imágenes reales están desenfocados.

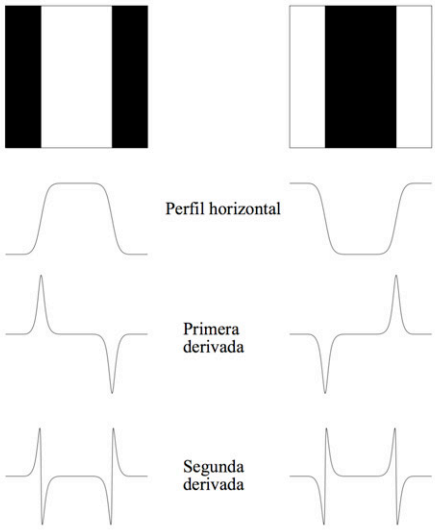


***Figura 2.27**: Detección de bordes empleando operadores de derivación.*

La primera derivada en cualquier punto de la imagen será dada por la magnitud del *gradiente*, mientras que la segunda derivada está dada por el operador Laplaciano.

- ***Gradiente*.-** El gradiente de una imagen *I(x,y)* en una posición *(x,y)* viene dado por el vector:

$$\nabla \mathrm{I} = \begin{bmatrix} I_x \\ I_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \quad \textbf{(15)}$$

La dirección del vector gradiente apunta en la dirección donde existe una máxima variación de la imagen *I* en el punto *(x,y)*, la magnitud de gradiente es muy importante en la detección de bordes y vienes dado por:

$$\nabla \mathrm{I} = \|\nabla \mathrm{I}\| = \sqrt{I_x^2 + I_y^2} \quad \textbf{(16)}$$

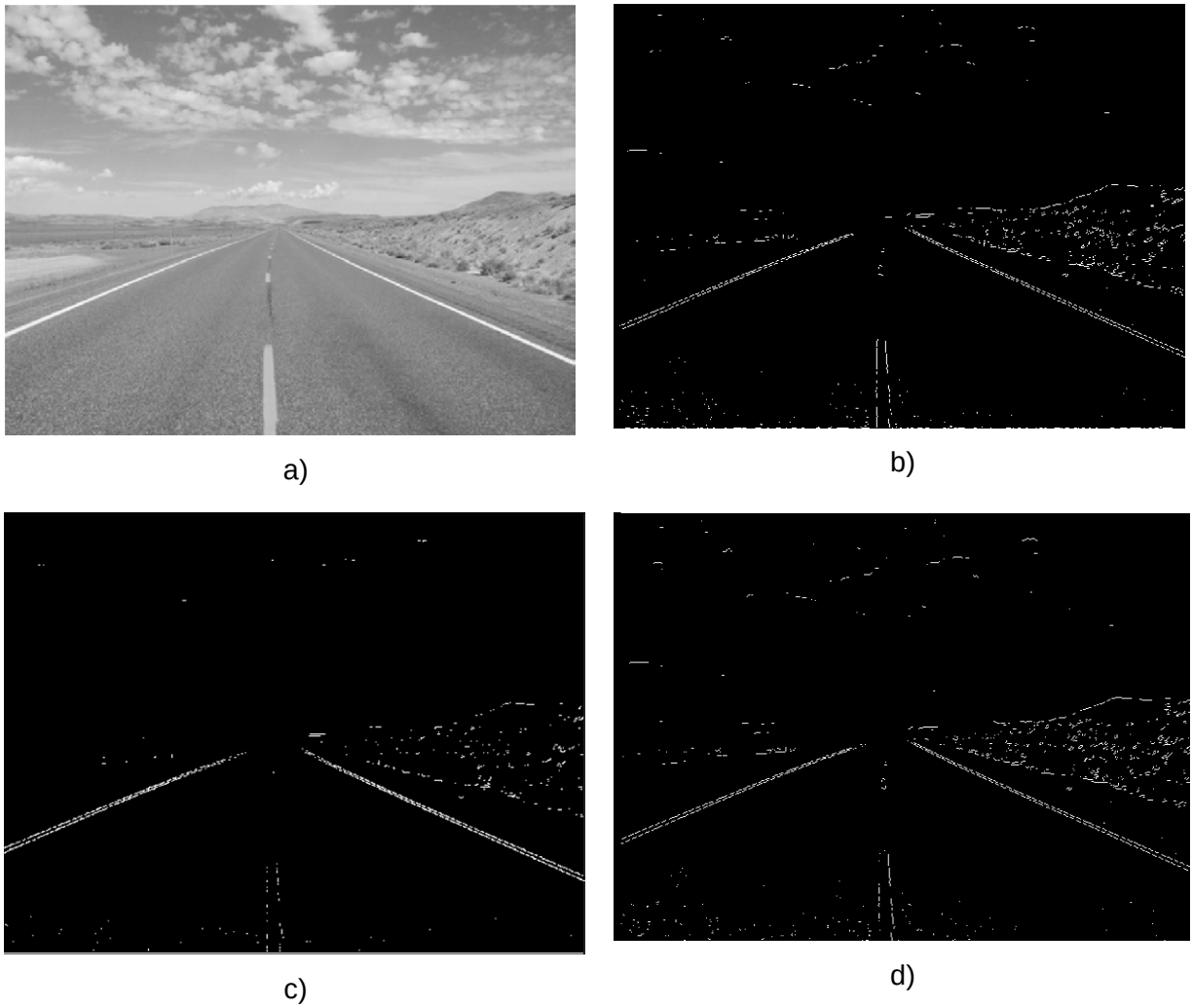
La cantidad que representa la variación de la imagen *I(x,y)* en la dirección del vector $\nabla \mathrm{I}$ se puede aproximar mediante la expresión:

$$\nabla \mathrm{I} \approx |I_x| + |I_y| \quad \textbf{(17)}$$

Un valor importante es la *dirección del gradiente*. Sea α(x, y) el ángulo del vector $\nabla \mathrm{I}$ en el punto (x, y). Entonces se tiene que:

$$\alpha(\mathrm{x}, \mathrm{y}) = \tan^{-1} \frac{I_y\ (x, y)}{I_x\ (x, y)} \quad \textbf{(18)}$$

Al momento de calcular el gradiente se obtiene las derivadas parciales para cada píxel, estas se pueden implementar de diferentes formas.



a) b)

c) d)

Figura 2.28: *a) Imagen Original en escala gris. b) Imagen gradiente de Prewitt. c) Imagen gradiente de Roberts d) Imagen gradiente de Sobel*

En la figura 2.28 podemos observar otros operadores tal como Prewitt, Roberts y Sobel. Aquí se aprecia que el operador Sobel brinda un suavizado, al momento de hacer la derivación se resalta el ruido, pero este suavizado elimina el ruido.

- ***Operador Laplaciano.-*** El operador Laplaciano de una imagen *I(x,y)* es una derivada de orden dos, esta definida de la siguiente manera:

$$\nabla^2 \mathrm{I} = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial x^2} \qquad \textbf{(19)}$$

Aunque el operador Laplaciano responde a transiciones en la intensidad de la imagen, es poco empleado en la práctica. Un uso más frecuente de este operador es a través de la propiedad de localización de los bordes usando la

propiedad de los cruces por cero (ver figura 2.29). Este concepto se basa en la relación la imagen con el operador Laplaciano de una función Gaussiana, quedando de la siguiente manera:

$$h_\sigma(x,y) = \exp(-\frac{x^2 + y^2}{2\sigma^2}) \quad \textbf{(20)}$$

Donde:

σ es la desviación estándar.

Si $r^2 = x^2 + y^2$, el operador Laplaciano de una función *h(r)* puede ser expresado como:

$$\nabla^2 \mathrm{h} = \frac{\partial^2 I}{\partial x^2} + \frac{1}{r}\frac{\partial h}{\partial r} \quad \textbf{(21)}$$

Se puede notar que el operador Laplaciano de h_σ se puede representar como sigue:

$$\nabla^2 h_\sigma = \left(\frac{r^2 - 2\sigma^2}{\sigma^4}\right) \exp(-\frac{x^2 + y^2}{2\sigma^2}) \quad \textbf{(22)}$$

Aquí se puede evidenciar que existe un valor negativo que representa cruces por cero para el valor $\pm\sigma$, el valor positivo esta en el origen y el valor negativo en $\pm\sigma$, una representación gráfica esta en la figura 2.29:

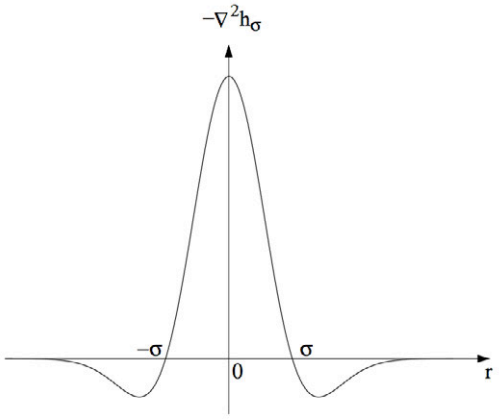
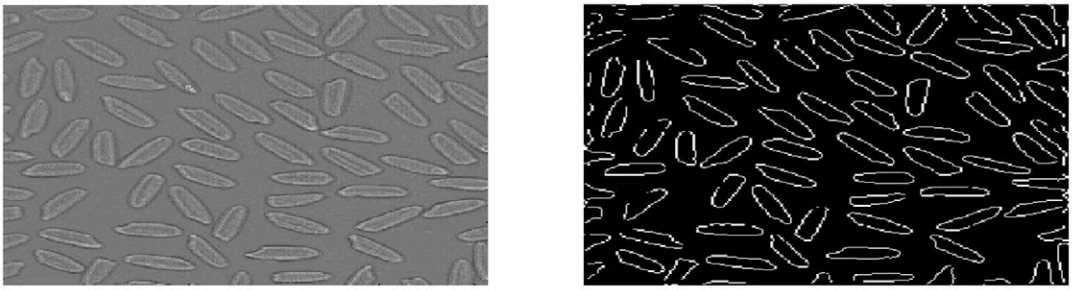


Figura 2.29: Corte por el origen del operador Laplaciano.

