SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE MÉRIDA

ITM

“ESTIMACIÓN DE TRÁFICO VEHICULAR EN CAPTURAS DE VIDEO AÉREO”

TESIS

PARA OPTAR EL GRADO DE:

MAESTRO EN INGENIERÍA

PRESENTA:

ISC. RAFAEL PUERTO VALLADARES

DIRECTOR:

M.C. MARIO RENÁN MORENO SABIDO

CO-DIRECTOR:

DR. FRANCISCO JAVIER HERNÁNDEZ LÓPEZ

MÉRIDA, YUCATÁN, MÉXICO

2020

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLOGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Mérida

"2020, Año de Leona Vicario, Benemérita Madre de la Patria"

DEPENDENCIA: DIV. DE EST. DE POSG. E INV.
No. DE OFICIO: X-372/20

Mérida, Yucatán, 11/Noviembre/2020

ASUNTO: AUTORIZACIÓN DE IMPRESIÓN

C. RAFAEL PUERTO VALLADARES
PASANTE DE LA MAESTRÍA EN INGENIERÍA
PRESENTE.

De acuerdo al fallo emitido por su director **Mario Renán Moreno Sabido** y la comisión revisora integrada por el Codirector Francisco Javier Hernández López, Mauricio Orozco del Castillo, Alejandro Castillo Atoche y Víctor Sandoval Curmina, considerando que cubre los requisitos establecidos en el Reglamento de Titulación de los Institutos Tecnológicos le autorizamos la impresión de su trabajo profesional con la TESIS:

"ESTIMACIÓN DE TRÁFICO VEHICULAR EN CAPTURAS DE VIDEO AÉREO"

ATENTAMENTE
Excelencia en Educación Tecnológica

HERMILA ANDREA ULIBARRI BENÍTEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

S.E.P.
INSTITUTO TECNOLOGICO DE MERIDA
DIVISION DE ESTUDIOS DE POSGRADO E INVESTIGACION

C.p. Archivo
HAUB/GME/zac,

Agradecimientos

En primer lugar, quiero agradecer al Dr. Francisco Javier Hernández y el M.C. Mario Renán Moreno Sabido, quienes con sus conocimientos y apoyo me guiaron a través de cada una de las etapas de este proyecto para alcanzar los resultados que buscaba.

También quiero agradecer al soporte del "Laboratorio de Supercómputo del Bajío" a través del proyecto 300832 del CONACYT por el uso del servidor Quadro8000 para la implementación del algoritmo realizado. No hubiese podido arribar a estos resultados de no haber sido por su incondicional ayuda.

Por último, quiero agradecer a mi familia, quienes han puesto toda su confianza para lograr un objetivo más en mi vida. Pero, sobre todo, gracias a mi esposa y a mis hijos, por su paciencia, comprensión y solidaridad con este proyecto, por el tiempo que me han concedido, un tiempo robado a la historia familiar. Sin su apoyo este trabajo nunca se habría escrito y, por eso, este trabajo es también el suyo.

Muchas gracias a todos.

Resumen

El presente trabajo de tesis se enfoca en la aplicación de técnicas de estimación de movimiento, reconocimiento y clasificación de vehículos, estabilización de video y construcción de panoramas a partir de señales de video, para el análisis y la obtención del flujo vehicular. La señal de video se conforma por un número consecutivo de imágenes, cada imagen es analizada en conjunto con la imagen inmediata anterior para obtener la información requerida en la estimación del flujo vehicular. Dado el crecimiento del uso de vehículos en los últimos años y su impacto ambiental y social, sumado al costo de implementación de sistemas de monitoreo y análisis del tráfico vehicular, en este trabajo de tesis, se plantea el uso de vehículos aéreos no tripulados (*UAV, Unamed Aerial Vehicle*) ofreciendo la información necesaria para el análisis del tráfico vehicular.

Considerando que el uso de un *UAV* queda fuera del alcance de los recursos para su implementación en este trabajo, se desarrolla un enfoque al análisis de la señal enviada por el UAV de tal manera que únicamente se utiliza el video capturado.

El análisis del tráfico vehicular consta de la detección de los vehículos presentes en la señal, la estimación del desplazamiento del vehículo y su velocidad a lo largo del video. Debido a que el *UAV* se encuentra expuesto al ambiente, la señal de video puede contener desplazamientos o vibraciones de la cámara que afectan directamente el análisis, por ello, se implementa un proceso de estabilización de video con la finalidad de reducir los errores al momento de analizar las imágenes. Pensando que el trabajo de tesis pueda ser utilizado en tiempo real para actividades de monitoreo, la

visualización del análisis es importante, por lo que se implementa una metodología de construcción de panoramas para mantener la escena analizada completa.

Abstract

This thesis focuses on the application of techniques such as motion detection, recognition and classification of vehicles, video stabilization and panorama construction from video signals, for the analysis and obtaining of the vehicular flow. The video signal is composed of a consecutive number of images, and each image is analyzed in conjunction with the immediate previous image to obtain the information required in estimating the traffic flow. Given the growth in vehicle use in recent years and its environmental and social impact, in addition to the cost of implementing vehicle traffic monitoring and analysis systems, this thesis proposes the use of unmanned aerial vehicles (UAVs) offering the necessary information for the analysis of vehicular traffic.

Considering that the use of a UAV is outside the scope of the resources for its implementation in this work, an approach is developed to the analysis of the signal sent by the UAV, in such a way that only the captured video is used.

Vehicular traffic analysis comprises detection of vehicles present in the signal, the estimation of vehicle displacement and its speed along the video. Because the UAV is exposed to the environment, the video signal may contain displacements or vibration of the camera directly affecting the analysis, therefore, a video stabilization process is implemented in order to reduce errors at the time to analyze the images. Considering that the thesis can be used for real-time monitoring activities, the visualization of the analysis is important, so a panorama construction methodology is implemented to keep the analyzed scene complete.

Índice de Contenido

Índice de Figuras

Índice de Tablas

Capítulo 1. Introducción

1.1 Antecedentes

Dado el crecimiento en los niveles de urbanización en los últimos años, es necesario que en la ciudad se pueda ofrecer una buena calidad de vida; una manera de combatir esto es proporcionar las condiciones adecuadas de movilidad vehicular.

Según el Instituto Nacional de Estadística y Geografía, en el último reporte presentado, menciona que en 2018 existían en México 47,790,950 vehículos de motor registrados en circulación [1]; debido a esto, en los últimos años el tráfico vehicular ha incrementado de manera alarmante como se puede apreciar en la Figura 1, lo que implica la necesidad del análisis del mismo para la toma de decisiones para poder afrontar el problema.

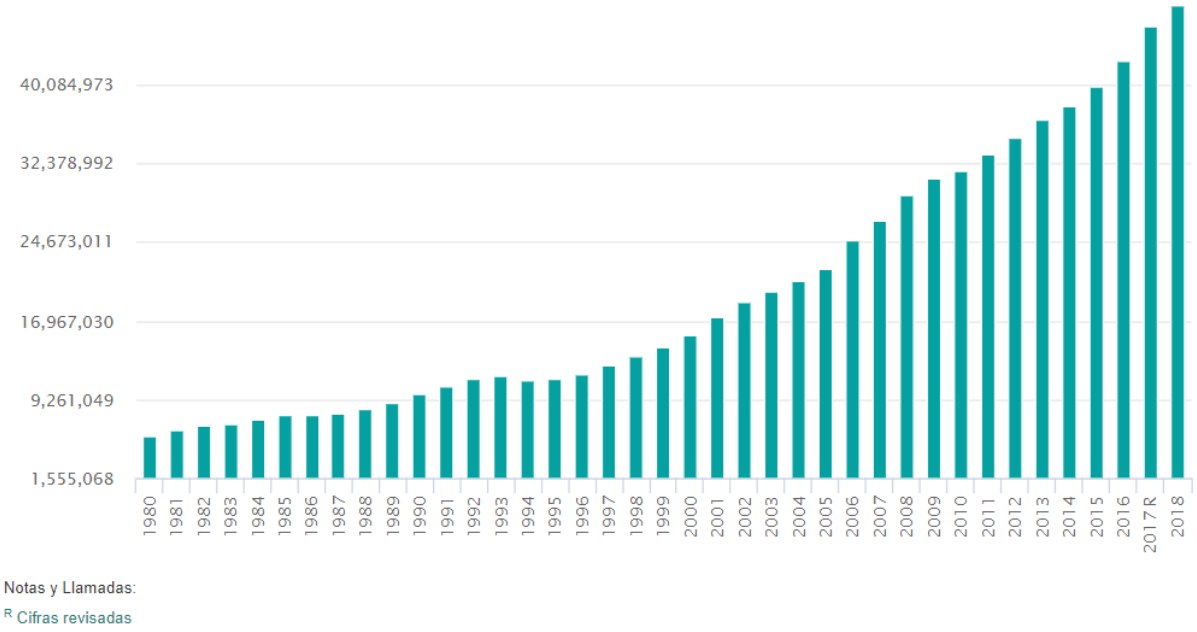


Figura 1. Crecimiento vehicular (INEGI).

Los ingenieros de tránsito, los controladores de tráfico, centros de control vial, dependencias, entre otros, necesitan de esta información para poder realizar trabajos

de ingeniería, diseño de nueva infraestructura vial, entre otros. Esto, con la finalidad de resolver la problemática presentada en las ciudades, así como también el poder ofrecer soluciones alternativas diferentes. Actualmente, el monitoreo de tráfico de una carretera es implementado por medio de sensores de radar, infrarrojos y cámaras de tráfico, sin embargo, estos muchas veces cuentan con puntos ciegos [2], [3], [4].

Una manera diferente de obtener información de tráfico vehicular es el análisis de fotografías aéreas, siendo éstas obtenidas por medio de satélites y aviones tripulados obteniendo imágenes como las presentadas en la Figura 2.

a) b)

Figura 2. Captura aérea de autopista y ciudad.

1.2 Planteamiento del problema

La obtención de la información por medio de satélites y aviones ofrece un campo de visión más amplio; por otra parte, el costo e implementación se vuelve problemático, ya que restringen su empleo práctico. Actualmente, los métodos tradicionales para el análisis de tráfico vehicular es la instalación de una gran cantidad de cámaras de vigilancia, personal dedicado al monitoreo de dichas cámaras, personal de tránsito en sitio, y en algunos casos personal extra; este grupo de trabajo analiza dicha información para poder tomar decisiones acerca de las acciones a tomar para establecer los criterios que se aplicarán para mantener el flujo vehicular de una

determinada zona. Por lo tanto, esto implica el invertir grandes cantidades de dinero, y las limitaciones en particular de las cámaras fijas en varios aspectos en particular. La obtención de la información se vuelve compleja, ya que se requiere de la integración de la información de las diferentes fuentes para posteriormente analizarla. Dado lo anterior, surge la siguiente pregunta:

¿De qué manera se puede automatizar la obtención de la información para el análisis del flujo vehicular?

1.3 Objetivos

1.3.1 Objetivo general

El objetivo de este trabajo es desarrollar una metodología capaz de realizar la estimación del flujo vehicular en una zona, mediante el análisis de una secuencia de imágenes aéreas obtenidas desde una cámara de video instalada en un *UAV*.

1.3.2 Objetivos específicos

- Investigar los diferentes métodos de reconocimiento, clasificación y registro de imágenes para su análisis y selección del más apropiado para el proyecto.
- Generar una base de datos de imágenes con vista aérea que cuenten con los objetos a reconocer y clasificar.
- Implementar un método de reconocimiento y clasificación para la detección de objetos mediante regiones de interés.
- Aplicar una metodología de estimación de flujo óptico para determinar el desplazamiento de los objetos detectados.

- Desarrollar un algoritmo de compensación de movimiento global de la imagen con la finalidad de reducir el margen de error en la estimación del desplazamiento de los objetos.
- Calcular la densidad y la velocidad vehicular de la zona analizada.

1.4 Justificación

Hoy en día, se cuenta con una gran cantidad de recursos audiovisuales, los cuales continúan creciendo exponencial día con día; un ejemplo es la utilización de un *UAV*. Estos *UAV* ofrecen la posibilidad de recopilar información en tiempo real por medio de la instalación de cámaras para el monitoreo del tráfico vehicular, permitiendo un mayor control. Como un dispositivo aéreo útil y poderoso, los *UAV* han estado desempeñando roles importantes en la adquisición de datos e imágenes; por ejemplo, se han utilizado ampliamente en la investigación de agricultura, geología y cinematografía [5], [6], [7]. En comparación con los sensores tradicionales ubicados en el suelo o en cámaras de ángulo bajo, los *UAV* presentan muchas ventajas, como bajo costo, fácil implementación, alta movilidad, gran alcance de vista, escala uniforme, entre otras. Los *UAV* pueden proporcionar información como la velocidad, la densidad y el flujo del promedio de tráfico en un área determinada, con la cual se pueden realizar procedimientos de monitoreo y análisis con el fin de tomar acciones para mitigar esta situación, de manera que el poder controlar el tráfico de un área determinada impactaría en los tiempos de llegada de los diferentes vehículos, detección de incidentes (choques), mejoramiento de rutas y desviación vehicular a rutas alternativas.

Dado lo anterior, se presentan los *UAV* como una solución particularmente atractiva para llevar a cabo el análisis de flujo de tráfico. Además de ser una tecnología mucho más económica que las tradicionales, permiten gran cantidad de ventajas, entre las cuales se pueden mencionar las siguientes: pueden trasladarse fácilmente a cualquier punto en donde se necesite analizar el tráfico vehicular, su rango de cobertura es mayor a las de las cámaras convencionales, además de que los *UAV* pueden ajustar su visión en el momento que se desee.

Entre los beneficios que se podrían obtener se pueden mencionar las siguientes (sin ser los únicos):

- Solución a problemas de congestionamiento y contaminación del medio ambiente: Por ejemplo, mejora en la calidad del aire.
- Seguridad: Para la prevención de accidentes.
- Sociales: Reflejado en una mejor calidad de vida de la población.
- Económicos: Tanto para el gobierno, para la iniciativa privada y para la sociedad en general, entre otros.

Esta solución, de acuerdo a la clasificación de desarrollo sostenible de la ONU, aporta en los siguientes puntos: industria, innovación e infraestructura, ciudades y comunidades sostenibles, y acción por el clima [8].

1.4.1 Delimitaciones

En este apartado se especifican los alcances y limitaciones con las que cuenta el desarrollo de software.

1.4.2 Alcances

El algoritmo se desarrollará en un equipo de escritorio o servidor, y contará con las siguientes funciones:

- El algoritmo será capaz de encontrar e identificar los vehículos presentes en el video.
- El algoritmo será capaz de conocer dónde ha estado un vehículo a lo largo de su “tiempo de vida” en el video.
- El algoritmo será capaz de compensar los desplazamientos producidos por el movimiento de la cámara.
- El algoritmo será capaz de construir la escena completa analizada.
- El algoritmo será capaz de calcular el desplazamiento de los vehículos en el video.
- El algoritmo será capaz de calcular la velocidad de los vehículos.
- El algoritmo será capaz de calcular la velocidad global de la escena.

Para obtener un algoritmo que pueda ser utilizado en el monitoreo de una zona específica (como una glorieta), se tomarán las ventajas de cada una de las técnicas estudiadas. Se aplicará el algoritmo de estimación de flujo vehicular a señales enviadas por el *UAV* de tal manera que únicamente se utiliza el video capturado, considerando que éstos estarán observando la misma escena por un tiempo determinado, y que durante este tiempo, pueden llegar a presentar movimientos a causa del viento o algún cambio de posición controlado.

1.5 Limitaciones

- No se contará con un *UAV*.
- El desarrollo del algoritmo únicamente será realizado de manera serial.
- La base de datos de imágenes utilizada es limitada para algunos tipos de vehículos.
- El algoritmo provee únicamente un aproximado de la velocidad real del vehículo.
- Se trabajará con el lenguaje C++, OpenCV y la plataforma Visual Studio Community, lo que en cierto momento pudiera dar limitantes al proyecto.

Capítulo 2. Marco Teórico

Para gestionar de manera óptima y eficiente la movilidad de las ciudades, y debido al gran crecimiento vehicular, actualmente es necesario el uso de tecnología y una eficiente operación y gestión de ésta. Gracias a que actualmente la tecnología es muy accesible para el público en general, y al incremento en la capacidad de cómputo con el que se cuenta actualmente, es muy factible su utilización para resolver diferentes problemas de visión por computadora. Entre estos problemas, se pueden destacar la detección del movimiento, la estabilización de video, y el reconocimiento y clasificación de objetos.

2.1 Detección del movimiento

2.1.1 Procesamiento de video

Una imagen puede ser definida como una función de dos dimensiones $f(x,y)$, donde x e y son coordenadas espaciales de un plano, donde f es considerada la intensidad de color en un punto de la imagen. Se considera una imagen digital un grupo definido valores de intensidad, las cuales tiene una posición particular.

El concepto de procesamiento de imagen está basado en el mejoramiento de las imágenes entre las cuales se encuentra el resaltar la información contenida. La representación más simple de una imagen es una colección de puntos en un arreglo bidimensional, donde para cada punto se almacena una serie de valores propios de la imagen. A cada punto de la imagen se le llama pixel, el cual representa el color o irradiancia en la posición de salida correspondiente, lo que quiere decir que estos pixeles son parte del rectángulo en una imagen. La Figura 3 muestra un ejemplo de

representación de una imagen de dos dimensiones donde cada cuadro representa un pixel y su intensidad de color.

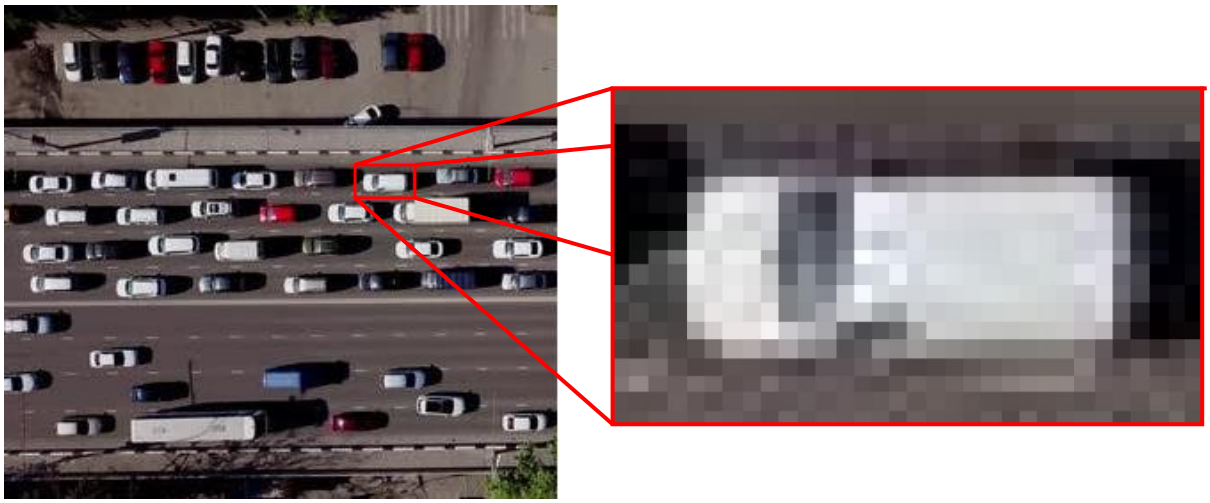


Figura 3. Representación de pixeles en una imagen.

La detección y el seguimiento de rostros humanos, peatones o movimiento vehicular son ahora aplicaciones comunes. Contar con información previa como el movimiento de la cámara y el intervalo de tiempo entre imágenes consecutivas, puede ayudar a elegir una técnica de análisis de movimiento adecuada. Existen diferentes tipos de problemas relacionados con el movimiento, de los cuales se pueden mencionar los siguientes [9]:

1. Detección de movimiento: Es el problema más simple ya que se registra cualquier movimiento detectado.

2. La detección y ubicación de objetos en movimiento: Una cámara generalmente en posición estática registra objetos en movimiento o estáticos; si solo se requiere la detección, dirección y la predicción de su trayectoria futura de los objetos que se encuentran en movimiento, la solución puede tomar en cuenta métodos de segmentación basados en movimiento.

El adquirir simplemente un determinado número de imágenes consecutivas en un determinado instante del tiempo hace posible la detección del movimiento; si se asume que la cámara se encuentra de manera estática, y la intensidad de iluminación no varía entre imágenes, se puede realizar una simple sustracción de imágenes. Al aplicar una simple diferencia entre dos imágenes binarias consecutivas se obtiene una imagen en la cual los valores distintos de cero representan áreas de la imagen en movimiento.

2.1.2 Enfoque básico

La manera más simple de detectar cambios entre dos imágenes I_1 e I_2, las cuales pertenecen a la misma escena, es realizar una comparación pixel por pixel. La manera más común es calcular una imagen I_{12} que contenga esta diferencia. La comparación de la imagen I_1 con la imagen posterior I_2 la cual incluye un movimiento, da como resultado que los pixeles que no tuvieron un movimiento (pixeles estacionarios) se cancelen dejando únicamente los pixeles distintos a cero, los cuales corresponden a los puntos no estacionarios [10]. Por lo tanto, la diferencia entre dos imágenes en tiempo t_i y t_j puede ser definida por:

$$d_{ij}(x) = \begin{cases} 1, & si \ \left| f(x,y,t_i) - f\left(x,y,t_j\right) \right| > T \\ 0, & otro \end{cases}, \qquad (1)$$

donde T es un umbral previamente especificado. La Figura 4 muestra un ejemplo de la aplicación de este método.

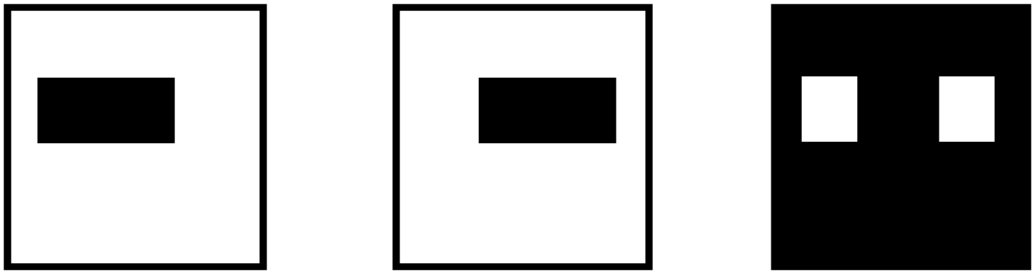


Figura 4. Enfoque básico de detección de movimiento. (a) Posición del objeto al tiempo t_1. (b) Posición del objeto al tiempo t_2. (c) Diferencia entre las imágenes al tiempo t_2.

2.1.3 Detección de puntos característicos

2.1.3.1 Método de Harris

Actualmente, existe un vasto número de algoritmos de detección de bordes, los cuales son utilizados para el emparejamiento (*matching*). Teniendo en cuenta que los bordes no son sensibles a cambios de intensidad, presentan algunos problemas al aplicarles otras transformaciones [11].

El método de detección de Canny [12], se enfoca en la detección de bordes que se encuentren cercanos a un umbral de detección; esta forma de implementar la detección provoca que un pequeño cambio en la intensidad del borde o en el pixelado dé como resultado un gran cambio en la topología del borde. Por lo tanto, al realizar una búsqueda, es muy probable el encontrar errores.

