Sistema de adquisición de parámetros de tráfico vehicular

ING. FRANCISCO CARLOS CALDERÓN BOCANEGRA

TRABAJO DE INVESTIGACIÓN PARA OPTAR POR EL TÍTULO DE MAGISTER EN INGENIERÍA ELECTRÓNICA

DIRECTOR
ING. ALEJANDRO FORERO GUZMÁN, M.Sc.

PONTIFÍCIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE ELECTRÓNICA
BOGOTÁ DC, 2010

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA ELECTRÓNICA

RECTOR MAGNÍFICO:

PADRE JOAQUÍN EMILIO SÁNCHEZ GARCÍA S.J

DECANO ACADÉMICO:

ING. FRANCISCO JAVIER REBOLLEDO MUÑOZ

DECANO DEL MEDIO UNIVERSITARIO:

PADRE SERGIO BERNAL RESTREPO S.J

DIRECTOR DE LA MAESTRIA EN INGENIERÍA ELECTRÓNICA:

ING. CARLOS ALBERTO PARRA RODRÍGUEZ Ph.D

DIRECTOR DE PROYECTO:

ING. ALEJANDRO FORERO GUZMÁN. M.Sc

ARTICULO 23 DE LA RESOLUCIÓN No. 13 DE JUNIO DE 1946

"La Universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado.

Solo velará porque no se publique nada contrario al dogma y a la moral católica y por que los trabajos no contengan ataques o polémicas puramente personales. Antes bien, que se vea en ellos el anhelo de buscar la verdad y la justicia.

1 Tabla de contenido

1.1 Índice de figuras

1.2 Índice de tablas

2 Introducción

La ingeniería del tráfico es una rama de la ingeniería civil que se encarga de planear, diseñar y organizar la operación del tráfico en calles y autopistas, con el fin de obtener una movilidad segura y eficiente[1].

Como herramienta en el diseño de vías, se usan modelos de predicción, que son diseñados basándose en muestras de los tres principales parámetros de tráfico, que se enumeran continuación. Primero, el volumen [1]que se define como el número total de vehículos que pasan por una determinada sección de la carretera en un intervalo de tiempo dado. Ya sean años, días, horas o menos. Suele ser un valor real medido directamente en la vía en el tiempo total de la medición, cuando el tiempo involucrado es de una hora el volumen es llamado comúnmente intensidad.[2]

Dentro de los sistemas actuales de adquisición de parámetros de tráfico vehicular están los detectores magnéticos, tubos de presión, pistolas de radar y sensores de microondas. De lejos, los detectores magnéticos son los más utilizados para el monitoreo de sistemas de tráfico [1], [3]. Son instalados en el suelo, sobre la superficie de la vía y cuentan el número de carros. Los sensores de presión cuentan los carros cuando han pasado. Las pistolas de radar son el método más preciso para monitorear de la velocidad y son el sistema más utilizado por la policía.[1], [3].

Estos sistemas solo permiten extraer un número limitado de información relacionada con el tráfico, mientras que la visión artificial posee beneficios potenciales sobre estos sistemas convencionales. Además de contar vehículos y medir la velocidad, se puede proporcionar ruta del vehículo, tipo de vehículo, densidad, identificación de congestiones de tráfico, accidentes e identificación de vehículos por número de placa, la longitud y tiempo de espera en una intersección, el número de cambios de carril, entre otros,[3] ,[4]. Otras posibles ventajas son su fácil instalación, su bajo costo de mantenimiento, la flexibilidad en la implementación y que más de un carril puede ser monitorizado mediante este sistema.

El uso de visión artificial en sistemas de monitoreo de tráfico puede ofrecer ventajas adicionales, se puede realizar en tiempo real y con los sistemas de comunicación actuales es posible la transmisión de información de manera remota para su posterior análisis, además de que su instalación no requiere modificar físicamente la vía[1],[2], [3], [4],[5].

En los últimos años, con el aumento de la población en la ciudad producto de diversos factores demográficos, desplazados, etc. Y distintos factores económicos producto de la globalización, hacen que el parque automotor aumente. Al haber una mayor afluencia de vehículos a las vías de la ciudad y sin tener una cultura vehicular y peatonal bastante arraigada genera que se presenten más accidentes de tránsito en la ciudad. Como menciona el fondo de prevención vial en su página Web [6].

"*Durante los primeros cuatro meses del año 2007 se presentaron en Bogotá 12.123 accidentes de tránsito que dejaron 162 pérdidas de vidas, 6.121 lesionados y 6.971 choques simples; lo cual significa que cada día se registraron 1,35 víctimas, 51 accidentes con heridos y 58 choques simples. El 56% de las víctimas son peatones y el 20% motociclistas. Administración distrital lanza ofensiva de educación, prevención y control en las vías*." [6]

Muchas otras ciudades en el mundo, como Londres, ven en la utilización de cámaras de video para monitorear el tráfico una solución para el control de tráfico y de esta manera detectar zonas vulnerables y de alta accidentalidad, posibles infractores, velocidad de flujo de vehículos, reporte de accidentes, entre otros, en especial si este monitoreo se realiza de manera continua y automática [6].

3 Marco teórico

3.1 Ingeniería de tráfico vehicular principios básicos**[1]**

La ingeniería del tráfico es una rama de la ingeniería civil que se encarga de planear, diseñar y organizar la operación del tráfico en calles y autopistas, con el fin de obtener una movilidad segura y eficiente. Como herramienta en el diseño de vías, se usan modelos de predicción, que son diseñados basándose en muestras de varios parámetros de tráfico:

3.1.1 Volumen e Intensidad De Circulación:

Son dos medidas que indican el número de vehículos que pasan por un segmento de vía durante un intervalo de tiempo determinado. El Volumen se define como el número total de vehículos que pasan por una determinada sección de la carretera en un intervalo de tiempo dado. Ya sean años, días, horas o menos. Y es un valor real medido directamente en la vía en el tiempo total de la medición

La intensidad horaria se define como el número de vehículos que pasan por un segmento de vía durante un intervalo de tiempo inferior a una hora, pero expresado como una intensidad horaria equivalente. Se obtiene dividiendo el volumen registrado en un periodo entre la duración del mismo expresado en horas “vehículos / hora”.

3.1.2 Velocidad

Se define como la distancia recorrida por un vehículo por unidad de tiempo y se expresa en kilómetros por hora; Para caracterizar el tráfico, se usa como valor representativo la velocidad media, que se obtiene al promediar las velocidades individuales de cada vehículo.

La velocidad media de recorrido se calcula tomando una porción de la vía y dividiéndola entre el tiempo medio que tardan los vehículos en recorrerla, en este cálculo están incluidas las demoras del tráfico por detenciones completas o congestión del mismo

Si se tiene en cuenta el tiempo en el cual el vehículo está en movimiento el resultado de velocidad seria la velocidad media de movimiento

3.1.3 Densidad

La densidad se define como el número de vehículos que ocupan un segmento de longitud especifico de una vía de tal manera que esta densidad está dada en vehículos por kilómetro, este valor depende del segmento considerado y del tiempo en el que se realizó la medición.

La densidad puede ser medida mediante un mecanismo que permita contar los vehículos en un tramo de la vía deseado, en un instante de tiempo. Teóricamente se puede llegar al mismo resultado aplicando la ecuación fundamental del tráfico:

$$\text{densidad}(veh/km) = \frac{\text{intensidad}(veh/h)}{\text{velocidad}(km/h)}$$

3.2 Visión por computador

La adquisición de variables de tráfico vehicular se realizará usando visión por computador, específicamente el análisis de una señal de video digital con un procesador digital usando técnicas estándar documentadas en la bibliografía.

3.3 Captura del video

El video es capturado de una cámara digital directamente conectada al computador, no existe un conversión analógica de por medio, esto con el fin de evitar los efectos del barrido generado por las cámaras análogas que consiste en la composición de un cuadro de video a partir del intercalado de dos imágenes tomadas en tiempos diferentes. Este efecto es indeseado en ciertas aplicaciones de visión por computador el video es adquirido mediante escaneo progresivo y entregado al computador usando el puerto USB.

3.4 Bibliotecas de software

Una biblioteca de software consiste en un conjunto de funciones descritas en un lenguaje de programación en específico que permiten agilizar el trabajo y brindar una interface de desarrollo al programador o API "*Application Program Interface*" en el tema de visión por computador existen distintas de estas bibliotecas, en el grupo de investigación SIRP la biblioteca usada es OpenCV [7].

3.5 Opencv

OpenCV es un conjunto de bibliotecas de código abierto u Open Source bajo licencia BSD [8] desarrolladas en un principio por Intel, disponibles desde [7]. Están desarrolladas en lenguaje C, con puertos a C++, phyton y octave. Además, se pueden ejecutar desde diversos sistemas operativos como Windows, GNU/Linux y Mac OS X y desde diversas plataformas de hardware basadas en procesadores x86, ARM y PowerPC [9].

Estas bibliotecas ofrecen código diseñado eficientemente, orientado a aplicaciones capaces de ejecutarse en tiempo real en procesadores modernos. Estas bibliotecas tienen como objetivo proveer las funciones más usadas en Visión por computadora y tiene más de 500 funciones implementadas. Se usan ampliamente en Vigilancia, imágenes médicas, inspección industrial, interfaces de usuario, calibración de cámaras y recientemente se han usado estas bibliotecas en imágenes aéreas y mapas de calles, por ejemplo en la herramienta Google's Street View. Se hace uso de la calibración de cámara y de funciones de OpenCV para hacer el proceso de *stitching*, que consiste en combinar múltiples imágenes para producir una imagen panorámica o una imagen de alta resolución. [6]

3.6 Tareas e hilos

Para implementar el algoritmo propuesto es necesario el uso de más de una tarea que sea ejecutada por el procesador, para esto se usó la biblioteca de *pthread* [10] presente en todo sistema operativo compatible con POSIX [11].

3.7 Estimación de primer plano

En visión por computador es común tener el caso del análisis de un video proveniente de una cámara fija en la que los pixeles de la imagen pueden ser separados en dos clases: el fondo y el primer plano.

La primera clase correspondería a el fondo de la imagen que corresponde a los objetos estáticos en la imagen o objetos que presentan un movimiento constante, por ejemplo en una escena de un bosque los arboles harían parte del fondo, sus troncos totalmente fijos y sus hojas que son movidas por el viento, o en una escena del interior de un centro comercial, si una cámara ve hacia unas escaleras eléctricas, estas harían parte del fondo junto con el piso y las barandas.

La segunda clase corresponde al primer plano, que son pixeles distintos al fondo de la escena que presenten un movimiento deseable a analizar.

Del fondo y el primer plano es posible crear una imagen lógica, se acostumbra asignar un uno lógico al primer plano y un cero lógico al fondo. Existen técnicas distintas para obtener esta máscara de primer plano, las más básicas consisten en encontrar una imagen que corresponde al fondo, realizar una resta en valor absoluto entre esta imagen de fondo y el cuadro de video y umbralizar el resultado[12].

Este tipo de algoritmo de obtención del primer plano mediante una imagen de fondo tiene la desventaja de solo poder modelar un fondo de la escena, por ejemplo en la escena del bosque, los troncos de los arboles serian bien modelados, pero las hojas movidas por el viento no, además se generan falsos positivos de primer plano con los cambios bruscos de iluminación o con las sombras de los objetos móviles.[13]

Se puede llegar a un modelo más complejo que el anterior usando imágenes creadas a partir de la media, de la varianza y la covarianza por pixel de las imágenes del video como se muestra en [12] en el capítulo 9 y en[14], en las que se usan las siguientes ecuaciones por pixel para obtener un método de estimación de la media, la varianza y la covarianza que puedan ser calculados conforme acontece el video.

$$\bar{x} = \frac{1}{N}\sum_{i=0}^{N-1}(x_i)$$

$$\sigma^2 = \left(\frac{1}{N}\sum_{i=0}^{N-1} {x_i}^2\right) - \left(\frac{1}{N}\sum_{i=0}^{N-1} x_i\right)^2$$

$$cov(x,y) = \left(\frac{1}{N}\sum_{i=0}^{N-1}(x_i y_i)\right) - \left(\frac{1}{N}\sum_{i=0}^{N-1} x_i\right)\left(\frac{1}{N}\sum_{i=0}^{N-1} y_i\right)$$

Para la última ecuación x es la imagen en el tiempo t y y es la imagen en el tiempo $T = t - d$, donde d es un atraso. Estas funciones de acumulación pueden ser usadas para crear una variedad de modelos estadísticos del fondo[13]. Una posible mejora a este modelo es variar las condiciones de actualización de la media móvil, añadiendo condiciones de color [13] para lograr una convergencia más rápida a colores de fondo esperados, como lo es el pavimento, o cambiar los umbrales de determinación del primer plano de manera adaptativa [10]; usando un modelo normal sobre la imagen, que permita analizar el mejor umbral a ser usado.

Existen algoritmos de obtención del primer plano más acertados a los mostrados anteriormente este se realiza al modelar el fondo en un conjunto de parámetros que luego son revisados para determinar a qué clase corresponde cada pixel.

El algoritmo conocido como "*FGDStatmodel"* por sus siglas en ingles, que corresponde a un conjunto de algoritmos de Detección estadística de primer plano, consiste en una parte de estimación estadística por ocurrencias de color descrita en [15] y otra parte de estimación de umbrales de movimiento descrito en [16]. El grupo de investigación SIRP ha trabajado este algoritmo y hecho algunas modificaciones propias a tráfico vehicular documentadas en [13]

El algoritmo de libro de códigos o "*Codebook*" descrito por primera vez en [17], e implementado y descrito en OpenCV [12]. La implementación de ejemplo básica de este algoritmo consiste en el análisis de los primeros N cuadros con el fin de obtener un modelo a usar en la totalidad del video, este modelo debe actualizarse cada cierto tiempo con el fin de poder se adaptable a cambios en el fondo como iluminación o condiciones atmosféricas. También trabaja en el espacio de color YCrCb y el ejemplo está implementado de esta manera para mejorar la separación de componentes claras y oscuras de un video de prueba del algoritmo [12], en el presente documento se realiza una prueba de este algoritmo en los espacios de color RGB y HSV y otra alternando dos de estos *Codebook* en HSV durante todo el video para mantener el modelo actualizado.

3.8 Calibración de la cámara

Con el objetivo de poder realizar medidas sobre la imagen y corregir los efectos ópticos que pueda introducir la fabricación física de la cámara de video, la cámara es calibrada de manera previa usando el modelo intrínseco de "*pin hole*" bastante sencillo y antiguo[18] y [19] y un modelo extrínseco implementado en la biblioteca de OpenCV descrito en el libro oficial de OpenCV[12] en el capítulo 11 que a su vez es basado en [20].

4 Especificaciones

El grupo de investigación SIRP como parte de su investigación en movilidad vehicular y peatonal, requiere de una plataforma en hardware y software para la implementación de los sistemas de adquisición de información del tráfico vehicular y peatonal mediante visión artificial, la primera parte de este proyecto consiste en la selección de una la plataforma de procesamiento en hardware que permita flexibilidad en la implementación de algoritmos de visión por computador relacionados con tráfico vehicular y peatonal y la caracterización de una ubicación física de la cámara sobre la vía que está determinada por la mejor posición para ver los vehículos sin que se presenten traslapes entre vehículos en la imagen como en la Figura 1 y la Figura 2. Este criterio depende de las condiciones geométricas de la instalación física de la cámara y del alto de los vehículos que transitan por la vía [14].

Para la Implementación del sistema se propone una ubicación o serie de ubicaciones en las cuales el sistema de adquisición pueda ser situado en la vía [2], definiendo un intervalo de altura con respecto a la cantidad de carriles, un intervalo de ángulos con respecto a la vertical transversales a la vía como se ve en la Figura 2. Y un ángulo con respecto a la vertical, paralelo a la vía mostrados en la Figura 1.