Se puede verificar que el valor medio del operador $-\nabla^2 h_\sigma$ es cero, al igual que el valor medio de la imagen que se obtiene al emplear este operador. Aunque

éste da lugar a una reducción del ruido, su utilidad fundamental es debido a los cruces por cero.



(a) (b)

***Figura 2.29**: a) Resultado del operador: $-\nabla^2 h_\sigma$. b) Cruces por cero*

2.4.3.3. Segmentación basada en la región.- Este tipo de segmentación intenta dividir una imagen en regiones similares a través de un criterio basado en su semejanza u homogeneidad. Algunos ejemplos de estas medidas usan la textura, el color y la intensidad. Los valores de los criterios son diferentes para cada región y las regiones se caracterizan típicamente por una distribución gaussiana de los píxeles que se encuentran ahí. Algunos ejemplos de los algoritmos de segmentación basada en la región son: el umbral del histograma, los métodos basados en gráficos, y la segmentación basada en la superficie (Pal & Pal, 1993) (Szeliski, 2010).

Este método puede experimentar problemas con los segmentos, si los criterios son demasiado imprecisos o muy estrictos, lo que puede dar que los segmentos abarquen a más de un objeto o varios segmentos dentro de un objeto(Kee, Souiai, Cremers, & Kim, 2014).

Sea *R* la región correspondiente a la imagen a segmentar, se puede ver la segmentación como un proceso que divide la región *R* en *n* subregiones R_1, R_2, . . . , R_n, tal que:

1. $U_{i=1}^{n} R_i = R$
2. R_i es una región conexa, $i = 1,2, \dots, n$
3. $R_i \cap R_j = \emptyset, \forall i, j, i \neq j$

4. $P\ (R_i) = \ VERDADERO$
5. $P\ (R_i \cup\ R_j) = \ FALSO, \forall i, j, i\ \neq j$

Donde:

$P(R_i)$, es un predicado lógico.

R_i, conjunto de puntos de la partición R.

$\emptyset$, conjunto vacío.

El primer enunciado implica que todo píxel debe pertenecer a una región. La segunda afirmación conlleva a que toda región en la imagen debe ser conexa, es decir, se asume que los píxeles de una región están todos conectados de alguna forma. El tercer enunciado asegura que las regiones deben ser totalmente disjuntas, es decir, que un píxel pertenezca a una sola región a la vez. La cuarta afirmación es la propiedad que cada píxel debe cumplir en la región segmentada (por ejemplo niveles de gris en la misma región) y finalmente el punto numero cinco indica que las regiones deben ser vecinas con respecto a la propiedad *P*.

2.5 Librería existentes para procesamiento digital de imágenes.

En la actualidad existen algunos métodos y técnicas para el procesamiento digital de imágenes, así también la existencia de muchas librerías de código abierto con métodos y funciones de visión por computador que posee ventajas e inconvenientes en el procesamiento de imágenes en tiempo real. A continuación se describen algunas de ellas.

2.5.1 OpenCV

El 13 de Junio del 2000, Intel® Corporation anunció que estaba trabajando con un grupo de reconocidos investigadores en visión por computador para realizar una nueva librería de estructuras/funciones en lenguaje C. Esta librería proporcionaría un marco de trabajo de nivel medio-alto que ayudaría al personal docente e investigador a desarrollar nuevas formas de interactuar con los ordenadores. Este anuncio tuvo lugar en la apertura del IEEE Computer

Society Conference on Computer Vision and Pattern Recognition (CVPR). Había nacido The Open Computer Vision Library (Intel., 2015) y lo hacía bajo licencia BSD (Software Libre).

OpenCV es multiplataforma, es decir, las interfaces C, C++, Python y Java que contiene, funcionan a la perfección en diversos Sistemas Operativos: Windows, GNU/Linux, Android y Mac OS X. Estas librerías contienen más de 500 funciones optimizadas, que abarcan distintas áreas en el proceso de visión por computador, como el reconocimiento de objetos, calibración de cámaras, visión estéreo y visión robótica avanzada. Gracias a esta diversidad, y a que OpenCV fue publicada bajo la licencia BSD (software libre), que permite su uso libre tanto para propósitos comerciales como de investigación, se ha utilizado en infinidad de aplicaciones, desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere el reconocimiento de objetos.

Algunas de las funcionalidades que tiene Open CV se detallan en (Szeliski, 2010):

• procesamiento de imágenes y transforma (filtrado, morfología, pirámides);
• transformaciones geométricas imagen (rotaciones, cambio de tamaño);
• histogramas.
• Segmentación.
• detección de características (Canny, Harris, Hough, MSER, SURF);
• Análisis de movimiento y seguimiento de objetos.
• calibración de la cámara y la reconstrucción 3D;
• Máquina de aprendizaje.

En la actualidad se tiene la versión 3.0

2.5.2 VXL

VXL (Vision-X-Libraries) es una colección de bibliotecas de C++ diseñados para la investigación y la aplicación de visión por computador. VXL está escrito

en C++ y está diseñado para ser portátil sobre muchas plataformas. Las bibliotecas del núcleo en VXL son:

- VNL (numéricos): contenedores numéricos y algoritmos. por ejemplo matrices, vectores, descomposiciones, optimizadores.
- VIL (imágenes): Cargar, guardar y manipular imágenes en muchos formatos de archivo comunes, incluyendo imágenes muy grandes.
- VGL (geometría): Geometría de puntos, curvas y otros objetos elementales en 1, 2 o 3 dimensiones.
- SL (streaming I/O), VBL (plantillas básicas), VUL (utilidades): miscelánea de funciones.

Además de las bibliotecas del núcleo, hay bibliotecas que cubren algoritmos numéricos, procesamiento de imágenes, sistemas de coordenadas, la geometría de la cámara, manipulación de vídeo, estructura de recuperación por el movimiento, modelado de probabilidad, de diseño GUI, de clasificación, de estimación robusta, seguimiento de función, manipulación estructura, imágenes 3D, entre otros.

Cada biblioteca núcleo es de peso ligero, y se puede utilizar sin referencia a las otras bibliotecas del núcleo. Del mismo modo, las bibliotecas no estratégicas no dependen de las otras, para que pueda compilar y vincular sólo las bibliotecas que realmente necesita (University, 2015).

2.5.3 ImageJ

ImageJ es un software en el lenguaje Java para uso en el procesamiento de imágenes, es de dominio público inspirado por el Instituto Nacional de la Salud. Se ejecuta, ya sea como un applet en línea o como una aplicación descargable, en cualquier ordenador con una máquina virtual de Java 1.4 o superior. Distribuciones descargables están disponibles para Windows, Mac OS, Mac OS X y Linux.

Puede mostrar, editar, analizar, procesar, guardar e imprimir 8 bits, imágenes de 16 bits y 32 bits. Puede leer muchos formatos de imagen, incluyendo TIFF, GIF, JPEG, BMP, DICOM, FITS y "crudo". Es compatible con "pilas", una serie de imágenes que comparten una sola ventana. Puede ejecutar multiprocesos, por lo que las operaciones que consumen mucho tiempo como la lectura del archivo de imagen se puede realizar en paralelo con otras operaciones.

Puede calcular el área y el valor de píxel estadísticas de selecciones definidas por el usuario. Puede medir distancias y ángulos. Puede crear histogramas de densidad. Es compatible con las funciones estándar de procesamiento de imágenes como la manipulación de contraste, nitidez, suavizado, detección de bordes y el filtrado de la mediana.

Calibración espacial está disponible para proporcionar mediciones dimensionales del mundo real en unidades tales como milímetros. Densidad o calibración de escala de grises también está disponible.

ImageJ fue diseñado con una arquitectura abierta que proporciona extensibilidad a través de plugins de Java. De adquisición de clientes, análisis y procesamiento de plugins se pueden desarrollar utilizando de ImageJ construido en editor y compilador de Java. Plugins escritos por el usuario permiten resolver casi cualquier problema de procesamiento de imágenes o análisis (National Institute Health, 2015).

CAPÍTULO 3

METODOLOGÍA

Hay una gran diversidad de aplicaciones para los sistemas de transporte inteligentes, con la única finalidad de evitar accidentes tránsito, algo básico en este tipo de aplicaciones son la detección de señales verticales y las líneas de la carretera (Collado et al., 2005).

3.1 Detección de líneas en la carretera.

A continuación en la figura 3.1 se describen las etapas utilizadas para la detección de la línea en la carretera.