Para poder realizar búsquedas sin presentar errores, el método de Harris busca determinar mediante la extracción y seguimiento de esquinas muy poco susceptibles a cambios de rotación y escala de una imagen, la descripción de superficies y objetos en un nivel superior. Para ello, el método se enfoca en realizar un filtrado de bordes confiable (es decir, consistente).

Aplicando un filtrado de tipo Gaussiano a la imagen desplazando una ventana de ocho direcciones, se pueden encontrar tres tipos de regiones, como se puede apreciar en la Figura 5:

- Plana: En la cual no existen cambios para ninguna dirección.
- Borde: Sin cambios en dirección del propio borde.
- Esquina: Con cambios en todas direcciones, los cuales se presentan de manera significativa.

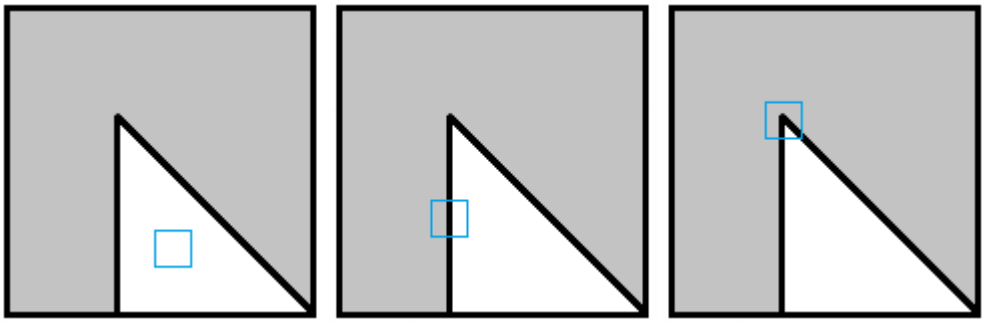


Figura 5. Ejemplo de áreas planas, borde y esquina.

Aplicando una matriz de correlación M, si se define λ_1 y λ_2 como los valores propios (*eigenvalores*) de la matriz M calculada, se podrán obtener los tres tipos de regiones comentadas anteriormente, por lo tanto:

- Si ambas curvaturas son pequeñas, de modo que la función de autocorrelación local es plana, entonces la región de la imagen tiene una intensidad aproximadamente constante.
- Si una curvatura es alta y la otra baja, de modo que la función de autocorrelación local tiene forma de cresta, entonces solo los cambios a lo largo de la cresta causan un pequeño cambio: esto indica un borde.

- Si ambas curvaturas son altas, de modo que la función de autocorrelación local tiene un pico pronunciado, los cambios en cualquier dirección aumentarán: esto indica una esquina.

Estas regiones pueden ser visualizadas como un mapa en función de los valores propios λ_1, λ_2, tal y como se muestra en la Figura 6.

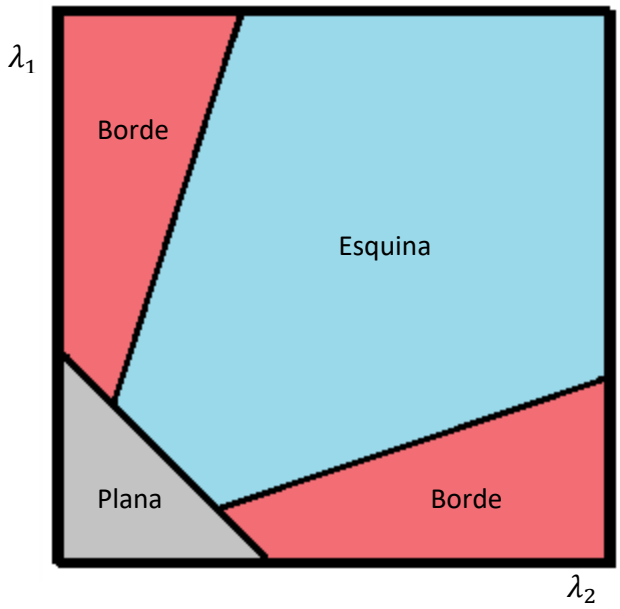


Figura 6. Regiones en función de los valores propios λ_1, λ_2 de la matriz de correlación M.

2.1.3.2 Método Shi-Tomasi

El detector de esquinas Harris tiene un criterio de selección de esquinas el cual calcula una puntuación para cada pixel a partir de dos valores propios y, si la puntuación supera un determinado valor, el pixel se marca como una esquina. En el año de 1994, Jianbo Shi y Carlo Tomasi [13] tomaron el algoritmo de Harris y eliminaron la función para calcular las puntuaciones de los pixeles a partir de sus valores propios,

proponiendo el uso únicamente de los valores propios para comprobar si el pixel es un borde, esquina o plano.

Este método muestra cómo monitorear la calidad de las características de la imagen y el cambio de apariencia de una característica entre el primer *frame* y el actual; este método propone una forma numéricamente sólida y eficiente de determinar cambios afines mediante un procedimiento de minimización al estilo Newton-Raphson para ofrecer las mejores características a las cuales aplicarles un seguimiento.

Como la cámara presenta movimientos, la imagen puede presentar cambios en la escena; a esto se le conoce como “movimiento de la imagen” (*Image motion*):

$$I(x, y, t + h) = I(x - \varepsilon(x, y, t, h), y - n(x, y, t, h)), \tag{2}$$

donde la imagen al tiempo $t + h$ puede ser obtenida desplazando la imagen actual o base moviendo cada pixel (desplazamiento global). La cantidad del movimiento $\delta = (\varepsilon, n)$ es lo que se conoce como desplazamiento de $\vec{x} = (x, y)$. Teniendo una ventana de características, se tiene no uno sino diferentes desplazamientos dentro de la misma.

La representación afín del vector de desplazamiento δ puede ser vista de la siguiente manera:

$$\delta = D\vec{x} + \vec{h}, \tag{3}$$

donde

$$D = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{bmatrix}, \tag{4}$$

es la matriz de deformación y $\vec{h}$ es la traslación del centro de la ventana. Por lo tanto, la coordenada $\vec{x}$ de la imagen I es movida al punto $A\vec{x} + \vec{h}$ en la segunda imagen J, donde $A = \mathbb{I} + D$ con $\mathbb{I}$ como la matriz de identidad de 2×2:

$$J(A\vec{x} + \vec{h}) = I(\vec{x}) \tag{5}$$

Utilizar ventanas pequeñas puede presentar variaciones de intensidad más pequeñas dentro de ellas y por lo tanto poco fiables, sin embargo, es preferible ya que son menos propensas a caer en situaciones donde los objetos presentan diferente profundidad en la escena. Por esta razón, la aplicación del modelo de traslación es preferible al realizar seguimiento, ya que con ello se puede considerar la matriz de deformación D como cero obteniendo:

$$\delta = \vec{h}. \tag{6}$$

Debido a la existencia de ruido y que el modelo afín de movimiento no brindan resultados perfectos, se puede aplicar un proceso de minimización del error con base en la ecuación (5)

$$\varepsilon = \iint_W \left[J(A\vec{x} + \vec{h}) - I(\vec{x})\right]^2 w(\vec{x})\, d\vec{x}, \tag{7}$$

donde W es la ventana de características y $w(\vec{x})$ es una función de peso, alternativamente, w podría ser una función similar a la de Gauss g para enfatizar el área central de la ventana. Para minimizar el residuo, se diferencia con respecto a las entradas desconocidas de la matriz de deformación D y el vector de desplazamiento $\vec{h}$ y se establece el resultado en cero

$$\frac{1}{2}\frac{\partial \epsilon}{\partial D} = \int\int \left[J\left(A\vec{x} + \vec{h}\right) - I(\vec{x})\right] g\vec{x}^T \, w d\vec{x} = 0$$
$$\frac{1}{2}\frac{\partial \epsilon}{\partial \vec{h}} = \int\int \left[J\left(A\vec{x} + \vec{h}\right) - I(\vec{x})\right] g \, w d\vec{x} = 0, \tag{8}$$

donde

$$g = \left(\frac{\partial J}{\partial x}, \frac{\partial J}{\partial y}\right) = \left(g_x, g_y\right),$$

de esta manera se puede linearizar el sistema mediante la expansión de Taylor

$$J\left(A\vec{x} + \vec{h}\right) = J(\vec{x}) + g^T(\vec{u}), \tag{9}$$

produciendo un sistema lineal de 6×6

$$T\vec{z} = \vec{a}, \tag{10}$$

donde $\vec{z}^T = \begin{bmatrix} d_{xx} & d_{yx} & d_{xy} & d_{yy} & h_x & h_y \end{bmatrix}$ las cuales corresponden a la matriz de deformación D y el desplazamiento $\vec{h}$, quedando el vector de error de la siguiente manera:

$$\varepsilon = \iint_W [I(\vec{x}) - J(\vec{x})]^2 \begin{bmatrix} xg_x \\ xg_y \\ yg_x \\ yg_y \\ g_x \\ g_y \end{bmatrix} w \, d\vec{x} \tag{11}$$

El cual depende de la diferencia entre las dos imágenes, y la matriz T de 6×6, que se puede calcular a partir de una imagen, esto se puede escribir como:

$$T = \iint_W \begin{bmatrix} U & V \\ V^T & Z \end{bmatrix}, \tag{12}$$

donde

$$U = \begin{bmatrix} x^2 & x^2 g_x g_y & x y g_x^2 & x y g_x g_y \\ x^2 g_x g_y & x^2 g_y & x y g_x g_y & x y \\ x y g_x^2 & x y g_x g_y & y^2 g_x^2 & y^2 g_x g_y \\ x y g_x g_y & x y g_y^2 & y^2 g_x g_y & y^2 g_y^2 \end{bmatrix},$$

$$V^T = \begin{bmatrix} x^2 g_x^2 & x g_x g_y & y g_x^2 & y g_x g_y \\ x g_x g_y & x g_y^2 & y g_x g_y & y g_y^2 \end{bmatrix},$$

$$Z = \begin{bmatrix} {g_x}^2 & g_x g_y \\ g_x g_y & g_y \end{bmatrix}.$$

La ecuación (10) apenas proporciona una aproximación, de modo que para obtener un resultado satisfactorio se puede utilizar de manera iterativa mediante el método de Newton-Raphson. Debido a que la matriz D es cero, se puede determinar que la matriz D y el desplazamiento $\vec{h}$ interactúan con la matriz V^T, por lo que cualquier error en D causaría un error en $\vec{h}$, por lo tanto el sistema

$$Z\vec{h} = \vec{e}, \tag{13}$$

puede ser resuelto por los últimos valores del vector $\vec{e}$ de la ecuación (10).

Independientemente del método utilizado en el seguimiento, no todas las partes de la imagen contienen información del movimiento (problema de apertura) [9]. Es decir, para un borde horizontal, únicamente se puede detectar el movimiento vertical.

En la Figura 7 se muestra un ejemplo simple de un objeto rectangular gris el cual se mueve de manera diagonal a la escena (recuadro negro). Si se aplica un análisis por movimiento diferencial, únicamente se detectará el movimiento horizontal, lo que vuelve imposible determinar el movimiento por completo.

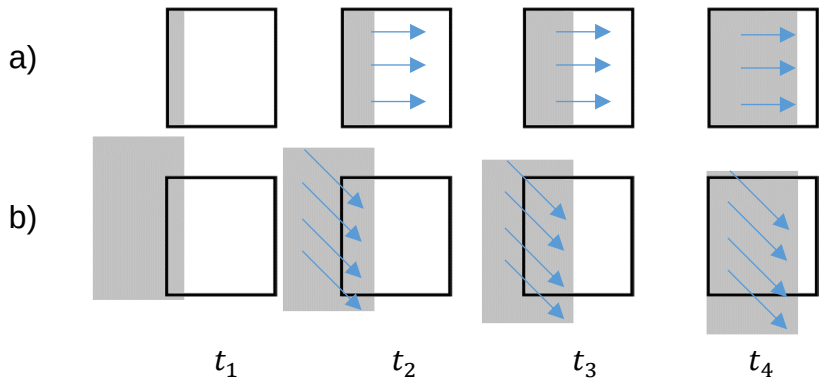


Figura 7. Problemas de construcción de campos de movimiento. (a) Posición y movimiento del objeto en la escena en los tiempos t_1, t_2, t_3 y t_4. (b) Posición y movimiento real del objeto en los tiempos t_1, t_2, t_3 y t_4.

Debido a este problema, el método se enfoca en rastrear esquinas o ventanas con un contenido de alta frecuencia espacial o regiones donde alguna mezcla de derivadas de segundo orden es suficientemente alta. Dado lo anterior, *Shi-Tomasi* define que “una buena característica es aquella que se puede rastrear bien, de modo que el criterio de selección sea óptimo por construcción” [13].

Si el sistema dado por la ecuación (13) presenta buenas medidas y se puede llegar a una solución confiable, entonces es posible rastrear una ventana de una imagen a otra. Por lo tanto, la matriz Z debe estar por encima del nivel de ruido de la imagen, esto quiere decir que ambos valores propios de Z deben ser grandes.

Dado lo anterior:

- Si se tienen dos valores propios pequeños, significa un perfil de intensidad aproximadamente constante dentro de una ventana, por lo tanto, representa una región plana.
- Un valor propio grande y uno pequeño corresponden a un patrón de textura unidireccional, lo que representa una región del borde.

- Dos valores propios grandes pueden representar esquinas, texturas de “sal y pimienta” o cualquier otro patrón que se pueda rastrear de manera confiable.

En conclusión, si los dos valores propios de Z, λ_1 y λ_2, se acepta la región como esquina si

$$\min(\lambda_1, \lambda_2) > \lambda, \tag{14}$$

donde λ es un umbral establecido de manera predeterminada.

Estas regiones pueden ser visualizadas como un mapa en función de los valores propios λ_1, λ_2, tal y como se muestra en la Figura 8.

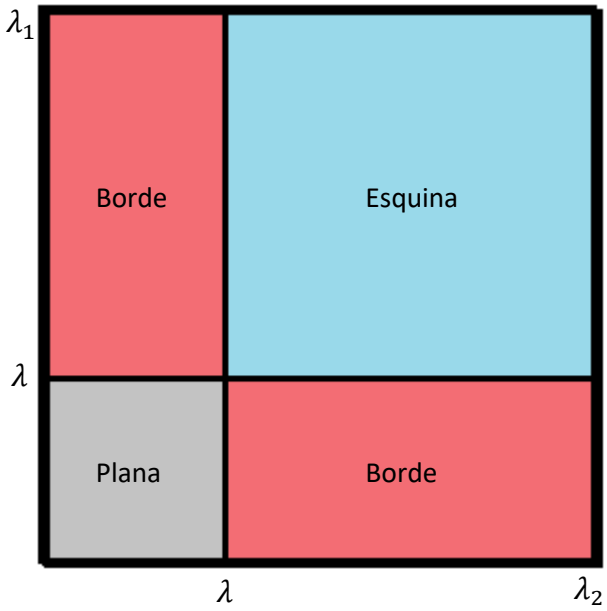


Figura 8. Regiones en función de los valores propios λ_1, λ_2 de la matriz de correlación M.

2.1.4 Descriptores de puntos característicos

2.1.4.1 Transformación de características invariantes de escala

El descriptor *SIFT (Scale-invariant feature transform)* [14] es un algoritmo desarrollado para detectar puntos característicos en una imagen. Estos puntos característicos tienen la propiedad de ser invariantes a transformaciones como la traslación, escalamiento, rotación, cambios de iluminación y transformaciones afines. Está compuesto principalmente de las siguientes cuatro etapas:

1. Detección de extremo a escala de espacio: Se aplica el algoritmo siguiendo una metodología piramidal, cuyo objetivo es el de identificar ubicaciones y escalas que pueden asignarse repetidamente bajo diferentes vistas de la misma escena u objeto, esto con el objetivo de sobresaltar aquellos puntos que son invariantes a los cambios de orientación y escala.
2. Localización de los puntos de interés: Una vez obtenidas las diferencias Gaussianas, se procede a encontrar los puntos extremos; para ello se realiza una comparación entre el punto analizado y sus 26 vecinos. El punto que se encuentra bajo análisis corresponde a un extremo, si es el mayor o el menor de sus 26 vecinos. Una vez que se encuentra un candidato clave, se realiza un ajuste detallado a los datos cercanos para determinar: ubicación, escala y relación de curvaturas principales.
3. Asignación de la orientación: Se calcula la magnitud y orientación del gradiente alrededor de los puntos de interés con los cuales se obtiene un histograma de direcciones de gradiente local en la escala seleccionada. El

histograma tiene 36 contenedores que cubren el rango de orientaciones de 360 grados.

4. Descriptores del punto de interés: Por lo tanto, el descriptor de cada punto de interés se encuentra formado por un vector que contiene los valores de todas las orientaciones del histograma, las cuales corresponden al tamaño de las flechas mostradas en el recuadro de la derecha en la Figura 9.

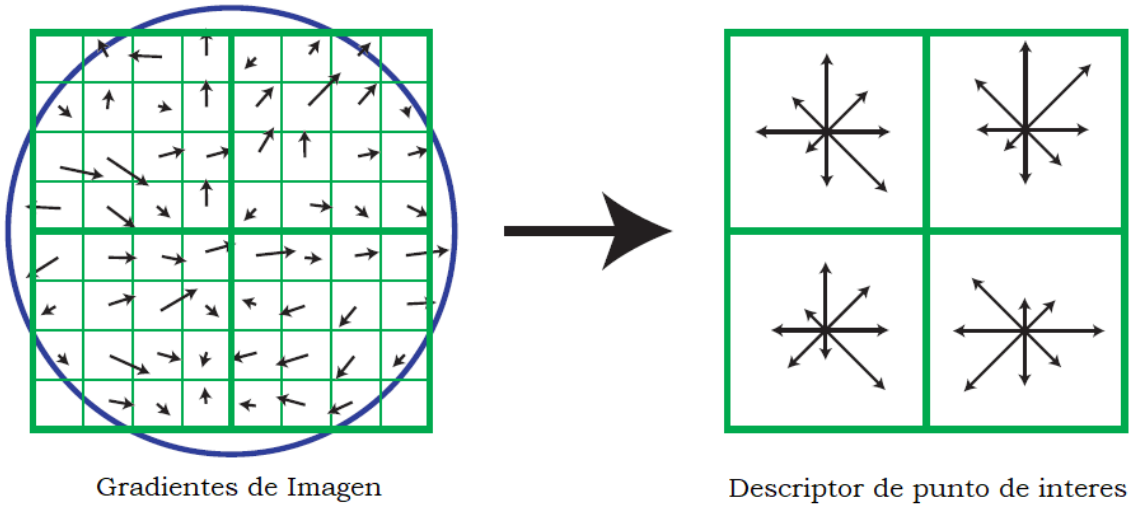


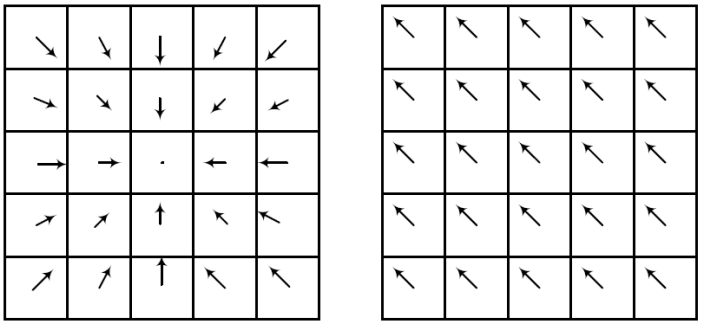
Figura 9. Matriz de descriptores de 2x2 calculada a partir de un conjunto de muestras de 8x8. Tomado de [14].

2.1.5 Flujo óptico

2.1.5.1 Movimiento global

En este caso, considerando que la cámara se desplaza en un plano, y que los objetos están fijos y el fondo es constante, se obtiene que todos los píxeles se mueven en la misma dirección, por lo tanto, cada uno de ellos tiene el mismo vector de movimiento. El movimiento que se puede encontrar entre dos *frames* sucesivos puede ser descrito mediante un solo vector de movimiento, lo cual simplifica enormemente el cálculo. Otro movimiento que puede existir es el que ocurre cuando la cámara se aleja o acerca a

la escena, estos movimientos se ven reflejados en todo el marco de la imagen, como se puede apreciar en la Figura 10.



a) b)

Figura 10. Campos de movimiento global. a) Acercamiento. b) Traslación. Tomado de [15].

El vector de movimiento global (*GMV, Global Motion Vector*) $(GMVx_t, GMVy_t)$ para el pixel actual con sus coordenadas (x_t, y_t) se determina como

$$\begin{cases} GMVx_t = x_t' - x_t \\ GMVy_t = y_t' - y_t \end{cases}, \tag{15}$$

donde (x_t', y_t') son las coordenadas deformadas en el marco de referencia por los parámetros de movimiento global de la coordenada (x_t, y_t) [15].

Una de las técnicas para determinar el desplazamiento de la imagen es la evaluación de diferentes desplazamientos; esto es por ejemplo, tomando la imagen $I(x)$ de $N \times N$ y teniendo una matriz de desplazamientos $\vec{h}$ candidatos de $M \times M$, se evalúa cada desplazamiento $\vec{h}$, por lo que se evaluarían $N^2 \times M^2$ veces y se toma la que tiene el desplazamiento más cercano al deseado. Esta técnica es realizada por fuerza bruta, lo que involucra mucho tiempo de cómputo, lo cual se vuelve poco eficiente si se desea aplicar para la obtención de resultados en tiempo real.

2.1.5.2 Algoritmo secuencial de detección de similitudes

Otra técnica es la aplicación del algoritmo secuencial de detección de similitudes [16], en donde se evalúan un número determinado de desplazamientos $\vec{h}$ para los cuales se realiza un análisis en el que se acumula el error de la diferencia entre la imagen I_1 y la imagen I_2, de manera que si el error para el desplazamiento $\vec{h}_i$ al hacer la evaluación supera un umbral de error establecido como aceptable, se descarta y se evalúa el siguiente desplazamiento.

En la técnica presentada por Lucas-Kanade, se inicia con un valor de $\vec{h}$ estimado con el cual se utiliza el gradiente espacial en cada punto de la imagen para modificar la estimación de la $\vec{h}$ y encontrar una mejor coincidencia.

2.1.5.3 Registro de imagen por el método de Lucas-Kanade

La técnica de registro de una imagen presentada por Bruce D., Lucas y Takeo Kanade [16] se basa en el procesamiento de imagen mediante el uso del gradiente espacial de las imágenes para encontrar la mejor coincidencia utilizando la iteración Newton-Raphson. Esta técnica presenta una búsqueda rápida de una mejor coincidencia, ya que la búsqueda se hace sobre menos coincidencias a valorar.

