Universitat d'Alacant
Universidad de Alicante

Tesis Doctoral

Monitorización visual automática de tráfico rodado

Redouane Kachach

Director
Dr. José M. Cañas Plaza

Programa de Doctorado en Informática
Tesis presentada para aspirar al grado de
DOCTOR POR LA UNIVERSIDAD DE ALICANTE

Instituto Universitario de Investigación Informática

Agradecimientos.

Al final de este trabajo, echo la mirada atrás para contemplar el camino recorrido. Ha sido sin duda un viaje largo, de mucho esfuerzo, lleno de momentos buenos y otros no tanto. No puedo negar la satisfacción personal al llegar a este punto, pues es la culminación de varios años de dedicación y sacrificio no sólo por mi parte sino por todas las personas que han estado ahí de alguna forma u otra. Este trabajo lleva mi nombre pero sabéis que es también vuestro.

Quiero empezar por mi director de tesis, José María, porque sé que sin tu apoyo este trabajo no tendría sentido. Quiero darte las gracias por tu dedicación, paciencia y por confiar en mí. Por todos estos años de colaboración que han marcado mi desarrollo personal y profesional. Gracias por enseñarme el camino y ayudarme a recorrerlo. También quiero aprovechar la ocasión para agradecer a mi tutor Miguel Angel Cazorla. Por las continuas ayudas y por estar ahí siempre que se le necesita. Muchas gracias por todo.

En segundo lugar quiero dar las gracias a *Hajar*, mi mujer y fuente de inspiración en esta vida. Tu apoyo, paciencia y sacrificio durante todos estos años han sido claves para culminar este trabajo. Juntos lo hemos logrado. Ha llegado el momento de recoger el premio y disfrutarlo en familia. Y como no, a nuestros *principitos Louay* y *Alae*. Gracias por interrumpir de vez en cuando mis largas jornadas de programación y aislamiento con la excusa de darme un beso. Sois mi razón de ser, el mejor regalo que me ha dado Dios en esta vida.

También me gustaría agradecer a mi familia, y en especial a mis padres y hermanos. Vuestro cariño y apoyo a lo largo de estos años han sido esenciales para llevar a cabo este trabajo. Sé que nunca agradeceré lo suficiente a mi Madre por su sacrificio y dedicación no sólo en estos años sino a lo largo de mi vida. Gracias por hacerme ver que nada en este mundo es imposible. Que el trabajo y la dedicación son las palabras mágicas para lograr los metas en esta vida. Gracias por todo.

Madrid, 12 de julio de 2016

Redouane Kachach

Resumen

La gestión del tráfico es una tarea muy compleja. La información generada por los sistemas tradicionales de monitorización (por ejemplo espirales) es muy limitada e insuficiente para realizar estudios más ambiciosos y complejos sobre el tráfico. Hoy en día esto es un problema en un mundo donde técnicas como el *Big Data* se han metido en todos los ámbitos. Esta tesis se enfoca en abordar el problema de monitorización automática de vehículos empleando sensores más modernos como las cámaras. Estos sensores llevan ya varias décadas instalados en las carreteras pero con una misión limitada a la monitorización pasiva de las mismas. Nuestro objetivo es aprovechar estos sensores con algoritmos capaces de extraer información útil de forma automática de las imágenes. Para ello, vamos a abordar dos problemas clásicos en este campo como son el seguimiento y la clasificación automática de vehículos en varias categorías.

Dentro del marco de los sistemas inteligentes de transporte (ITS, por sus siglas en inglés), el trabajo presentado en esta tesis aborda los problemas típicos relacionados con el seguimiento de vehículos como la eliminación de sombras y el manejo de oclusiones. Para ello se ha desarrollado un algoritmo que combina criterios de proximidad espacial y temporal con un algoritmo basado en *KLT* para el seguimiento tratando de aprovechar las ventajas de cada uno de ellos. En el contexto de la clasificación se ha desarrollado un algoritmo híbrido que combina plantillas 3D que representan las distintas categorías de vehículos junto con un clasificador *SVM* entrenado con características visuales de camiones y autobuses para afinar la clasificación. Todos los algoritmos utilizan una sola cámara como sensor principal. Los sistemas desarrollados han sido probados y validados experimentalmente sobre una amplia base de vídeos tanto propios como otros independientes. Hemos recopilado y etiquetado una amplia colección de vídeos de tráfico representativos de un variado abanico de situaciones que ponemos a disposición de la comunidad científica como banco de pruebas.

Abstract

Traffic monitoring is a complex task. The information provided by legacy traffic monitoring systems (based on inductive loops, etc) is not sufficient to perform advanced studies about the behavior of vehicle traffic. Nowadays, this is an issue in a world where the *Big Data* is being applied everywhere. The main goal of this PhD thesis is to tackle the problem of automated vehicle traffic monitoring by using advanced sensors such as cameras. These have been installed everywhere for a long time but with a limited mission consisting in simple tasks such as passive monitoring, etc. Our goal is to provide them with advanced algorithms so they become able to extract useful information about the traffic status from images. For this purpose, we will try to tackle vehicles tracking and classification, two classic problems in the traffic monitoring field.

This thesis focuses on basic problems related with the vehicle tracking such as shadow removal and occlusions handling. Novel algorithms that combine spacial proximity criteria with *KLT* algorithm have been developed for this purpose trying to benefit from the strengths of both. Vehicle classification is performed by means of a hybrid approach that combines 3D templates representing the different supported vehicle categories and an *SVM* classifier trained on *HOG* descriptors to distinguish between Trucks and Buses categories. The developed system has been validated experimentally over a wide range of real traffic videos gathered as part of this thesis and others provided by the scientific community. We gathered and tagged many traffic videos with different illumination, weather and traffic density conditions. This data set is made available to the scientific community as part of this thesis.

Índice general

1. Introducción **1**

1.1. Cámaras en tecnologías automóviles y su entorno 3

1.2. Las cámaras como sensor de tráfico 9

1.2.1. Visión computacional en tráfico urbano 11

1.2.2. Monitorización de autopistas basada en visión 15

1.3. Sistemas comerciales basados en visión 16

1.4. Objetivos 17

1.5. Estructura de la memoria 19

2. Estado del arte **20**

2.1. Detección de fondo 21

2.1.1. Distribuciones unimodales 21

2.1.2. Mezcla de gaussianas 22

2.1.3. Codebooks 25

2.1.4. Métodos basados en la estimación no paramétrica de la función de densidad 26

2.2. Seguimiento de vehículos 28

2.2.1. Seguimiento basado en regiones 29

2.2.2. Seguimiento basado en características 30

2.2.3. Seguimiento basado en modelos 30

2.3. Clasificación de vehículos 31

2.3.1. Técnicas basadas en características 32

2.3.2. Técnicas basadas en modelos 3D 39

3. Diseño global del sistema **44**

3.1. Diseño 44

3.2. Escena y calibración semi-automática de la cámara 45

3.3. Base de datos de vídeos 52

4. Seguimiento de vehículos 55

4.1. Detección de vehículos 55
4.1.1. Detección de fondo 56
4.1.2. Detección automática de la carretera 57
4.1.3. Extracción de blobs 59
4.1.4. Tratamiento de sombras 61
4.2. Seguimiento de vehículos 67
4.2.1. Zonas de entrada y seguimiento de vehículos 68
4.2.2. Seguimiento por características visuales 70
4.2.3. Emparejamiento entre blobs y vehículos 72
4.2.4. Resolución de oclusiones 78
4.2.5. Estimación de la velocidad 84
4.3. Experimentos sobre tráfico real 85
4.3.1. Métricas 85
4.3.2. Robustez del seguimiento frente a las oclusiones 87
4.3.3. Evaluación del sistema de detección y eliminación de sombras 93
4.3.4. Seguimiento en distintas condiciones meteorológicas 96
4.4. Conclusiones 97

5. Clasificación de vehículos 99

5.1. Clasificación basada en modelos 3D 99
5.1.1. Función similitud de categoría 102
5.1.2. Acumulación de similitud 107
5.2. Clasificación basada en HOG/SVM 108
5.2.1. Histogramas de Gradientes Orientados 111
5.2.2. La parte trasera de camiones y autobuses 115
5.2.3. Parametrización del clasificador SVM 120
5.2.4. Parametrización de HOG 125
5.3. Experimentos sobre tráfico real 131
5.3.1. Métricas 132
5.3.2. Clasificación en distintas condiciones de iluminación y densidad de tráfico 132
5.3.3. Clasificación basada en HOG/SVM 137
5.3.4. Análisis de la acumulación de similitud 139

5.3.5. Tráfico denso y distintas condiciones de meteorología 143
5.4. Conclusiones 144

6. Conclusiones **148**
6.1. Objetivos 148
6.2. Contribuciones 151
6.3. Trabajos futuros 152

A. Modelo pinhole de cámara **153**

B. Algoritmo Kanade-Lukas-Tomasi **156**

C. Maquinas de Vectores Soporte **160**

D. Resultados de clasificación sobre tráfico real **167**

Índice de figuras

1.1. Centro de monitorización de tráfico 1
1.2. División del sector de SIT a nivel de aplicación 3
1.3. Sistema de reconocimiento automático de matrículas 4
1.4. Radar de tramo basado en reconocimiento de matrículas 4
1.5. Sistema de aparcamiento automático 5
1.6. Sistema de ayuda a bordo del vehículo 5
1.7. Sistema de visión de *ojo de pájaro* para asistencia en el aparcamiento 6
1.8. Cámara trasera de BMW para asistencia en el aparcamiento . . . 6
1.9. Vehículo autónomo participante en el *DARPA Grand Challenge* . 7
1.10. Conjunto típico de sensores empleados en un vehículo autónomo . 8
1.11. Ejemplo de análisis de oclusión basado en la predicción de trayectoria del vehículo 10
1.12. Control de semáforo rojo con cámara 12
1.13. Sistema multi-sensor para la detección de infracción de luz roja . 12
1.14. Sistema de monitorización automática de una intersección basado en visión 13
1.15. Sistema de detección de incidencias en túnel 14
1.16. Sistema basado en una cámara termal para la detección de incidentes en una autopista 15
1.17. Sistema de detección de peatones basado en visión 16
1.18. El sistema Meta Fusion de la empresa Morpho 17

2.1. Utilización de la información de carriles para eliminar las sombras y resolución de oclusiones 33
2.2. Plantillas reconfigurables basadas en HOG 35
2.3. Parte frontal de vehículos empleada para generar una base de datos de sus *eigenfaces* 36

2.4. Muestras de entrada utilizadas en la fase de entrenamiento 38
2.5. Reconstrucción 3D empleando puntos de fuga 39
2.6. Clasificación basada en modelos 3D construidos mediante el uso de puntos de fuga 40
2.7. Plantillas con información 3DHOG para clasificación de vehículos 40
2.8. Reconstrucción del histograma 3DHOG a partir de una imagen de entrada 41
2.9. Modelo 3D compuesto por cinco partes: una para modelar el cuerpo del vehículo y otras cuatro para las ruedas 42
2.10. Modelo de vehículo 3D genérico para representar varias categorías 43
2.11. Clasificación de vehículos basada en plantillas 3D 43

3.1. Diagrama de bloques del TrafficMonitor 46
3.2. Configuración de la cámara 47
3.3. Configuración de los ejes XYZ respecto a la carretera 48
3.4. Disposición de la cámara sobre la carretera 50
3.5. Estimación de puntos de fuga: El punto P_1 se obtiene como la intersección entre las líneas (A, D) y (B, C) mientras que el punto P_2 es la intersección entre las líneas (A, B) y (C, D) 50
3.6. Calibración semi-automática de la cámara 51
3.7. Resultados de los experimentos utilizando la herramienta semi-automática de calibración sobre varios vídeos con distintos ángulos y altura de la cámara 52

4.1. Fases del algoritmo de detección de vehículos 55
4.2. Ventana de detección automática de la carretera 58
4.3. Resultado de la dilatación 58
4.4. Carretera detectada de forma automática 59
4.5. Varios ejemplos del área mínima utilizada para filtrar los blobs . . 60
4.6. Ejemplos de detección de fondos en varias condiciones de ilumina-ción 61
4.7. Ejemplo de fallo de detección de una moto en un escenario con nubes. La imagen de la derecha corresponde a la salida del módulo de detección de fondo y movimiento. En este caso la detección de sombras ha fallado e incluido gran parte del cuerpo del blob como si fueran sombras. 61
4.8. Ejemplos fallos de detección de fondo en escenarios muy soleados . 62

4.9. Problemas de detección de sombras: En la primera imagen podemos ver que las sombras del autobús son detectadas correctamente (color gris en el fondo), mientras que en el caso del camión tanto las sombras como el reflejo del camión sobre la pared adyacente son clasificados erróneamente como parte del objeto en movimiento 62
4.10. Efecto de las sombras sobre el ancho de un vehículo: $W_{vehiculo}$ es el ancho real del vehículo mientras que $W_{sombras}$ es el ancho del vehículo incluyendo su sombra cuando el sol esta en la posición Sh_3. $Sh_{1,2,3}$ son las sombras correspondientes a tres posiciones distintas del sol 63
4.11. Efecto de las sombras sobre el ancho de los vehículos 64
4.12. Algoritmo para eliminación de sombras 65
4.13. Resultados intermedios del algoritmo de eliminación de sombras . 66
4.14. Experimentos de detección y eliminación de sombras 67
4.15. Seguimiento de vehículos: cada vehículo tiene un identificador único que le acompaña durante toda la zona de seguimiento. 68
4.16. Zonas de entrada y seguimiento de vehículos 69
4.17. Procedimiento recursivo para el cálculo del desplazamiento en la implementación piramidal de KLT 71
4.18. Fases del algoritmo de seguimiento espacial 73
4.19. Elipse 2D asociada al vehículo 73
4.20. Seguimiento mediante proximidad espacial entre dos instantes consecutivos 74
4.21. Algoritmo de asociación entre blobs y vehículos 75
4.22. Búsqueda de puntos característicos KLT 76
4.23. Representación gráfica del cuadro envolvente del vehículo (VBB) y el cuadro mínimo que contiene los puntos KLT (FPBB) 77
4.24. Asociación entre blob y vehículo mediante el algoritmo de votación 78
4.25. Algoritmo de seguimiento de vehículos 79
4.26. Proceso de detección de oclusión entre dos vehículos 82
4.27. Reglas del cálculo de los puntos característicos KLT 82
4.28. **Oclusión parcial**: uno o varios vehículos están parcialmente ocultados pero el seguimiento puede seguir hasta que vuelvan a ser visibles. **Oclusión Total**: a medida que los vehículos se acercan al horizonte la furgoneta acabará tapando por completo al coche . . 83

4.29. Detección de oclusión: los vehículos 77,78 y 79, 80 están entrando en oclusión (el color del identificador de los vehículos en oclusión cambia de amarillo a rojo) 84
4.30. Estimación de velocidad 85
4.31. Ejemplos de seguimiento con pocas oclusiones 90
4.32. Secuencias de detección de oclusión 91
4.33. Ejemplos de fallos de detección de oclusión 93
4.34. Ejemplo de fallo del sistema de detección causado por las sombras. Ambos vehículos están unidos por su sombra provocando un error de detección. 94
4.35. Respuesta del sistema de detección de oclusiones ante fallos en la eliminación de sombras 95
4.36. Ejemplos de seguimiento con distintas condiciones climatológicas . 96

5.1. Diagrama de bloques del algoritmo de clasificación 100
5.2. Ejemplo de varias plantillas proyectadas sobre los vehículos 101
5.3. Ejemplo de re-proyección de un punto de la carretera al punto 3D correspondiente 102
5.4. Las áreas B, I y D correspondientes respectivamente al área del blob, la intersección y diferencia simétrica con una plantilla 3D proyectada. 103
5.5. Figura (a): polígono generado por la proyección de la plantilla sobre la carretera. Figura (b) puntos de escaneo utilizados para calcular las áreas B, I y D correspondientes de la plantilla proyectada . . 104
5.6. Ejemplo de distintos tamaños de plantillas y blob 104
5.7. Análisis de la función de cálculo de similitud para dos valores de β 105
5.8. Simulación de carretera y vehículos creada con Gazebo 106
5.9. Análisis del comportamiento de la función de cálculo de similitud para las categorías intermedias. Cada curva representa los valores de similitud obtenidos para todas las categorías cuando fijamos la categoría de referencia a la clase representada por la curva 107
5.10. Proceso de clasificación de las categorías camión/autobús 109
5.11. Dos soluciones distintas para el mismo conjunto de datos. En la solución de la izquierda los vectores soporte elegidos (color verde) dan lugar a un margen más estrecho que la solución de la derecha. 110
5.12. Detección de bordes utilizando la información del gradiente 112
5.13. Efecto del cambio de contraste sobre el valor del gradiente 114

5.14. Solapamiento entre bloques en la configuración 2×2 de HOG . . 114
5.15. Distribución de los coeficientes de aportación de cada celda en el descriptor final. Las celdas interiores participan en cuatro bloques distintos. 115
5.16. Configuraciones geométricas del descriptor HOG: **(a)** configuración rectangular (R-HOG), **(b)** configuración circular (C-HOG) . 115
5.17. Imagen de la vista de un autobús con la configuración empleada de la cámara. La parte lateral (B) del vehículo sufre mucha deformación por el efecto de la proyección, además de estar afectada por las sombras. La parte superior (A) no tiene información visual suficiente para diferenciar entre camiones y autobuses. La parte (C) no sufre ningún efecto además contiene información suficiente para diferenciar entre ambas categorías. 116
5.18. Histogramas HOG superpuestos sobre las imágenes de las categorías soportadas 117
5.19. Extracción de la parte trasera del camión/autobús. La proyección de la plantilla 3D del vehículo sobre el blob es utilizada para extraer de forma precisa su parte trasera mediante rectificación de planos. 120
5.20. Ejemplos de muestras de entrenamiento de autobuses, camiones y camiones cisterna 121
5.21. Resultados de validación cruzada para la función radial gaussiana para las tres categorías soportadas. Los resultados de precisión son representados en una escala de colores donde las zonas amarillas corresponden a las combinaciones con los mejores valores de precisión. 124
5.22. Media de los resultados obtenidos para la función radial gaussiana 125
5.23. Valores de clasificación obtenidos para las tres categorías: camión, autobús y camión cisterna mediante le uso de una función lineal . 125
5.24. Efecto del número de orientaciones elegido sobre el resultado final del descriptor HOG para un tamaño de celda fijo de 16×16 . . . 126
5.25. Efecto del tamaño de la celda elegido sobre el resultado final del descriptor HOG para un número de orientaciones fijado en $\beta = 16$ 127
5.26. Tiempo empleado en el entrenamiento del clasificador SVM en función del tamaño de la celda y el número de orientaciones elegidos para el descriptor HOG 128

5.27. Tiempo empleado en el entrenamiento del clasificador SVM en función del tamaño de la celda para distintas configuraciones de orientación 129
5.28. Tiempo necesario para calcular el descriptor HOG para distintas combinaciones de tamaño de celda y número de orientaciones. . . 131
5.29. Tiempo necesario para predecir el resultado mediante SVM en función del tamaño del descriptor HOG. Los tamaños corresponden respectivamente a configuraciones con: 2, 4, 8, 16 y 32 orientaciones con un tamaño de celda fijo de $s = 8 \times 8$. 131
5.30. Ejemplo de detección de sombras en un escenario nublado. Los grises de la imagen de la derecha corresponden a las sombras detectadas por el módulo de detección de fondo. 133
5.31. Ejemplos de clasificación en escenarios con nubes. 134
5.32. Ejemplos de clasificación en escenarios soleados con pocas oclusiones. 135
5.33. Ejemplos de clasificación en escenarios soleados con oclusiones . . 136
5.34. Ejemplos de detección de las categorías camión, autobús y camión cisterna mediante HOG/SVM en escenarios tanto con como sin sombras. La imagen en miniatura corresponde a la imagen rectificada del trasero de cada vehículo. 137
5.35. Ejemplos de fallos de clasificación mediante HOG/SVM 138
5.36. Análisis de la acumulación de similitud para cada categoría en condiciones favorables. El rectángulo que parte desde el origen de tiempos indica el tiempo que se tarde en iniciar la clasificación del vehículo. 139
5.37. Análisis de la acumulación de similitud en presencia de oclusiones. Las ventanas de oclusión representan intervalos de tiempo donde el vehículo no es clasificado porque está involucrado en una oclusión 140
5.38. Comportamiento de la acumulación de similitud para dos vehículos que entran en oclusión 141
5.39. Fallo de clasificación debido a fallos en la detección de fondo . . . 142
5.40. Errores de clasificación ocasionados por fallos en al detección de fondo en escenarios con niebla 143
5.41. Clasificación con distintas condiciones meteorológicas 144

5.42. Fallos de clasificación bajo condiciones meteorológicas extremas. En (a) hay tres falsos positivos dado que no hay ningún vehículo en la escena. En (b) el subsistema de detección no detecta el vehículo que hay en la imagen. 145
5.43. Resultados de clasificación en escenarios de tráfico denso. 145

A.1. Modelo pinhole 154

C.1. Elementos geométricos relacionados con SVM 161
C.2. Conjunto de datos que no se puede separar de forma lineal 164
C.3. Variables de holgura 165

Índice de cuadros

3.1. Base de datos de vídeos recopilados 54

4.1. Resultados de seguimiento en escenarios sin oclusiones 88
4.2. Resultados de seguimiento en escenarios con pocas oclusiones . . . 89
4.3. Resultados de seguimiento en escenarios con oclusiones 90
4.4. Resultados de la detección de oclusiones en varios escenarios . . . 92
4.5. Resultados de seguimiento para los mismos vídeos con el mecanismo de detección y eliminación sombras activado y desactivado . . 94
4.6. Resultados de seguimiento con distintas condiciones meteorológicas 97

5.1. Tamaño medio en metros de las categorías soportadas 100
5.2. Tamaño de las categorías soportadas y las categorías intermedias . 107
5.3. Resultados del clasificador SVM para distintas configuraciones de tamaño de celda y β. Los resultados marcados con * corresponden a resultados inestables del clasificador. 130
5.4. Matriz de confusión para escenarios con nubes y bajas oclusiones . 133
5.5. Resultados de clasificación en escenarios con nubes y bajas oclusiones 133
5.6. Matriz de confusión para escenarios soleados con bajas oclusiones 134
5.7. Resultados de clasificación en escenarios soleados con bajas oclusiones 135
5.8. Matriz de confusión para escenarios soleados con oclusiones 136
5.9. Ejemplos de clasificación en escenarios soleados con oclusiones. . . 136
5.10. Resultados de clasificación para las categorías Camión/Autobús. . 138

D.1. Resultados de clasificación en escenarios con nubes y bajas oclusiones 168
D.2. Resultados de clasificación en escenarios soleados con bajas oclusiones 169
D.3. Resultados de clasificación en escenarios soleados con oclusiones . 169

Capítulo 1

Introducción

El término Sistemas inteligentes de transporte (SIT) se refiere a aplicar sistemas avanzados de información y comunicación a las infraestructuras de transportes y a los vehículos para mejorar la seguridad vial, la eficiencia y el rendimiento de las carreteras. El término se puede extender a más medios de transporte pero en general se aplica a medios de transporte terrestre.

Se trata de un área en continuo avance dada la importancia que tiene el transporte terrestre en el desarrollo socioeconómico de los países. Hoy en día la cantidad de vehículos que circulan por la carreteras es cada vez mayor debido al crecimiento y los cambios en la densidad la población. El efecto sobre las infraestructuras viales es un incremento en la densidad de vehículos, tiempos de viajes en hora punta cada vez más largos y altos niveles de contaminación generada por el tráfico terrestre. En definitiva, la gestión del tráfico se ha vuelto una tarea muy complicada.

Figura 1.1: Centro de monitorización de tráfico

La información ofrecida por los sistemas tradicionales de monitorización (basados en su mayoría en espirales) es limitada e insuficiente para llevar a cabo estudios más ambiciosos con el fin de entender la naturaleza compleja del tráfico en las grandes ciudades y gestionarlo de forma eficiente. En este contexto aparecen las cámaras como sensores que llevan ya varias décadas instaladas en las carreteras y que hasta ahora su uso se ha limitado a entregar información en directo a un operador humano capaz de analizarla y tomar las decisiones oportunas. Sin embargo, los avances en las últimas décadas en el campo de la visión artificial apoyados por el abaratamiento de estos sensores han abierto la puerta de par en par a nuevos campos de investigación cuya finalidad es procesar de forma automática los datos recibidos desde las cámaras con el fin de extraer información valiosa acerca del tráfico como la densidad de flujo, incidencias, condiciones meteorológicas, etc.

En la practica los SIT se pueden dividir en varias áreas, sin embargo los principales sectores son:

1. Gestión avanzada del transporte
2. Gestión avanzada del transporte público
3. Sistemas de cobro automático

El primer sector representa la gestión de carreteras a nivel global. Entre sus misiones se encuentra agilizar el tráfico en las carreteras, optimizar el uso de las infraestructuras, mejorar la movilidad y hacer que las redes de transporte sean cada vez más seguras. Para llevar a cabo estas funciones este sector se basa en una red extensa de sensores de todo tipo, donde las cámaras juegan un papel primordial. Los datos sensoriales son recibidos y procesados en centros especializados donde son analizados por operadores humanos o de forma automática para realizar las acciones oportunas en caso de problemas (accidentes, alta densidad de tráfico, etc).

El segundo sector tiene el mismo fin que el anterior pero esta vez limitado a sistemas de transporte público. Las misiones de este sector incluyen una gestión eficaz y optimizada de las redes de transporte público, ofrecer a los ciudadanos información en tiempo real de las distintas alternativas de transporte disponibles para trayectos de media o larga distancia. Esto incluye las rutas y los horarios así como información adicional como incidencias en la red de transporte, etc. Este sector se encarga también de los sistemas de cobro aplicados al sector de

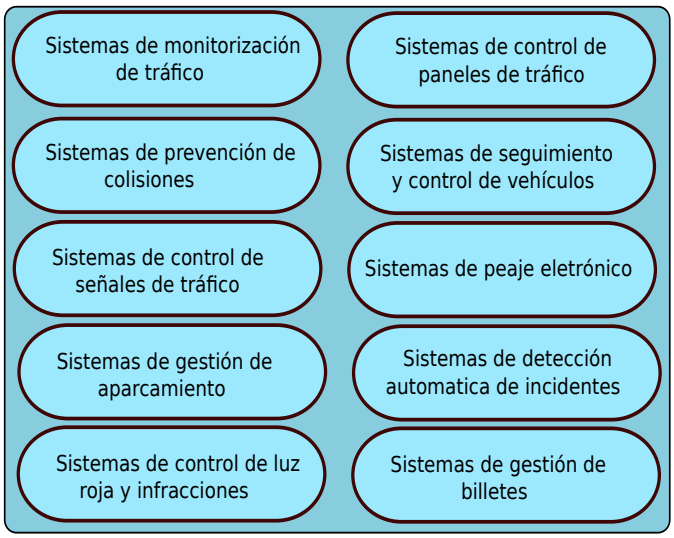


Figura 1.2: División del sector de SIT a nivel de aplicación

transporte y el manejo de billetes. El objetivo de esta área es facilitar la integración y la unificación de los sistemas de cobro para facilitar la creación de billetes multimodales o las distintas tarjetas de transporte subvencionados por el estado.

Finalmente, el sector de cobro automático abarca todos los sistemas eletrónicos de gestión de cuotas. La gestión de cobros de forma automática en autopistas con sistemas como telepeaje (*free flow*) se basa en el uso de sistemas inalámbricos de comunicación entre el vehículo y el sistema de gestión de cobro. En este tipo de sistemas los vehículos llevan a bordo un dispositivo que permite identificarlos de forma segura y gestionar la cuota sin obligar al vehículo a detenerse. Este sistema ayuda a mejorar el flujo vehicular y reducir los atascos en los puntos de peaje.

Hasta ahora hemos visto una división a nivel funcional del los SIT. Este sector se puede también dividir a nivel de aplicaciones, la figura 1.2 enumera las aplicaciones más importantes

1.1. Cámaras en tecnologías automóviles y su entorno

El concepto de la ciudad inteligente no se puede entender sin el uso de las cámaras. La calidad de información ofrecida por estos sensores los hace únicos. El desafío está en procesarla mediante algoritmos capaces de extraer, analizar y entender esta información. Hasta hace poco, si analizamos el uso más común de las cámaras en este campo, la aplicación más popular era la de detección automática de matrículas. Podemos decir que ésta fue una de las aplicaciones pioneras donde la visión artificial ha demostrado ser una solución práctica frente a

otras alternativas. El éxito en este caso se debe principalmente a la gran fiabilidad que ofrecen estos sistemas sobre todo en entornos controlados como a las puertas de acceso de un aparcamiento. En estos escenarios se parte con la ventaja de poder controlar el objeto bajo análisis y las condiciones ambientales como la iluminación, la posición del vehículo, etc.

Figura 1.3: Sistema de reconocimiento automático de matrículas

Otra aplicación directa de esta tecnología son los denominados *radares de tramo*. La idea principal en este caso es aprovechar la capacidad de reconocimiento de matrículas para calcular la velocidad media de un vehículo. Normalmente son situados a ambos lados de un túnel dado que en esta situación la trayectoria del vehículo está bajo control.

Otro campo donde el uso de las cámaras ha ido en aumento es en la seguridad vial. Hoy en día ya hay varios modelos comerciales de coches que son capaces de detectar las señales de tráfico de forma automática. De esta forma el vehículo puede avisar al conductor en caso de sobrepasar el límite de velocidad, avisarle de la dirección de la próxima curva, etc. En la actualidad hay varias marcas

Figura 1.4: Radar de tramo basado en reconocimiento de matrículas

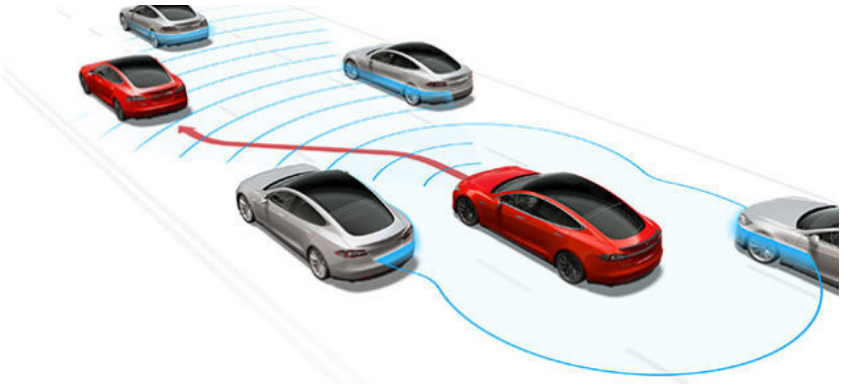


Figura 1.5: Sistema de aparcamiento automático

Figura 1.6: Sistema de ayuda a bordo del vehículo

comerciales como Ford Focus, BMW 7-series o Opel Insignia entre otras que incorporan este tipo de sistemas.

Los datos sensoriales ofrecidos por una cámara son ricos en información. Esto ha hecho posible implementar varios sistemas de seguridad activa que ayudan al conductor a determinar y resolver situaciones de riesgo. Hoy en día es bastante común encontrar modelos comerciales con posibilidad de detección automática de peatones y mecanismos para frenar de forma autónoma en situaciones de riesgo de atropello. Sin duda se trata de un sistema importante para salvar vidas reduciendo el número de accidentes. Otra aplicación destacada de las cámaras es como asistente en el aparcamiento. En este caso pueden actuar como ayudante durante las maniobras de aparcamiento o como parte de un sistema multisensorial para aparcar el vehículo de forma totalmente automática. La figura 1.5 muestra un ejemplo de estos sistemas.

En la actualidad hay varios modelos que ofrecen asistencia avanzada de aparcamiento basada en cámaras. Por ejemplo el sistema de *ojo de pájaro* de Nissan

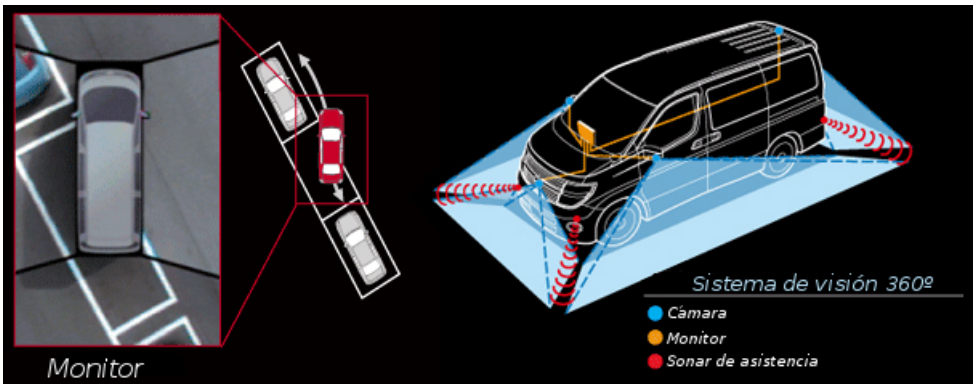


Figura 1.7: Sistema de visión de *ojo de pájaro* para asistencia en el aparcamiento

ofrece al conductor una vista área de la escena del aparcamiento incluyendo tanto el vehículo (simulado) como sus alrededores. Esto permite al usuario llevar la maniobra de forma sencilla y segura sabiendo en cada momento la posición exacta del vehículo respecto a los elementos que lo rodean. En realidad el sistema hace uso de cuatro cámaras, las imágenes emitidas por las mismas son procesadas para componer una vista área simulada. Al mismo tiempo el conductor puede utilizar la cámara trasera para obtener más detalles de la escena. La figura 1.7 ilustra gráficamente el sistema de Nissan.

BMW tiene un sistema parecido al ofrecido por Nissan junto con una camera trasera apoyada por sensores para medir las distancias con los obstáculos. El sistema proyecta una serie de líneas que ayudan en la orientación del vehículo durante la maniobra del aparcamiento. Además, las imágenes de la cámara son enriquecidas con marcas 3D que emplean un código de color para marcar la distancia de los obstáculos más cercanos. Adicionalmente el sistema muestra al usuario en cada momento un mapa del vehiculo y los obstáculos que tiene alrededor. La figura 1.8 muestra un ejemplo de este sistema en funcionamiento.

Figura 1.8: Cámara trasera de BMW para asistencia en el aparcamiento

Figura 1.9: Vehículo autónomo participante en el *DARPA Grand Challenge*

Otro de los campos más activos en la investigación es el de los coches autónomos. Se trata de un tema clásico desde el punto de vista de la investigación. De hecho sí nos remontamos a la historia reciente veremos que hay intentos de crear coches autónomos que datan de 1950. Desde entonces hubo varios avances en su mayoría destinados a equipar el coche con sistemas cada vez más sofisticados para navegar de forma autónoma. En este contexto el *DARPA Grand Challenge* es uno de los eventos más importantes en la última década. Se trata de una competición donde varios vehículos autónomos tratan de completar un circuito en el desierto en un tiempo determinado. El camino en este caso está libre de señales de tráfico. Este evento ha sido creado por la agencia de investigación DARPA para incentivar el desarrollo y la creación de vehículos completamente autónomos capaces de llevar a cabo misiones pre-planificadas. La idea es poder utilizar esta tecnología avanzada en misiones de exploración así como para aplicaciones militares. Se trata de una competición de formato abierto en la que pueden participar todo tipo de entidades tanto públicas como privadas. Entre los participantes podemos encontrar desde universidades hasta empresas privadas.

La misma agencia organiza también la competición *DARPA Urban Challenge*, que tiene el mismo fin que la anterior sin embargo en este caso el circuito se desarrolla en un entorno urbano donde los vehículos tienen que recorrer $96km$ de forma autónoma en menos de seis horas. Durante el trayecto éstos tienen que interactuar con otros vehículos que circulan de forma autónoma u ocupados por conductores profesionales. Esta competición tiene más complejidad que la versión del desierto dado que los vehículos tienen que interpretar de forma correcta todas

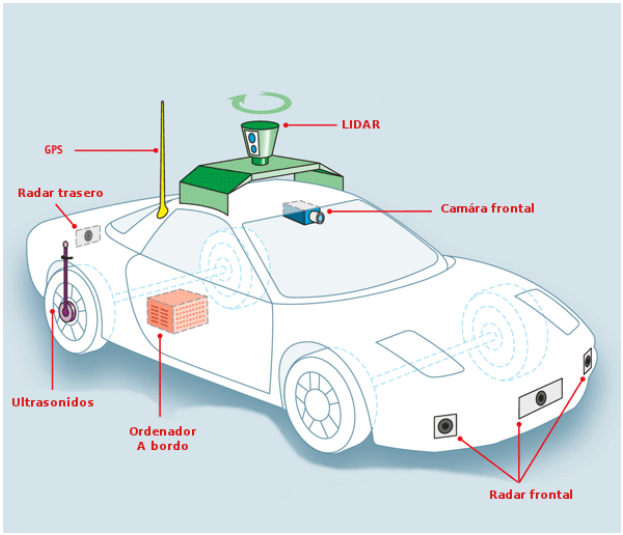


Figura 1.10: Conjunto típico de sensores empleados en un vehículo autónomo

las señales y las reglas de tráfico que afrontan durante la trayectoria. El objetivo principal de este evento es desarrollar vehículos autónomos para su uso en la vida cotidiana.

La mayoría de las grandes marcas automovilísticas han estado investigando de una forma u otra en este campo, sin embargo si analizamos la historia reciente quizás el coche más famoso (por lo menos cara al público) es el de Google. La figura 1.10 muestra el diseño común que tiene este tipo de vehículos. Esta figura ilustra una configuración típica de un vehículo autónomo. La parte frontal del coche está equipada con un radar para detectar vehículos cercanos al mismo. Este sensor se utiliza para frenadas de emergencia y para mantener la distancia de seguridad con el vehículo que circula por delante. El sensor de ultrasonidos en las ruedas traseras permite al vehículo realizar movimientos con alta precisión dado que es capaz de detectar obstáculos muy cercanos al vehículo. Este sensor es muy útil para llevar a cabo maniobras que requieren de mucha precisión como el aparcamiento automático. La antena GPS permite al vehículo localizarse con alta precisión en exteriores. La cámara Lidar (por sus siglas en inglés Light detection and ranging) es un sensor que está realizando un barrido continuo en 360º con el fin de construir un mapa tridimensional de todo lo que tiene a su alrededor. En el caso del coche de Google el barrido de este sensor es capaz de alcanzar una distancia máxima de 200 metros. Finalmente, la misión de la cámara frontal es analizar los objetos que el coche tiene en frente: peatones, señales de tráfico, otros vehículos, etc. El ordenador a bordo se encarga de analizar toda la información recibida a través de los sensores y tomar las acciones oportunas.

Otra de las empresas que está pujando fuertemente sobre el desarrollo de coches autónomos es Tesla. Los modelos de esta marca son en general conocidos por ser eléctricos, pero además de esta característica esta marca ha estado trabajando en modelos que son completamente autónomos. De hecho en la actualidad el *Test Modelo S* ya dispone de un piloto automático, que es un concepto distinto al de un coche totalmente autónomo. En este caso el vehículo a petición del conductor es capaz de iniciar un modo en el cual puede navegar sin intervención del conductor. Sin embargo, éste tiene que estar atento a las maniobras que realiza el vehículo y tomar el control en caso de emergencia. La empresa está trabajando a corto plazo para tener modelos completamente autónomos. Sus planes son ambiciosos y pretende tener un coche que sea capaz de viajar incluso sin conductor. Así podrá recoger a su propietario e incluso hacerlo en base a su calendario.

1.2. Las cámaras como sensor de tráfico

La aplicación de visión computacional al análisis de tráfico es un campo de estudio bastante joven en comparación con otras disciplinas. Las primeras aplicaciones de este tipo han sido presentadas por [Onoe M, Ohba k, 1976] y [Hilbert, 1978]. Sin embargo, si analizamos las fechas de los trabajos científicos en este campo se puede ver que hasta la década de los 80 no hubo una actividad constante de publicaciones en el tiempo. En este periodo podemos encontrar ya varias referencias a aplicaciones basadas en visión y orientadas al análisis del tráfico de vehículos. Por ejemplo [Hoose, N, 1989] presenta una técnica para calcular de forma automática la longitud de la cola formada por vehículos parados o circulando a velocidades muy bajas. En el mismo año, [J.M Blosseville, C. Krafft, F. Lenior, V. Motyka, S. Beucher, 1989] presenta el sistema *TITAN*. Se trata de un sensor de tráfico capaz de analizar escenas de entre 100 hasta 300 metros de longitud, día y noche. Este sistema es capaz de extraer varios parámetros del tráfico como la velocidad, el flujo de vehículos y la longitud de cola en caso de atascos. Las principales características de los vehículos utilizadas en este trabajo son las luces de noche y los techos, capós y las sombras frontales de día con una capacidad de procesamiento de hasta 4 imágenes por segundo.

Otro ejemplo de la misma década es el *CCATS* presentado por [Versavel, J. ; Lemaire, F. ; Van der Stede, D., 1989]. Se trata de un sistema capaz de extraer varios parámetros del tráfico como el número de vehículos, ocupación de carriles y la velocidad media de los mismos. Además, este sistema es capaz de detectar anomalías en el flujo de vehículos como atascos o largas colas y emitir alarmas

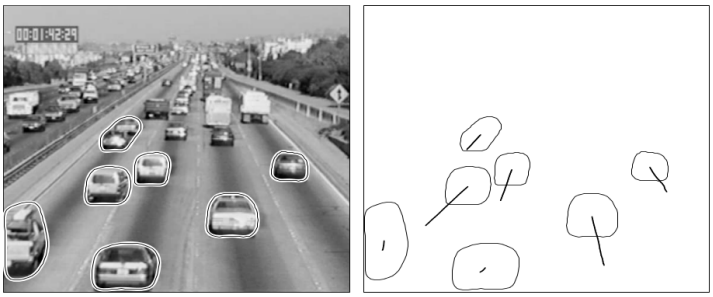


Figura 1.11: Ejemplo de análisis de oclusión basado en la predicción de trayectoria del vehículo

alertando de la situación. Según los autores este sistema ha sido instalado en su momento en varias ciudades en Bélgica con el fin de mejorar el control de tráfico.

Todos los trabajos realizados hasta finales de los 80 se concentraban en analizar el tráfico de vehículos intentando extraer métricas e información relativamente simples para utilizarlas en la extracción de macrovariables del tráfico. Sin embargo, esto va cambiar con la entrada de la nueva década. En los 90 se puede notar un aumento considerable en el número de publicaciones en esta línea de investigación y en vez de hacer un macro análisis de la escena se empiezan a dar los primeros pasos hacia un análisis más detallado de la misma basado en las características visuales de los vehículos. En este apartado podemos encontrar los primeros trabajos enfocados al uso de plantillas 3D de vehículos para tareas como la clasificación y el seguimiento. [Baker, K.D Sullivan, G.D., 1992] presenta un análisis de la viabilidad del seguimiento basado en modelos 3D aplicado al tráfico. Los autores de este trabajo han tratado de aplicar las técnicas de reconocimiento y estimación de posición de objetos pensadas normalmente para una sola imagen a una secuencia de imágenes. Adicionalmente, el algoritmo presentado se basa en un conocimiento a priori de la escena 3D mediante el uso de una cámara calibrada. El algoritmo presentado en este trabajo está lejos de funcionar en tiempo real pero se puede considerar uno de los primeros trabajos en dar el paso hacia este objetivo.

[Koller, D., Weber, J., Malik, J., 1994] presenta un trabajo extenso donde se describe un sistema experto orientado al seguimiento de vehículos haciendo especial hincapié en el manejo de oclusiones una de las grandes dificultades que hay que resolver en la mayoría de sistemas dedicados al análisis de tráfico y que será uno de los problemas que intentaremos abordar como parte de esta tesis. Este trabajo analiza de forma detallada la formación de oclusiones en una autovía y propone una solución para manejar estas situaciones basada en la estimación de

profundidad de cada vehículo en la escena. En la figura 1.11 podemos ver un ejemplo de este análisis.

[G.D. Sullivan, K.D. Baker, A.D. Worral, C.I. Attwood, P.M. Remagnino, 1997] presenta un algoritmo para la clasificación de vehículos utilizando una combinación de plantillas en 1-D generadas sobre varios modelos de vehículos y otro conjunto de plantillas en 2-D empleadas para la verificación y aceptación de las hipótesis generadas en la primera fase del algoritmo de clasificación. [Coifman et al., 1998] presenta un sistema que trata de seguir un conjunto de características visuales de los vehículos (en vez del vehículo entero) para ser más robusto frente a oclusiones parciales.

Si analizamos las tendencias y las técnicas presentadas en la última década éstas se pueden clasificar de varias maneras. La primera de ellas es el entorno donde se aplican. En este caso se puede distinguir claramente entre dos tipos de escenarios: tráfico urbano y tráfico en autopistas. La diferencia entre los dos en general radica en el tipo de escenarios que se presenta en cada caso.