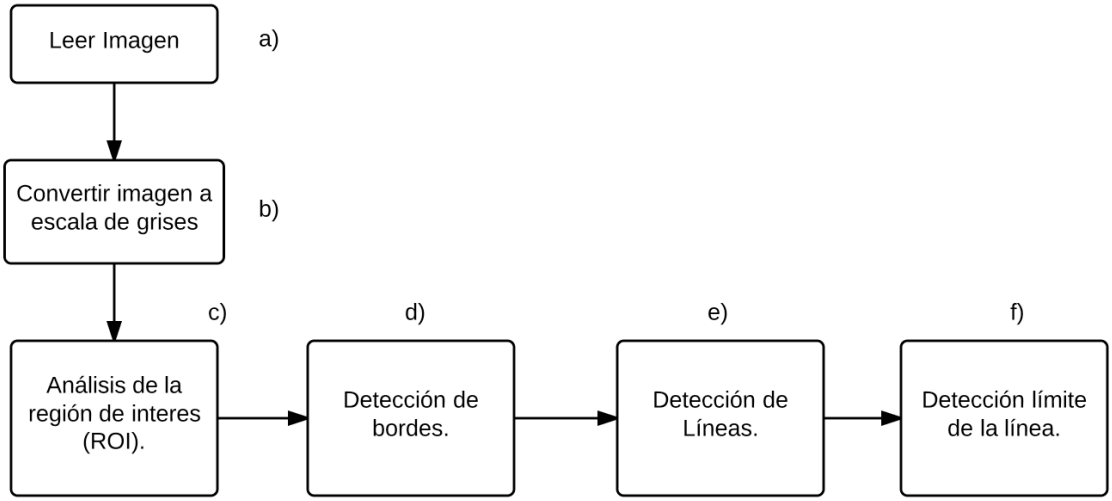


***Figura 3.1**: Proceso general*

Se presenta una breve descripción del proceso general:

a) *Leer Imagen*.- Consiste en obtener la imagen mediante una cámara filmadora ubicada dentro del vehículo.
b) *Convertir imagen a escala de grises*.- El segundo paso es convertir la imagen original a escala de grises.
c) *Análisis de la región de interés (ROI)*.- Se analiza la región de interés de la imagen con la finalidad de evitar ruido.

d) *Detección de bordes*.- A través del filtro Canny que utiliza múltiples etapas para detectar una amplia gama de bordes en imágenes.
e) *Detección de líneas*.- Utilizamos la transformada de Hough como método base para el reconocimiento de líneas en la carretera.
f) *Detección límite de la línea*.- En este proceso se hace el reconocimiento de la(s) línea(s) central de la carretera.

3.1.1 Análisis de la imagen (Leer imagen).

Recordemos que uno de nuestros objetivos es la detección de las líneas de la carretera, es decir, tratar de interpretar la mayor cantidad de información de manera precisa, pero en la etapa a) se tuvo algunos problemas:

a) Flechas impresas sobre el pavimento.

b) Sombra generada por los protectores en la carretera.

***Figura 3.2**: Dificultades en la líneas de la carretera (I).*

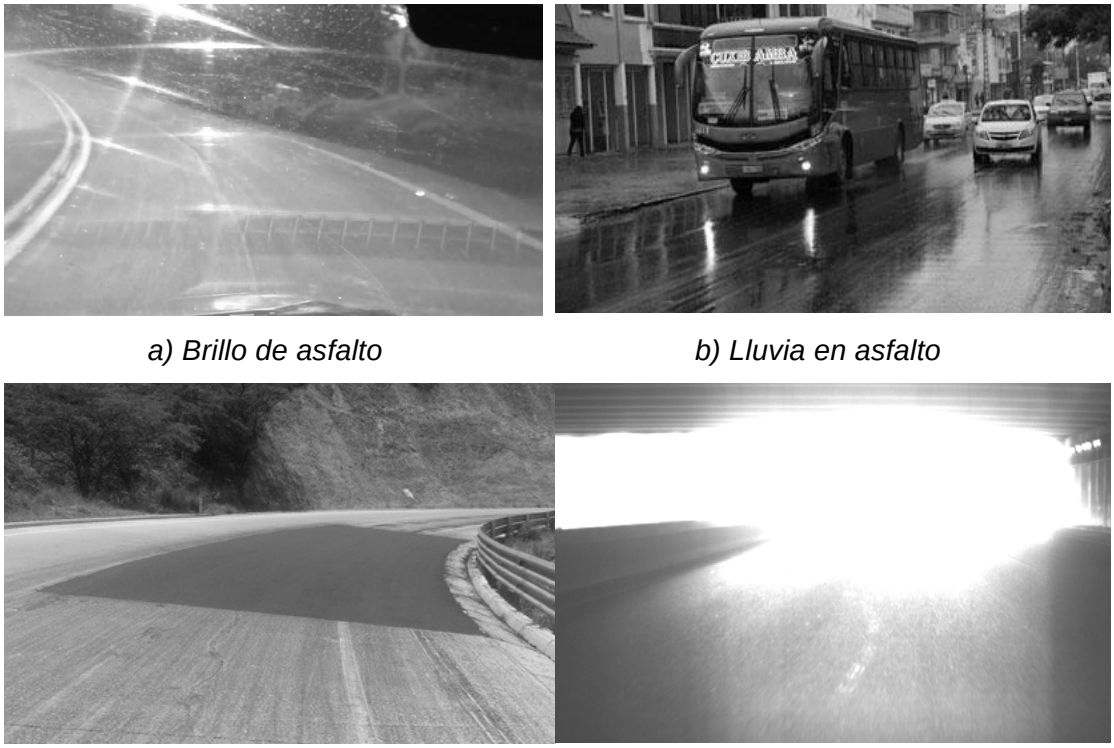
1) **Ruido estructurado.-** Es producido por algún patrón regular, por ejemplo la presencia de objetos que al momento de procesar dan lugar a formas similares a marcas viales lo que puede confundir la detección de líneas.
   - Marcas viales como flechas en el asfalto(figura 3.1a).
   - Sombras que se han generado por la protectores de carretera(figura 3.2b).

a) Líneas escondidas parcialmente por vehículos.

b) Falla de pintura asfáltica.

***Figura 3.3**: Dificultades en la líneas de la carretera (II).*

2) **Oclusión.-** Obstrucción de las líneas generadas por sombras o los vehículos (figura 3.3a).
3) **Falla en la pintura asfáltica.-** La pintura asfáltica no es visible en determinados tramos de la carretera (figura 3.3b).



a) Brillo de asfalto *b) Lluvia en asfalto*

c) Parches en la carretera. *d) Saturación de la imagen*

***Figura 3.4**: Dificultades en la líneas de la carretera (III).*

4) Condiciones climáticas y de ambiente:

4.1) **Brillo.-** Esto se genera cuando tenemos al Sol en el horizonte, especialmente cuando la luz da de frente al vehículo, se produce reflejos del asfalto y parabrisas (figura 3.4a).

4.2) **Lluvia.-** La lluvia produce que la pintura asfáltica no se reflejen desde el asfalto mojado, ya que la superficie se vuelve lisa y brillante, esto produce un ocultamiento de las líneas de la carretera (figura 3.4b).

4.3) **Cambio en la iluminación.-** El cambio de la iluminación se presenta en la salida de túneles (figura 3.4d).

3.1.2 Conversión de la imagen a escala de grises.

Los píxeles son representados de dos maneras: escala de grises y a color (Collado Hernáiz, 2009). La escala de grises es la representación de una imagen en la que cada pixel se dibuja usando un valor numérico individual que

representa su luminancia, en una escala que se extiende entre blanco y negro. Cada pixel tiene un valor entre 0 y 255, donde el 0 corresponde al color “negro” y el valor 255 corresponde al color “blanco”, es decir, los valores entre 0 y 255 están variando tonalidades de gris (256 niveles), donde los valores más cerca a 0 son más oscuros y los valores más cerca de 255 son más claros (Anali & Alvaro, n.d.).

a) Imagen a color *b) Imagen en escala de grises*

Figura 3.5: *Conversión de imagen a escala de grises.*

Los algoritmos de segmentación se basan en una de estas dos propiedades básicas de los valores del nivel de gris: *discontinuidad* o *similitud* entre los niveles de gris de píxeles vecinos (Mart, 2004).

Discontinuidad.- Se divide la imagen basándose en cambios bruscos de nivel de gris:

• Detección de puntos aislados.
• Detección de líneas.
• Detección de bordes.

Similitud.- Se divide la imagen basándose en la búsqueda de zonas que tengan valores similares, conforme a unos criterios prefijados:

• Crecimiento de región.
• Umbralización.

En nuestro caso queremos detectar bordes y líneas, que se describirá mas adelante.

3.1.3 Región de Interés (ROI)

Una vez que tenemos la imagen en escala de grises, es necesario elegir la ROI con la finalidad de reducir el ruido en procesamiento de la imagen.

Se puede apreciar la aplicación del ROI en la figura 3.6b:

a) Imagen de la carretera. b) Imagen de la carretera utilizando ROI.

***Figura 3.6**: Aplicación de ROI en la imagen.*

La ROI es el conjunto o área de píxeles en la imagen o secuencia de imágenes, a ese conjunto se lo quiere clasificar en clases, aplicando diferentes técnica de segmentación, lo cual será de mucha importancia al momento de interpretar los objetos y tratar de disminuir los errores.

Si observamos la figura 3.7 podemos ver que la carretera contiene muchos elementos, donde en primera instancia lo mas evidente es la carretera, en segundo plano tenemos los vehículos y por último elementos que no son necesarios detectar como: el cielo, faros de luz, árboles, etc, que se ubican en la mitad superior de la imagen.

Figura 3.7: *Imagen con la región de interes (ROI).*

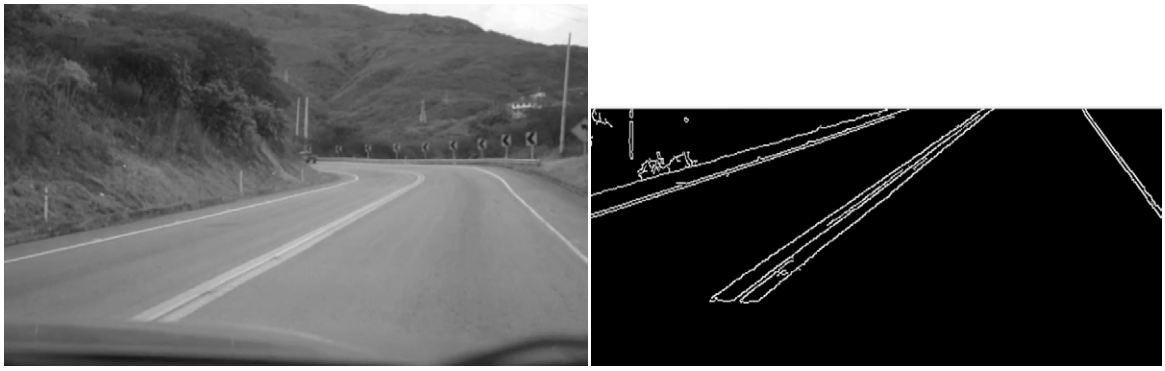
El ROI lo utilizaremos para la detección de líneas en la carretera, por lo que la mayor cantidad de información se encuentra en la mitad inferior de la imagen, si observamos el rectángulo de color rojo, vemos que contiene algunos elementos (vehículos) que no forman parte de nuestro primer análisis, que es identificar líneas en la carretera, por lo que se debe buscar una nueva área de interés tal como se visualiza en el rectángulo amarrillo.