El análisis de una imagen se puede ver de la siguiente manera (ver Figura 11):

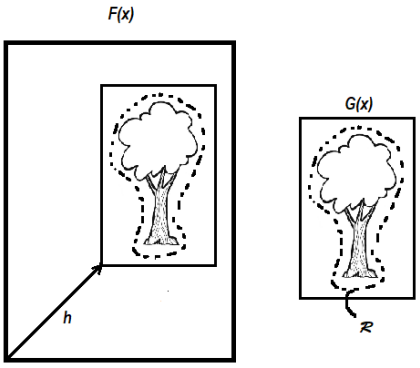


Figura 11 El problema de registro de la imagen. Tomado de [16]

Donde $F(\vec{x})$ y $G(\vec{x})$ son los respectivos pixeles de cada imagen en la posición $\vec{x}$, donde $\vec{x}$ es un vector. Lo que se desea encontrar, es la diferencia mínima del desplazamiento $\vec{h}$, esto entre la aplicación del desplazamiento $\vec{h}$ a la $F(\vec{x})$ y comparándolo con la región $G(\vec{x})$, esto quiere decir que $F(\vec{x}+\vec{h})$ debe ser aproximadamente $G(\vec{x})$. Un cálculo típico de la diferencia entre $F(\vec{x}+\vec{h})$ y $G(\vec{x})$ es el uso de la norma entre éstas:

$$L_1\ norm = \sum\nolimits_{x\varepsilon R} \left|F\left(\vec{x}+\vec{h}\right) - G(\vec{x})\right|, \tag{16}$$

$$L_2\ norm = \left(\sum\nolimits_{x\varepsilon R}\left[F\left(\vec{x}+\vec{h}\right) - G(\vec{x})\right]^2\right)^{1/2}, \tag{17}$$

y el negativo de la correlación:

$$C = \frac{\sum_{x\varepsilon R} F\left(\vec{x}+\vec{h}\right)G(\vec{x})}{\left(\sum_{x\varepsilon R} F\left(\vec{x}+\vec{h}\right)^2\right)^{1/2}\left(\sum_{x\varepsilon R} G(\vec{x})^2\right)^{1/2}}. \tag{18}$$

2.1.5.4 Algoritmo de registro - una Dimensión

Lo primero que se debe hacer es aplicar el algoritmo para el caso de una dimensión, donde se busca la diferencia entre dos curvas $F(\vec{x})$ y $G(\vec{x}) = F(\vec{x} + \vec{h})$ (ver Figura 12).

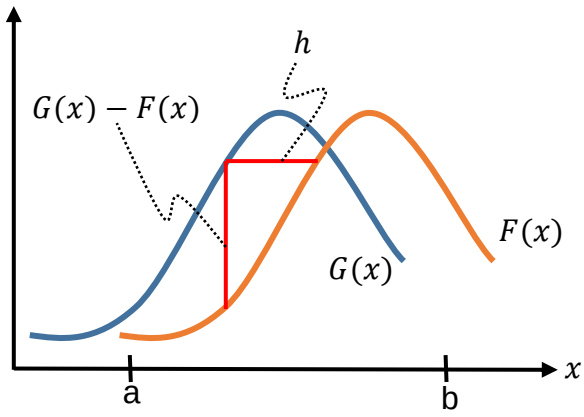


Figura 12. Dos curvas a emparejar. Tomado de [16].

La solución a esto depende de la aproximación del comportamiento de $F(\vec{x})$ en los vecinos de $\vec{x}$, por lo que para una $\vec{h}$ muy pequeña se tiene que

$$F'(x) \approx \frac{F(\vec{x} + \vec{h}) - F(\vec{x})}{\vec{h}} \approx \frac{G(\vec{x}) - F(\vec{x})}{\vec{h}} \tag{19}$$

$$\vec{h} \approx \frac{G(\vec{x}) - F(\vec{x})}{F'(\vec{x})} \tag{20}$$

Para que el algoritmo tenga éxito se requiere que $\vec{h}$ sea lo suficientemente pequeño, esto va a depender de $\vec{x}$. Un método común para la estimación de $\vec{h}$ aproximada es la utilización del promedio de los desplazamientos $\vec{h}_x$ (20), por lo que:

$$\vec{h} \approx \left(\sum_x \vec{h}_x\right) / (\sum_x 1) \approx \left(\sum_x \frac{G(\vec{x}) - F(\vec{x})}{F'(\vec{x})}\right) / (\sum_x 1). \tag{21}$$

Tomando la ecuación (19) se le asigna un peso a cada $\vec{h}_x$ para determinar su contribución al promedio $\vec{h}$, El peso estará dado por $w(\vec{x}) = |F''(\vec{x})|^{-1}$ con

$$F'(\vec{x}) \approx \frac{G(\vec{x}) - F(\vec{x})}{\vec{h}}, \tag{22}$$

$$F''(\vec{x}) \approx \frac{G'(\vec{x}) - F'(\vec{x})}{\vec{h}}, \tag{23}$$

descartando $1/\vec{h}$ en (23) se tiene el peso dado por:

$$w(\vec{x}) \approx \frac{1}{|G'(\vec{x}) - F'(\vec{x})|}. \tag{24}$$

Se aplica el peso $w(\vec{x})$ a la ecuación (21)

$$\vec{h} \approx \left(\sum_x w(\vec{x})\vec{h}_x\right) / \left(\sum_x w(\vec{x})\right) \approx \left(\sum_x \frac{w(\vec{x})[G(\vec{x}) - F(\vec{x})]}{F'(\vec{x})}\right) / \left(\sum_x w(\vec{x})\right). \tag{25}$$

Teniendo esta estimación de $\vec{h}$ se puede determinar el valor mínimo aplicando el método de Newton-Raphson para encontrar el mejor $\vec{h}$

$$\vec{h}_0 = 0$$

$$\vec{h}_{k+1} = \vec{h}_k + \left(\sum_x \frac{w(\vec{x})\left[G(\vec{x}) - F\left(\vec{x} + \vec{h}_k\right)\right]}{F'\left(\vec{x} + \vec{h}_k\right)}\right) / \left(\sum_x w(\vec{x})\right). \tag{26}$$

2.1.5.5 Generalización a múltiples dimensiones

Para generalizar el algoritmo de registro para múltiples dimensiones se debe minimizar el promedio de error L_2, donde $\vec{x}$ y $\vec{h}$ son vectores lineales

$$F\left(\vec{x} + \vec{h}\right) \approx F(\vec{x}) + \vec{h}\frac{\delta}{\delta\vec{x}}F(\vec{x}), \tag{27}$$

donde $\frac{\delta}{\delta \vec{x}}$ es el operador de gradiente con respecto x, como vector de columna,

$$\frac{\delta}{\delta \vec{x}} \approx \left[\frac{\delta}{\delta \vec{x}_1} \frac{\delta}{\delta \vec{x}_2} \frac{\delta}{\delta \vec{x}_3} \dots \frac{\delta}{\delta \vec{x}_n}\right]^T. \tag{28}$$

Usando la ecuación (28) para aproximar el error:

$$0 = \frac{\delta E}{\delta \vec{h}} \approx \frac{\delta}{\delta \vec{h}} \sum_x \left[F(\vec{x}) + h\frac{\delta F}{\delta \vec{h}} - G(\vec{x})\right]^2$$

$$\approx \sum_x 2\frac{\delta F}{\delta \vec{h}}\left[F(\vec{x}) + \vec{h}\frac{\delta F}{\delta \vec{h}} - G(\vec{x})\right], \tag{29}$$

donde se puede determinar h obteniendo:

$$\vec{h} \approx \frac{\sum_x \left(\frac{\delta F}{\delta h}\right)^T [G(\vec{x}) - F(\vec{x})]}{\sum_x \left(\frac{\delta F}{\delta \vec{h}}\right)^T \left(\frac{\delta F}{\delta \vec{h}}\right)}. \tag{30}$$

2.2 Reconocimiento y clasificación de objetos

2.2.1 Análisis de componentes principales

El análisis de componentes principales (*PCA, Principal Component Analysis*) tiene como enfoque el tomar un conjunto de datos que cuentan con un amplio número de variables y aplicarle una reducción, manteniendo la mayor cantidad posible de la variación del conjunto inicial. Esto se realiza creando un nuevo conjunto de datos formado inicialmente por las variables que tienen la mayor parte de la variación presente en todas las variables originales [17].

Este método al manejar información estadística permite simplificar espacios de muestra, los cuales cuentan con múltiples dimensiones manteniendo su información. Suponiendo que se tiene un grupo de personas X, cada una con un determinado número de características $\{x_1, \dots, x_n\}$, por lo tanto, el espacio muestral tiene n

dimensiones. El *PCA* ofrece la posibilidad de encontrar un determinado número de características $\{\hat{x}_1, \dots, \hat{x}_m\}$, que puedan definir aproximadamente el espacio muestral P, donde $m < n$. Estos valores $\hat{x}$ reciben el nombre de componente principal. Por lo tanto, este método permite el reducir la información original en pequeñas cantidades de componentes, lo que lo convierte en un método útil para la aplicación de técnicas como *clustering* [17].

Lo primero a realizar es localizar el punto x_0 para el cual la suma de las distancias de todas las características $\{x_1, \dots, x_n\}$ al punto sea mínima, este punto x_0 se conoce como centro de masa (ver Figura 13).

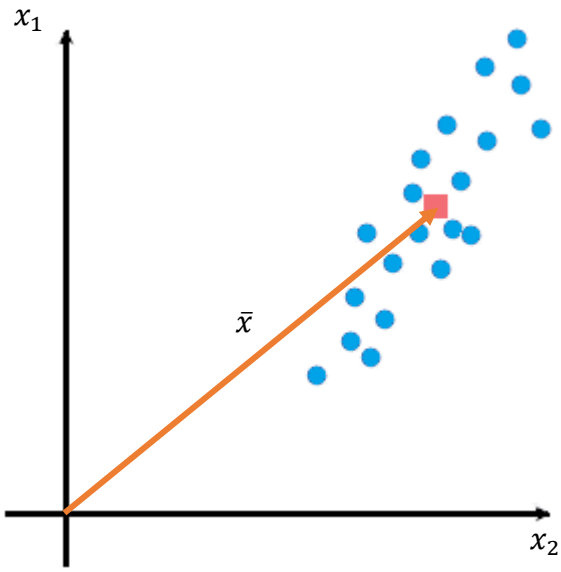


Figura 13. Búsqueda de punto x_0 con error mínimo.

Lo siguiente es encontrar aquella línea (la mejor línea) que tiene un punto base en el centro de masa y la misma dirección que el vector propio con el valor

propio más grande de la matriz de dispersión (ver Figura 14).

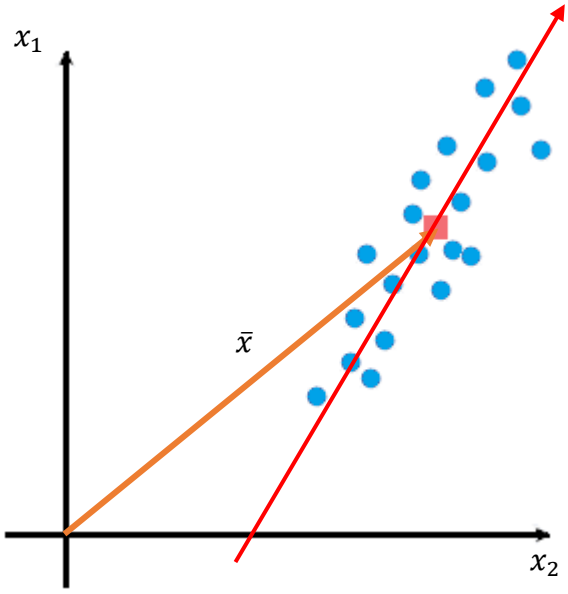


Figura 14. Búsqueda de la línea con error mínimo en la reconstrucción.

2.2.2 Detección de objetos en tiempo real usando el método YOLO

You Only Look Once (*YOLO*) [18] es una detección avanzada de objetos de aproximación el cual aplica una única Red Neuronal Convolucional (*CNN, Convolutional Neural Network*) a toda la imagen, lo que la divide aún más en cuadrículas (ver Figura 15) a las cuales se les aplica una predicción y puntuación de confianza respectivamente. Estos cuadros delimitadores se analizan mediante la puntuación de confianza prevista.

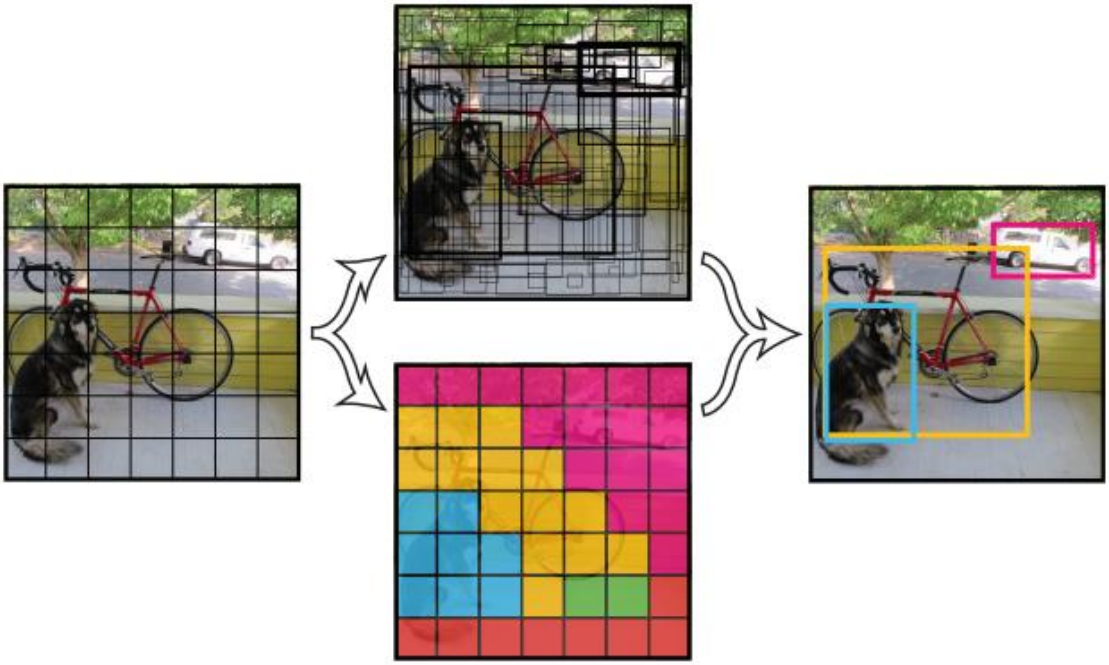


Figura 15. División de la imagen en una cuadrícula uniforme y simultáneamente se predice los cuadros delimitadores, la confianza en esos cuadros y las probabilidades de clase. Tomado de [18].

La Arquitectura de *YOLO* cuenta con 24 capas convolucionales y 2 capas completamente conectadas. Las capas convolucionales iniciales de la red extraen características de la imagen, mientras que las capas completamente conectadas predicen las probabilidades y coordenadas de salida. En la Figura 16 se muestra dicha arquitectura.

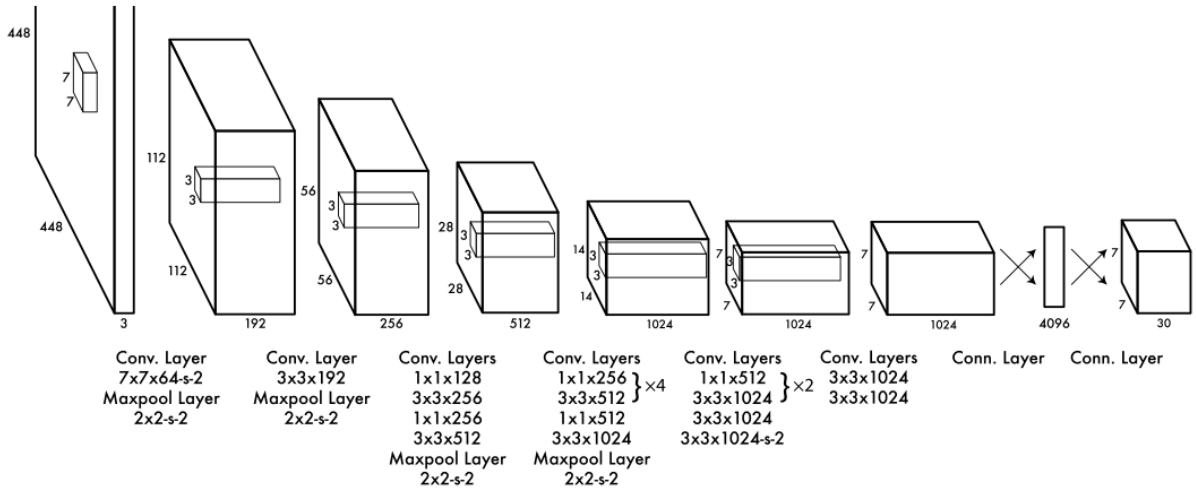


Figura 16. Arquitectura de la *CNN* de *YOLO*. Tomado de [19].

YOLO toma una imagen de entrada y la redimensiona a 448×448 pixeles pasándola por la *CNN*, dando como resultado un tensor de $7 \times 7 \times 30$, el cual contiene la información de las coordenadas del rectángulo delimitador y la distribución de probabilidad en todas las clases para las que está entrenado el sistema.

2.2.3 Entrenamiento del método YOLO

Para el entrenamiento *YOLO* se utiliza un conjunto de datos dividido en archivos de etiquetas; estos archivos están formados por una imagen y un archivo ".txt", donde a cada imagen le corresponde el archivo ".txt" que lleva su mismo nombre. Cada archivo de etiquetas cuenta con un listado de objetos que aparecen en la imagen asociada con el siguiente formato:

$$< Clase - Objeto > < x\_centro > < y\_centro > < ancho > < alto >$$

donde

- $< Clase - Objeto >$ - Número entero que va de 0 a $Clases - 1$

- $< x\_centro >$, $< y\_centro >$ - Corresponde al punto central del objeto con valores flotantes resultado de la normalización entre $[0,1]$.
- $< ancho >$, $< alto >$ - Corresponde al ancho y alto del objeto con valores flotantes resultado de la normalización entre $[0,1]$.

Los valores anteriores son obtenidos mediante la implementación de la siguiente relación:

$$< x\_centro > = < x\_objeto > / < ancho\_imagen >$$

$$< ancho > = < ancho\_objeto > / < ancho\_imagen >$$

Adicionalmente, *YOLO* requiere algunos archivos para comenzar a entrenar, los cuales son:

- Número total de clases.
- Archivo de texto *".names"* con los nombres de todas las clases.
- Archivo de texto *".data"* con la ruta a todas las imágenes a utilizar para el entrenamiento.
- La ruta para guardar archivos de pesos resultado del entrenamiento de la *CNN*.
- Un archivo de configuración *".cfg"* con todas las capas de la arquitectura *YOLO*.
- Pesos convolucionales previamente entrenados.

El valor de los filtros para la penúltima capa no es arbitrario y depende del número total de clases; este valor está dado por:

$$filtros = 5 * (2 + Numero\_de\_clases). \quad (31)$$

2.3 Estabilización de video

2.3.1 Transformación proyectiva

Si se tiene un mapeo lineal inyectivo $U: E \rightarrow F$ entre dos espacios vectoriales E y F, entonces U mapea rayos de E sobre rayos de F, por lo tanto, induce un mapeo $P(U): P(E) \rightarrow P(F)$ entre los espacios del cociente $P(E)$ y $P(F)$. Este mapeo se llama mapeo proyectivo y transformación proyectiva si es biyectivo [20]. Dada alguna elección de marco de coordenadas, la transformación proyectiva también llamada homografía se puede representar convenientemente mediante matrices. Considerando dos espacios n-dimensionales $P(E)$ y $P(E')$, sea $\boldsymbol{P}$ el vector de coordenadas del punto P en $P(E)$ y $\boldsymbol{P}'$ el vector de coordenadas del $P' = P(E')$; si H es una matriz no singular de $(n+1) \times (n+1)$ entonces la ecuación

$$\boldsymbol{P}' = H\boldsymbol{P}, \tag{32}$$

define una homografía entre los puntos P de $P(E)$ y los puntos P' de $P(E')$. Por lo tanto, toda homografía puede ser representada por una matriz homogénea no singular de $(n+1) \times (n+1)$.

Este trabajo de tesis utiliza la homografía específicamente para la proyección sobre un plano [21]; tomando la ecuación (32) aplicando $n = 2$, se tiene

$$\begin{pmatrix} P_1' \\ P_2' \\ P_3' \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \end{pmatrix}. \tag{33}$$

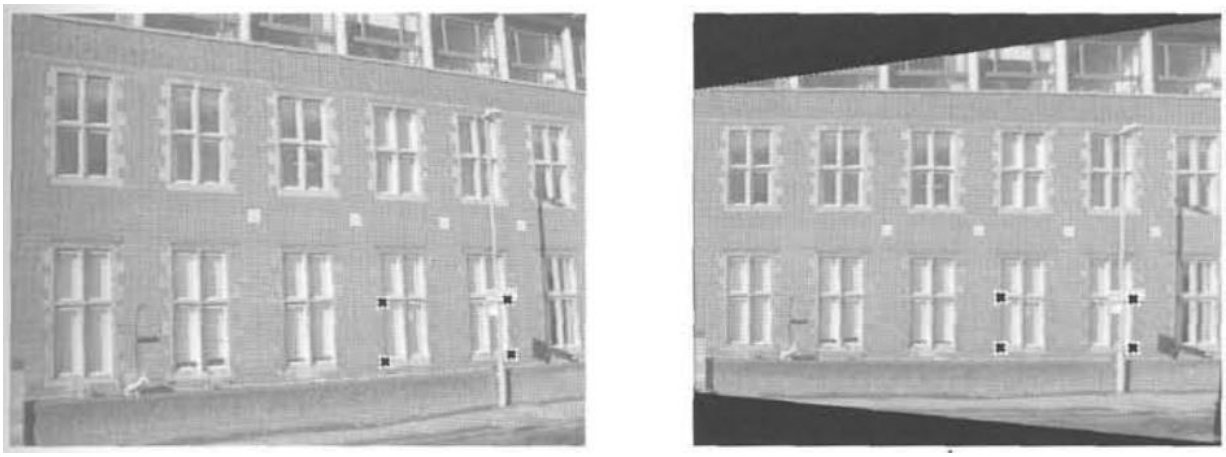
Observando que en la ecuación (33) la matriz H puede ser multiplicada por un valor de escala arbitrario no cero sin alterar la matriz de proyección, confirmando que la matriz H es una matriz homogénea. Un plano está determinado de forma única por

tres puntos que se encuentran en él, pero un marco proyectivo para ese plano está definido por cuatro puntos: tres puntos fundamentales que forman un triángulo no degenerado, y un punto unitario que no se encuentra en uno de los bordes de este triángulo.

2.3.2 Eliminando la distorsión proyectiva.

Ver la imagen bajo cierta perspectiva puede distorsionar su forma. En la Figura 17 se puede observar que las formas de las ventanas no son rectangulares, aunque en la realidad si lo son. Viendo la imagen completa se puede notar que en general las líneas paralelas no aparecen de esta forma en la imagen.

Una imagen de proyección central de un plano (o sección de un plano) está relacionada con el plano original a través de una transformación proyectiva, por lo tanto, la imagen es una distorsión proyectiva del original. Es posible "deshacer" esta transformación proyectiva calculando la transformación inversa y aplicándola a la imagen. El resultado será una nueva imagen sintética en la que se muestran los objetos del plano con su forma geométrica correcta [21]. La Figura 17b muestra la aplicación de esta transformación inversa aplicada a la Figura 17a.



a) b)

Figura 17. a) Imagen original con distorsión. b) Imagen sintética modo vista frontal. Tomado de [21].