1.2.1. Visión computacional en tráfico urbano

La rama del tráfico urbano intenta aplicar las técnicas de visión computacional para analizar el tráfico dentro de las ciudades o entornos urbanos en general. Las aplicaciones en este caso pueden variar desde una simple recogida de datos hasta fines mucho más complejos como el análisis del comportamiento de vehículos para velar por la aplicación de las normas de tráfico o la detección de anomalías. Estos entornos suelen ser bastante complejos ya que el número de actores que pueden aparecer en la escena es muy elevado: vehículos, peatones, ciclistas, árboles, edificios, símbolos de señalización, etc. A esto hay que sumarle la alta probabilidad de la existencia y formación de atascos y retenciones sumado al problema de las oclusiones.

Quizás la aplicación más famosa de las cámaras en entornos urbanos es la de controlar los semáforos para detectar los conductores que no respetan la luz roja. El funcionamiento de estas cámaras es muy simple. Básicamente están conectadas a las luces que controlan tráfico. La cámara se activa para detección de forma automática cada vez que se enciende la luz roja (figura 1.12). A partir de este momento cualquier vehículo que rebase la línea blanca que marca la frontera señalizada será fotografiado y multado. Este esquema sencillo es bastante común, sin embargo hay legislaciones más estrictas donde para multar un vehículo se necesita disponer de una prueba irrefutable de que el coche ha violado la luz roja

Figura 1.12: Control de semáforo rojo con cámara

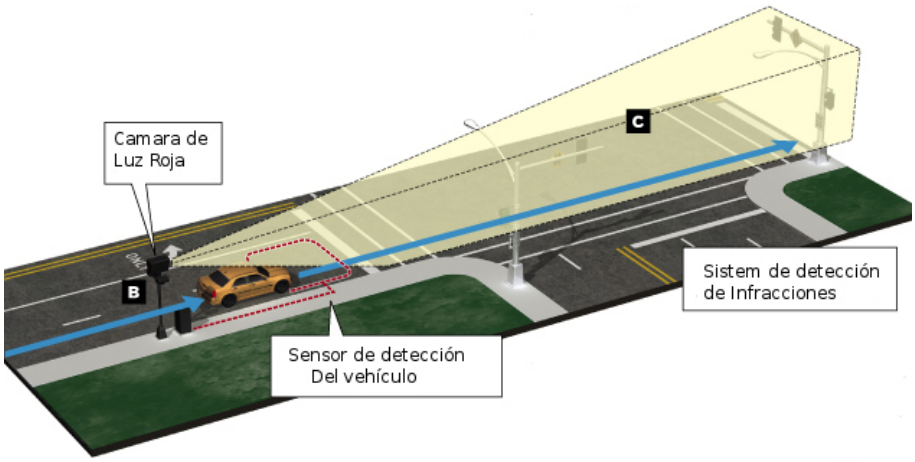


Figura 1.13: Sistema multi-sensor para la detección de infracción de luz roja

cuando ésta ya estaba encendida. Es un problema bastante común dado que muchos conductores aceleran en la luz ámbar corriendo el riesgo de infringir la luz roja cuando ésta acaba de encenderse. Ante este problema los sistemas de detección de infracción en semáforos han evolucionado hacia la grabación completa de la infracción. Para ello, como se puede ver en la figura 1.13, la cámara se sitúa en un punto anterior al semáforo que tiene que vigilar. La idea es que sea capaz de grabar toda la escena de los vehículos infractores para que se vea claramente el momento de la infracción. Además, con esta nueva disposición el sistema es capaz de predecir los conductores con alta probabilidad de violar la luz roja. Para ello la cámara es equiparada con mecanismos para medir la velocidad de los coches que pasan, cuando ésta rebasa un límite la cámara se activa para grabar la posible infracción.

Otra de las aplicaciones más comunes de visión en el contexto del tráfico

Figura 1.14: Sistema de monitorización automática de una intersección basado en visión

interurbano es la detección automática de incidentes. Se trata de un caso de uso vital de las cámaras dado que los centros de monitorización de tráfico por lo general disponen de miles de cámaras instaladas en varias carreteras y de muy poco personal para prestar atención a las mismas. Un sistema automático de detección de incidentes (AID, por sus siglas en inglés) facilita esta labor de tal forma que el personal que está monitorizando el tráfico en estos centros sólo tiene que prestar atención a alarmas generadas por estos sistemas. Además de mejorar el tiempo de respuesta en caso de accidentes, estos sistemas ayudan a salvar vidas al minimizar el tiempo de intervención empleado para dar la voz de alarma y mandar los primeros servicios de auxilio al lugar del incidente si hace falta. En la misma línea, el personal puede actualizar rápidamente los paneles cercanos al lugar del incidente evitando un posible peligro al acercarse al área afectada.

Analizando la literatura podemos ver que dentro del análisis de tráfico urbano esta área es una de las más activas. [Vermeulen, 2014] presenta un sistema que combina información visual con información recibida de una cámara térmica para detectar incidentes. La combinación de ambas tecnologías da como resultado un sistema robusto a condiciones meteorológicas extremas que pueden afectar a la información visual recibida por la cámara. El sistema presentado ha sido instalado y probado de forma exitosa en el puente de *Rion* en Grecia. [Chang et al., 2012] presenta un sistema basado en la detección de matrícula para el mismo fin. La idea básica en este caso es utilizar la información del tiempo de viaje del vehículo para inferir incidencias en la trayectoria del mismo.

Otro de los temas con creciente interés es el análisis del comportamiento de vehículos en una rotonda o bifurcación (figura 1.14). Saber con exactitud el núme-

Figura 1.15: Sistema de detección de incidencias en túnel

ro de vehículos que circulan en cada dirección es de gran utilidad a la hora de analizar el flujo vehicular por las redes de transporte. Este tipo de estudios suele ser el paso previo antes de realizar cualquier tipo de modificación en las infraestructuras viales. En este contexto [Nateghinia and Moradi, 2014] presenta un sistema capaz de seguir los vehículos en una intersección y resolver el problema de las oclusiones. La fase de detección se basa en un mecanismo simple que modela cada píxel como una distribución gaussiana combinado con un mecanismo de modelado dinámico de textura. [Shirazi and Morris, 2016] presenta un sistema similar pero con más funcionalidad dado que además de seguir y contar los vehículos en cada salida de la intersección, este sistema es capaz de estimar la velocidad y el tiempo de espera en la intersección para todos los vehículos. Adicionalmente, este trabajo presenta un algoritmo para reconstruir la trayectoria de los vehículos.

Finalmente, dentro del tráfico urbano una parte esencial de la infraestructura de redes son los túneles. Se trata de un elemento que juega un papel esencial en la reducción del flujo de tráfico sobre superficie. En Madrid por ejemplo los túneles de las M30 suman una longitud de 43 kilómetros. Dicho esto, monitorizar toda la red de túneles es una tarea muy importante y compleja al mismo tiempo. Abogando por la constitución física de estos espacios de difícil acceso, la rápida detección de incidencias juega un papel muy importante en la seguridad de esta infraestructura. Un factor esencial en este caso es la intervención rápida en caso de incidencias para evitar segundos accidentes o catástrofes en caso de incendio dentro del túnel. La figura 1.15 muestra un sistema de detección de incidencias en funcionamiento dentro de un túnel.

1.2.2. Monitorización de autopistas basada en visión

Otra rama es la del análisis de tráfico en autopistas. En este caso las técnicas tratan de analizar el tráfico de vehículos sobre una autopista para extraer datos de interés como el tipo de vehículos que viajan por la autopista, su velocidad, índice de ocupación, etc. A diferencia del tráfico urbano, en este caso la escena es más controlada ya que el número de actores que puede aparecer es limitado. Normalmente el tipo de vehículos que pueden viajar por una autopista varía de un país a otro pero habitualmente suele estar regulado por ley. Así que sin pérdida de generalidad se puede asumir que el tráfico está limitado a las distintas categorías de coches y vehículos pesados. Normalmente las bicicletas y los peatones no pueden compartir la misma calzada con los vehículos con motor en este tipo de escenarios. Esta reducción de complejidad viene bien a la hora de diseñar algoritmos para este tipo de escenarios.

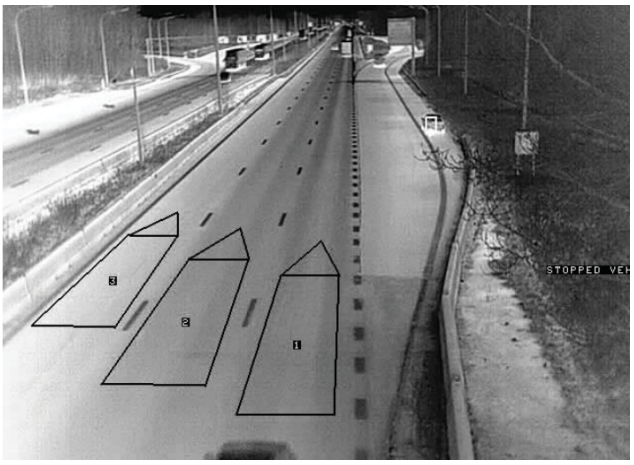


Figura 1.16: Sistema basado en una cámara termal para la detección de incidentes en una autopista

En general, la mayoría de los sistemas de detección de incidencias que hemos visto para el tráfico urbano valen igual para el tráfico en autopistas ya que hay muchas incidencias que son comunes a ambos escenarios. Quizás el factor diferencial más importante que distingue entre los sistemas de monitorización de tráfico en entornos urbanos y los sistemas para monitorización en autopistas es la velocidad de los vehículos. Mientras que en el primer caso en general el límite de velocidad suele ser bajo, en las autopistas estos límites son mayores y su rango varía dependiendo de la legislación de cada país. Por ejemplo, en España este límite suele ser de 120km/h mientras que en Alemania hay muchas autopistas donde la velocidad no está limitada. Este factor influye a la hora de diseñar sistemas de control o monitorización para estos escenarios.

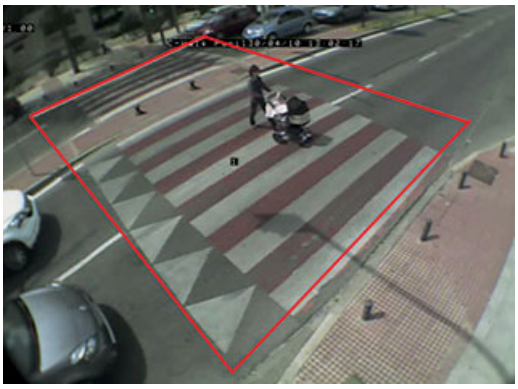


Figura 1.17: Sistema de detección de peatones basado en visión

1.3. Sistemas comerciales basados en visión

Dentro del contexto de los SIT, en el caso de monitorización de tráfico hay muchas tecnologías y sensores siendo las cámaras uno de los más importantes. En la actualidad el uso de visión ha pasado de la investigación teórica al ámbito comercial. En el mercado hay varias soluciones comerciales basadas completamente en visión. El creciente aumento en fiabilidad de estos sistemas junto con su coste competitivo y el abanico amplio de soluciones e información que ofrecen hace que sean la alternativa ideal a los sistemas clásicos de detección.

El uso de la tecnología en los sistemas inteligentes de transporte ha ido en crecimiento desde la década de los ochenta. Se trata de un sector muy importante tanto en los países desarrollados como en países en vías de desarrollo. A nivel mundial hay varias empresas que se dedican íntegramente a producir y mantener estos sistemas, entre ellas se pueden citar: Flir (EEUU), Siemens (Alemania), Indra (España), Kapsch (Austria), Q-Free (Norguega), Thales (Francia), Sigtec (Austria).

Siemens es uno de los uno de proveedores de sistemas de control de tráfico a nivel mundial. Esta empresa ofrece sistemas completos para todas las funciones que un centro de control de tráfico requiere. Su familia de sistemas *SitTraffic* ofrecen un amplio abanico de funcionalidades que cubren las necesidades que suele tener un centro de monitorización de tráfico.

FLIR es una empresa muy activa en este sector, su catalogo de productos abarca sistemas con varias funciones desde soluciones para la detección automática de la formación de atascos en autopistas o para la detección de presencia de vehículos en intersecciones hasta la detección automática de peatones en un paso de cebra (figura 1.17). Su sistema *TrafiOne* está equipado con una cámara térmica para regular las luces de tráfico. Esto se lleva a cabo mediante algoritmos que procesan

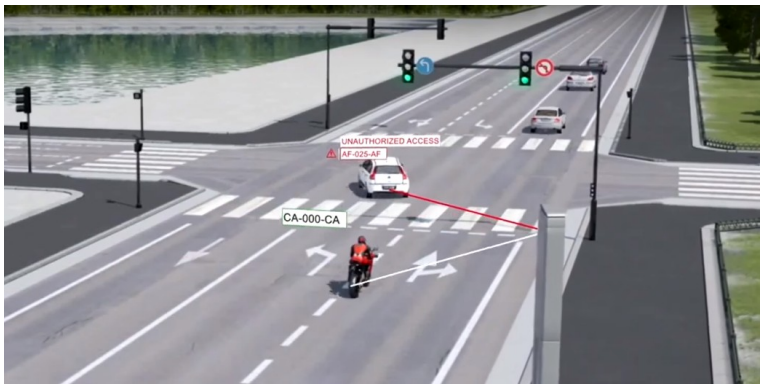


Figura 1.18: El sistema Meta Fusion de la empresa Morpho

la información de la cámara para detectar la presencia de bicicletas y peatones de forma automática. Su serie de productos *TrafiCam* incorpora soporte para la detección automática y seguimiento de vehículos en intersecciones. El sistema es capaz también de medir la longitud de las colas que se forman en estos puntos estratégicos.

Siempre dentro del contexto de la monitorización, la empresa Morpho está trabajando para la comercialización de un nuevo tipo de monitores de tráfico con más funcionalidad que los radares tradicionales. Su sistema *Meta Fusion* (figura 1.18) es capaz de monitorizar y medir la velocidad sobre ocho carriles de forma simultánea. Además, esta solución viene con un sistema de análisis de trayectoria del vehículo que es capaz de detectar aquellos vehículos que realicen maniobras prohibidos o que no respeten la distancia de seguridad. Este sistema en la actualidad está en fase de pruebas.

1.4. Objetivos

Esta tesis ha sido desarrollado en el marco de los sistemas inteligentes de transporte, concretamente en la monitorización visual del tráfico rodado. El trabajo realizado tiene un enfoque teórico-práctico. Además de profundizar en los conceptos teóricos relacionados con todas y cada una de las fases que componen el sistema final de monitorización, el resultado tiene que ser un sistema funcional que además de cumplir con los objetivos establecidos, sea capaz de funcionar sobre tráfico real. Dicho esto, el objetivo principal de este trabajo es la validación experimental del sistema final sobre tráfico real con distintas condiciones de iluminación, tráfico, etc.

Antes de seguir con más detalles relacionados con el marco de desarrollo de esta tesis vamos a concretar sus objetivos. Dentro del campo de monitorización

visual aplicada al tráfico, vamos a centrarnos en técnicas de seguimiento y clasificación de vehículos en autopistas. El objetivo principal es desarrollar un sistema completo de monitorización que cubre todas las fases típicas en este tipo de sistemas que consisten en: detección de vehículos, seguimiento y clasificación. Nos detendremos en cada fase para evaluar las técnicas que se emplean en la misma, ver cúales son sus puntos fuertes y débiles, intentando siempre aportar soluciones a los problemas típicos relacionados con cada fase.

Dentro de las etapa de detección y seguimiento abordaremos el problema de oclusiones, uno de los problemas más complejos y que impacta de forma directa en la calidad de los resultados de cualquier sistema de monitorización. Además de detectar los errores hay que poner en marcha mecanismos para evitar que se propaguen al resto del sistema. Nuestro objetivo es cometer el mínimo número de errores posible en esta fase dado que es la entrada a la fase de seguimiento. El objetivo de la fase de clasificación es distinguir los vehículos que pasan por una autopista en cinco categorías: Motocicletas, Coches, Furgonetas, Camiones y Autobuses. Después de repasar las técnicas que se emplean en esta área de investigación en concreto, vamos a centrarnos principalmente en métodos de clasificación híbridos que combinan información de plantillas 3D con características visuales. Finalmente, forma parte de los objetivos recopilar una amplia base de vídeos con tráfico real bajo distintas condiciones para validar experimentalmente el sistema desarrollado y poner a disposición de la comunidad científica la base de datos creada.

Además de conseguir estos objetivos, el sistema final ha de cumplir una serie de requisitos básicos que nos hemos planteado al principio del desarrollo de la tesis. Estos son:

1. El sistema tiene que emplear como sensor principal una sola cámara
2. El sistema tiene que funcionar en tiempo real
3. El sistema tiene que soportar distintas configuraciones de la posición de la cámara
4. Los algoritmos desarrollados tienen que permitir el seguimiento y clasificación de varios vehículos circulando por varios carriles de forma simultánea
5. Los algoritmos tienen que estar preparados para funcionar de día bajo distintas condiciones de iluminación y condiciones normales de meteorología.

Dicho esto, no forma parte de los objetivos de esta tesis construir un sistema capaz de funcionar de noche o bajo condiciones climatológicas extremas como lluvia intensa, nieve, etc.

1.5. Estructura de la memoria

El capítulo 1 además de hacer un repaso sobre la aplicación de visión al campo de tráfico, presenta un breve repaso de la historia reciente de esta disciplina y su evolución a lo largo de las últimas décadas. El capítulo 2 hace un extenso repaso de todos los trabajos científicos y las referencias bibliográficas más importantes que sirven de antesala para la realización de esta tesis, profundizando en los temas principales como el seguimiento y la clasificación. El capítulo 3 contiene una descripción detallada del escenario físico sobre el cual el sistema tiene que monitorizar el tráfico. Además, presenta el diseño global del sistema elaborado en esta tesis. El capítulo 4 describe los detalles del subsistema de detección de vehículos y repasa los algoritmos empleados en esta fase del sistema. Además, profundiza en los algoritmos diseñados e implementados para resolver el problema del seguimiento mientras que el capítulo 5 hace lo propio pero esta vez para la etapa de clasificación. Ambos capítulos tienen una sección extensa donde se repasan los resultados experimentales obtenidos durante la validación del sistema sobre tráfico real. Finalmente, el capítulo 6 resume las conclusiones derivadas del trabajo realizado, y hace un repaso de los objetivos planteados, el grado de satisfacción de cada uno de ellos y las contribuciones de esta tesis. La memoria se cierra describiendo los aportes, contribuciones y las posibles líneas futuras de esta tesis.

Capítulo 2

Estado del arte

A lo largo de este capítulo completaremos la información presentada en el capítulo anterior donde hemos visto de forma general las distintas aplicaciones que tiene hoy en día la monitorización automática de tráfico. Revisaremos la bibliografía científica más relevante relacionada con este campo y veremos las técnicas que han sido propuestas por la comunidad científica para resolver los distintos problemas relacionados con el mismo.

Dentro del contexto global de monitorización haremos hincapié en las técnicas relacionadas con el problema de clasificación automática de vehículos basada en visión. Nos detendremos en las técnicas más relevantes para ver los fundamentos teóricos en los que se basan, sus ventajas y desventajas frente a otras alternativas. Para llevar a cabo este análisis clasificaremos las técnicas estudiadas en distintas familias según las soluciones que emplean. Esto nos servirá también para profundizar en el trabajo y describir las líneas de investigación que han servido como base o inspiración para esta tesis.

Los sistemas presentados hasta ahora han constituido las primeras líneas de investigación y los primeros pasos para aplicar la visión computacional al análisis de tráfico. A partir de finales de los 90 el número de publicaciones y artículos relacionados con este campo han crecido de forma notable motivados por el aumento de la potencia de cómputo cada vez mayor y la bajada de precios de las cámaras. Esto ha abierto las puertas de par en par para la implementación de nuevos algoritmos en tiempo real y el uso de visión 3D.

En general, independientemente del escenario elegido para llevar a cabo la monitorización de tráfico, hay una serie de problemas comunes y básicos que hay que resolver. Éstos en general son:

1. Detección de vehículos

2. Seguimiento de vehículos

3. Clasificación de vehículos

El primer problema consiste en dada una secuencia de imágenes, detectar dentro la misma donde se sitúa cada vehículo. Hay dos técnicas clásicas para llevar a cabo esta tarea, la primera consiste en estimar el fondo de la escena para poder detectar movimiento en la misma. La segunda técnica consiste en intentar averiguar la posición de los vehículos tratando de identificarlos dentro de la misma.

El problema del seguimiento consiste en asociar los vehículos que han sido previamente identificados entre distintos fotogramas de la secuencia de imágenes. Finalmente, la clasificación consiste en averiguar a qué categoría pertenece cada vehículo. En las siguientes secciones veremos las técnicas más comunes que se suelen emplear para resolver cada uno de estos problemas.

2.1. Detección de fondo

El problema de estimar el fondo de una escena partiendo de una secuencia de fotogramas ha sido ampliamente tratado en la literatura. Hay varias maneras de abordarlo pero si analizamos los trabajos en la materia podemos ver que en general éstas se basan en modelar matemáticamente el valor de intensidad del píxel y su variación. Estos modelos matemáticos pueden ser paramétricos como una distribución normal, este es el caso de [C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, 1997] y [J. Hu, T. Su, and S. Jeng, 2006], o una mezcla de gaussianas como en [C. Stauffer and W. E. Grimson, 1999]. Otras soluciones se basan en modelos no paramétricos como las técnicas basadas en *codebooks* utilizadas en [K. Kim, T. H. Chalidabhongse, D. Harwood, L. Davis,, 2005] y [Amit Pal and Celebi, 2010] o las técnicas basadas en la estimación no paramétrica de funciones de densidad como en [A. Elgammal, D. Harwood, and L. Davis, 2000]. A continuación haremos un repaso breve de las bases teóricas detrás de cada técnica, sus ventajas y desventajas.

2.1.1. Distribuciones unimodales

En las técnicas basadas en distribuciones unimodales el fondo es modelado como una matriz con el mismo tamaño que la imagen de entrada y cuyas entradas representan la distribución estadística del píxel en cuestión. En este caso en concreto esta distribución está basada en una función unimodal que tiene un solo

máximo (o valor más frecuente). Una distribución normal es un ejemplo sencillo de este tipo de distribuciones, sin embargo una distribución normal no tiene porqué ser simétrica. En este contexto un píxel puede ser representado de varias maneras donde la más simple es utilizar la información de color de forma directa o separando la información de color y brillo como en [C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, 1997]. En este trabajo la media por cada píxel es actualizada utilizando la siguiente fórmula dónde el vector $y = (U, Y, V)$ representa la información de color por cada píxel.

$$\mu_t = \alpha y + (1 - \alpha) * \mu_{t-1} \tag{2.1}$$

[Orhan Bulan and, 2013] se basa en esta técnica para detectar el fondo en un sistema de detección automática de ocupación de plazas de aparcamiento. La ventaja de estas técnicas radica en un modelo matemático simple como consecuencia se pueden implementar de forma sencilla y no suelen requerir de soporte hardware para funcionar en tiempo real. Sin embargo, al basarse en un modelo unimodal no pueden representar fondos más complejos como una escena con árboles en movimiento u olas por ejemplo. Otro problema que tiene esta técnica es la dificultad a la hora de seleccionar el valor óptimo para la variable α. Un valor grande de este parámetro aumenta el peso de la imagen actual sobre el valor final del fondo. Esto favorece cambios muy rápidos en la escena (como cambios rápidos de iluminación), sin embargo puede introducir efectos no deseados como detectar parte de la imagen actual como fondo. Por otro lado, un valor pequeño de α reduce el peso de la imagen actual sobre el fondo, así que sólo incorporamos en el fondo un objeto que lleva varias iteraciones siendo parte del mismo (un vehículo parado por ejemplo). Sin embargo, la desventaja en este caso es una lenta reacción a cambios rápidos como cambios de iluminación. Estos problemas han sido reportados por varios trabajos basados en esta técnica como [Zhang et al., 2010]. Como consecuencia, el uso de este tipo de técnicas se limita a escenarios controlados como en interiores.

2.1.2. Mezcla de gaussianas

En la vida real, sobre todo en exteriores, los escenarios son mucho más complejos y la probabilidad de la aparición de objetos móviles en la escena, cambios de iluminación o cambios frecuentes (hojas de árboles, por ejemplo) es muy alta. Necesitamos un modelo de fondo que sea capaz de representar este tipo de cambios y tenerlos en cuenta a la hora de clasificar un píxel como estático u objeto en

movimiento. *MOG* (o *GMM* por sus siglas en inglés) es una técnica que ha sido inicialmente aplicada al problema de sustracción de fondo por [C. Stauffer and W. E. Grimson, 1999]. La idea básica de la misma es utilizar varias gaussianas para modelar cada píxel a diferencia de la técnica anterior que sólo utiliza una distribución. Esto hace posible modelar varios *estados* del fondo para el mismo píxel. Por ejemplo esta técnica es capaz de modelar un píxel cuyo valor está frecuentemente cambiando de un valor a otro como consecuencia de movimiento periódico en la escena. En *MOG* dada la historia reciente de un píxel, $X_1, ..., X_t$ la probabilidad de observar el valor actual del píxel se formula como:

$$P(X_t) = \sum_{i=1}^{k} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{2.2}$$

Donde k es el número de gaussianas, η es una gaussiana de media $\mu_{i,t}$ y cuya matriz de covarianza es $\Sigma_{i,t}$. El valor $\omega_{i,t}$ es una estimación de su peso que puede variar con el tiempo (representa que proporción aporta esta gaussiana en la decisión final). La función de densidad de probabilidad η tiene la siguiente forma:

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu_t)^T \Sigma^{-1} (X_t - \mu_t)} \tag{2.3}$$

El número de gaussianas k viene determinado por el número de *modos* posibles para el fondo determinados por el escenario. Este número afecta de forma directa al tiempo de procesamiento y a la memoria requerida para implementar el algoritmo. En la implementación básica de MOG se suele asumir que los canales (R,G,B) son independientes de ahí obtenemos una matriz de covarianza simplificada en la forma:

$$\Sigma_{k,t} = \sigma_k^2 I \tag{2.4}$$

En esta versión de *MOG* la pertenencia de cada nuevo píxel al fondo es evaluada contra las distintas gaussianas que lo forman. Un píxel pertenece a una gaussiana si su valor está comprendido en un rango de 2.5 su desviación estándar. Si el nuevo píxel pertenece a alguna de ellas entonces es clasificado como fondo y los parámetros $\mu_{k,t}$, $\sigma_{k,t}^2$ y $w_{k,t}$ de la gaussiana en cuestión son actualizados con el nuevo valor mediante las siguientes fórmulas:

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \tag{2.5}$$

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho(X_t) \tag{2.6}$$

$$\sigma^2 = (1 - \rho)\sigma^2_{t-1} + \rho(X_t - \mu_t)^T(X_t - \mu_t) \tag{2.7}$$

Donde:

$$\rho = \alpha\eta(X_t|\mu_k, \sigma_k) \tag{2.8}$$

La variable α representa el factor de aprendizaje mientras que $M_{k,t}$ es 1 para la gaussiana que describe el píxel y 0 para las demás. Los valores de μ y σ para el resto de las distribuciones no cambian. Después de esta actualización los valores de $w_{k,t}$ son normalizados entre las k distribuciones que modelan el píxel. Si el píxel no pertenece a ninguna de las distribuciones entonces la distribución de menor peso es remplazada por otra con una varianza inicial grande, una media igual al valor del nuevo píxel y un peso inicial bajo.

Normalmente no todas las gaussianas son utilizadas para formar parte del modelo de un píxel. En cambio, se escoge un número T de éstas para formarlo. Idealmente las gaussianas que forman parte del modelo representan algún modo del fondo, sin embargo dada la naturaleza cambiante del mismo sólo una parte de éstas en realidad contienen un modo válido para el fondo. Para facilitar el acceso a las mismas, las distribuciones son ordenadas en una lista en función del valor ω/σ de esta manera los componentes que tienen poca desviación y un alto peso en la mezcla de gaussianas son las primeras utilizadas para evaluar la pertenencia de un píxel al fondo. El resto de las distribuciones con pesos más bajos y altas variaciones se mantienen al final de la lista y son susceptibles a ser remplazadas con nuevas distribuciones.

Si repasamos la literatura de la última década podemos ver que la detección de fondo basada en la mezcla de gaussianas ha sido y sigue siendo uno de los modelos más utilizados por la mayoría de los trabajos en este campo. [Pablo Barcellos, 2014] y [Xia et al., 2014] basan su sistema de detección de vehículos mediante espirales virtuales en esta técnica. En la misma línea [Buch et al., 2009] presenta un sistema de clasificación basado en características 3DHOG que se basa en esta técnica para detectar el fondo. [Bjorn Johansson, Johan Wiklund, Per-Erik Forssén, Gösta Granlund, 2009] basa su trabajo de clasificación de vehículos y estimación automática de sombras en está técnica con ligeras modificaciones que

consisten en cambiar las reglas de actualización del fondo junto con el uso de límites de regularización más bajos.

2.1.3. Codebooks

Este tipo de técnicas han sido originalmente utilizadas para tareas de clasificación de texto. El problema consiste en clasificar un texto de entrada basándose en la frecuencia de aparición de ciertas palabras. Esta clasificación se hace en base a un diccionario de palabras que constituye el vocabulario empleado en un dominio en concreto. La elección de las palabras es una tarea clave y es la base de la clasificación. Si por ejemplo queremos detectar textos que hacen referencia a la bolsa un conjunto de palabras clave podría ser: acciones, inversión, activo, cotización, crédito, compra, venta, etc. Así que dependiendo del dominio objetivo del estudio haremos uso de unas palabras u otras. Estas se utilizan para formar los denominados *codebooks* (o vocabulario). Éste, se puede utilizar más tarde para calcular la frecuencia de aparición de las palabras clave en un texto de entrada formando un histograma del mismo. Éste, se puede utilizar para detectar si se trata de un texto relacionado con la bolsa o no.

Estas mismas técnicas se pueden aplicar en el contexto de la visión computacional, en concreto para clasificar imágenes o partes de las mismas. En vez de palabras en este caso tendremos *palabras visuales* que son básicamente vectores formados por elementos que describen ciertas características importantes de la imagen. En general, para reducir el número de palabras empleadas por cada elemento éstas son agrupadas en forma de *clusters* cuyos entradas son escogidas para formar parte del diccionario.

Los algoritmos de detección de fondo basadas en esta técnica se pueden considerar como un caso particular del problema general de clasificación. En este caso, cada píxel puede ser clasificado de dos maneras: como parte del fondo o como parte de un objeto en movimiento. Estas técnicas forman un *codebook* con un conjunto de palabras (*codewords*) que describen el píxel. Al tratarse de una técnica no paramétrica estas palabras no se pueden asociar con distribuciones gaussianas. En principio el número de palabras por cada píxel sólo está limitado por la cantidad de la memoria en el sistema, sin embargo, según [K. Kim, T. H. Chalidabhongse, D. Harwood, L. Davis,, 2005] seis palabras son suficientes para describir un píxel. En este trabajo como palabra los autores han utilizado un vector **RGB** $vi = (R_i, G_i, B_i)$ y una tupla $aux_i = [I_{min}, I_{max}, f_i, \lambda_i, p_i, q_i]$ donde:

- I_{min}, I_{max}: Son los valores mínimos y máximos respectivamente para cada

píxel asignado a esta palabra

- fi: Es la frecuencia de aparición de esta palabra
- λ_i: Es el máximo intervalo durante el cual esta palabra no ha aparecido
- pi, qi: Son la primera y la última vez que esta palabra ha aparecido

El valor λ en este trabajo se utiliza para filtrar las palabras que corresponden a objetos temporales que no forman parte del fondo. Éstos son fácilmente detectados ya que su λ_i normalmente es un valor grande en comparación con un umbral establecido. En este punto, hay que resaltar que un objeto con una baja frecuencia podría aún formar parte del fondo (un objeto que aparece de forma periódica pero a baja frecuencia). La estimación de fondo en este caso consiste en comparar cada píxel de la nueva imagen con la palabra cuya media es más cercana al mismo. La comparación en realidad se divide en dos partes: la primera compara la distancia en color (para los tres componentes RGB), y la segunda compara la distancia en la escala de brillo. Si el píxel pertenece a la palabra en cuestión entonces esta es actualizada, de otra manera se trata de un píxel que forma parte de un objeto en movimiento.

Pese a ser una de las técnicas más famosas para la detección de fondo, si analizamos la literatura de clasificación y detección automática de vehículos podemos ver que esta técnica no ha sido adoptada para la detección de fondo. Podemos encontrar pocos trabajos donde la detección está basada totalmente en ésta técnica como en [M. Mazaheri, 2011]. Otros trabajos como [Mandellos et al., 2011] tan sólo utilizan esta técnica para comparar sus resultados contra los obtenidos por su técnica.

2.1.4. Métodos basados en la estimación no paramétrica de la función de densidad

La estimación no paramétrica de la función de densidad es una técnica estadística para hallar la función de densidad desconocida de una variable aleatoria a partir de un conjunto finito de muestras. El caso más sencillo y popular de esta técnica son los histogramas. En este caso los datos son representados con una serie de barras cuya altura (cuando trabajamos en una dimensión) es proporcional al número de muestras que caen en el rango representado por la misma. Cuando los histogramas son empleados como estimadores de densidad tienen que estar normalizados para integrar a uno. Se trata de un método sencillo que sirve para

hacerse una idea rápida sobre la forma que tiene la distribución de los datos de entrada. El problema que tiene esta representación es que dependiendo del tamaño del rango escogido para la cuantización el resultado puede cambiar. Para resolver este problema, la alternativa es utilizar una función matemática cuyo resultado es una representación más suave de los datos. Esta función se llama el *núcleo*. Dado un conjunto de valores de entrada $(X_1, X_2, ..., X_n)$ provenientes de una distribución desconocida f, su *núcleo* tiene la siguiente forma:

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - x_i}{h}) \tag{2.9}$$

Donde K es una función no-negativa y que integra a uno. El parámetro h es un parámetro positivo que se denomina el *ancho de la ventana*.

Esta técnica ha sido aplicada al problema de estimación de fondo por [A. Elgammal, D. Harwood, and L. Davis, 2000]. En este caso el autor como núcleo hace uso de una gaussiana $N(0, \Sigma)$, donde Σ representa el ancho de la ventana del estimador. En este caso la ecuación 2.9 se puede escribir:

$$Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_t - x_i)^T \Sigma^{-1}(x_t - x_i)} \tag{2.10}$$

Asumiendo que los canales de color son independientes, con un ancho de ventana σ_j distinto por cada canal, entonces:

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & 0 \\ 0 & 0 & \sigma_3^2 \end{bmatrix}$$

De esta manera la ecuación de la función de densidad se puede simplificar en:

$$Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N}\prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(x_{tj} - x_{ij})^2}{2\sigma_j^2}} \tag{2.11}$$

Utilizando esta ecuación un píxel se considera parte de un objeto en movimiento si $Pr(x_t) < th$ dónde th es un umbral fijo y global aplicado a todas las imágenes. Según [A. Elgammal, D. Harwood, and L. Davis, 2000] esta técnica obtiene mejores resultados que la mezcla de gaussianas en general ya que es capaz de manejar cambios muy rápidos de iluminación mejor que esta última. Esta técnica se puede considerar una generalización de la mezcla de gaussianas donde cada dato se puede considerar una distribución gaussiana $N(0, \Sigma)$. Según el mismo autor, bajo las mismas condiciones de memoria y ajustando ambos modelos para obtener el mismo ratio de falsos-positivos la técnica basada en estimación

de densidad obtiene mejores resultados que la técnica basada en la mezcla de gaussianas.

2.2. Seguimiento de vehículos

El problema de seguimiento de objetos en una secuencia de imágenes ha sido ampliamente tratado en la literatura [Yilmaz et al., 2006]. El seguimiento se puede definir como la tarea de estimar la localización de un objeto en movimiento en el plano imagen a medida que éste se mueve por la escena. Este análisis podría ser utilizado a posteriori para analizar la trayectoria del objeto y estudiar su comportamiento. Aunque se trata de una tarea bastante sencilla para los humanos dado nuestro complejo sistema de visión, este problema cobra vital importancia en sistemas automáticos donde la tarea pasa a ser mucho más compleja debido a factores como el cambio de iluminación, tamaño y forma de los objetos, el ángulo de visión o el fondo de la escena. Detectar este tipo de eventos para el ojo humano es trivial sin embargo para una máquina es todo un desafío. Entre los distintos problemas que hay que resolver en este caso podemos encontrar:

- Efectos del clima: La calidad de la imagen obtenida por una cámara cambia bajo efectos adversos del clima como la lluvia o la nieve
- Cambios bruscos de iluminación: el ojo humano se caracteriza por una capacidad enorme para adaptarse a cambios de iluminación. Las cámaras no tanto. Así que en nuestro escenario en particular podemos sufrir cambios bruscos de iluminación provocados por la entrada de objetos blancos en la escena. Es un caso bastante habitual por ejemplo cuando entra un camión (de color blanco en general) en la escena.
- Pérdida de perspectiva 3D: las cámaras proyectan el mundo 3D en una imagen 2D. Durante esta operación se pierde una dimensión. Este efecto provoca que el tamaño del mismo vehículo va cambiando a medida que éste se aleja de la cámara.
- Movimiento de la cámara: incluso cuando la cámara está bien fijada sobre un soporte solido, ésta puede sufrir movimientos o vibraciones causados por el paso de vehículos pesados, fuerte viento, etc. Esto puede provocar movimientos inesperados en el fondo de la escena.
- Oclusiones: es uno de los problemas más importantes que hay que resolver cuando estamos tratando de seguir varios objetos a la vez. Se trata de un

problema común en el caso del tráfico en autopistas ya que los vehículos pueden tapar unos a otros sobre todo en escenarios de alta densidad de tráfico.

El problema de seguimiento de objetos ha sido ampliamente tratado en la literatura y se han propuesto multitud de métodos para resolverlo. Como no podemos ir uno por uno sobre todos los trabajos relacionados con este tema a continuación vamos a enumerar las técnicas más utilizadas para intentar resolverlo.

2.2.1. Seguimiento basado en regiones

Esta familia de técnicas está basada en el seguimiento de regiones conexas del objeto. Estas partes normalmente están caracterizadas por una propiedad como el color que es la característica más común en estos métodos. En la literatura podemos encontrar una variedad de espacios de color utilizados en estas técnicas desde espacios no uniformes y simples como RGB, hasta espacios uniformes pero más complejos como CEI lab o CEI LUV. El espacio de color HSV es otra buena alternativa data su capacidad para separar entre la información de saturación e intensidad del color. Entre estas técnicas las más comunes son las que se basan en histogramas de color para representar las regiones. Esta familia de técnicas ha sido inicialmente introducida por [Dorin Comaniciu and Meer, 2003]. En el trabajo original, el seguimiento se basa en la comparación de histogramas de nuevas imágenes con el histograma de referencia calculado sobre la región de interés. La comparación está basada en una métrica derivada de la distancia de *Bhattacharyya* mientras que *mean-shift* es utilizado para optimizar la selección del candidato.

En la misma categoría, recientemente [Duffner, S. ; LIRIS, Univ. de Lyon, Lyon, France ; Garcia, C., 2013] presentaron un algoritmo capaz de seguir objetos en tiempo real incluso en condiciones desfavorables como fondo cambiante y oclusiones. El algoritmo bautizado como *PixelTrack* combina un detector basado en la transformada de Hough con un modelo genérico de detección de fondo. Ambos se retro-alimentan para seguir el objeto. Este algoritmo necesita como inicialización una ventana sobre el objeto a seguir y sólo puede seguir un objeto al mismo tiempo. En [Lili Huang, 2010] los autores combinan dos algoritmos para llevar a cabo seguimiento de vehículos y resolución de oclusiones. En la parte que nos interesa, en esta sección hacen uso un modelo de color basado en *mean-shift* para determinar a qué vehículo pertenece cada parche (de 3x3 píxeles) una vez detectada una oclusión

2.2.2. Seguimiento basado en características

A diferencia de la técnica anterior, en este caso el objetivo es realizar el seguimiento del objeto a base de seguir ciertas características que le definen. La diferencia entre los métodos que pertenecen a esta familia radica en el conjunto de características elegido. Éstas pueden ser simples como puntos, líneas y esquinas o estructuras más complejas como perímetros, elipses, etc. Este tipo de características son en general más robustas que las basadas en texturas o color ya que son más resistentes a cambios en el fondo, la iluminación de la escena o oclusiones parciales. Entre las técnicas más utilizadas en este contexto podemos citar a SIFT [Lowe, 2004] y KLT [Shi, J. ; Tomasi, C., 1994]. Otra característica interesante es *HOG* (Histogramas de Gradientes Orientados) [Dalal N. and Triggs B, 2005] que siendo similar a SIFT en la base teórica y la construcción ha sido utilizada ampliamente para tareas de seguimiento y clasificación dando buenos resultados en combinación con el clasificador SVM como ha sido reportado por [Zezhi Chen, Tim Ellis, Sergio A Velastin, 2012]. De hecho, si analizamos las publicaciones más recientes sobre sistemas de reconocimiento la combinación de SVM y HOG es de las técnicas más utilizadas en la comunidad científica.

2.2.3. Seguimiento basado en modelos

Esta familia de técnicas trata de aprovechar el conocimiento a priori de los objetos, en este caso los vehículos, para tratar de modelarlos con plantillas en 3D o 2D. Éstas son utilizadas para detectar los vehículos en la imagen para poder seguirlos. Por ejemplo, [Matthew J. Leotta and Joseph L. Mundy, 2011] se basa en esta técnica para modelar los vehículos utilizando una plantilla genérica deformable capaz de ajustarse para simular varias formas de vehículos. Para el seguimiento los autores combinan esta técnica con un filtro extendido de Kalman. En la misma línea [Marcos Nieto, Luis Unzueta, Javier Barandiaran, Andoni Cortés, Oihana Otaegui and Pedro Sánchez, 2011] trata de estimar la forma 3D del vehículo de forma dinámica sobre un espacio de búsqueda formado tanto por las posibles posiciones x,y del vehículo como por posibles tamaños WHL. El estado del sistema en este caso está modelado como un solo vector cuya distribución a posteriori es estimada mediante un filtro de partículas basado en MCMC.

2.3. Clasificación de vehículos

El objetivo principal de esta tesis es construir un sistema automático para la clasificación de vehículos basado únicamente en visión. En la literatura podemos encontrar muchos trabajos relacionados con el mismo, algunos de ellos como [Kanwal Yousaf, 2007] y [Zezhi Chen, Tim Ellis, Sergio A Velastin, 2011] se han encargado de hacer un listado muy extenso de todos estos trabajos incluyendo comparaciones entre los resultados obtenidos por cada uno de ellos. El análisis que haremos a continuación en vez de listar todos y cada uno de los trabajos se centra más bien en describir las técnicas empleadas en la mayoría de estos trabajos y sus bases teóricas.

Si analizamos los trabajos relacionados con la clasificación automática de vehículos veremos que las técnicas empleadas se pueden organizar en dos categorías:

- Técnicas basadas en características
- Técnicas basadas en modelos 3D

Las técnicas basadas en características se apoyan en la extracción de ciertas propiedades geométricas o visuales del vehículo y utilizarlas para llevar a cabo la tarea de la clasificación. Estas técnicas en general parten de la definición del conjunto de características que serán utilizadas para distinguir entre las distintas categorías de vehículos. Estas pueden ser propiedades simples como la longitud o más complejas como características visuales, estructurales o cualquier tipo de propiedad que se pueden extraer de forma automática de las imágenes de entrada. Cuantas más características utiliza el sistema más robusta será su capacidad de discriminación entre las distintas categorías. En general, esta familia de técnicas suele requerir de una fase previa de aprendizaje donde el sistema es entrenado sobre un conjunto de imágenes que contienen las distintas categorías de vehículos a clasificar. La fase de entrenamiento contiene normalmente tanto ejemplos positivos como negativos. Durante el funcionamiento normal, el sistema analiza el vídeo de entrada en busca de vehículos. Éstos son procesados para extraer las características base de la clasificación y pasarlas como entrada a un algoritmo clasificador que estima cuál es la categoría más cercana al vehículo.

La segunda familia de técnicas son las basadas en modelos. Éstas utilizan una serie de modelos 3D para representar las distintas categorías de los vehículos. En general, en este tipo de técnicas se asume el uso de una cámara cuyos parámetros

de calibración son conocidos de antemano. La clasificación en este caso se basa en la comparación de la *huella* producida por el vehículo con las huellas generadas por cada una de las plantillas de los modelos que representan las distintas categorías. El resultado de esta comparación es utilizado para estimar a qué categoría pertenece el vehículo bajo análisis.

En la siguientes secciones veremos las distintas técnicas más recientes que han sido presentadas en cada categoría.