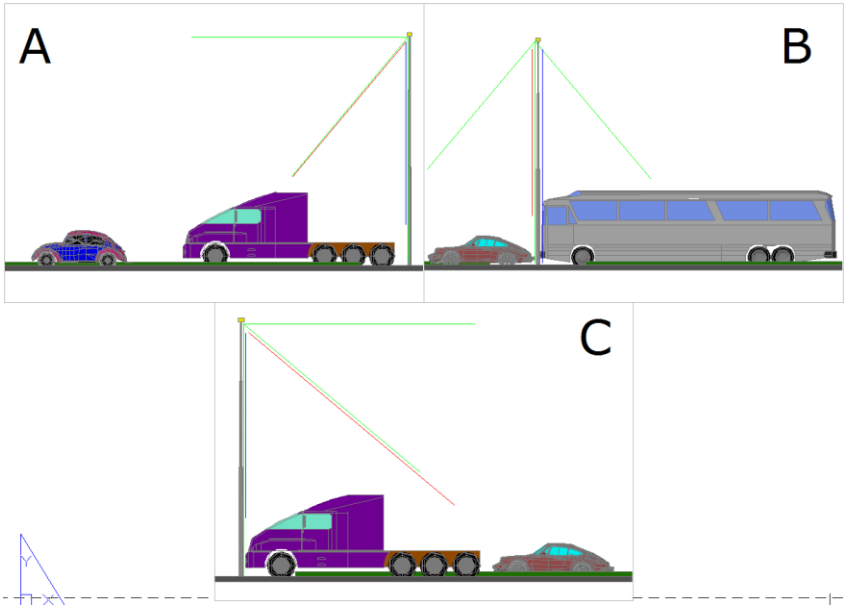

Figura 1. Ángulos Horizontales en la vista de la cámara.

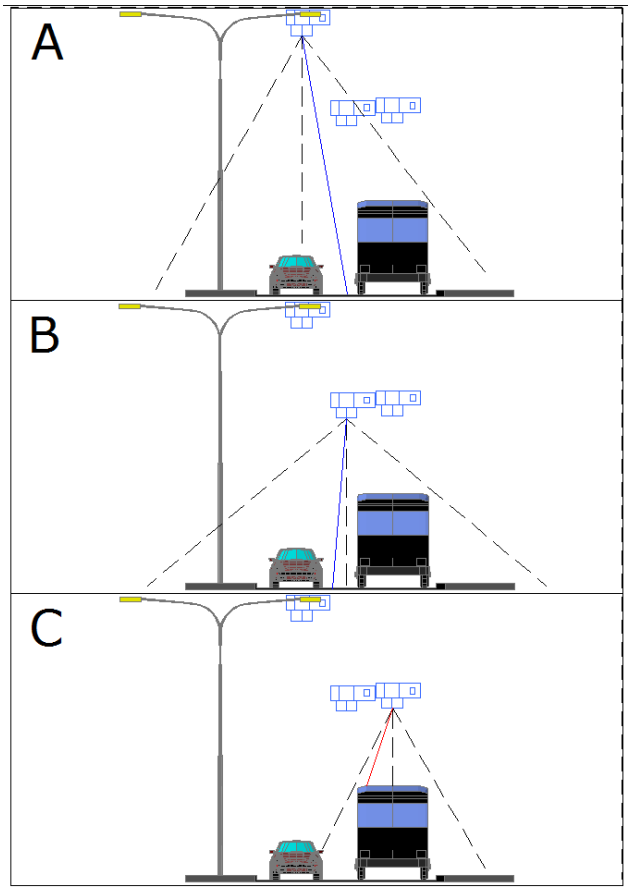

Figura 2. Ángulos transversales en la vista de la cámara.

Estas ubicaciones son propuestas de tal manera que por la geometría ocurran los menores traslapes posibles en la imagen, minimizando el error por traslape y ocultación de vehículos en base a cálculos sobre los valores estimados del tamaño de los vehículos.

Para Estimar los parámetros de tráfico basta con encontrar dos y despejar el tercero de la ecuación fundamental de Tráfico. Esta estimación se puede hacer por medio de video, el cual puede ser procesado en línea, "tiempo real", o ser tomado en la ubicación y almacenado para su posterior procesamiento. La salida del algoritmo consiste en estos tres parámetros de tráfico, entregados cada cierto tiempo según lo especifique el usuario. El procesamiento solo depende de la entrada de video y la tasa muestreo de la grabación, sin tener ninguna otra entrada de datos.

Seguido a la caracterización física de la plataforma se procede al desarrollo de algoritmos de estimación de la velocidad, de la densidad y del volumen de tráfico sobre una vía de no más de 3 carriles por sentido. Estos algoritmos deben ser diseñados teniendo en cuenta las prestaciones de la plataforma y deben reportar los datos a un usuario local o remoto.

El algoritmo de estimación de las variables de tráfico puede descomponerse en 4 grandes bloques. Primero, la Captura del video que puede ser de una cámara en vivo o de una grabación hecha en la vía. Segundo, una etapa de pre procesamiento y detección de vehículos en la que está contemplado el mejoramiento de la imagen para un posterior análisis que permita realizar la detección de los vehículos en la escena. Tercero, un procesamiento y análisis de la posición de los vehículos detectados que permita estimar la velocidad. Por último la síntesis de los datos de detección posición y velocidad de los vehículos que permita extraer del video una estimación de las tres variables de tráfico vehicular. Puede verse un diagrama en bloques del algoritmo descrito en la Figura 3.

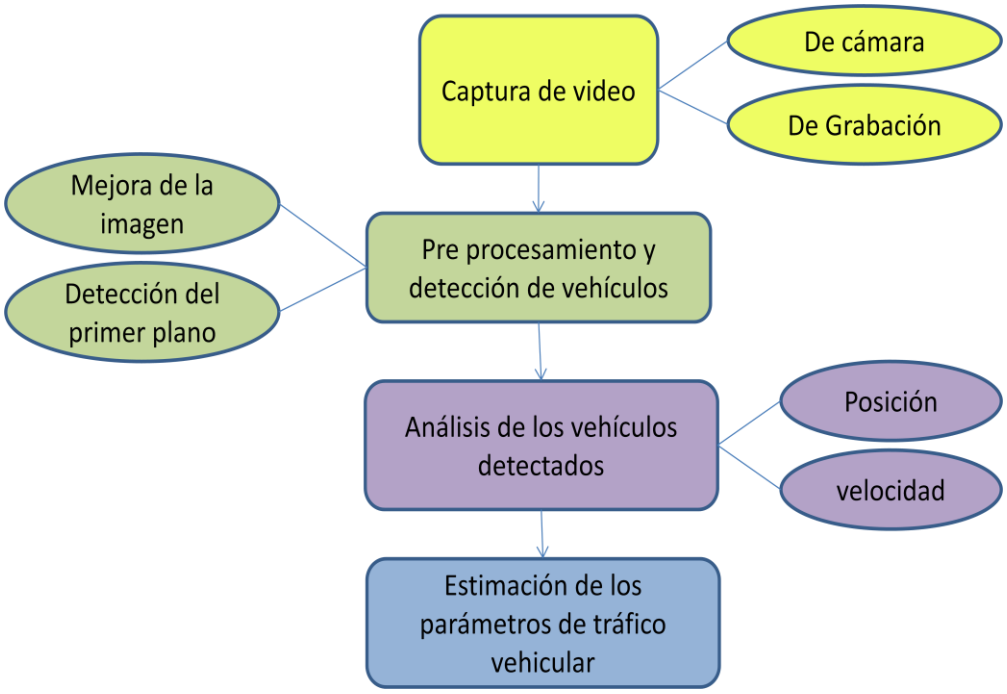

Figura 3. Diagrama en bloques estimado del algoritmo

Los algoritmos de estimación de parámetros de tráfico vehicular estarán basados en la estimación del primer plano. A partir de esta imagen binaria y sus características se procederá a extraer la información de cuantos vehículos hay en escena y el tamaño y posición de estos vehículos, realizando la estimación cuadro a cuadro de la posición de cada vehículo y con el número de cuadros por segundo dado por la cámara, puede encontrarse la velocidad media de cada vehículo en escena, además, usando la ecuación fundamental del tráfico puede encontrarse la densidad vehicular.

Las primeras etapas del algoritmo consisten en el aprendizaje continuo del fondo de la vía, este fondo no es del todo estático[14], las variaciones en la luminosidad, cambios en la vía, o las condiciones climáticas pueden cambiar el fondo actual, por lo que se requiere de un modelo capaz de lidiar con estos cambios y obtener un primer plano de la vía que se mantenga estable a pesar de los cambios que pueda sufrir el fondo.

Seguido a la etapa de detección de los objetos, es necesario limpiar la imagen de primer plano para disminuir el ruido presente en la etapa de detección, esto se hace mediante el filtrado de los contornos de la imagen binaria por área y perímetro. Posterior a esto es necesaria una etapa de seguimiento del objeto y de identificación cuadro a cuadro lo que permite estimar la velocidad, con el objeto detectado puede encontrarse un estimado de objetos en la vía por unidad de tiempo, el tiempo puede encontrarse de los cuadros por segundo de la captura de video o de cámara y del seguimiento de los objetos puede encontrarse la velocidad de los vehículos y despejarse el tercer parámetro de tráfico directamente de la ecuación fundamental del tráfico [1], [14].

Por último la selección de hardware de la plataforma depende de los tiempos de ejecución del algoritmo y podrá ser propuesta uno o más configuraciones dependiendo de las condiciones de la vía.

5 Desarrollos

5.1 Descripción del hardware del sistema

Con el fin de obtener un sistema de medición de variables de tráfico, es necesario proporcionar todos los detalles físicos o de "hardware" en este caso la selección de la cámara de video, la ubicación de la cámara sobre la vía, la transmisión de datos de entrada salida y el sistema de procesamiento de la información.

5.1.1 Entrada y salida de datos

La entrada de datos al sistema corresponde al video y la tasa de captura, adicionalmente se deben fijar los parámetros de calibración intrínsecos de la cámara, la altura a la que se encuentra la cámara y las variables de calibración de los algoritmos de estimación de fondo y seguimiento que dependen del algoritmo usado. La salida del sistema son múltiples archivos de texto generados por el programa cada cierto tiempo especificado por el usuario, donde se consignan las variables de tráfico: velocidad, densidad e intensidad.

5.1.2 Cámara de video

Para el desarrollo de este sistema se escogió la cámara de video *PlayStationEYE* [21], esta es desarrollada por SONY® y fabricada por ECEE para su uso en la consola de juegos PlayStation3, en el que es usada como medio de entrada de video para videoconferencia, pero también como control de juegos interactivos, por lo cual fue diseñada con características deseables para cualquier sistema de visión por computador. Además, al ser de producción masiva, puede conseguirse fácilmente a un bajo precio comparado con otros sistemas de similares características.

En su selección, se tuvo en cuenta las características del sensor CMOS, que en el caso de la cámara *PlayStationEYE* es el OV7720 [22], este funciona en un amplio intervalo de iluminación, característica fundamental ya que el sistema va a ser usado en ambientes exteriores y debe funcionar con iluminación natural y artificial. También se debe anotar que se descartaron los sistemas análogos debido a que modifican la imagen al usar interlineado en la captura.

La salida de la cámara es digital por un puerto USB 2.0 lo que maximiza la compatibilidad con los sistemas de procesamiento. Respecto a la compatibilidad con diversos sistemas operativos, existen controladores de software o "*drivers*" para el controlador de USB presente en la cámara OV0534-LB50 tanto para el sistema Windows, disponibles en [23], como para GNU/Linux, presentes directamente en los núcleos del SO o "*kernel*" superiores al 2.6.29 [24]. En ambas plataformas estos drivers proveen unas resoluciones y tasas de captura como se indican en la Tabla 1:

Tabla 1. Listados Características de resolución y tasa de captura la Cámara *PlayStationEYE*.

Código de operación en el driver	Resolución en pixeles ancho x alto	Tasa de captura Cuadros / Segundo
00	640x480	15
01	640x480	30
02	640x480	40
03	640x480	50
04	640x480	60
10	320x240	30
11	320x240	40
12	320x240	50
13	320x240	60
14	320x240	75
15	320x240	100
16	320x240	125

5.1.3 Plataformas de procesamiento

Para procesar la información proveniente del video, se escogieron dos procesadores el primero de arquitectura x86 [25] como lo es el Intel Atom, junto con el ARM v7 como el núcleo OMAP[26].

La selección obedeció a que ambos procesadores pueden trabajar con la cantidad de información requerida por el proceso y actualmente se encuentran opciones comerciales que entregan el sistema completo en empaques que permiten proponer soluciones integradas a la cámara estos lo que permite que todo el sistema sea fácilmente ubicable sobre la vía y cuente con bajos requerimientos de potencia. Para desarrollar el algoritmo y hacer las pruebas del software se trabajó con un computador PC. Las plataformas usadas para implementar los pilotos del sistema se listan en la Tabla 2 junto con sus principales características.

Tabla 2. Plataformas en Hardware

Nombre en documento	corei7	Atom	OMAP
Procesador	Intel® CORE i7 720QM	Intel® Atom N270	OMAP3530DCBB72 + NEON
Frecuencia de reloj	1,60 GHz máximo	1,6 GHz	720 MHz
Nucleos/Hilos	4/8	1/2	uno por procesador dedicado
Memoria RWM	3.072 GB DDR3@532MHz	1.016GB DDR2@266MHz	256MB MDDR SDRAM
GPU	Mobility Radeon HD4570	Intel® 945 Express	PowerVR SGX530 @200MHz
DSP	no tiene	no tiene	TMS320C64x+ @520MHz
Tarjeta de desarrollo o PC	Dell studio 1557	Hp mini 110-1125LA	Beagleboard-revC4
Sistema Operativo	Ubuntu 9.10 x86 Kernel 2.6.31-19	Ubuntu 9.10 x86 Kernel 2.6.31-19	Angstrom 2010-3 Kernel 2.6.32-47
Entorno gráfico	Gnome 2.28.1	Gnome 2.28.1	Enlightenment 16.1.0.2
Compilador	GCC 4.4.1 x86	GCC 4.4.1 x86	GCC 4.4.1 ARMv7

5.2 Software empleado

5.2.1 Sistema Operativo

La biblioteca de OpenCV, requiere de un sistema operativo funcionando sobre el procesador, la implementación de esta librería hace que pueda funcionar bajo Windows y cualquier sistema operativo tipo UNIX o que cumpla el estándar POSIX o IEEE 1003 [11]. Para el desarrollo de este trabajo se escogió GNU/Linux por su característica de portabilidad entre arquitecturas de hardware como x86 [25] y ARM [27]. Se trabajó con dos distribuciones de Linux, Para la familia x86 sobre Ubuntu GNU/Linux [28] y para la familia ARM sobre Ångström GNU/Linux [29]. Se usan características de Soft Real time y el código fuente es compilado usando GCC [30], en sus versiones para x86 y ARM respectivamente.

5.2.2 Contenedor y Codec del video

Si se usa la plataforma x86 preferiblemente se debe usar "*.AVI", como contenedor de video soportado, codificado con el códec correspondiente a "XVID". Sin embargo si se va a usar la plataforma OMAP[26] es preferible usar el contenedor m4v codificado con "H264", ya que la versión actual del *Kernel* cuenta con soporte al DSP para decodificación del video.

5.3 Ubicación de la cámara sobre la vía

Una correcta ubicación de la cámara sobre la vía puede aumentar la efectividad del algoritmo de detección de vehículos [14]. De la ubicación de la cámara sobre la vía, es posible variar la altura de esta con el piso y el ángulo de la cámara, este ángulo puede ser de dos maneras, con respecto a la vía o paralelo y perpendicular a la vía o transversal, a continuación se tratan los dos casos.

5.3.1 Ángulo con respecto a la vía o Paralelo.

Puede verse en la Figura 1, donde la línea que forma la hipotensa de los dos triángulos, corresponde al centro de visión de la cámara y forma el ángulo a analizar el cual genera el siguiente problema.