3.1.4 Detección de bordes.

El campo del procesamiento digital de imágenes se refiere al estudio de técnicas que permitan de alguna forma mejorar una imagen, de manera que pueda ser utilizada en etapas posteriores en procesos de visión, como por ejemplo: análisis de imágenes (Mart, 2004).

Al momento de procesar la imagen las tareas más comunes son: eliminación del ruido, mejorar el contraste de la imagen, segmentación, detección de bordes, etc.

Uno de los algoritmos utilizados para detectar bordes es Canny (Szeliski, 2010), es muy usado porque implementa búsqueda de bordes de diferentes grosores e implementa un filtro gaussiano que reduce notablemente el ruido (He, Rong, Gong, & Huang, 2010).



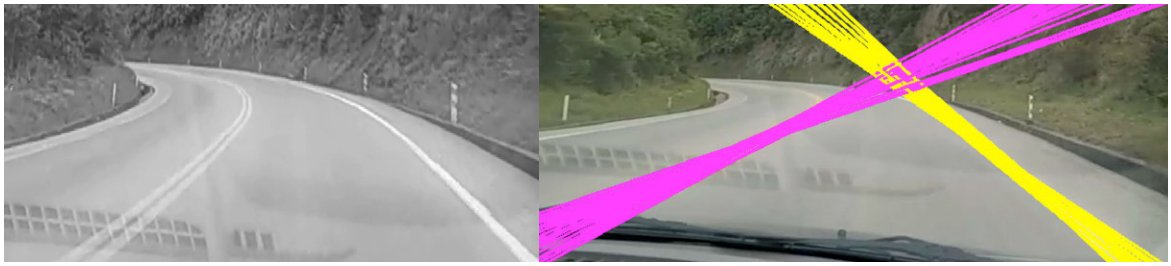
a) Imagen de la carretera sin filtro. *b) Imagen aplicado filtro Canny*

Figura 3.8: *Filtro Canny.*

3.1.5 Detección de Líneas.

Una vez que tenemos el borde de la imagen a través del filtro de Canny, en esta sección se analiza la detección de las líneas a través de la transformada de Hough (Talib et al., 2013).

En la sección 2.4.2 se explica la transformada de Hough, si recordamos que este método se basa en un sistema de votación, donde busca las rectas que pueden existir en la imagen como resultado de la unión de sus píxeles.



a) Imagen de la carretera. *b) Detección de líneas en la carretera.*

Figura 3.9*: Detección de líneas en la carretera.*

Para encontrar las líneas izquierda y derecha nos ayudamos de la ecuación 10, donde despejamos θ para encontrar el ángulo de la recta. El rango de θ esta dentro de [-180°, 180°] medido con respecto al eje de la abscisa.

La primera técnica implementada es la transformada normal de Hough, la cual detecta las líneas rectas en la imagen, utilizando la librería OpenCV usamos el método señalado HoughLines (Intel., 2015), el mismo que retorna un vector (r, θ) para cada línea. Este método utiliza coordenadas polares por si se quiere realizar la transformación al sistema de coordenadas cartesianas como en la ecuación 8.

La segunda técnica es la transformada probabilística de Houhg, método que examina segmentos de la línea (Szeliski, 2010) (Y. Zhao et al., 2015) y usa una mejora en el método de la transformada normal de Hough (Matas, Galambos, & Kittler, 2000). Este método retorna 4 coordenadas que hacen referencia a los puntos iniciales y finales de la recta en el sistema de cartesiano. También se puede ingresar la longitud y brecha de las líneas encontradas.

a) Línea detectada transformada normal de Hough.

b) Línea detectada transformada probabilística de Hough.

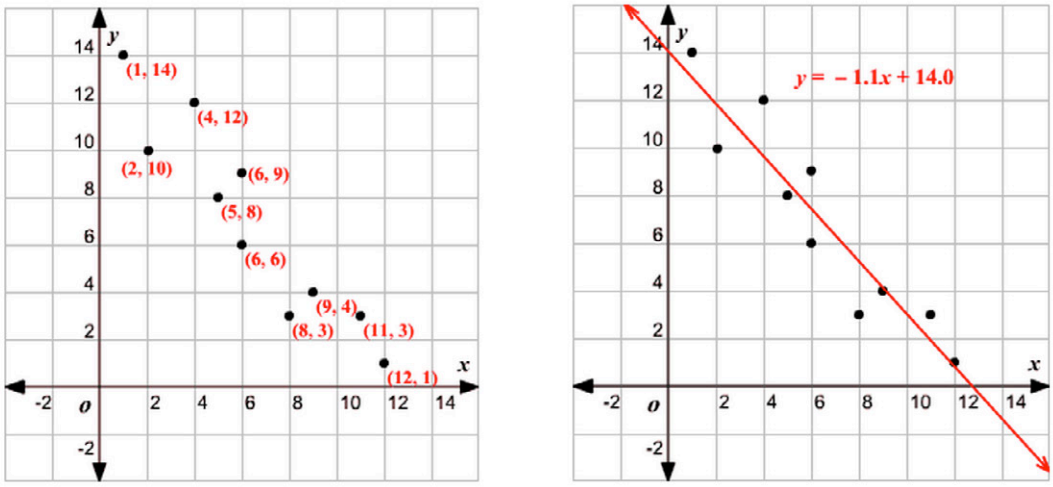
Figura 3.10: *Líneas detectadas con la transformada de Hough.*

En la figura 3.10a se visualiza muchas líneas que encuentra el algoritmo, para mejorar el trazado de la línea utilizamos la técnica de análisis numérico “mínimos cuadrados” (York, 1968) (Bar Hillel et al., 2012), que ajusta de mejor manera una línea sobre un conjunto de puntos dispersos, esta línea que está por encima o debajo de los puntos deberá ajustarse a los puntos tanto como sea posible.

Por ejemplo si tenemos las siguientes coordenadas:

x	8	2	11	6	5	4	12	9	6	1
y	3	10	3	6	8	12	1	4	9	14

Al aplicar la técnica de mínimos cuadrados se obtiene la figura 3.11:



a) Coordenadas de puntos en el sistema cartesiano.

b) Línea ajustada a través del método de mínimos cuadrados.

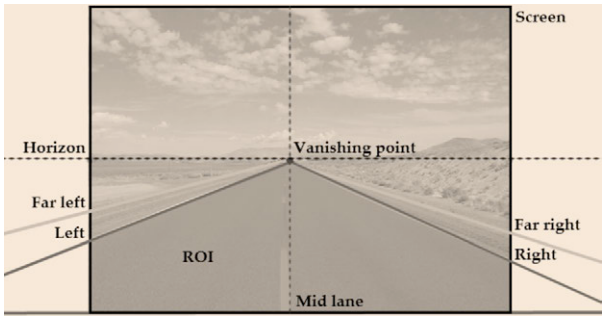
***Figura 3.11**: Técnica de mínimos cuadrados.*

Al aplicar esta técnica sobre la imagen que se esta procesando, da como resultado la figura 3.12.

***Figura 3.12:** Línea detectada y ajustada a través del método de mínimos cuadrados.*

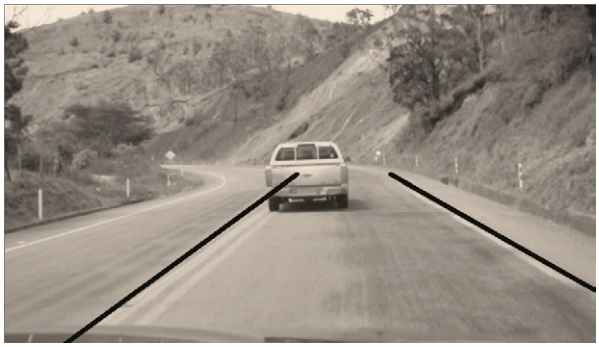
3.1.6 Detección del límite de la línea.

Delimitar la línea es importante y no es más que trazar la misma hasta el límite de la ROI o en otras ocasiones hasta el punto de fuga (vanishing point), esta información la obtenemos del tamaño de la imagen que en nuestro caso es 640 x 480 píxeles.



a) *Imagen de la carretera aplicando ROI*

b) Delimitar recta hasta el punto de fuga.



c) Delimitar recta hasta la ROI.

Figura 3.13*: Límite de la linea en la carretera.*

3.2 Detección de vehículos

Existe muchas investigaciones desarrolladas en el tema de detección y seguimiento del vehículo, esta información la podemos encontrar en las Tablas resumen II, III y IV de (Sivaraman & Trivedi, 2013), donde hace mención a los últimos 10 años en este tema. En este trabajo menciona el uso del método Histograma de Gradientes orientados y el clasificador Haar.

En la presente investigación se utilizará el clasificador Haar Cascade, en la sección 2.4.1 se menciona sobre el método, algo adicional es que también se investigó de la técnica de “back subtraction” (Zhan & Ji, 2011), pero no suministró buenos resultados ya que la cámara esta fija en un punto, mientras que nuestro estudio la cámara esta en movimiento.

En (Luis, 2012) se indica el proceso de creación y entrenamiento de un clasificador Haar Cascade a través del uso de la librería OpenCV, el cual se

toma como base para la investigación, además, usamos otras Haar Cascade encontradas en diferentes proyectos.