2.4 Trabajos relacionados

El trabajo realizado en [22] consiste principalmente en la aplicación de cuatro etapas, en las cuales se desarrolla un conjunto de clasificadores para la detección de vehículos, y un método robusto de estimación de parámetros de flujo de tráfico basado en el flujo óptico y la teoría del flujo de tráfico.

Debido a que en trabajos previos suponen que los puntos característicos calculados provienen principalmente del fondo del video en lugar de los vehículos (primer plano); en casos donde el tráfico se vuelve denso, esta suposición puede no ser válida, ya que los vehículos con poco movimiento pueden perderse en el fondo ocasionando una estimación inexacta. Estas consideraciones motivaron a los autores a pensar en la idea de integrar el flujo óptico con la detección de vehículos supervisada basada en el aprendizaje.

Las dos primeras etapas están basadas en una combinación del método *Haar cascade* [23] y *CNN* dando como resultado un conjunto de clasificadores. En la tercera

y cuarta etapa se realiza una compensación del movimiento del fondo con respecto a los vehículos detectados, y posteriormente la estimación de los parámetros de flujo del tráfico.

Para el aprendizaje de su clasificador se utilizó una base de datos de 20,000 imágenes aéreas de vehículos y segmentos de carretera. Esta base de datos fue dividida en 18,000 imágenes para el entrenamiento de la CNN y 2,000 para pruebas (40% imágenes positivas y 60% imágenes negativas).

- Etapa 1: El clasificador *Haar cascade* fue entrenado con la librería de OpenCV 2.4.12, estableciendo un tamaño de imagen a 60×40 pasando por 7 etapas de 2500 muestras para entrenamiento (1000 muestras positivas y 1500 muestras negativas). La tasa mínima de aciertos y la tasa máxima de falsos positivos por etapa se establecen en 0.999 y 0.5, respectivamente.
- Etapa 2: Las regiones de interés propuestas en la etapa 1 son analizadas por una *CNN* con el fin de finalizar la clasificación del vehículo.
- Etapa 3: En esta etapa el *frame* ya puede ser divido en tráfico (dentro de las ventanas de detección) y fondo (fuera de las ventanas de detección), por lo tanto, el movimiento del tráfico se estima mediante la resta del movimiento promedio del vehículo y el movimiento de fondo promedio.
- Etapa 4: Para la estimación del flujo de tráfico, se toman las dimensiones conocidas de un objeto en el mundo real y se obtiene la relación en pixeles en la imagen (pixel-real), esto para poder realizar el cálculo del desplazamiento del objeto y la velocidad. La velocidad del vehículo es calculada como su

movimiento por la velocidad del *frame* por la relación pixel-real del tamaño del objeto.

La Figura 18 muestra un resumen del proceso implementado en este trabajo.

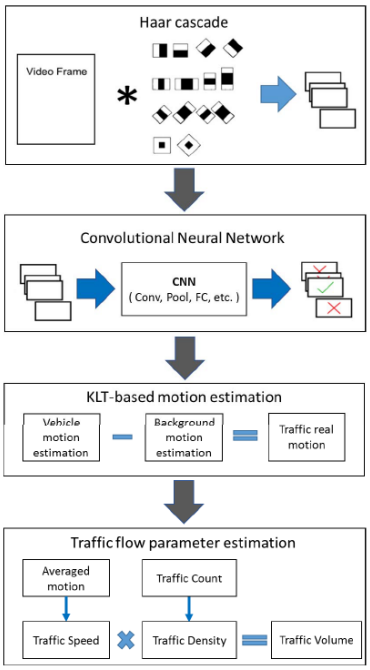


Figura 18. Resumen de las cuatro etapas del proceso. Tomado de [22].

Se realiza la aplicación del método a dos videos uno tomado por un *UAV* que se movía sobre un segmento de autopista, monitoreando una autopista de tres carriles con un flujo de tráfico fluido, y el segundo tomado en una arteria urbana donde el tráfico era denso. Los autores consideran cinco fotogramas razonables como intervalo de medición. Básicamente, si el intervalo fuera demasiado pequeño, el error de medición manual de la velocidad sería grande, y si fuera demasiado grande, la resolución de los datos se deterioraría. El procesamiento alcanza la velocidad de 25 *frames* por segundo (fps) para los videos analizados.

Los resultados mostraron que los métodos propuestos lograron una muy buena precisión de estimación y velocidad de procesamiento en tiempo real tanto en escenarios de flujo libre como de tráfico congestionado.

Por otra parte, el trabajo presentado en [24] se basa en un método con el cual se puede realizar la detección de vehículos en una imagen de 21 MPixeles en pocos segundos sin necesidad de aplicar procesamiento paralelo. Con este método es posible detectar, tanto la región del vehículo, como la dirección y tipo de vehículo (camión, carro).

El método está estructurado en dos fases, la primera es la aplicación de un detector binario y la segunda aplicando un clasificador multi-clase al resultado de la primera fase ofreciendo un alto rendimiento, velocidad de ejecución y la orientación del vehículo.

La caja que delimita los vehículos es detectada en la primera etapa mediante el detector binario anteriormente mencionado; esto se realiza deslizando una ventana de características a través de las imágenes integrales por cada canal de color (*Viola and Jones* [23], ver Figura 19) y un clasificador *AdaBoost* con una estructura *soft Cascade*. En esta etapa aún no se cuenta con información acerca del tipo ni orientación del vehículo.

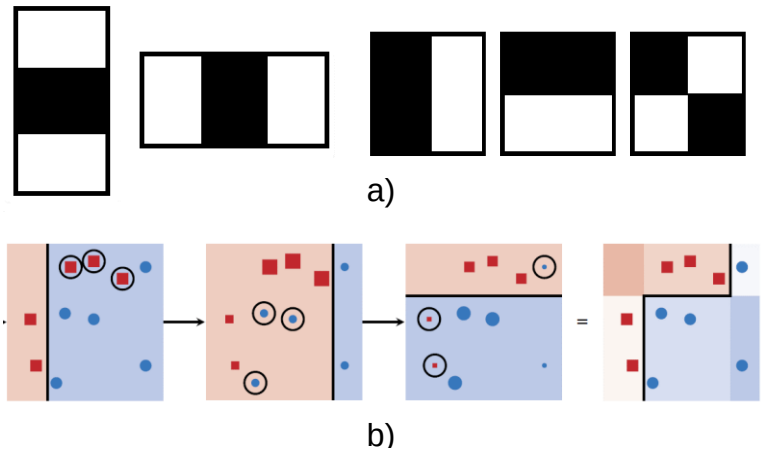


Figura 19. a) Características *Haar Cascade* principales. b) Representación visual del clasificador AdaBoost obtenido después de tres iteraciones. Tomado de [17].

Para la detección de la orientación se proponen dos métodos: el primero consiste en entrenar un clasificador único, el cual utiliza todas las variaciones de las clases con el fin de que sea capaz de detectar vehículos orientados de manera diferente. El otro, es agregar diferentes clasificadores simples, cada uno entrenado con una clase de manera que sea sensible a una orientación específica, implementando estos clasificadores individuales de manera consecutiva, y al final armar la colección de vehículos detectados por cada uno de ellos.

La clasificación se realiza en dos pasos, formados por un estimador de orientación y un clasificador de tipos; primero se envía una muestra al estimador de orientación, y finalmente, se procesa mediante el clasificador de tipo para identificar a qué categoría de tipo pertenece este vehículo.

En la estimación de orientación se cuenta con direcciones agrupadas, cada grupo es considerado como una clase; con ello las características se calculan mediante la función de histograma de gradientes orientados (*HOG*, *Histogram of*

Oriented Gradients) y usando una red neuronal con capa oculta como clasificador de clases múltiples.

Una vez determinada la orientación del vehículo, se procede a la clasificación mediante un clasificador de tipos para determinar si es carro o camión. Se gira el cuadro delimitador (vehículo) en dirección horizontal según la estimación de orientación previa, se extraen las características *HOG* y se clasifica nuevamente por el clasificador de tipos.

En la Figura 20 se puede observar el esquema completo de la metodología propuesta para este método.

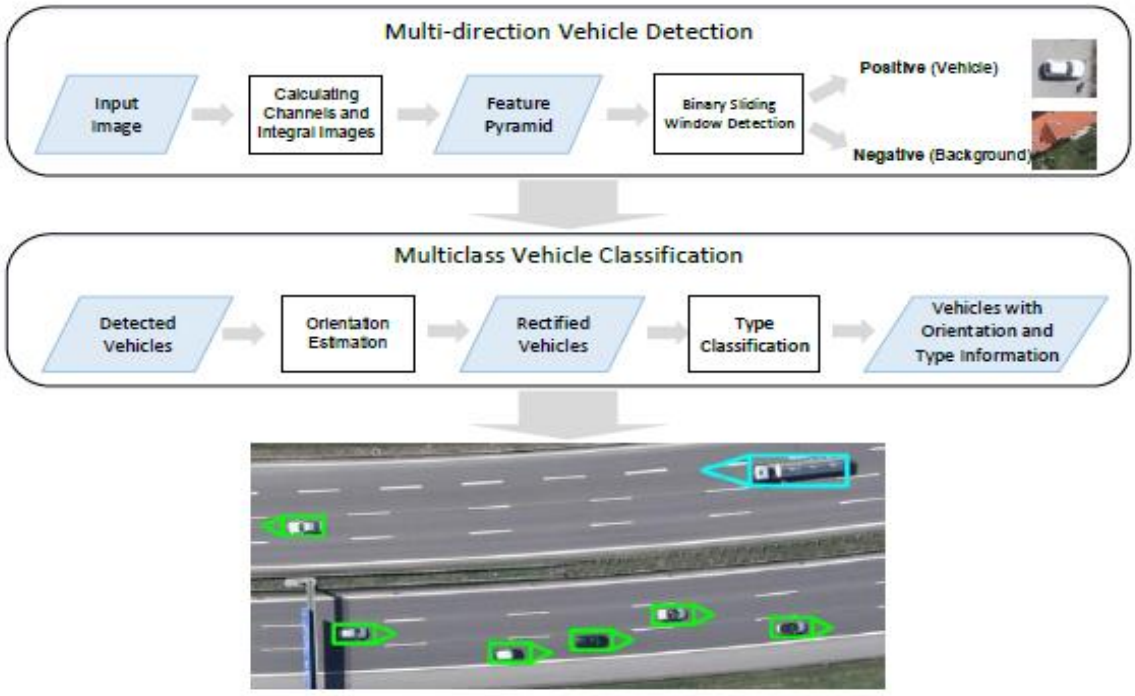


Figura 20. Metodología de detección de vehículos. Tomado de [24].

En [25] se presenta una herramienta para la edición de video en tiempo real, *Augmentation Virtual Screen* (*AVScreen*). Fue desarrollada para la aplicación de anuncios, comerciales, videos musicales, películas, entre otros. Esta metodología

utiliza un proceso para segmentar el video en primer plano (*FG, Foreground*) y fondo (*BG, Background*).

Los problemas centrales en la aplicación del *AVScreen* es la estabilización de video, la segmentación de video y el cálculo del panorama. Como resolver estos problemas dependerá de la adquisición del video; algunas características generales a considerar son:

- La cámara está estática.
- El video se graba desde una única ubicación.
- La cámara solo tiene movimientos de rotación o traslación.
- Las regiones del primer plano son más grandes que las regiones del fondo.
- Todos los *frames* están disponibles simultáneamente.

Teniendo estas restricciones iniciales, se van eliminando las restricciones hasta la aplicación de la metodología a videos obtenidos con cámaras inestables en tiempo real, dando como resultado:

1. Un método robusto de segmentación de video *FG-BG*.
2. Un procedimiento de estabilización en tiempo real para videos en cámaras inestables.
3. Un procedimiento para la composición de un panorama que requiere interacción con el usuario y un proceso en tiempo real.

El proyecto utiliza una segmentación probabilística basada en los modelos cuadráticos de campos de medida de Markov (*QMMF*, *Quadratic Makov Measure*

Field), utilizando un mapa de etiquetas (región de interés, región no interesante y una no etiquetada).

La aplicación de homografía plana para el aumentado de la imagen es fundamental, esto debido a que se realiza una transformación proyectiva que mapea puntos de un plano a otro. Para calcular una homografía entre dos imágenes, se requieren al menos cuatro puntos correspondientes (esquinas) que se encuentran en un plano.

Para la realización del aumentado de imagen se siguen cinco pasos generales:

1. Definición y marcado del mapa de etiquetas.
2. Selección de puntos (esquinas) para la aplicación de la homografía.
3. Cálculo de la homografía (proyección de una imagen sobre otra).
4. Segmentación binaria interactiva (*QMMF*).
5. Composición o armado de imagen: Imagen de referencia más la imagen proyectada (paso 3).

Implementar como paso adicional la segmentación de vídeo de *FG-BG* entre los pasos 4 y 5 aporta la detección de objetos en movimiento en el fondo para que dichos objetos puedan sobreponerse a la imagen aumentada.

Cuando se trabaja con videos obtenidos de cámaras inestables, se vuelve difícil la modificación de una región; para atacar este problema, se emplea la homografía como recurso para la estabilización del video. El proceso de estabilización es realizado de la siguiente manera.

1. Se considera que la región de interés es siempre visible en el video.
2. Se establece una imagen base para establecerla de referencia.

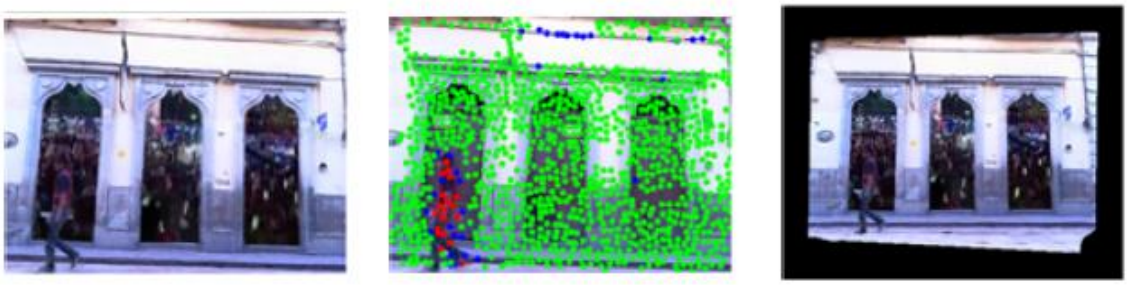
3. Mediante el rastreo de los puntos de interés de la imagen de referencia (paso 2) en las imágenes posteriores, se utiliza esta relación para realizar la proyección de la imagen analizada hacia la imagen de referencia.
4. Se aplica la segmentación de vídeo de *FG-BG*.
5. Se calcula la composición o armado de imagen.

Con la aplicación de la estabilización se puede tomar información suficiente para la construcción del panorama apreciado durante el video, por lo que como funcionalidad adicional se incluye como parte del procedimiento al analizar el video.

Para el cálculo del panorama o mosaico lo primero a considerar es que la cámara cuenta únicamente con movimientos de rotación y traslación, el fondo dominara la escena y todos los *frames* del video serán almacenados en memoria simultáneamente. Para la creación del mosaico se siguen los siguientes pasos:

1. Definición y marcado del mapa de etiqueta.
2. Definición y cálculo de homografía entre el panorama y la primera imagen.
3. Segmentación binaria interactiva.
4. Segmentación de vídeo de *FG-BG*, similar a procesos anteriores solo que para este caso es entre el *frame* actual y el panorama.
5. Se calcula la composición o armado de imagen.

Los resultados demostraron que el AVScreen, ofrece buenos resultados en sus aplicaciones, tanto en videos con poco movimiento, como en videos con cierto grado de inestabilidad. La Figura 21 presenta la aplicación del proceso de estabilización y construcción del panorama de manera paralela al aumentado de imagen realizado en las puertas del frente del edificio.



a) b) c)

Figura 21. Representación del proceso al tiempo t. a) Imagen inestable. b) Selección de puntos para estabilización (verdes) y puntos de objeto en movimiento (rojo). c) Estabilización y panorama de la imagen al tiempo t. Tomado de [25].

El sistema DataFromSky [26] es una plataforma de análisis de video basada en la nube para un análisis de tráfico a partir de videos tomados por drones o cámaras estándar. El resultado del análisis de video es un pequeño extracto en forma de trayectorias que contiene datos detallados sobre cada objeto en el video. Hoy en día, DataFromSky es una plataforma confiable y probada que se utiliza en muchos proyectos comerciales y de investigación en más de 90 países. Este sistema para su implementación requiere que al momento de realizar la grabación del video el *UAV* no presente desplazamientos ni acercamientos de la cámara provocados por los lentes o por el ascenso o descenso del *UAV.* El sistema presenta problemas a cambios de iluminación y ángulo de grabación de la imagen (ver Figura 22). Las imágenes de la cámara deben ser estables y libres de vibraciones o deformaciones de la imagen.

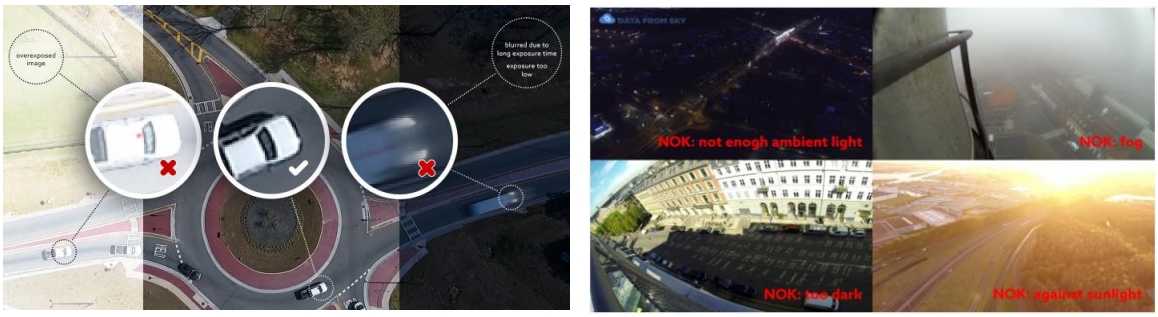


Figura 22. Cambios de iluminación. Tomado de [26].

Para el sistema, la resolución juega un papel muy importante para la detección y clasificación de objetos. El tamaño mínimo de un objeto dentro de la imagen (para su detección) es de 15x15 píxeles, mientras que para su correcta clasificación se necesita una resolución de 30x30 píxeles. Un tamaño recomendado para los objetos dentro de la imagen es entre 30x30 píxeles y hasta 150x150 píxeles en toda el área monitoreada dentro de la intersección (ver Figura 23).

Figura 23. DataFromSky: Representación de un objeto en imagen. Tomado de [26].

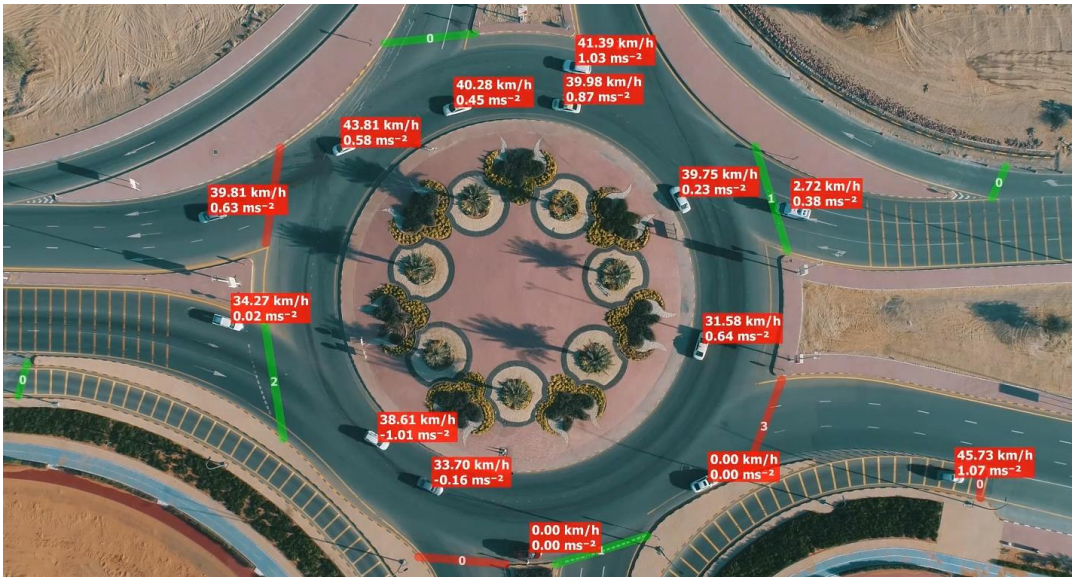


Figura 24. DataFromSky: vista aérea de glorieta. Tomado de [26].

Capítulo 3. Desarrollo

Realizando un análisis de los trabajos existentes, es claro que existen diferentes técnicas para el análisis y adquisición de información a partir de un video; tomando como referencia algunos de estos trabajos, se plantea la integración de diferentes conceptos para poder llegar al objetivo de este trabajo de tesis: establecer una metodología que integre la detección de movimiento, reconocimiento de objetos, estabilización y construcción de panorama, la cual jugará un papel clave para la realización de este trabajo de tesis.

3.1 Propuesta

A continuación, se presenta la metodología a implementar para la estimación del tráfico vehicular, para ello, se considerarán los siguientes requisitos, de manera que se tenga un ambiente controlado:

- Una grabación de un video en cuya calidad se puedan visualizar correctamente los objetos en movimiento (vehículos), de preferencia en HD.
- Altitud mínima de vuelo es de 50 m y máximo de 80 m.
- Posición estacionaria sobre un punto único (velocidad cero y altitud de vuelo constante).

El procedimiento, como se puede observar en la Figura 25, consta de dos pasos principales [22]:

1. Detección de vehículos
    a. Reconocimiento de vehículos: reconocimiento de las regiones donde se encuentra el vehículo.

b. Almacenamiento de historial: posición a de un vehículo mientras fue observado en la escena.

2. Estimación de flujo vehicular
    a. Estimación de flujo óptico: Cálculo del desplazamiento del vehículo.
    b. Estimación de velocidad: Cálculo de la velocidad de un vehículo al desplazarse de una posición a otra entre dos *frames*.

La aplicación de esta metodología se basa en la detección de los vehículos para los cuales se determinará su movimiento a lo largo de todo el video, con el fin de determinar el historial de cada vehículo; este historial es el que brindará la información necesaria para poder estimar su desplazamiento a lo largo de un determinado tiempo y con esto determinar su velocidad.