2.3.1. Técnicas basadas en características

Como ya hemos comentado anteriormente, estas técnicas tratan de clasificar los vehículos en base a alguna característica física. Después de haber analizado muchos trabajos basados en esta técnica podemos afirmar que no hay una serie de propiedades universales utilizadas para identificar a los vehículos. Aunque éstos en general tienen una serie de componentes físicos que les distinguen visualmente de otros objetos (ruedas, luces, parabrisas, etc) no es fácil encontrar patrones comunes que valgan de forma genérica para cualquier vehículo en cualquier escenario. En este sentido, cada uno de los trabajos trata de definir un número finito de categorías relacionadas con el problema de clasificación particular bajo estudio. Éstas pueden ser categorías simples como: *vehículos cortos* y *vehículos largos* (como en [Guohui Zhang, 2007] y [Yang et al., 2011]) o categorías concretas como: coche, moto, autobús y camión. Sin embargo, si analizamos las características en sí, éstas se pueden dividir en dos categorías:

- Geométricas: Longitud, área, perímetro, relación entre alto y ancho, etc.
- Características visuales

Técnicas basadas en característica geométricas

Estas características se pueden extraer de la imagen de forma directa (longitud en píxeles) o empleando un paso adicional para hacer mediciones métricas sobre la imagen. En este caso para llevar a cabo las mediciones necesitamos disponer de una cámara calibrada o de una técnica similar como la rectificación de planos. En esta categoría, [Shih-Hao Yu and Hu, 2003] presenta un sistema de clasificación basado en dos características: tamaño y *linealidad*. Esta última propiedad es empleada por los autores para diferenciar entre camiones y autobuses. El sistema es capaz de distinguir entre tres categorías de vehículos: coches, camiones y autobuses. Este trabajo presenta un sistema para detectar de forma automática los

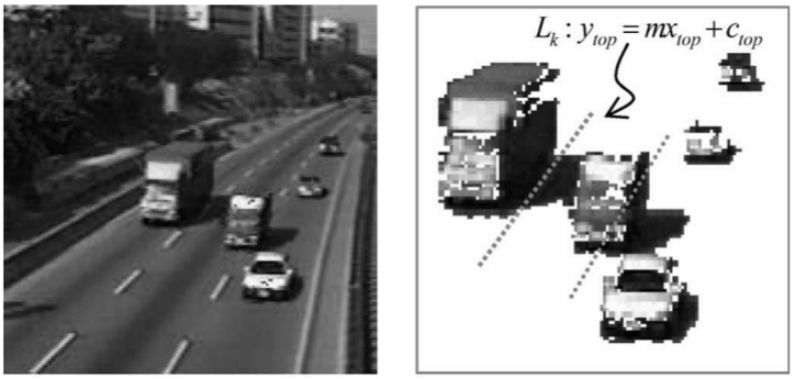


Figura 2.1: Utilización de la información de carriles para eliminar las sombras y resolución de oclusiones

carriles analizando las trayectorias de los vehículos (figura 2.1) para emplearlos posteriormente en la eliminación de sombras y resolución de oclusiones .

[Guohui Zhang, 2007] y [Yang et al., 2011] presentan un sistema para clasificar vehículos basado en la longitud. Ambos sistemas no requieren una cámara calibrada, puesto que hacen uso de la longitud en píxeles en vez de la longitud real del vehículo. Para ello, se utiliza una línea virtual en la imagen para medir la longitud de los vehículos al atravesarla. Esta medida es comparada con un umbral establecido previamente para diferenciar entre las dos categorías soportadas por estos sistemas: SV (vehículos cortos) y LV (vehículos largos).

Siempre dentro de la misma categoría [Jin-Cyuan Lai and Shih-Shinh Huang and Chien-Cheng Tseng, 2010] presenta un sistema para clasificar los vehículos en tres categorías: coche, camión y autobús. La clasificación se basa en dos medidas: compacidad y la proporción entre alto y ancho de los blobs. Este trabajo utiliza un grafo para resolver el problema de asociación de vehículos entre los distintos fotogramas donde el peso de los arcos es la distancia entre los centroides de los vehículos. Además, se emplea un filtrado de blobs basado en restricciones geométricas para eliminar blobs muy pequeños y ruido en la escena.

Finalmente [H. Asaidi a,n, A. Aarab b, M. Bellouki c, 2014] presentan un sistema que se basa en siete características geométricas para clasificar los vehículos en tres categorías: coche, furgoneta, camión. Las características elegidas para la clasificación incluyen el área del objeto, el centro de masas y los momentos de inercia. Éstas, son utilizadas en una fase previa para construir una base de datos de las categorías soportadas. La fase de clasificación consiste en extraer las características para el vehículo detectado y compararlas con la base datos mediante el cálculo de la distancia euclídea entre el vehículo y las distintas categorías previamente establecidas. El vehículo es asignado a la categoría con la menor distancia

a su vector de características.

Técnicas basadas en característica visuales

El mundo de las características visuales es en principio más complejo dada la dificultad para *describir* la forma o el aspecto que tienen los vehículos en particular y cualquier otro objeto en general dentro de una imagen. En este caso cabe señalar que la mayoría de los descriptores que hemos estudiado han sido originalmente introducidos para detectar humanos. Dos buenos ejemplos son el descriptor *Haar* introducido por [Papageorgiou et al., 1998] para detectar caras humanas o el descriptor *HOG* propuesto originalmente por [Dalal N. and Triggs B, 2005] para detectar humanos en una imagen. Podemos encontrar otras variantes creadas a partir de este descriptor como *3DHOG* o *IPHOG*. Estos descriptores han sido aplicados con éxito al mundo de la clasificación de vehículos dada su versatilidad y su capacidad para resumir y codificar la forma de un objeto de forma robusta a cambios en la intensidad o aspecto del objeto en la imagen. De los trabajos analizados hemos visto que *HOG* es uno de los descriptores más utilizados en este contexto en combinación con algún clasificador. En general, según la bibliografía consultada la combinación de *HOG* con un clasificador da lugar a buenos resultados.

En el mismo contexto [Yang Lv and Zhu, 2012] presenta un sistema de detección y clasificación de vehículos basado en plantillas configurables. La idea es representar la composición física de la parte frontal o trasera del vehículo en base a un conjunto de plantillas que no se solapan. Las características de las plantillas en realidad son descriptores HOG. La figura 2.2 muestra un ejemplo de las plantillas creadas como parte de este trabajo. El sistema presentado emplea un algoritmo de búsqueda *And-Or* para encontrar las distintas configuraciones posibles que mejor representan la ventana de búsqueda (escogida sobre la delantera o el trasero del vehículo). Cada una de las distintas configuraciones enumeradas por este algoritmo generan una puntuación. La configuración con la máxima puntuación determina la categoría del vehículo.

[Bailing Zhang, Yifan Zhou and Hao PanTammam Tillo, 2013] presenta una técnica de clasificación basada en un *KAA (Kernel Auto Associator)*. Para la detección de vehículos el sistema se basa en el descriptor HOG combinando con el clasificador SVM. Este sistema emplea una ventana de búsqueda para encontrar el vehículo dentro de la imagen de entrada. La capacidad de detección de

Figura 2.2: Plantillas reconfigurables basadas en HOG

los vehículos es mejorada mediante el uso de varios clasificadores SVM en cascada. La fase de clasificación se basa en un KAA usando *EOH (Edge Orientation Histogram)* como característica discriminante. El autor hace uso de EOH ya que su computación es más rápida que HOG para imágenes de grandes dimensiones. Cabe destacar que este sistema necesita una fase previa de entrenamiento para aprender las distintas clases de vehículos por clasificar.

Otra de las técnicas que ha sido introducida originalmente para detectar y reconocer caras humanas y que ha sido aplicada al campo de clasificación de vehículos es la técnica de *eigenfaces* ([Sirovich, 1987]). En realidad comparada con HOG podemos decir que esta técnica no ha sido adoptada de forma masiva por la comunidad científica para el campo de clasificación de vehículos. En esta línea de trabajo, [Wei Wang, 2011] presenta un sistema de clasificación de vehículos basado en la técnica de *eigenfaces*. Los autores hacen uso de la misma base teórica para construir un subespacio de vectores con características de los vehículos y utilizarlo posteriormente para su clasificación. El proceso consiste en capturar la *cara* del vehículo, aplicarle unas cuantas operaciones de post-procesado para poder utilizarla como entrada al clasificador. La clasificación consiste en proyectar

Figura 2.3: Parte frontal de vehículos empleada para generar una base de datos de sus *eigenfaces*

esta cara en el subespacio construido en la fase de aprendizaje y comparar esta proyección con las recopiladas en la base de datos construida durante la fase de aprendizaje. Si la comparación está por debajo de un umbral establecido entonces la clasificación es correcta, de otra manera, es un vehículo desconocido. Cabe destacar que este trabajo sólo hace a coches obviando el resto de categorías de los vehículos. La figura 2.3 muestra ejemplos de *caras* de vehículos capturados por el sistema.

En la misma línea de trabajo [Robert, 2009] presenta un sistema para detectar y clasificar los vehículos de noche. Este método se base en la detección de las luces de los vehículos. Éstos son clasificados en primera instancia en base a la distancia entre sus luces y el diámetro de los mismos. También tiene en cuenta la presencia del parabrisas. Su presencia confirma la detección de un coche, en este caso se pasa a la fase de clasificación de forma directa. De otra menara, la imagen del vehículo recortada entorno a las luces es procesada nuevamente (después de reducir la dimensión del vector de características mediante un PCA) por dos clasificadores más potentes: el sistema soporta tanto una red neuronal como un clasificador SVM. En ambos casos este sistema hace uso de un *eigencar* para representar coches y un *eigenHV* para representar vehículos pesados. La imagen recortada es multiplicada por ambos para obtener el vector de características y compararlo con los generados en la fase de aprendizaje.

Técnicas híbridas

Hasta ahora hemos visto las técnicas basadas en características geométricas y las que se basan en propiedades visuales. Las primeras tienen varias ventajas como una fácil extracción de características, son fáciles de entender y mapear al mundo real y una gran versatilidad dado que podemos reutilizar las mismas herramientas y algoritmos para otras categorías de vehículos u objetos. Sin embargo, entre sus desventajas podemos citar su limitación para describir los objetos o diferenciar entre varias categorías sobre todo cuando hay más de una categoría con la misma medida. En el contexto de tráfico urbano por ejemplo si queremos diferenciar entre taxis y coches normales no podremos utilizar la longitud (ni otras medidas geométricas). En este caso la única diferencia en general suele ser el color. Otro de los problemas que sufren estas características es su sensibilidad frente a cambios en la posición de la cámara (en el caso de no disponer de una cámara calibrada). Las características visuales presentan ventajas como una representación más abstracta del objeto que recoge muchos detalles acerca del mismo, se pueden combinar fácilmente con un clasificador como SVM dando lugar a un sistema de clasificación robusto. Sin embargo, para llevar esto a cabo estas técnicas necesitan una fase de aprendizaje previa donde el sistema es entrenado con muestras representativas escogidas cuidadosamente para evitar problemas de sobre-entrenamiento. Adicionalmente, el conjunto de características visuales escogido suele ser particular del problema de clasificación y del entorno escogido. Como consecuencia, la reutilización o la aplicación de la misma técnica a otro entorno con otras características suele requerir muchos cambios. Esto sin duda es una desventaja frente a las técnicas basadas en propiedades geométricas dado que éstas no suelen necesitar una fase de entrenamiento. Otra de las desventajas de las características visuales es su complejidad (en comparación con las medidas geométricas) que hace que sean más difíciles de entender para un ser humano y de extraer para una máquina. En la práctica, esto se traduce en costes más altos de procesamiento si las comparamos con las propiedades geométricas. Esto limita su uso en aplicaciones de tiempo real que requieren imágenes de entrada de altas resoluciones.

Las técnicas híbridas tratan de combinar características geométricas junto con características visuales para aprovechar las ventajas de cada una de ellas. Como resultado obtenemos sistemas más complejos pero más robustos. En este contexto [Zezhi Chen, Tim Ellis, 2011] presenta un sistema para clasificar vehículos basado

Figura 2.4: Muestras de entrada utilizadas en la fase de entrenamiento

en características híbridas. El sistema hace uso de dos conjuntos distintos de características. El primero de ellos *MBF* (*Measurement Based Features*) compuesto por varias características entre ellas podemos encontrar: el área, el perímetro, ancho, largo y así hasta un conjunto de 13 características. El segundo conjunto empleado por este sistema es el llamado *IPHOG*. Se trata de un descriptor espacial que codifica la forma y la distribución de los objetos dentro de la imagen. El sistema hace uso de PCA para reducir la dimensión del conjunto de características compuesto por *MBF* y *IPHOG* a tres dimensiones. El sistema necesita de una fase de aprendizaje donde las características son extraídas manualmente y utilizadas como entrada a un clasificador SVM. En la figura 2.4 podemos ver un ejemplo de estas muestras. Posteriormente, en el funcionamiento normal el SVM es utilizado para encontrar el vehículo más cercano al objeto bajo análisis empleando el conjunto de características anteriormente mencionado.

[Tao Wang, 2013] presenta un sistema capaz de clasificar los vehículos en varias categorías empleando para ello un enfoque multimedia que hace uso tanto de características visuales como sonoras de los vehículos. En la parte que nos interesa las características visuales utilizadas son: la proporción, tamaño, contorno y un descriptor HOG. Estas características junto con las sonoras son empleadas por un SVM para resolver la categoría del vehículo.

Finalmente [Niluthpol Chowdhury Mithun, Nafi Ur Rashid, and S. M. Mahbubur Rahman, 2012] presenta un sistema para clasificación de vehículos basado en el uso de *TSI* (*Time-Spacial Image*). Varias líneas virtuales de detección son

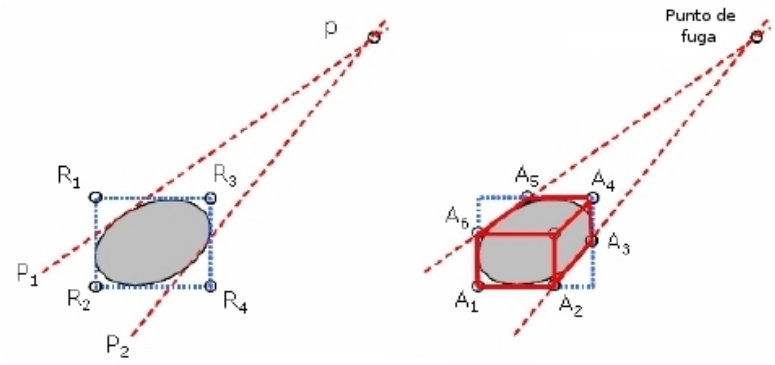


Figura 2.5: Reconstrucción 3D empleando puntos de fuga

utilizadas para capturar los vehículos en la escena. Este sistema ha sido probado hasta con siete distintas categorías de vehículos. La categoría del vehículo es finalmente estimada mediante un *KNN (K Nearest-Neighbour)* de dos pasos que trata de encontrar el vehículo más cercano basándose en una serie de características invariantes basadas en la forma y la textura del vehículo.

2.3.2. Técnicas basadas en modelos 3D

Las técnicas basadas en modelos 3D tratan de aprovechar el conocimiento a priori de las distintas categorías de vehículos. La idea básica es modelar cada una de éstas en forma de un modelo 3D que recoge la forma del vehículo. Estos modelos pueden ser desde simples paralelepípedos rectangulares hasta modelos CAD complejos que recogen todos los detalles del vehículo. Optar por un modelo u otro dependerá de los requisitos de cada aplicación y de cuánta granularidad necesitamos a la hora de la clasificación. Cuantas más categorías queremos soportar más detallado tiene que ser el modelo elegido. Por ejemplo, si queremos distinguir entre varios tipos de berlinas necesitamos disponer de varios modelos que describen cada una de ellas en detalle. Dado que el objetivo principal de esta tesis es clasificar vehículos utilizando una técnica basada en modelos vamos a describir en detalle las distintas técnicas que han sido utilizadas en este ámbito durante los últimos años.

[sun Shin et al., 2009] presenta un sistema basado en plantillas 3D que no necesita una cámara calibrada. La idea en este caso es reconstruir la forma 3D de cada vehículo empleando los puntos de fuga de los carriles. La figura 2.5 muestra un ejemplo de la reconstrucción. El sistema de clasificación se basa en el algoritmo C4.5 [Quinlan, 1993] de construcción de árboles de decisión para aprender y clasificar las siluetas de los vehículos. En la figura 2.6 se puede ver un

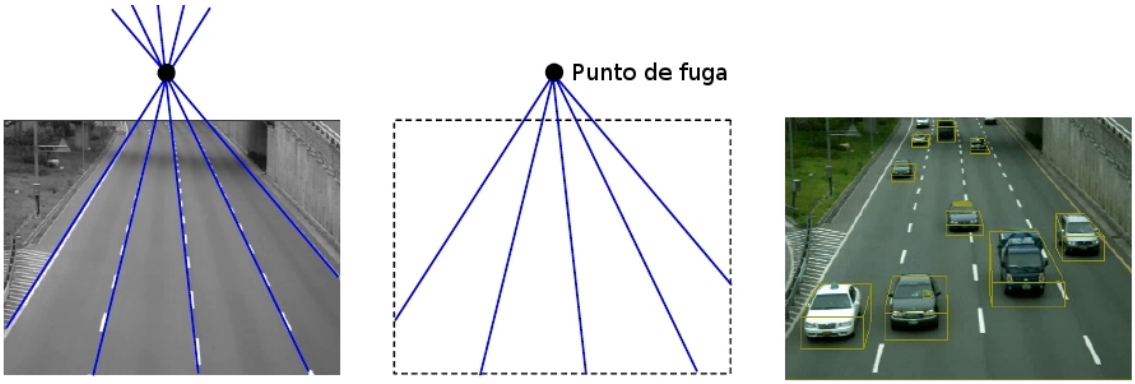


Figura 2.6: Clasificación basada en modelos 3D construidos mediante el uso de puntos de fuga

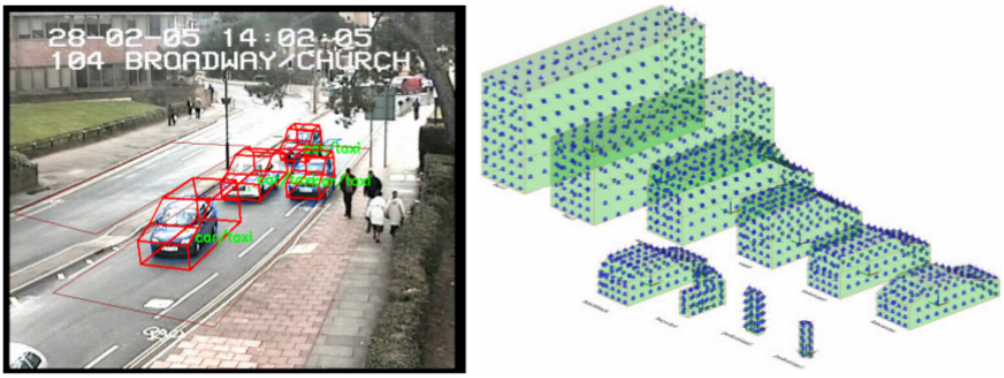


Figura 2.7: Plantillas con información 3DHOG para clasificación de vehículos

ejemplo de clasificación de vehículos empleando la técnica descrita en el artículo.

[Buch et al., 2009] presenta un sistema que se basa en histogramas *3DHOG*. Básicamente son histogramas HOG aplicados a plantillas 3D que representan los vehículos. Este trabajo tiene como objetivo clasificar los usuarios de una carretera urbana incluyendo peatones. Dada la dificultad y las formas variadas que pueden tener los vehículos de las distintas categorías, este trabajo trata de implementar una técnica robusta a cambios en el ángulo de vista. Para ello, el sistema define una plantilla 3D simplificada para cada categoría que el sistema tiene que detectar incluyendo los peatones. La figura 2.7 muestra un ejemplo de las plantillas definidas junto con su proyección sobre un escenario real. El sistema requiere de una fase de entrenamiento donde varias muestras para cada uno de los vehículos es segmentada de forma manual calculando su correspondiente histograma 3DHOG.

Durante el funcionamiento normal el sistema detecta los vehículos y proyecta las distintas plantillas 3D sobre ellos. La idea es recrear la forma 3D del vehículo mediante transformaciones afines de los distintos rectángulos que componen la proyección. Una vez reconstruido el modelo 3D del vehículo, se calculan sus histo-

Figura 2.8: Reconstrucción del histograma 3DHOG a partir de una imagen de entrada

gramas 3DHOG y se comparan contra los modelos que el sistema había aprendido previamente. La figura 2.8 muestra un ejemplo de este proceso.

[Matthew J. Leotta and Joseph L. Mundy, 2011] presenta un sistema de clasificación de vehículos basado en un modelo 3D deformable. El objetivo de este trabajo es crear un modelo intermedio que recoge los detalles del vehículo sin llegar a ser un CAD pero tampoco tan simple como un paralelepípedo. Para ello, el autor presenta un modelo formado por dos partes: Un malla 3D compuesta por cinco partes: una para modelar el cuerpo del vehículo y otras cuatro para las ruedas. La segunda parte es un conjunto de polígonos 2D ajustables para representar las partes del vehículo que pueden estar formadas por otros materiales distintos a los utilizados en el cuerpo. Éstas básicamente son utilizadas para representar los faros del vehículo, la matrícula, el techo solar y el parabrisas. La figura 2.9 muestra el modelo 3D utilizado en este trabajo. Estas partes son flexibles y se pueden deformar para ajustarse a distintas berlinas. La clasificación es llevada a cabo mediante el ajuste del modelo deformable al vehículo detectado en la imagen. La fase de detección se basa en el flujo óptico (utilizando KLT en este caso) para detectar y estimar la orientación del vehículo. A continuación, el sistema trata de capturar la forma del vehículo incluyendo su sombra calculando el desplazamiento de los vectores del flujo óptico. Finalmente, una vez extraído el contorno la imagen es analizada para extraer los bordes del vehículo y el sistema intenta ajustar el modelo 3D para hacerlo coincidir con ellos. El problema de ajustar el modelo es convertido en un problema de minimización iterativo Gauss-Newton, en cada iteración el error entre el mapeo del actual conjunto de bordes y de los formados por el modelo es reducido. Este sistema está pensado sólo para modelos de coches pequeños, de ahí, según lo presenta el autor no vale para representar vehículos de tamaño grande como furgonetas, autobuses o camiones. Esto reduce bastante el número de categorías de vehículos que se pueden clasificar utilizando

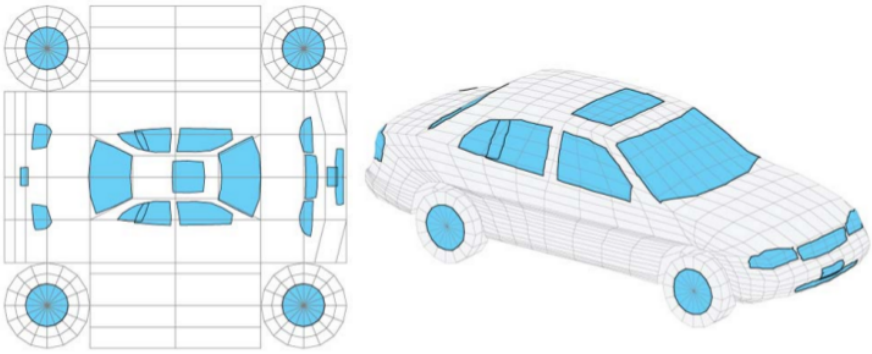


Figura 2.9: Modelo 3D compuesto por cinco partes: una para modelar el cuerpo del vehículo y otras cuatro para las ruedas

este método. Tampoco tiene en cuenta la interacción entre varios vehículos, un tema básico a la hora de clasificar vehículos en tráfico urbano o en autopistas. Este punto se deja como una posible mejora para trabajos futuros.

[haoxiang Zhang, Tieniu Tan, Kaiqi Huang, and Yunhong Wan, 2012] presentan un sistema de clasificación de vehículos basado en un modelo 3D genérico deformable para representar los vehículos. Se trata básicamente de un modelo 3D que emplea 12 parámetros para representar una gran variedad de vehículos, desde coches hasta modelos más complejos como furgonetas, mini-buses, *pick-ups*, etc. Este sistema es similar al presentado por [Matthew J. Leotta and Joseph L. Mundy, 2011] sin embargo tiene la ventaja de ser capaz de representar una gran variedad de vehículos de distintos tamaños. Además de clasificar el vehículo este sistema trata de estimar la posición (x,y,θ) del mismo. Para ello, hace uso del algoritmo dinámico EDA para encontrar el modelo que mejor *encaja* con la proyección del vehículo comparando la proyección de los distintos modelos con los datos de la imagen. La figura 2.10 muestra el modelo genérico empleado por este sistema deformado para representar varias categorías de vehículos.

[Marcos Nieto, Luis Unzueta, Javier Barandiaran, Andoni Cortés, Oihana Otaegui and Pedro Sánchez, 2011] presenta un sistema para seguir y clasificar vehículos utilizando varios modelos 3D. Como requisito preliminar, este sistema requiere de una fase previa de calibración donde cuatro puntos que forman un rectángulo son elegidos por el usuario además de especificar dos medidas de referencia sobre la carretera. Este paso es necesario para poder rectificar el plano de imagen y deshacer el efecto de la proyección de la cámara. A partir de este momento, todas las operaciones y el procesamiento posterior son realizadas en el plano rectificado de la imagen. El sistema representa el estado global incluyendo

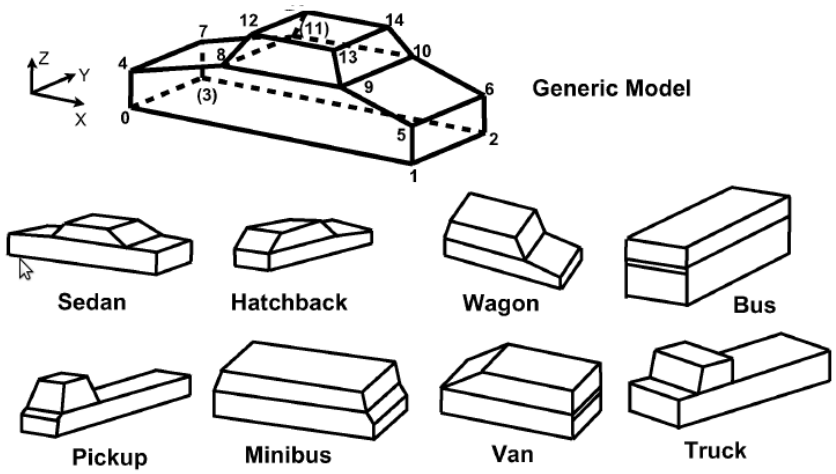


Figura 2.10: Modelo de vehículo 3D genérico para representar varias categorías

los vehículos, sus posiciones y tamaño con un solo vector de estado. Para evaluar la mejor distribución a posteriori el sistema emplea un filtro de partículas MCMC que se basa en el método de *Gibbs* para el muestreo. El estado de cada vehículo está formado por su posición (x,y) y su tamaño WHL. De esta manera, según los autores, se consigue manejar de forma sencilla la entrada y salida de la escena ya que durante estas áreas en particular el vehículo cambia de tamaño (sobre todo de longitud) hasta llegar a su tamaño real. El sistema es capaz de detectar vehículos que desparecen durante la fase de seguimiento (si un vehículo permanece en esta situación durante un número determinado de iteraciones entonces es eliminado de forma automática) sin embargo, no hay un tratamiento explicito de oclusiones. La figura 2.11 muestra un ejemplo de clasificación de este trabajo.

Figura 2.11: Clasificación de vehículos basada en plantillas 3D

Capítulo 3

Diseño global del sistema

Este capítulo presenta el diseño global de todo el sistema. Para ello, además de presentar el diagrama de bloques del mismo, hace un análisis detallado de la composición física del escenario incluyendo los elementos que componen el mismo y sus características físicas. Se profundiza en la disposición y configuración de la cámara como sensor principal. En este apartado se presenta en detalle la implementación del subsistema de calibración semi-automática de cámara y los distintos parámetros que intervienen en todo el proceso. Finalmente, la última sección del capítulo presenta la base de datos propia que ha sido recopilada como parte de esta tesis y la caracterización que se ha llevado a cabo de la misma.

3.1. Diseño

TrafficMonitor es un sistema para la monitorización del tráfico de vehículos en tiempo real en una autopista, capaz de seguir y clasificar vehículos en cinco categorías: motocicleta, coche, furgoneta, autobús y camión. Se trata de un sistema híbrido que combina modelos 3D junto con una técnica basada en HOG y SVM para llevar a cabo estas tareas. El sistema emplea la información de tamaño 3D de los vehículos como primer criterio para la clasificación, adicionalmente, hace uso de SVM/HOG para distinguir entre autobuses y camiones ya que ambas categorías tienen un tamaño similar. Si nos ceñimos a la clasificación presentada en el capítulo 2, TrafficMonitor se puede clasificar como un sistema híbrido que emplea técnicas basadas tanto en modelos 3D como en características.

TrafficMonitor introduce una nueva técnica para resolver las oclusiones basada en KLT. Se trata de una de los dificultades más importantes cuando estamos intentando resolver el problema de la clasificación en general. Si las oclusiones

no son resueltas de forma correcta y eficiente, sus consecuencias pueden afectar de forma directa a la capacidad del sistema tanto para el seguimiento como a la hora de clasificar los vehículos en distintas categorías.

El sistema se puede dividir en los siguientes grandes bloques:

- Detección de vehículos
- Seguimiento
- Clasificación

La figura 3.1 presenta un diagrama de bloques de todo el sistema. Este tiene como entrada principal un flujo de imágenes que puede venir en directo de una cámara o de un fichero almacenado en disco. El primer subsistema se encarga de procesar este flujo para estimar el fondo, detectar la carretera y detectar los vehículos. Como salida, genera un conjunto de blobs caracterizados por su centro de masas y su cuadro envolvente (*bounding box*). El siguiente subsistema (seguimiento) se encarga de procesarlos y crear vehículos para poder seguirlos entre las distintas imágenes. Como salida genera un conjunto de vehículos cada uno con un identificador único. Finalmente, el subsistema de clasificación se encarga de determinar su categoría. Este proceso dura mientras el sistema sea capaz de seguir al vehículo de forma correcta. Los datos generados por el sistema son almacenados en una base de datos gestionada por el subsistema de estadísticas.

A lo largo de los próximos capítulos, analizaremos en detalle cada uno de los bloques y veremos qué algoritmos han sido empleados para resolver todos y cada uno de los problemas relacionados.

3.2. Escena y calibración semi-automática de la cámara

El escenario típico en una aplicación de monitorización visual de tráfico está formado por tres elementos: La carretera, los vehículos y el sistema de cámara. En la literatura podemos encontrar varios trabajos donde se describe este tipo de escenarios. Por ejemplo [Tomás Rodríguez, Narciso García, 2010] realiza una descripción detallada del escenario y los distintos elementos que lo forman. A diferencia de los escenarios de monitorización de vehículos en entornos interurbanos, el entorno físico de una autopista tiene unas características específicas. El asfalto en general es de color gris, sin texturas excepto las correspondientes

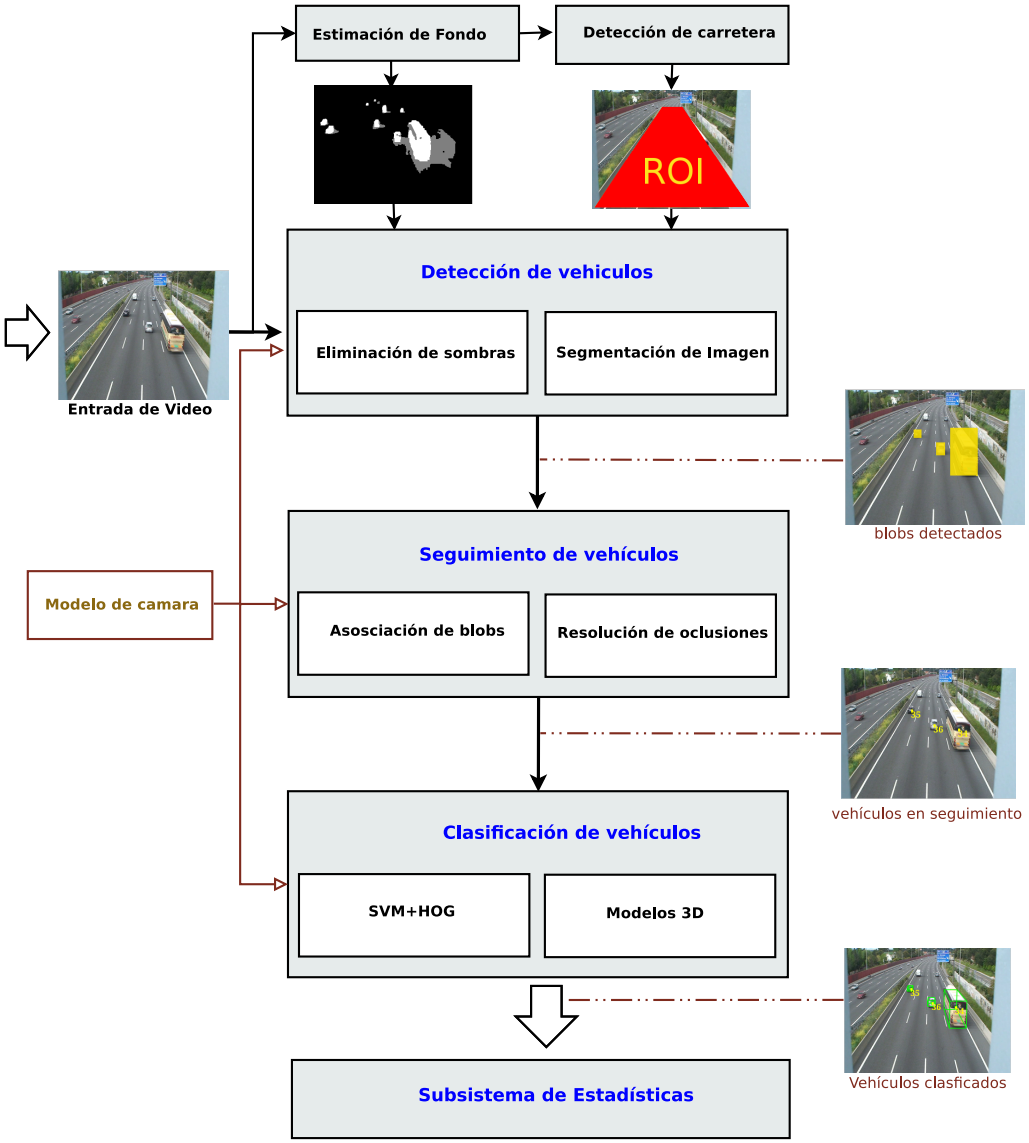


Figura 3.1: Diagrama de bloques del TrafficMonitor

a las marcas divisorias de carriles o símbolos de señalización en raras ocasiones. Normalmente está dividida en dos o más carriles donde sólo se permite circular a un conjunto limitado de vehículos motorizados, ya que en general ni los peatones ni las bicicletas pueden circular en este tipo de autovías. Los vehículos pueden viajar en cualquier carril en el mismo sentido, sin embargo muchas normativas pueden limitar la circulación de vehículos pesados al carril derecho por seguridad. Este hecho incrementa la posibilidad oclusiones en dicho carril y los carriles adyacentes.

El tercer elemento de la escena es la cámara. Aún cuando es un actor pasivo, su configuración y posicionamiento es crucial para obtener buenos resultados. La posición de la cámara tiene que ser la óptima para tener una buena visión de toda la carretera y minimizar el riesgo de oclusiones que es uno de los problemas más

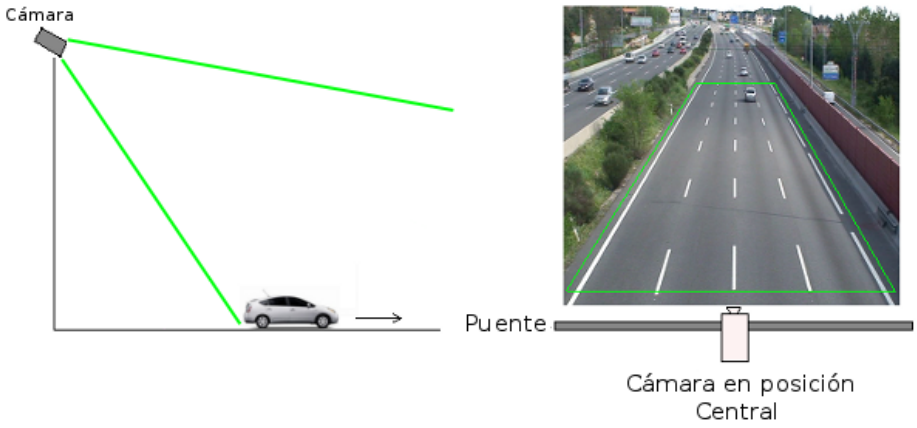


Figura 3.2: Configuración de la cámara

importantes que abordaremos a lo largo de esta tesis. Además de lo anterior, la cámara tiene que abarcar una superficie amplia de los carriles por monitorizar. Esto incrementa la posibilidad de obtener mejores resultados, puesto que podremos observar los vehículos durante más tiempo y resolver oclusiones temporales si las hay. Dicho esto, una buena posición de cámara es aquella que permite monitorizar por lo menos cuatro carriles abarcando una distancia de más de cincuenta metros, distancia suficiente para estimar la velocidad de un vehículo que está viajando a 120km/h (un vehículo viajando a esta velocidad tarda 2 segundos en cubrir una distancia de 66 metros). Cabe destacar que esta configuración es posible, ya que la mayoría de puentes sobre autopistas están elevados por más de siete metros. La figura 3.2 describe un ejemplo de posicionamiento de cámara sobre la autopista.

Para llevar a cabo la clasificación basada en modelos 3D tenemos que ser capaces de recuperar e interpretar la geometría 3D de la escena. Para ello, haremos uso de una cámara calibrada como sensor principal. Asumiendo un modelo *pinhole* (ver apéndice A) de cámara donde los rayos atraviesan la caja por un agujero (foco de la cámara) para impactar en el plano imagen. El comportamiento de las lentes según este modelo es lineal. Hoy en día las cámaras comerciales se aproximan razonablemente a este modelo, dado que no tienen mucha distorsión radial. En este modelo el sistema de referencia de la cámara se sitúa en el centro de proyección, haciendo que el eje Z sea el eje óptico de la cámara de tal forma que el plano imagen se sitúa perpendicular al eje óptico a una distancia igual a la distancia focal de la cámara. La intersección del eje principal con el plano imagen

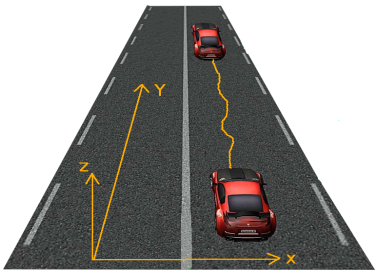


Figura 3.3: Configuración de los ejes XYZ respecto a la carretera

se denomina punto principal.

El principal sensor utilizado en el sistema para monitorizar el tráfico es una simple cámara calibrada. Se trata de un requisito básico para llevar a cabo tareas como la estimación de velocidad o la clasificación de vehículos. Empleando una cámara calibrada somos capaces de determinar qué punto m de la imagen 2D corresponde a un punto M del mundo 3D. La transformación inversa no es posible con una sola cámara. Para ello hacen falta varias cámaras. Recientemente han aparecido varios trabajos y técnicas para resolver el problema de la calibración en el contexto de monitorización de tráfico de forma automática. [Álvarez et al., 2014] presenta un técnica jerárquica que se apoya sobre los distintos elementos estructurales disponibles en la escena para estimar dos puntos de fuga ortogonales (o el punto principal) y de ahí estimar los parámetros de calibración de la cámaras. Los elementos pueden ser desde pasos de peatones, hasta líneas provenientes desde los edificios que hay que en la escena. [Dubska et al., 2014] presenta un sistema para el mismo fin pero utilizando una técnica distinta. En este caso toda la detección se basa en el análisis de varios minutos de vídeo para extraer tres puntos de fuga.

En nuestro caso estamos interesados en proyectar plantillas 3D conocidas de antemano sobre la imagen de la cámara (proceso de clasificación), además, necesitamos saber la distancia física entre dos puntos del plano 2D de la imagen para la estimación de velocidad. Cuanto más precisa es la calibración de nuestra cámara, mejores resultados obtendremos tanto para la estimación de velocidad como para la clasificación en distintas categorías. La figura 3.3 muestra el sistema de coordenadas 3D escogido respecto a la carretera. El plano $Z = 0$ corresponde a la carretera, mientras que los vehículos viajan en la dirección positiva del eje Y.

Dada esta configuración, la proyección de un punto del espacio 3D $P(X, Y, Z)$ a un punto $p(u, v)$ del plano imagen se puede expresar con la ecuación A.8. La

técnica empleada en esta tesis para calibrar la cámara está basada en el método VVH (*Two vanishing points and known camera height*) descrito en [Neeraj K. Kanhere; Stanley T. Birchfield, 2010]. Esta técnica parte de las siguientes suposiciones:

- Altura de cámara conocida
- Un ángulo nulo de roll
- Un valor de *skew* nulo
- Punto principal situado en el centro de la imagen

Estas suposiciones son válidas en general para nuestro escenario dado que podemos configurar la cámara de forma sencilla con un ángulo nulo de roll. La altura se puede obtener de forma exacta durante la instalación. Las últimas dos suposiciones son válidas para la mayoría de las cámaras digitales de hoy en día. En la técnica VVH, dados dos puntos de fuga $P_0(u_0, v_0)$ y $P_1(u_1, v_1)$ podemos calcular el resto de parámetros de calibración de la cámara: la distancia focal f, el *tilt* θ y el *pan* ϕ mediante las siguientes ecuaciones:

$$f = \sqrt{-(v_0^2 + u_0 u_1)} \tag{3.1}$$

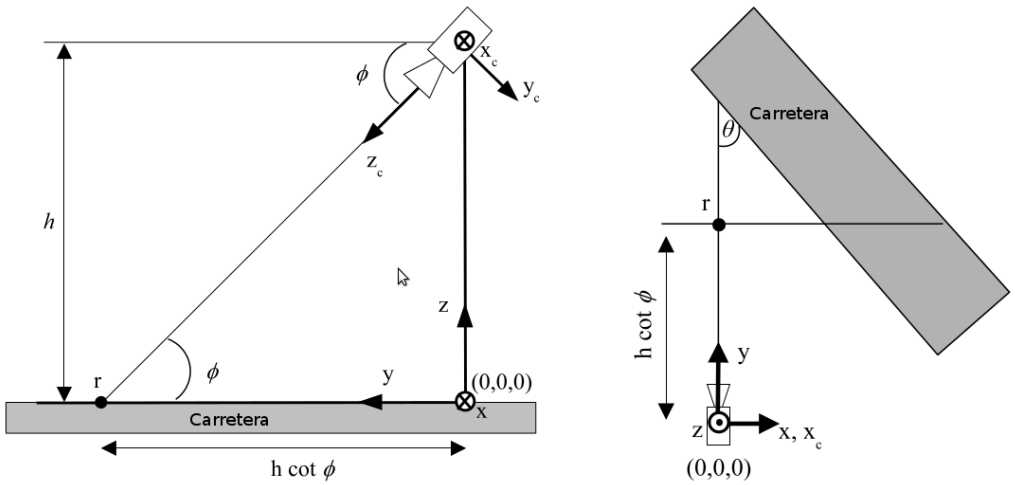
$$\phi = \tan^{-1}(\frac{-v_0}{f}) \tag{3.2}$$

$$\theta = \tan^{-1}(\frac{-u_0 \cos\phi}{f}) \tag{3.3}$$

La figura 3.4 muestra la disposición de la cámara en el mundo real sobre la carretera.

El proceso de la calibración en sí es simple y ha sido diseñado para ser utilizado por operadores sin mucho conocimiento técnico. El punto P_1 se obtiene como la intersección entre las líneas (A, D) y (B, C) en la figura 3.5 mientras que el punto P_2 se obtiene como el resultado de la intersección entre las líneas (A, B) y (C, D). Normalmente tanto P_1 como P_2 caen fuera de la imagen.