En la Figura 4A donde, h es la altura de la cámara con respecto a la vía, y l representa la altura de los objetos, en este caso la altura de los vehículos que pasan por la carretera. A partir de estos dos valores, es posible definir otros dos, que representan la distancia con respecto al eje x del punto más extremo del vehículo: X_1 es la distancia real en la cual se encuentra el extremo y X_2 la distancia observada por la cámara, estas cumplen la relación:

$$\frac{h}{X_2} = \frac{l}{X_2 - X_1}$$

Despejando X_2:

$$X_2 = \frac{X_1}{1 - \frac{l}{h}}$$

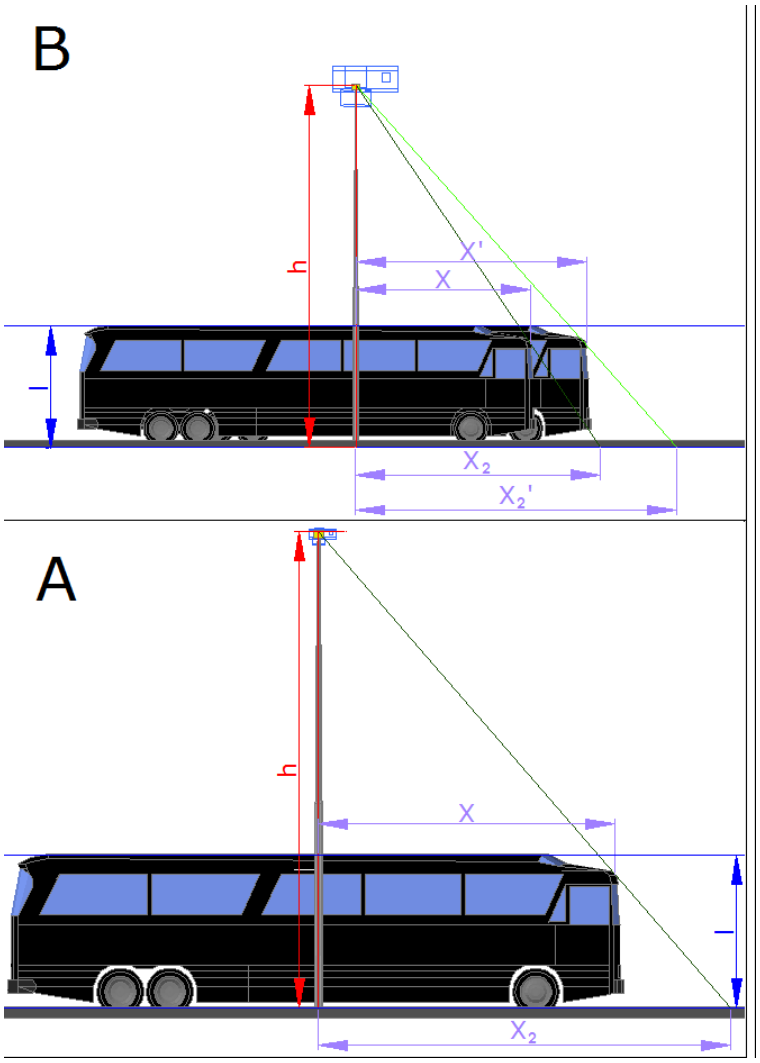

Figura 4. A. Ángulo con respecto a la vía B. cambio de posición del objeto.

La relación l/h es siempre menor a uno y la distancia vista por la cámara depende solamente de esta relación y de X_1. A la hora de contar vehículos este efecto presenta el problema de unir varios vehículos sobre la vista de la cámara, un ejemplo de esto puede verse en la Figura 5. Además del problema presente en la detección del objeto, también se presenta un problema a la hora de estimar la velocidad del vehículo, ya que la distancia vista por la cámara depende de la distancia X_1, el cambio de esta distancia de un cuadro a otro, dado por el movimiento del vehículo provoca que el cambio de posición observado por la cámara sea mayor al real.

Figura 5. Ejemplo real de traslape horizontal.

En la Figura 4 B, se representa el cambio de posición del vehículo de un cuadro a otro, con los respectivos nuevos valores X'$_1$ y X'$_2$, el cambio de posición real está dado por $(X'_1 - X_1)$. Mientras que el visto por la cámara está dado por:

$$(X'_2 - X_2) = \frac{(X'_1 - X_1)}{1 - \frac{l}{h}}$$

Que de nuevo depende de la relación l/h. Una primera consideración a partir de los anteriores resultados, muestra que este efecto puede minimizarse de dos maneras. La primera, haciendo esta relación lo más pequeña posible, lo que es decir que la altura a la cual se ubica la cámara sea grande comparada con el vehículo más alto esperado en la vía. La segunda, midiendo el desplazamiento en la parte delantera de los vehículos si este se mueve hacia la cámara, como se aprecia en la Figura 6 o en la parte de atrás, si este se aleja de la cámara.

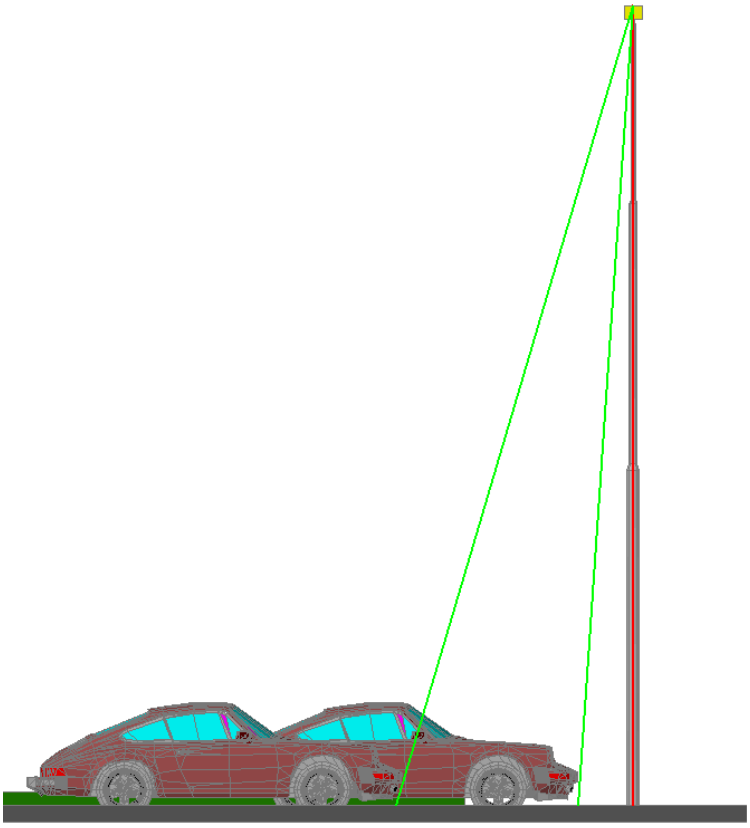

Figura 6. Medición de posición en la parte delantera de los vehículos

5.3.2 Ángulo perpendicular a la vía

Puede verse en la Figura 2, como este ángulo perpendicular afecta la vista de la cámara sobre los vehículos, haciendo que existan oclusiones. La relación entre la altura vehículo y la altura de la cámara, cumple la misma ecuación ya mostrada en el caso paralelo a la vía. La diferencia el caso perpendicular a la vía, costa en que el traslape ocurre entre los costados de los vehículos. A continuación se hace el análisis lateral para vías de uno, dos y tres carriles.

a. Vía de un carril.

Al hacerse la medición en una vía de un solo carril, el traslape entre los costados de los vehículos pierde importancia, ya que solo uno puede pasar al tiempo, por lo que no existe una limitante en la ubicación de la cámara en este ángulo transversal, inclusive si las condiciones laterales de la vía lo permiten como en el caso de una vía sin anden peatonal, la cámara podría estar paralela al suelo y ser ubicada a un costado de la vía dando una vista lateral de los carros.

b. Vía de dos carriles.

En una vía de dos carriles, la ubicación de la cámara si afecta la forma en la que los carros de traslapan, dependiendo del ángulo trasversal y de la altura a la que se encuentre la cámara.

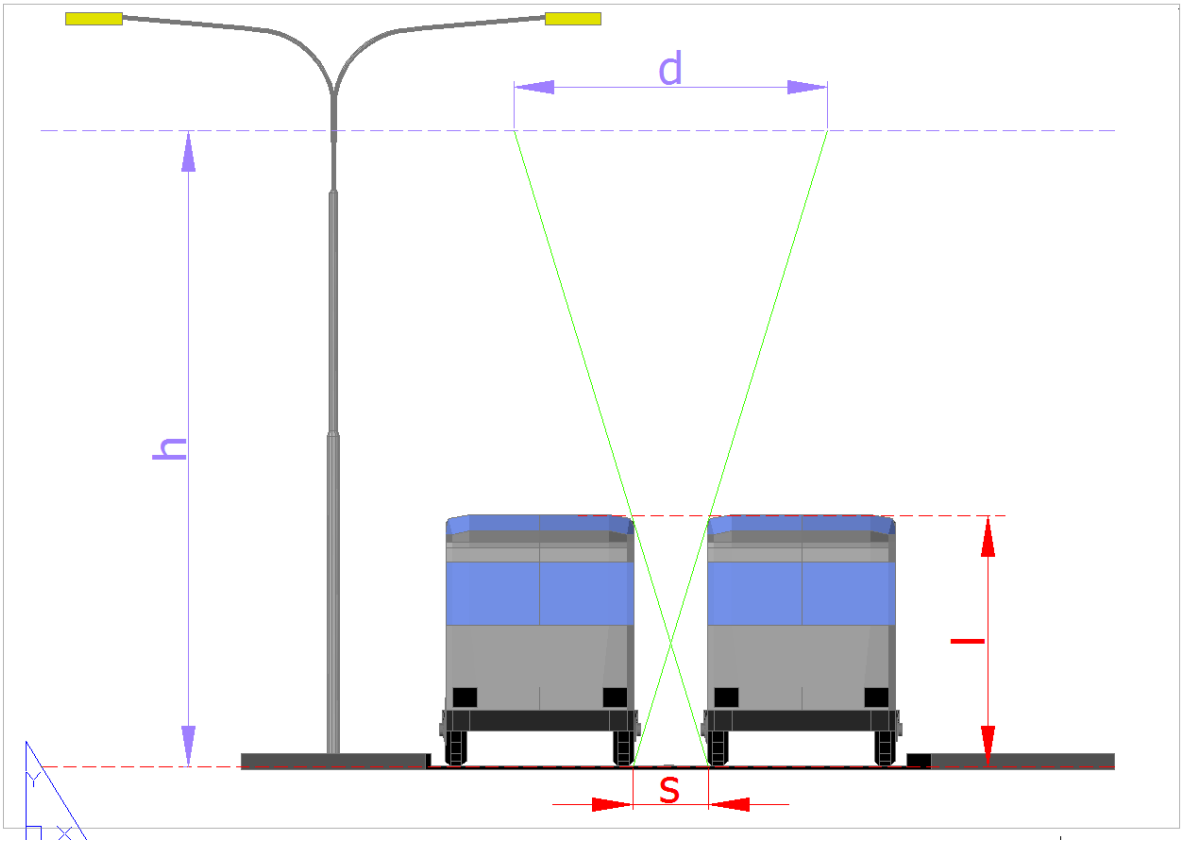

Figura 7 Ubicación de la cámara con 2 carriles.

E la Figura 7, s representa la separación entre vehículos, que puede despejarse del ancho de la vía y del ancho esperado del vehículo de mayor tamaño que transite, l representa la altura del mayor vehículo y h la altura a la cual se encuentra la cámara.

Puede encontrarse la siguiente expresión para la distancia d:

$$d = s\left(\frac{2h}{l} - 1\right)$$

Donde d representa el ancho de la ubicación en la cual puede encontrarse la cámara sin que ocurran traslapes entre vehículos esta también varía con respecto a la relación h/l, si h es mucho mayor a l, la cámara puede colocarse en un lugar fuera de la vía sin presentarse problemas de traslape.

La altura mínima a la cual se deba ubicar la cámara estará dada solamente por la apertura de la cámara y debe ser ubicada preferiblemente de tal manera que cubra la totalidad del ancho la vía.

c. Vía de 3 o más carriles

En este caso en especial, la zona en la cual puede ubicarse la cámara, está dada por la zona común al analizar cada carril por aparte como se muestra en la Figura 8. Al hacer esto se crea esta nueva zona en la que se minimizan los traslapes. Notada como d que se ubica en la parte central de la vía, a una altura superior a h, donde h cumple:

$$h > l\left(\frac{a}{2S} + 1\right)$$

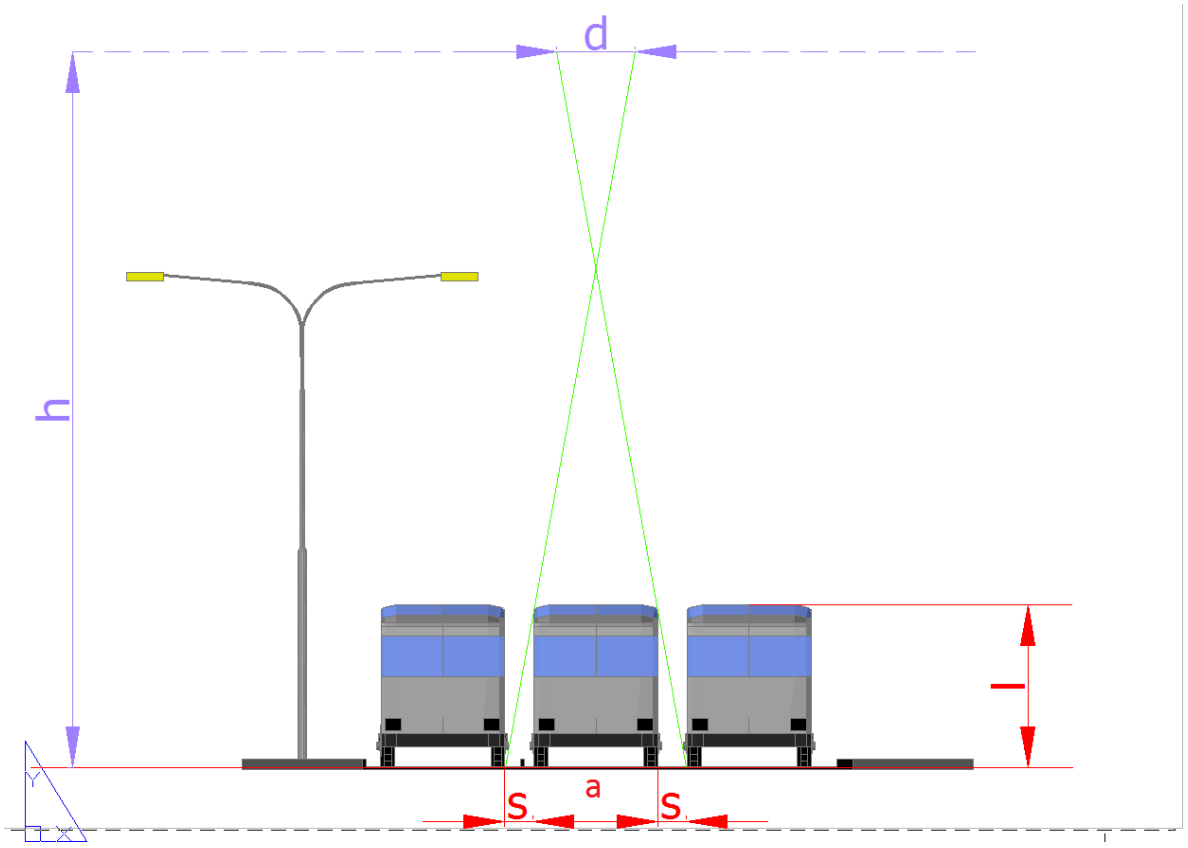

Figura 8 Ubicación de la cámara 3 carriles

En vías de más de tres carriles, pueden encontrarse ubicaciones para las cuales no se presenten traslapes, el inconveniente es que la altura la cual debe colocarse la cámara se vuelve impráctica. Ya que aumenta proporcionalmente con la cantidad de carriles, sin embargo la mejor ubicación sigue estando sobre el centro de la vía.