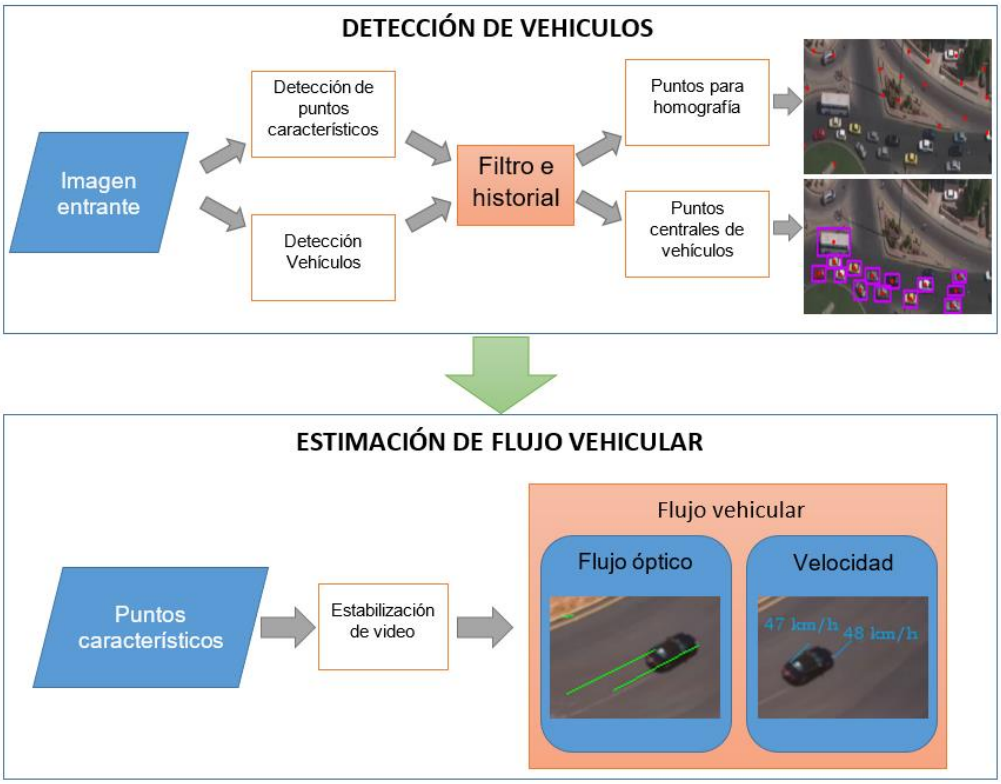


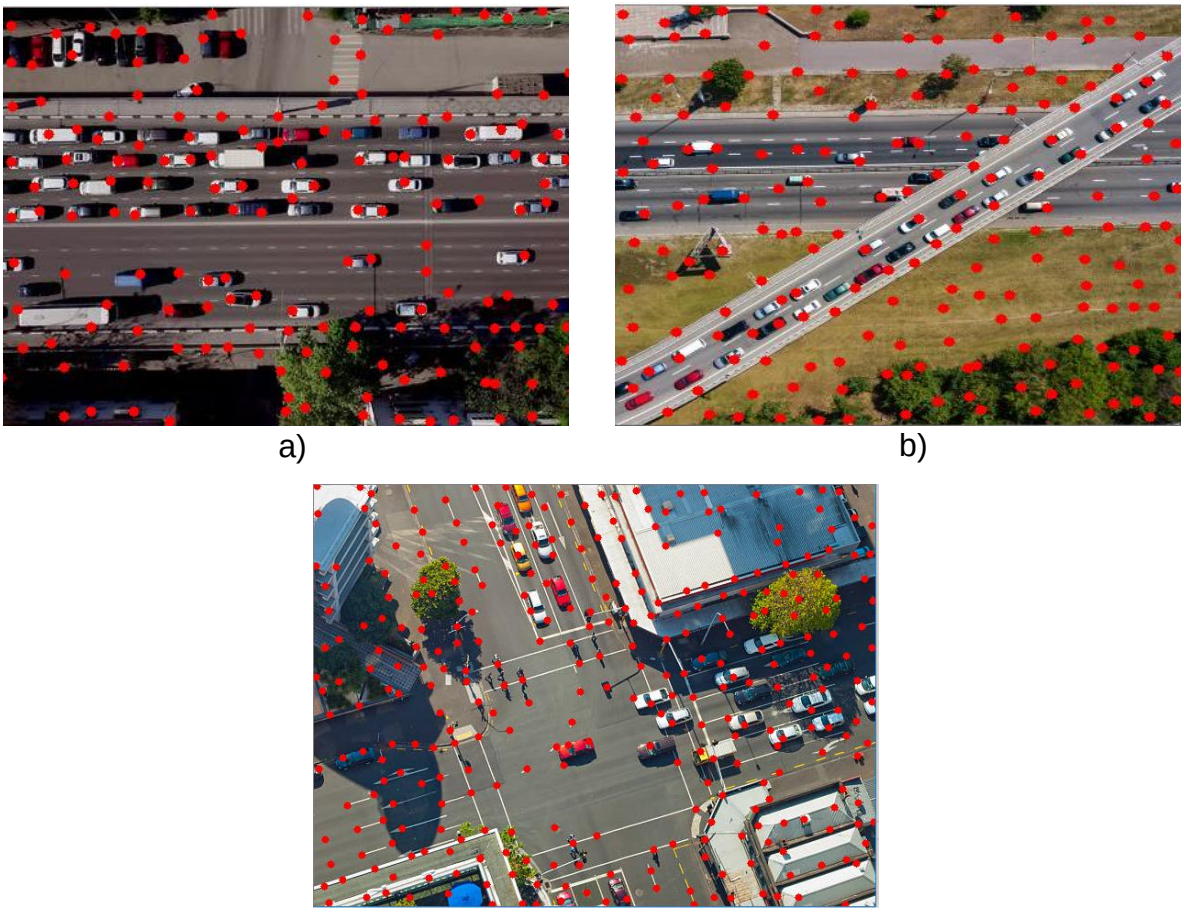
Figura 25. La detección de vehículos muestra de manera general la implementación de la clasificación de puntos para estabilización y flujo vehicular. La estimación de velocidad representa de manera general el cálculo de la estabilización y flujo vehicular.

3.2 Detección de vehículos

Para determinar los vehículos existentes en una imagen, se pueden implementar diferentes técnicas, como *PCA*, *Haar Cascade Classifier*, entre otros. Para este trabajo, se utilizará un conjunto de clasificadores, implementando el método de "*Shi-Tomasi*", el cual consiste en la detección de puntos característicos de una imagen y la aplicación de una *CNN* por medio del método *YOLO* para reconocer y clasificar los objetos presentes en la imagen de acuerdo al entrenamiento aplicado a la red neuronal.

3.2.1 Metodo Shi-Tomasi

Este método se implementa en el lenguaje C++, utilizando Visual Studio 2017 como entorno de desarrollo; para el desarrollo se implementa la librería de OpenCV utilizando la función *cv2::goodFeaturesToTrack*. En la Figura 26 se pueden observar tres imágenes aplicando el método *Shi-Tomasi* para la de búsqueda de puntos característicos a una captura aérea con una distancia mínima entre puntos de 20 pixeles, y un máximo de 1000 puntos característicos a detectar en cada imagen.



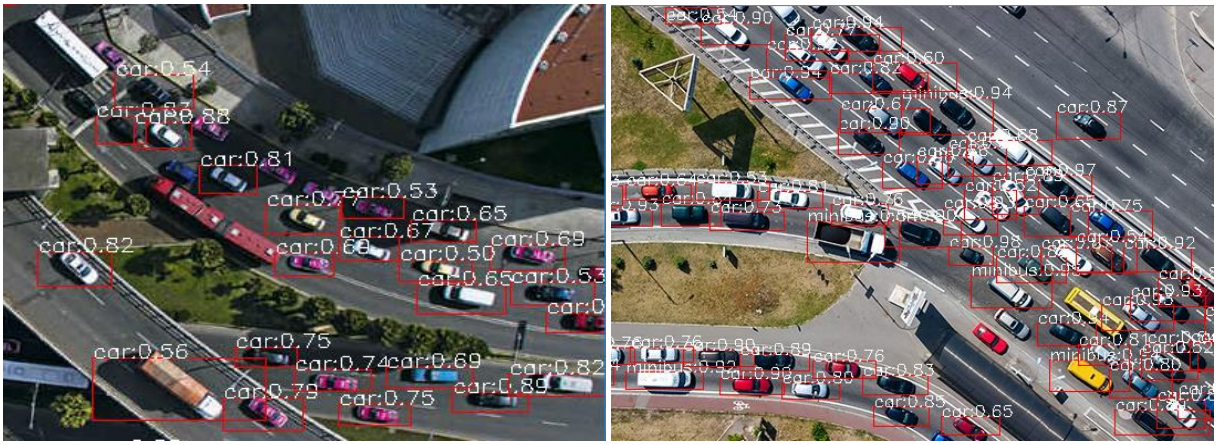
a) b)

c)

Figura 26. Aplicación de método goodFeaturesToTrack por medio de OpenCV.

3.2.2 Clasificación *YOLO*

Al igual que el método *Shi-Tomasi*, la implementación del *YOLO* se realiza en el lenguaje C++ utilizando Visual Studio 2017, utilizando los pesos calculados previamente mediante las funciones *dnn::Net*, *dnn::readNetFromDarknet, dnn::NMSBoxes*. En la Figura 27 se presenta un ejemplo de detección y clasificación de objetos.



a) b)

Figura 27. Aplicación de clasificación utilizando *YOLO*.

3.2.2.1 Conjunto de datos y entrenamiento

Para el entrenamiento de la *CNN* del *YOLO* se utilizó un total de 231 imágenes; cada imagen puede contener objetos de una o más clases, por lo tanto, debido a que la aparición de un objeto de una clase varia por imagen y teniendo en cuenta que el entrenamiento del YOLO se basa en las imágenes ingresadas y no el número de objetos totales por clase, las imágenes son divididas en 70% entrenamiento y 30% validación con la finalidad de contar con al menos un número de objetos de cada clase en ambas partes. Para la configuración de esta base de datos se utilizó el programa

Yolo_Mark [27]; este programa permite la selección de un área para identificar la región de la imagen en la cual se encuentra un determinado objeto perteneciente a una clase. En la Figura 28 se pueden apreciar ejemplos de la implementación del programa. Para el entrenamiento de la red neuronal se utilizó la información generada con el *Yolo_Mark* mediante la aplicación *darknet* [28] y el servidor Quadro8000 perteneciente al Centro de Investigación en Matemáticas, A.C. (CIMAT), el cual cuenta con las siguientes características:

- Intel(R) Xeon(R) Gold 5222 3.80 GHz.
- Ubuntu 18.04 (64-bits).
- 16 cores lógicos (hyper-threading).
- 48 GB de RAM.
- 1TB de disco duro.
- Tarjeta de video NVIDIA Quadro RTX 8000 con 48GB RAM.

Figura 28. Configuración de base de datos de imágenes por *Yolo-Mark*.

Para el entrenamiento se consideraron 5 clases: carro, tráiler, autobús, minibús y motocicleta. En la Tabla 1 se puede ver el total de objetos configurados para cada clase para su entrenamiento.

Tabla 1. Relación de objetos utilizados para entrenamiento de la *CNN*.

Identificador	Clase	Entrenamiento		Pruebas		Total
		Imágenes	%	Imágenes	%	
0	Carro	4613	76	1475	24	6088
1	Tráiler	59	97	2	3	61
2	Autobús	165	53	147	47	312
3	Minibús	132	69	62	31	194
4	Motocicleta	212	67	105	33	317

Se realizó el entrenamiento con tres versiones diferentes de *YOLO*, con el objetivo de comparar cual versión presenta una mejor relación precisión-velocidad. Las versiones que se utilizaron son las siguientes: *YoloV3Tiny*, *YoloV3* y *YoloV4*. Los valores utilizados para cada versión implementada corresponden a 5 clases, un lote (*batch*) de 64 con 16 subdivisiones, tamaño de imagen de 416x416, y 10000 iteraciones ($max\_batches = 2000 \times N\_Clases$).

Como resultado del entrenamiento, y de acuerdo a la configuración mencionada anteriormente, el YOLO proporciona la información necesaria para realizar el análisis usando las curvas *ROC* (*Receiver Operating Characteristic Curve*). Las Figuras Figura 29, Figura 30 y Figura 31 muestran las gráficas del *Mean Average Precision* (mAP) para cada entrenamiento aplicado.

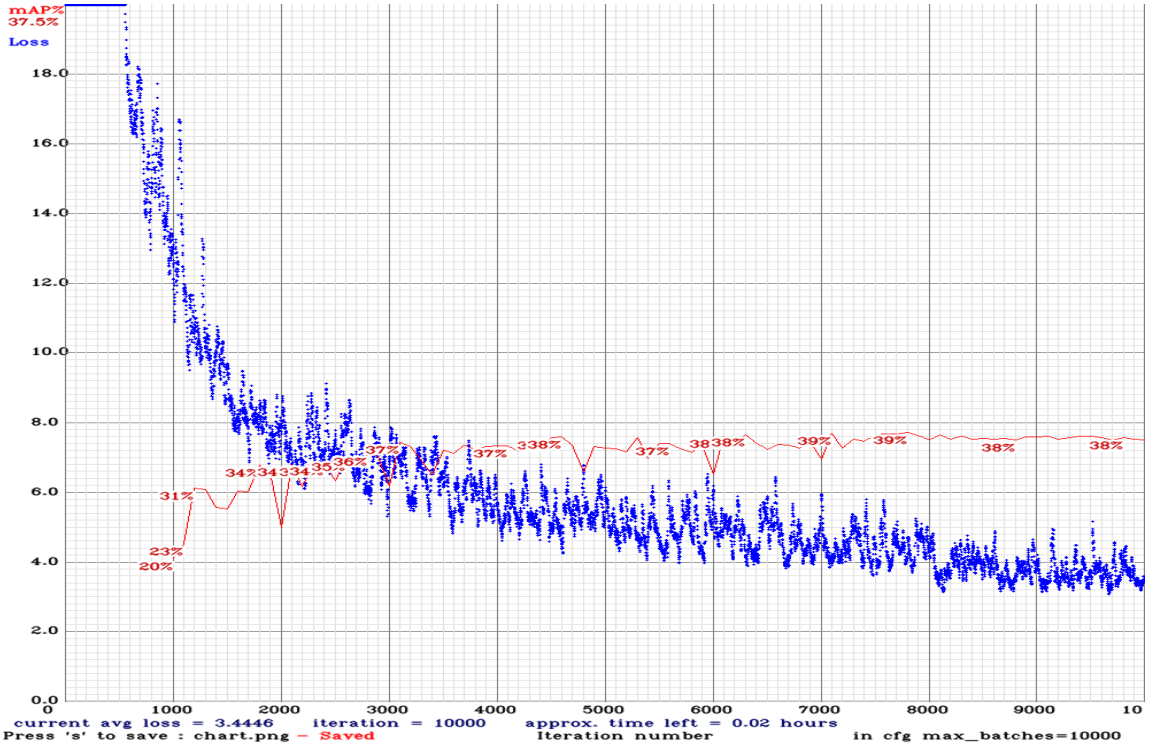


Figura 29. Gráficas de entrenamiento de la versión YoloTiny.

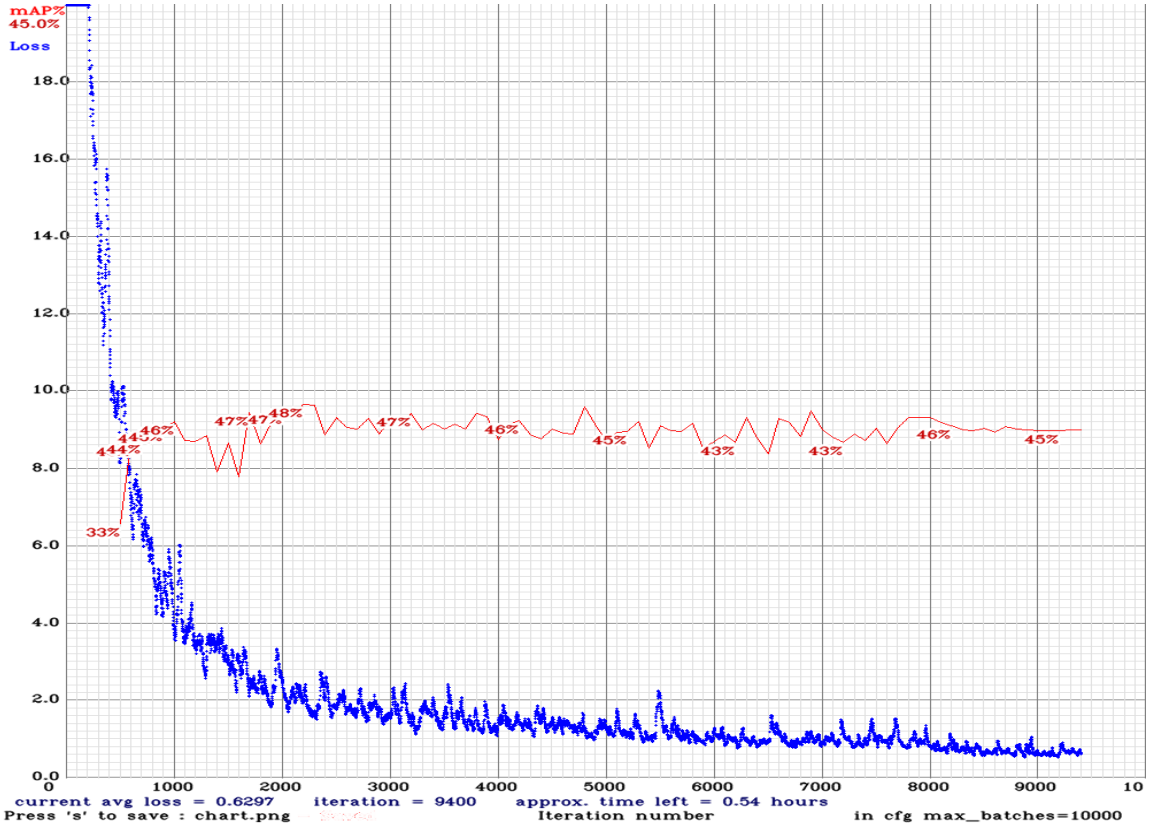


Figura 30. Gráficas de entrenamiento de la versión YoloV3.

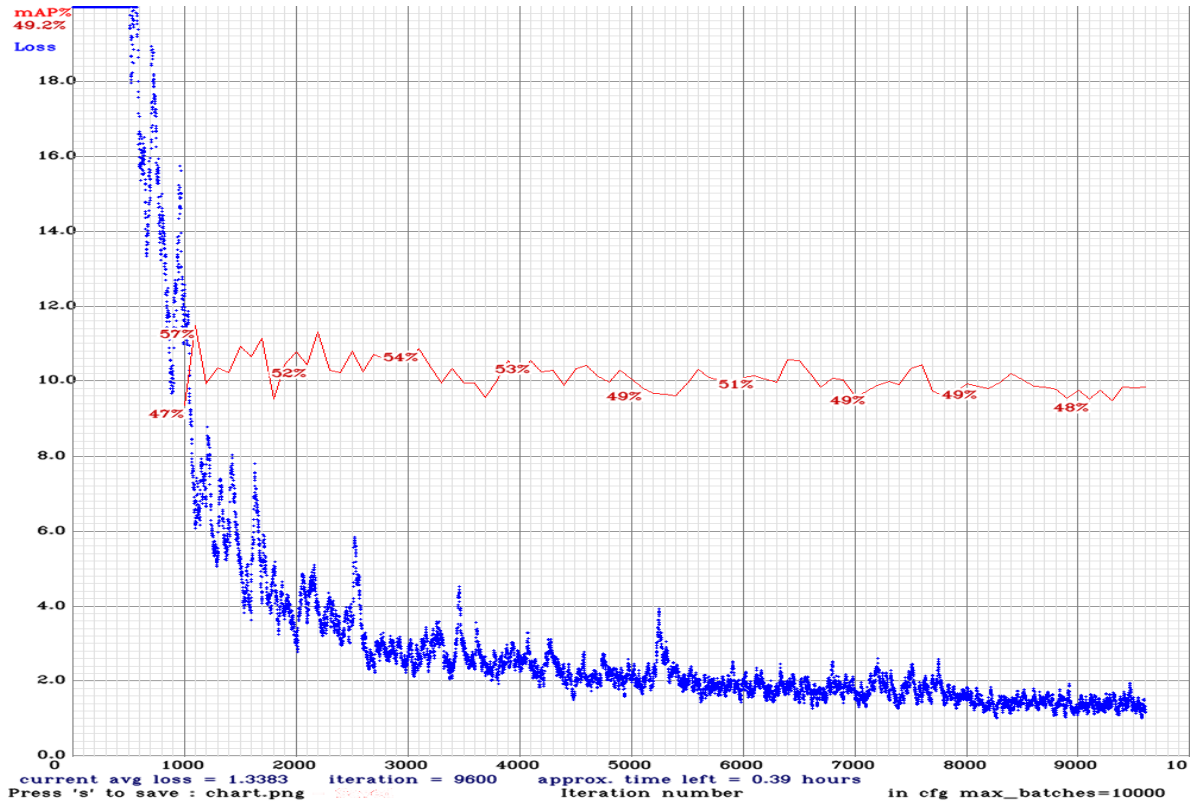


Figura 31. Gráficas de entrenamiento de la versión *YoloV4*.

En la Tabla 2 se presenta la información de la aplicación del reconocimiento, y su resultado en falsos positivos (*FP*, *False-Positive*), y verdaderos positivos (*TP, True-Positive*).

Tabla 2. Validaciones aplicadas a diferentes versiones de YOLO.

Versión	Detecciones	Carro		Tráiler		Autobús		Minibús		Motocicleta	
		TP	*FP*	*TP*	*FP*	*TP*	*FP*	*TP*	*FP*	*TP*	*FP*
YoloV3Tiny	*2889*	*1043*	*524*	*0*	*2*	*28*	*0*	*19*	*8*	*69*	*5*
YoloV3	*2595*	*1367*	*254*	*0*	*2*	*28*	*3*	*22*	*9*	*72*	*9*
YoloV4	*7058*	*1390*	*414*	*0*	*18*	*31*	*4*	*34*	*36*	*78*	*14*

Como se puede apreciar en la Figura 31, la versión *YoloV4* presenta un *mAP* mayor, sin embargo, en la Tabla 3 se observa que YoloV3 presenta mayor precisión en la detección de objetos, por lo cual los pesos derivados del entrenamiento en esta

versión se toman para la implementación del desarrollo del algoritmo del proyecto. La precisión se calcula de la siguiente manera:

$$Precision = \frac{TP}{(TP+FP)} . \quad (34)$$

Tabla 3. Precisión de las diferentes versiones de YOLO.

Versión	Detecciones	Totales		Precisión
		TP	*FP*	
YoloV3Tiny	*2889*	*1159*	*539*	*0.682567727*
YoloV3	*2595*	*1489*	*277*	*0.843148358*
YoloV4	*7058*	*1533*	*486*	*0.759286776*

3.2.3 Almacenamiento de historial

Para poder determinar la velocidad, densidad y volumen, es necesario saber dónde ha estado cada vehículo a lo largo del campo de visión del video; para esto se realiza el almacenamiento de su historial $\vec{H}$, para determinar la posición de cada vehículo en un *frame* k, y se calcula el flujo óptico mediante el método Lucas-Kanade. En la Figura 32 se puede observar la estructura establecida para el almacenamiento del historial.