Las etapas que seguimos para detectar los vehículos es el siguiente:

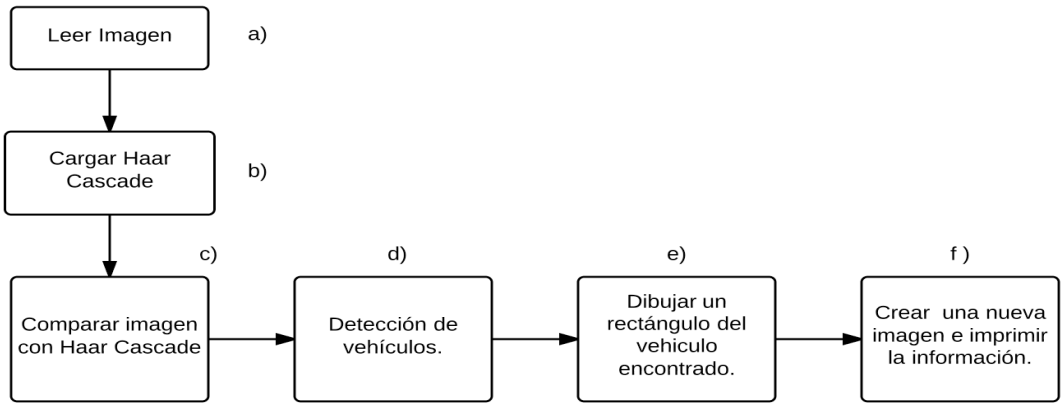


Figura 3.14: *Proceso general para detectar vehículos con el clasificador Haar Cascade.*

Se presenta una breve descripción del proceso general:

a) *Leer Imagen*.- Consiste en obtener la imagen mediante una cámara filmadora ubicada dentro del vehículo.
b) *Cargar Haar Cascade*.- Utilizar el clasificador Haar Cascade que se basa en los cambios de intensidad de la imagen para detectar un objeto.
c) *Comparar imagen con Haar Cascade.* Comparar los objetos encontrados en la imagen con los que se tiene en el clasificador.
d) *Detección de vehículos.*- Detectar el vehículo que coincide con las dimensiones descritas en el clasificador.
e) *Dibujar rectángulo:* Graficar un rectángulo o circulo del vehículo encontrado.
f) *Crear una nueva imagen:* Se debe crear una copia de la imagen original para poder representar los objetos.

El literal a) fue analizado en la sección 3.2.1 que es la lectura de la imagen.

3.2.1 Cargar Haar Cascade.

Ahora es tiempo de llamar al archivo que contiene el clasificador Haar, en nuestro caso con el nombre "*haarcascade_cars3.xml*", que fue el archivo más estable y que generó menos ruido. En las pruebas se emplearon "*haarcascade_cars1.xml*" con 15 etapas y "*haarcascade_cars3.xml*" con 20 etapas, tal como se puede ver, la diferencia radica en el número de etapas de cada uno de los archivos empleados. Algo que se debe indicar es que un *Haar Cascade* mientras más etapas tenga, es más preciso, es decir, si se entrena la misma muestra a ambos archivos, el *Haar Cascade* de 20 etapas contendrá un mayor número de clasificadores y menor número de errores que se los conoce como "falsos positivos".

Es necesario comentar que a mayor número de etapas de un Haar Cascade, el tiempo de procesamiento aumenta, por lo que hay que tener presente la capacidad de procesamiento en nuestro sistema de visión artificial, un excesivo uso de etapas puede generar una cola de imágenes a ser procesadas y causar congestión que se verá reflejada en el retardo del vídeo analizado.

Un valor promedio según (Luis, 2012), es de 20 etapas es decir, no tiene que ser demasiada estricta o demasiado débil.

Los literales c), d) y e) trabajan conjuntamente por lo que se menciona su proceso colaborativo.

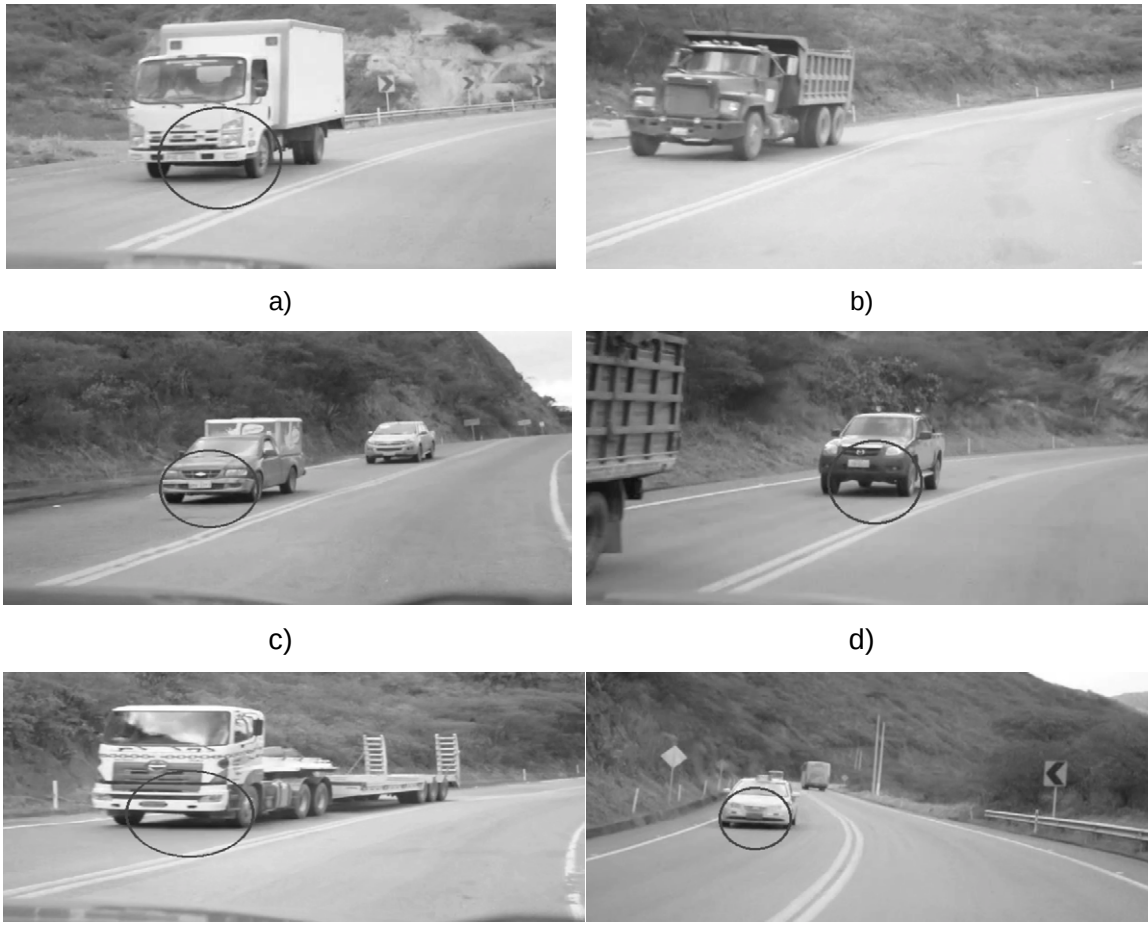
La comparación de la imagen con el clasificador es muy importe, de ello dependerá el vehículo pueda ser detectado con el menor error posible.

a)

b)

c) d)

Figura 3.15: Detección de vehículos con Haar Cascade 1



a) b)

c) d)

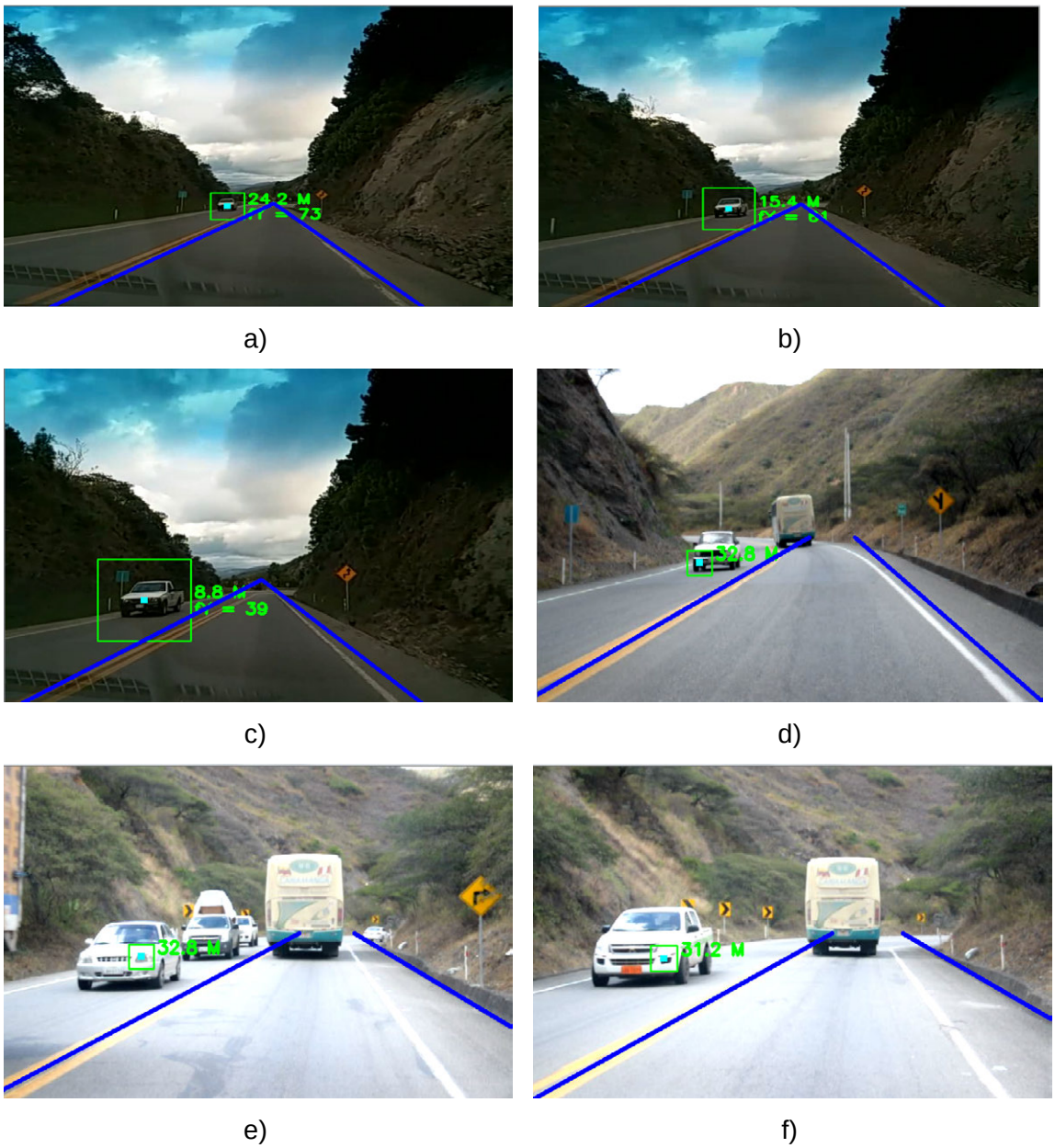
e) f)