5.4 Diagrama en bloques del algoritmo propuesto

El algoritmo propuesto consta de 3 secciones, un diagrama en bloques de este puede verse en la Figura 9:

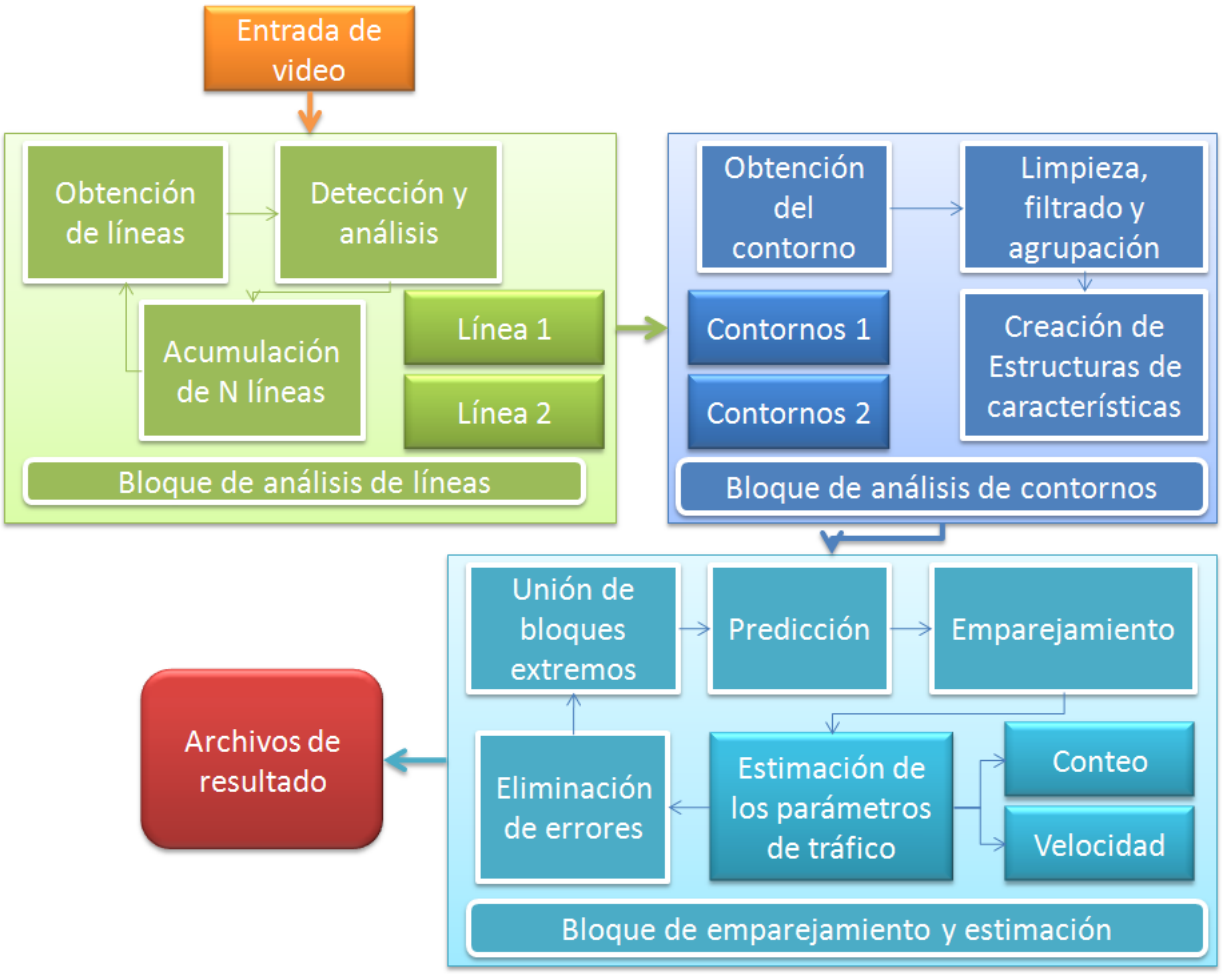

Figura 9. Algoritmo propuesto.

A continuación se explica bloque por bloque del algoritmo.

5.5 Bloque de análisis de líneas

5.5.1 Obtención de líneas acumuladas en el tiempo

Sobre el cuadro de video solo se analizan dos líneas ubicadas de manera perpendicular a la vía como se muestra en la Figura 10, con las líneas en amarillo y verde. Cada una de estas líneas es acumulada en el tiempo en una imagen en la cual cada línea de pixeles horizontales, corresponde a una línea en un cuadro del video. Sobre cada línea se ejecuta el algoritmo de detección basado en primer plano y se acumula la línea binaria de resultado.

Esta forma de capturar información, fue reportada para su aplicación en tráfico vehicular en 1997 por Setchell[4], como un método para aumentar la tasa de cuadros por segundo que puede procesar el algoritmo. También se ha propuesto el uso de la acumulación por líneas en conjunto al uso de sensores laser como detectores de parqueo [31], o como una técnica de análisis de movimiento en la escena y de segmentación [32], este también introdujo el término de tajada espacio temporal o "*Spatio-Temporal Slice*".

En el trabajo documentado en [33], se aplica este concepto para realizar la detección y conteo de vehículos usando acumulación por media y más tarde, los mismos autores [34], fijan algunas consideraciones en cuanto a la velocidad máxima de los vehículos y la tasa de captura requerida para poder detectar los vehículos en la vía. Más recientemente en el trabajo documentado en [35] se aplica la acumulación temporal sobre una línea ubicada por cada carril, en forma paralela a la vía, para estimar la velocidad.

Este método de "*Spatio-Temporal Slice*", es aplicado en este trabajo como un paso a seguir fruto de trabajos previos del grupo SIRP [14] y como una técnica que permite disminuir la cantidad de información a ser procesada y por ende el tiempo de procesamiento por cuadro.

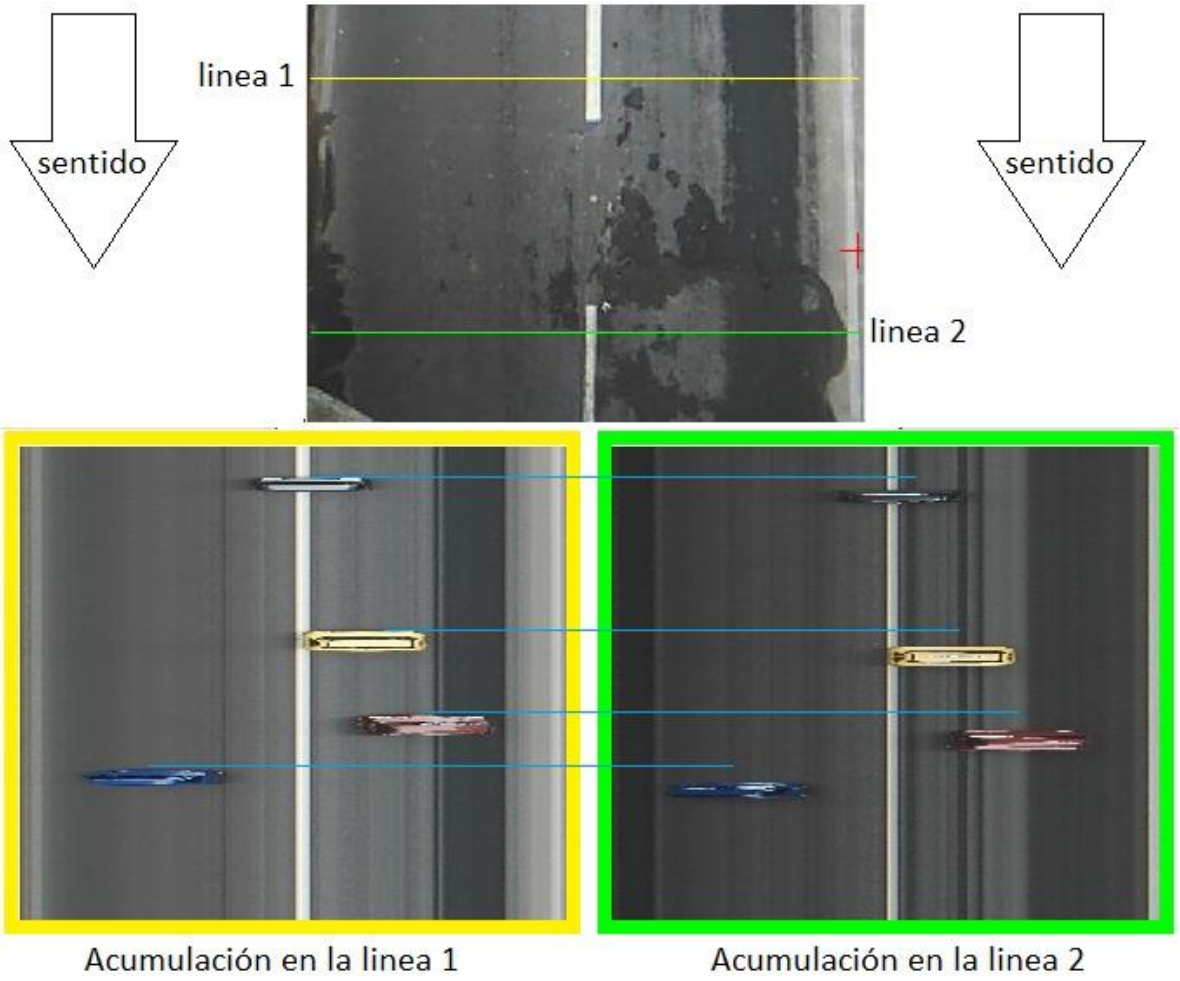

Figura 10. Ejemplo de acumulación de líneas

5.5.2 Detección de los vehículos y acumulación

Cuadro a cuadro, sobre las líneas definidas es aplicado el algoritmo de detección y la línea binaria de resultado es acumulada, como se muestra en la Figura 11 C. Este algoritmo de detección se basa en la estimación de primer plano. Para la selección del algoritmo se realizaron dos pruebas. La primera, que se muestra en el numeral a de esta sección, se realizó sobre la captura por líneas para evaluar el desempeño de los algoritmos trabajado sobre una línea por cuadro. La segunda, evalúa el desempeño de estos tres algoritmos de detección, en conjunto con el algoritmo de análisis de bloques y estimación de variables de tráfico, que se muestra en una sección posterior.

El largo de la imagen acumulada por líneas, notado con la letra N, representa la cantidad de cuadros a acumular. Esto junto con la tasa de captura equivale a un intervalo en tiempo de análisis del video.

a. Pruebas sobre los algoritmos de primer plano

Como una primera comparación de los desempeños de estos algoritmos se tomaron 4 videos representativos de distintas condiciones, descritas en la Tabla 3. Estos videos fueron usados únicamente en la etapa de diseño y calibración de los algoritmos de estimación de primer plano.

Tabla 3. Videos usados para probar el fondo.

Nombre en pruebas	Punto inicio	Punto fin	Número M, o acumulación de N cuadros	Tiempo	fps	Características iluminación	Flujo vehicular
Video 1	(1 359)	(633 359)	4	00:01:30	30	muchas sombras con alto contraste en el piso	normal
Video 2	(1 359)	(633 359)	7	00:03:18	30	iluminación normal con pocos cambios de contraste en el video	normal
Video 3	(1 359)	(633 359)	12	00:03:32	25	iluminación normal, altos cambios de contraste en el video	alto trafico
Video 4	(1 359)	(633 359)	8	00:07:47	25	iluminación normal, altos cambios de contraste en el video	muy lento

Una imagen de acumulación cuenta con un largo determinado por N, por lo que existen M imágenes de acumulación dependiendo del tiempo que dure el video.

De cada uno de estos videos, sobre determinadas imágenes M, compuestas de la acumulación por líneas del video, se halló de manera manual una máscara que demarca el primer plano, de color blanco y del fondo, color de negro. Estas máscaras se utilizan para comparar los algoritmos y cuantificar cuantos pixeles son correctamente clasificados y cuantos presentan un error de clasificación. Tanto falsos positivos, es decir, pixeles que fueron tomados como primer plano siendo fondo, como falsos negativos o pixeles que fueron tomados como fondo siendo primer plano. La obtención de estos resultados se hace a partir de operaciones binarias tal como se muestra en la Figura 11.

Los parámetros de estos algoritmos, de tenerlos se encontraron en pasos de comparación previos en base a la mayor cantidad de aciertos por pixel encontrados por algoritmo. De estos resultados se escogieron los 3 que mas aciertos tenían y que visiblemente mejor definían el contorno de los vehículos, para ser usados más adelante en conjunto con el algoritmo definitivo.

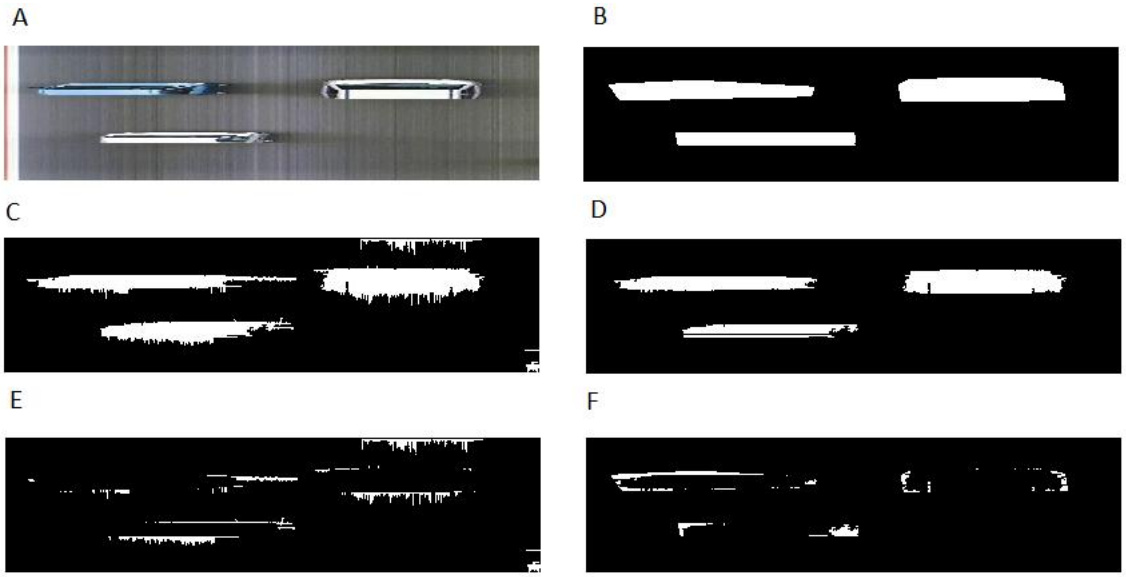

Figura 11. Ejemplo de máscaras, aciertos, falsos positivos y falsos negativos.

De la Figura 11, la imagen A, muestra un ejemplo típico de captura de líneas. La imagen B, es una máscara hallada a mano pixel por pixel, en la que se tiene un uno lógico en los pixeles que pertenecen a primer plano y un cero lógico en los que pertenecen a fondo. La imagen C, es una muestra de la salida de un algoritmo de estimación de primer plano, con las mismas características de la imagen B. La imagen D, muestra el resultado de la operación AND entre las imágenes B y C, con lo que se obtienen los pixeles comunes a las dos imágenes, es decir los pixeles bien clasificados. Negando una de las dos a la vez, se obtienen los falsos positivos, mostrados en la imagen E y los falsos negativos, mostrados en la imagen F.

Los casos de aciertos, falsos positivos y falsos negativos, están dados por las siguientes ecuaciones:

$$\text{Aciertos}=\boldsymbol{D = B \cap C}$$
$$\text{Falsos Positivos}=\boldsymbol{E = \sim B \cap C}$$
$$\text{Falsos Negativos}=\boldsymbol{F = B \cap \sim C}$$

Como factor de normalización de las áreas en pixeles, se toma la cantidad de pixeles de primer plano de la máscara hecha a mano. A partir de esta se comparan las imágenes de aciertos, falsos positivos y falsos negativos. Los resultados normalizados al 100% se muestran a continuación, primero en la Tabla 4 y luego en la Figura 12, la Figura 13 y la Figura 14.

Tabla 4 Resultados pruebas de algoritmos de primer plano.

algoritmo	% acierto	% falsos positivos	% falsos negativos	% error total
Codebook HSV alternantes	54,8	8,78	45,2	53,98
Codebook tradicional	43,22	18,15	56,78	74,93
FGDStatmodel	74,87	26,42	25,13	51,54
media normalizada	32,65	4,62	67,35	71,97
media normalizada umbral	32,98	5,37	67,02	72,39
media umbral movimiento	67,59	33,32	32,41	65,73
Codebook hsv alternante umbral movimiento	60,99	14,98	39,01	53,98

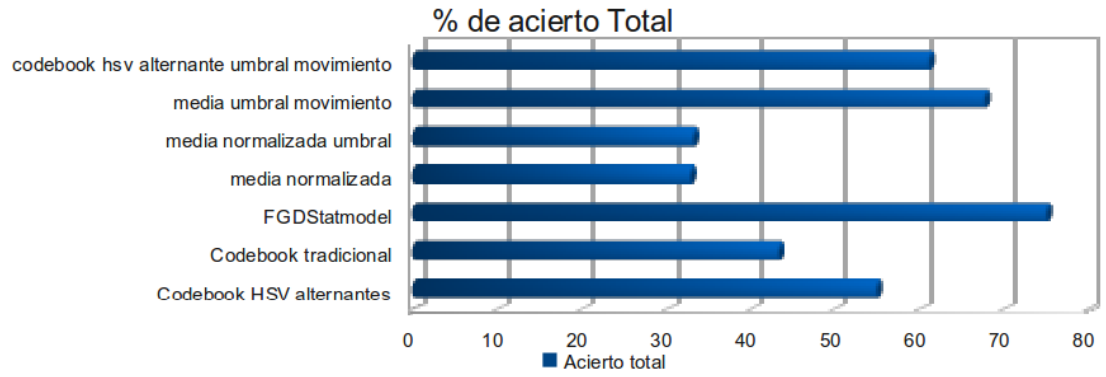

Figura 12 % de acierto algoritmos de fondo probados.