El proceso de calibración es sencillo. El usuario puede parar el vídeo en cualquier momento mientras haya un vehículo en la escena. En este momento, tiene que elegir la categoría del vehículo en cuestión entre las categorías soportadas por el sistema: motocicleta, coche, furgoneta, autobús y camión. A continuación, tiene que especificar y ajustar la altura de la cámara de forma aproximada. En



(a) Escenario visto desde la izquierda (b) Vista superior del escenario

Figura 3.4: Disposición de la cámara sobre la carretera

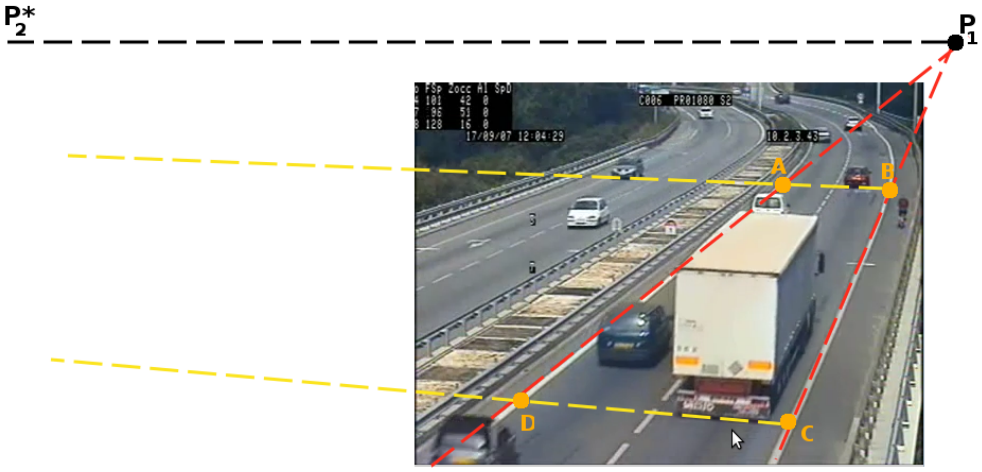


Figura 3.5: Estimación de puntos de fuga: El punto P_1 se obtiene como la intersección entre las líneas (A, D) y (B, C) mientras que el punto P_2 es la intersección entre las líneas (A, B) y (C, D)

Figura 3.6: Calibración semi-automática de la cámara

este momento la herramienta integrada de calibración automática dibuja la proyección de la plantilla 3D correspondiente a la categoría seleccionada sobre el vehículo y ofrece cuatro puntos al usuario para dibujar un rectángulo en el plano de la carretera ($ABCD$ en la figura figura 3.5). La manera más sencilla de lograrlo es hacer coincidir dos líneas con los bordes de la carretera. Durante es proceso el usuario puede seguir ajustando la altura de la cámara hasta que la plantilla 3D encaje lo mejor posible sobre el vehículo seleccionado. Cuando el usuario está satisfecho con los resultados de calibración puede guardar esta configuración de la cámara para futuros usos. En la figura 3.6 podemos ver un ejemplo de la interfaz gráfica de la herramienta de calibración.

En resumen, los parámetros intrínsecos y extrínsecos de la cámara se obtienen mediante la técnica VVH. Para calcular los extrínsecos partimos de una altura conocida de la cámara y calculamos la orientación de la misma en forma de ángulos Euler. Estos se pueden convertir en una matriz de rotación $R = XYZ$ mediante las fórmulas 3.4, donde θ_x, θ_y y θ_z son ángulos de Euler calculados previamente.

$$X = \begin{bmatrix} 1 & 0 & 0 \\ 1 & cos(\theta_x) & -sin(\theta_x) \\ 1 & sin(\theta_x) & cos(\theta_x) \end{bmatrix} \quad Y = \begin{bmatrix} cos(\theta_y) & 0 & sin(\theta_y) \\ 0 & 1 & 0 \\ -sin(\theta_y) & 0 & cos(\theta_y) \end{bmatrix}$$

$$Z = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z) & 0 \\ sin(\theta_z) & cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

Esta herramienta ha sido probada con varios vídeos provenientes de varias fuentes distintas y con parámetros de calibración desconocidos a priori. La figura

Figura 3.7: Resultados de los experimentos utilizando la herramienta semi-automática de calibración sobre varios vídeos con distintos ángulos y altura de la cámara

3.7 muestra varios ejemplos de estos vídeos y los resultados de la calibración semi-automática. Con los resultados obtenidos la herramienta ha demostrado ser una manera sencilla y rápida para la calibración semi-automática de cámaras. Dicho esto, aún cuando la precisión de los resultados obtenidos es suficiente para ser utilizada en los procesos de seguimiento y clasificación, sin embargo no se puede utilizar con el fin de medir la velocidad con alta precisión. Para este tipo de aplicaciones se recomienda utilizar otra herramienta de calibración más exacta.

3.3. Base de datos de vídeos

Para validar el sistema desarrollado como parte de esta tesis y los distintos algoritmos implementados como parte de la misma se han llevado a cabo varios experimentos con la idea de probar el sistema con distintas condiciones de tráfico. Uno de los problemas más grandes que hemos encontrado a lo largo del desarrollo de la tesis es la falta de bases de datos universales para evaluar el desempeño de

los algoritmos implementados. A diferencia de otras áreas donde la comunidad científica ha hecho un gran esfuerzo creando bases de datos de referencia, en el área de visión computacional aplicada al tráfico falta todavía este tipo de recursos. Esto se debe en gran medida a la variedad de escenarios y configuraciones posibles de la escena. Ante este problema se ha creado una base de datos propia compuesta por cien vídeos con varias condiciones de tráfico y configuración de la escena. Los vídeos generados se ponen a disposición de la comunidad como parte de esta tesis con el fin de ayudar a futuros investigadores a validar sus sistemas contra los mismos datos de referencia.

Durante la creación de la base de datos se ha intentado recopilar vídeos con destinas configuraciones teniendo en cuenta parámetros como la densidad de tráfico, el ángulo de visión, la configuración de la carretera (número de carriles, trazado, etc), condiciones meteorológicas, etc. La idea siempre ha sido crear una base de datos amplia que cubra distintos escenarios. De esta manera, podemos validar los algoritmos implementados de forma relativamente sencilla.

Los vídeos recopilados han sido procesados de uno en uno con el fin de caracterizar los distintos parámetros de la escena como la meteorología, sombras, oclusiones, etc y establecer la verdad absoluta para cada uno de ellos. Esto incluye contar el número de coches y clasificarlos en las categorías que tenemos que soportar.

La tabla 3.1 contiene un resumen de la base de datos recopilada.

Además de la base de datos propia se ha utilizado la base de datos *GRAM-RTM* creada por [Guerrero-Gomez-Olmedo, R. and Lopez-Sastre, R. J. and Maldonado-Bascon, S. and Fernandez-Caballero, A., 2013] sobre todo para probar el sistema y evaluar su rendimiento en presencia de niebla y escenarios soleados.

vídeo	Densidad de tráfico	Nº carriles	Meteorología	Ángulo de visión	Sombras	Oclusiones
video-0001	baja	3	Soleado	Vista lateral izq	No	bajo
video-0002	media	4	Nublado	Vista lateral izq	No	Alto
video-0005	media	4	Soleado	Vista Aérea centrada	No	Bajo
video-0006	baja	4	Soleado	Vista Aérea centrada	No	Bajo
video-0008	baja	2	Soleado	Vista Aérea centrada	No	Bajo
video-0009	baja	2	Soleado	Vista Aérea centrada	Sí	Nulo
video-0010	baja	2	Soleado	Vista Aérea centrada	Sí	Bajo
video-0011	baja	2	Soleado	Vista Aérea centrada	No	Bajo
video-0012	baja	2	Nublado	Vista Aérea centrada	No	Nulo
video-0013	baja	2	Nublado	Vista Aérea centrada	No	Nulo
video-0014	baja	2	Soleado	Vista Aérea centrada	No	Nulo
video-0015	baja	2	Soleado	Vista Aérea centrada	No	Nulo
video-0022	media	4	Nublado	Vista Aérea centrada	No	Nulo
video-0023	media	4	Nublado	Vista Aérea centrada	No	Nulo
video-0024	media	4	Nublado	Vista Aérea centrada	No	Nulo
video-0025	media	4	Nublado	Vista Aérea centrada	No	Bajo
video-0026	alta	4	Nublado	Vista Aérea centrada	No	Bajo
video-0027	alta	4	Nublado	Vista Aérea centrada	No	Medio
video-0028	media	4	Nublado	Vista Aérea centrada	No	bajo
video-0029	media	4	Nublado	Vista Aérea centrada	No	bajo
video-0031	media	4	Soleado	Vista Aérea centrada	Sí	bajo
video-0032	alta	4	Soleado	Vista Aérea centrada	Sí	Alto
video-0033	alta	4	Soleado	Vista Aérea centrada	Sí	Alto
video-0034	alta	4	Soleado	Vista Aérea centrada	Sí	Bajo
video-0035	muy alta	4	Soleado	Vista Aérea centrada	Sí	muy alta
video-0039	media	4	Trancisión	Vista Aérea centrada	Sí	media
video-0040	media	4	Nublado	Vista Aérea centrada	Sí	media
video-0041	baja	4	Nublado	Vista Aérea centrada	Sí	baja
video-0042	baja	4	Nublado	Vista Aérea centrada	Sí	baja
video-0043	baja	4	Trancisión	Vista Aérea centrada	Sí	baja
video-0044	baja	4	Soleado	Vista Aérea centrada	Sí	baja
video-0045	media	4	Soleado	Vista Aérea centrada	Sí	baja
video-0046	media	4	Soleado	Vista Aérea centrada	Sí	media
video-0047	media	4	Soleado	Vista Aérea centrada	Sí	media
video-0048	alta	4	Soleado	Vista Aérea centrada	Sí	alta
video-0049	media	4	Soleado	Vista Aérea centrada	Sí	media
video-0050	media	4	Soleado	Vista Aérea centrada	Sí	media
video-0051	alta	4	Soleado	Vista Aérea centrada	Sí	media
video-0052	media	4	Soleado	Vista Aérea centrada	Sí	baja
video-0053	media	4	Soleado	Vista Aérea centrada	Sí	baja
video-0054	alta	4	Soleado	Vista Aérea centrada	Sí	media
video-0055	baja	4	Soleado	Vista Aérea centrada	Sí	baja
video-0056	media	4	Soleado	Vista Aérea centrada	Sí	baja
video-0057	media	4	Soleado	Vista Aérea centrada	Sí	media
video-0058	media	4	Soleado	Vista Aérea centrada	Sí	media
video-0059	media	4	Soleado	Vista Aérea centrada	Sí	media
video-0060	media	4	Soleado	Vista Aérea centrada	Sí	baja
video-0061	media	4	Soleado	Vista Aérea centrada	Sí	baja
video-0062	media	4	Nublado	Vista Aérea centrada	No	media
video-0063	media	4	Nublado	Vista Aérea centrada	No	baja
video-0064	media	4	Nublado	Vista Aérea centrada	No	baja
video-0065	media	4	Nublado	Vista Aérea centrada	No	baja
video-0066	media	4	Nublado	Vista Aérea centrada	No	baja
video-0067	media	4	Nublado	Vista Aérea centrada	No	baja
video-0068	media	4	Nublado	Vista Aérea centrada	No	baja
video-0069	media	4	Nublado	Vista Aérea centrada	No	media
video-0070	media	4	Nublado	Vista Aérea centrada	No	baja
video-0071	media	4	Nublado	Vista Aérea centrada	No	nula
video-0072	media	4	Nublado	Vista Aérea centrada	No	nula
video-0073	baja	4	Nublado	Vista Aérea centrada	No	nula
video-0074	baja	4	Nublado	Vista Aérea centrada	No	baja
video-0075	baja	4	Nublado	Vista Aérea centrada	No	baja
video-0076	baja	4	Nublado	Vista Aérea centrada	No	baja
video-0077	meda	4	Nublado	Vista Aérea centrada	No	baja
video-0078	baja	4	Nublado	Vista Aérea centrada	No	nula
video-0079	media	4	Nublado	Vista Aérea centrada	No	baja
video-0079	baja	4	Nublado	Vista Aérea centrada	Sí	baja
video-0080	baja	4	Nublado	Vista Aérea centrada	Sí	baja
video-0081	baja	4	Nublado	Vista Aérea centrada	Sí	baja
video-0082	baja	4	Nublado	Vista Aérea centrada	Sí	baja
video-0083	baja	4	Nublado	Vista Aérea centrada	Sí	baja
video-0084	alta	4	Nublado	Vista Aérea centrada	Sí	media
video-0085	baja	4	Nublado	Vista Aérea centrada	Sí	nula
video-0085	alta	4	Nublado	Vista Aérea centrada	No	baja
video-0086	alta	4	Nublado	Vista Aérea centrada	No	baja
video-0087	media	4	Nublado	Vista Aérea centrada	No	nula
video-0088	alta	4	Nublado	Vista Aérea centrada	No	baja
video-0089	media	4	Nublado	Vista Aérea centrada	No	baja
video-0090	media	4	Nublado	Vista Aérea centrada	No	baja
video-0091	baja	3	Nublado	Vista Aérea centrada	No	nula
video-0092	media	3	Nublado	Vista Aérea centrada	No	baja
video-0094	muy alta	4	soleado	Vista Aérea centrada	Sí	alta
video-0095	muy alta	4	soleado	Vista Aérea centrada	Sí	alta
video-0096	muy alta	4	soleado	Vista Aérea centrada	Sí	alta
video-0099	media	4	muy soleado	Vista Aérea centrada	Sí	media
video-0100	media	3	muy soleado	Vista Aérea centrada	Sí	baja
video-0101	media	3	Nublado	Vista Aérea centrada	No	baja
video-0104	muy alta	4	Nublado	Vista Aérea centrada	No	muy alta
video-0110	media	4	Lluvia	Vista Aérea centrada	No	baja

Cuadro 3.1: Base de datos de vídeos recopilados

Capítulo 4

Seguimiento de vehículos

La detección y seguimiento de vehículos suponen las primeras fases del sistema tal y como se ha visto en el capítulo 3. Se trata de dos etapas fundamentales de las cuales depende el resto de fases como la clasificación. La primera se encarga de procesar el vídeo de entrada, detectar los vehículos y filtrar el ruido generado como parte de la detección. Como salida genera un conjunto de blobs caracterizados por su centro de masas y cuadro envolvente. La segunda etapa recibe esta información como entrada y después de analizarla y procesarla genera un conjunto de vehículos caracterizados por varios atributos que se utilizan para llevar a cabo la tarea del seguimiento y constituyen la entrada para la fase de clasificación.

4.1. Detección de vehículos

Esta fase tiene como objetivo detectar todos los vehículos que hay en una imagen. Para ello, esta etapa a su vez se basa en varias fases: detección de fondo, detección de carretera, extracción de blobs y eliminación de sombras. La figura 4.1 muestra los distintos pasos que forman el algoritmo de detección de vehículos.

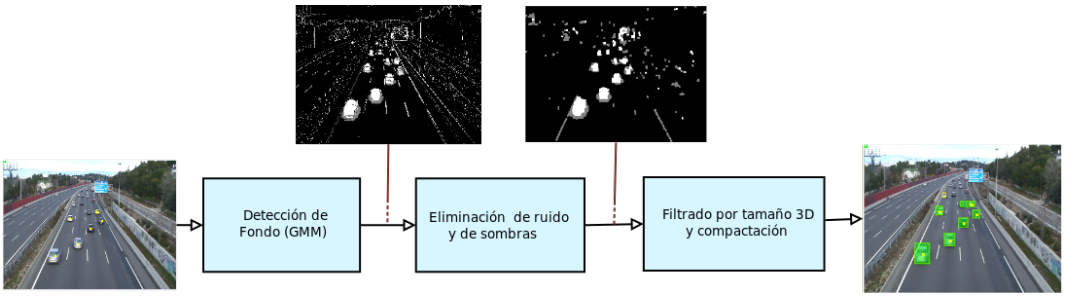


Figura 4.1: Fases del algoritmo de detección de vehículos

4.1.1. Detección de fondo

La detección de fondo es una de las etapas fundamentales en la mayoría de los algoritmos de seguimiento y clasificación basados en visión. En este caso es la primera etapa sobre cuyo resultado se basan el resto de los algoritmos de nuestro sistema. Se trata de una parte importante ya que cualquier error en esta fase podría propagarse hacia el resto de las fases de procesamiento introduciendo errores en el resultado final de la clasificación. De ahí, que se busque minimizar los errores provenientes de esta fase a la hora de clasificar un píxel como fondo u objeto en movimiento. Con el fondo estimado el sistema es capaz de detectar los vehículos calculando la diferencia entre el fondo estimado y la imagen actual. En general, teniendo en cuenta las restricciones del escenario descritas en la sección 3.2, un movimiento en el fondo es a priori considerado un vehículo al menos de que el sistema de eliminación de ruido lo descarta por los criterios que veremos más adelante en este capítulo.

La detección de fondo está basada en *MOG*. La versión básica de este algoritmo funciona bastante bien en general, sin embargo, sufre de dos problemas cuyas consecuencias pueden afectar al procesamiento de nuestros algoritmos. El primero de ellos es la respuesta a cambios bruscos en la iluminación. MOG en su versión básica no maneja bien este tipo de cambios y como consecuencia pueden producir falsos cambios que afectan a todo el fondo. El segundo problema son las sombras. En la descripción detallada que hemos visto en la sección 2.1.2 en ningún momento se han tenido en cuenta. Las sombras representan un problema importante en la visión artificial sobre todo en los campos de seguimiento y clasificación. El problema aparece normalmente en estas etapas, puesto que es complicado diferenciar entre un objeto en movimiento y su sombra. En nuestro caso al basar la clasificación en modelos 3D tenemos que eliminar las sombras antes de la fase de clasificación para evitar confundirlas con el cuerpo del vehículo. Si este procesamiento no se lleva a cabo de forma correcta podemos producir errores de clasificación ya que la huella producida por un coche y su sombra podría ser equivalente al tamaño de una furgoneta sin sombra por poner un ejemplo. El mismo problema se puede dar con otras combinaciones cuyo resultado final será siempre un resultado erróneo de clasificación.

Para solventar este problema hemos utilizado una versión mejorada de MOG descrita en [Zoran Zivkovic, 2004]. La ventaja de esta técnica respecto a la original es el uso de un número de componentes adaptable por cada píxel. Esto hace que el procesamiento en global vaya más rápido ya que el número de gaussianas

a tener en cuenta es optimizado a nivel de píxel. En nuestro trabajo utilizamos la implementación de este algoritmo ofrecida por la librería de OpenCV. Esta versión, además de los cambios descritos en el trabajo original, incluye detección de sombras. En este caso el autor asume que los píxeles que pertenecen a la sombra mantienen el mismo color pero con menor intensidad. En la implementación ofrecida por OpenCV además de los valores típicos 0, 255 para representar fondo y objeto respectivamente, se añade un nuevo valor 127 para representar las sombras.

4.1.2. Detección automática de la carretera

En una escena de tráfico normalmente la cámara capta más de un carril o incluso carriles del sentido contrario del tráfico. Puede haber además zonas u objetos en movimiento que no interesa analizar. Estas pueden ser una fuente indeseada de ruido para el sistema. Para descartar estas zonas se ha diseñado e implementado un sistema de detección automática de la carretera para realizar las tareas de monitorización sólo sobre las zonas más relevantes de la escena. Como consecuencia, se minimizan las fuentes de ruido en el sistema y se acelera el procesamiento al reducir el número de objetos por analizar.

Para evitar que los vehículos en movimiento influyan en la detección de carretera se utiliza la imagen del fondo. El algoritmo de detección asume que la cámara está más o menos centrada sobre la carretera. También suponemos que la carretera está delimitada por trazado o barreras de color distinto al del asfalto (normalmente las autopistas están acotadas por marcas viales blancas). El algoritmo de detección parte desde el centro de la imagen y realiza un crecimiento de regiones recursivo para detectar zonas similares. De esta forma la zona correspondiente a la carretera va creciendo hasta toparse con los bordes. Para evitar posibles problemas con el punto de partida (puede caer sobre un marca vial por ejemplo) el proceso elige un píxel de forma aleatoria sobre una ventana situada en el centro de la imagen. Para determinar si el punto de partida elegido ha sido bueno se evalúa el número de puntos cubiertos de la ventana al final el proceso de crecimiento, si éste supera el 80 % del número total de píxeles de la ventana entonces damos el resultado por bueno. En caso contrario se elige otro píxel de la ventana al azar y se repite el proceso. En la figura 4.2 se puede ver la posición que ocupa la ventana sobre la carretera.

Aplicando el algoritmo de detección descrito en el apartado anterior cabe la posibilidad de que ciertas zonas de la carretera quedan fuera la máscara binaria

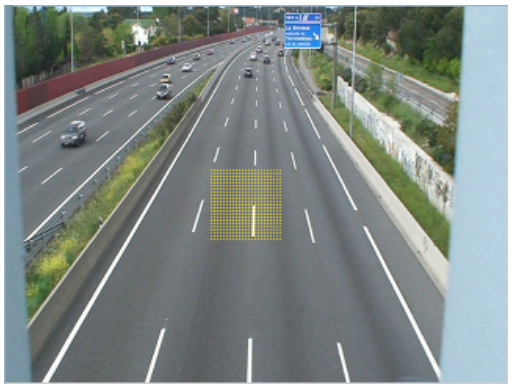


Figura 4.2: Ventana de detección automática de la carretera

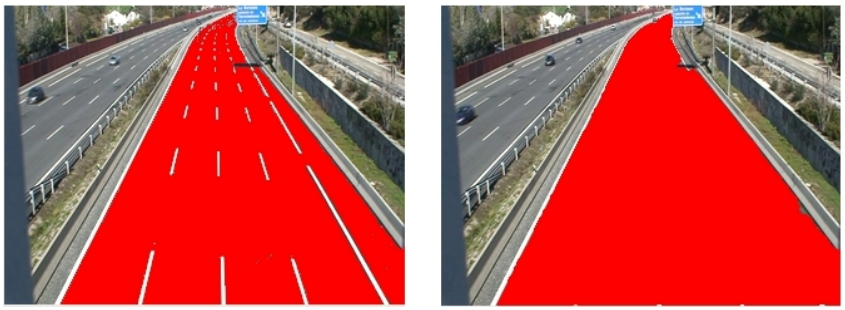


Figura 4.3: Resultado de la dilatación

resultante. Estas suelen ser en general las zonas correspondientes a marcas viales. Para incorporar estas zonas a la carretera aplicamos el operador morfológico de dilatación, de esta manera obtenemos una máscara más compacta que incorpora todas las partes de la carretera. La figura 4.3 muestra un ejemplo del resultado de este proceso.

Una vez construida la máscara correspondiente a la carretera, el siguiente paso es delimitarla para eliminar las zonas más lejanas que el alcance máximo fijado para el seguimiento. La línea de salida de la zona de seguimiento por defecto está fijada a unos sesenta metros desde el origen del sistema de coordenadas descrito en la figura 3.3. Para limitar la carretera se proyecta un punto $S(0, 60, 0)$ sobre la carretera utilizando la información de calibración de la cámara. A continuación, se calcula la intersección de la línea paralela a la imagen que pasa por el punto proyectado (línea azul de la figura 4.4) y la carretera estimada anteriormente. Fruto de esta intersección obtenemos los puntos A y B. Los puntos C y D son hallados con un proceso de búsqueda sencillo en la parte inferior de la carretera. De esta manera quedan determinados los cuatro puntos que delimitan la carretera y que serán utilizados como área de interés por las etapas posteriores. En la figura 4.4 se puede ver el resultado final de este procesamiento. La línea azul

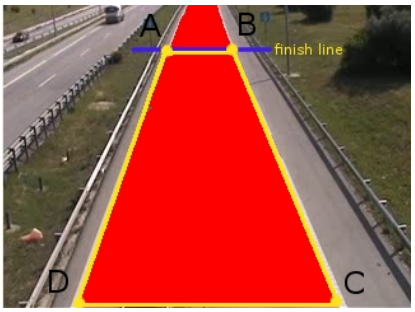


Figura 4.4: Carretera detectada de forma automática

constituye la salida de la zona de seguimiento y es utilizada durante este proceso para determinar si un vehículo está a punto de abandonar esta zona.

4.1.3. Extracción de blobs

El objetivo principal de esta fase es construir y modelar de alguna forma los vehículos en la imagen o en general los *blobs* que hay actualmente en el fotograma. Para llevar a cabo esta tarea utilizamos como entrada la máscara binaria de puntos en movimiento detectada por el algoritmo de MOG después de eliminar las sombras mediante el algoritmo descrito en la sección 4.1.4. A esta imagen primero se le aplica de forma consecutiva un operación de dilatación utilizando un elemento pequeño con el objetivo de llenar los huecos que tienen normalmente los blobs. A continuación se le aplica un operador de erosión con un elemento grande para terminar de descartar todo el ruido de la imagen (puntos, blobs muy pequeños, etc). La máscara binaria resultante es utilizada como entrada a un algoritmo de crecimiento de regiones iterativo, que se encarga de etiquetar los píxeles que pertenecen a cada objeto y compactar los blobs que forman el mismo vehículo. Para ello, la distancia 3D entre cada blob y los demás es analizada. Si el valor está por debajo de un umbral establecido entonces los dos blobs son fusionados en el mismo vehículo (fusionando sus correspondientes cuadros envolventes). La idea detrás de esta operación es que dos vehículos no pueden solaparse físicamente en posiciones muy cercanas.

Para cada blob detectado, el sistema calcula su centro de masas utilizando las ecuaciones 4.1. De esta manera obtenemos el punto más cercano al centro físico real de los vehículos. Durante el procesamiento todos los blobs que caen fuera de la carretera son descartados de forma automática. El resto de los blobs son filtrados por tamaño. TrafficMonitor introduce un algoritmo de descarte basado en el tamaño 3D de los blobs. La idea básica es aprovechar la información de tamaño de la que disponemos para descartar todos los blobs cuyo tamaño está por

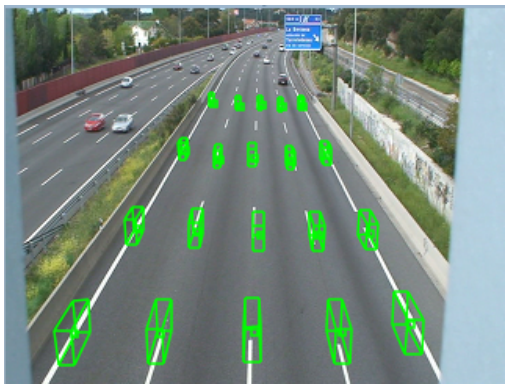


Figura 4.5: Varios ejemplos del área mínima utilizada para filtrar los blobs

debajo del tamaño de una motocicleta, la categoría más pequeña de vehículo que podemos tener. Para ello, utilizando la información de calibración de la cámara proyectamos la plantilla correspondiente a una motocicleta sobre el centro de cada blob y calculamos el área de la proyección (ver figura 4.5). Si el área del blob es menor que el área de la plantilla proyectada entonces el blob en cuestión queda descartado. Este paso adicional termina de eliminar el ruido que hay en la escena de forma adaptativa dejando sólo los blobs que corresponden a vehículos. La ventaja de este esquema que se apoya en la información 3D de la escena es su adaptabilidad dado que el tamaño utilizado para filtrar los blobs depende de la posición del blob sobre la carretera.

$$X_c = \frac{\sum_{i=1}^{N} X_i}{N} \quad Y_c = \frac{\sum_{i=1}^{N} Y_i}{N} \tag{4.1}$$

Para evaluar el rendimiento del algoritmo de segmentación se han llevado a cabo varias pruebas con distintas condiciones de iluminación. En general el algoritmo de *GMM* mejorado en combinación con las etapas de post-procesado que se han añadido genera buenos resultados. La figura 4.6 muestra varios ejemplos del fondo obtenido con distintas condiciones de iluminación. En general se ha podido comprobar que GMM se comporta bastante bien en dichas condiciones.

Sin embargo, el algoritmo tiene problemas en algunos casos. En escenarios con nubes la detección de vehículos funciona bien con la excepción de blobs muy pequeños. En este caso el mecanismo de detección de sombras confunde parte del cuerpo del blob con las sombras. Como consecuencia, el área del blob detectado es menor que su tamaño real. La figura 4.7 muestra un ejemplo de este problema. El algoritmo sufre algunos problemas también en escenarios muy soleados. En este caso ocurre un problema similar al anterior. El exceso de luz en la escena hace que el algoritmo confunda algunas partes del cuerpo del vehículo con sombras.

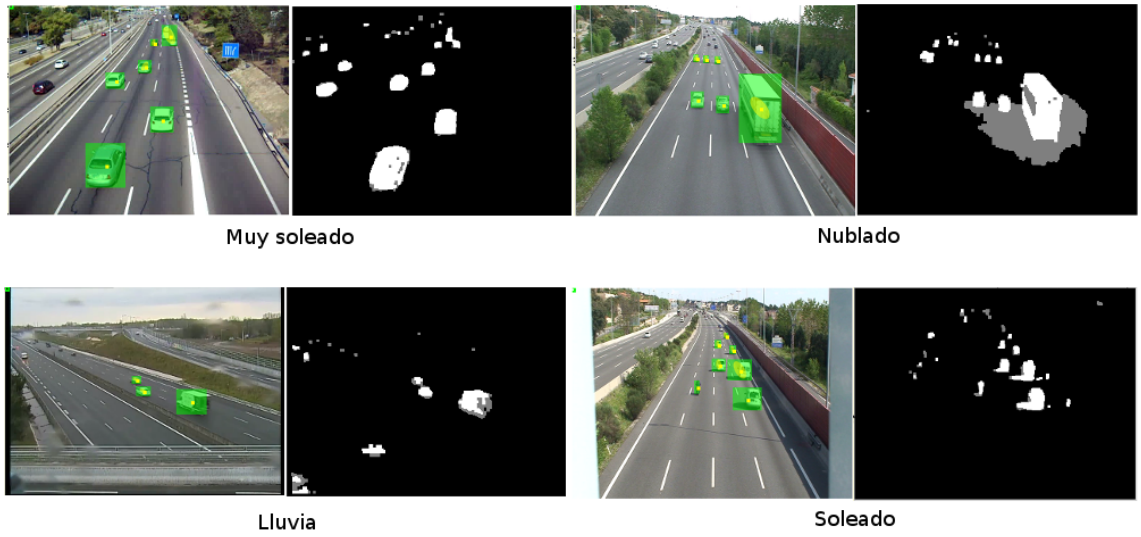


Figura 4.6: Ejemplos de detección de fondos en varias condiciones de iluminación

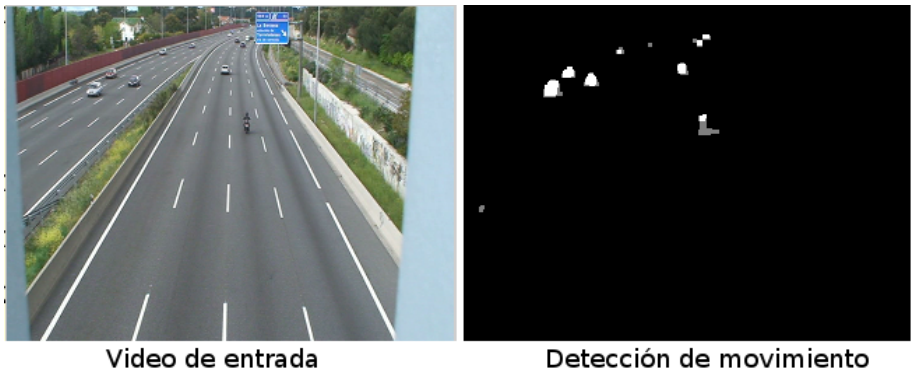


Figura 4.7: Ejemplo de fallo de detección de una moto en un escenario con nubes. La imagen de la derecha corresponde a la salida del módulo de detección de fondo y movimiento. En este caso la detección de sombras ha fallado e incluido gran parte del cuerpo del blob como si fueran sombras.

La figura 4.8 muestra un ejemplo de fallo de detección en este escanrio.

4.1.4. Tratamiento de sombras

La detección y eliminación de sombras es uno de los problemas más importantes dentro del campo de la visión computacional. Sin embargo, este problema cobra especial importancia en el contexto de clasificación ya que cuanto más preciso es nuestro proceso de segmentación, más posibilidades tendremos de extraer las características que nos interesan del objeto y por ende clasificarlo correctamente. En el apartado anterior hemos visto varios ejemplos de algoritmos para la detección y estimación de fondo y hemos explicado las razones por las cuales

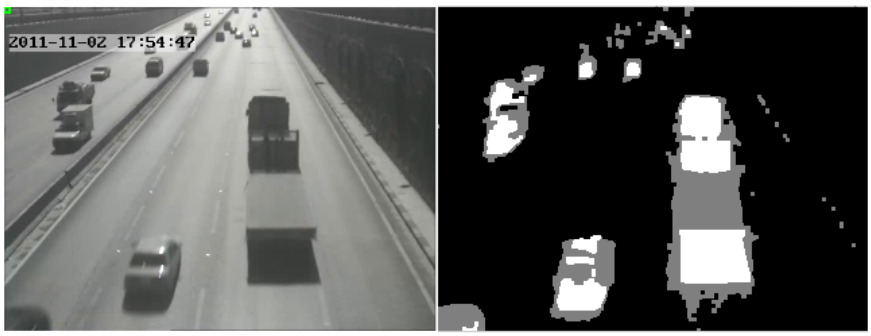


Figura 4.8: Ejemplos fallos de detección de fondo en escenarios muy soleados

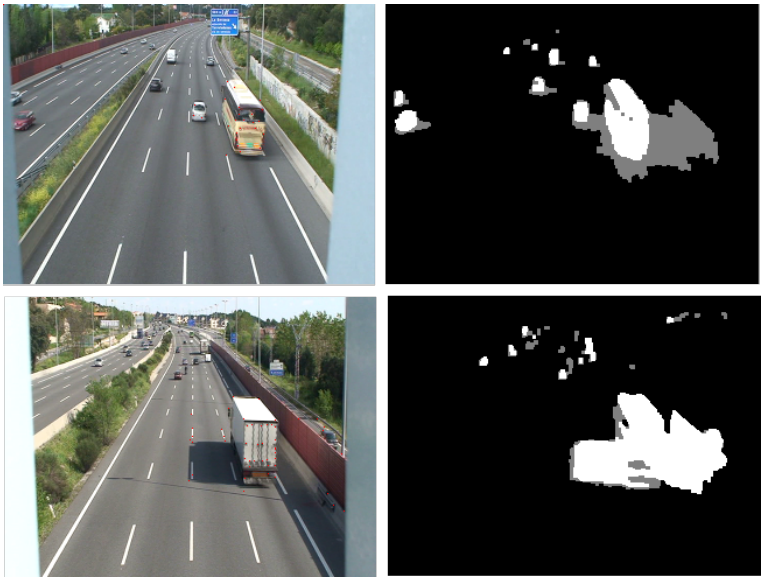


Figura 4.9: Problemas de detección de sombras: En la primera imagen podemos ver que las sombras del autobús son detectadas correctamente (color gris en el fondo), mientras que en el caso del camión tanto las sombras como el reflejo del camión sobre la pared adyacente son clasificados erróneamente como parte del objeto en movimiento

hemos optado al final por la mezcla de gaussianas. Además, el algoritmo seleccionado es una versión mejorada del algoritmo original que incorpora la detección de sombras como novedad. Durante nuestros experimentos hemos observado que esta funcionalidad se comporta bien en general en el sentido de que las sombras son detectas de forma correcta, sin embargo, en escenarios soleados dónde las sombras tienen un color diferente del color del asfalto (las sombras tienen un alto contraste), esta funcionalidad no las detecta de forma correcta. La figura 4.9 muestra dos ejemplos de escenarios donde en el primero las sombras son detectadas correctamente mientras en el segundo podemos ver que las sombras son segmentadas de forma errónea como parte del objeto.

Vista la importancia de este problema y sus consecuencias sobre el resulta-

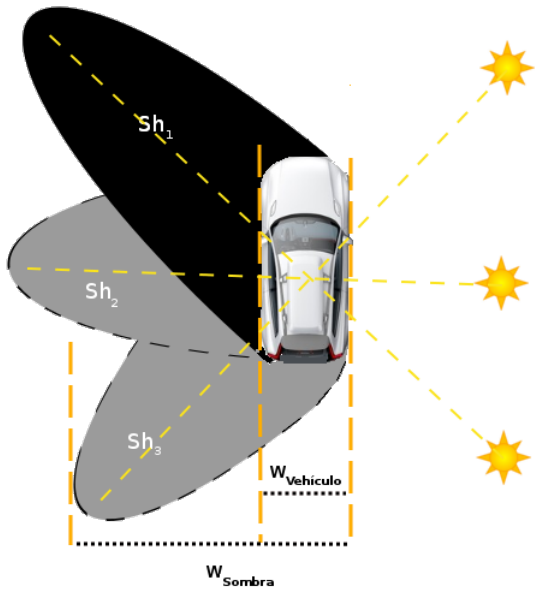


Figura 4.10: Efecto de las sombras sobre el ancho de un vehículo: $W_{vehiculo}$ es el ancho real del vehículo mientras que $W_{sombras}$ es el ancho del vehículo incluyendo su sombra cuando el sol esta en la posición Sh_3. $Sh_{1,2,3}$ son las sombras correspondientes a tres posiciones distintas del sol

do final de la clasificación, se ha añadido una etapa adicional explícita para el tratamiento de sombras. De esta manera, si la parte de detección de sombras incorporada en la detección de fondo falla disponemos de un filtro adicional para detectar estas situaciones e intentar quitar las sombras antes de seguir con el resto de las etapas de procesamiento descritas en el capítulo 5.

El algoritmo empleado en nuestro sistema para la eliminación de sombras está inspirado por el trabajo de [Zhou Zhu and Xiaobo Lu, 2010], sin embargo, en nuestro caso hemos optado por el filtro de bordes *Canny* en vez de *Sobel* para mejorar la detección de bordes. La idea principal detrás del algoritmo se basa en que las sombras suelen ser zonas sin textura (si obviamos las marcas viales) de esta manera, utilizando un detector de bordes lo normal es tener como resultado una zona poblada con arcos y líneas correspondientes al cuerpo del vehículo y otra zona libre de ellos correspondiente a las sombras. Para llegar al resultado final (eliminación por completo de las sombras) además de la detección de bordes hace falta aplicar una serie de operaciones morfológicas cuyo fin es *aumentar* la zona correspondiente al cuerpo del vehículo y quitar las zonas correspondientes a las sombras.

En un primer intento las operaciones morfológicas se han aplicado a todos los vehículos en la escena. El problema en este caso es que las operaciones morfológicas en cascada conllevan un alto coste computacional que impide su funciona-

miento en tiempo real (uno de los requisitos fundamentales de nuestro sistema). Dada esta situación, se ha optado por aplicar el algoritmo sólo sobre los objetos que presentan un tamaño anormal. Después de analizar muchos vídeos con sombras hemos observado que geométricamente la característica a la que más afecta es el ancho de los vehículos, en el sentido de que el ancho de un vehículo con sombras es mucho mayor que el del mismo vehículo sin sombras. En la figura 4.10 se puede ver un ejemplo de cómo afectan las sombras al ancho del vehículo para tres posiciones distintas del sol.

Detectar estas situaciones es posible gracias a la información 3D de la que disponemos de la escena. La idea básica es que los vehículos normalmente pueden tener una longitud variable desde una motocicleta hasta un camión tráiler. Sin embargo, la varianza en el ancho entre los distintos vehículos es pequeña dado que los vehículos tienen que caber en un carril de 3 metros aproximadamente.

En la figura 4.11 podemos ver el efecto que producen las sombras sobre el ancho de los vehículos. En esta figura se ve claramente que las sombras aumentan el ancho de los vehículos de forma considerable.

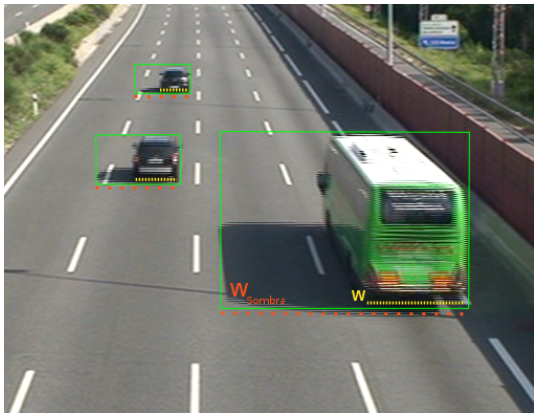


Figura 4.11: Efecto de las sombras sobre el ancho de los vehículos

La idea es aprovechar esta información para limitar la aplicación del algoritmo de eliminación de sombras a los *blobs* cuyo tamaño es anormal reduciendo la carga computacional de todo el proceso. En general se puede distinguir entre varios escenarios que dan lugar a blobs cuyo tamaño es anormal:

- Sombras sin oclusiones: Blob formado por un vehículo y su sombra sin estar involucrado en una oclusión.
- Oclusiones causadas por sombras: Blobs formados por dos o más vehículos dónde la única intersección es la sombra.

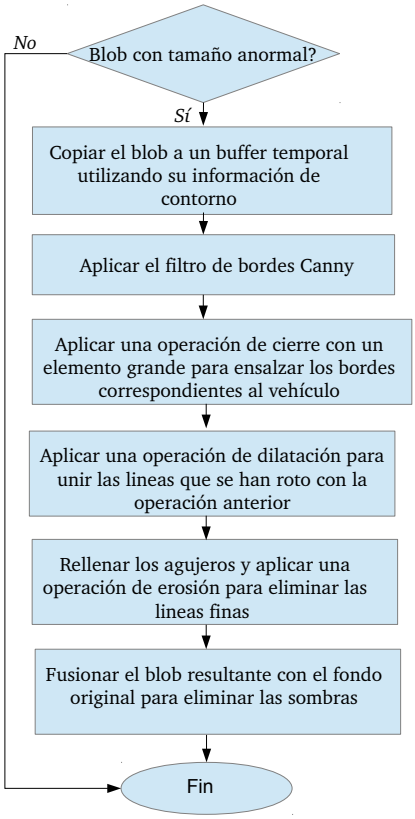


Figura 4.12: Algoritmo para eliminación de sombras

- Oclusiones parciales con sombras: Escena con sombras. Blobs formados por dos o más vehículos dónde la intersección es parte de alguno de los vehículos y/o su sombra.

- Oclusiones parciales sin sombras: Escena sin sombras. Blobs formados por dos o más vehículos dónde la intersección es parte de alguno de los vehículos.

Entre nuestros objetivos se encuentra la detección y resolución de los casos mencionados arriba. El algoritmo que vamos a describir a continuación tiene como objetivo resolver los tres primeros casos. El diagrama de bloques 4.12 describe los distintos pasos del algoritmo de eliminación de sombras. El último escenario corresponde a un problema de oclusión. para este caso particular se ha desarrollado un algoritmo propio que será presentado en la sección 4.2.4.

El algoritmo de eliminación de sombras se puede resumir por la ecuación 4.2

$$Bg' = Bg \cap \nabla((Canny(I) \bullet B) \ominus C) \tag{4.2}$$

Donde:

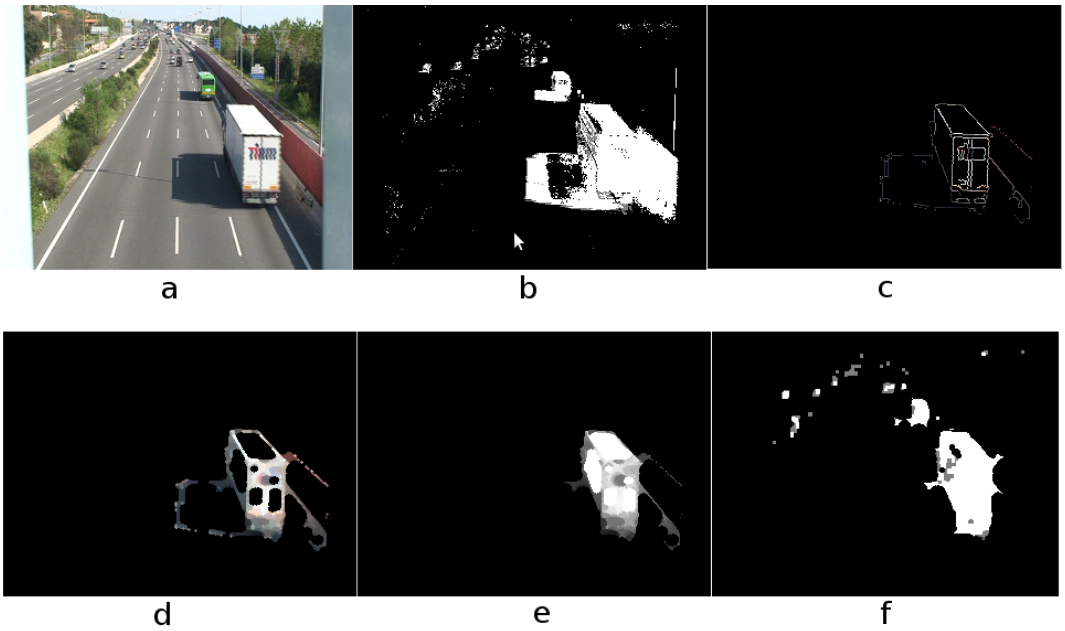


Figura 4.13: Resultados intermedios del algoritmo de eliminación de sombras

[a] Imagen original
[b] Fondo detectado por MOG
[c] Bordes detectados por Canny
[d] Resultado después de la operación de cierre
[e] Resultado después de llenar los agujeros resultantes
[f] Nuevo fondo después de eliminar las sombras

- ∇: representa el operador *fill-holes*
- B: Es el elemento utilizado para la operación de cierre
- C: Es el elemento utilizado para la operación de erosión

Cabe señalar que las operaciones morfológicas son aplicadas solamente a los blobs con un tamaño anormal. Estos son copiados a otra imagen temporal (utilizando su contorno) donde se les aplican todas las operaciones correspondientes al algoritmo de eliminación de sombras. De esta manera, evitamos afectar al resto de los blobs presentes en la escena ya que las operaciones morfológicas podrían afectar el tamaño de éstos y por ende su resultado final de clasificación. En la figura 4.13 podemos ver un ejemplo de las distintas fases del algoritmo. En la misma figura podemos ver el resultado de la eliminación de sombras sobre varios escenarios soleados. Finalmente la figura 4.14 muestra varios ejemplos de experimentos de detección y eliminación de sombras sobre tráfico real.