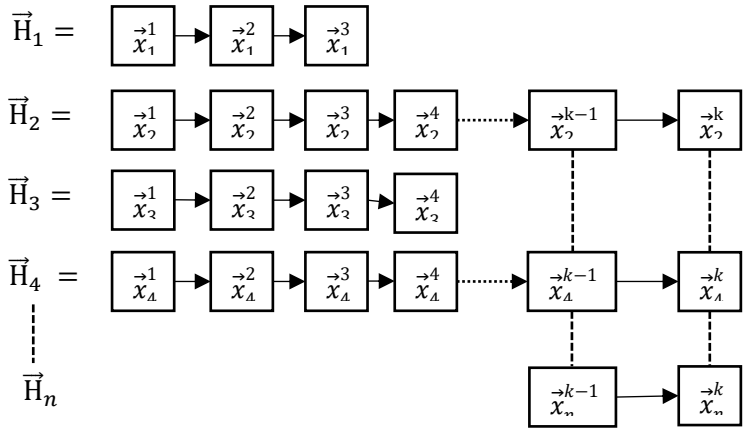


Figura 32. Estructura de almacenamiento de historial vehicular.

En el historial (ver Figura 32), $\vec{x}_n^k$ es el vector de posición a lo largo de la historia de un punto característico en la imagen k; para el caso del historial $\vec{x}_n^{k-1}$ se refiere al punto característico $\vec{x}_n$ en el frame $k-1$. Para poder considerar un punto característico debe cumplir un distancia mínima d_m con otro punto característico ya existente.

Para almacenar el historial se considera un listado de vectores de puntos, donde cada vector contiene la posición que va tomando cada punto de interés desde su primera aparición hasta su último movimiento (ver Figura 33).

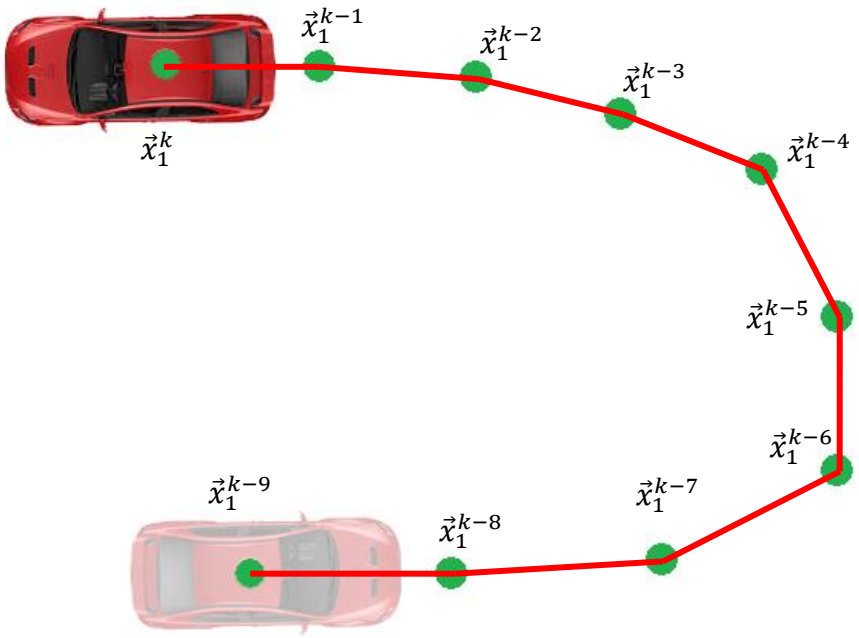


Figura 33. Historial de desplazamiento del punto característico $\vec{x}_1$ al *frame* k.

3.2.4 Filtrado e integración de clasificadores

Dado que el reconocimiento de vehículos consiste en la implementación de dos clasificadores (2.1.3.2 , 2.2.2), tomar ambos listados sin aplicar algún tipo de filtro puede generar que existan múltiples puntos asociados con un solo objeto; esto influye en el tiempo de procesamiento derivado de redundancia de información, y dando como resultado posibles errores de estimación global del movimiento. Para atacar este problema se implementa un procedimiento de filtrado para la integración de la información generada por ambos clasificadores con el cual la información es únicamente la necesaria para su evaluación. El filtrado está basado en la región que ocupa cada punto característico de acuerdo a la configuración de cada clasificador, las regiones para el método *Shi-Tomasi* están dadas por $d_m \times d_m$ donde d_m es la distancia mínima entre dos puntos característicos, mientras que el clasificador *YOLO*

proporciona la información de la región en la que se encuentra el vehículo. Teniendo la región perteneciente a cada punto característico, se sigue el siguiente proceso:

1. Se toma como base la información del *YOLO* para la creación de una máscara binaria de las regiones ocupadas.
2. Utilizando la máscara creada en el paso 1, se realiza la validación del listado de puntos característicos generados por el método *Shi-Tomasi*.
3. Se descartan aquellos que su región a ocupar toque alguna región ya registrada, y se regresa al paso 2.
4. Si la región no se sobrepone a otra existente en la máscara, se realiza el registro de la región en la máscara.

Se toma como base la información que proporciona el *YOLO* debido a que este clasificador brinda una información más completa de la región en la que aparece un vehículo en la escena. En la Figura 34 se puede observar un ejemplo de la aplicación del proceso de filtrado de la información, donde los puntos en rojo son los puntos obtenidos por el clasificador *Shi-Tomasi*, y los puntos verdes los obtenidos empleando el clasificador *YOLO*; se puede observar también que al hacer una integración directa, los puntos del primer método caen dentro de la región ya seleccionada para un vehículo.

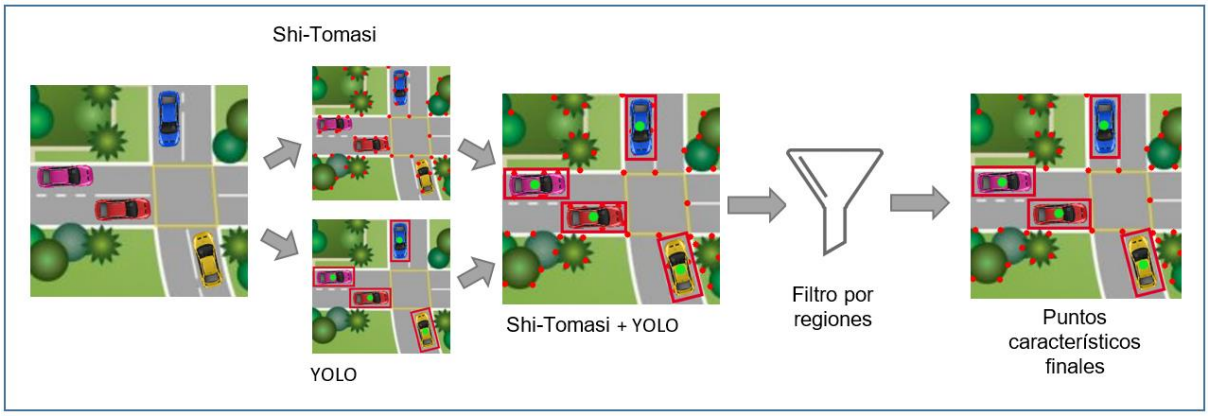


Figura 34. Proceso de filtrado de puntos característicos.

3.3 Estimación de flujo vehicular

Para poder estimar el flujo vehicular se utiliza la información del historial de cada vehículo y punto característico detectado en el paso 1. Para calcular la velocidad promedio $\bar{M}$, primero se deben clasificar los tipos de movimiento en dos clases, movimientos pequeños $\vec{\mu}_{sm}$, el cual es considerado como los movimientos generados por la cámara, y los movimientos grandes $\vec{\mu}_{hm}$, para los movimientos que se consideran que son generados por los vehículos, para determinar el punto inicial $\vec{x}_{t-F_d}$ y punto final $\vec{x}_t$, donde t es un *frame* tomado en un tiempo determinado, y F_d es la distancia en *frames* con el *frame* al tiempo t al comparar la posición (ver Figura 35).

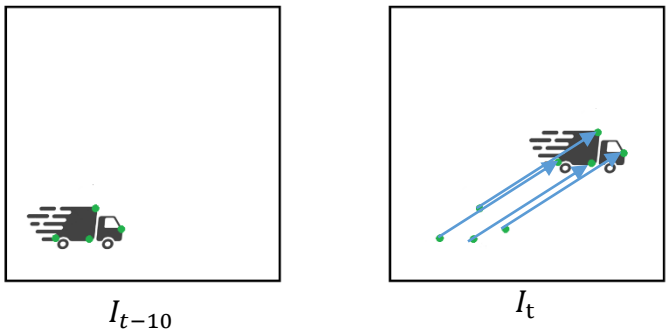
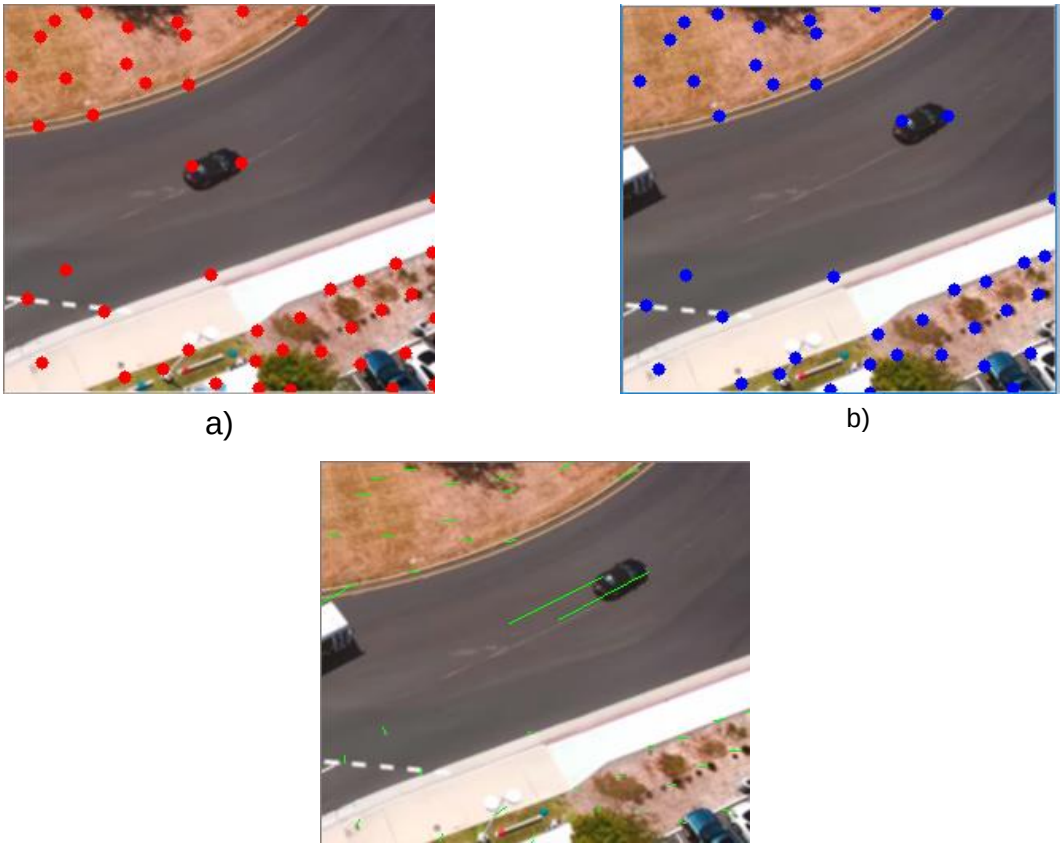


Figura 35. Representación del flujo óptico entre el frame t y el frame $t - F_d$ donde $F_d = 9$.

3.3.1 Detección de movimiento

Para determinar el movimiento de los objetos de una imagen se considera el punto central de dicho objeto; estos puntos previamente definidos utilizando los métodos de detección de objetos mencionados anteriormente. La aplicación de la estimación del flujo óptico es realizada mediante la función ***calcOpticalFlowPyrLK*** de la librería OpenCV. La Figura 36 muestra el resultado de aplicar esta función sobre 2 imágenes diferentes.



a) b)

c)

Figura 36. Aplicación de flujo óptico y vista de desplazamiento. a) Puntos característicos I_1. b) Puntos correspondientes de I_1en I_2 usando flujo óptico. c) Desplazamiento de I_1a I_2.

3.3.2 Estabilización

Debido a que la cámara se encuentra adherida a un *UAV*, puede contar con inestabilidad al momento de adquirir los videos que se analizarán, dando como resultado un análisis y resultados con un margen de error más alto. A continuación, se describen los pasos realizados para mitigar esta problemática aplicando la estabilización de la imagen:

1. Se establece la I_1 como la Imagen base I_{base}.
2. Se calculan los puntos característicos P_1 de la imagen actual I_1 y se asignan a los puntos base P_{base}, los cuales servirán para la proyección (ver Figura 37).

I_1

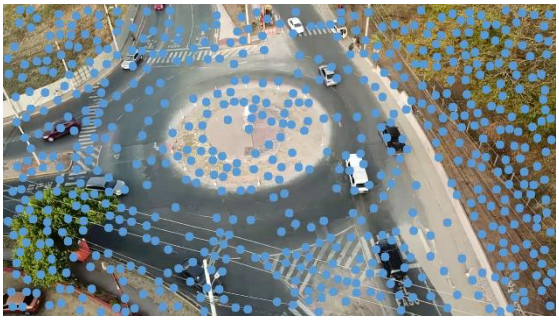


Figura 37. Cálculo de puntos característicos.

3. Se lee la siguiente imagen y se almacena en I_{k+1}.
4. Se calcula el flujo óptico de los puntos P_k de la imagen I_k y se almacenan en P_{k+1} (ver Figura 38).

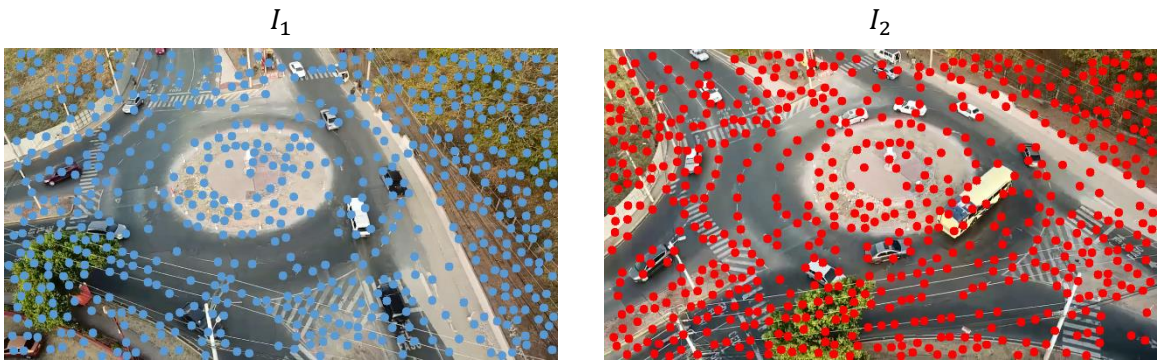


Figura 38. Cálculo de flujo óptico entre las imágenes I_1 y I_2, usando los puntos característicos de la imagen I_1.

5. Se calcula la homografía $H_{k,base}$ de P_{k+1} a P_{base} (ver Figura 39).

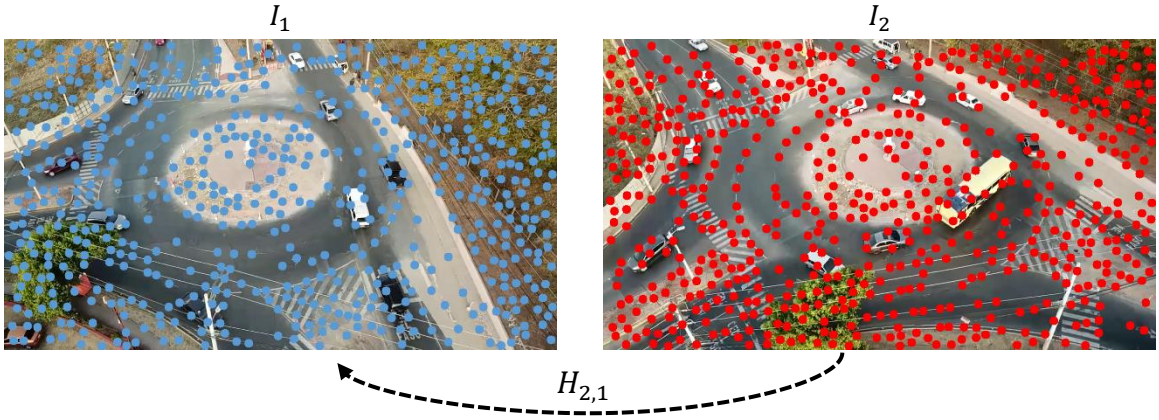


Figura 39. Cálculo de la homografía para la proyección.

6. Se calculan los puntos característicos en la imagen I_{k+1}, y se agregan los que no existen en P_{k+1} (ver Figura 40).

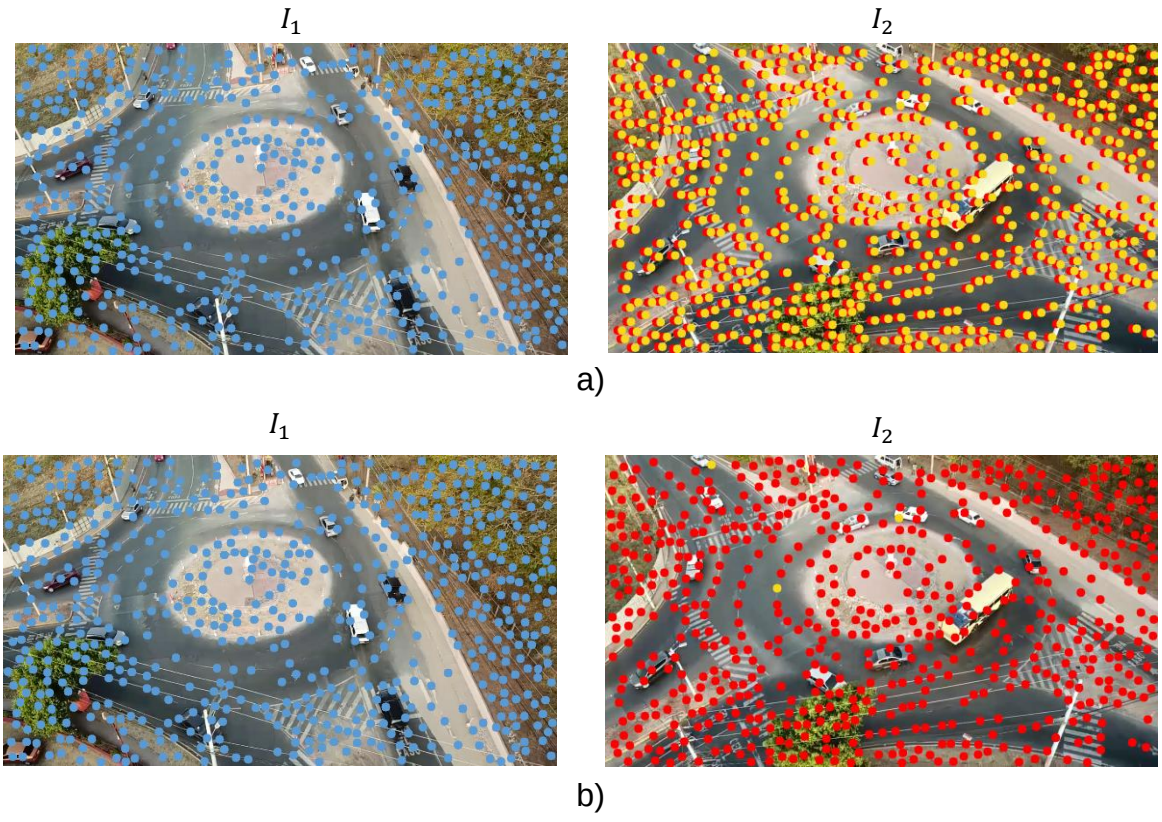


Figura 40. a) Comparación de puntos característicos base y puntos caracteristicos nuevos estimados. b) Filtrado de puntos caracteristicos nuevos.

7. Se aplica la homografía $H_{k+1,base}$ a los puntos P_{k+1}, y se asignan a P_{base}(ver la Figura 41).

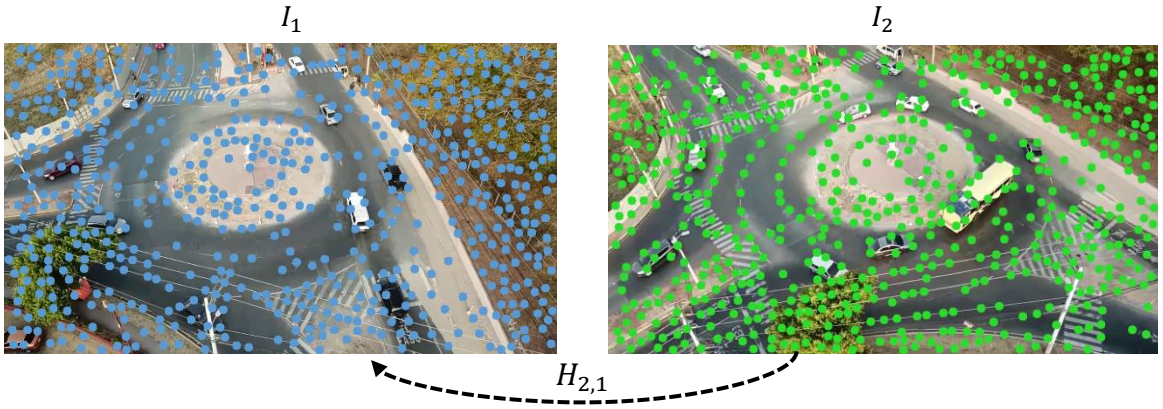


Figura 41. Proyección de puntos característicos de la imagen I_2 a la imagen base I_1.

8. Repetir desde el paso 3 (ver Figura 42).

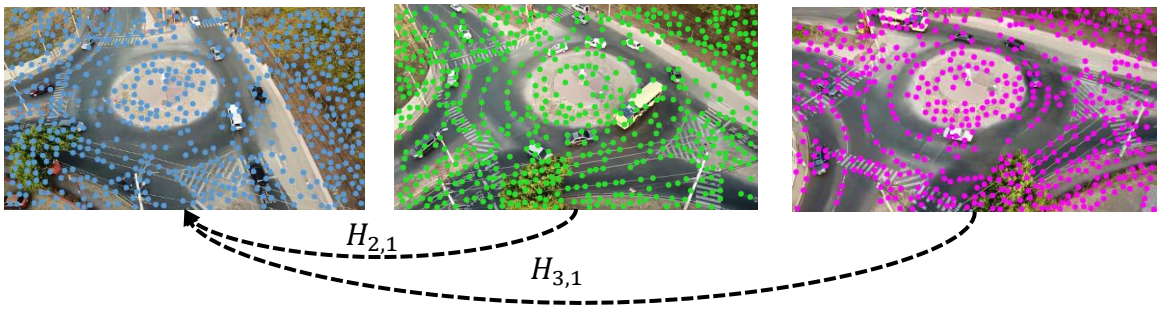


Figura 42. Representación de la siguiente iteración.