Figura 3.16: Detección de vehículos con Haar Cascade 2

En la figura 3.15 que es el Haar Cascade entrenado con 15 etapas se puede visualizar la captura de ruido, en 3.15a y 3.15b el clasificador detecta ruido, esto se debe al número de etapas entrenadas que no son suficientes para evitar los falsos positivos o errores, por el contrario en 3.15c y 3.15d detecta los vehículos pero también ruido.

En la figura 3.16 se utilizó un clasificador con 20 etapas, en 3.16b, se observa que el camión no es detectado, esto se debe a que faltó más imágenes sobre este vehículo para que el clasificador sea entrenado, por el contrario en los demás literales podemos ver que los vehículos son detectados.

Para descubrir que se ha detectado un vehículo, se utilizó en las primeras pruebas un círculo, luego un rectángulo, además se aproximó la distancia a la que se encuentra el mismo.



a) b)

c) d)

e) f)

Figura 3.17: *Detección de líneas y vehículos en la carretera.*

En la figura 3.17 se presentan ambas detecciones, por una parte el vehículo y por otra las líneas en la carretera. Desde 3.17a hasta 3.17c se visualiza que las líneas de la carretera se unen en el punto de fuga, además observamos la distancia del vehículo y el lugar de la coordenada en el ordenadas, mientras que desde 3.17d hasta 3.17f las líneas de la carretera no llegan al punto de fuga.

Para el cálculo de la distancia entre el vehículo que se utilizó como punto de referencia para realizar las mediciones y la proximidad de los objetos (vehículos), primero se debe calibrar la cámara, es decir, ubicar en el vehículo y regular el lente. Un concepto que se maneja aquí es la distancia focal, la misma que determina el campo de visión y su unidad es en milímetros, estos valores son diferentes de acuerdo al tipo de cámara y del lente que se vaya a utilizar, en nuestro caso usamos un lente de 35mm y un valor focal de 700.

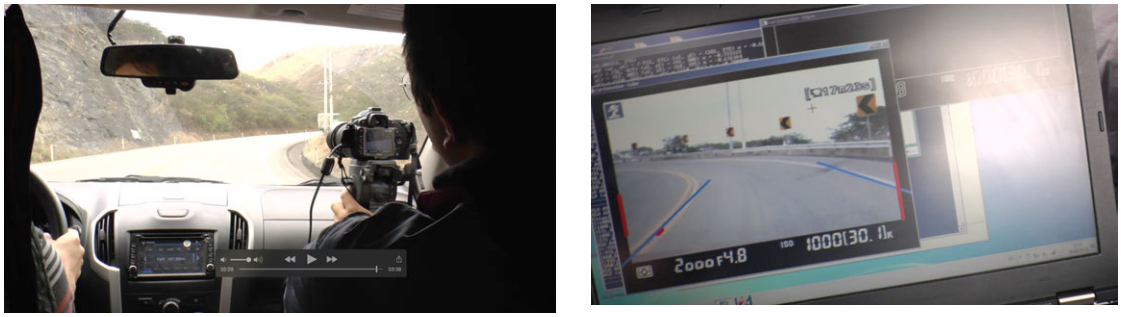
En la figura 3.18 se observa la posición de los elementos utilizados para las mediciones.

a) Cámara con trípode b) Ubicación de cámara y trípode en el vehículo.

Figura 3.18: *Imagen de cámara y trípode*

Al momento de hacer pruebas en el vehículo uno de los problemas fue la estabilidad de la cámara para poder realizar el procesamiento del vídeo, en tal

caso se pudo contralar fijando bien el trípode. En la figura 3.19 se muestra el procesamiento del vídeo en el PC.

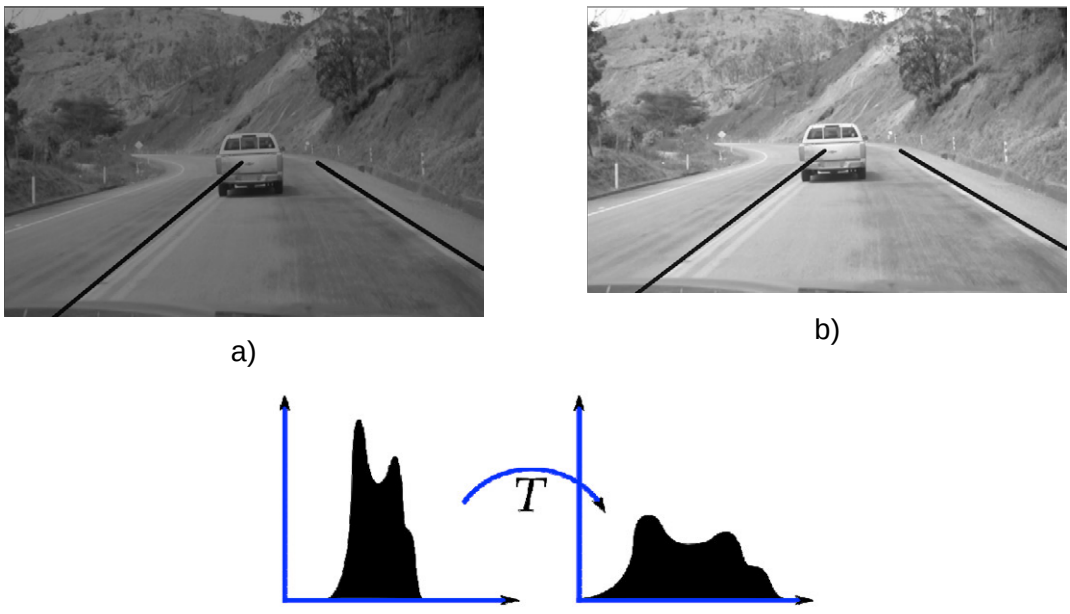


a) Ubicación de cámara en el vehículo. b) Monitoreo de la aplicación en el PC.

***Figura 3.19**: Visualización de datos en la aplicación*

Por último, para mejorar la detección de bordes que realiza el algoritmo de Canny, se aplica la ecualización del histograma a través de la librería OpenCV que permite realizar este paso a través de la función *cvEqualizeHist.* Este método lo podemos ver en la figura 3.20.

La ecualización de una imagen hace que exista una distribución uniforme de píxeles, es decir, que exista un mismo número de píxeles de la escala de grises en la imagen.



a) b)

c)

***Figura 3.20:** Ecualización de la imagen*

CAPÍTULO 4

ANÁLISIS DE RESULTADOS

En las pruebas realizadas en la aplicación informática se usó vídeos en los siguientes formatos: MP4, AVI, y MOV con diferentes tamaños en alta definición 1280 x 720, 1920 x 1080 y definición estándar 640 x 480 píxeles, se trabajó con el sistema operativo OS X Yosemite 10.10.4. Sin embargo se debe señalar que se tuvo inconvenientes con los archivos de extensión AVI, debido a la incompatibilidad con el reproductor QuickTime Player que usa la plataforma Apple, por tal motivo se evaluó los archivos con extensión AVI en la plataforma Windows 7.

Descartamos las cámaras web que se contaba en el laboratorio, la razón principal es que este tipo de cámaras no permiten controlar el brillo exterior. A continuación se muestran las características de la cámara que se utilizó:

Datos de la cámara :

Cámara	Nikon d50
Posición de la cámara	Frontal(frente a la carretera)
Formato de codificación	AVI
Razón de compresión	3 a 2
Resolución de muestreo	4:2:0
Tamaño de la imagen	640 x 480
Velocidad de reproducción	25 fps
Sensibilidad (ISO)	200 - 1600

Fuente: www.nikon.com

Para las pruebas utilizamos dos vídeos:

a) Via_lojacata.mov, 640x424 píxeles y un tiempo de 03:08 minutos. (Video_1)
b) Via_lojacata.mp4, 640x480 píxeles y un tiempo de 02:07 minutos. (Video_2)

A continuación se muestra las tablas de acierto de los vehículos detectados, utilizando los dos archivos Haar Cascades que son: "*haarcascade_cars3.xml*" y "*haarcascade_cars1.xml*" respectivamente:

Parámetros	**Video_1: Via_lojacata.mov**	**Video_2: Via_lojacata.mp4**
Vehículos reales	13	12
Vehículos detectados	11	12
Error Relativo de detección	15.38 %	0 %
Porcentaje de aciertos	84.62 %	100 %
Ruido detectado	14	140

Tabla 4.1 Resultados obtenidos con el haarcascade_cars3.xml

Parámetros	**Video_1: Via_lojacata.mov**	**Video_2: Via_lojacata.mp4**
Vehículos reales	13	12
Vehículos detectados	9	12
Error Relativo de detección	30.74 %	0 %
Porcentaje de aciertos	69.23 %	100 %
Ruido detectado	500+	500+

Tabla 4.2: Resultados obtenidos con el haarcascade_cars1.xml

En la tabla 4.1, el Video_1 presenta un 84,62% de acierto y un bajo nivel de ruido detectado (14 falsos positivos). Por el contrario el Video_2 tiene un 100% de aciertos pero con un nivel de ruido alto (140 falsos positivos). Se observa que el Video_1 tiene un menor porcentaje de acierto que Video_2, pero la cantidad de ruido que detecta Video_2 es 100% superior que el detectado por Video_1. El Haar Cascade utilizado en el Video_1 tiene mayor validez a la hora de detectar los vehículos.