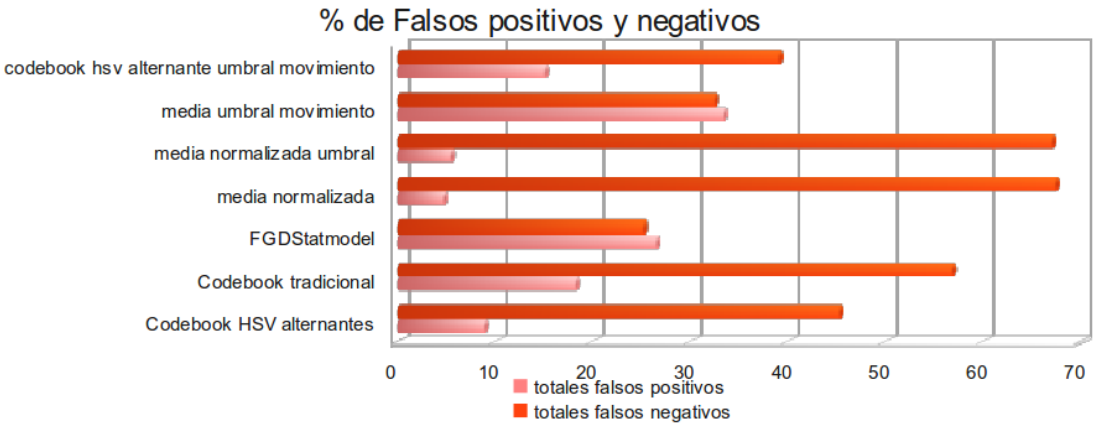

Figura 13. % de falsos positivos y negativos algoritmos probados.

Figura 14. % de error total sumando falsos positivos y negativos.

De estas pruebas se seleccionaron 3 algoritmos de estimación del primer plano para la detección de los vehículos, el algoritmo de *FGDStatmodel*, [15], [16]. El algoritmo de *Codebook* sobre la imagen en componentes HSV y actualizando el modelo mediante la técnica de umbral de movimiento, [12], [17]. Y el algoritmo de acumulación de media usando la técnica de umbral de movimiento para la actualización[13]. Descritos en el anexo de estimación de primer plano y en el marco teórico.

La salida de este primer bloque de acumulación de líneas, de color verde en la Figura 9. Consiste en dos imágenes binarias sin limpiar del primer plano, capturado por líneas cada N cuadros como se muestra en la Figura 15, este proceso se repite cada vez que se termina de acumular. En la implementación del algoritmo, este bloque se ejecuta en una tarea independiente que es llamada una vez por cada cuadro de video.

Figura 15. Ejemplo de imagen acumulada de primer plano sin limpiar.

5.6 Algoritmo de análisis de bloques

A cada imagen binaria de la Figura 15 le es extraída los contornos, usando el algoritmo de extracción de contornos basado en [36], presente en las bibliotecas de OpenCV. Son extraídos los contornos más externos de las dos imágenes, a estos se les realiza un análisis por área y perímetro para determinar si pueden ser unidos o no, donde estos parámetros dependen directamente del tamaño de la imagen. Unos valores típicos son 100 pixeles cuadrados de área y 30 pixeles de perímetro, para una imagen de 640 x 480 y la mitad para una imagen de 320 x 240. A partir de los contornos, se generan las estructuras de características que identifican a un posible vehículo detectado, que son llamados bloques en este documento.

Cada bloque tiene las siguientes características: el contorno, el rectángulo contenedor, las coordenadas en las cuales fue detectado y dos banderas lógicas explicadas más adelante.

Estos bloques son analizados uno a uno. Son agrupados en nuevos bloques, dependiendo si cumplen ciertas características geométricas sobre los rectángulos contenedores, mostradas en la Figura 16 y descritos por las siguientes ecuaciones:

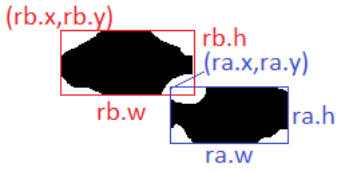

Figura 16. Agrupación de bloques

$$
\begin{aligned}
pa.x &= ra.x + ra.w * 0.5 \\
pa.y &= ra.y + ra.h * 0.5 \\
pb.x &= rb.x + rb.w * 0.5 \\
pb.y &= rb.y + rb.h * 0.5 \\
f\_w &= (ra.w + rb.w) * 0.5 \\
f\_h &= (ra.h + rb.h) * 0.5 \\
dx &= ((|pa.x - pb.x|) - f\_w) \\
dy &= ((|pa.y - pb.y|) - f\_h) \\
f_{wt} &= -0.2 * f\_w \\
f_{ht} &= -0.2 * f\_h
\end{aligned}
$$

Los bloques se unen como uno solo si se cumplen las condiciones:

$$(dx < f_{wt}\ AND\ dy < f_{ht})$$

Si dx es negativo, es porque los bloques están unidos en x; de igual forma, si dy es negativo, es porque los bloques están unidos en y. Cuanto más negativo los valores, más cerca están en cada una de las coordenadas. El criterio de f_{wt} y f_{ht}, de 0.2 se basa en la experimentación y análisis de los bloques como un parámetro de distancia que depende de que tanto se desee unir a los bloques cercanos[13].

Estos bloques se generan en dos listas independientes, una por cada una de las Imágenes de acumulación. Cada uno de estos bloques cuenta con un identificador de la posición del bloque, que consiste en el rectángulo contenedor del contorno y el número de la imagen de acumulación M en la que se encontró. También se registra en una bandera lógica, si el bloque hace parte de los extremos superior o inferior de la imagen de acumulación de primer plano, esto con el fin de poder unirlo a otro bloque posteriormente, previendo la posibilidad de que la separación de N líneas, corte la aparición de uno o más vehículos; la Figura 18 muestra un ejemplo de este caso.

Figura 17. Ejemplo de salida de características de bloques

La salida de este algoritmo consiste en dos secuencias, una por cada línea de captura, en las que son almacenados bloques filtrados y agrupados correspondientes a cada línea de captura.

Una forma gráfica de la salida de este algoritmo puede verse en la Figura 17, teniendo en cuenta que cada bloque de características corresponde a los datos adquiridos de la imagen y no a una imagen.

En la implementación del algoritmo, este proceso es realizado junto con el tratado en la siguiente sección cada vez que es encontrada una imagen de acumulación, es decir cada N cuadros.

5.7 Algoritmo de Emparejamiento de secuencias

En este punto se cuenta con la información procesada de la ubicación de cada vehículo en cada imagen acumulada del video y es necesario analizarla para encontrar los parámetros de tráfico.

5.7.1 Unión de bloques extremos

De las dos secuencias de estructuras de características también llamadas bloques, se aplica primero un algoritmo de unión de bloques extremos. Lo que es decir, bloques en los que se da el caso de que la acumulación cada N líneas corta la aparición de uno o más vehículos como se muestra en la Figura 18. La unión de estos bloques es realizada con los mismos criterios ya expuestos en la sección 5.6 en el algoritmo de análisis de bloques.

Figura 18. Ejemplo de separación de bloques por acumulación cada N líneas

5.7.2 Predicción

Se cuenta con dos secuencias de bloques, una por cada línea de acumulación, el ancho de los bloques depende del vehículo que representa. Este puede ser, pequeño en el caso de motos, normal, en el caso de vehículos ligeros (generalmente automóviles y camionetas en las vías urbanas analizadas) y grande en el caso de vehículos pesados (Por lo general buses, en las vías urbanas analizadas), este ancho en pixeles depende solamente de la altura a la cual fue ubicada la cámara y de la resolución a la cual se esté capturando. Sin embargo, el largo de los objetos en las imágenes de acumulación, depende de la tasa de captura del video y de la velocidad del vehículo.

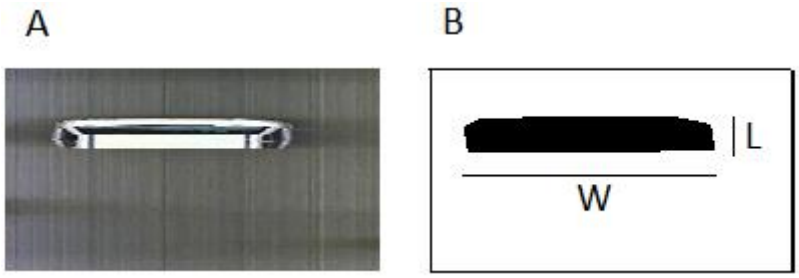

Figura 19. Ancho W y Largo L de los bloques en las imágenes acumuladas

Habiendo calibrado la cámara y teniendo la relación entre pixeles en la imagen y metros en la vía, se plantean las siguientes ecuaciones, todas usando como dimensionalidad espacial los pixeles en la imagen.

Primero, de cada imagen acumulada, se puede encontrar la velocidad del objeto de la siguiente manera

$$V = \frac{L_v}{L * T_s}$$

Donde V es la velocidad del objeto en pixeles por segundo, L_v es el largo real en pixeles del objeto, L es el largo del objeto en la imagen acumulada y T_s es el periodo de muestreo de la cámara en hercios Hz. Debido a que solo se está procesando el video por líneas, no se cuenta con el tamaño real T de cada objeto que pasa por la vía, por lo que es necesario estimarlo a partir del ancho del objeto W, que permanece igual en la imagen de video y en la acumulada por líneas.

Esta estimación se hace por medio de la clasificación del ancho W en tres tamaños posibles. El Primero para motos. El segundo, para Automóviles y camionetas. El tercero para buses y camiones. Es asignado un largo estimado del vehículo, dependiendo a que clase pertenezca.

De la primera etapa del algoritmo, Bloque de análisis de líneas, en verde en la Figura 9. Se capturaron dos líneas sobre el cuadro de video estas estaban separadas una distancia J en pixeles, de esta distancia es posible encontrar una expresión para el número de cuadros que pasaron, o el número de líneas a los que se espera encontrar el objeto notado por m:

$$m = \frac{\frac{J}{V}}{T_s} = \frac{\frac{J}{\frac{T_{Estimado}}{L*T_s}}}{T_s} = \frac{J * L}{T_{Estimado}}$$

Esta expresión debe calcularse sobre los bloques de la línea por la que pasan primero los vehículos. Este m representa una predicción de en donde se encontrará el objeto, en la imagen acumulada por la otra línea a partir de los bloques predicho de la línea 1 u de los bloques de la línea 2 parte el algoritmo de emparejamiento descrito en la siguiente sección.

5.7.3 Emparejamiento

A partir de la predicción hecha, sobre en donde se encontraran los objetos representados por los bloques de la línea por la que primero pasan los vehículos, se encuentran todas las posibles distancias euclidianas, desde el punto superior central del rectángulo contenedor del bloque predicho, hasta puntos superiores centrales de los rectángulos contenedores de la secuencia bloques de la otra línea. Se seleccionan los elementos que tengan los menores valores, para asignarlos como parejas válidas, siempre y cuando no se supere una distancia máxima, dada por la mínima velocidad que pueda esperarse en la vía, esto se realiza para todos los bloques que no estén marcados como extremos. Este proceso de emparejamiento se realiza Q veces, donde Q está dado por el menor número de bloques no marcados como extremos, que se presenten en cada una de las dos secuencias de bloques. Los bloques que no son emparejados después de Q iteraciones son marcados para analizar la siguiente vez que se aplique el algoritmo.

Se escogen los puntos superiores centrales ya que estos presentan una mejor estimación de la velocidad, siempre y cuando el video sea tomado siguiendo las indicaciones presentadas en la sección 5.3.

5.8 Estimación de los parámetros de tráfico

En este punto del algoritmo se procede a examinar los bloques emparejados, cada una de estas parejas de bloques representa un posible vehículo que pasó por ambas líneas de acumulación. Sin embargo, para que sea contado como vehículo y extraída su velocidad debe pasar por un segundo análisis, el cual se describe a continuación.

5.8.1 Estimación de la velocidad

Si se cumple la condición de ubicar las líneas de acumulación sobre cada sentido de carril, en el orden propuesto en la Figura 10. La distancia entre los bloques siempre va a ser positiva, ya que un vehículo debe pasar primero por la línea uno, de color amarillo y luego por la línea dos, de color verde. Si se da el caso de encontrar una distancia negativa, esta se elimina como un error de emparejamiento.

Debido al muestreo sobre los objetos, causado por la acumulación de líneas, existe una incertidumbre en la estimación de la velocidad de más o menos una muestra, dada por la distancia *e* recorrida por el objeto a velocidad *v* en T_s segundos como puede verse en la Figura 20.

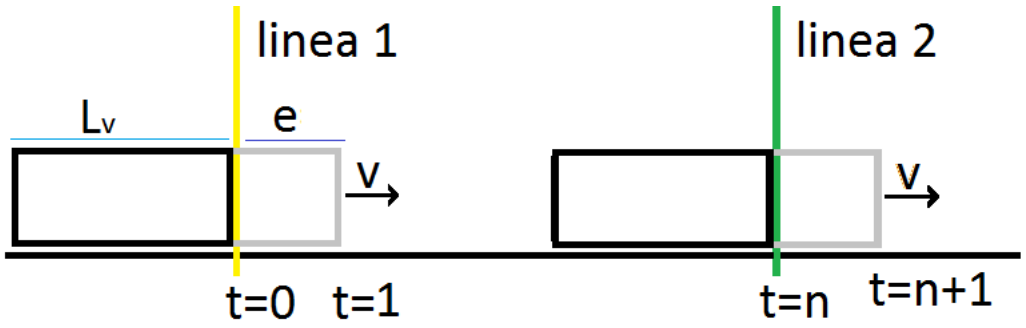

Figura 20. Bloque a velocidad "v" acumulado en t=0, 1, n, n-1

La velocidad real del objeto *v* puede estar dada en metros o pixeles por segundo y de esta depende la magnitud de la distancia *e*, que está dada por:

$$e = vT_s$$

El ancho de los bloques L en pixeles, resultado de la acumulación de líneas, está dado por:

$$L = \left[\!\left[\frac{L_v}{T_s v}\right]\!\right]$$

Donde L_v es el largo del vehículo, de igual manera puede encontrarse una ecuación similar para el número de muestras de diferencia entre la acumulación de la primera línea con respecto a la segunda, n:

$$n = \left[\left[\frac{J}{T_s v}\right]\right]$$

En la Figura 21, se muestra el número de muestras ya sea para el ancho de los bloques L como para la distancia entre bloques n, en una separación de líneas J. Para distintas tasas de captura y distintas velocidades de los vehículos, sobre un objeto de un metro de largo o de manera equivalente, para una distancia entre líneas de un metro. La gráfica puede usarse como medida normalizada para encontrar la misma acumulación sobre objetos de distinto tamaño o líneas con una separación diferente. En el anexo de tablas se encuentra una tabulación de esta gráfica, útil para el diseño del sistema.