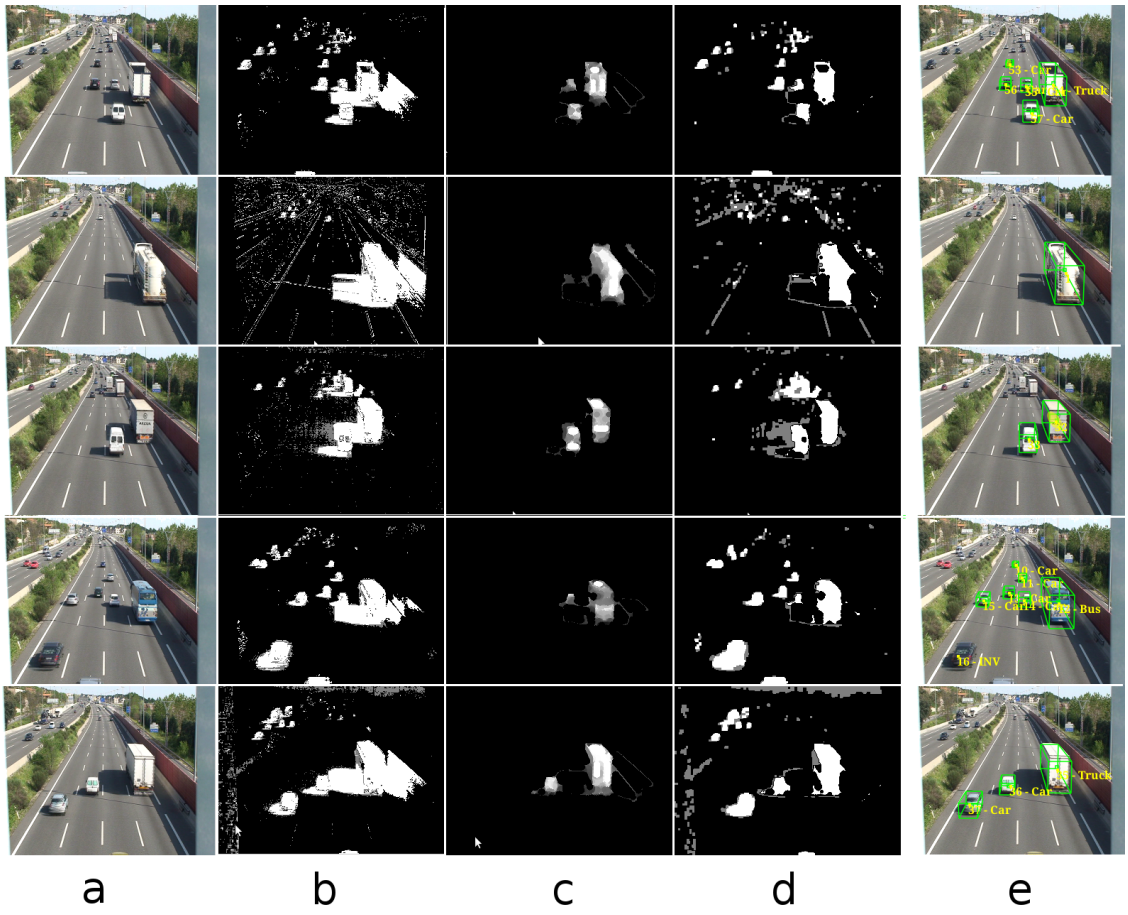


Figura 4.14: Experimentos de detección y eliminación de sombras

[a] Imagen original
[b] Fondo detectado por MOG
[c] Resultado después de llenar los agujeros resultantes
[d] Nuevo fondo después de eliminar las sombras
[e] Resultado de clasificación

4.2. Seguimiento de vehículos

Como se ha visto hasta ahora, los vehículos son detectados en toda la *ROI*. La salida de la fase de detección es un conjunto de blobs representados por su centro de masas y su cuadro envolvente. Todavía no sabemos si cada uno representa uno o varios vehículos. Las tareas principales de la fase de seguimiento son:

- Detectar nuevos vehículos
- Asociar vehículos que ya están en seguimiento con los blobs detectados en el fotograma actual
- Resolver problemas de oclusión

Figura 4.15: Seguimiento de vehículos: cada vehículo tiene un identificador único que le acompaña durante toda la zona de seguimiento.

- Detectar los vehículos que han abandonado la zona de seguimiento para generar sus estadísticas

Durante el proceso de seguimiento el sistema asigna a cada vehículo un identificador único que le acompañará durante toda esta fase. La figura 4.15 muestra un ejemplo de varios vehículos en la zona de seguimiento. A diferencia de otros problemas de detección y seguimiento donde los objetos a seguir pueden entrar y abandonar la escena desde cualquier punto, en nuestro escenario los vehículos tienen un movimiento determinado. Normalmente entran y abandonan la escena desde zonas específicas. Su movimiento es lineal (entre dos fotogramas consecutivos) y todos viajan en la misma dirección. Sin pérdida de generalidad, podemos considerar que durante la zona de seguimiento (que abarca unos sesenta metros aproximadamente) la velocidad de los coches es constante. Conceptualmente los vehículos no pueden ocupar el mismo espacio, así que cualquier oclusión entre ellos en principio se debe al ángulo de la cámara o a las sombras. El sistema trata de aprovechar estas características para mejorar la asociación y finalmente su capacidad de seguimiento de vehículos.

4.2.1. Zonas de entrada y seguimiento de vehículos

En la práctica, dada la posición de la cámara sobre la autopista descrita en la sección 3.2, los vehículos sólo pueden entrar en la escena por la parte inferior de la imagen. Dicho esto, aún cuando estamos detectando vehículos sobre toda la imagen, sólo se pueden crear nuevos vehículos sobre esta parte de la misma. La carretera se divide en dos zonas: zona de entrada y zona de seguimiento. La figura

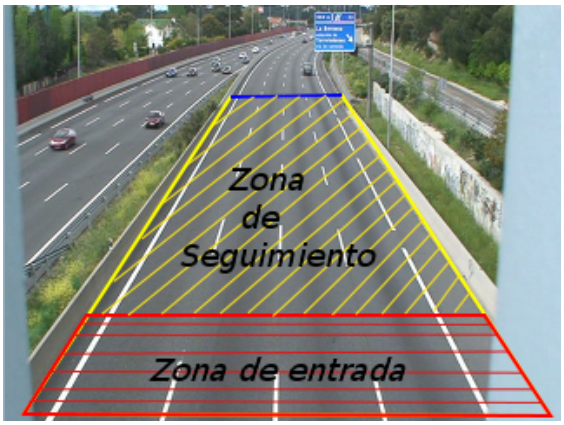


Figura 4.16: Zonas de entrada y seguimiento de vehículos

4.16 muestra la disposición de cada una de ellas. La primera zona es utilizada para crear y confirmar nuevos vehículos, mientras que la segunda es utilizada para seguir el vehículo y acumular sus datos como la trayectoria, categoría, velocidad, etc. Las razones tras esta separación en dos zonas son las siguientes:

1. Sólo los blobs cuyo centro de masas está situado en la zona de entrada pueden crear nuevos vehículos

2. El criterio de asociación entre vehículos y blobs es distinto en la zona de entrada

3. Reducir las fuentes de ruido en el sistema: vehículos que nunca abandonan la zona de entrada son eliminados de forma automática

4. Vehículos que no han sido detectados en la zona de entrada (por estar ocultados por ejemplo) no alteran el seguimiento de los demás vehículos

5. La clasificación de vehículos empieza cuando el vehículo entra en la zona de seguimiento

En resumen, la idea detrás de la división en dos zonas es minimizar las fuentes de ruido que pueden impactar nuestro sistema confirmando primero la detección del vehículo antes de empezar la clasificación o la recogida de cualquier tipo de estadísticas relacionadas con el mismo. Estos son acumulados a lo largo de la zona de seguimiento una vez el vehículo ha sido confirmado.

El algoritmo de seguimiento se divide en dos fases: *emparejamiento entre blobs* y *seguimiento de vehículos*. El objetivo de la primera fase está limitado a encontrar y registrar las asociaciones entre los vehículos que ya están en seguimiento

y los nuevos blobs que acabamos de detectar en el fotograma actual. Otras acciones como actualizar la posición actual del vehículo, su cuadro envolvente o cualquier dato relacionado son responsabilidad de la segunda fase del algoritmo. La separación del algoritmo en dos fases es necesaria para nuestro mecanismo de detección de oclusiones. Básicamente, cuando el mismo blob es asociado con más de un vehículo entonces una oclusión está teniendo lugar entre varios vehículos y hay que tomar ciertas acciones que veremos más adelante para resolver esta situación.

4.2.2. Seguimiento por características visuales

El seguimiento de vehículos se basa principalmente en dos algoritmos: *KLT* (por sus autores Kanade, Lukas y Tomasi, el apéndice B contiene una descripción detallada del algoritmo) y un algoritmo de seguimiento basado en proximidad espacial. KLT es un algoritmo de seguimiento de características en imágenes. Es el resultado de la combinación del trabajo de [Lucas and Kanade, 1981] que desarrolló la idea de utilizar gradientes ponderados como una aproximación de la segunda derivada para emparejar dos imágenes y el trabajo de [Tomasi and Kanade, 1991] que mejoró el proceso de seguimiento mediante la selección de características apropiadas para esta tarea. Las razones que motivaron la selección de este algoritmo son las siguientes:

- KLT ofrece una buena separación entre las características seleccionadas y el proceso de seguimiento
- La implementación piramidal de este algoritmo funciona en tiempo real
- La implementación piramidal de este algoritmo es capaz de manejar grandes desplazamientos entre dos cuadros

La implementación utilizada en esta tesis es la descrita en el trabajo de [yves Bouguet, 2000a]. La novedad de esta implementación radica en la aplicación de KLT de forma recursiva sobre una pirámide de imágenes. Dadas dos imágenes I y J en escala de grises, definimos $I(p) = I(x, y)$ y $J(p) = J(x, y)$ como los valores en escala de grises de la posición correspondiente al punto $p = [x, y]^T$. Dado un punto $u = [u_x, u_y]^T$ en la imagen I el objetivo del algoritmo es encontrar el punto $v = u + d = [u_x + d_x, v_y + d_y]^T$ en la segunda imagen J de tal forma que $I(u)$ y $J(v)$ sean similares. Dicho en otras palabras, estamos buscando el desplazamiento $d = [d_x, d_y]^T$ que ha sufrido el punto u entre las imágenes I y J. Queda por definir

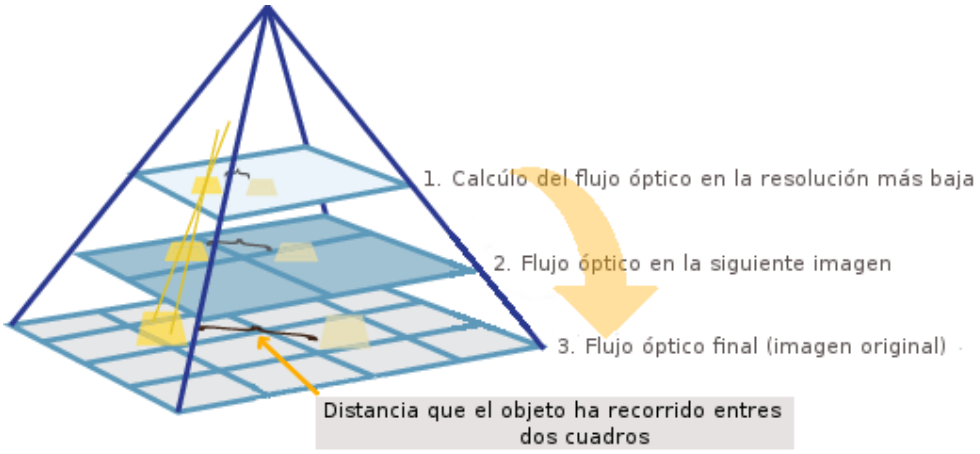


Figura 4.17: Procedimiento recursivo para el cálculo del desplazamiento en la implementación piramidal de KLT

el concepto de *similitud* en un área 2D. Dados dos valores positivos w_x y w_y, la velocidad d de la imagen se define como el vector que minimiza la función ϵ:

$$\epsilon(d) = \epsilon(d_x, d_y) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (I(x,y) - J(x+d_x, y+d_y))^2 \tag{4.3}$$

La función de similitud se mide alrededor del punto en una ventana $(2w_x + 1) \times (2w_y + 1)$. Esta ventana también se conoce como la ventana de integración. La elección del tamaño idóneo de la ventana es un compromiso entre precisión y robustez. Un valor pequeño de ventana permite encontrar el valor idóneo alrededor del punto de partida suponiendo que el punto no ha efectuado un gran desplazamiento. Un valor grande de ventana permitirá cubrir estos casos pero al mismo tiempo aumenta el riesgo de calcular de forma incorrecta el desplazamiento, ya que naturalmente pasamos a tener más candidatos. La implementación piramidal trata de resolver este problema.

La figura 4.17 muestra la propagación recursiva del cálculo del desplazamiento entre los distintos niveles de la pirámide. Es importante recordar que el valor inicial del desplazamiento (caso base de la recesión) es cero. Esto tiene sentido dado que inicialmente el valor del desplazamiento es desconocido.

La ventaja de la implementación recursiva piramidal del algoritmo es que mientras calculamos el valor final del vector desplazamiento d, el valor del flujo óptico d^L de cada nivel intermedio es muy pequeño. Asumiendo que cada nivel de la pirámide puede manejar un desplazamiento máximo de d_{max}, el valor final del desplazamiento que puede manejar una pirámide de m niveles es $d = (2^{L+1} - 1)d_{max}$. Esto nos permite manejar largos desplazamientos con un tamaño de ventana relativamente pequeño.

4.2.3. Emparejamiento entre blobs y vehículos

El objetivo de esta fase es asociar los vehículos en seguimiento con los blobs detectados en el cuadro actual. El criterio utilizado para el emparejamiento depende de la posición actual del vehículo. En la zona de entrada utilizamos un criterio simple basado en la proximidad: cada vehículo es asociado con al blob cuyo centro 2D es el más cercano. La simplicidad del criterio elegido en esta fase del algoritmo se debe a que el vehículo esta en una fase temprana del seguimiento. De hecho, es muy probable que el vehículo esta entrando en la escena y parte del mismo todavía está oculta (es el caso típico para vehículos largos como camiones o autobuses). La información disponible sobre su dirección es escasa y poco fiable. Una vez superada la zona de entrada, el vehículo es confirmado y el seguimiento sigue su curso esta vez combinando dos algoritmos más avanzados el primero basado en la proximidad espacial y el segundo basado en seguimiento de características mediante *KLT*. La idea detrás de esta combinación es utilizar el algoritmo de proximidad para casos triviales de asociación y conmutar a *KLT* cuando este criterio no es suficiente para resolver correctamente el emparejamiento.

El algoritmo de emparejamiento basado en la proximidad trata de aprovechar las restricciones físicas del escenario y la alta frecuencia de la cámara. La idea básica es respetar la coherencia temporal y espacial a la hora de establecer la asociación entre los vehículos y los nuevos blobs. En un mundo ideal la diferencia entre los instantes $t-1$ y t cuando estamos procesando una entrada de vídeo a 30fps es de 33 milisegundos aproximadamente. En este tiempo tan pequeño un vehículo que viaja a $120km/h$ avanzaría una distancia 1,1 metros aproximadamente en el mundo 3D. Dicho esto, y dadas las restricciones físicas su centro de masas en el plano 2D sufrirá un pequeño desplazamiento así que cuando buscamos el nuevo centro del vehículo en el tiempo t deberíamos ser capaces de encontrarlo en un radio circular pequeño alrededor de su posición inicial en $t-1$. Teniendo estos factores en cuenta la distancia recorrida por un vehículo entre dos cuadros consecutivos es pequeña. De ahí, en vez de predecir la localización exacta de la próxima localización del vehículo, intentamos predecir el área donde ésta podría encontrarse. Al principio, esta área es representada en forma de un círculo alrededor del centro 2D del vehículo dado que la orientación del mismo es desconocida. A medida que el vehículo avanza por la zona de seguimiento el historial de su localización es actualizado. Cuando el número de localizaciones acumuladas es mayor que el mínimo requerido (seis posiciones por defecto) entonces la orientación del vehículo es estimada mediante una regresión lineal que

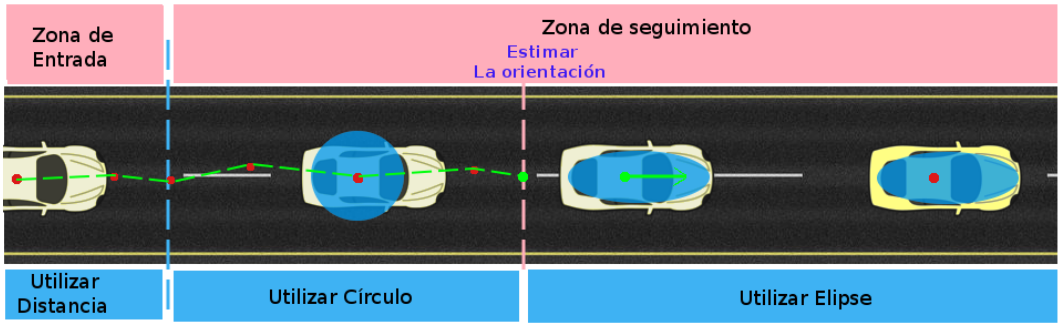


Figura 4.18: Fases del algoritmo de seguimiento espacial

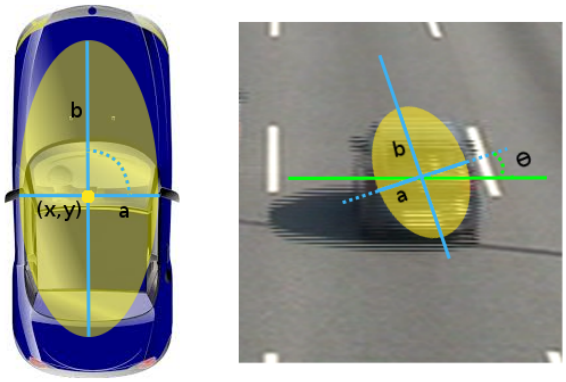


Figura 4.19: Elipse 2D asociada al vehículo

consiste en minimizar $\sum_i e(r_i)$ Donde ri es la distancia con el iesimo punto y $e(ri)$ es una función de la distancia:

$$e(r_i) = 2(\sqrt{1 + \frac{r^2}{2}} - 1) \tag{4.4}$$

A partir de este instante, el circulo se convierte en una elipse que tiene el mismo centro y dirección que la orientación del vehículo. De aquí en adelante, los emparejamientos para este vehículo se limitan a aquellos blobs cuyo centro 2D cae dentro de la elipse asociada con el vehículo. La figura 4.18 muestra la evolución del algoritmo de emparejamiento a lo largo de las zonas de entrada y seguimiento.

En términos matemáticos, cada vehículo tiene asociado una elipse $C_{x_c,y_c,\theta}$ donde (x_c, y_c) es el centro del cuadro del vehículo, θ es su orientación y a, b son el eje menor y mayor de la elipse respectivamente. Los valores de a y b son derivados de su cuadro envolvente. La figura 4.19 muestra un vehículo y los parámetros de su elipse. Dado un blob con un centro 2D $B(x, y)$, éste cae dentro de la elipse

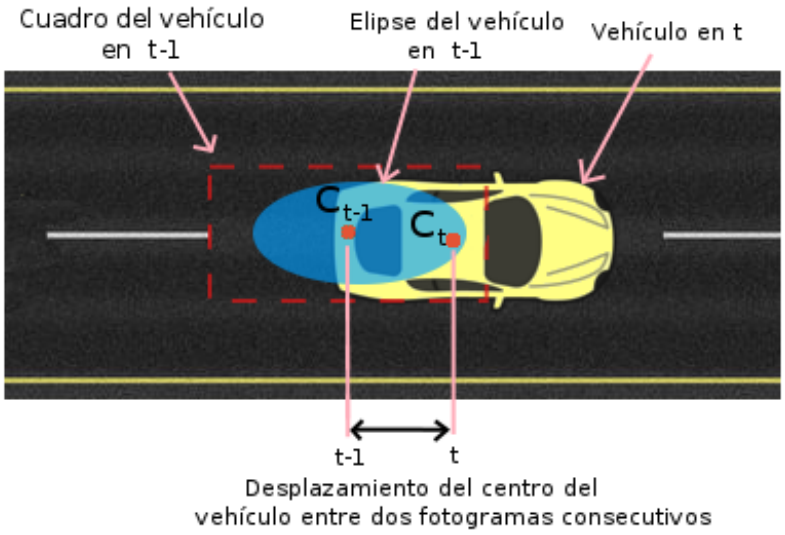


Figura 4.20: Seguimiento mediante proximidad espacial entre dos instantes consecutivos

$C_{x_c,y_c,\theta}$ si y sólo si la desigualdad 4.6 se cumple.

$$C_\theta = \arctan\left(\frac{a_x}{a_y}\right) \tag{4.5}$$

$$\left(\frac{cos(C_\theta)(B_x - C_{x_c}) + sin(C_\theta)(B_y - C_{y_c})}{a}\right)^2 + \left(\frac{cos(C_\theta)(B_y - C_{y_c}) - sin(C_\theta)(B_x - C_{x_c})}{b}\right)^2 \leq 1 \tag{4.6}$$

Donde a_x, a_y son los componentes del vector de orientación del vehículo. La figura 4.20 muestra el emparejamiento entre dos cuadros consecutivos.

Cuando el algoritmo de seguimiento basado en proximidad falla en resolver el emparejamiento para un vehículo sea porque está avanzando a una velocidad alta (su nuevo centro cae fuera de la elipse predicha en $t-1$) o porque se encuentra involucrado en una oclusión parcial, entonces el segundo algoritmo basado en *KLT* es empleado para determinar el emparejamiento. El diagrama de la figura 4.21 muestra los pasos del algoritmo de asociación entre vehículos y blobs. Para llevar esto a cabo, además del centro de masas y del cuadro envolvente el sistema mantiene registro de un número determinado de características visuales por cada vehículo. Estas básicamente son píxeles característicos escogidos de la superficie del vehículo para que se puedan seguir entre dos fotogramas consecutivos. Si repasamos las fórmulas de *KLT* descritas en el apéndice B, la ecuación B.9 es válida si y sólo si la matriz G es invertible. Para ello, el valor mínimo de sus autovalores tiene que ser lo suficientemente grande. Esto significa que la imagen tiene información de gradiente tanto en la dirección vertical y como en la horizontal x. De hecho, esta propiedad tiene una implicación directa a la hora de determinar qué puntos son apropiados para el seguimiento. En la práctica estos

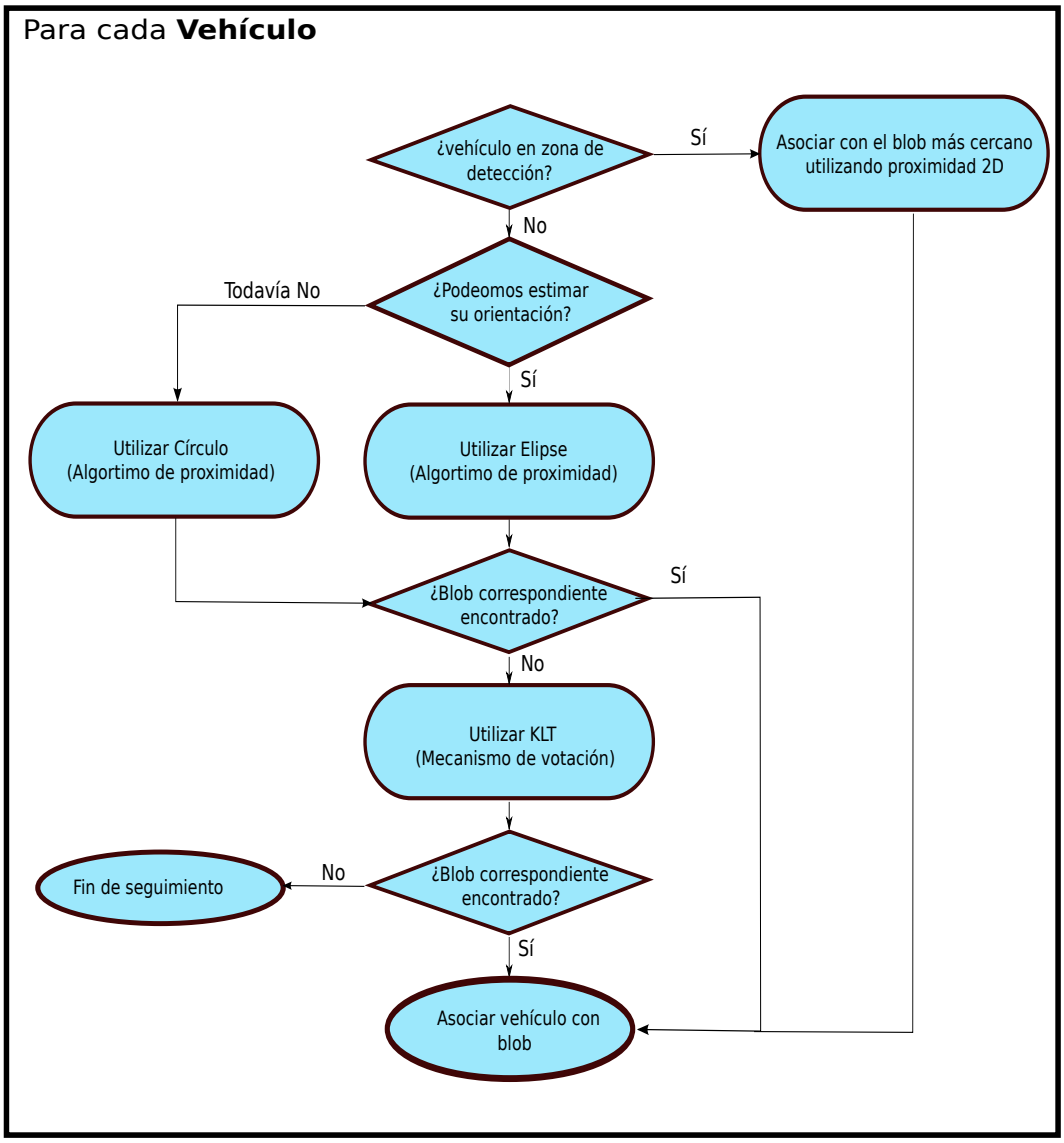


Figura 4.21: Algoritmo de asociación entre blobs y vehículos

suelen corresponder con esquinas sobre la textura del objeto. El algoritmo para escoger los puntos característicos es el siguiente:

1. Calcular la matriz G y su mínimo autovalor λ_m en cada píxel de la imagen de entrada I
2. Calcular λ_{max} como el máximo valor de λ_m sobre toda la imagen
3. Mantener los puntos de la imagen cuya λ_m es mayor con un umbral configurable λ_{max}
4. Mantener los valores representativos en bloques de 3×3

Normalmente el resultado de estas etapas es filtrado para eliminar los puntos

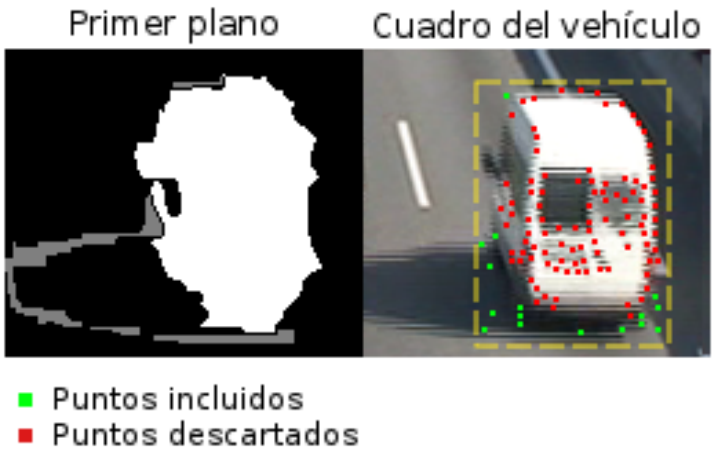


Figura 4.22: Búsqueda de puntos característicos KLT

cuya distancia está debajo de un umbral establecido. Teniendo en cuenta la disposición de la cámara en nuestro escenario los puntos sufren un efecto de escalado a medida que el vehículo se aleja de la cámara. Sin embargo, este problema no afecta al seguimiento de características en nuestro caso dado que estamos procesando la imagen a una alta frecuencia ($30 fps$). Así que los puntos sólo sufren pequeños desplazamientos entre dos fotogramas consecutivos. Los puntos se detectan sobre el cuadro envolvente del vehículo. Sin embargo, no todos los puntos obtenidos sobre esta área son válidos dado que ésta además del cuerpo del vehículo puede contener una pequeña parte del fondo. Para detectar y eliminar estos puntos se hace uso de la información obtenida de la etapa de detección de fondo (figura 4.22). De esta forma el resultado final sólo contiene los puntos que pertenecen al cuerpo del vehículo (donde hubo movimiento). Finalmente, además del cuadro envolvente del vehículo (VBB por las siglas en inglés: *Vehicle bounding box*) el sistema guarda información sobre el mínimo cuadro que contiene todos los puntos característicos del vehículo (FPBB *feature points based bounding box*). Más adelante veremos en qué caso se emplea cada cuadro. La figura 4.23 muestra la diferencia entre los dos cuadros.

Dados dos fotogramas consecutivos F_{t-1} y F_t producidos en el instante $t-1$ y t respectivamente y un vehículo V con un conjunto de puntos característicos P_i donde $i \in 1..N$ calculados en el instante $t-1$ sobre el cuadro envolvente del vehículo. F_t es procesado en el instante t para detectar los blobs correspondientes al movimiento detectado en el fotograma actual. *KLT* calcula el emparejamiento para cada uno de los puntos P_i, como resultado, esto genera un nuevo conjunto de puntos Q_j donde $j \in 1..M$ (M no tiene porque ser igual a N). En la asociación de puntos característicos cada emparejamiento $P_i \rightarrow Q_j$ puede dar lugar a una

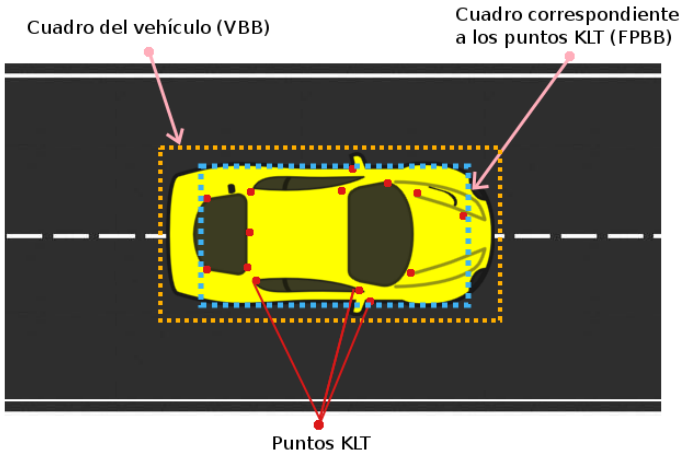


Figura 4.23: Representación gráfica del cuadro envolvente del vehículo (VBB) y el cuadro mínimo que contiene los puntos KLT (FPBB)

de las siguientes situaciones:

- Punto mapeado al blob correspondiente al vehículo.
- Punto mapeado a otro blob que no corresponde al vehículo.
- Punto perdido: *KLT* no es capaz de mapearlo en el fotograma actual.

Para identificar el emparejamiento correcto el sistema hace uso de un mecanismo basado en votos: cada asociación entre dos puntos $P_i \rightarrow Q_j$ es un voto al blob que contiene el punto Q_j. Los puntos perdidos no producen votos. La figura 4.24 muestra un ejemplo sencillo del mecanismo de votación. Al final de este proceso el vehículo es asociado con el blob más votado. Normalmente los puntos mapeados incorrectamente suelen ser la excepción y su número es mucho menor en comparación con los puntos correctamente mapeados. Al final, el vehículo es emparejado con el blob que le corresponde o con ninguno en caso de que el vehículo está totalmente ocultado en el fotograma actual. El seguimiento de los vehículos que no han sido asociados es terminado y los parámetros asociados al vehículo (velocidad, categoría, etc) son estimados utilizando la información acumulada hasta ese momento. En este caso además se introducen una serie de comprobaciones para asegurar que el vehículo haya permanecido un número mínimo de iteraciones en la escena. De esta forma evitamos la generación de parámetros inválidos en caso de ruido o de vehículos que han estado muy poco tiempo en la escena.

En resumen, nuestro algoritmo siempre prepara los puntos característicos de KLT para todos los vehículos en la iteración $t-1$. Si en la iteración t el vehículo es asociado de forma trivial (por proximidad) y no ha entrado en oclusión entonces

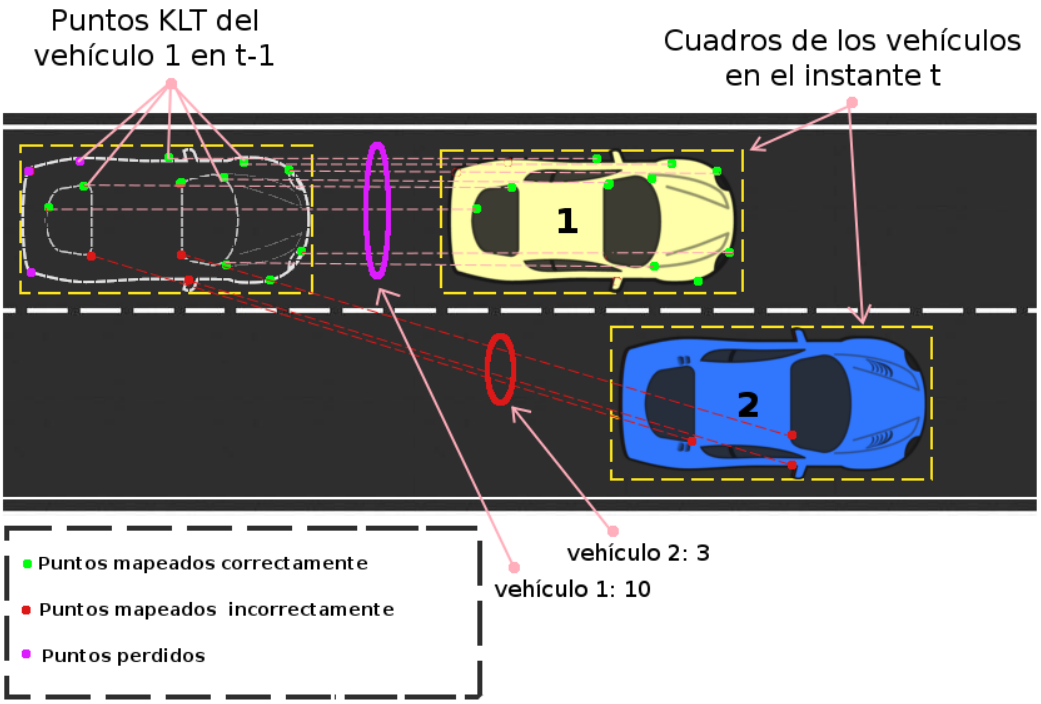


Figura 4.24: Asociación entre blob y vehículo mediante el algoritmo de votación

sus puntos son regenerados para obtener un conjunto de características más actualizado para la imagen actual. Si no somos capaces de asociarlo por proximidad sea porque ha entrado en oclusión o cualquier otra razón entonces hacemos un segundo intento, esta vez utilizando el algoritmo de KLT piramidal para hallar su nueva posición. En caso de encontrarlo los puntos de KLT son regenerados si y sólo si el vehículo no ha entrado en oclusión. En dicho caso los puntos se mantienen hasta que esta condición desparece o el proceso de seguimiento para este vehículo llega a su fin. Mientras dura la oclusión la posición del vehiculo es actualizada mediante su cuadro FPBB correspondiente a los puntos KLT. El diagrama 4.25 resume los pasos del algoritmo de seguimiento.

4.2.4. Resolución de oclusiones

Independientemente del algoritmo elegido para llevar a cabo el seguimiento, la oclusión se puede considerar uno de los problemas más complicados que hay que resolver. Se trata de un problema típico en el seguimiento de múltiples objetos. En el escenario de tráfico de vehículos puede surgir cuando hay mucho tráfico en la carretera o simplemente con la entrada de vehículos grandes como camiones o autobuses en la escena. En la sección 3.2, durante el estudio preliminar, hemos visto que entre los criterios que se tienen en cuenta a la hora de elegir la posición y

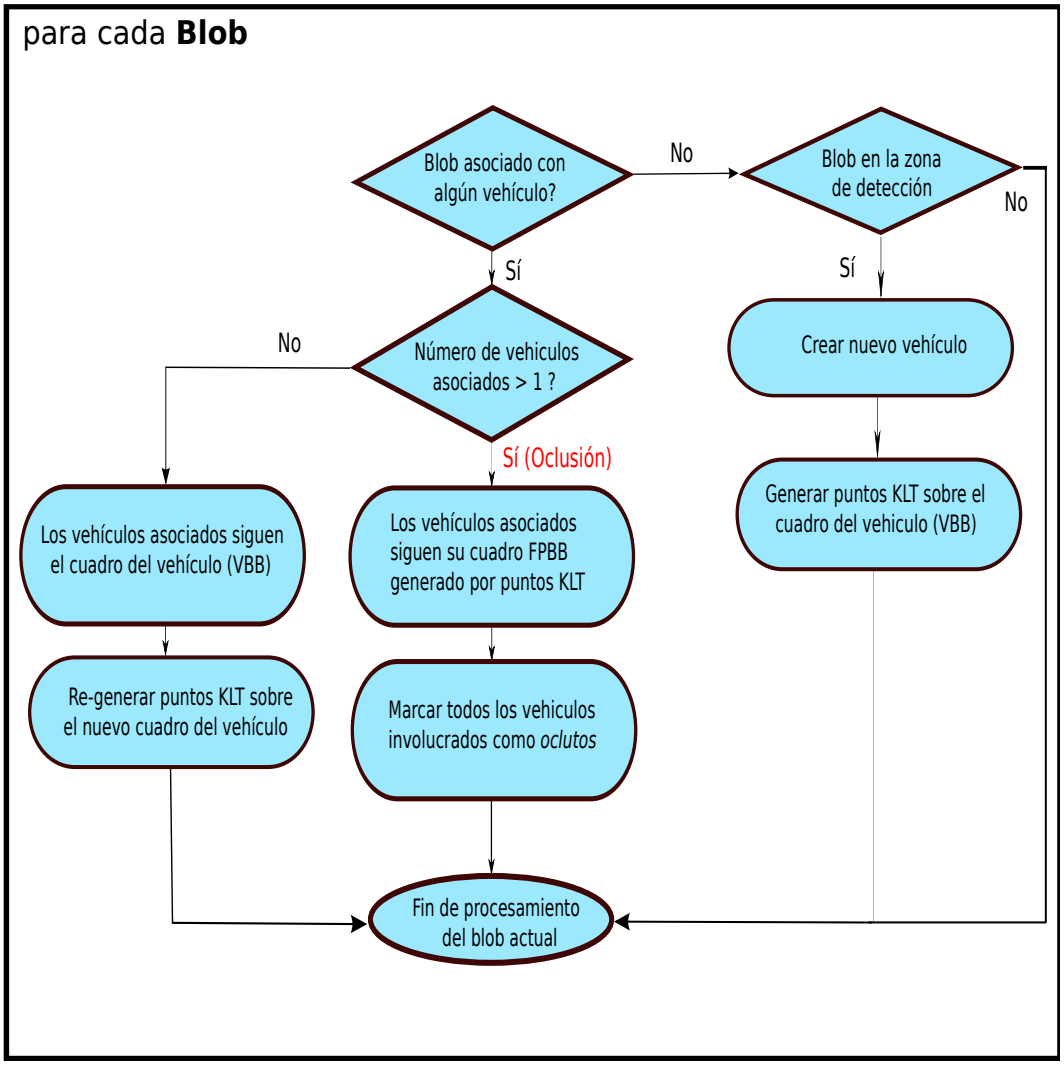


Figura 4.25: Algoritmo de seguimiento de vehículos

orientación de la cámara está el de evitar las oclusiones. Una buena posición de la cámara ayuda a minimizar las oclusiones hasta cierto punto pero no a eliminarlas del todo. Este problema siempre existirá y hay que tomar medidas para detectarlo y resolverlo. De otra manera cualquier error en la etapa de seguimiento puede propagarse a la fase de clasificación.

Dentro del campo de monitorización de tráfico basada en visión podemos encontrar varios trabajos empleando distintas técnicas para llevar a cabo esta tarea. Sin embargo, en la bibliografía revisada como parte de esta tesis la oclusión es uno de los problemas que se suele obviar o tratar de manera superficial. Más sorprendente aún es la escasez de trabajos dedicados exclusivamente a resolver este problema dada su importancia e impacto sobre los resultados tanto del seguimiento como de la clasificación.

Si analizamos los trabajos relacionados vemos que en [Pang, 2007] se utiliza

un modelo genérico deformable para representar los vehículos. La idea básica se basa en emplear una cámara calibrada para detectar los segmentos del contorno pertenecientes a los vehículos y utilizar la información de la escena (los puntos de fuga) para analizarlos y detectar los vehículos que están parcialmente ocultos. A cada uno de ellos se le asigna una probabilidad de resolución dependiendo del número de segmentos que han sido detectados y su configuración. Cuando el sistema determina que el vehículo es *recuperable* se utiliza la información de geometría de la escena (básicamente los puntos de fuga) para recuperar el modelo 3D del vehículo. Los mismos autores ya habían presentado un trabajo similar [Pang, 2004], sin embargo en ese caso el objetivo estaba enfocado a resolver el problema de oclusión para el caso de dos vehículos.

[Lili Huang, 2010] aborda el problema de las oclusiones de forma distinta. En este caso el objetivo no es una segmentación perfecta de cada vehículo involucrado en la oclusión, sino más bien resolver el problema de la oclusión e identificar las zonas que pertenecen a los vehículos involucrados. Para ello, los autores hacen uso de un algoritmo que combina tanto KLT como un modelo de color basado en *mean-shift*. Cada vehículo es representado en forma de una elipse y un conjunto de puntos característicos escogidos dentro del mismo. Este modelo se utiliza para estimar la posición del vehículo entre dos fotogramas consecutivos. De esta manera, cuando el vehículo entra en oclusión con otro la elipse y los puntos escogidos se utilizan para estimar la posición del vehículo en el blob resultante de la oclusión. Éste último se divide en parches de 3×3 píxeles cada uno. Dependiendo de su posición, estos son marcados como ocultos o no (los parches que pertenecen al mismo objeto son marcados de forma directa como no ocultos). Los parches que se marcan como ocultos pasan una etapa adicional de procesamiento donde se determina a qué vehículo pertenecen basándose esta vez en la información del color.

Antes de entrar en detalle del algoritmo propio para resolver las oclusiones implementado en esta tesis vamos a exponer los objetivos de nuestro caso particular. Éstos consisten en:

- Detectar las oclusiones justo en el momento de ocurrir y detener la clasificación de los vehículos involucrados
- Identificar todos los vehículos involucrados y seguirlos de forma correcta
- Ser capaces de recuperarse de oclusiones parciales
- Identificar las oclusiones totales cuando ocurran

- Detectar oclusiones tempranas (que se producen en la zona de entrada)

Listados los objetivos hay que resaltar que no forma parte de los mismos segmentar los blobs resultantes de una oclusión de dos o más vehículos. Esta decisión se ha tomado en fases iniciales dado que el objetivo final de nuestro trabajo es clasificar los vehículos en varias categorías. Dicho esto y como nuestro criterio principal para la clasificación es el tamaño no tiene sentido intentar clasificar vehículos que están involucrados en una oclusión. De hacerlo, el hecho de mapear la huella producida por el conjunto de vehículos en oclusión a 3D podría generar más perjuicios que beneficios para el resultado final de clasificación. En nuestro caso, como veremos en el próximo capítulo de clasificación, intentamos clasificar el vehículo siempre cuando esté totalmente visible. Como los datos de clasificación son acumulados a lo largo de la zona de entrada basta con tener el vehículo bien visible (no oculto) durante un pequeño tramo para clasificarlo de forma correcta.

En la fase anterior del algoritmo, el sistema ha asociado los vehículos en seguimiento con sus correspondientes blobs detectados en el cuadro actual. Durante la fase de seguimiento el sistema actualiza todos los parámetros del vehículo (cuadro envolvente, centro de masas y puntos característicos) para todos los vehículos que han sido correctamente asociados. Durante esta fase el sistema identifica los siguientes casos:

Nuevos vehículos: son los blobs cuyo centro 2D está en la zona de entrada y que no han sido asociados con ningún vehículo en seguimiento. Para cada uno de éstos el sistema crea un nuevo vehículo y lo inicializa con los parámetros del blob: cuadro envolvente y centro de masas. El seguimiento es inicializado con la obtención de puntos característicos sobre el cuadro del vehículo.