En la tabla 4.2, el Video_1 presenta un 69,23% de acierto y un muy elevado ruido detectado (más de 500 falsos positivos). Por el contario el Video_2 tiene un 100% de acierto pero su nivel de detección de ruido es muy elevado (más

de 500 falsos positivos). El Haar Cascade utilizado en ambos videos, no brindó buenos resultados, por lo que se puede concluir que es necesario incrementar el número de etapas en el entrenamiento.

Detección

El tamaño inicial para empezar a buscar este objeto es de 20 x 20 píxeles, por lo que los vehículos que están muy distantes no son detectados. Utilizamos una resolución de imagen de 640 x 480 píxeles por ser el tamaño donde se obtuvo mejores resultados en la detección de vehículos. Para videos de mayor resolución el tiempo de procesamiento del equipo informático aumento, generando retardo en el procesamiento del vídeo, se procedió a estandarizar el tamaño vídeo en una sola resolución, pues trabajar con diferentes resoluciones impide un óptimo desempeño de los filtros.

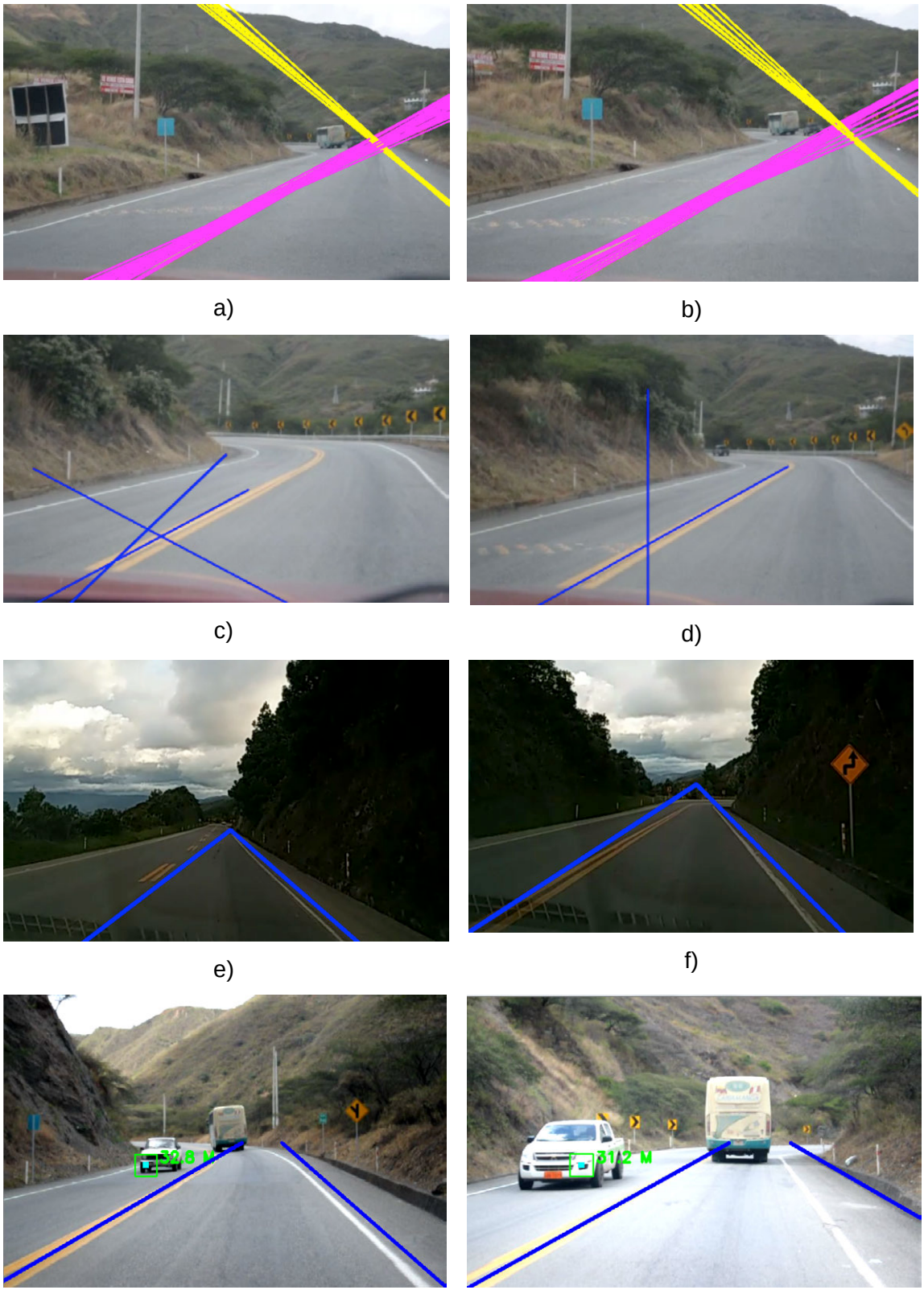
Filtrado

El filtro que se utilizó en el procesamiento del video fue el Gaussiano, esto con la finalidad de quitar el ruido inherente a la resolución , es decir, cada pixel tiene un color distinto con su vecino. Con la ecualización del histograma también se logró mejorar la distribución de grises en toda la imagen, lo que permite mejorar la eficiencia en detectar el vehículo y líneas en la carretera. El uso de máscaras también facilitó que se procese una determinada área de la imagen, mejorando el rendimiento del equipo computacional (Nieto et al., 2011).

Detección de líneas

El objetivo principal de este trabajo de investigación es la detección de las líneas, es así que se utilizó el método de la transformada de Hough mencionada en 2.4.2. Dentro de las pruebas realizadas se determinó el uso de la transformada normal del Hough, a través de la función *cv2.HouhLines* de la librería OpenCV, se obtiene el ángulo θ de la recta y sus coordenadas

cartesianas, estos valor son importantes para poder dibujar la recta en el vídeo procesado.



a) b)

c) d)

e) f)

g) h)

***Figura 4.1**: Detección de Líneas y vehículos*

En la figura 4.1 se visualiza las líneas en la carretera y algunos vehículos. En 4.1a y 4.1b se ha dibujado las líneas en el lado izquierdo y derecho de la carretera, el ángulo θ tiene que estar en el siguiente rango entre -15° y 15° (Collado et al., 2005), si observa con cuidado hay muchas líneas detectadas por el método de la transformada normal de Hough, para mejorar el resultado en la visualización, es decir, que no existan una saturación de líneas sino que se cree una sola por cada lado de la carretera (izquierdo y derecho), para esto se utiliza la técnica de mínimos cuadrados que realiza un ajuste de la recta, se puede ver la información en 3.1.5.

En 4.1c y 4.1.d se observa una mejora la detección de las líneas, pero aún se tiene problemas con los ángulos de la recta que esta fuera del rango establecido, esto puede ser generado por el ruido que tiene la imagen procesada, una solución que se desarrollo es que las líneas se encuentre en el *punto de fuga*, la información la encuentra en 3.1.6.
Uno de los problemas que se encontró durante la investigación en la detección de las líneas de la carretera, es la iluminación de la imagen, que en algunas partes era muy claro o muy oscura, para mejorar en algo el tema de la iluminación se utilizó la ecualización del histograma que es mencionado en 3.2.1.

Luego de aplicar la ecualización del histograma se observa en 4.1g y 4.1h una mejorar en imagen.

Cabe indicar que solo se ha detectado las líneas rectas en la carretera, en las líneas curvas según (Bar Hillel et al., 2012) (Collado Hernáiz, 2009) existen otros métodos más complejos para detectar, pero no forma parte de la presente investigación.

CAPÍTULO 5

CONCLUSIONES Y TRABAJOS FUTUROS

5.1 Conclusiones

En este apartado se trata de hacer un repaso de las conclusiones más importantes que se ha llegado a obtener en este trabajo de investigación.

- El método del clasificador Haar Cascade usado para la detección de vehículos es una buena alternativa, aunque tiene ciertos inconvenientes que inician desde su creación hasta la detección de ruido. El número de imágenes recomendada para su creación y entrenamiento esta entre 1000 y 10000, también un valor promedio del número de etapas es 20 donde el rendimiento y su efectividad en moderada.

- La aplicación informática creada fue desarrollada en el lenguaje de programación Python con la versión 2.7.9 y utilizando la librería OpenCV en su versión 2.4.11 que es de código abierto, se probó los métodos de visión artificial que formaron parte de los objetivos de esta investigación.

- El uso de filtros tal como: Sobel, Canny, Gaussiano, entre otros y la ecualización del histograma, son parte fundamental en el procesamiento de la imagen, esto permite que se eliminen ruidos y se mejore la eficacia para del clasificador Haar Cascade al momento de detectar un vehículo, así también mejora la búsqueda de líneas en la carretera.

- El ajuste de líneas que se encontraron en la carretera se lo realizó con el método de mínimos cuadrados, técnica sencilla pero con grandes resultados al momento de expresar la recta.

- El uso de la transformada normal de Hough, es el método tradicional para encontrar líneas, a este método se lo aplicó en la carretera, por si solo el método no es capaz de encontrar en todos los casos las líneas,

se necesita crear algoritmos para mejorar su efectividad en la detección, y ayudarse con cierto rango del ángulo y coordenadas de la recta.