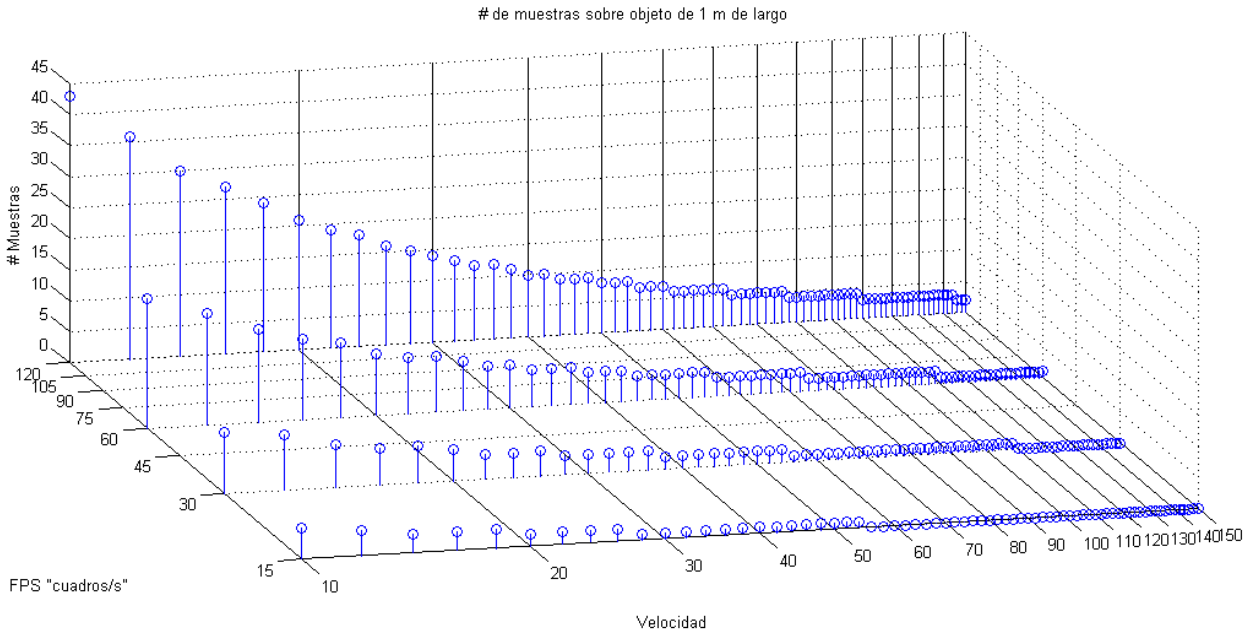

Figura 21. Número de muestras contra velocidad en Km/h para varias tasas de captura

Para estimar la velocidad del objeto V, usando dos líneas de acumulación, como la velocidad real v será proporcional al número de líneas entre los objetos emparejados ver en la Figura 10 las líneas azules, entonces la estimación de la velocidad se realiza encontrando la diferencia entre los puntos superiores centrales de cada pareja, así:

$$V = \frac{J}{(n)T_s}$$

Donde V es la velocidad en pixeles por segundo, n la diferencia en pixeles entre los dos puntos superiores centrales de los rectángulos contenedores, J la distancia de separación de las líneas de captura y T_s el periodo de muestreo de la cámara. Para encontrar un equivalente en metros por segundo basta usar los parámetros intrínsecos de la cámara calibrada y convertir esta medida en metros por segundo o kilómetros por hora.

5.8.2 Conteo de los vehículos o intensidad horaria

Cada pareja de bloques con una velocidad mayor a cero se convierte en un vehículo detectado y contado. Por lo que el conteo de vehículos depende del algoritmo de estimación de velocidad.

5.8.3 Cálculo de la densidad vehicular

Después de M intervalos de N acumulaciones es encontrada la densidad vehicular, usando la ecuación fundamental de tráfico vehicular. A partir de la Intensidad horaria y la velocidad promedio.

5.9 Eliminación de errores

Los bloques que no fueron emparejados después de 3 periodos de acumulación de N cuadros, se consideran como errores o casos aislados de detección. Por ejemplo, peatones que pasen por una sola de las líneas o vehículos que pasen en sentido contrario, estos son eliminados sin afectar el conteo ni la estimación de velocidad.

5.10 Archivos de resultados

El usuario puede escoger un número M de intervalos de N acumulaciones, en los que desee generar un archivo separado por comas, de nombre "variables de tráfico M.csv". Donde M va variando con respecto a la cantidad de archivos generados. El tipo de archivo seleccionado para entregar los resultados, presenta la extensión .csv, que es compatible con Open officeCalc, Microsoft Excel y Google Spreadsheet. El archivo contiene una tabla con la velocidad encontrada por vehículo y al final un estimado de la velocidad promedio en kilómetros por hora, el volumen en vehículos por hora y la densidad en vehículos por kilometro. Dado el caso que la partición de M sea equivalente a una hora, el conteo de vehículos pasa por definición de ser intensidad horaria a Volumen vehicular. Este archivo puede ser visualizado de manera remota usando las diferentes herramientas que provee el sistema operativo como por ejemplo VNC [37] o SAMBA [38] .

Tabla 5. Ejemplo de salida del sistema

# del vehículo	Velocidad Km/h
1	85
…	…
Velocidad Promedio Km/h	60
Volumen Veh/h	1200
Densidad Veh/Km	20
Tiempo de ejecución horas	0.3

5.11 Implementación del algoritmo

La implementación de algoritmo fue hecha para poder ser procesada en paralelo. En la Figura 9, El bloque de análisis de líneas, en verde, debe ejecutarse cada cuadro de video; este realiza la acumulación de N líneas por lo que cada N cuadros de video se ejecutan los demás bloques en azul y aguamarina. La implementación de este algoritmo usa técnicas de diseño de software para sistemas operativos en tiempo real, "s*oft real time",* para cumplir la tarea de mayor prioridad al mismo tiempo de las otras sin que se vea disminuido el rendimiento del sistema.

6 Prueba de los algoritmos propuestos

6.1.1 Tablas de toma de los videos de pruebas

En la sección 5.5.2 se presenta la primera prueba sobre varios algoritmos de detección detallados en el anexo de estimación del primer plano. De estos se tomaron 3 algoritmos que presentaban el menor número de falsos positivos y negativos junto con la mayor cantidad de aciertos.

Tabla 6. Algoritmos de detección probados

Algoritmo 1	Usando el algoritmo de *FGDStatmodel*
Algoritmo 2	Usando dos modelos de *Codebook* alternantes en el espacio HSV
Algoritmo 3	Usando acumulación de la media con umbral de movimiento en la acumulación y umbral adaptativo en la diferencia.

Se implementó tres versiones distintas del algoritmo de estimación de parámetros de tráfico, visto en la Figura 9, cambiando únicamente el algoritmo de detección por uno de los 3 listados. Estos algoritmos se probaron en los cuatro videos tomados teniendo en cuenta las consideraciones listadas en la sección 5.3.

Tabla 7. Características de los archivos de video de prueba

Nombre del video	Duración "s"	Tasa "cuadros/s"	Tamaño	Velocidad de datos kbps
Video 1	180	125	320x240	4915
Video 2	180	30	640x480	19660
Video 3	180	125	320x240	4915
Video 4	180	30	640x480	19660

En la Tabla 7 se listan las características de los archivos de prueba, para todos los videos se usó como códec XVID y como contenedor avi; fueron grabados con la cámara *PlayStationEYE*[21], usando las configuraciones mostradas en la Tabla 8 del módulo controlador de la cámara, en el núcleo de GNU/Linux.

Tabla 8. Configuración del modulo controlador de la cámara *PlayStationEYE*.

parámetros del modulo gspca-ov534	30 cuadros/s	125 cuadros/s
videomode	"01"	"16"
autogain	0	0
gain	0	0
awb	0	0
exposure	128	128
brightness	35	35
contrast	35	35
redlc	128	128
bluelc	128	128
hue	128	128
sharpness	0	0
vflip	0	0
hflip	0	0
freqfltr	1	1

Se removió toda compensación automática para evitar el efecto de cambio de brillo y cambio de tonalidad en la imagen, cuando pasa algún vehículo blanco o de colores luminosos. Estos ajustes automáticos generan errores en algoritmo de detección [14].

Las características de la vía y los parámetros de tráfico en los videos tomados se listan en la Tabla 9.

Tabla 9. Características de la toma de los videos de prueba.

Nombre del video	Altura "m"	Carriles	Conteo	Velocidad media Km/h
Video 1	6,5	2	56	39,2
Video 2	6,5	2	64	40,5
Video 3	8	3	86	30,2
Video 4	8	3	94	28

6.1.2 Medición de las variables de tráfico

Sobre los cuatro videos se escogieron dos ubicaciones de líneas distintas. La primera, cercanas y ubicadas teniendo en cuenta las consideraciones presentadas en la sección 5.3, para la estimación correcta de velocidad. La segunda, usando todo el largo de la imagen separando las líneas la mayor distancia posible. En la Tabla 10, se listan los puntos de inicio y fin para las dos líneas de acumulación ubicadas en la imagen. Como se cuenta con dos tasas de captura distintas, 30 y 125 cuadros por segundo, los parámetros de calibración en tiempo para el primer algoritmo fueron adecuados proporcionalmente al incremento en la tasa, es decir fueron multiplicados por 4,16

Tabla 10. Puntos de inicio "a" y final "b" de las líneas 1 y 2.

líneas	video1	video2	video3	video4
Cercanas y en la parte superior	pta1 (3,78) pta2 (315,78)	pta1 (4,100) pta2 (635,100)	pta1 (3,74) pta2 (317,74)	pta1 (3,13) pta2 (637,13)
	ptb1 (3,158) ptb2 (315,158)	ptb1 (4,256) ptb2 (635,256)	ptb1 (3,146) ptb2 (317,146)	ptb1 (3,127) ptb2 (637,127)
separadas	pta1 (1,2) pta2 (318,1)	pta1 (1,2) pta2 (639,2)	pta1 (1,4) pta2 (316,4)	pta1 (3,5) pta2 (637,5)
	ptb1 (1,211) ptb2 (318,211)	ptb1 (1,440) ptb2 (639,440)	ptb1 (1,204) ptb2 (316,204)	ptb1 (3,413) ptb2 (637,413)

La distancia entre líneas, en pixeles y metros, se muestra en la Tabla 11.

Tabla 11. Distancia entre líneas de captura.

distancia entre líneas		video 1	video 2	video 3	video 4
cercanas	pixeles	80	156	72	114
	metros	1,342524	1,30896	1,487104	1,17728
separadas	pixeles	209	438	200	408
	metros	3,507345	3,67516	4,130844	4,21342

De la tasa de muestreo de la cámara, la separación de las líneas y el tamaño mínimo de los vehículos a detectar, pueden encontrarse las velocidades máximas detectables por el algoritmo. Estas se listan en la Tabla 12.

Tabla 12. Velocidades máximas detectables en los videos de pruebas.

Máximos detectables		video 1	video 2	video 3	video 4
Cuadros por	Segundo	125	30	125	30
cercanas	velocidad 1 Km/h	604,136	141,368	669,1968	127,146
separadas	velocidad 1 Km/h	7628,089	1870,38	9951,653	1928,59
ambos casos	velocidad 2 Km/h	1659	398,16	1659	398,16

La velocidad 1, es la velocidad a la cual pasa el objeto y es detectado por un solo pixel de diferencia en las dos imágenes acumuladas.

La velocidad 2, es la máxima velocidad a la que puede ir un objeto de un tamaño de 4,2 metros de largo y ser detectado por cada línea de acumulación, en lo menos una línea.

La máxima velocidad detectable por video está dada por la menor entre las dos velocidades anteriores.

El resultado del conteo total de los tres algoritmos se lista en la Tabla 13 y la Tabla 14.

Tabla 13. Conteo sobre líneas cercanas

Líneas cercanas							
	conteo	Algoritmo 1		Algoritmo 2		Algoritmo 3	
#video	Manual	conteo	%	conteo	%	conteo	%
Video 1	56	56	100	52	92,86	38	67,85
Video 2	64	62	96,875	53	82,81	50	78,12
Video 3	86	85	98,83	83	96,51	63	73,3
Video 4	94	93	98,93	91	96,80	65	69,15
TOTAL	300	296	98,7	279	93	216	72

Tabla 14. Conteo sobre las líneas separadas

Líneas separadas							
	conteo	Algoritmo 1		Algoritmo 2		Algoritmo 3	
#video	Manual	conteo	%	conteo	%	conteo	%
Video 1	56	53	94,64	43	76,78	20	35,7
Video 2	64	62	96,875	44	68,75	18	28,125
Video 3	86	75	87,2	52	60,46	53	61,6
Video 4	94	80	85,1	65	69,14	72	76,6
TOTAL	300	270	90	204	68	163	54,34

Para los 3 algoritmos existe un tiempo de establecimiento, el cual está dado por las variables de aprendizaje del algoritmo y la tasa de captura del video. En el caso del primer algoritmo, este es despreciable, ya que solo incluye unos pocos cuadros gracias a la posibilidad de establecer la convergencia inicial. Sin embargo, en los algoritmos dos y tres la convergencia inicial está fija en los primeros 300 cuadros, durante este tiempo no se estima el primer plano, por lo que los vehículos que pasan en este intervalo de tiempo no son contados, así que el conteo de los vehículos en condiciones estables del algoritmo, se encuentra al restar del total, los vehículos que pasaron en este intervalo de estabilización y se lista en la Tabla 15 y la Tabla 16.

Tabla 15. Conteo sobre líneas cercanas bajo condiciones estables.

Líneas cercanas								
Conteo manual menos tiempo de convergencia para el algoritmo 2 y 3								
		Conteo -	Algoritmo 1		Algoritmo 2		Algoritmo 3	
#video	Conteo	convergencia	conteo	%	conteo	%	conteo	%
Video 1	56	56	56	100	52	92,86	38	67,86
Video 2	64	61	62	96,875	53	86,9	50	81,9
Video 3	86	84	85	98,83	83	98,8	63	75
Video 4	94	90	93	98,93	91	101,1	65	72,2
TOTAL	300	291	296	98,66	279	95,88	216	74,2

Tabla 16. Conteo sobre líneas separadas bajo condiciones estables.

Líneas separadas								
Conteo manual menos tiempo de convergencia para el algoritmo 2 y 3								
		Conteo -	Algoritmo 1		Algoritmo 2		Algoritmo 3	
#video	Conteo	convergencia	conteo	%	conteo	%	conteo	%
Video 1	56	56	53	94,64	43	76,79	20	35,7
Video 2	64	61	62	96,88	44	72,2	18	29,5
Video 3	86	84	75	87,2	52	61,9	53	63,1
Video 4	94	90	80	85,1	65	72,2	72	80
TOTAL	300	291	270	90	204	70,1	163	56

La medición de velocidad se realizó comparándola con la obtenida manualmente sobre los videos, tomando el desplazamiento en pixeles del vehículo en un tiempo determinado por la tasa de captura y convirtiendo estos datos a metros por segundo usando los parámetros intrínsecos de la cámara. Este procedimiento se realizó con todos los vehículos presentes en los 4 videos de pruebas. Las medidas de velocidad están expresadas en Kilómetros por hora y se listan en la Tabla 17 y la Tabla 18.

Para el algoritmo 1 y 2, los errores consisten en solo falsos negativos, para el algoritmo 3, se presentan casos de falsos positivos y negativos, un análisis más detallado de el rendimiento por vehículo puede encontrarse en el anexo de tablas en la carpeta de Tablas resultados algoritmos.

Tabla 17. Velocidad media sobre líneas cercanas estimada por los algoritmos.

Líneas cercanas							
Km/h	Velocidad media	Algoritmo 1		Algoritmo 2		Algoritmo 3	
#video	manual	Vel media	%	Vel media	%	Vel media	%
Video 1	38,59	36,7	95,09	40,8	105,71	47,1	122,04
Video 2	43,72	42,7	97,65	35,3	80,73	48,4	110,69
Video 3	24,34	24,5	100,65	25,94	106,57	25,9	106,40
Video 4	25,25	26,5	104,94	26,98	106,84	39	154,45
TOTALES	% de acierto total		99,58		99,96		123,39
	desviación estándar		4,23		12,83		21,72

Tabla 18. Velocidad media sobre líneas separadas estimada por los algoritmos.

Líneas separadas							
Km/h	Velocidad media	Algoritmo 1		Algoritmo 2		Algoritmo 3	
#video	manual	Vel media	%	Vel media	%	Vel media	%
Video 1	38,59	39,2	101,57	41,4285714	107,34	34,7	89,91
Video 2	43,72	40,5	92,62	39,13	89,49	34,5	78,9
Video 3	24,34	30,2	124,07	24,34	99,99	19,5	80,11
Video 4	25,25	28	110,88	28,8	114,05	30,2	119,6
TOTALES	% de acierto total		107,28		102,72		92,13
	desviación estándar		13,44		10,52		18,96

6.1.3 Tiempos de procesamiento sobre las arquitecturas propuestas

Los tiempos de procesamiento se midieron sobre el algoritmo de análisis de líneas, ya que las demás etapas del algoritmo se implementaron sobre una tarea de proceso aparte y solo se ejecutan cada N líneas mientras el procesador se encuentra desocupado, por lo que no afecta al rendimiento en general del algoritmo. En la Tabla 19, se listan los tiempos de ejecución por cuadro, en microsegundos. Que fueron medidos sobre las plataformas de hardware detalladas en la Tabla 2.