3.3.3 Desplazamiento vehicular

Para determinar los movimientos o desplazamientos μ de cada vehículo o punto característico, se utilizan los puntos ya estabilizados por medio de la aplicación de la homografía; utilizando la norma L_2, se determina el movimiento de un punto característico a partir del tiempo $t - F_d$ al tiempo t, tomando la información almacenada en su historial:

$$m(i) = \left\| \vec{x}_t^i - \vec{x}_{t-F_d}^i \right\|_2 \tag{35}$$

Dichos movimientos se clasifican en dos conjuntos aplicando un umbral λ, el conjunto de movimientos pequeños, probablemente ocasionados por la cámara $m_p = \{m(i): m(i) < \lambda\}$ y el conjunto de movimientos grandes, que probablemente contienen los movimientos de los vehículos $m_g = \{m(i): m(i) < \lambda\}$. Una vez determinados los elementos que pertenecen a cada clase, se calcula el promedio de desplazamientos para cada una.

$$\mu_p = \frac{1}{N_p} \sum_{i=1}^{N_p} m_p(i)\,, \qquad \mu_g = \frac{1}{N_g} \sum_{i=1}^{N_g} m_g(i) \tag{36}$$

Donde N_p y N_g están definidos por el total de elementos en cada clase. Una vez calculados μ_p y μ_g, se procede a calcular el promedio general de desplazamiento de todos los vehículos; esto se obtiene realizando el cálculo de la diferencia entre los desplazamientos de las clases

$$\mu_v = \mu_g - \mu_p. \tag{37}$$

Para calcular la velocidad se utiliza la siguiente formula

$$\bar{V} = \frac{1}{F_d} \cdot \mu_v \cdot F \cdot r \qquad \text{donde } r = \frac{L_1}{L_2} \tag{38}$$

F: Velocidad de fotograma (*frames*/seg).

L_1: Tamaño real de objeto que aparece en video (m).

L_2: Tamaño de pixeles del objeto L_1(pixel).

F_d: Distancia entre *frames.*

3.4 Resultados experimentales

3.4.1 Detección de vehículos

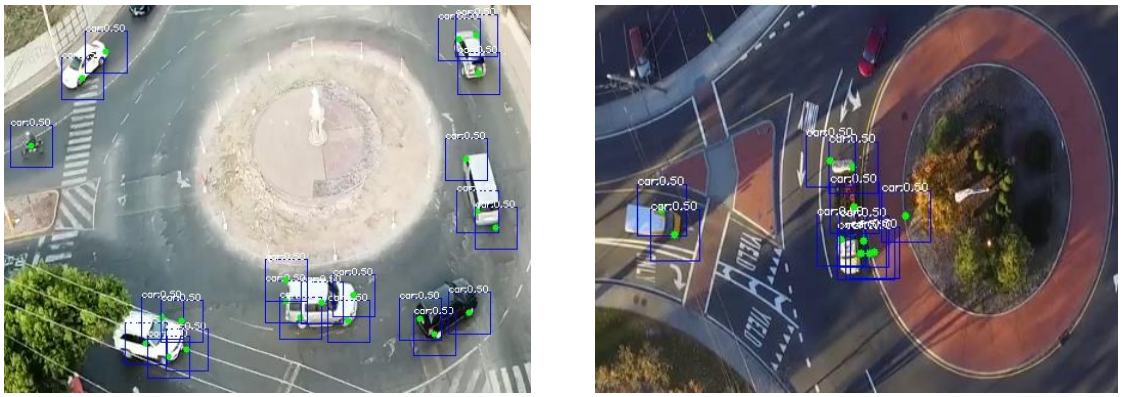
Como se mencionó anteriormente, la detección de vehículos se encuentra dividida en dos partes: la implementación del método *Shi-Tomasi*, y la detección por medio de la red neuronal implementando *YOLO*.

3.4.1.1 Prueba 1

Para la detección de vehículos mediante el método *Shi-Tomasi* es necesario contar con dos *frames* consecutivos; la razón reside en la necesidad de identificar objetos en movimiento. Las pruebas de detección de vehículos se realizaron aplicando la librería OpenCV, utilizando principalmente las funciones siguientes:

- *cv2::goodFeaturesToTrack*: Esta función permite establecer los puntos característicos del *frame*.
- *cv2::calcOpticalFlowPyrLK*: Función con la cual se calcula el flujo óptico de *frame* a *frame* a un determinado número de puntos característicos.

Se requiere una distancia mínima entre puntos característicos como se menciona en la sección 3.2.1, de manera que utilizando esta distancia mínima, se construye una ventana para denotar la región en la que considera que existe un vehículo considerando el punto característico como el centro de esta región. Se realizó el desarrollo del algoritmo y las pruebas de detección de vehículos.



a) b)

Figura 43. Detección de vehículos por el método *Shi-Tomasi*.

Como se puede apreciar en la Figura 43, se consideran como puntos característicos pertenecientes a un vehículo, aquellos que se clasifican como movimientos grandes; como se puede observar, un vehículo puede contar con uno o más puntos característicos como resultado de la búsqueda de esquinas, dado que este método se utiliza para la detección de movimiento en la escena; se establece un nivel

de confianza de 50%, indicando que el punto característico es de la clase carro, como valor de clase predeterminado para la implementación de este método.

3.4.1.2 Prueba 2

En la segunda etapa de las pruebas, se aplica el *YOLO* para la detección de los vehículos utilizando los pesos obtenidos del entrenamiento, desarrollando un algoritmo para su implementación en una serie de imágenes. Para ello, se utilizaron las librerías mencionadas anteriormente en la sección 3.2.2, cuyos resultados son presentados en la Figura 44. Considerando que para el entrenamiento de la red neuronal se utilizaron imágenes capturadas de manera perpendicular al suelo (vista aérea), en la Figura 44a se puede observar que la red neuronal detecta objetos incluso con cierto grado de inclinación; en la Figura 44b se puede apreciar que la red neuronal realiza la detección de vehículos que presentan algún tipo de oclusión; esto es importante, ya que en la base de datos de imágenes para el entrenamiento no fueron utilizadas imágenes que tuvieran alguna oclusión a la forma del vehículo. Por otro lado, en la Figura 44c, se puede apreciar en la parte inferior izquierda que un peatón es detectado como motociclista; la red neuronal detecta al peatón estableciendo un 94% de probabilidad de ser una motocicleta; es importante mencionar que la clase referente a motociclista se entrenó con un número pequeño de objetos.

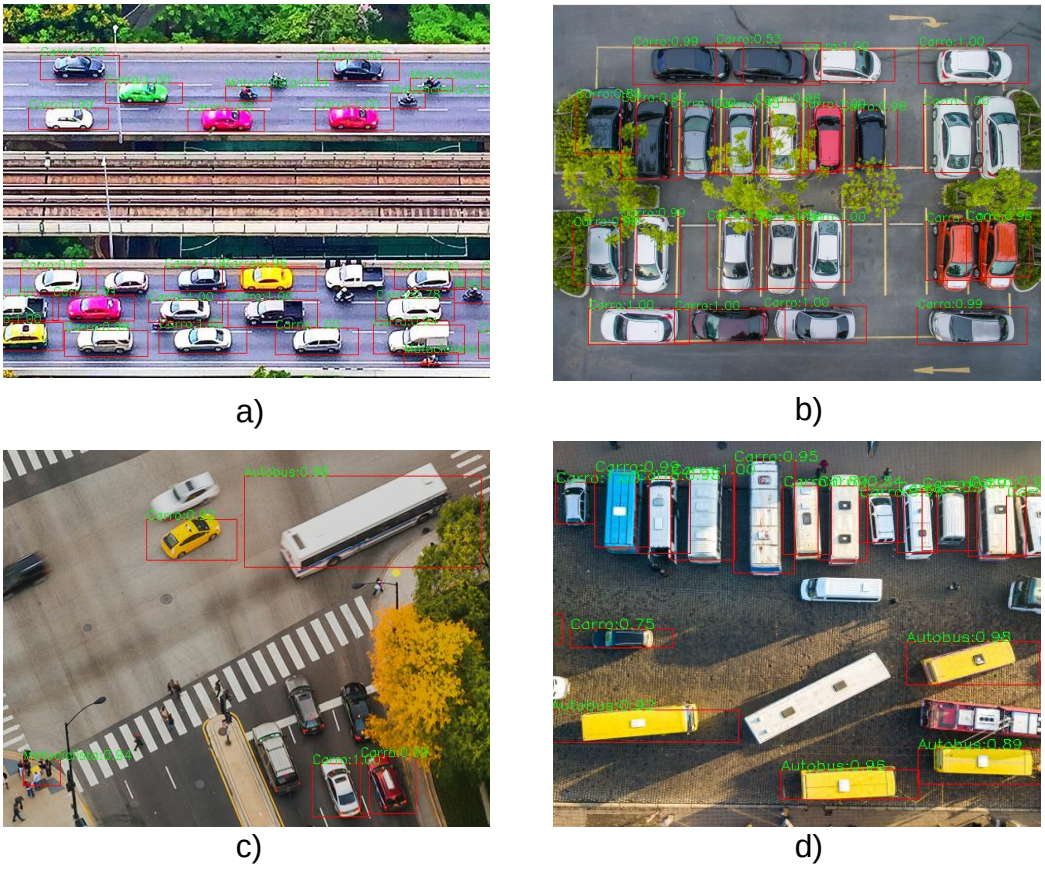


Figura 44. Aplicación de pesos candidatos de YOLO para su implementación en el proyecto.

3.4.1.3 Prueba 3

Se realiza la integración de los resultados de las pruebas 1 y 2 aplicando el filtrado (véase sección 3.2.4) de puntos característicos; para ello se creó una función que recibe los puntos característicos de ambos clasificadores y con sus respectivas regiones. En la Figura 45 se observa la aplicación del filtro sobre una imagen entrada; se puede observar la distribución de los puntos calculados por ambos métodos de manera individual; para poder integrar estos puntos en un solo listado, se crea una máscara binaria a partir del listado generado por el método YOLO. Lo siguiente es realizar la comparación punto por punto del listado *Shi-Tomasi*; tomando las

coordenadas del punto característico, se realiza la evaluación de esas mismas coordenadas en la máscara, y en caso de existir un valor de 1, el punto es descartado; en caso contrario, se marca en la máscara la región reservada para el punto con valores de 1, de tal modo que los siguientes puntos a evaluar no puedan ocupar la misma región. Contando con ambos listados ya filtrados, se procede a la unificación de ellos.

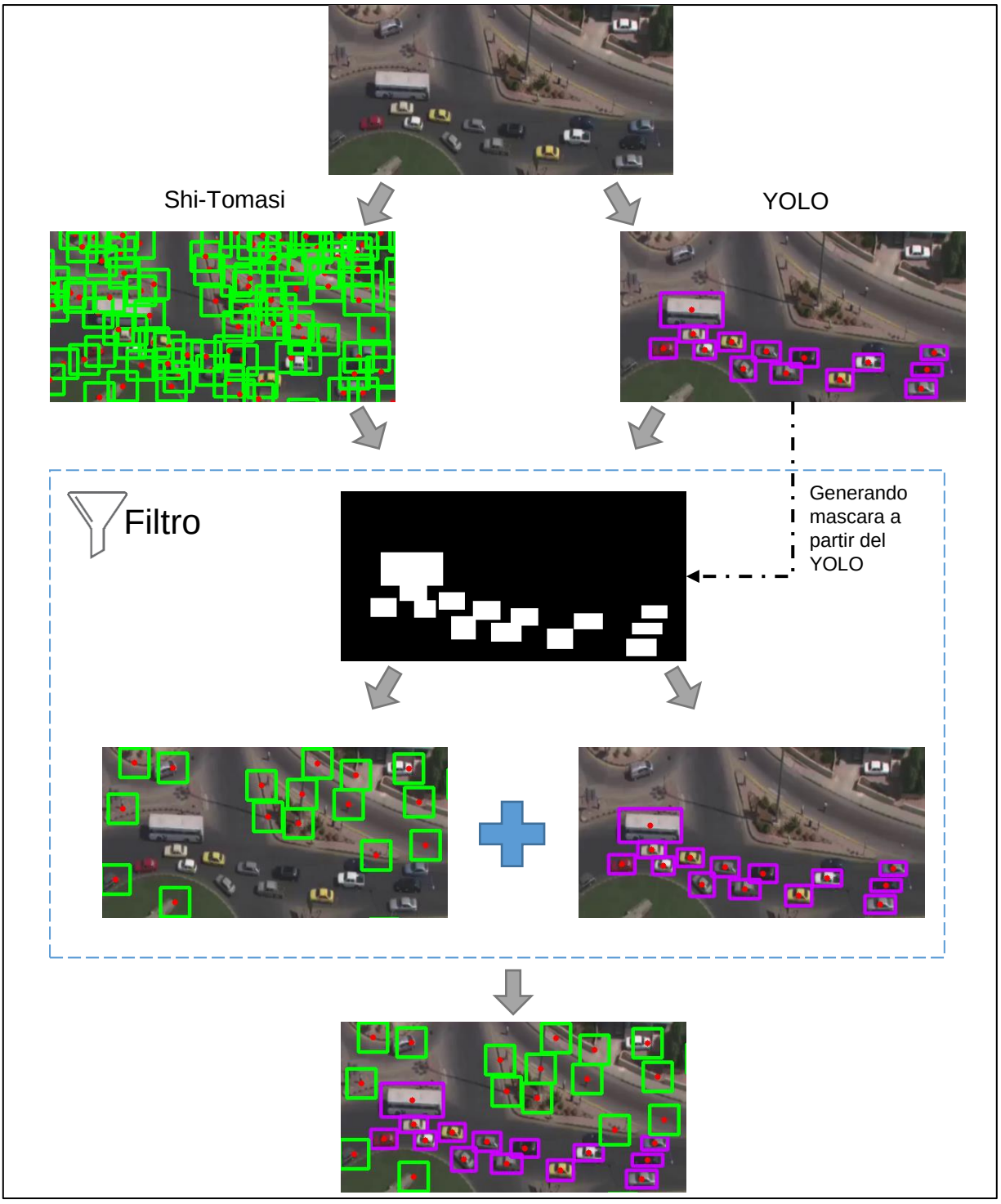


Figura 45. Pruebas de filtrado de puntos característicos.

3.4.2 Estabilización de video

Para la implementación de la estabilización, se realizó un algoritmo con base en el proceso presentado en la sección 3.3.2, y se codificó en lenguaje C++ por medio de Visual Studio 2017. Para su realización se implementó la librería OpenCV, utilizando principalmente las funciones siguientes:

- *cv2::goodFeaturesToTrack*: Esta función permite establecer los puntos característicos del *frame* para su estimación del flujo óptico.
- *cv2::calcOpticalFlowPyrLK*: Función con la cual se calcula el flujo óptico entre dos *frames* consecutivos.
- *cv2::findHomography*: Función para calcular la homografía a partir de los puntos característicos de un frame con los puntos característicos del frame consecutivo de acuerdo al flujo óptico.
- *cv2::warpPerspective*: Esta función permite realizar la proyección por medio de la homografía del *frame* actual hacia el *frame* base.

Para probar el algoritmo desarrollado, se tomaron dos videos que presentan inestabilidad o movimientos de la cámara. En el segundo video, se muestra el escenario de una intersección, y cuenta con una glorieta sobre la cual el *UAV* captura el video realizando un movimiento de rotación.

3.4.2.1 Prueba 1

El primer video muestra la aplicación de la estabilización sobre un video con un escenario con vista aérea; este video presenta ligeros movimientos rotatorios y de

traslación. En la Figura 46 se puede observar la captura de la aplicación del algoritmo para los *frames* 20, 240 y 1400.

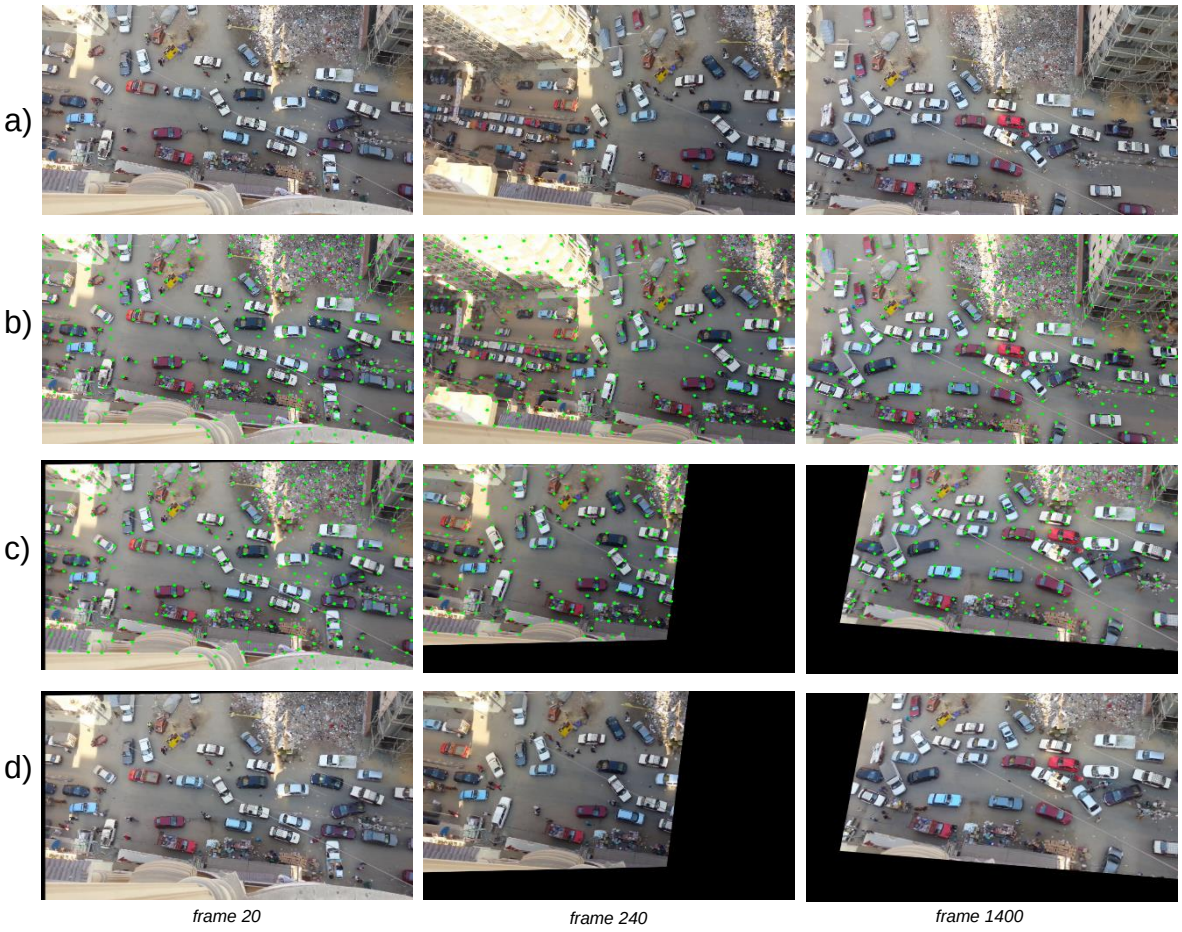


Figura 46. Aplicación de estabilización del video 1. a) Frames del video original. b) Puntos característicos para estabilización en video original. c) Puntos característicos proyectados mediante la homografia. d) Imagen estabilizada resultante.

En las pruebas realizadas se puede apreciar la existencia de múltiples planos visibles en la escena, y debido a que la homografía trabaja sobre un plano, la implementación de la estabilización para este video presenta pequeños errores de desplazamientos, como se puede apreciar en la Figura 47. Existen trabajos desarrollados para abordar el problema de la existencia de múltiples planos [29], sin embargo, no es el propósito de este trabajo de tesis. Los videos analizados para la

implementación del algoritmo observan un único plano, el suelo, por lo que la implementación de la homografía en el proyecto es factible.

Figura 47. Error de proyección para imágenes con multi-planos.

3.4.2.2 Prueba 2

Considerando que la aplicación de la estabilización se realiza sobre un plano, la siguiente prueba es aplicada sobre un video que toma un escenario con vista aérea. El video utilizado para esta prueba no solo presenta inestabilidad, sino que también incluye rotación sobre la escena y cambios de zoom intermitentes. En la Figura 48 se puede apreciar la implementación de la estabilización, presentando un escenario que contiene características aproximadas al escenario deseado para la implementación del trabajo final.

Figura 48. Aplicación de estabilización del video “glorieta_hacienda.mp4”. a) *Frames* del video original. b) Puntos característicos para estabilización en video original. c) Imagen estabilizada resultante.

El algoritmo de estabilización presentado proporciona de manera cualitativa resultados satisfactorios para su integración al algoritmo final del proyecto. Implementar este algoritmo ayudará a reducir el margen de error para la estimación del desplazamiento y velocidad de los vehículos.

3.4.3 Panorama

Analizando los resultados de la estabilización, se puede apreciar que existen sectores que se pierden en la escena que se está observando, por lo que tomando el trabajo realizado en *AVScreen* para mantener la visualización de un escenario completo, incentiva la implementación de la construcción del panorama visualizado a lo largo del video para cubrir las áreas que se han visto y debido a la instabilidad de la cámara se hayan perdido.

3.4.3.1 Prueba 1

Tomando nuevamente el video utilizado para la estabilización, se realiza la implementación de la construcción del panorama, el cual se puede apreciar en la Figura 49. Para la prueba se estableció que el segundo *frame* es tomado como *frame* de referencia.

Figura 49. Video 1, aplicación de construcción de panorama. a) *Frames* del video original. b) Estabilización del video original. c) Región que ocupa la imagen estabilizada en el panorama. d) Imagen resultante o panorama.

3.4.3.1 Prueba 2

Tomando el video 2 utilizado para las pruebas de estabilización, se aplicó la implementación de la construcción del panorama; dado que éste cuenta con las características aproximadas a las requeridas para la implementación de este trabajo de tesis, el resultado de la construcción del panorama presentado en la Figura 50 ofrece una referencia para los resultados esperados.