Seguimiento Trivial: estos son los blobs que han sido asociados con un solo vehículo. En este caso el sistema actualiza el cuadro y la localización del vehículo con las del blob asociado y regenera los puntos característicos de *KLT* empleando el nuevo cuadro del vehículo.

Oclusiones: son identificadas cuando dos o más vehículos en seguimiento son asociados con el mismo blob (ver figura 4.26)

Durante una oclusión el centro del blob resultante se desplaza en una distancia grande respecto a los centros de los vehículos en oclusión sobre todo si hay un vehículo de gran tamaño involucrado en la misma. Como consecuencia, el seguimiento por proximidad falla, puesto que el centro del blob en oclusión cae fuera de la elipse asociada a cada vehículo. Ante esta situación, nuestro algoritmo hace uso de los puntos característicos de KLT generados en la iteración anterior para

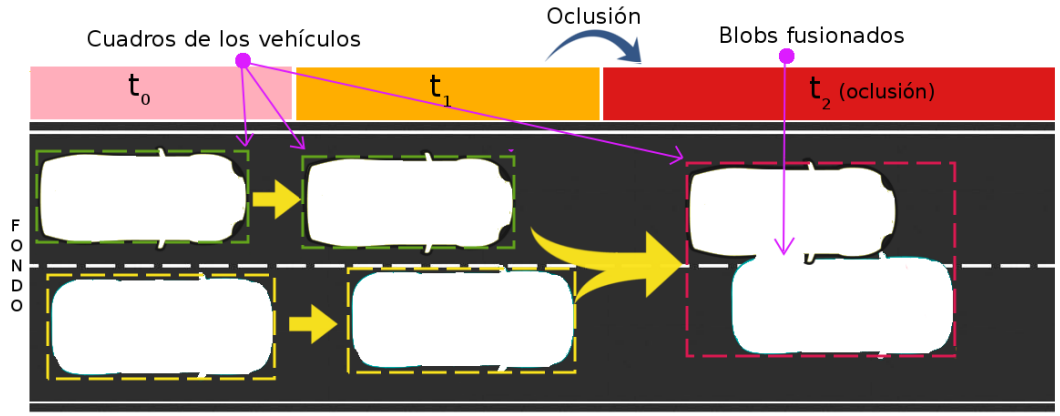


Figura 4.26: Proceso de detección de oclusión entre dos vehículos

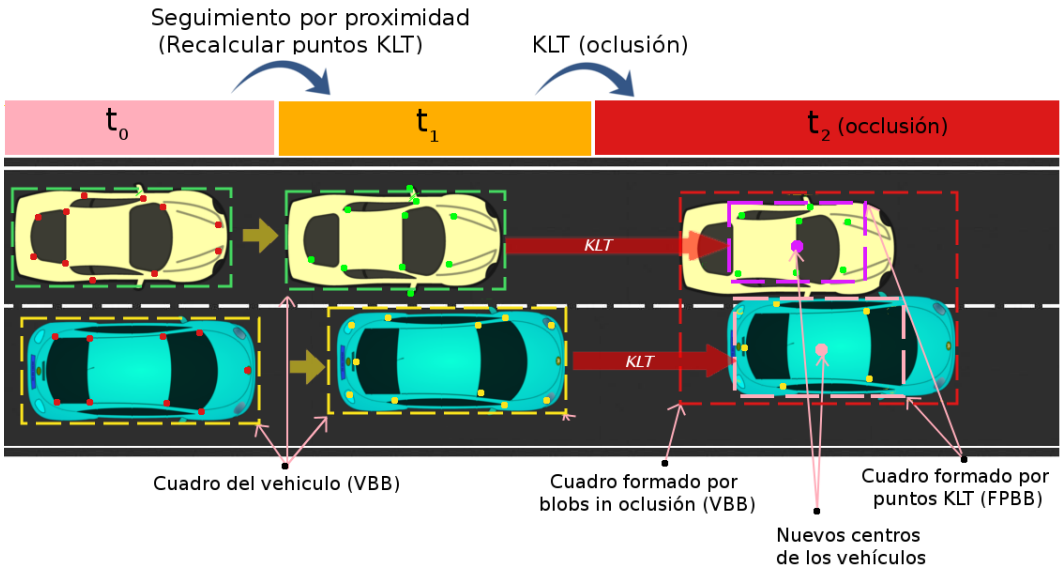


Figura 4.27: Reglas del cálculo de los puntos característicos KLT

buscar la nueva posición del vehículo iniciando una búsqueda mediante el algoritmo de KLT piramidal. El resultado de este proceso es un conjunto de nuevos puntos asociados a los puntos originales. El tamaño de este conjunto puede llegar a ser menor que el conjunto inicial de búsqueda debido a que algunos puntos se pueden perder entre dos fotogramas consecutivos por cambios en la imagen producidos por oclusiones, ruido, etc. Nuestro sistema utiliza este conjunto para calcular el nuevo centro del vehículo y utilizarlo para el seguimiento (cuadro FPBB del vehículos). A diferencia de los casos anteriores donde los puntos característicos de KLT son regenerados, en este caso los puntos se mantienen tanto tiempo como dure la oclusión.

La figura 4.27 muestra un ejemplo de oclusión. En este caso los vehículos entran en oclusión en el instante t_2. El nuevo cuadro FBPP correspondiente al

vehículo es más pequeño en este caso porque parte de los puntos característicos del vehículo se han perdido durante el mapeo. Finalmente en la figura 4.29 podemos ver un caso real de oclusión. En este caso los vehículos 77,78 y 79,80 están entrando en oclusión respectivamente. El identificador de los vehículos que están involucrados en una oclusión es marcado en color rojo.

En general podemos distinguir entre dos tipos de oclusiones: *parcial* y total. En la primera de ellas existe una alta probabilidad de que la situación se acabe resolviendo en las próximas iteraciones. Un ejemplo de este caso es cuando dos vehículos están viajando en carriles adyacentes y uno de ellos es más rápido que el otro. En este caso puede haber una oclusión que dure por unas cuantas iteraciones pero que se acabará resolviendo. En este caso, deberíamos ser capaces de retomar y seguir con la clasificación para ambos vehículos. La figura 4.28 muestra un ejemplo de este caso.

El segundo tipo de oclusión es más difícil de resolver. Ocurre cuando un vehículo oculta por completo a otro. Este caso se puede dar cuando un vehículo de gran tamaño entra en escena y tapa a uno o varios vehículos más pequeños. En estos casos nuestro algoritmo intenta seguir el vehículo hasta que éste pasa a ser completamente invisible (el número de puntos KLT mapeados en el fotograma es cero). En este momento el proceso de seguimiento es terminado para este vehículo y calculamos sus estadísticas en base a la información acumulada hasta ese momento. De nuevo, en estos casos hay una serie de comprobaciones básicas para asegurar de que se trata de un vehículo valido.

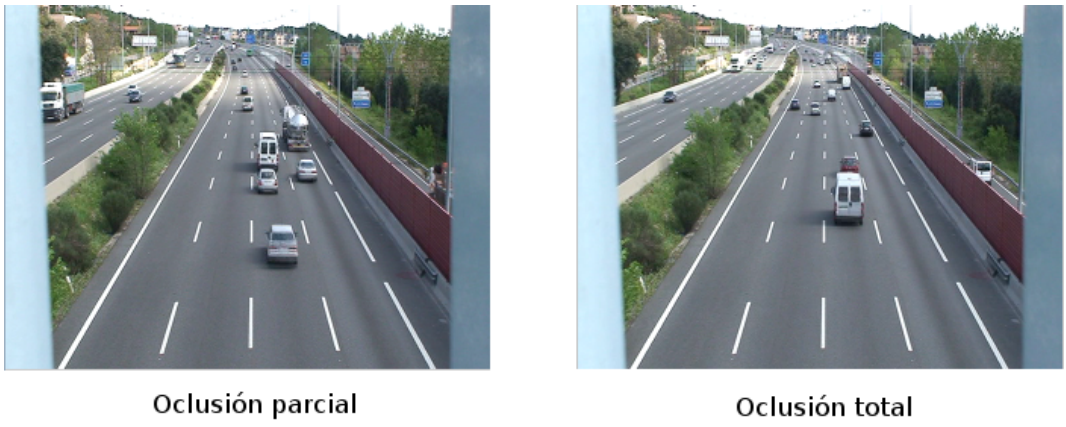


Figura 4.28: **Oclusión parcial**: uno o varios vehículos están parcialmente ocultados pero el seguimiento puede seguir hasta que vuelvan a ser visibles. **Oclusión Total**: a medida que los vehículos se acercan al horizonte la furgoneta acabará tapando por completo al coche

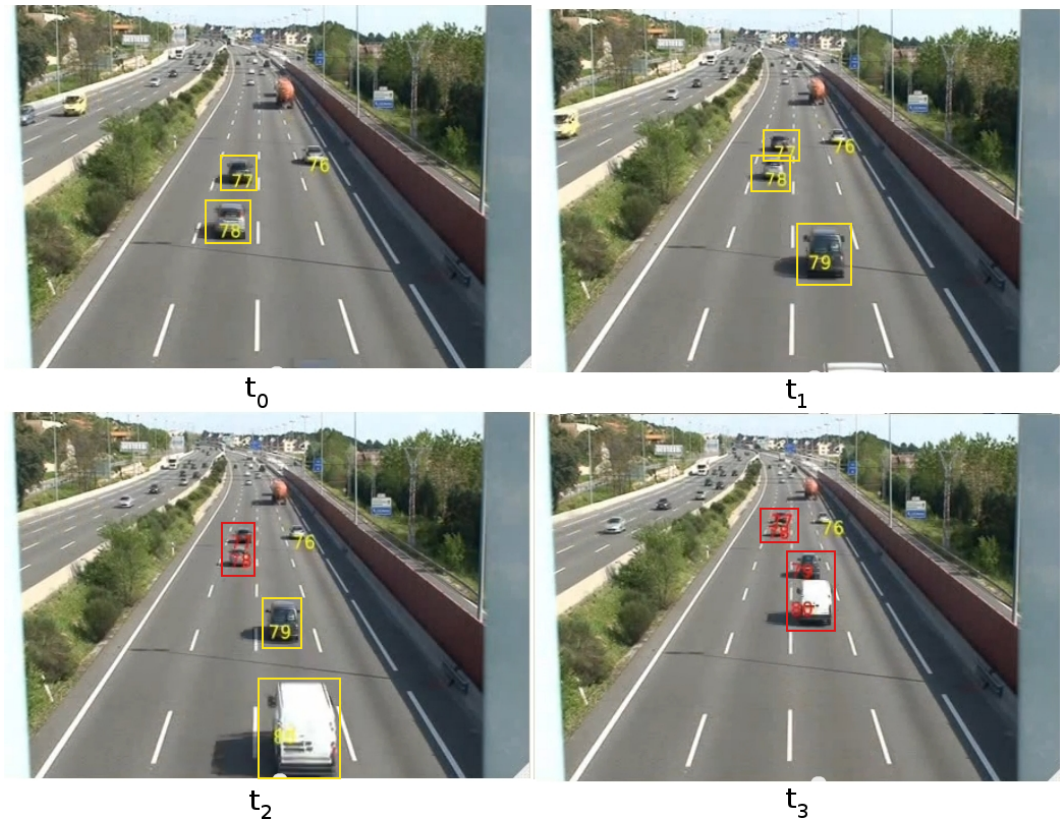


Figura 4.29: Detección de oclusión: los vehículos 77,78 y 79, 80 están entrando en oclusión (el color del identificador de los vehículos en oclusión cambia de amarillo a rojo)

4.2.5. Estimación de la velocidad

Dado que disponemos de la información 3D de la escena gracias a la cámara calibrada somos capaces de medir la distancia recorrida por el vehículo desde el punto de entrada hasta abandonar la zona de seguimiento. Estamos asumiendo que el vehículo viaja en línea recta a una velocidad constante a lo largo del eje Y. Esta información combinada con la medición de tiempo y posición 3D en cada momento del vehículo nos permite estimar la velocidad de tramo del vehículo de forma sencilla. Las estadísticas acerca del recorrido de cada vehículo son guardadas en una base de datos con la idea de utilizarla a posteriori para hacer análisis más complejos acerca del comportamiento de vehículos dependiendo de su categoría y sacar más métricas sobre el tráfico. La figura 4.30 ilustra proceso de cálculo de velocidad.

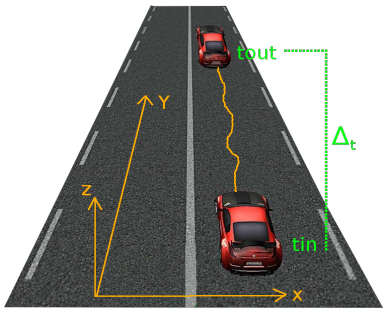


Figura 4.30: Estimación de velocidad

4.3. Experimentos sobre tráfico real

Para validar la etapa de seguimiento se han llevado a cabo varios experimentos en distintas condiciones de tráfico y meteorología. Estos se han basado en la base de datos propia descrita en la sección 3.3 y la base de datos *GRAM-RTM* creada por [Guerrero-Gomez-Olmedo, R. and Lopez-Sastre, R. J. and Maldonado-Bascon, S. and Fernandez-Caballero, A., 2013]. Esta última ha sido empleada sobre todo para probar el sistema y evaluar su rendimiento en condiciones de niebla y algunos escenarios soleados. En la figura 4.15 se puede ver un ejemplo de una escena de seguimiento. Cada vehículo tiene asignado un identificador único que le acompaña durante toda la zona de seguimiento.

4.3.1. Métricas

La validación del sistema se basa principalmente en dos medidas para evaluar los resultados obtenidos: *precisión* y *exhaustividad*. Ambas son métricas estándar que han sido introducidas originalmente para medir el rendimiento sistemas de búsqueda de información. Sin embargo, las mismas métricas son utilizadas en otros campos como la medicina o la estadística para evaluar la calidad de sistemas de clasificación. Estas métricas han sido adoptadas en el campo de inteligencia artificial para evaluar el rendimiento de clasificadores binarios o multiclase en tareas como clasificación o reconocimiento de patrones en general. El concepto de precisión fue propuesto inicialmente por [James W. Perry, 1955] y es calculado como la fracción entre los valores correctos y los valores recuperados. En nuestro contexto es calculado como la fracción entre el número de vehículos correctamente detectados y el número total de vehículos detectados.

La exhaustividad (*recall* en inglés) hace referencia a la *sensibilidad* del sistema de clasificación para detectar las muestras relevantes. En nuestro contexto es calculada como la fracción entre el número total de vehículos correctamente

detectados y el número total de vehículos que han transitado por la escena. Cabe destacar que ambas métricas son a menudo criticadas por no tener en cuenta de forma directa los casos negativos. En este contexto esto sería la capacidad del sistema para descartar ruido (detecciones que no son vehículos). Este aspecto es recogido de forma indirecta por la precisión dado que si el sistema genera muchas detecciones falsas el valor de esta métrica bajará considerablemente.

Adicionalmente, tanto la precisión como la exhaustividad tienen una interpretación probabilística: la primera representa la probabilidad de que una detección seleccionada de forma aleatoria sea un vehículo mientras que la segunda es la probabilidad de que un vehículo que pasa por la escena sea detectado por el sistema. En otras palabras la precisión determina la fiabilidad del sistema cada vez que reporta la detección de un vehículo mientras que la exhaustividad determina su capacidad para la detección de los vehículos.

En nuestro contexto ambas métricas son importantes y nuestro objetivo es maximizar las dos medidas. El sistema ideal es aquel que es capaz de detectar todos los vehículos que pasan por la carretera y además todas las detecciones que reporta son correctas. Dicho esto, en la práctica no tiene utilidad un sistema que tiene un 100 % de precisión si su exhaustividad es muy baja. Éste es el caso de un sistema que detecta muy pocos vehículos pero las detecciones que reporta son siempre correctas. De forma análoga, tampoco valdría un sistema con alta exhaustividad pero muy baja precisión. Esto pasa cuando tenemos un sistema que detecta todos los vehículos que pasan pero además genera muchos falsos positivos.

Ponderar un sistema de detección y seguimiento en función de la precisión y exhaustividad por separado no es sencillo. De ahí que exista otras métricas que combinan ambas en un solo valor. La más famosa es el *Valor-f* (F-Measure en inglés). Este valor es calculado de forma genérica con la siguiente fórmula introducida por [Rijsbergen, 1979]:

$$F_\beta = (1 + \beta^2)\frac{P \cdot E}{\beta^2 \cdot P + E} \tag{4.7}$$

Donde E es la exhaustividad, P la precisión y β es un factor que regula la importancia que le damos a la exhaustividad frente a la precisión:

- $\beta = 1$: la precisión tiene la misma importancia que la exhaustividad
- $\beta < 1$: La precisión tiene más importancia que la exhaustividad
- $\beta > 1$: La exhaustividad tiene más importancia (β veces más) que la precisión

En nuestro contexto tanto la precisión como la exhaustividad son factores muy importantes, así que hemos optado por F_1 (Valor-F con $\beta = 1$) como medida combinada para evaluar los resultados obtenidos en los distintos experimentos de seguimiento que hemos llevado a cabo.

4.3.2. Robustez del seguimiento frente a las oclusiones

Como ya se ha visto en la sección 4.2, detectar las oclusiones y resolverlas es uno de los objetivos principales de esta tesis. Es una forma de evitar posteriores errores en la clasificación al intentar clasificar dos o más vehículos que han entrado en oclusión. Se han llevado a cabo varios experimentos con distintos niveles de oclusiones con los tres algoritmos de seguimiento que se han implementado en la tesis:

- Algoritmo basado en proximidad espacial
- Algoritmo basado en *KLT*
- Combinación de proximidad con *KLT*

En todos los experimentos se ha contabilizado (y derivado) los siguientes parámetros:

- **Total**: Verdad absoluta, número calculado por un humano
- $\boldsymbol{T_d}$: es el número de vehículos detectados por el sistema
- $\boldsymbol{T_p}$: es el número de vehículos reportados correctamente
- $\boldsymbol{F_p}$: es el número de falsos positivos (detecciones erróneas de vehículos)
- $\boldsymbol{F_p}$: es el número de falsos negativos (vehículos no detectados)
- $\boldsymbol{Err}(\%)$: es el número de errores de seguimiento $\boldsymbol{Err} = \frac{\boldsymbol{Numerrores}}{\boldsymbol{T_d}}$
- $\boldsymbol{P}(\%)$: La precisión $\boldsymbol{P} = \frac{\boldsymbol{T_p}}{\boldsymbol{T_d}}$
- $\boldsymbol{R}(\%)$: La exhaustividad $\boldsymbol{R} = \frac{\boldsymbol{T_p}}{\boldsymbol{Total}}$
- $\boldsymbol{F1}(\%)$: El valor-F calculado con la fórmula 4.7 con $\boldsymbol{\beta = 1}$

La métrica *Err* indica el número de errores que hubo durante el seguimiento. La precisión y la exhaustividad son buenos indicadores para evaluar el comportamiento del sistema, sin embargo, ninguno de ellos refleja la fiabilidad del

seguimiento como proceso continuo. Podemos tener un sistema que detecta los vehículos de forma correcta en los fotogramas de forma separada pero esto es insuficiente si además queremos que sea capaz de seguir los vehículos de forma correcta entre los distintos fotogramas. Dicho esto, cada vez que el vehículo pierde su identificador único el número de errores de seguimiento es incrementado en uno. La precisión del seguimiento es importante en nuestro caso dado que los resultados de clasificación para cada vehículo son acumulados durante toda la zona de seguimiento. De ahí que necesitamos seguir los vehículos y no perderles la pista durante toda esta fase si queremos clasificarlos correctamente.

La tabla 4.1 muestra los resultados obtenidos para varios vídeos representativos en un escenario sin oclusiones. Si analizamos la media del Valor-F para los tres algoritmos podemos ver que es mayor que 98 % para todos ellos. En ausencia de oclusiones los tres algoritmos son igualmente válidos para llevar a cabo la tarea del seguimiento. Esto tiene lógica, dado que en este caso la asociación entre dos fotogramas consecutivos en general es trivial y es la razón de que el porcentaje de errores de seguimiento cometidos para los tres algoritmos sea tan bajo.

video	Algoritmo	Total	Td	Tp	Fp	Fn	Err(%)	P(%)	R(%)	F_1(%)
video-0008	Proximidad	15	15	15	0	0	0.00	100	100	100
video-0008	KLT	15	15	15	0	0	0.00	100	100	100
video-0008	Combinación	15	15	15	0	0	0.00	100	100	100
video-0071	Proximidad	70	70	70	0	0	0.00	100	100	100
video-0071	KLT	70	70	70	0	0	0.00	100	100	100
video-0071	Combinación	70	69	69	0	0	1.45	100	98.57	99.28
video-0072	Proximidad	125	123	123	0	0	1.63	100	98.40	99.19
video-0072	KLT	125	123	123	0	0	1.63	100	98.40	99.19
video-0072	Combinación	125	123	123	0	0	1.63	100	98.40	99.19
video-0086	Proximidad	120	120	117	3	3	0.85	97.50	97.50	97.50
video-0086	KLT	120	120	116	4	4	0.86	96.67	96.67	96.67
video-0086	Combinación	120	122	119	3	3	0.84	97.54	99.17	98.35
video-0102	Proximidad	268	265	264	2	2	1.52	99.62	98.51	99.06
video-0102	KLT	268	265	264	2	2	1.52	99.62	98.51	99.06
video-0102	Combinación	268	263	261	2	2	2.68	99.24	97.39	98.31
	Proximidad						0.8	99.42	98.88	99.15
Media	KLT						0.67	99.26	98.71	98.98
	Combinación						1.10	99.36	98.71	99.03

Cuadro 4.1: Resultados de seguimiento en escenarios sin oclusiones

La tabla 4.2 muestra los resultados obtenidos para vídeos en escenarios con una baja cantidad de oclusiones. A diferencia del experimento anterior, en este caso la asociación de vehículos entre dos cuadros no es trivial cuando éstos entran en oclusión. De nuevo los resultados obtenidos siguen siendo buenos (todos mayores de 96 %) pero la cantidad de errores cometidos durante el seguimiento empieza a subir, sobre todo para el algoritmo basado en la proximidad. Es

importante recalcar que en este caso la precisión media de los tres algoritmos sigue siendo la misma, sin embargo, la exhaustividad ha disminuido respecto al experimento anterior. Esto queda reflejado en el Valor-F de los tres algoritmos. La figura 4.31 muestra varios ejemplos de seguimiento con pocas oclusiones.

video	Algoritmo	Total	Td	Tp	Fp	Fn	Err	P(%)	R(%)	F_1(%)
video-0075	Proximidad	132	131	128	3	3	1.56	97.71	96.97	97.34
video-0075	KLT	132	134	132	2	2	0.76	98.51	100.00	99.25
video-0075	Combinación	132	133	131	2	2	0.76	98.50	99.24	98.87
video-0044	Proximidad	82	73	73	0	9	2.7	100.00	89.02	94.19
video-0044	KLT	82	80	80	0	2	0.00	100.00	97.56	98.77
video-0044	Combinación	82	79	79	0	3	0.00	100.00	96.34	98.14
video-0026	Proximidad	165	160	160	0	0	3.13	100.00	96.97	98.46
video-0026	KLT	165	161	161	0	0	1.86	100.00	97.58	98.77
video-0026	Combinación	165	162	162	0	0	1.85	100.00	98.18	99.08
video-0086	Proximidad	119	117	114	3	3	4.39	97.44	95.80	96.61
video-0086	KLT	119	116	113	3	3	5.31	97.41	94.96	96.17
video-0086	Combinación	119	118	115	3	3	3.48	97.46	96.64	97.05
video-0079	Proximidad	138	134	134	0	0	2.99	100.00	97.10	98.53
video-0079	KLT	138	137	136	1	1	2.21	99.27	98.55	98.91
video-0079	Combinación	138	135	135	0	0	2.22	100.00	97.83	98.90
	Proximidad						2.95	99.36	94.72	96.95
Media	KLT						2.02	99.04	97.73	98.37
	Combinación						1.66	98.99	97.60	98.28

Cuadro 4.2: Resultados de seguimiento en escenarios con pocas oclusiones

Finalmente, la tabla 4.3 muestra los resultados obtenidos para varios vídeos esta vez con un cantidad media de oclusiones. Los nuevos datos reflejan el impacto que tiene el aumento de las oclusiones sobre los tres algoritmos probados en el experimento. Mientras el porcentaje de errores de los algoritmos basados en *KLT* se mantiene bajo, en el caso del algoritmo basado en la proximidad espacial este número supera el 10 %. Esto tiene un impacto directo sobre la exhaustividad de este algoritmo que en este caso se sitúa por debajo del 90 %. Los algoritmos basados en *KLT* también se han visto afectados pero mantienen una exhaustividad media de 95 % y un porcentaje de error cercano al 4 %.

A la luz de los resultados obtenidos queda claro que el algoritmo de seguimiento por proximidad no maneja bien los escenarios con oclusiones. Las siguientes pruebas que veremos han sido todas realizadas con el algoritmo que combina *KLT* con la proximidad para el seguimiento.

4.3.2.1. Capacidad del sistema para detectar y resolver las oclusiones

Para evaluar el desempeño del algoritmo de detección y resolución de oclusiones se han llevado a cabo experimentos sobre varios vídeos representativos. Cada vez que el sistema detecta una oclusión entre dos o más vehículos éstos se

Figura 4.31: Ejemplos de seguimiento con pocas oclusiones

video	Algoritmo	Total	Td	Tp	Fp	Fn	Err(%)	P(%)	R(%)	F_1(%)
video-0040	Proximidad	60	55	54	1	5	9,26	98,18	90,00	93,91
video-0040	KLT	60	61	59	2	0	5,08	96,72	98,33	97,52
video-0040	Combinación	60	60	59	1	0	10,17	98,33	98,33	98,33
video-0049	Proximidad	111	103	103	0	2	11,65	100,00	92,79	96,26
video-0049	KLT	111	111	110	1	2	1,82	99,10	99,10	99,10
video-0049	Combinación	111	110	109	1	2	1,83	99,09	98,20	98,64
video-0051	Proximidad	103	89	89	0	14	13,48	100,00	86,41	92,71
video-0051	KLT	103	95	95	1	8	2,11	100,00	92,23	95,96
video-0051	Combinación	103	97	97	1	6	3,09	100,00	94,17	97,00
video-0057	Proximidad	82	72	72	0	9	5,33	100,00	87,80	98,04
video-0057	KLT	82	75	75	0	6	4,00	100,00	91,46	95,54
video-0057	Combinación	82	71	71	0	10	5,63	100,00	86,59	92,81
video-0058	Proximidad	93	80	80	0	13	12,5	100,00	86,02	92,49
video-0058	KLT	93	85	84	1	8	2,38	98,82	90,32	94,38
video-0058	Combinación	93	88	87	1	6	1,15	98,86	93,55	96,13
Media	Proximidad						10.43	99.64	88.61	93.78
	KLT						3.03	98.96	95.28	97.03
	Combinación						4.33	99.36	94.32	96.70

Cuadro 4.3: Resultados de seguimiento en escenarios con oclusiones

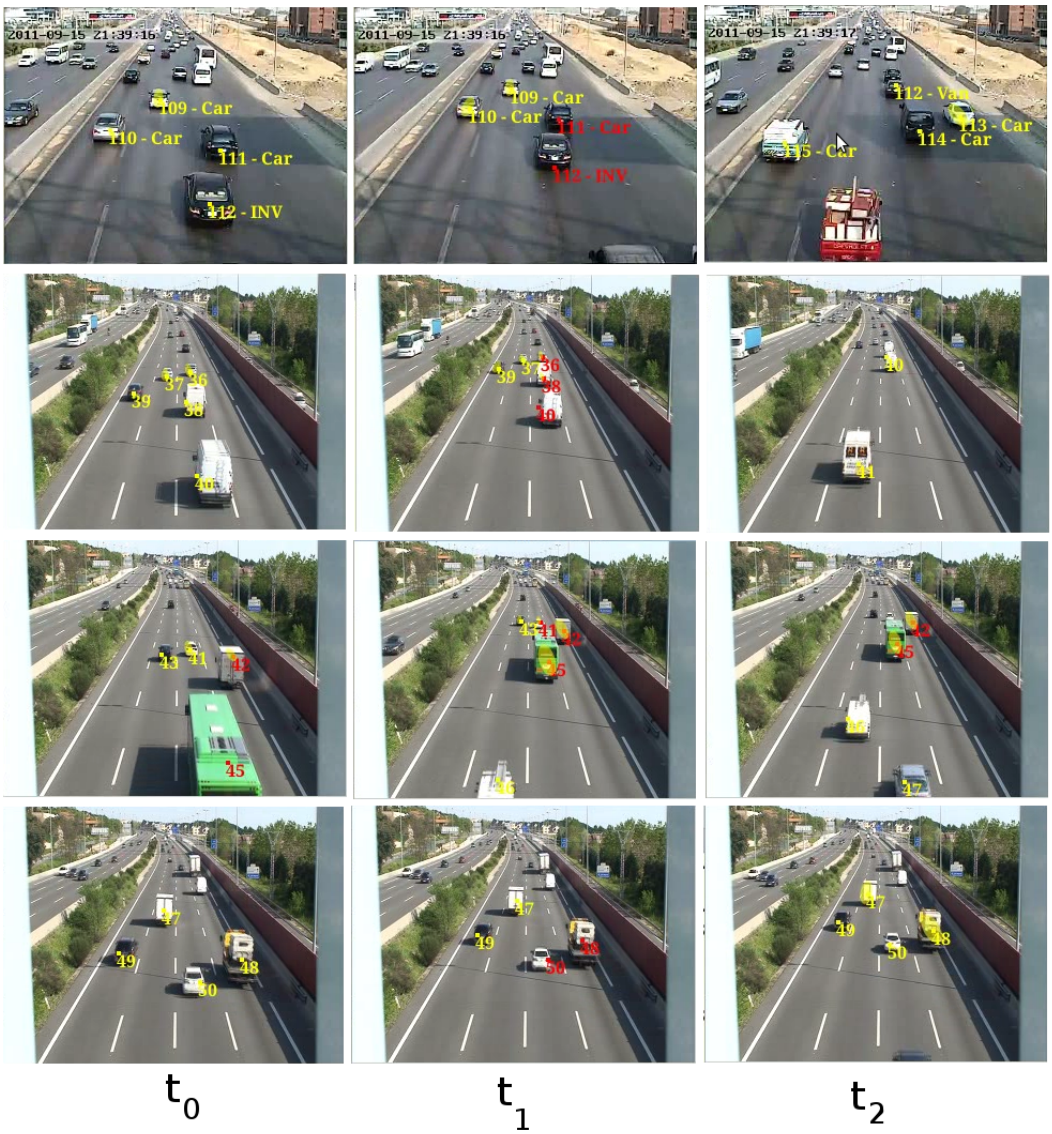


Figura 4.32: Secuencias de detección de oclusión

marcaran como *vehículos en oclusión* y el sistema dejará de acumular resultados de clasificación para los mismos.

La tabla 4.4 muestra los resultados de la detección de oclusiones para varios vídeos. Las columnas de la tabla tienen el siguiente significado:

- **Total**: Verdad absoluta, número de oclusiones calculado por un humano
- $\boldsymbol{T_d}$: es el número de oclusiones detectados y reportadas por el sistema
- $\boldsymbol{T_p}$: es el número de oclusiones reportados correctamente
- $\boldsymbol{F_p}$: es el número de falsos positivos (detecciones erróneas de oclusiones)
- $\boldsymbol{F_n}$: es el número de falsos negativos (oclusiones no detectados)

- $\boldsymbol{T_r}$: es el número de oclusiones que el sistema ha resuelto correctamente
- $\boldsymbol{T_{nr}}$: es el número de oclusiones que el sistema no ha sido capaz de resolver
- $\boldsymbol{P}(\,\%)$: La precisión $\boldsymbol{P} = \frac{\boldsymbol{T_p}}{\boldsymbol{T_d}}$
- $\boldsymbol{R}(\,\%)$: La exhaustividad $\boldsymbol{R} = \frac{\boldsymbol{T_p}}{\boldsymbol{Total}}$
- $\boldsymbol{F_1}(\,\%)$: El valor-F calculado con la fórmula 4.7 con $\boldsymbol{\beta = 1}$

Analizando los resultados de esta tabla se puede ver que la capacidad del sistema para detectar y resolver las oclusiones tiene un Valor-F de 83 %, con una precisión cercana al 90 % y una exhaustividad del 80 %. Éste último valor es más bajo que la precisión debido a la dificultad que tiene en la práctica detectar las oclusiones en ciertos escenarios. Dicho esto, el sistema es capaz de detectar y aislar las oclusiones en la mayoría de los casos. La figura 4.32 muestra varios ejemplos del mecanismo de detección de oclusiones en funcionamiento. Cada vez que se detecta una oclusión, el color de los identificadores únicos de los vehículos que están involucrados en la misma pasa a ser rojo. Si el sistema consigue resolver la oclusión sea porque los vehiculos ya no están ocultos o porque la oclusión es total entonces el identificador del vehículo recupera el color amarillo.

video	Total	T_d	T_p	F_p	F_n	T_r	T_{nr}	P(%)	R(%)	F_1(%)
video-0046	10	9	8	8	1	8	1	88.89	80.00	84.21
video-0048	14	16	12	4	2	12	0	75.00	85.71	80.00
video-0049	22	21	21	3	1	21	0	100.00	95.45	97.67
video-0051	9	5	5	3	4	4	1	100.00	55.56	71.43
video-0058	13	12	12	1	1	12	0	100.00	92.31	96.00
video-0059	18	20	15	3	1	13	2	75.00	83.33	78.95
video-0060	9	7	6	1	1	6	0	85.71	66.67	75.00
Media								89.23	79.86	83.32

Cuadro 4.4: Resultados de la detección de oclusiones en varios escenarios

La figura 4.33 muestra unos cuantos ejemplos donde el sistema no ha sido capaz de identificar de forma correcta la oclusión. En la escenas (a), (b), (d) y (f) entran en la escena varios vehiculos de forma consecutiva sin que el sistema sea capaz en ningún momento de detectar su entrada por separado al estar ocultos en la zona de entrada. En la escena (c) la zona de entrada está situada en una zona con sombras dificultando la detección de algunos vehículos. Finalmente, en la escena (e) la oclusión entre el camión con el identificador 20 y el camión de color verde no es detectada porque el blob de este último es eliminado incorrectamente por el mecanismo de detección y eliminación de sombras.

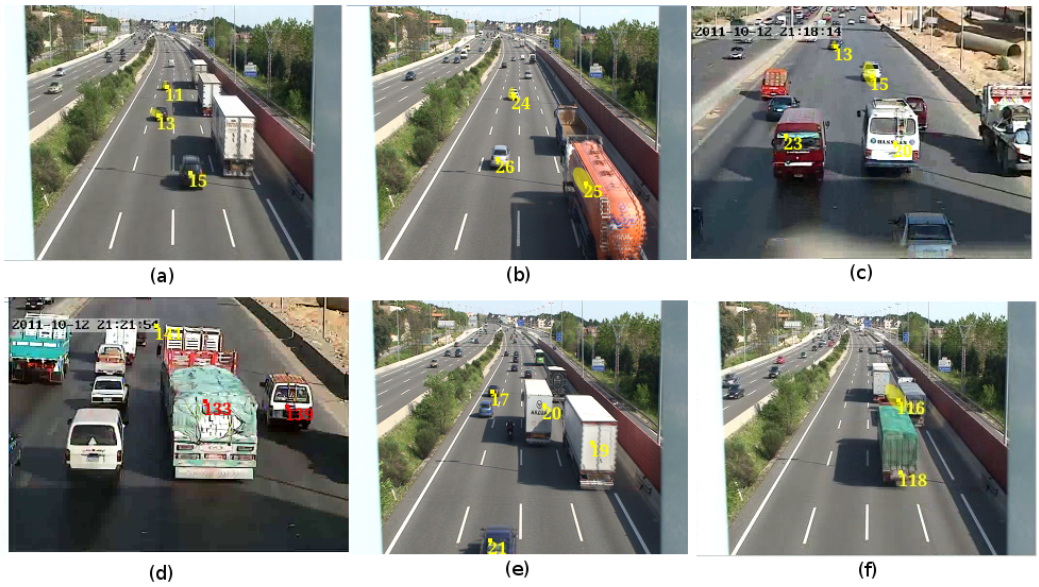


Figura 4.33: Ejemplos de fallos de detección de oclusión

4.3.3. Evaluación del sistema de detección y eliminación de sombras

Para comprobar la efectividad del mecanismo de detección y eliminación de sombras se han llevado a cabo varios experimentos sobre varios vídeos que representan distintas escenas con sombras. Las pruebas se han repetido dos veces sobre los mismos vídeos, primero con el mecanismo de detección de sombras desactivado y a continuación con esta funcionalidad activada. El objetivo es evaluar la mejora introducida por este mecanismo en los resultados de seguimiento. La tabla 4.5 recoge los resultados para cada vídeo.

Analizando los datos se puede ver que los resultados obtenidos con el mecanismo de detección de sombras son mejores que cuando este mecanismo está deshabilitado. El parámetro que mejor refleja la diferencia es el de la exhaustividad. Mientras que el valor de la media de esta métrica cuando la detección de sombras está deshabilitada es de 91 % cuando habilitamos esta funcionalidad la media mejora de forma considerable hasta alcanzar el 98 % sobre el mismo conjunto de vídeos. Un efecto parecido es observado con el número de errores cometido en el seguimiento.

Si analizamos con más detalle los datos se puede ver que la métrica que provoca esta diferencia entre ambos experimentos es la columna F_n que corresponde al número de vehículos no detectados. Este efecto tiene una explicación sencilla: básicamente cuando hay sombras y la función de detección está deshabilitada todos los vehículos que están unidos por su sombra en la zona de entrada cuentan

vídeo	Detección de Sombras	Total	Td	Tp	Fp	Fn	Err(%)	P(%)	R(%)	F_1(%)
video-0072	No	125	116	116	0	9	0.00	100.00	92.80	96.27
	Sí	125	123	123	0	0	1.63	100.00	98.40	99.19
video-0075	No	132	130	130	0	2	1.54	100.00	98.48	99.24
	Sí	132	133	131	2	1	0	98.50	99.24	98.87
video-0026	No	165	144	144	0	20	6.25	100.00	87.27	93.20
	Sí	165	162	162	0	0	1.85	100.00	98.18	99.08
video-0079	No	138	127	127	0	11	4.72	100.00	92.03	95.85
	Sí	138	135	135	0	3	3	100.00	97.83	98.90
video-0086	No	119	106	106	0	13	4.72	100.00	89.08	94.22
	Sí	120	122	119	3	3	0.84	97.54	99.17	98.35
Media	No						3.45	100.00	91.93	95.76
	Sí						1.98	99.19	98.06	98.62

Cuadro 4.5: Resultados de seguimiento para los mismos vídeos con el mecanismo de detección y eliminación sombras activado y desactivado

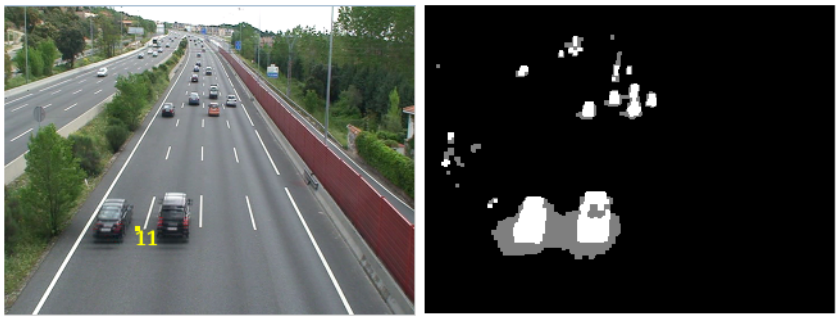


Figura 4.34: Ejemplo de fallo del sistema de detección causado por las sombras. Ambos vehículos están unidos por su sombra provocando un error de detección.

como una sola detección. Aunque se acaban separando más tarde (ya en la zona de seguimiento) sólo uno de ellos quedara registrado en el sistema. El segundo vehículo pasa desapercibido en el mejor de los casos (ya que sólo se detectan nuevos coches en la zona de entrada por las razones explicadas en la sección 4.1. En el caso contrario éste podría provocar errores de seguimiento al confundirse con el vehículo que ha sido detectado de forma correcta y que avanza en paralelo. La figura 4.34 muestra un ejemplo de este efecto.

Por último, el mecanismo de eliminación de sombras descrito en la sección 4.1.4 funciona razonablemente bien en la mayoría de los escenarios. Sin embargo, en ocasiones la eliminación de sombras falla y como consecuencia puede dar lugar a oclusiones. En estos casos el mecanismo de detección de oclusiones se activa de forma automática para marcar todos estos vehículos como ocultos y evitar que este fallo se propague a la etapa de clasificación. La figura 4.35 muestra varios ejemplos de este escenario de fallo y la respuesta del sistema.

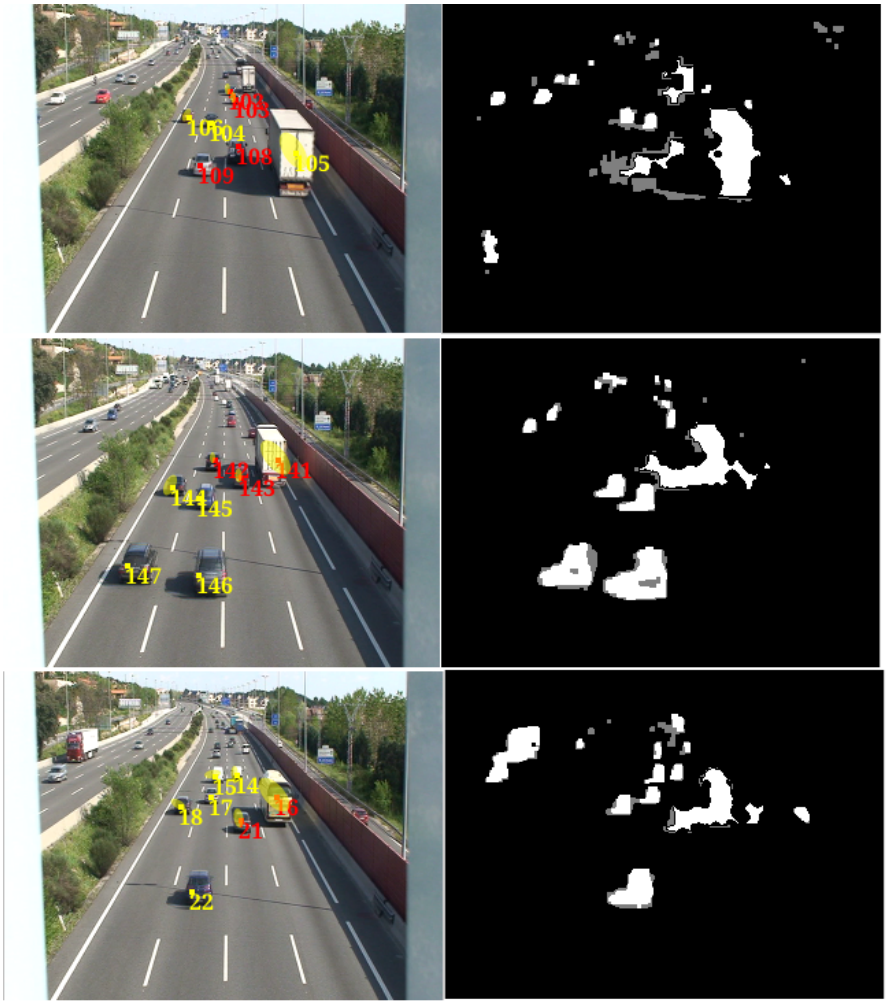


Figura 4.35: Respuesta del sistema de detección de oclusiones ante fallos en la eliminación de sombras

Figura 4.36: Ejemplos de seguimiento con distintas condiciones climatológicas

4.3.4. Seguimiento en distintas condiciones meteorológicas

Uno de los problemas más grandes de los sistemas de monitorización automática basados en visión es su robustez frente a cambios en el clima y en las condiciones de iluminación. Para comprobar el funcionamiento y la respuesta del sistema en dichas condiciones se han llevado a cabo varios experimentos con distintas condiciones meteorológicas y de iluminación. La figura 4.36 muestra varios ejemplos de seguimiento con las distintas condiciones que se han probado.

La tabla 4.6 resume la información de los resultados obtenidos para cada caso. Analizando en detalle el rendimiento del sistema vemos que los resultados obtenidos en general han sido buenos con un Valor-F que varia desde 93 % hasta alcanzar el 100 % cuando las condiciones climatológicas son favorables. Estos se dan cuando el escenario es soleado. Si analizamos los resultados obtenidos con el resto de las condiciones podemos ver que en general son buenos a excepción del video-0100. En este escenario en particular se puede ver que la métrica que más ha sufrido es la precisión. Esto se debe a los errores provenientes de la detección de fondo para los escenarios muy soleados en general. Este efecto ya lo habíamos visto en la sección 4.1.3. Cuando la detección de fondo falle en segmentar correctamente los *blobs* la etapa de seguimiento genera muchos falsos positivos dado que el mismo vehículo es contabilizado como varios.