Tabla 19. Tiempos de ejecución sobre las 3 plataformas probadas

tiempo de compilación "us"	Algoritmo 1		Algoritmo 2		Algoritmo 3	
Tamaño de imagen	640x480	320x240	640x480	320x240	640x480	320x240
Corei7	2198,265055	1910,037399	742,8588998	719,269149	1029,081395	922,9280577
Atom	14091,78689	12909,36955	2182,228583	2036,5988	6614,960906	6597,883474
OMAP	51855,96172	47497,47513	7962,764784	8100,29081	23003,51072	22298,78036

La frecuencia en Hz, a la cual se ejecuta el algoritmo, puede encontrarse invirtiendo los valores en microsegundos de la Tabla 19. Si se está usando video grabado, estos valores listados se les debe sumar tiempo que le toma a la plataforma decodificar el archivo, tiempo que depende de la implementación del algoritmo de decodificación y del códec usado. Se lista la tasa de ejecución del algoritmo usando la cámara de video *PlayStationEYE* configurada con la máxima tasa de captura limitada a 125 cuadros por segundo, a una resolución de 320x240, los resultados de esta prueba de muestran en la Tabla 20.

Tabla 20. Tasa de ejecución máxima de los algoritmos en distintas plataformas.

cuadros por segundo	Algoritmo 1	Algoritmo 2	Algoritmo 3
Corei7	125	125	125
Atom	65	125	119
OMAP	10	21	14

7 Análisis de resultados

7.1 Detección del primer plano

De la primera prueba, para la selección del algoritmo de primer plano usado en la detección, se seleccionaron 3 algoritmos de detección para probar junto con el resto del algoritmo de estimación de parámetros de tráfico, esto, porque la estimación por líneas hace que el algoritmo cuente con menos información y por ende los errores en la estimación del primer plano son más notorios. Este es el resultado de reducir área de la imagen de máximo 320 o 640 pixeles a una sola línea. Además, si los errores son recurrentes en un pixel se generan líneas verticales en la imagen acumulada de primer plano como en la Figura 22, que unen los contornos en el tiempo, generando problemas en la detección.

Un algoritmo de primer plano por media, presenta este tipo de errores con frecuencia y es necesario complementarlo usando un criterio en la acumulación. De las pruebas sobre los algoritmos de detección, se encontró que es mejor acumular cuando los vehículos se encuentran en movimiento, ya que si un vehículo permanece estático mucho tiempo, Figura 24, la media del fondo se ve afectada y varia, después cuando el vehículo se retira el modelo de fondo cambia con respecto al fondo real y se generan estas líneas verticales.

Figura 22. Primer plano con error en la estimación por pixel recurrente en el tiempo.

Además, los algoritmos escogidos fueron los que presentaron mejor rechazo a las sombras de los vehículos en especial los algoritmos 1 y 2, usando *FGDStatmodel* y *codebook* en el espacio de color HSV y con detección de movimiento para la actualización. Una muestra de esta detección del primer plano puede verse en la Figura 23. El algoritmo 3, en la Figura 23D, fue seleccionado como comparación de los otros dos algoritmos para una mejor referencia.

El algoritmo 1 en la Figura 23B, tiene una cantidad alta de falsos positivos, pero también un mayor porcentaje de aciertos. El algoritmo 2 en la Figura 23C, tiene un porcentaje más bajo de falsos positivos, pero falla en la detección de algunos vehículos, en especial los de color blanco y negro.

Figura 23. Detección del primer plano en los 3 algoritmos seleccionados.

7.2 Resultados en el conteo de vehículos

En los tres algoritmos probados, la eficiencia en el conteo disminuyó con forme se alejaban las líneas de conteo, este efecto fue más notorio en los videos tres y cuatro, que fueron tomados en una zona cercana a un semáforo, en el que los vehículos quedan detenidos. Dependiendo de la zona del video en el que se detienen la acumulación de líneas puede verse o no afectada, como se muestra en la Figura 24 en la imagen A, en la que el vehículo pasó la primera línea antes de detenerse por el semáforo y fue acumulado en su totalidad, pero no pasó totalmente la línea 2 y siguió siendo acumulado durante todo el tiempo que duró detenido. Esto genera diferencias importantes en la detección a ser analizada por el algoritmo de emparejamiento de bloques, este caso en especial no genera una pareja válida, ya que la velocidad predicha de un bloque de la línea uno, no corresponde a la presente en el bloque correspondiente de la línea dos.

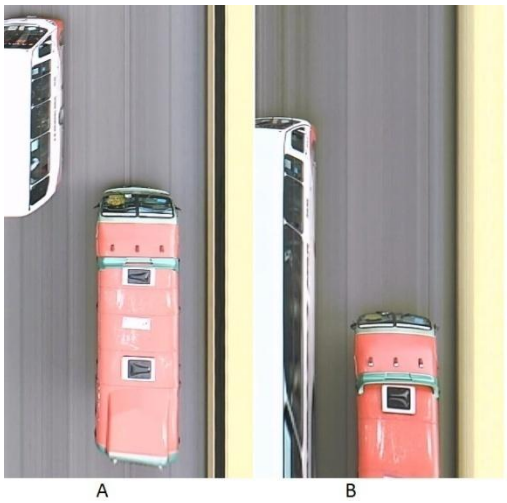

Figura 24. Acumulación con líneas separadas

Si las líneas están demasiado separadas y el objeto acelera o frena, el bloque que produce en cada línea, va a ser de un tamaño diferente, provocando al igual que en caso anterior, errores en la predicción y un error por falso negativo en el emparejamiento.

El alto error presente en el algoritmo 3 en el caso de líneas separadas, se da por las diferencias en la iluminación entre las dos líneas, en la Figura 25, las imágenes B y D muestran las imágenes de acumulación de las líneas 1 y 2, en la línea 1, el primer plano encontrado por el algoritmo no presenta errores que afecten la detección como se ve en la imagen A. Sin embargo en la línea 2, el bajo contraste en la imagen D hace que el umbral encontrado por el algoritmo implementado con este fin [16], no sea lo suficiente para segmentar la imagen y se presenta el error mostrado en la imagen C.

Comparando los resultados mostrados en la Tabla 15 con los presentados en la Tabla 16. Para los algoritmos 1 y 2, se aprecia que mientras más cercanas se encuentren las líneas mejor es la estimación del conteo, esto porque existe una mayor similitud entre las imágenes acumuladas por líneas, por lo que el emparejamiento es más sencillo y tendrá menos errores.

El algoritmo 2 tuvo un menor desempeño en el conteo debido a que es más selectivo en la estimación del primer plano, la razón por la cual los vehículos no se contaron fue porque no se detectaron en una o en ambas imágenes de acumulación.

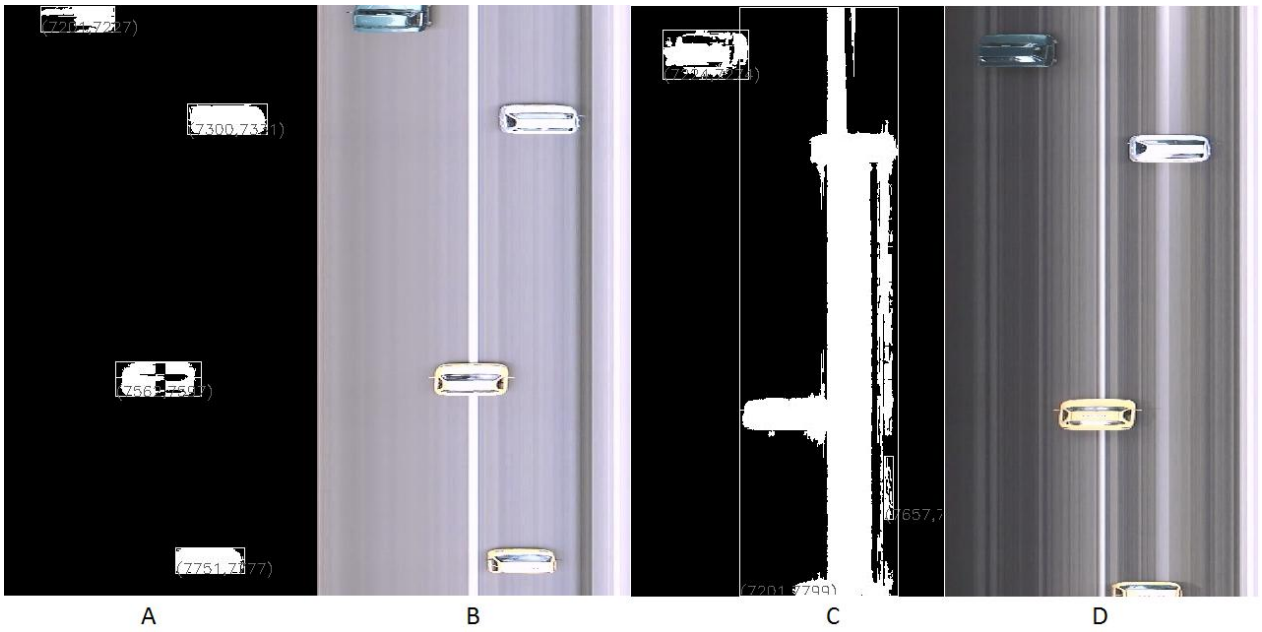

Figura 25. Diferencia en la estimación del primer plano usando el algoritmo de media por movimiento.

Un análisis de los resultados por vehículo de la ejecución de los tres algoritmos de estimación de parámetros de tráfico, muestra que en los algoritmos 1 y 2 no se presentan casos de falsos positivos en el conteo, solo falsos negativos causados por la no detección de un vehículo por el algoritmo de primer plano. En el algoritmo 3 si se presentaron casos de falsos positivos, debido a la detección de sombras como objetos móviles y la separación de bloques que corresponden a un solo vehículo.

El emparejamiento de los bloques hace más difícil que los errores en la estimación de primer plano o los objetos que pasaron por una sola de las líneas sean contados como un vehículo, por ejemplo, en la Figura 25 solo se contó un objeto, ya que el algoritmo de emparejamiento descartó los demás bloques encontrados.

En la Figura 26, la imagen A y B son imágenes resultado de la acumulación de las líneas 1 y 2, en la primera pasó un peatón pero en la segunda no, este peatón fue detectado por el algoritmo de estimación de primer plano, “imágenes C y D”, pero este se descartó, ya que no se encontró un equivalente en la imagen de acumulación de la segunda línea.

También en las imágenes C y D de la Figura 26, puede verse como el algoritmo 2 de estimación del primer plano, descartó las sombras de los vehículos y del peatón, como primer plano. La imagen acumulada del peatón tiene un tamaño similar a la generada por los vehículos ya que este se mueve a una menor velocidad.

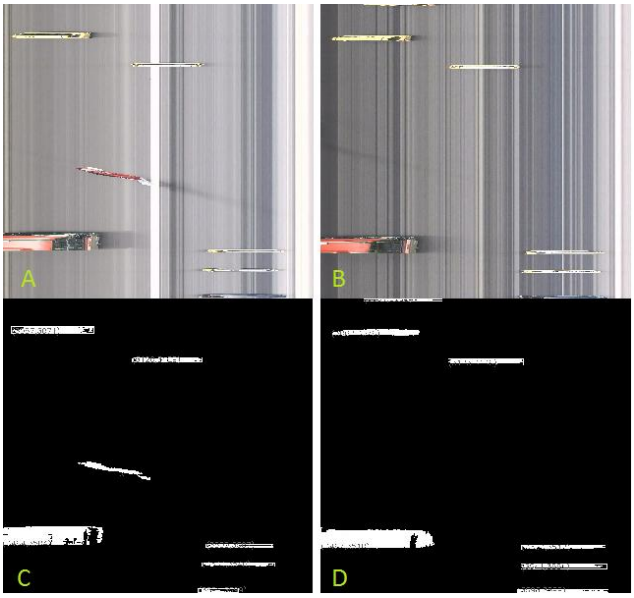

Figura 26. Peatón transitando por la vía y no detección de sombras del algoritmo 2.

7.3 Medición de la velocidad y relación con la tasa de captura

Los resultados de la medición de la velocidad media hecha por los algoritmos se encuentran en la Tabla 17, para el caso de líneas de acumulación cercanas y en la Tabla 18, para líneas separadas. Del análisis de estas tablas, se puede ver que la estimación de la velocidad se ve afectada por cómo se mide la velocidad sobre el vehículo. Los resultados con las líneas separadas reportan mayor velocidad debido a que la línea 2 se encuentra en un punto en el que la perspectiva de la imagen aumenta el recorrido del vehículo como se expuso en la sección 5.3. Es por esto que los errores cuando las líneas están más separadas tienden a reportar mayor velocidad que la real.

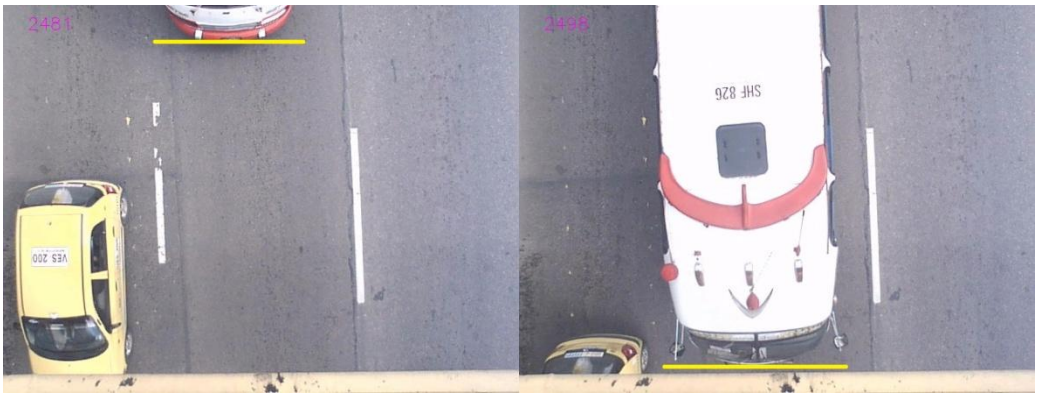

Figura 27. Como la ubicación de la línea afecta la medición de la velocidad.

Los algoritmos 1 y 2 tuvieron un alto acierto en la estimación de la velocidad media, el algoritmo 3 presentó resultados superiores a la velocidad media real, esto porque se veía afectado por la sombra que se crea en la parte superior de los vehículos y esto generaba bloques un poco más grandes en algunos casos, lo que ocasiona este incremento en la velocidad medida. Además de algunos falsos positivos, que fueron presentes solo en este algoritmo.

Para obtener la misma eficacia en los resultados mostrados, debe respetarse la relación entre la tasa de muestreo de la cámara y la velocidad máxima esperada en la vía. Una tasa de muestreo baja implica que se toman pocas muestras por vehículo, si este pasa rápido por la escena. Además la disparidad de las imágenes acumuladas depende de la distancia entre las líneas de acumulación. Si están muy cercanas se deberá tener una tasa de captura más elevada para poner medir la misma velocidad máxima.

Si se cumplen las recomendaciones para la ubicación de la cámara sobre la vía, la selección de la tasa de muestreo de la cámara y la ubicación de las líneas. El algoritmo presenta resultados en la medición superiores al 85% en el conteo y 95% en la velocidad para una vía con un semáforo. Y superiores al 94% en el conteo y 96% en la velocidad para una vía de tráfico fluido en la que no ocurran detenciones de los vehículos.

7.4 Tiempos de ejecución

Bajo un procesador moderno como la serie de procesadores corei7 de Intel, el algoritmo puede ser ejecutado a la tasa de captura máxima de la cámara, 125 cuadros por segundo. También pueden ejecutarse más de una instancia del programa al tiempo, permitiendo tener una plataforma de análisis posterior de archivos de video o un único núcleo de procesamiento para varias cámaras en simultáneo. La velocidad en la plataforma corei7, está limitada solamente por la tasa de captura de la cámara.

Los resultados del algoritmo 2 de detección de primer plano, presentan tiempos promedios ya que este algoritmo con forme pasa el tiempo incrementa su tiempo de procesamiento. Es recomendable reiniciar el modelo cada cierta cantidad de cuadros para evitar demoras en el algoritmo. En el caso del algoritmo propuesto en este modelo, se reinicia cada M acumulaciones de N cuadros, ambos especificados por el usuario, para las pruebas se usó un tiempo de 3 minutos antes de reiniciar los modelos de *codebook*.