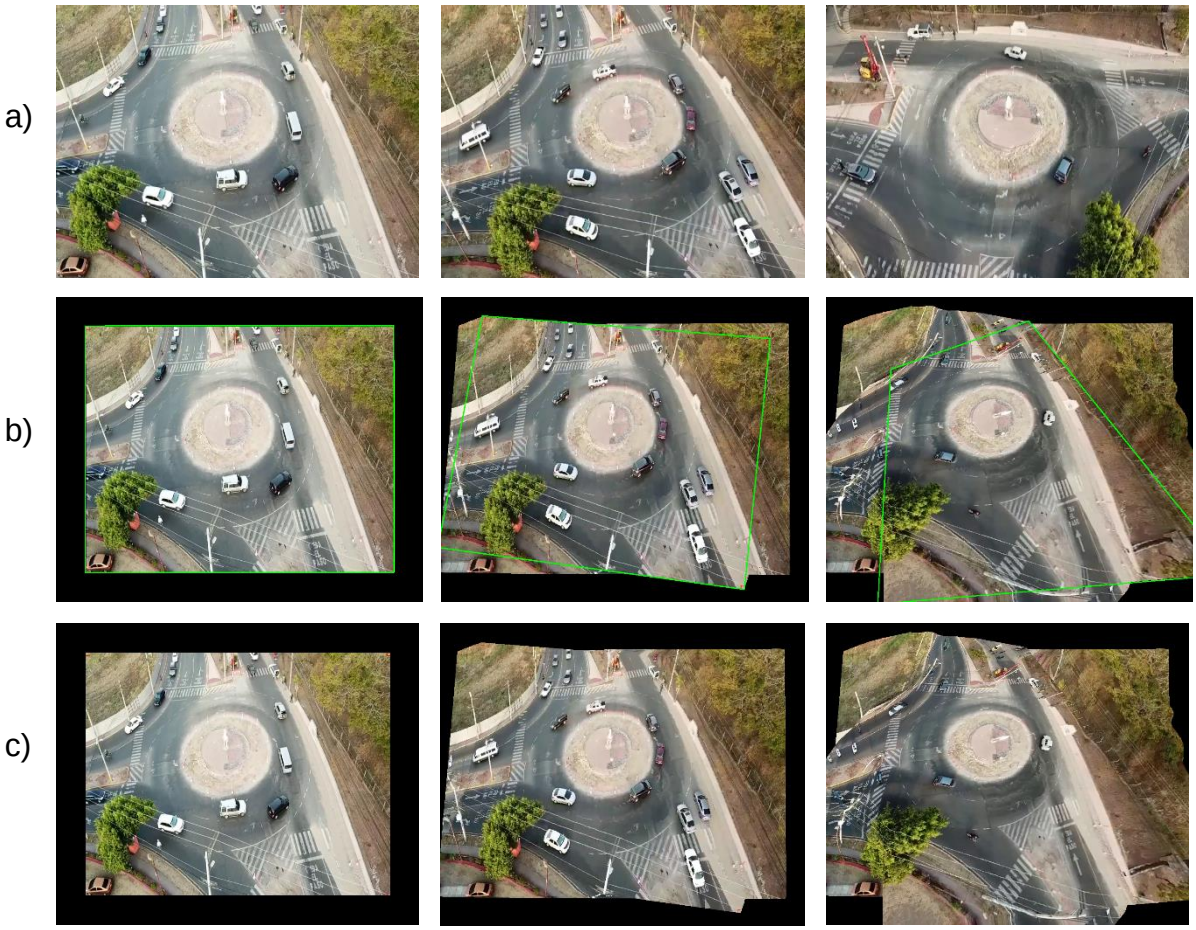


Figura 50. Video 2, aplicación de construcción de panorama. a) Frames del video original. b) Región que ocupa la imagen estabilizada en el panorama. c) Imagen resultante o panorama.

3.4.4 Desplazamiento

El cálculo del desplazamiento requiere la implementación de lo visto en las secciones 3.2.3 y 3.3.3; para ello se desarrolló el algoritmo y se realizaron las pruebas en dos etapas: a) almacenamiento de historial, b) estimación de desplazamiento.

3.4.4.1 Prueba 1

La primera prueba se realizó utilizando los puntos característicos proporcionados por la implementación del método *Shi-Tomasi*; para ello se considera cada punto como un vehículo a analizar. La Figura 51 muestra el historial de la posición de los puntos característicos en 30 *frames* consecutivos.

Figura 51. Visualización del historial de los puntos característicos.

3.4.4.1 Prueba 2

La segunda etapa de pruebas, consistió en la estimación del desplazamiento de los vehículos para su clasificación en movimientos grandes (m_g) y movimientos pequeños (m_p). En la Figura 52 se puede apreciar la aplicación de la clasificación de los puntos de acuerdo a su desplazamiento y el umbral establecido inicialmente.

Figura 52. Clasificación de desplazamientos en movimientos grandes (azul) y movimientos pequeños (verde).

3.4.5 Estimación de velocidad

Contando con los resultados de las pruebas de desplazamiento, se realizaron las pruebas de estimación de velocidad utilizando la ecuación (38). La implementación se realizó agregando la funcionalidad de cálculo de velocidad al algoritmo utilizado para la estimación del desplazamiento.

3.4.5.1 Prueba 1

Se realiza la aplicación del algoritmo de estimación de velocidad utilizando los puntos característicos obtenidos mediante el método Shi-Tomasi, el video utilizado presenta un movimiento circular en la escena, sin embargo, no se realiza la implementación de la estabilización de video, por lo cual se utiliza la ecuación (37) como método de compensación de movimiento y estimación de la velocidad compensada. Para la aplicación de la prueba se establecen los siguientes parámetros:

- *Tamaño del video:* 960×540.
- $L_1 = 5m$ *y* $L_2 = 46\ pixeles.$
- $F_d = 30.$

Figura 53. Muestra la aplicación de la estimación de velocidad, tomando los puntos característicos calculados mediante el método Shi-Tomasi al *frame* 172.

Como se puede observar en la Figura 53, debido a que la imagen presenta movimiento circular (Video no estabilizado) el número de objetos detectados es alto. Tomando las velocidades calculadas para cada punto característico, se realiza el cálculo de la velocidad global del escenario dando como resultado una velocidad global de 36.95 y una velocidad global compensada de 31.72 Km/h (ver Figura 54).

Figura 54. Cálculo de velocidad global al *frame* 110 que presenta mayor estabilidad en la captura.

Tomando los resultados de esta prueba y viendo la diferencia entre velocidades para el mismo tipo de clase, se hace la revisión del algoritmo detectando cálculos incorrectos para puntos con historial menor a F_d.

3.4.5.2 Prueba 2

La siguiente prueba se realizó tomando un video que no se ha utilizado para el entrenamiento de la *CNN YOLO,* con la vista aérea perpendicular al suelo. Para la aplicación de la prueba se establecen los siguientes parámetros:

- *Tamaño del video:* 960×540.
- $L_1 = 5m$ *y* $L_2 = 42\ pixeles.$
- $F_d = 30.$

La Figura 55 muestra la aplicación de la estimación de velocidad, tomando los puntos característicos calculados mediante los métodos *Shi-Tomasi* y *YOLO*.

a)

b)

c)

Figura 55. Cálculo de velocidad implementando ambos métodos de clasificación. a) Vista de la estimación de cálculo de velocidades. b) Zona 1: Vehículos calculados por el método Shi-Tomasi. c) Zona 2: vehículos estimados por YOLO.

3.4.5.3Prueba 3

La siguiente prueba se realiza aplicando los métodos de detección Shi-tomasi y la *CNN* YOLO aplicando adicionalmente la metodología de estabilización. En la Figura 56 se puede apreciar como la escena se visualiza de manera inclinada como resultado de la estabilización del video. Para la aplicación de la prueba se establecen los siguientes parámetros:

- *Tamaño del video:* 960×540.
- $L_1 = 5m$ *y* $L_2 = 42\ pixeles.$
- $F_d = 30.$

Figura 56. Frame 172. Calculo de velocidades implementando estabilización, Shi-Tomasi y YOLO.

3.4.5.4 Prueba 4

Para la siguiente prueba se aplica toda la metodología desarrollada para este trabajo de tesis, la cual consiste en la clasificación, estabilización y construcción de panorama para la estimación del flujo vehicular. La Figura 57 muestra el *frame 172* y su vista.

Figura 57. *Frame* 172, cálculo de velocidades implementando clasificación, estabilización y panorama.

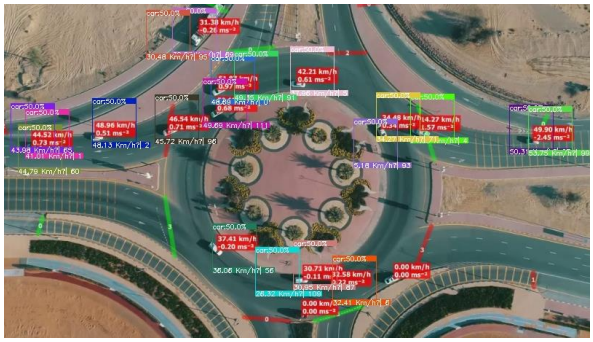
3.4.6 Resultados del método

A continuación, se presentan los resultados obtenidos de la aplicación de la metodología sobre el escenario (glorieta). Para la validación del algoritmo se realizó una comparación entre la información presentada por la empresa *RCE Systems* [30] para la promoción de su sistema de monitoreo de tráfico *Data From Sky*, y la información generada por el trabajo realizado para este trabajo de tesis. La Figura 58 muestra un ejemplo de uno de los *frames* evaluados.

Figura 58. Frame 230 del video tomado para validación de algoritmo. Tomado de [26].

Se tomaron 11 vehículos (Figura 59a) y se realizó la validación manual (ver Tabla 4) de las velocidades. Se toma como punto de partida el *frame* 255 tomando la información calculada cada 5 *frames* hasta tener un total de 24 valores para cada vehículo; esta información es comparada con el valor real tomada de la información que presenta el video.

a)



b)

c)

Figura 59. Selección de vehículos para validación. a) Vehículos seleccionados. b) Cálculo de velocidades. c) Visualización del cálculo de velocidad de los vehículos 2 y 96.

Tabla 4. Tabla comparativa entre la Información de las velocidades calculadas y reales.

Km/h		Carro 0	Carro 1	Carro 2	Carro 4	Carro 5	Carro 6	Carro 56	Carro 71	Carro 87	Carro 96	Carro 99
255	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
260	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
265	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
270	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
275	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
280	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
285	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
290	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17

295	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
300	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
305	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
310	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
315	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
320	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
325	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
330	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
335	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
340	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
345	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
350	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
355	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
360	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
365	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17
370	Calculada	44.9711	42.053	44.8467	5.47672	41.4012	33.8424	31.2185	34.8463	35.2479	41.7169	62.42
	Real	45.73	42.95	46.27	5.35	40.74	33.02	35.85	33.42	34.22	42.23	62.17

La Figura 60 presenta las gráficas comparativas de los valores presentados en la Tabla 4.

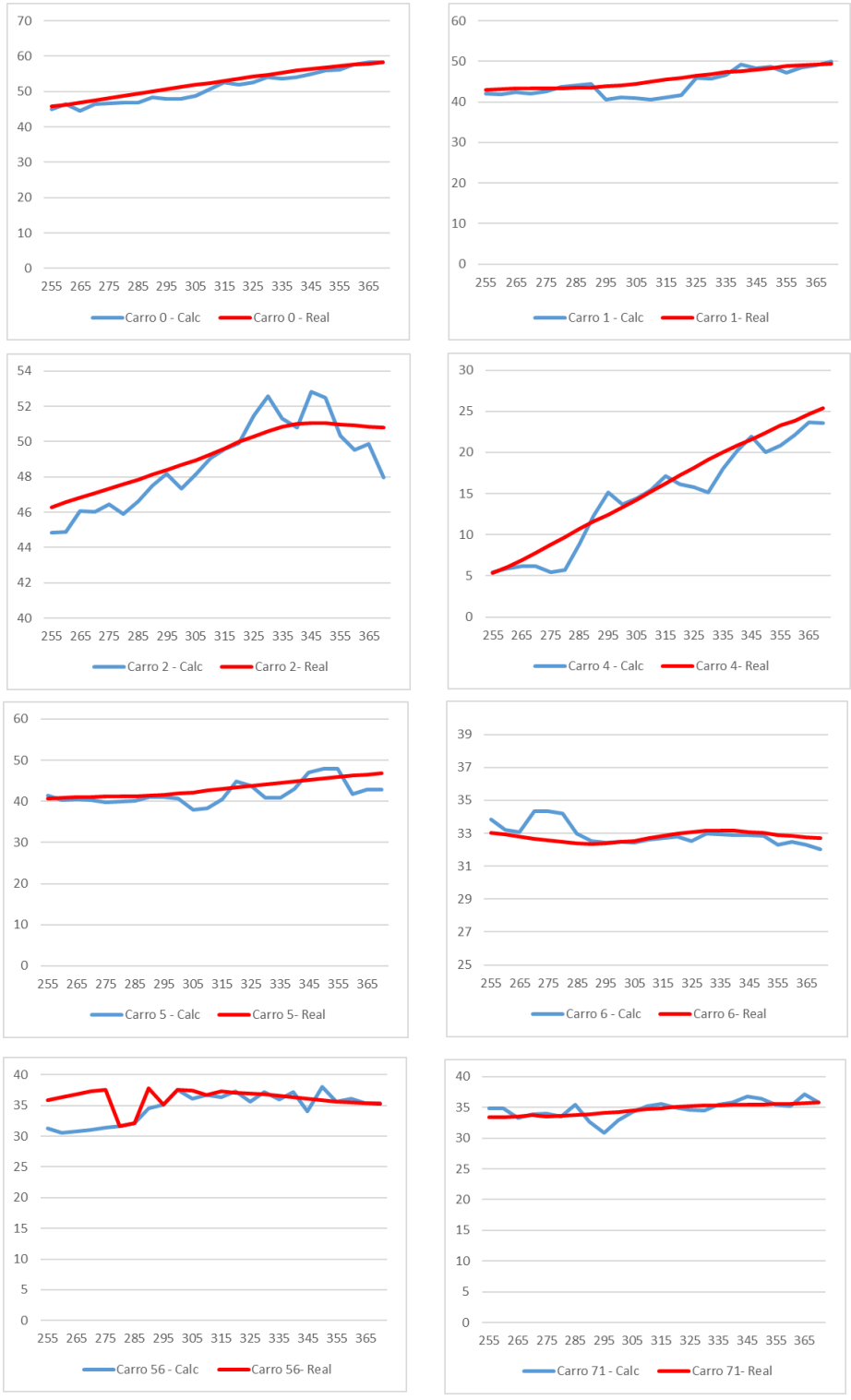


Figura 60. Comparación de resultados vehículos: 0, 1, 2, 4, 5, 6, 56 y 71.

Dada la información de la Tabla 4, se puede calcular el error cuadrático medio, la varianza σ^2, y la desviación estándar σ (ver Tabla 5).

Tabla 5. Error cuadrático medio, varianza σ^2 y desviación estándar σ.

	Error cuadrático medio	σ^2	σ
Carro 0	3.000121975	1.035718534	±1.017702576
Carro 1	4.294152973	3.047468701	±1.745700061
Carro 2	1.53666542	1.372395991	±1.17149306
Carro 4	3.65099618	2.642152981	±1.62547008
Carro 5	5.955570297	4.389285632	±2.095062202
Carro 6	0.499719141	0.504420873	±0.710225931
Carro 56	8.13229877	6.340563076	±2.518047473
Carro 71	1.153640353	1.193956066	±1.092682967
Carro 87	1.672094234	1.739951185	±1.319072092
Carro 96	1.161262363	0.452537527	±0.672709095
Carro 99	2.066490967	1.223532765	±1.106134153

Como se puede observar en la Tabla 5, el carro 6 presenta el mejor resultado; por otro lado, el carro 56 presenta el mayor error y variación al momento de calcular su velocidad. El cálculo de las velocidades de los vehículos tiene una varianza promedio de 2.18, con una desviación estándar promedio de ±1.37.

Capítulo 4. Conclusiones y trabajo a futuro

4.1 Conclusiones

Al iniciar este trabajo de tesis, se consideraba que la imagen debía ser estática o con movimientos apenas visibles, por lo que se aplicó una metodología de compensación de movimiento a partir de la clasificación de los movimientos, sin embargo, conforme se fue avanzando en el trabajo, se decidió aplicar la homografía para la estabilización, y de esta manera tener reducir el error al momento de realizar los cálculos, contrarrestando el movimiento de la cámara. La aplicación de la estabilización proporcionó buenos resultados, y en combinación con la construcción del panorama, se puede analizar un escenario completo.

Al principio, se consideraba únicamente la detección de objetos mediante el método *Shi-tomasi*, sin embargo, debido a que un vehículo podía contener más de 1 punto característico, se buscó la implementación de otro método de clasificación con el que pudiese trabajar en conjunto. Implementar una red neuronal como lo es el *YOLO* para la clasificación y localización del vehículo, proveería no solo la identificación del tipo de vehículo, sino también las dimensiones en las que se encuentra, ofreciendo la información necesaria para reducir la sobrecarga de puntos característicos por vehículo.

El método puede ser utilizado para el análisis de tráfico vehicular, ya que los resultados obtenidos nos muestran que existe un error muy bajo en las velocidades estimadas por el algoritmo propuesto en comparación con las que presenta la empresa RCE Systems [30]. Dado el resultado obtenido en la validación, se puede concluir que se cumplieron los objetivos establecidos para este trabajo de tesis.

4.2 Trabajo a futuro

Como trabajo a futuro se desea realizar lo siguiente:

- Dado que el desarrollo del algoritmo fue realizado de manera secuencial, se considera la implementación del algoritmo utilizando procesamiento paralelo mediante el OpenMP, y de esta manera reducir los tiempos de procesamiento para las diferentes etapas del proceso para el desarrollo de este trabajo de tesis.
- La implementación del OpenMP abre la puerta a la implementación del proceso para su uso en tiempo real.
- Adecuar el algoritmo para la implementación de métodos que proporcionen mayor información descriptiva para un punto característico (SIFT) para el mejoramiento del filtrado y almacenamiento del historial.
- Implementar métodos de creación de fondo para la construcción del escenario al momento de armar la vista panorámica.

Referencias

[1] «Instituto Nacional de Estadística y Geografía. (2019). Estadísticas de Vehículos de Motor Registrados,» INEGI, [En línea]. Disponible: https://www.inegi.org.mx/temas/vehiculos/. [Último acceso: 14 11 2019].

[2] A. Puri, «Survey of Unmanned Aerial Vehicles (UAV) for Traffic Surveillance,» p. 29, 2005.

[3] E. Barmpounakis, E. Vlahogianni y J. Golias, «Extracting Kinematic Characteristics from Unmanned Aerial Vehicles,» de *TRB 95th Annual Meeting*, Washington, DC, USA, 2016.

[4] B. Coifman, M. McCord, R. G. Mishalani, M. Iswalt y Y. Ji, «Roadway traffic monitoring from an unmanned aerial vehicle,» *IEE Proceedings-Intelligent Transport Systems,* vol. 153, nº 1, pp. 11-20, March 2006.

[5] P. Tokekar, J. V. Hook, D. Mulla y V. Isler, «Sensor Planning for a Symbiotic UAV and UGV system for Precision Agriculture,» *IEEE Transactions on Robotics,* vol. 32, nº 6, pp. 1498-1511, 2016.

[6] . P. De Berardinis, D. Dominici, D. Godone, P. Hobbs, V. Lechner, T. Niedzielski, M. Piras, M. Rotilio, R. Salvini, V. Segor, B. Sotier y F. Troilo, «The use of unmanned aerial vehicles (UAVs) for engineering geology applications,» *Bulletin of Engineering Geology and the Environment,* 2020.

[7] F. Patrona, I. Mademlis, A. Tefas y I. Pitas, Computational UAV Cinematography for Intelligent Shooting Based on Semantic Visual Analysis, 2019, pp. 4155-4159..

[8] «Organización de las Naciones Unidas,» 2019. [En línea]. Disponible: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/.

[9] M. Sonka, V. Hlavac y R. Boyle, Image Processing, Analisys, and Machine Vision, Fourth ed., C. Learning, Ed., 2015, p. 869.

[10] R. C. Gonzales y R. E. Woods, Digital Image Processing, Third Edition ed., Pearson Prentice Hall, 2008.

[11] C. Harris y M. Stephens, «A combined corner and edge detector,» *Plessey Research Roke Manor,* pp. 125-151, 1988.

[12] J. F. Canny, *Finding Edges and Lines in Images,* 1983.

[13] J. Shi y C. Tomasi, «Good features to track,» *IEEE,* p. 593–600, 1994.

[14] D. G. Lowe, «Distinctive Image Features,» *International Journal of Computer Vision,* vol. 60, nº 2, pp. 91-110, 2004.

[15] A. Punchihewa, Ed., Video Compression, 1 ed., InTech, 2012, p. 160.

[16] B. K. T. Lucas, «An iterative image registration technique with an application to stereo vision,» *Proceedings of IJCAI,* p. 121–130, 1981.

[17] J. Beyerer, R. Matthias y N. Matthias, Pattern recognition: introduction, features, classifiers and principles, Berlin/Boston, 2017.

[18] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection,» *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 779-788, 2016.

[19] M. Chablani, «towardsdatascience.com,» 20 Agosto 2017. [En línea]. Disponible: https://towardsdatascience.com/yolo-you-only-look-once-real-time-object-detection-explained-492dc9230006. [Último acceso: octubre 2020].

[20] D. Forsyth y J. Ponce, Computer Vision: A Modern Approach, 2 ed., Pearson, Ed., 2012, p. 761.

[21] R. Hartley y A. Zisserman, Multiple View Geometry in computer vision, segunda ed., CAMBRIDGE UNIVERSITY PRESS, 2003.

[22] R. Ke, Z. Li, J. Tang, Z. Pan y Y. Wang, «Real-Time Traffic Flow Parameter Estimation From UAV Video Based on Ensemble Classifier and Optical Flow,» *IEEE Transactions on Intelligent Transportation Systems,* vol. 20, nº 1, pp. 54-64, jan. 2019.

[23] P. Viola y M. Jones, «Rapid object detection using a boosted cascade of simple features,» de *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001.

[24] L. Kang y M. Gellert, «Fast multiclass vehicle detection on aerial images,» *IEEE Geoscience and Remote Sensing Letters,* vol. 12, nº 9, pp. 1938-1942, 2015.

[25] F. J. Hernandez Lopez y M. Rivera, «AVScreen: a real-time video augmentation method,» *Journal of Real-Time Image Processing,* vol. 10, pp. 1-13, 2013.

[26] «Data From Sky,» REC Systems, 20202. [En línea]. Disponible: https://datafromsky.com/. [Último acceso: 2020].

[27] «AlexeyAB - Yolo_mark,» [En línea]. Disponible: https://github.com/AlexeyAB/Yolo_mark. [Último acceso: 2019].

[28] «AlexeyAB - darknet,» [En línea]. Disponible: https://github.com/AlexeyAB/darknet. [Último acceso: 2019].

[29] M. Santés y J. F. Vigueras, «Automatic camera localization, reconstruction and segmentation of multi-planar scenes using two views,» de *Mexican International Conference on Artificial Intelligence*, 2009.

[30] «RCE Systems,» 2018. [En línea]. Disponible: https://www.rcesystems.cz/. [Último acceso: 2020].

[31] K. Kanistras, G. Martins, M. Rutherford y K. Valavanis, «Survey of unmanned aerial vehicles (UAVs) for traffic monitoring. In Handbook of Unmanned Aerial Vehicles,» *Springer: Dordrecht,,* p. 2643–2666, 2015.

[32] P. Alvarez Lopez , M. Behrisch , L. Bieker-Walz, J. Erdmann, Y.-P. Flotterod, R. Hilbrich, L. Lucken, J. Rummel, P. Wagner y E. Wiessner, «Microscopic Traffic Simulation using SUMO,» [En línea]. Disponible: https://www.eclipse.org/sumo/. [Último acceso: 2020].

[33] A. M. Torres Alzamora , «Análisis y Comparación de Criterios de Diseño Geométrico en las Rotondas Modernas,» 2015. [En línea].