- El formato de video que mejor se procesa es MOV, con una resolución 640 x 480.

5.2 Trabajos futuros

Tras cumplir con los objetivos marcados para este proyecto, se podrían plantear una serie de trabajos futuros que tomen como partida el trabajo que aquí se presenta y que tengan como finalidad mejorar la aplicación desarrollada.

- Uno de los punto débiles es quizá el clasificador Haar Cascade que a pesar de haber sido entrenado y tener las 20 etapas, aún sigue detectando ruido, uno tarea pendiente sería adicionar mas Haar Cascades que ayuden a la detección correcta del vehículo, como por ejemplo detección de placas, así se tendría dos Haar Cascade para poder validar la detección del vehículo.

- Aplicar nuevos algoritmos para el reconocimiento de líneas curvas tal como: algoritmos genéticos, filtro de Kalman, etc. Estos algoritmos son más complejos y profesionales que permiten identificar objetos a través de su color (Bar Hillel et al., 2012).

- Desarrollar una aplicación móvil que permita integrar el código desarrollado en esta investigación. Para realizar esta tarea se debe instalar los paquetes necesarios para el sistema móvil, existiendo en la actualidad soporte para el sistema Android.

REFERENCIAS.

Alonso, A. (2013). *Modelado y detección de elementos de Interés en secuencias de video de carreteras mediante técnicas de visión artificial*. Universidad Autónoma de Madrid. Retrieved from http://repositorio.eumfrayluis.com/index.php/Trabajos-Fin-de-Grado/La-estimulación-temprana-en-las-aulas-de-Educación-Infantil/

Anali, A., & Alvaro, S. (n.d.). *Segmentación de imagenes aplicando el funcional de mumford-Shah*. Universidad Nacional de Trujillo.

Bar Hillel, A., Lerner, R., Levi, D., & Raz, G. (2012). Recent progress in road and lane detection: a survey. *Machine Vision and Applications*, 1–19. http://doi.org/10.1007/s00138-011-0404-2

Barcelona, U. A. de. (2015). Advanced Driver Assistance Systems (ADAS). Retrieved May 1, 2015, from http://www.cvc.uab.es/?page_id=216

Barreto, J., Menezes, P., & Dias, J. (2004). Human-robot interaction based on Haar-like features and eigenfaces. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, *2*(April), 1888–1893. http://doi.org/10.1109/ROBOT.2004.1308099

Brice, C. R., & Fennema, C. L. (1970). Scene analysis using regions. *Artificial Intelligence*, *1*(3), 205–226.

Brown, R. C. (2014). *IRIS : Intelligent Roadway Image Segmentation using an Adaptive Region of Interest*. Virginia Polytechnic Institute.

Canny, J. (1986). A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6), 679–698.

Chan, Y.-M., Huang, S.-S., Fu, L.-C., Hsiao, P.-Y., & Lo, M.-F. (2012). Vehicle detection and tracking under various lighting conditions using a particle filter. *IET Intelligent Transport Systems*, *6*(1), 1–8.

Chen, Y., He, M., & Zhang, Y. (2011). Robust lane detection based on gradient direction. In *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on* (pp. 1547–1552).

Collado Hernáiz, J. M. (2009). *Detección y modelado de carriles de vías interurbanas mediante análisis de imágenes para un sistema de ayuda a la conducción*. Universidad Carlos III de Madrid. Retrieved from papers2://publication/uuid/162A7BDA-E825-40ED-8B94-5A811CE80E6C

Collado, J. M., Hilario, C., De La Escalera, A., & Armingol, J. M. (2005). Detection and classification of Road Lanes with a frequency analysis. *IEEE Intelligent Vehicles Symposium, Proceedings*, *2005*, 78–83. http://doi.org/10.1109/IVS.2005.1505081

Daigavane, P. M., & Bajaj, P. R. (2010). Road lane detection with improved Canny edges using ant colony optimization. In *Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference on* (pp. 76–80).

Felisa, M., & Zani, P. (2010). Robust monocular lane detection in urban environments. In *Intelligent Vehicles Symposium (IV), 2010 IEEE* (pp. 591–596).

Hart, P. E. (2009). How the Hough transform was invented [DSP History]. *Signal Processing Magazine, IEEE*, *26*(6), 18–22.

He, J., Rong, H., Gong, J., & Huang, W. (2010). A lane detection method for lane departure warning system. In *Optoelectronics and Image Processing (ICOIP), 2010 International Conference on* (Vol. 1, pp. 28–31).

Intel. (2015). OpenCV. Retrieved from http://opencv.org/

Jafri, R., & Arabnia, H. R. (2009). A Survey of Face Recognition Techniques. *Journal of Information Processing Systems*, *5*(2), 41–68. http://doi.org/10.3745/JIPS.2009.5.2.041

Kee, Y., Souiai, M., Cremers, D., & Kim, J. (2014). Sequential Convex Relaxation for Mutual Information-Based Unsupervised Figure-Ground Segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on* (pp. 4082–4089).

Lienhart, R., Kuranov, A., & Pisarevsky, V. (2003). Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *Pattern Recognition* (pp. 297–304). Springer.

Lienhart, R., & Maydt, J. (2002). An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on* (Vol. 1, pp. I–900).

Luis, A. (2012). *SISTEMA DE SEGURIDAD BASADO EN EL CÁLCULO DE LAS TRAYECTORIAS DE LOS VEHÍCULOS MEDIANTE CÁMARA EMBARCADA*. Universidad Carlos III de Madrid.

Mart, M. (2004). Técnicas Clásicas de Segmentación de Imagen.

Matas, J., Galambos, C., & Kittler, J. (2000). Robust detection of lines using the progressive probabilistic hough transform. *Computer Vision and Image Understanding*, *78*(1), 119–137.

Mendieta, A. (2013). *Detección Y Reconocimiento De Semáforos*. UNIVERSIDAD CARLOS III DE MADRID. Retrieved from http://e-archivo.uc3m.es/bitstream/handle/10016/18081/PFC_Victor_Alonso_Mendieta.pdf?sequence=1

National Institute Health. (2015). ImageJ. Retrieved from http://imagej.nih.gov/ij/index.html

Nieto, M., Arróspide Laborda, J., & Salgado, L. (2011). Road environment modeling using robust perspective analysis and recursive Bayesian segmentation. *Machine Vision and Applications*, *22*(6), 927–945. http://doi.org/10.1007/s00138-010-0287-7

Pal, N. R., & Pal, S. K. (1993). A review on image segmentation techniques. *Pattern Recognition*, *26*(9), 1277–1294.

Papageorgiou, C. P., Oren, M., & Poggio, T. (1998). A general framework for object detection. In *Computer vision, 1998. sixth international conference on* (pp. 555–562).

Patel, C. I., & Patel, R. (2013). Counting Cars in Traffic Using Cascade Haar with KLP. *International Journal of Computer and Electrical Engineering*, *5*(4), 435–437. http://doi.org/10.7763/IJCEE.2013.V5.747

Peng, Y., Xu, M., Jin, J. S., Luo, S., & Zhao, G. (2011). Cascade-based license plate localization with line segment features and haar-like features. In *Image and Graphics (ICIG), 2011 Sixth International Conference on* (pp. 1023–1028).

Prati, A., Mikic, I., Trivedi, M. M., & Cucchiara, R. (2003). Detecting moving shadows: algorithms and evaluation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *25*(7), 918–923.

Shukla, D. (2013). Speed Determination of Moving Vehicles using Lucas-Kanade Algorithm, *2*(1), 32–36.

Sivaraman, S., & Trivedi, M. M. (2013). Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *Intelligent Transportation Systems, IEEE Transactions on*, *14*(4), 1773–1795.

Szeliski, R. (2010). *Computer Vision : Algorithms and Applications*. *Computer* (Vol. 5). Springer. http://doi.org/10.1007/978-1-84882-935-0

Talib, M. L., Rui, X., & Ghazali, K. H. (2013). Comparison of Edge Detection Technique for Lane Analysis by Improved Hough Transform, 176–183.

Uke, N. J. (2013). Moving Vehicle Detection for Measuring Traffic Count Using OpenCV, *1*(4), 349–352. http://doi.org/10.12720/joace.1.4.349-352

University, B. (2015). VLX. Retrieved from http://vxl.sourceforge.net/

Viola, P., & Jones, M. (2011). Rapid Object Detection Using a Boosted Cascade of Simple Features, CVPR 2001. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer*.

Viola, P., & Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision*, *57*(2), 137–154.

Yang, S., Xu, J., & Wang, M. H. (2012). Onboard vehicle detection and tracking using boosted Gabor descriptor and sparse representation. *Electronics Letters*, *48*(16), 995–997.

York, D. (1968). Least squares fitting of a straight line with correlated errors. *Earth and Planetary Science Letters*, *5*, 320–324.

Zhan, W., & Ji, X. (2011). Algorithm research on moving vehicles detection. *Procedia Engineering*, *15*, 5483–5487. http://doi.org/10.1016/j.proeng.2011.08.1017

Zhao, W., Zhao, W., Chellappa, R., Chellappa, R., Phillips, P. J., Phillips, P. J., … Rosenfeld, a. (2003). Face recognition: A literature survey. *ACM Comput. Surv.*, *35*(4), 399–458. http://doi.org/10.1145/954339.954342

Zhao, Y., Pan, H., Du, C., & Zheng, Y. (2015). Principal direction-based Hough transform for line detection. *Optical Review*, *22*(2), 224–231. http://doi.org/10.1007/s10043-015-0033-5

Zuva, T., Olugbara, O. O., Ojo, S. O., & Ngwira, S. M. (2011). Image segmentation, available techniques, developments and open issues. *Canadian Journal on Image Processing and Computer Vision*, *2*(3), 20–29.