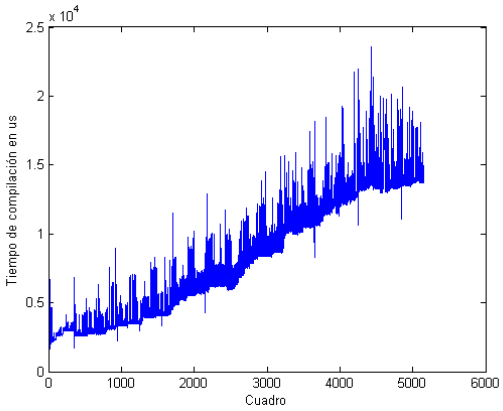

Figura 28. Tiempo de compilación algoritmo 2

Los tiempos de procesamiento de las plataformas Atom y OMAP son suficientes para implementar el sistema, limitando la selección de una u otra, a las velocidades máximas de los vehículos en la vía, como se explicará en detalle más adelante.

8 Conclusiones y trabajo futuro

8.1 Conclusiones

La técnica de análisis de video propuesta que consiste en procesar solamente sobre dos líneas de captura, incrementa la velocidad a la cual son procesados los cuadros del video. Partiendo de esta técnica se pueden hacer algoritmos más livianos computacionalmente que pueden ser embebidos en plataformas de un tamaño muy reducido (como las que se están produciendo para teléfonos inteligentes como el OMAP y computadores netbook como el Atom) y a la vez se gana en la reducción de consumo de energía. Ambas condiciones favorecen que los sistemas desarrollados puedan ser ubicadas directamente en la vía, realizando el procesamiento de los datos en el sitio, reduciendo la cantidad de datos que debe analizar el usuario, al reportar a los centros de información solamente los datos requeridos.

Por otra parte, la ubicación de la cámara resulta un aspecto muy relevante. Ubicar bien la cámara es beneficioso para realizar la estimación de los parámetros de tráfico a partir de visión por computador, además de disminuir la complejidad de los algoritmos, al evitar traslapes de los vehículos; también mejora la eficiencia de los algoritmos, al poder ubicar mejores puntos para realizar la medición. Por ejemplo la medición de la velocidad en los puntos recomendados sobre el objeto se ve beneficiada al evitar hacer una corrección en la cual se tendrá incertidumbre, si no se cuenta con la altura de los vehículos en la vía.

En cuanto a los algoritmos de detección propuestos, son capaces de identificar un objeto así este sea capturado en un solo cuadro y por ende acumulado en una sola línea. Sin embargo, si esto ocurre, es más difícil separar un vehículo del ruido y se crearan más errores, es decir la relación señal a ruido disminuye. Por ejemplo, a una tasa de captura de 30 cuadros por segundo, un automóvil que tiene en promedio 4.2 metros de largo si se desplaza a 100 Km/h, es capturado en 4 líneas. Si hay una distancia de un metro entre las dos líneas de captura, solo habrá una línea de diferencia entre los dos bloques de 4 líneas acumulados, por lo que la resolución en la velocidad se verá afectada. Por esto la ubicación de las líneas debe hacerse dependiendo de la tasa de captura de la cámara y de la velocidad y el largo esperado de los vehículos. Para esto puede usarse la tabla de diseño, presente en el anexo de tablas o las ecuaciones de la sección 5.3.

Respecto al conteo, este es mejor mientras más cercanas están las líneas y exista una menor disparidad entre las imágenes de acumulación. Así mismo, la velocidad puede estimarse siempre y cuando la línea se encuentre lo suficientemente separada, de tal manera que exista disparidad entre las dos imágenes acumuladas por líneas, así, que al ubicar la cámara en la vía, se deberá encontrar la relación adecuada dependiendo de las necesidades de la implementación. Otra alternativa a revisar sería el uso de tres líneas, de forma que las dos exteriores se utilicen para medir velocidad y las otras tres, en parejas, se usen para validar la existencia y posterior conteo de los vehículos.

Al pensar en la implementación en la vía, dados los tiempos de ejecución y la tasa de procesamiento del algoritmo, esta es determinante a la hora de seleccionar cual de los sistemas propuestos ubicar en la vía, si es una vía lenta en la que no se presentan velocidades mayores a 20 Km/h, por ejemplo, la entrada a un peaje o un parqueadero, el sistema basado en OMAP usando el algoritmo 1 o 2 es más que suficiente. Sin embargo, si es una vía rápida en la que se esperan velocidades mayores a 100 km/h puede usarse la plataforma basada en el procesador Atom, usando el algoritmo 2 si se desea tener una alta precisión en la velocidad, al tener este un alta tasa de cuadros por segundo; o el algoritmo 1 si lo que se desea es tener la mayor precisión en el conteo. Si se desea analizar uno o varios archivos de video tomados en la vía, puede hacerse en menor tiempo usando la plataforma corei7.

8.2 Trabajo futuro

La acumulación de líneas se realizó en forma perpendicular a la vía, sin embargo esto trae con sigo los problemas de muestreo que limitan la estimación de la velocidad. Para evitar esta limitación, se podrían ubicar líneas de acumulación paralelas a la vía como se muestra en la Figura 29.

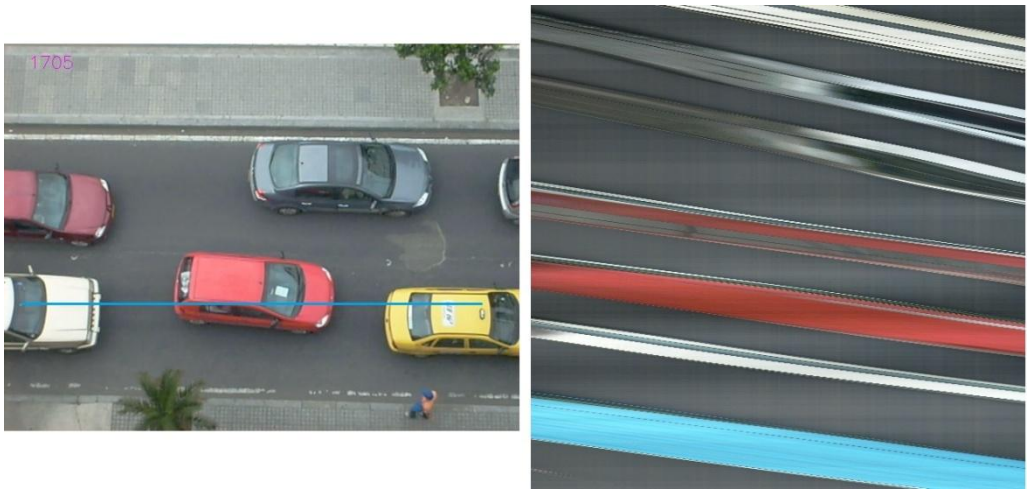

Figura 29. Línea de captura horizontal con respecto a la vía

Esta acumulación permite encontrar la velocidad de manera más precisa, midiendo la pendiente de las líneas formadas al pasar el vehículo y permite separar los peatones de los vehículos por la forma que estos dejan al pasar la línea. Sin embargo, tiene el inconveniente de no detectar vehículos que no pasen por el carril o de perder la detección de los vehículos que cambien de carril.

De pruebas hechas durante la realización de este proyecto, se trabajó con la correlación entre las imágenes de acumulación, a partir de esta es posible estimar directamente la velocidad media de todos los vehículos en las imágenes, ya que permite encontrar la disparidad. También es posible encontrar un estimado del volumen vehicular buscando los máximos en la correlación entre líneas de las imágenes acumuladas. Tiene la desventaja de no poder detectar un solo vehículo y no poder estimar la velocidad de un solo vehículo por aparte. En el desarrollo del proyecto se implementaron funciones compatibles con OpenCV, capaces de funcionar con varios hilos de proceso, que facilitan el análisis de las imágenes a partir de correlación y diferencia de cuadrados, estas están anexas en el DVD que acompaña este libro. Ver Figura 30.

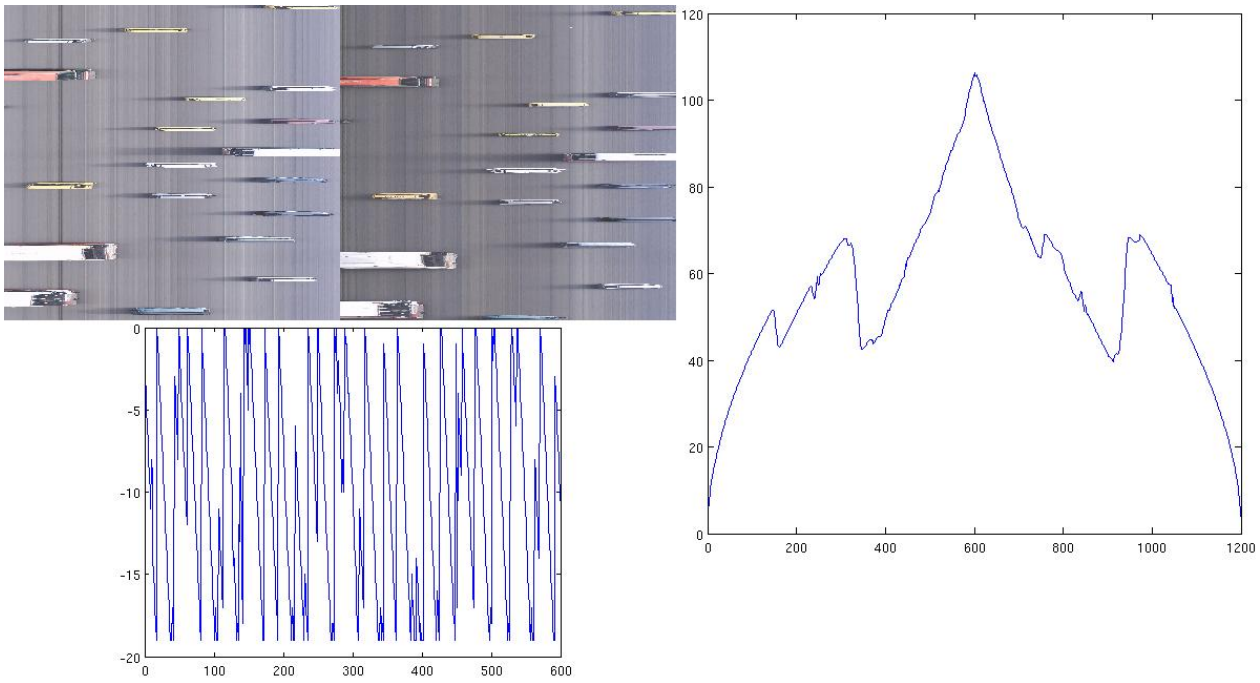

Figura 30. Resultados por correlación de imágenes acumuladas.

La implementación de algoritmo usando el núcleo OMAP solo usó el procesador ARMv7 y no el núcleo de DSP ni la GPU de la tarjeta, por lo que podría acelerarse el algoritmo en general, implementando el algoritmo de análisis de línea sobre el DSP, ya que este se basa en acumulación y ordenamiento de vectores y el de análisis de bloques y conteo sobre la GPU ya que este tiene más operaciones en punto flotante. En el anexo 5 en el dvd adjunto, se encuentra los pasos de configuración y compilación de OpenCV y GCC sobre la tarjeta Beagleboard, para referencia de futuros trabajos sobre esta plataforma.

9 Bibliografía

[1] M. José and M. Pardillo, *Apuntes de ingeniería de tráfico*, Madrid, España: E.T.S. Ingenieros de Caminos, Canales y Puertos, 2003.

[2] US Departament of Transportation, *Traffic Detector Handbook*, Georgetown Pike, VA: Federal Highway Administration, 2006.

[3] J. Kim-Sung and L. Ming, *Computer vision based real-time information acquisition for transport traffic*, IEEE, 2005.

[4] C. Setchell, "Applications of Computer Vision to Road-traffic Monitoring," *Interpretation A Journal Of Bible And Theology*, 1997, p. 170.

[5] M.W. Green, "The Appropriate and Effective Use of Security Technologies in US Schools," *ojp.usdoj.gov*, 1999.

[6] "cifras sobre accidentalidad, http://www.fonprevial.org.co/index.php?option=com_content&view=article&id=70&Itemid=89," 2009.

[7] "Welcome - OpenCV Wiki, http://opencv.willowgarage.com/wiki/," 2010.

[8] creative commons, "creative commons — bsd license, http://creativecommons.org/licenses/BSD/deed.es_CO," 2010.

[9] H. Sugano and R. Miyamoto, "Opencv Implementation Optimized for a Cell Broadband Engine Processor," *2009 IEEE 13th Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*, 2009, pp. 182-187.

[10] L.C. Training, "POSIX Threads Programming, https://computing.llnl.gov/tutorials/pthreads/#Thread,Training, Livermore Computing."

[11] L. Comunidad, "POSIX - Wikipedia, la enciclopedia libre, http://es.wikipedia.org/wiki/POSIX."

[12] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, Sebastopol, Canada: O'Reilly, 2008.

[13] G. Urrego, F. Calderon, A. Forero, and J. Quiroga, "Adquisición de variables de tráfico vehícular usando visión por computador," *Revista de ingeniería Universidad de los Andes*, vol. 1, 2009, pp. 7-15.

[14] F. Calderon and G. Urrego, "Conteo automatico de vehículos," *Tesis de Pregrado*, 2008, p. 62.

[15] L. Li, W. Huang, I.Y. Gu, and Q. Tian, *Foreground object detection from videos containing complex background*, In MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia, 2003.

[16] P.L. Rosin, *Thresholding for Change Detection*, 1998.

[17] K. KIM, T. CHALIDABHONGSE, D. HARWOOD, and L. DAVIS, "Real-time foreground–background segmentation using codebook model," *Real-Time Imaging*, vol. 11, 2005, pp. 172-185.

[18] J.J. O'Connor and E.F. Robertson, "classical Light, http://www-history.mcs.st-andrews.ac.uk/HistTopics/Light_1.html," 2002, p. 1.

[19] D.C. Brown, "Close-range camera calibration," *PHOTOGRAMMETRIC ENGINEERING*, vol. 37, 1971, pp. 855-866.

[20] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1106-1112.

[21] SONY, "PlayStation EYE, http://uk.playstation.com/ps3/peripherals/detail/item78698/PlayStation®Eye/," 2010, p. 1.

[22] OmniVision, "ov7720VGA product brief," *Product Brief*, 2008, p. 2.

[23] A. Popovich, "PS3Eye Disassembly & IR Filter, http://codelaboratories.com/," *codelaboratories*, 2008, p. 1.

[24] A. Ospite, "gspca:Subdriver ov534 , http://git.kernel.org/?p=linux/kernel/git/torvalds/linux-2.6.git;a=commit;h=fbb4c6d20f29f2b10daad31cc6238d91f93d70d4," *Página oficial del repositorio GIT del Kernel de GNU/Linux*, 2008, p. 1.

[25] "x86 - Wikipedia, la enciclopedia libre, http://es.wikipedia.org/wiki/X86."

[26] "Texas Instruments OMAP - Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Texas_Instruments_OMAP."

[27] "ARM - Wikipedia, la enciclopedia libre, http://es.wikipedia.org/wiki/ARM."

[28] "Ubuntu homepage, http://www.ubuntu.com/," 2010.

[29] "The Ångström Distribution | Embedded power, http://www.angstrom-distribution.org/," 2010.

[30] "GCC, the GNU Compiler Collection - GNU Project - Free Software Foundation (FSF), http://gcc.gnu.org/."

[31] K. Hirahara and K. Ikeuchi, *Detection of street-parking vehicles using line scan camera and scanning laser range sensor*, IEEE, 2003.

[32] C. Ngo, T. Pong, and H. Zhang, "Motion analysis and segmentation through spatio-temporal slices processing.," *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 12, 2003, pp. 341-55.

[33] L. Anan and Y. Zhaoxuan, *Video Vehicle Detection Algorithm through Spatio-Temporal Slices Processing*, IEEE, 2006.

[34] L. Anan and Y. Zhaoxuan, *Video Vehicle Detection Algorithm based on Virtual-Line Group*, IEEE, 2006.

[35] Y. Malinovskiy, Y. Wu, and Y. Wang, "Video-Based Vehicle Detection and Tracking Using Spatio-Temporal Maps.," *Annual Meeting of the Transportation Research Board and for Publication in Transportation Research Record: the Journal of the Transportation Research Board*, vol. 88, 2008, pp. 1-18.

[36] S. SUZUKI and K. BE, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, 1985, pp. 32-46.

[37] "VNC - Virtual Network Computing from AT&T Laboratories Cambridge, http://www.hep.phy.cam.ac.uk/vnc_docs/index.html."

[38] "What is Samba?, http://www.samba.org/samba/what_is_samba.html."