Otro de los escenarios problemáticos es el de transición. El paso de soleado

Vídeo	Metereologia	Total	Td	Tp	Fp	Fn	Err	P(%)	R(%)	F_1(%)
video-0103	Niebla	237	236	234	2	2	0	99.15	98.73	98.94
video-0110	Lluvia	45	43	43	0	2	1	100.00	95.56	97.73
video-0028	Muy nublado	86	86	85	1	1	3	98.84	98.84	98.84
video-0029	Muy nublado	70	68	68	0	2	1	100.00	97.14	98.55
video-0075	Nublado	132	133	131	2	1	0	98.50	99.24	98.87
video-0079	Nublado	138	135	135	0	3	3	100.00	97.83	98.90
video-0005	Soleado	116	116	116	0	0	0	100.00	100.00	100.00
video-0006	Soleado	97	97	97	0	0	0	100.00	100.00	100.00
video-0099	Muy soleado	64	71	63	8	1	0	88.73	98.44	93.33
video-0100	Muy soleado	62	58	57	1	5	3	98.28	91.94	95.00
video-0040	Transición	59	60	59	1	0	10	98.33	100.00	99.16
Media								98.35	97.97	98.12

Cuadro 4.6: Resultados de seguimiento con distintas condiciones meteorológicas

a nublado y viceversa provoca falsas detecciones en el fondo que en general no suponen un problema dado que el seguimiento está preparado para detectar y filtrar *blobs* que no corresponden a vehículos. Aún así, estas falsas detecciones pueden incrementar el número de errores cometidos en el seguimiento y por lo tanto bajar la precisión global del sistema.

4.4. Conclusiones

En este capítulo se ha llevado a cabo un estudio detallado de la detección y seguimiento de vehículos incluyendo todos los algoritmos involucrados. Se ha presentado el sistema de detección y segmentación de vehículos y los distintos algoritmos que han sido utilizados para llevar a cabo esta tarea, poniendo énfasis en la eliminación de ruido y el algoritmo de eliminación de sombras. Para lo último, se ha hecho un estudio del efecto que tienen estas en general sobre la geometría del blob y en función de ello se ha elaborado un algoritmo para resolver este problema.

Se han presentado los dos algoritmos principales del seguimiento: el algoritmo basado en proximidad y el algoritmo basado en *KLT*. Además, se ha presentado una de las contribuciones principales de esta tesis que es la detección y resolución de oclusiones. Hemos visto como *KLT* es combinado con el algoritmo de proximidad para resolver este problema.

El capítulo es reforzado con una sección extensa con varios experimentos que se han llevado a cabo tanto sobre la base de datos propia (puesta a la disposición de la comunidad como parte de esta tesis) como sobre la base de datos *GRAM-RTM*. Los experimentos realizados incluyen:

- Experimentos de seguimiento bajo distintas condiciones de iluminación
- Robustez del sistema frente a oclusiones
- Capacidad del sistema para detectar y resolver las oclusiones
- Experimentos de evaluación del sistema de eliminación de sombras
- Experimentos de seguimiento bajo distintas condiciones meteorológicas

Los experimentos han demostrado que el sistema es capaz de detectar el 80 % de las oclusiones y resolver de forma correcta el 90 % de ellas. Además, analizando los resultados de seguimiento obtenidos en escenarios sin y con oclusiones para los distintos algoritmos diseñados e implementados en esta tesis hemos visto que la combinación de *KLT* con el algoritmo de proximidad da lugar a un sistema robusto frente a las oclusiones y la precisión final del seguimiento apenas se ve afectada en estos escenarios.

Hemos podido comprobar como el algoritmo de detección y eliminación de sombras introduce una mejora significativa en los resultados de seguimiento y conteo de vehículos tanto en la exhaustividad como en la precisión de los valores obtenidos.

Finalmente, Los resultados obtenidos sobre distintas condiciones de tráfico e iluminación han demostrado que el sistema es capaz de realizar el seguimiento de varios vehículos en tiempo real con una precisión muy alta que varia desde el 95 % hasta alcanzar el 99 % en condiciones favorables. El sistema también ha mostrado buen comportamiento frente a distintas condiciones meteorológicas alcanzando valores mayores que el 95 % de precisión en general.

Capítulo 5

Clasificación de vehículos

El objetivo principal de la etapa de clasificación es averiguar la categoría de cada vehículo. En el capítulo 2 hemos visto las distintas técnicas que han sido utilizadas para clasificar los vehículos en la literatura y las bases teóricas de cada una de ellas. El sistema desarrollado como parte de esta tesis es híbrido ya que combina información acerca del modelo 3D con las características visuales del vehículo para determinar su categoría. El sistema puede considerarse también como un clasificador jerárquico donde en la primera capa utilizamos el tamaño como criterio principal para distinguir entre las categorías principales mientras que en la segunda capa utilizamos la información visual para ir un poco más lejos y distinguir entre vehículos que tienen el mismo tamaño. En nuestro caso particular esto es aplicado para diferenciar entre camiones tráiler y autobuses. El diagrama 5.1 muestra los distintos pasos que conforman el algoritmo de clasificación.

5.1. Clasificación basada en modelos 3D

Antes de diseñar los algoritmos de clasificación y dado que el tamaño 3D es una de las características principales en la que se basan estos, se ha llevado a cabo un estudio para determinar las medidas de las categorías que tenemos que detectar y clasificar. Este estudio, se ha basado en los datos técnicos disponibles de varias marcas de vehículos para las distintas categorías por clasificar.

Hasta ahora hemos visto las fases iniciales de nuestro algoritmo formadas por la detección y el seguimiento. Tal y como se ha visto en el capítulo 3 la clasificación es la última etapa del sistema, recibe como entrada un conjunto de vehículos representados con su centro de masas y su cuadro envolvente. Para cada uno de ellos tenemos que estimar a qué categoría pertenece. El sistema es

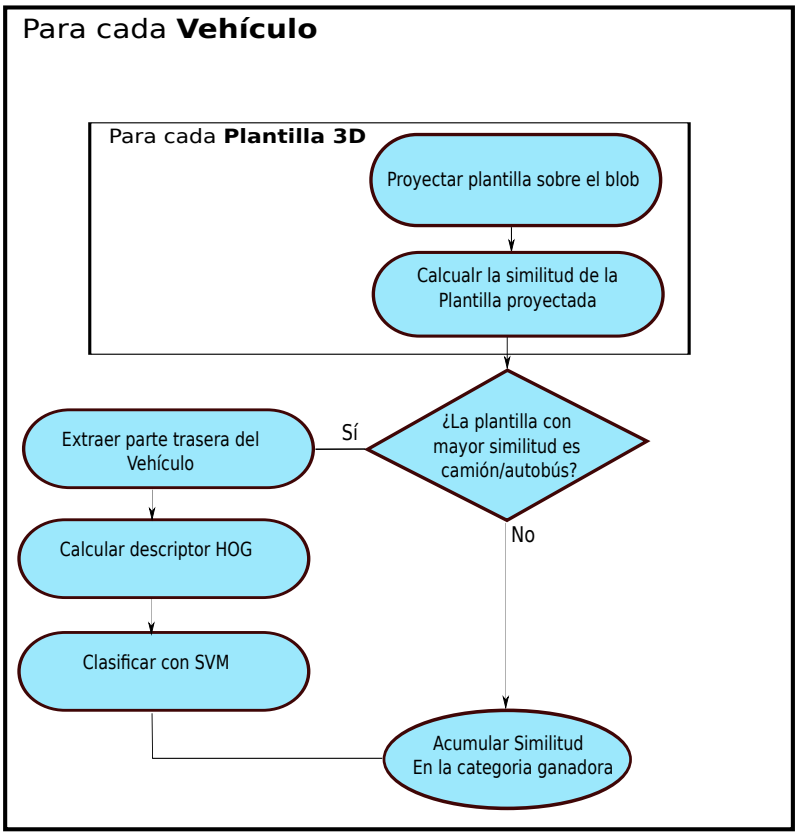


Figura 5.1: Diagrama de bloques del algoritmo de clasificación

capaz de distinguir entre cinco categorías: motocicleta, coche, furgoneta, camión y autobús. La tabla 5.1 resume la información recopilada paras las categorías soportadas.

Categoría	Alto(m)	Ancho(m)	Largo(m)
Motocicleta	1.7	1	2
Coche	1.5	1.7	4
Furgoneta	2	2	7
Camión	3.5	2.5	13.5
Autobús	3.2	2.5	12

Cuadro 5.1: Tamaño medio en metros de las categorías soportadas

La clasificación de cada vehículo consiste en los siguientes pasos:

1. Calcular el centro 3D del vehículo a partir de su centro 2D
2. Proyectar las plantillas 3D de la categorías sobre el blob
3. Calcular la similitud entre las plantillas proyectadas y el blob del vehículo
4. Si el resultado es camión/autobús hallar la categoría exacta utilizando SVM

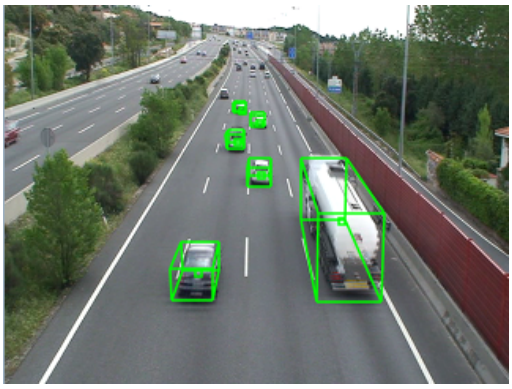


Figura 5.2: Ejemplo de varias plantillas proyectadas sobre los vehículos

Los tres primeros pasos son repetidos para cada categoría soportada por el sistema. En la figura 5.2 podemos ver un ejemplo de plantillas 3D proyectadas sobre varios vehículos.

El primer paso consiste en hallar el punto 3D correspondiente al centro 2D del vehículo. Cuando la cámara *proyecta* el mundo 3D en una imagen 2D perdemos una dimensión como efecto de esta operación, así que realizar la operación inversa con una sola cámara en principio no es posible ya que dado un punto 2D de la imagen hay un número infinito de puntos 3D (que forman una línea) cuya proyección coincide con este punto. Sin embargo, aprovechando la geometría particular de nuestra escena somos capaces de encontrar el punto exacto sabiendo que los vehículos se mueven en el plano 3D de carretera $Z = 0$. Sabemos que tanto el centro de la cámara C como el punto M^+x, dónde M^+ es la matriz pseudoinversa de la matriz de proyección M pertenecen a esta línea (R. I. Hartley and A. Zisserman [Hartley and Zisserman, 2004], página 162). La figura 5.3 muestra la línea 3D formada juntando estos dos puntos. Dado que el número de puntos que pertenecen a esta línea es infinito, ésta se puede representar con la siguiente ecuación:

$$X(\lambda) = M^+x + \lambda C \tag{5.1}$$

Con esto en mente basta con utilizar esta ecuación para obtener un punto $[A_X, A_Y, A_Z, A_T]^T$ en coordenadas homogéneas que unido a la posición 3D de la cámara determina la línea 3D en la que se encuentra el centro del vehículo. Para calcular el punto 3D exacto tenemos que hallar la intersección de esta línea con el plano $Z = \frac{h}{2}$ dónde h es la altura de la categoría cuya plantilla estamos proyectando. Para ello, conectamos el punto A con la posición de la cámara que ya habíamos fijado previamente como parte de la calibración semi-automática de la cámara. Esto da lugar al punto $m = [-A_x, -A_y, C_z - A_z]$. Finalmente el

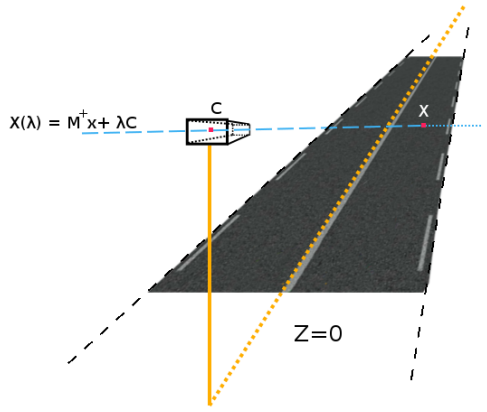


Figura 5.3: Ejemplo de re-proyección de un punto de la carretera al punto 3D correspondiente

punto 3D P es determinado calculando la intersección con el plano $Z = \frac{h}{2}$ como $P = [m_x\lambda, m_y\lambda, \frac{h}{2}]$ donde $\lambda = \frac{\frac{h}{2} - C_z}{m_z}$.

5.1.1. Función similitud de categoría

Para comparar la huella del vehículo (su blob) contra la huella producida por la proyección de cada plantilla definimos las áreas I, D y B, definidas como:

- I: es la intersección entre el blob y la proyección de la plantilla.
- D: es la diferencia simétrica entre el blob y la proyección de la plantilla.
- B: es el área del blob.

Dado que la imagen es un conjunto discreto de píxeles, estas áreas se pueden calcular como:

$$I = BLOB \cap PROJ$$
$$D = (BLOB - PROJ) \cup (PROJ - BLOB)$$
$$B = BLOB$$

Donde $PROJ$ es el área que ocupa la proyección de la plantilla 3D sobre la carretera y $BLOB$ es el área del blob. La figura 5.4 muestra un ejemplo gráfico de las áreas B, I y D (color amarillo, rojo y azul respectivamente). El área B se obtiene directamente de la información de detección de fondo. En la figura 5.4 la plantilla ha sido deliberadamente proyectada de forma incorrecta sobre el vehículo para ilustrar claramente la diferencia entre las áreas I y D.

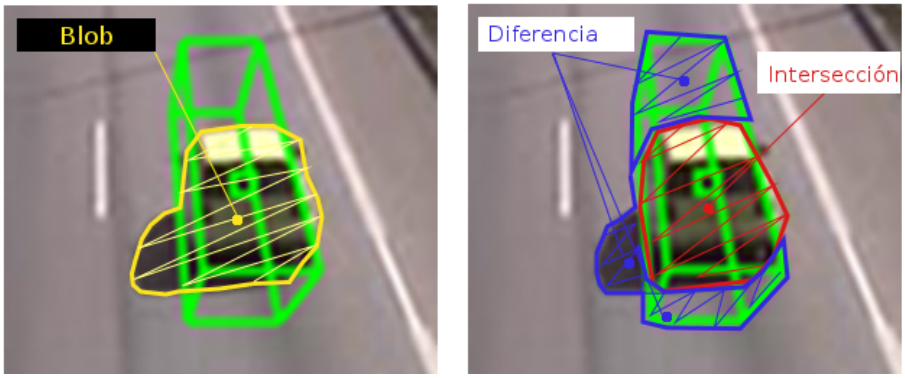


Figura 5.4: Las áreas B, I y D correspondientes respectivamente al área del blob, la intersección y diferencia simétrica con una plantilla 3D proyectada.

Todas las áreas mencionadas hasta ahora son calculadas mediante un escaneo acelerado. Un número fijo de puntos distribuido de forma uniforme es comprobado sobre el blob independientemente del tamaño de éste. Por cada uno de estos puntos se comprueba a que área pertenece. En principio se pueden dar tres casos:

- El punto pertenece sólo al blob
- El punto pertenece sólo a la proyección de la plantilla
- El punto pertenece a ambos

Para comprobar el primer caso la información del fondo es empleada con el fin de determinar si hubo movimiento o no en este punto. A la hora de realizar esta comprobación el sistema tiene en cuenta si la detección de sombras está o no habilitada. La comprobación que determina si el punto pertenece o no a la proyección de la plantilla es llevada a cabo comprobando si el punto pertenece al polígono correspondiente a la componente conexa generada por la proyección. La figura 5.5 muestra un ejemplo del polígono A, B, C, D, E, F generado y de los puntos de escaneo empleados para el mismo ejemplo.

La similitud entre un vehículo V y una categoría C_i es calculada mediante la ecuación 5.2. Dado que estos valores son calculados para todas y cada una de las categorías soportadas y para simplificar la fórmula vamos a quitar el subíndice i de las áreas I y D.

$$\alpha = \frac{I}{D+1}, \quad \beta = \frac{I}{B}$$
$$S_{V/C_i} = \beta \cdot \frac{I \cdot \alpha}{D + I \cdot \alpha} \tag{5.2}$$

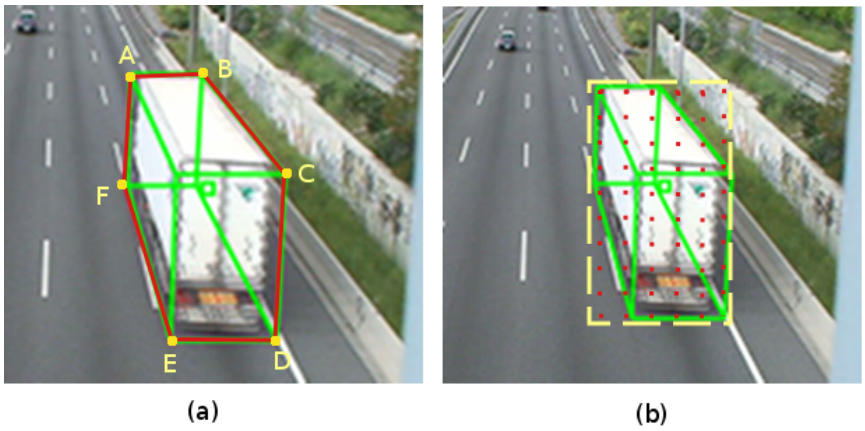


Figura 5.5: Figura (a): polígono generado por la proyección de la plantilla sobre la carretera. Figura (b) puntos de escaneo utilizados para calcular las áreas B, I y D correspondientes de la plantilla proyectada

La función de similitud varia en el rango $0-1$. Se puede interpretar como un porcentaje de similitud que varia desde $0\,\%$ hasta $100\,\%$ para las categorías cuya proyección es muy similar al blob del vehículo. La ecuación 5.2 se puede dividir en tres partes. La primera parte es el término α que es un factor cuyo objetivo principal es favorecer las plantillas cuya área de intersección I más grande que su área de diferencia D. La segunda parte es la formada por el término $\frac{I\cdot\alpha}{D+I\cdot\alpha}$. Cuando I tiende a ∞ este término tiende a 1, sin embargo cuando D es grande este término tiende a cero. Por último el término β penaliza las plantillas que no cubren toda la superficie del blob. Sin embargo, es un factor neutral para las plantillas que cubren el blob por completo ya que su valor en este caso es 1. En resumen, la ecuación 5.2 asigna mayor similitud a las plantillas que maximizan el área de intersección I y minimizan la diferencia D.

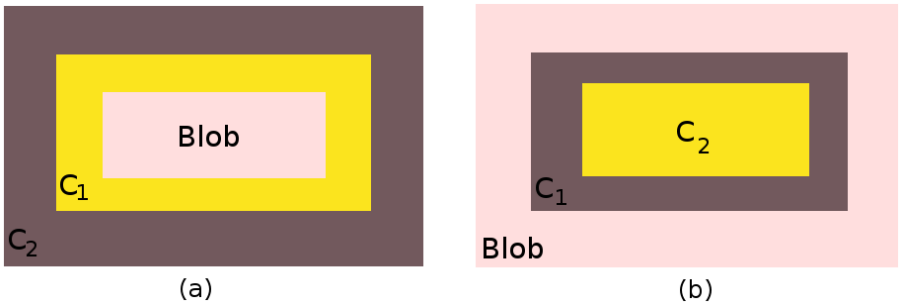


Figura 5.6: Ejemplo de distintos tamaños de plantillas y blob

De esta manera por ejemplo dadas dos plantillas C_1 y C_2 donde la primera tiene un área total mayor que la segunda y el área de proyección de ambas plantillas cubre al blob bajo análisis (figura 5.6 (a)). En este caso ambas categorías

tendrán el mismo valor de intersección, pero el valor de diferencia será mayor para la categoría C_1. Aquí interviene el factor α para hacer que la categoría C_2 tenga mayor similitud que C_1 ya que su valor de diferencia es menor.

Ahora vamos a suponer otro ejemplo de dos categorías C_1 y C_2 dónde de nuevo la primera tiene un área total mayor que la segunda pero esta vez tanto C_1 como C_2 tienen un área total menor que el del blob (figura 5.6 (b)). En este caso D es cero ya que el blob es más grande que ambas categorías y la intersección I es igual al tamaño de cada categoría. Con estos datos el primer término $\frac{I \cdot \alpha}{D + I \cdot \alpha}$ es 1 y la ecuación 5.2 se reduce al término β. En este caso el resultado final de la similitud será más alto para C_1 ya que cubre más espacio del blob que la categoría C_2.

La figura 5.7 muestra el comportamiento de la función de similitud en función de las áreas I y D para el caso de $\beta = 1$ y $\beta = 0{,}2$. Analizando el gráfico se puede ver cómo esta función de forma gradual asigna más similitud (zona amarilla) cuando el valor de I es superior al de D. Para los casos cuando I tiene un valor similar a D la función asigna un valor objetivo de similitud entorno a 0,5 (zona diagonal del gráfico).

Ahora vamos a analizar la misma gráfica para el caso donde la proyección de la plantilla no cubre el área del blob ($\beta = 0{,}2$). En este caso se puede ver cómo el valor máximo alcanzado de similitud es igual al término β. Este término además de penalizar las plantillas que no cubren el blob establece un límite máximo que puede alcanzar la similitud generada por esta plantilla. El límite establecido es proporcional a la relación entre el área del blob y el área de la intersección con la plantilla.

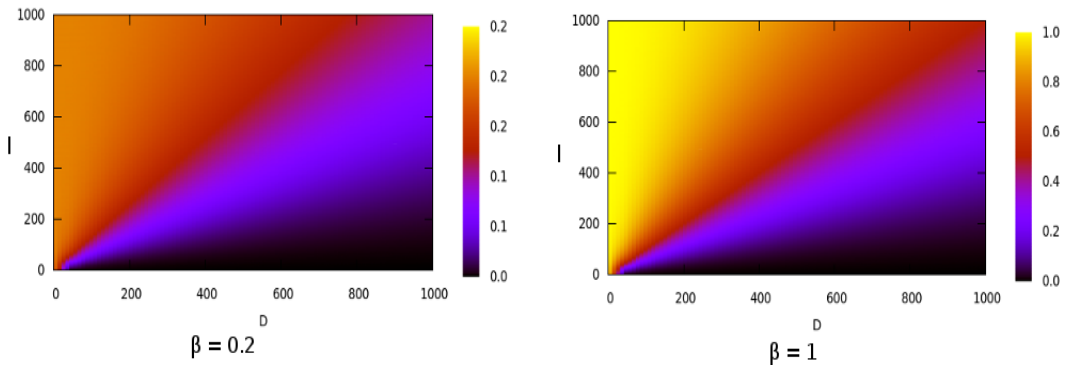


Figura 5.7: Análisis de la función de cálculo de similitud para dos valores de β

Aún cuando en general la mayoría de los vehículos entran en la clasificación que hemos establecido, en la práctica es común encontrarse con vehículos cuyo tamaño cae en una categoría intermedia en cuanto a tamaño. El caso más claro

es de los coches donde podemos encontrar desde coches muy pequeños hasta berlinas largas. Lo mismo pasa con el resto de las categorías. De ahí que es importante comprobar que la función de similitud se comporta de forma correcta para vehículos que pertenecen a la misma categoría pero con pequeñas diferencias de tamaño y forma. Esto significa que el valor de similitud asignado para vehículos con un tamaño parecido tiene que ser similar al valor asignado a la categoría más cercana.

Para ello, se ha llevado a cabo un experimento que consiste en simular vehículos con tamaños intermedios entre las categorías principales y comprobar el valor asignado por la función de similitud para cada uno de ellos. La tabla 5.2 muestra el tamaño de cada una de estas categorías. La simulación se ha llevado a cabo mediante *Gazebo*[1] creando un simulación sencilla de una carretera y un vehículo (figura 5.8). En cada iteración del experimento se fija la categoría de referencia (moto, coche, etc) cuya plantilla se va utilizar para calcular la similitud con la función 5.2 y se simula un vehículo para cada una de las categorías intermedias. La similitud de éste respecto a la categoría de referencia es calculada. La figura 5.9 muestra el comportamiento de la función de similitud para cada una de las categorías intermedias. El eje X representa las diferentes categorías mientras que Y es el valor de similitud respecto a la categoría de referencia establecida.

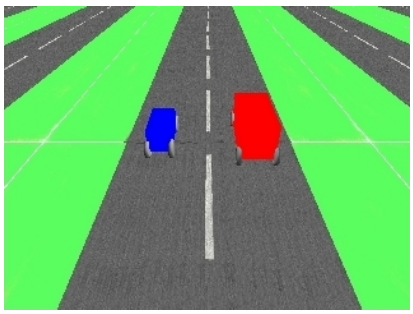


Figura 5.8: Simulación de carretera y vehículos creada con Gazebo

Idealmente, el sistema tiene que clasificar cada categoría intermedia como la categoría soportada cuyo tamaño es más cercano. Dado un coche pequeño por ejemplo nuestro objetivo es clasificarlo como un coche y no como una motocicleta. Analizando las distintas gráficas para las categorías principales en la figura 5.9 vemos que la función de similitud cumple con este cometido. Por ejemplo, cuando la categoría de referencia es coche (curva verde) la similitud de la categoría $P3$ con la categoría coche es de $\approx 0{,}9$ mientras que su similitud con una motocicleta

[1] http://gazebosim.org/

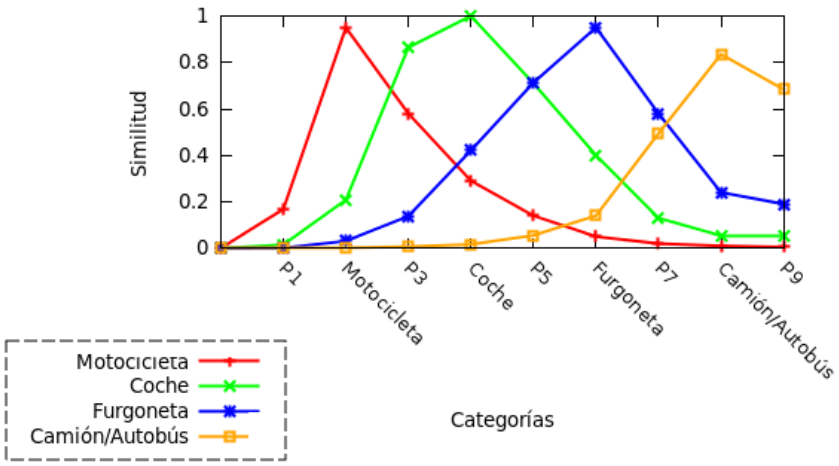


Figura 5.9: Análisis del comportamiento de la función de cálculo de similitud para las categorías intermedias. Cada curva representa los valores de similitud obtenidos para todas las categorías cuando fijamos la categoría de referencia a la clase representada por la curva

Categoría	Alto(m)	Ancho(m)	Largo(m)
P1	1	0.5	1
P2(Motocicleta)	1.7	1	2
P3	1.6	1.35	3
P4(Coche)	1.5	1.6	4
P5	1.75	1.85	5.5
P6(Furgoneta)	2	2	7
P7	2.75	2.25	10.25
P8(Camión)	4	2.5	14
P9	4	2.5	16

Cuadro 5.2: Tamaño de las categorías soportadas y las categorías intermedias

es de 0,5. Se trata de un buen resultado dado que esta categoría en concreto se acerca más a la categoría de coches en términos de tamaño.

5.1.2. Acumulación de similitud

La robustez es uno de los objetivos más importantes a la hora de diseñar el sistema presentado en esta tesis. Ya hemos visto que la etapa de seguimiento ha sido preparada especialmente para tratar casos complicados como oclusiones parciales o totales. La etapa de clasificación es la última fase del procesamiento y los algoritmos diseñados para la misma tienen que ser fiables y capaces de manejar situaciones de error que se pueden dar en la vida real. Esto incluye ruido, errores en la clasificación, oclusiones, etc. En un mundo ideal (sin oclusiones, ruido ni errores), el vehículo es asignado a la misma categoría durante todas las rondas

de clasificación. Al final de su trayectoria, la categoría que le representa será la clase que más similitud haya acumulado. El valor de similitud para el resto de las categorías debería ser cero. Lamentablemente el escenario real que manejamos está lejos de ser este mundo ideal. En la práctica hay varias fuentes de ruido y errores: imperfecciones en la detección de fondo, ruido en la escena, vibración de la cámara, oclusiones, etc. Todos estos factores pueden afectar a los resultados de clasificación si no se tienen en cuenta a la hora de diseñar los algoritmos.

Para aumentar la fiabilidad de los resultados de clasificación frente a los errores puntuales en la clasificación de algún vehículo éstos son acumulados durante toda la zona de seguimiento (dónde en realidad empieza la clasificación). Cada iteración se considera una ronda de clasificación independiente del resto, en la cual, todos los vehículos son clasificados de nuevo y la categoría ganadora para cada vehículo acumula el valor de similitud para la clase que representa. Al final del seguimiento, el vehículo es clasificado como la categoría que más similitud haya acumulado. Este proceso le da a la clasificación mucha estabilidad dado que cualquier tipo de error puntual afecta a la ronda de clasificación actual pero no al resultado final acumulado. Para evitar que las oclusiones afecten a la clasificación todos los vehículos que hayan sido marcados en la etapa de seguimiento como ocultos no se clasifican para evitar riesgos de clasificarlos incorrectamente. Mientras permanezcan ocultos, los resultados acumulados de clasificación se mantienen igual para todos los vehículos involucrados hasta que esta situación se resuelva o alguno de los vehículos entra en oclusión total. En este último caso el sistema finaliza el seguimiento y estima la categoría final del vehículo utilizando los datos acumulados hasta ese momento.

5.2. Clasificación basada en HOG/SVM

Las categorías de camión y autobús son tratadas de forma especial dado que en este caso el tamaño 3D es insuficiente para diferenciar entre ambas. Cada vez que se detecta que un vehículo podría pertenecer a una de estas categorías lanzamos una segunda etapa de clasificación esta vez basándose en las características visuales del vehículo para determinar si es un camión o un autobús. La idea básica de esta etapa es capturar el trasero del vehículo dado que es la parte que menos efectos de proyección sufre teniendo en cuenta la posición de la cámara descrita en la sección 3.2. La imagen capturada es procesada mediante un clasificador basado en SVM y HOG para determinar finalmente la categoría del vehículo. La figura 5.10 muestra un diagrama simplificado de las distintas etapas del algoritmo.

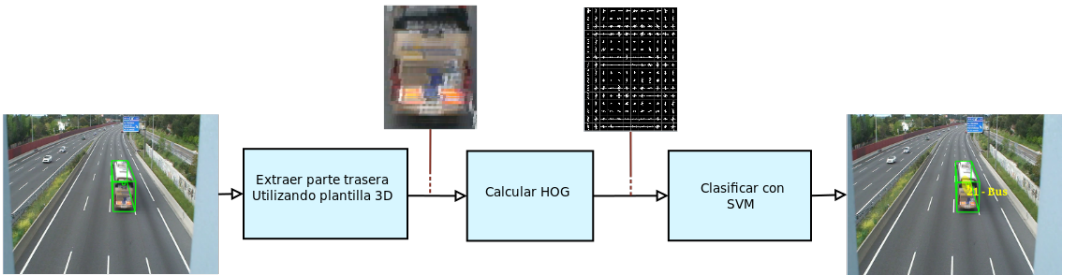


Figura 5.10: Proceso de clasificación de las categorías camión/autobús

Las máquinas de vectores soporte (SVM por sus siglas en inglés de *Support Vector Machine*) son un clasificador lineal capaz en su versión básica de clasificar las muestras en dos clases. Fueron introducidos en los años 90 como parte del trabajo de [C. Cortes and V. Vapnik, 1995]. Aunque sus inicios se remontan a los 60, no ha sido hasta la época de los 90 donde empezó a ganar en popularidad gracias a la introducción del *truco del kernel* por el trabajo de [Boser et al., 1992]. Esta modificación abrió el abanico al uso de SVM con éxito en multitud de problemas no lineales.

En el caso de dos dimensiones la solución es un hiperplano que separa las muestras en dos subconjuntos. En general se puede distinguir entre dos técnicas para llevar a cabo esta tarea:

1. Modelos generativos

2. Modelos discriminatorios

La primera de ellas trata de estimar la función de densidad de probabilidad que genera la distribución de cada clase. La segunda familia de técnicas trata de clasificar las muestras sin tener que generar la densidad de probabilidad de las mismas. En cambio, éstas basan su análisis en un conjunto de muestras para hallar la frontera de separación entre las dos clases a partir del conjunto de muestras utilizadas para el entrenamiento. Dicho esto, es de vital importancia intentar que las muestras escogidas sean lo suficientemente representativas de las dos clases. SVM pertenece a esta categoría de clasificadores.

SVM es un clasificador lineal que intenta maximizar el margen entre dos clases basándose en un número determinado de muestras que se denominan vectores soporte. Dado un conjunto de muestras de entrada SVM intenta hallar el hiperplano que lo divide en dos regiones disjuntas maximizando el área libre de muestras entre ellas. En el caso particular de dos dimensiones la solución es una línea recta. Como se puede ver en la figura 5.11, el conjunto de líneas que separan ambos

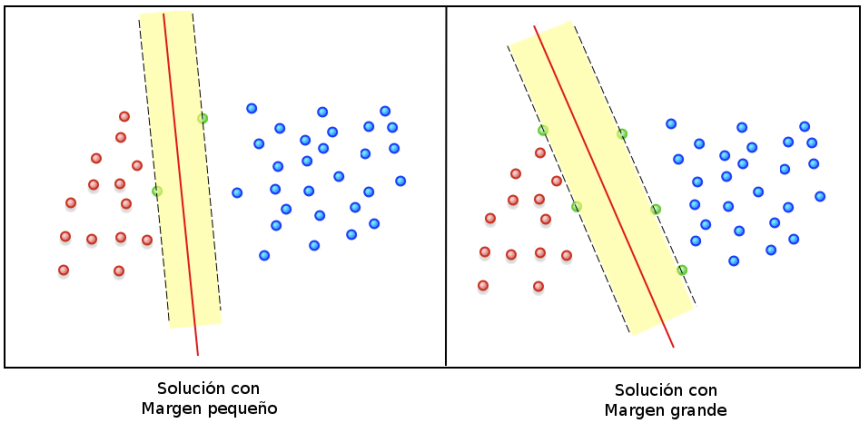


Figura 5.11: Dos soluciones distintas para el mismo conjunto de datos. En la solución de la izquierda los vectores soporte elegidos (color verde) dan lugar a un margen más estrecho que la solución de la derecha.

conjuntos es infinito, sin embargo, el objetivo es encontrar la línea que separa las dos clases generando la mayor separación entre ambas. En este ejemplo, el margen generado por la segunda solución es mucho mayor que el generado por la primera. Dado que la solución encontrada por SVM es la óptima, cualquier otra solución nos dará un margen más estrecho y por lo tanto con menos seguridad a la hora de clasificar muestras adyacentes a la frontera entre las dos clases.

En general, las muestras no siempre son fácilmente separables como en el ejemplo de la figura 5.11. En la práctica lo normal es encontrarse con problemas de clasificación donde el espacio de muestras puede tener muchas dimensiones y donde las muestras no son linealmente separables. SVM resuelve este problema mediante el *truco del kernel*, donde en vez de utilizar el espacio original de las muestras, éstas son transformadas a otro espacio con más dimensiones que el espacio original con la esperanza de que en este nuevo espacio estas sean separables. La formulación final de la clasificación queda determinada por el *signo* de la siguiente función:

$$f(x) = sgn(\sum_i y_i \alpha_i K(x_i, x_j) + b) \tag{5.3}$$

Donde $K(x_i, x_j)$ es la función *kernel* elegida para mapear las características. Adicionalmente, durante la fase de entrenamiento y dependiendo del kernel elegido SVM suele requerir un ajuste de un conjunto de parámetros para optimizar el modelo generado. El apéndice C contiene un análisis matemático detallado para llegar a la ecuación 5.3 así como un análisis detallado de los parámetros que in-

tervienen en la optimización del modelo y el efecto de estos sobre los resultados de la clasificación.

5.2.1. Histogramas de Gradientes Orientados

El descriptor HOG (por su siglas en inglés de: *Histogram of oriented gradients*) es uno entre los más famosos y exitosos descriptores basados en gradientes. En general, el objetivo principal de un descriptor es *resumir* la información contenida en una una imagen como formas y/o texturas de modo que sea invariante a cambios de iluminación, aspecto y posición entre otros parámetros de la escena. Este último concepto es muy importante dado que nos permitirá reconocer el objeto modelado en distintas imágenes, con distintos niveles de intensidad, fondo, etc. Finalmente, la información resumida se puede utilizar como entrada a un clasificador. De esta forma además de reconocer los objetos en la imagen seremos capaces de clasificarlos en distintas categorías.

El descriptor HOG ha sido originalmente introducido por [Dalal N. and Triggs B, 2005] con el objetivo de detectar personas en una imagen. Según los autores originales, el descriptor es capaz de clasificar a la perfección todas las muestras de la base de datos del MIT con personas en distintas configuraciones [Papageorgiou and Poggio, 2000]. Para evaluar la capacidad real del descriptor los autores tuvieron que introducir una nueva base de datos con más imágenes mucho más complejas que las ofrecidas por la base datos del MIT. La idea principal detrás de HOG es resumir la información de la forma del objeto recogido en la imagen. Para ello, tanto el cálculo del descriptor como la información codificada por el mismo se basan en el cálculo de gradientes en la imagen. Estos básicamente nos indican la magnitud y la orientación del cambio en la intensidad. Dado un píxel $p(x, y)$ en una imagen I, la derivada en el píxel p en la dirección x se puede calcular con la siguiente fórmula:

$$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x) \tag{5.4}$$

En general dada una función $f(x, y)$, su gradiente básicamente está formado por las derivadas parciales en cada dirección. Éste se puede calcular con la siguiente fórmula:

$$\nabla f(x, y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix} \tag{5.5}$$

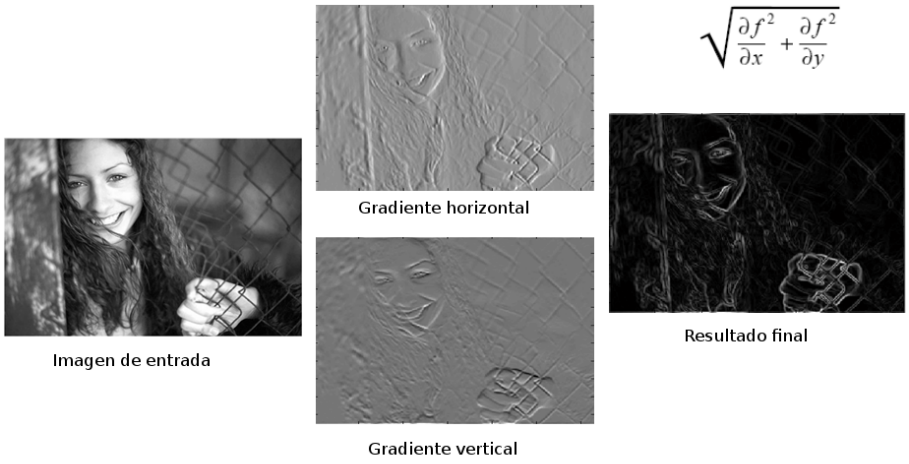


Figura 5.12: Detección de bordes utilizando la información del gradiente

Mientras que su magnitud y orientación son calculados con las fórmulas:

$$\left|\nabla f(x,y)\right| = \sqrt{f_x^2 + f_y^2} \tag{5.6}$$

$$\theta = tan^{-1}\frac{f_x}{f_y} \tag{5.7}$$

Al calcular el gradiente sobre la imagen de entrada tanto en la dirección horizontal como en la vertical obtenemos como resultado todos los bordes contenidos en la misma. Este efecto proviene del hecho de que el gradiente señala las zonas dónde se han producido grandes cambios en la intensidad de los píxeles. Naturalmente estas suelen corresponder con bordes del objeto siempre que éste tenga un color distinto al del fondo de la escena. En la figura 5.12 podemos ver un ejemplo de la imagen resultante del cálculo del gradiente en la dirección x, y y el resultado final.

La información que hemos obtenido hasta ahora se puede representar con una matriz del mismo tamaño que la imagen cuyas entradas son el valor del gradiente. Esto nos sirve para extraer la forma del objeto representado en la imagen, sin embargo, esta representación tiene todavía dos problemas:

1. Problema 1: La cantidad de información por guardar sigue siendo grande ya que tenemos un vector de gradiente por cada píxel

2. Problema 2: Es sensible a cambios en la posición del objeto

3. Problema 3: Es sensible a cambios en la iluminación y contraste de la imagen

Para resolver el primer problema HOG resume la información de la forma del objeto en la escena guardando un *resumen* de la misma. Este básicamente está

formado por una serie de histogramas que recogen tanto la magnitud como la orientación de todos los gradientes. Para ello:

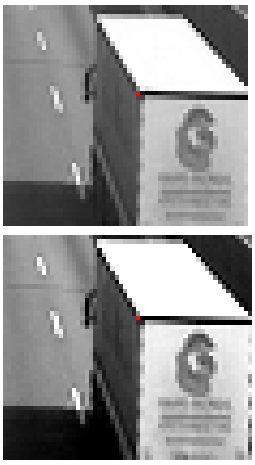
- La imagen se divide en varias celdas cada una de $b \times b$ píxeles.
- El intervalo de los valores para la orientación del gradiente es dividido en pequeños rangos de m grados.

Para cada celda se calcula un histograma de orientaciones. Cada entrada del histograma representa un pequeño rango y su valor es la suma de todos los píxeles cuya orientación cae en este rango. La contribución de cada píxel al histograma es proporcional a la magnitud de su gradiente. De esta manera, los gradientes más fuertes tendrán más impacto en la distribución final del histograma. En realidad, la aportación del vector gradiente es dividida entre los dos rangos más cercanos. Por ejemplo, si el rango de valores está dividido en pequeños rangos cada uno de 20 grados. En este caso un gradiente con un valor de 35 grados aporta $\frac{3}{4}$ de su magnitud al rango centrado en 40 y $\frac{1}{4}$ al rango centrado en 20. Esta interpolación permite distribuir las orientaciones dentro del histograma de forma más robusta a pequeños cambios en la posición del objeto.

Hasta ahora hemos visto cómo ha resuelto HOG el problema 1 y el problema 2, nos queda por ver cómo trata este descriptor el tercer problema relacionado con los cambios de iluminación y contraste en la imagen. Antes de ver la técnica que se emplea para resolverlos vamos a ver el impacto que tiene un cambio en la intensidad en toda la imagen sobre los histogramas generados por el descriptor.

En general, podemos demostrar fácilmente que cambios que consisten en un aumento (o disminución) global en la intensidad de toda la imagen no afectan a los valores del gradiente dado que con la fórmula 5.4 estos términos se acabarán anulando en la operación de la resta. Sin embargo, cualquier cambio en el contraste de la imagen afectará al valor final del gradiente y como consecuencia afectará a la forma del histograma resultante de la celda. En la figura 5.13 podemos ver cómo afecta un cambio en el contraste al valor final del gradiente del un píxel (punto rojo). Dado que uno de los objetivos de HOG es obtener una representación invariante respecto a cambios en la iluminación tenemos que realizar un paso adicional para eliminar este problema. Este paso es la normalización.

La normalización de un vector consiste en dividir los valores de sus componentes entre su magnitud. En [Dalal N. and Triggs B, 2005] los autores han experimentado con un conjunto de técnicas de normalización como *L1-norm*, $v \rightarrow v/(\|\mathbf{v}\|_1 + \epsilon)$ o *L2-norm*, $v \rightarrow v/\sqrt{(\|\mathbf{v}\|_2^2 1 + \epsilon^2)}$. El estudio concluye que



$$\nabla f = \begin{bmatrix} 66 \\ 63 \end{bmatrix}$$

$$|\nabla f| = \sqrt{(66)^2 + (63)^2} = 91{,}24$$

$$\nabla f = \begin{bmatrix} 96 \\ 73 \end{bmatrix}$$

$$|\nabla f| = \sqrt{(96)^2 + (73)^2} = 120{,}6$$

Figura 5.13: Efecto del cambio de contraste sobre el valor del gradiente

todas las normas probadas dan resultados satisfactorios así que la única diferencia a tener en cuenta en la práctica es su impacto en el rendimiento global del algoritmo. HOG parte de la idea de que los cambios de contraste no suelen ser globales para toda la imagen sino más bien locales a ciertas áreas de la misma. Con esto en mente, para llevar a cabo la normalización el descriptor agrupa las celdas en pequeños bloques formados por $m \times m$ celdas. El conjunto de valores de todos los histogramas que pertenecen a un bloque son concatenados para formar un único vector. Finalmente, los valores de éste son normalizados respecto a la norma elegida. Los bloques se definen de tal forma que haya solapamiento entre ellos, en la configuración típica de HOG los bloques están formados por 2×2 celdas. En este caso en concreto hay un solapamiento de 2 celdas entre dos bloques adyacentes tal y como se muestra en la figura 5.14.

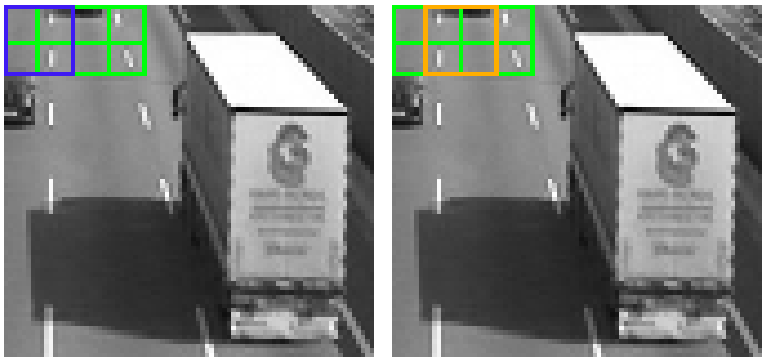


Figura 5.14: Solapamiento entre bloques en la configuración 2×2 de HOG

La normalización se lleva a cabo para cada bloque. Dado el solapamiento entre éstos, una misma celda puede participar en el resultado final tantas veces como bloques a los que pertenece. El valor del histograma de la celda cambiará

Figura 5.15: Distribución de los coeficientes de aportación de cada celda en el descriptor final. Las celdas interiores participan en cuatro bloques distintos.

dependiendo del bloque sobre el cual se ha llevado a cabo la normalización. Este hecho otorga más robustez al valor final del descriptor, puesto que además de normalizar el valor de los histogramas nos permite recoger la disposición espacial de las celdas. La figura 5.15 muestra un ejemplo de una celda que pertenece a varios bloques y la matriz de contribuciones para todas las celdas.

El descriptor HOG soporta dos configuraciones distintas con el fin de adaptarse a la geometría de los objetos del dominio de la aplicación. La primera configuración es rectangular (*R-HOG*) y la segunda es la denominada *C-HOG* (circular HOG). En este caso la forma de las celdas están participadas en forma polar. La figura 5.16 muestra ambas configuraciones del descriptor. En esta tesis hemos utilizado la configuración *R-HOG* dado que es la que mejor se adapta a la geometría de los vehículos.

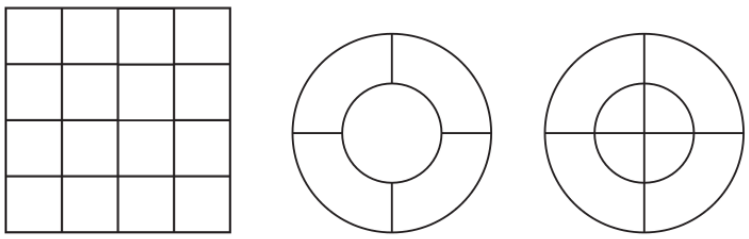


Figura 5.16: Configuraciones geométricas del descriptor HOG: **(a)** configuración rectangular (R-HOG), **(b)** configuración circular (C-HOG)

5.2.2. La parte trasera de camiones y autobuses

Las categorías de camión y autobús representan un caso especial de clasificación dado que sólo por tamaño 3D somos incapaces de distinguir entre ellas.

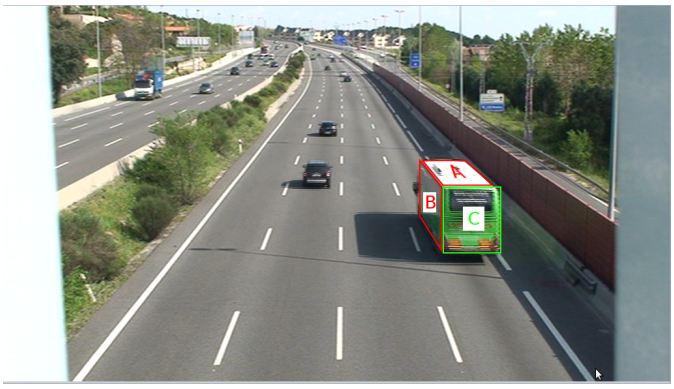


Figura 5.17: Imagen de la vista de un autobús con la configuración empleada de la cámara. La parte lateral (B) del vehículo sufre mucha deformación por el efecto de la proyección, además de estar afectada por las sombras. La parte superior (A) no tiene información visual suficiente para diferenciar entre camiones y autobuses. La parte (C) no sufre ningún efecto además contiene información suficiente para diferenciar entre ambas categorías.

La solución planteada en esta tesis para resolver este problema se basa en una segunda etapa de clasificación esta vez basada en característica visuales para diferenciar entre ambas categorías. Si analizamos el aspecto de ambas categorías (ver figura 5.18) veremos que visualmente tienen muchas diferencias entre las cuales podemos encontrar:

1. El color: los camiones tráiler tienden a tener un color blanco mientras que los autobuses tienen colores variados

2. Las ventanas: los autobuses disponen de ventanas tanto en la parte trasera como en los laterales mientras que los camiones en general no tienen ventanas

3. Los camiones disponen de una cabecera y un tráiler mientras que los autobuses están compuestos por una única cabina

4. La parte trasera: los camiones disponen de puertas en la parte trasera mientras que los autobuses en general tienen una gran ventana

Analizando estas diferencias vemos que el color es una buena característica visual que se puede emplear en general, pero en este caso particular no es totalmente discriminatoria dado que puede haber camiones de otro color que no sea blanco o autobuses de este color. Otra desventaja que tiene el color como característica es su alta sensibilidad a cambios en la iluminación. Las características número 2 y 3 a priori son buenas para diferenciar entre un camión y un autobús

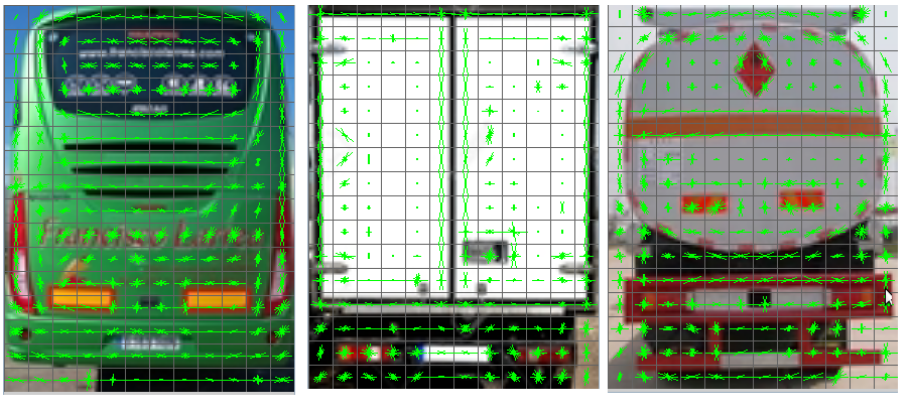


Figura 5.18: Histogramas HOG superpuestos sobre las imágenes de las categorías soportadas

visualmente, sin embargo dada la configuración y el ángulo de nuestra cámara la primera de ellas no se puede utilizar ya que estamos procesando tráfico en sentido saliente mientras que la segunda sufre mucho el efecto de la proyección tal y como se puede ver en la figura 5.17. Por último, nos queda la parte trasera. Analizando en detalle esta parte podemos ver que tiene las siguientes características:

- Es una característica que se puede utilizar para diferenciar entre camiones y autobuses: la mayoría de los camiones tráiler tienen puertas traseras mientras que los autobuses disponen de una ventana trasera.
- Dada nuestra configuración de la cámara esta parte es perfectamente visible y no sufre en exceso el efecto de la proyección.
- Es fácilmente extraíble ya que pertenece al mismo plano 3D.

Dadas las ventajas que presenta esta parte física hemos optado para utilizarla como la base para distinguir entre las dos categorías. La idea principal es extraer la parte trasera del vehículo aprovechando la información 3D de las plantillas, generar su descriptor HOG y clasificarlo mediante un SVM. Esta idea es similar al trabajo de [Buch et al., 2009] con la diferencia de que en este caso se crea un histograma 3DHOG para todas las plantillas mientras que nuestro trabajo tan sólo utilizamos la información 3D de la plantilla para extraer la parte trasera del vehículo (camión o autobús en nuestro caso). La figura 5.18 muestra algunos ejemplos de descriptores HOG calculados para las categorías soportadas.

Para llevar a cabo el primer paso, además de recortar la parte posterior tenemos que rectificar el plano que la contiene para deshacer el efecto de la proyección. La rectificación de planos es una técnica clásica en visión computacional

cuyo objetivo es deshacer el efecto que sufren los planos a causa de la proyección y recuperar el plano original. En términos geométricos, una cámara proyecta figuras del mundo 3D en un plano 2D (la imagen) manteniendo su propiedades de proyección invariantes. Aprovechando el modelo *pinhole* descrito en el apéndice A somos capaces de deshacer las transformaciones que sufre el plano y recuperar su aspecto original. De esta manera recuperamos propiedades geométricas importantes sobre el plano original como: ángulos, distancias, paralelismo, etc.

El problema de la rectificación ha sido ampliamente tratado en la literatura. Hay varias maneras de abordarlo, [Liebowitz and Zisserman, 1998] descompone la homografía como una combinación de tres transformaciones distintas:

$$H = H_s H_a H_p$$

Donde H_p es la matriz que contiene la información de proyección, H_a la transformación afín y por último H_s contiene la información de similitud. El primer paso para calcular la matriz H_p es obtener la línea en el infinito que contiene los puntos de fuga. Esta se puede obtener a partir de dos puntos de fuga p_1 y p_2 con la siguiente fórmula:

$$l_\infty = (l_1, l_2, l_3)^T = p_1 \times p_2$$

Una vez calculadas las coordenadas de l_∞ podemos construir la matriz H_p de forma directa:

$$H_P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} \tag{5.8}$$

Las matrices H_a y H_s se pueden obtener con las siguientes fórmulas:

$$H_a = \begin{bmatrix} \frac{1}{\beta} & -\frac{\alpha}{\beta} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.9}$$

$$H_s = \begin{bmatrix} sR & t \\ 0^T & 1 \end{bmatrix} \tag{5.10}$$

([Liebowitz and Zisserman, 1998]) contiene los detalles del cálculo de los coeficientes α y β. La matriz H_s se puede construir mediante las matrices de rotación R, escalado s y traslación t. Finalmente, la matriz H se obtiene multiplicando las tres matrices: H_p, H_a y H_s.

Otra técnica para rectificar el plano es la basada en puntos de correspondencia. Este método se basa en aprovechar conocimiento geométrico previo de la

escena para deshacer los efectos de la proyección. Por ejemplo si conocemos la geometría de algún objeto de la escena podemos aprovecharla para establecer una correspondencia entre puntos del espacio 3D con puntos de la imagen 2D. Según el modelo pinhole, la ecuación general que relaciona un punto P_w cualquiera con un píxel P_{im} de la imagen se puede escribir como:

$$P_{im} = MP_w \tag{5.11}$$

Donde $M_{3\times4}$ es la matriz genérica de la proyección. Añadiendo la restricción adoptada en el proceso de rectificación que consiste en que los puntos que forman la correspondencia tienen que pertenecer al mismo plano, la ecuación anterior se puede reescribir como:

$$P_{im} = HP_w \qquad P_w \in \pi_w \tag{5.12}$$

Donde π_w es el plano que queremos reconstruir y $H_{3\times3}$ una matriz homogénea de ocho grados de libertad (el noveno es la escala). Para deshacer la transformación basta con calcular la matriz H y aplicar la transformación inversa:

$$P_w = H^{-1}P_{im} \tag{5.13}$$

La relación 5.12 de proyección se puede representar con una matriz homogénea H de nueve elementos:

$$\begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} \tag{5.14}$$

Dado que H es homogénea, kH también es una solución válida para el sistema anterior. Así que podemos dividir todos los elementos de H por *h33* y pasamos a tener ocho incógnitas en vez de nueve. Para hallar los ocho elementos necesitamos ocho ecuaciones linealmente independientes. Si desarrollamos el sistema anterior y dividimos por la coordenada homogénea obtenemos lo siguiente:

$$x' = \frac{x'_1}{x'_3} = \frac{h_{11}.x + h_{12}.y + h_{13}}{h_{31}.x + h_{32}.y + 1} \tag{5.15}$$

$$y' = \frac{x'_2}{x'_3} = \frac{h_{21}.x + h_{22}.y + h_{23}}{h_{31}.x + h_{32}.y + 1} \tag{5.16}$$

Desarrollando este sistema llegamos a:

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} \end{bmatrix}^T = \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{5.17}$$

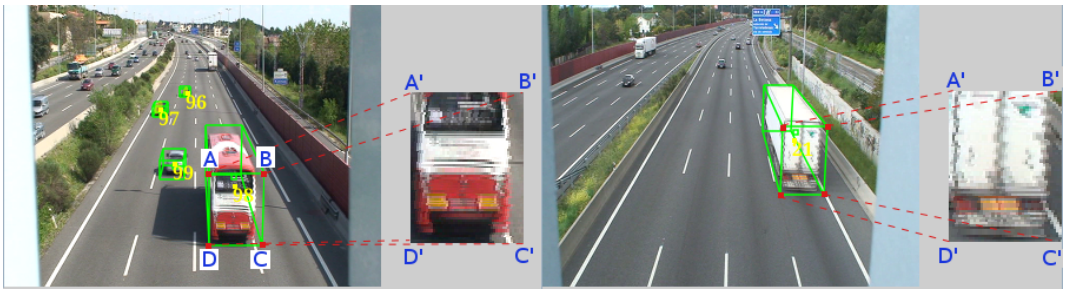


Figura 5.19: Extracción de la parte trasera del camión/autobús. La proyección de la plantilla 3D del vehículo sobre el blob es utilizada para extraer de forma precisa su parte trasera mediante rectificación de planos.

En este sistema de ecuaciones las incógnitas son las h_{ij}. Dada la correspondencia entre dos puntos (x, y) y (x', y') que pertenecen a la imagen 2D de entrada y de salida respectivamente somos capaces de obtener dos ecuaciones. Para resolver las ocho incógnitas hacen falta cuatro puntos de correspondencia entre el plano original π_{im} y el plano rectificado π_w. Para obtener ocho ecuaciones linealmente independientes, los cuatro puntos tienen que ser coplanares pero no colineales. El sistema de ecuaciones construido en este caso es compatible determinado y tiene una solución única.

Con la técnica de rectificación somos capaces de extraer la parte trasera de forma bastante precisa cada vez que la clasificación por tamaño determina que el vehículo es un autobús o un camión. Los puntos elegidos para establecer la correspondencia son los vértices A, B, C y D del polígono trasero de la plantilla proyectada sobre el vehículo. Estos son mapeados a las esquinas A', B', C' y D' de una imagen de tamaño fijo de 96×128 píxeles (figura 5.19). Cabe destacar que esta transformación conlleva una operación de escalado que se hace manifiesta a medida que el vehículo se va alejando de la cámara. Esto es necesario dado que HOG necesita una imagen de tamaño fijo. La clasificación basada en SVM tiene que seguir funcionando dado que la distribución de bordes generada por el descriptor HOG debería ser la misma después de aplicar la operación de escalado.

5.2.3. Parametrización del clasificador SVM

Hasta ahora hemos visto las bases del clasificador que se va construir para clasificar los grandes vehículos en autobuses y camiones, donde en esta segunda categoría a su vez tenemos que diferenciar entre camiones normales y camiones cisterna. En esta sección veremos en detalle el proceso de selección y ajuste de sus parámetros. Para llevar a cabo estas pruebas se han fijado los parámetros

Figura 5.20: Ejemplos de muestras de entrenamiento de autobuses, camiones y camiones cisterna

del descriptor HOG en la configuración óptima que discutiremos más en detalle en la sección 5.2.4. Para validar el modelo generado se ha creado una base de datos formada por 252 ejemplos de la parte trasera para las categorías: autobús, camión y camión cisterna que tenemos que clasificar (ochenta y cuatro ejemplos para cada categoría). Estas son empleadas para llevar a cabo un entrenamiento supervisado del clasificador *SVM*. En la figura 5.20 se puede ver varios ejemplos de traseros recortados de camiones y autobuses que han sido utilizados para el entrenamiento.

El clasificador es entrenado mediante validación cruzada (*cross validation* en inglés). Esta técnica estadística permite obtener modelos más fiables y robustos. Para ello, en vez de dividir las muestras de entrenamiento en dos partes, una para el entrenamiento y otra para la validación como hacen la mayoría de métodos tradicionales, esta técnica parte el conjunto de muestras de entrada en varios subconjuntos. No hay un número estándar de estos pero hay que hacer la partición de tal forma que el número de elementos en un solo subconjunto sea suficiente para poder validar el modelo generado sobre el resto de subconjuntos. Dentro de la validación cruzada hay varios algoritmos para validar el modelo final:

1. Validación cruzada de K iteraciones
2. Validación cruzada aleatoria
3. Validación cruzada dejando uno fuera

En la validación cruzada de K iteraciones los datos se dividen en K subconjuntos, de los cuales $K-1$ son utilizados para entrenar el modelo y el conjunto restante es empleado para validar el modelo obtenido. El proceso es repetido K iteraciones, en cada iteración se escoge un conjunto de validación distinto hasta cubrir los K subconjuntos generados inicialmente. El resultado final es calculado como la media aritmética de todas las iteraciones. El modelo final suele ser bastante preciso dado que el conjunto de datos utilizado para entrenar y validar el modelo tiene mucha variedad de elementos minimizando el riesgo de sobre-entrenamiento. En la validación cruzada aleatoria la división del conjunto de datos en muestras para entrenamiento y validación es aleatoria. Esto por una parte tiene la ventaja de incrementar todavía más si cabe la variedad de los elementos elegidos para entrenar y validar el modelo generado. Por la otra, corremos el riesgo de repetir muestras, dejarlas sin evaluar o de evaluar la misma combinación varias veces. Esto puede influir en el resultado final dado que éste es calculado como la media aritmética para todas las iteraciones. Finalmente la validación cruzada dejando uno fuera es un caso particular de la variante de K iteraciones, donde el tamaño de los subconjuntos es 1. Esta técnica normalmente requiere de un coste computacional superior a las demás dado que el número de combinaciones generadas por la misma es más grande que las otras variantes. En esta tesis el entrenamiento del clasificador *SVM* se ha llevado a cabo mediante la técnica de validación cruzada de seis iteraciones. En cada iteración, 210 elementos son utilizados para entrenar el modelo y 42 para la validación.

Los parámetros a configurar para un clasificador SVM en general dependen del kernel elegido. En esta tesis se han llevado a cabo varios experimentos con dos funciones de kernel distintas: función lineal y función de base radial gaussiana. En realidad la primera no es una función de kernel, puesto que el producto escalar en sí es una medida de similitud entre dos vectores (ver apéndice C para más detalles). La primera función que se ha evaluado ha sido la función de base radial gaussiana. Esta función tiene la siguiente forma:

$$K(x, x') = e^{-\gamma \left\| x - x' \right\|^2} \tag{5.18}$$

Analizando matemáticamente esta función podemos ver que el área de influencia de cada elemento depende exclusivamente del parámetro γ. Un valor grande de esta variable permite estrechar el área de influencia de cada muestra mientras que valores más pequeños extienden esta área. El parámetro γ es inversamente proporcional al radio de influencia de las muestras elegidas como vectores de

soporte.

La metodología de evaluación que se ha seguido está basada en las recomendaciones del trabajo de [wei Hsu et al., 2010] con algunas modificaciones. Básicamente se han seguido los siguientes pasos:

1. Preparación y adaptación de los datos de prueba
2. Preparar las posibles combinaciones de los parámetros C y γ mediante búsqueda en rejilla

Para cada combinación de valores C_i y γ_i:

1. Generar un modelo con la función radial gaussiana: $K(x, x') = e^{-\gamma\left\|x-x'\right\|^2}$
2. Utilizar validación cruzada para evaluar los parámetros C_i y γ_i
3. Probar el modelo obtenido sobre vídeos reales

La figura 5.21 muestra los resultados obtenidos con esta función para cada una de las categorías. Las zonas amarillas del gráfico corresponden a combinaciones con los mejores resultados de clasificación. En base a los resultados obtenidos se puede ver que tanto para la categoría de autobús como la de camión cisterna el parámetro que más peso tiene sobre el resultado final es γ. Cuanto más pequeño es este parámetro mejores son los resultados de clasificación. En el caso de la categoría camión observamos un comportamiento similar sin embargo esta vez el parámetro C sí tiene efecto sobre el resultado final: los mejores resultados son obtenidos con valores inversamente proporcionales de C y γ. A medida que va creciendo γ el abanico de valores de C que dan buenos resultados es cada vez mayor.

En general, para las tres categorías analizadas los resultados de clasificación empeoran de forma clara a partir de $\gamma = 10^{-1}$. La figura 5.22 muestra la media aritmética de esta función para las tres categorías. Como conclusión podemos decir que los mejores resultados de clasificación se han obtenido combinando un valor grande de C con un $\gamma <= 10^{-2}$.

La segunda función analizada como parte de esta tesis ha sido la función lineal. En este caso el único parámetro variable es C. El rango analizado para este valor es de $[10^{-5}, 10^{5}]$. La figura 5.23 muestra los resultados obtenidos. Analizando esta figura se puede ver que en general los resultados obtenidos independientemente del valor de C son muy buenos tanto para la categoría de autobús como la de camión cisterna. Para la categoría de camión la precisión de clasificación para

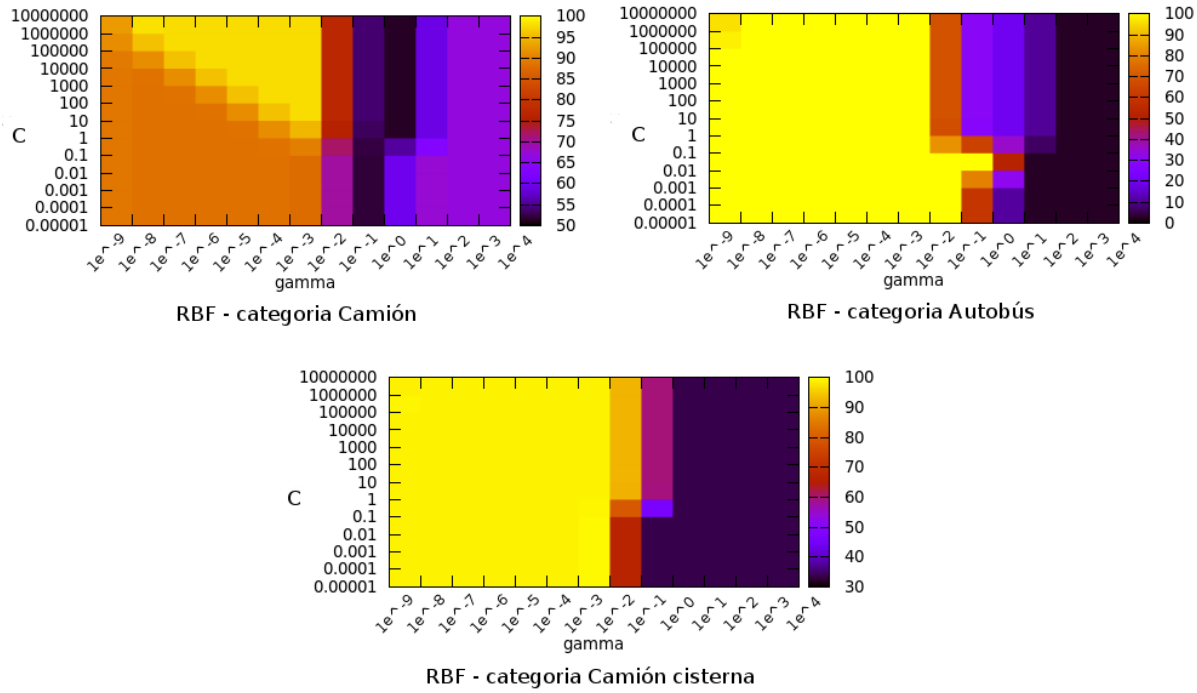


Figura 5.21: Resultados de validación cruzada para la función radial gaussiana para las tres categorías soportadas. Los resultados de precisión son representados en una escala de colores donde las zonas amarillas corresponden a las combinaciones con los mejores valores de precisión.

valores pequeños de C (menores de 10^{-1}) es menor al resto de las categorías. Sin embargo, la precisión mejora de forma clara a partir de valores de C superiores a 10^{-1}.

Si comparamos el rendimiento de la función radial gaussiana con la función lineal podemos decir que esta última es la que mejor resultados ha dado en nuestro problema de clasificación. De hecho, si analizamos más en detalle el comportamiento de la función radial gaussiana podemos ver que los mejores resultados de ésta se han obtenido para valores muy pequeños de $\lambda <= 10^{-2}$. Esto tiene su lógica dado que en este caso la función radial gaussiana se comporta como un modelo lineal. Según demuestra el trabajo de [Keerthi and Lin, 2003] cuando el valor de γ tiende a cero y $C = C_1\sigma^2$ siendo C_1 un valor fijo, entonces el clasificador SVM resultante es equivalente a un clasificador SVM lineal con un parámetro de penalización $C = C_1$. En general según [wei Hsu et al., 2010] cuando la dimensión del espacio de características es muy grande en comparación con el número de muestras disponibles para el entrenamiento (como en nuestro caso), es aconsejable utilizar una función lineal en vez de una función radial gaussiana (o otro tipo de función).

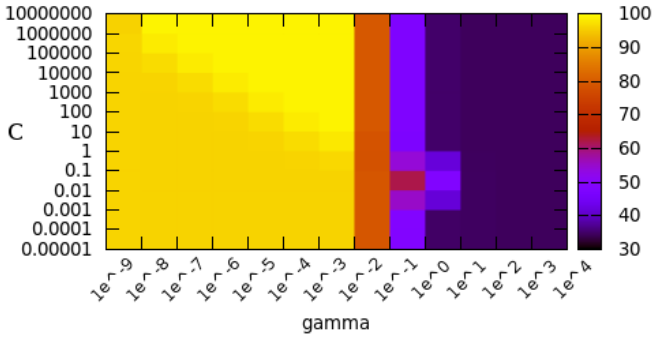


Figura 5.22: Media de los resultados obtenidos para la función radial gaussiana

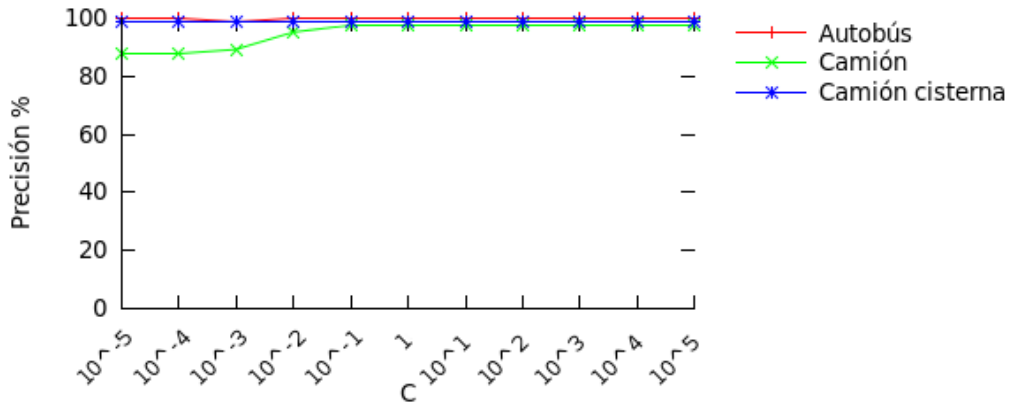


Figura 5.23: Valores de clasificación obtenidos para las tres categorías: camión, autobús y camión cisterna mediante le uso de una función lineal

5.2.4. Parametrización de HOG

En la sección 5.2.1 hemos visto en detalle los distintos parámetros que conforman el descriptor *HOG*. En esta sección veremos el proceso de selección y ajuste de los parámetros de este descriptor y el efecto de éstos sobre los resultados obtenidos en la clasificación. Para llevar a cabo estas pruebas se ha utilizado la misma base de datos descrita en la sección anterior formada por 252 ejemplos de la parte trasera para las tres categorías que tenemos que clasificar. De nuevo, la validación cruzada es utilizada para entrenar y validar el clasificador *SVM* resultante de las distintas configuraciones del descriptor HOG. Dado que en esta sección el objetivo es comprobar el efecto de la parametrización de *HOG* sobre el resultado, se ha fijado la configuración del clasificador *SVM* en la mejor combinación: función lineal con $C = 10^{-1}$. Para facilitar las pruebas y la caracterización del clasificador se ha automatizado el proceso para obtener los resultados de la clasificación correspondientes a configuraciones distintas del descriptor *HOG*.

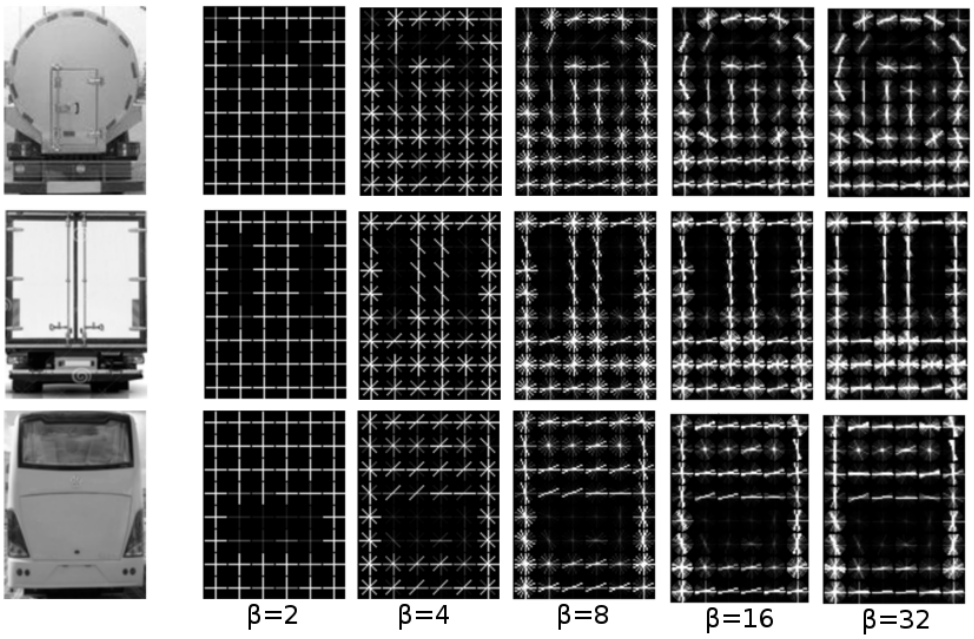


Figura 5.24: Efecto del número de orientaciones elegido sobre el resultado final del descriptor HOG para un tamaño de celda fijo de 16×16

Como se ha visto en la sección 5.2.1 un descriptor *HOG* se caracteriza principalmente por el número de orientaciones β, el tamaño de celda $s = s_1 \times s_2$, el número de celdas por bloque $b = b_1 \times b_2$ y la norma elegida para normalizar los bloques. Los siguientes experimentos han sido llevados a cabo utilizando la misma norma $L2-norm$ y una celda cuadrada (caso habitual en *HOG*). El solapamiento entre bloques ha sido configurado para que sea siempre igual a dos celdas.

Antes de repasar los resultados de la parametrización vamos a comprobar los efectos que tienen los distintos parámetros del descriptor *HOG* de forma visual. La figura 5.24 muestra el efecto que tiene cambiar el número de orientaciones. Para realizar esta prueba hemos fijado el tamaño de celda a 16×16. En esta figura se puede ver que a medida que crece el número de posibles orientaciones el descriptor *HOG* resultante captura mejor los bordes de los objetos representados. Este efecto se puede apreciar de forma clara para el camión cisterna. Si comparamos el *HOG* de esta categoría para los valores de $\beta = 2$ y $\beta = 32$ podemos ver que en el primero es casi imposible apreciar la forma redonda del objeto mientras que en el último la parte circular del camión cisterna se ve de forma clara. Este parámetro tiene un efecto similar sobre el resto de las categorías (camión y autobús). En general se puede ver que a partir de $\beta = 8$ la mejora no es significativa.

El segundo parámetro importante de *HOG* es el tamaño de la celda. A la hora de elegir este parámetro hay que tener en cuenta dos aspectos. El primero es el

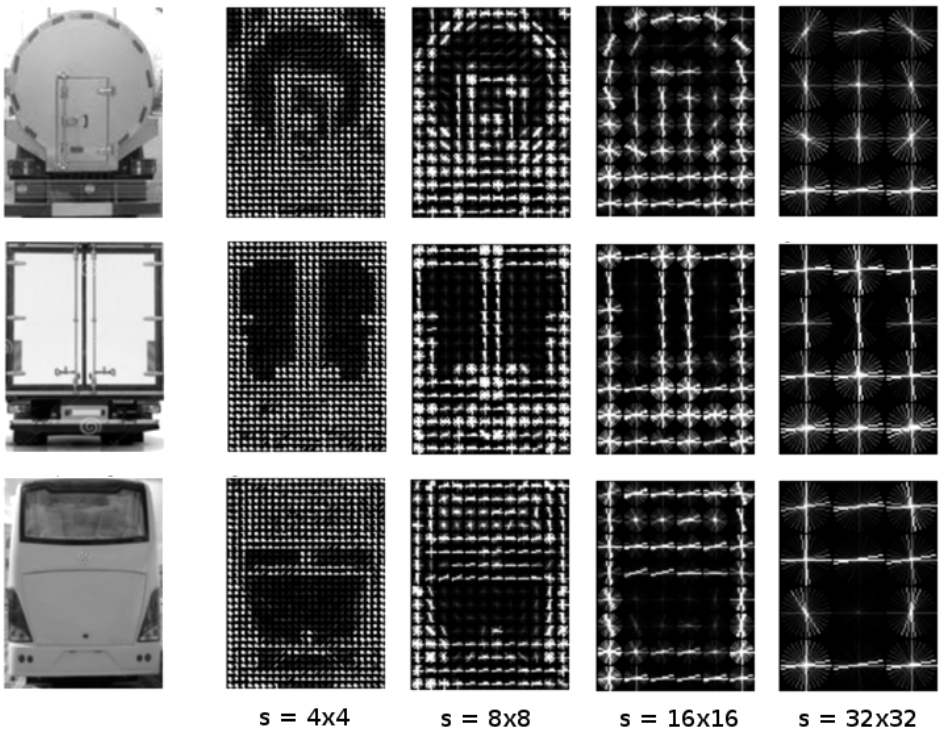


Figura 5.25: Efecto del tamaño de la celda elegido sobre el resultado final del descriptor HOG para un número de orientaciones fijado en $\beta = 16$

efecto que tiene sobre el resultado final. Un tamaño pequeño de celda permite capturar con más detalle el objeto a representar, sin embargo, esto puede ser un inconveniente dado que podemos caer en un problema de sobre-caracterización al recoger demasiados detalles. Esto hará que el descriptor sea muy sensible a cambios en la disposición y orientación del objeto dentro de la imagen. En el otro extremo un tamaño muy grande de celda permite añadir más flexibilidad al descriptor dado que pequeños cambios en la disposición del objeto o la forma de sus bordes no afectan al descriptor resultante. Sin embargo, a medida que crece el tamaño de la celda la información que el descriptor recoge sobre el objeto disminuye hasta llegar a ser insignificante si el tamaño de la celda es demasiado grande.

La figura 5.25 muestra el resultado obtenido al calcular el descriptor *HOG* sobre la misma imagen de entrada y fijando el número de orientaciones en $\beta = 16$. En esta figura podemos ver como un tamaño pequeño de celda $s = 4 \times 4$ permite recoger mucha información sobre el objeto mientras que con un tamaño grande $s = 32 \times 32$ la información recogida sobre el objeto es escasa y apenas se aprecian detalles del mismo en el descriptor resultante.

Finalmente, otro aspecto importante a tener en cuenta es el impacto que

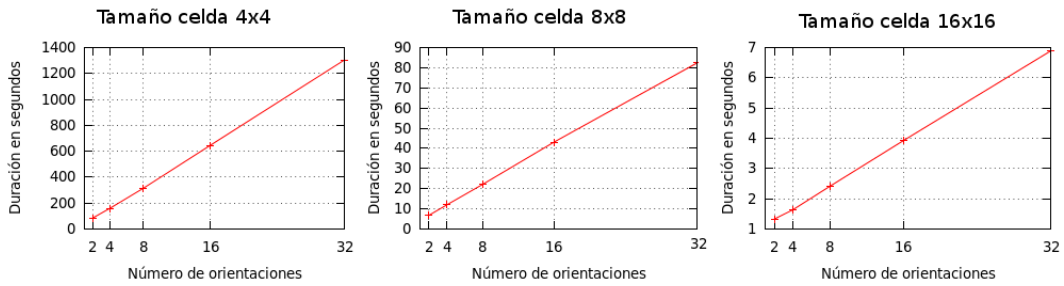


Figura 5.26: Tiempo empleado en el entrenamiento del clasificador SVM en función del tamaño de la celda y el número de orientaciones elegidos para el descriptor HOG

tiene este parámetro sobre el rendimiento de todo el sistema. De las pruebas que se han realizadas hemos observado que cuanto más pequeño es el tamaño de la celda más tiempo de procesamiento necesitamos para calcular el descriptor *HOG*. Puesto que uno de los requisitos esenciales de nuestro sistema es correr en tiempo real tenemos que elegir el tamaño de celda óptimo para cumplirlo.

Todos los experimentos que veremos a continuación han sido ejecutados sobre un AMD Phenom(tm) II X6 1055T a 2.8Ghz y una memoria RAM de 4GB. La figura 5.26 muestra los resultados de los tiempos de entrenamiento necesarios para cada una de estas configuraciones. Si analizamos los datos contenidos en detalle podemos ver que para un tamaño de celda fijo, el tiempo de entrenamiento crece de forma lineal en función del número de orientaciones. Sin embargo, si nos fijamos en la figura 5.27 donde se fija el valor del número de orientaciones y se cambia el valor del tamaño de la celda, podemos ver cuanto más pequeño es el tamaño de celda más tiempo necesitamos para entrenar el clasificador *SVM*. Analizando la operación de entrenamiento que se divide en dos fases: en la primera de ellas se calcula el descriptor *HOG* para cada muestra y en la segunda estas muestras son utilizadas para entrenar el clasificador *SVM*, se ha observado que la mayor parte del tiempo es empleada en el cálculo del descriptor *HOG*. El tiempo empleado para entrenar el clasificador es insignificante en comparación con este tiempo. En la practica los tiempos de entrenamiento en realidad tiene una importancia relativa. Lo que de verdad importa es el tiempo que se tarda en calcular el descriptor HOG para una imagen y predecir a que categoría pertenece mediante SVM.

La tabla 5.3 muestra los resultados de validación del clasificador *SVM* generado para las distintas configuraciones del descriptor *HOG*. Para poder comparar los resultados obtenidos todos los experimentos han sido realizados utilizando los mismos parámetros de SVM: *kernel* lineal con $C = 10^{-1}$. En un primer análisis se puede observar que los mejores resultados son obtenidos para configuraciones

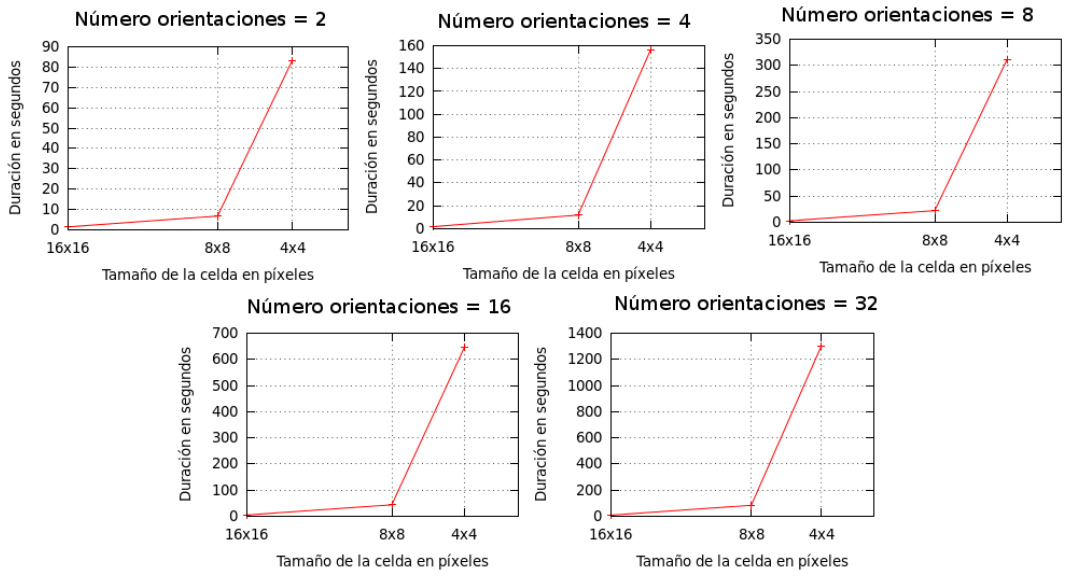


Figura 5.27: Tiempo empleado en el entrenamiento del clasificador SVM en función del tamaño de la celda para distintas configuraciones de orientación

de alta resolución: un pequeño tamaño de celda combinado con un número alto de orientaciones. A medida que sube el número de orientaciones el hecho de subir el tamaño de celda tiene menor impacto sobre el resultado final de clasificación. Si profundizamos un poco más en el análisis, podemos ver que en general el porcentaje de acierto para las tres categorías es alto (superando 94 % en la mayoría de los casos) para la mayoría de las configuraciones exceptuando casos extremos como $\beta = 2$ y un tamaño de celda mayor que $s = 4 \times 4$ donde el clasificador falla para las categorías de camión y camión cisterna, o cuando el tamaño de celda es muy grande como el caso de $s = 32 \times 32$. En general el resultado de la clasificación es muy bueno alcanzando valores muy altos para varias configuraciones donde el tamaño de celda $s = 4 \times 4$ incluso cuando el valor de orientaciones es bajo ($\beta = 2$) los resultados obtenidos con este tamaño de celda son aceptables (todos mayores de 90 %). Otras configuración que resulta en altos valores de clasificación es con un tamaño $s = 8 \times 8$ combinado con un número de orientaciones mayor que 4. Para tamaños de celda de $s = 16 \times 16$ hay que utilizar por lo menos 8 orientaciones para obtener un porcentaje alto de acierto en la clasificación.

El inconveniente de utilizar un tamaño de $s = 4 \times 4$ es su alto coste computacional incluso para un número de orientaciones pequeño como $\beta = 4$. A esto hay que añadir el incremento en el coste de la predicción del SVM dado que el descriptor generado para un tamaño más pequeño de celda contiene más elementos. Esto queda reflejado en la figura 5.28 donde se mide el tiempo empleado para calcular el descriptor HOG de una sola imagen para dos configuraciones de celda

$\beta = 2$	s=4x4	s=8x8	s=16x16	s=32x32
autobús	97.92	96.88	83,33*	100*
camión	93.75	71.88	16,67*	0*
cisterna	92.71	85.42	3,12*	4*

$\beta = 4$	s=4x4	s=8x8	s=16x16	s=32x32
autobús	100.00	98.96	95.83	56,25
camión	97.92	98.96	86.46	54,17
cisterna	98.96	97.92	97.92	96,88

$\beta = 8$	s=4x4	s=8x8	s=16x16	s=32x32
autobús	100.00	98.96	94.79	83,33
camión	97.92	98.96	94.79	77,08
cisterna	100.00	98.96	98.96	95,83

$\beta = 16$	s=4x4	s=8x8	s=16x16	s=32x32
autobús	100.00	100.00	96.88	91,67
camión	97.92	97.92	95.83	90,62
cisterna	98.96	98.96	98.96	97,92

$\beta = 32$	s=4x4	s=8x8	s=16x16	s=32x32
autobús	100.00	100.00	97.92	96,88
camión	98.96	98.96	96.88	95,83
cisterna	98.96	98.96	98.96	96,88

Cuadro 5.3: Resultados del clasificador SVM para distintas configuraciones de tamaño de celda y β. Los resultados marcados con * corresponden a resultados inestables del clasificador.

distintas. En esta figura se puede ver que el tiempo empleado para calcular el descriptor HOG crece de forma lineal con el número de orientaciones. Sin embargo, el tiempo necesario para calcular el descriptor de una configuración con $s = 4 \times 4$ es el triple en comparación con el tiempo necesario para calcular el descriptor con $s = 8 \times 8$ para el mismo número de orientaciones. La figura 5.29 muestra el tiempo empleado para predecir la categoría mediante SVM en función del tamaño del descriptor HOG. De nuevo se puede este tiempo es lineal en función del tamaño del descriptor HOG. Para procesar el vídeo en entrada en tiempo real el sistema tiene que ser capaz de realizar varias operaciones cada $33ms$. Dicho esto, hay que reducir el coste computacional asociado al cálculo del descriptor HOG. Hay que tener en cuenta que el cálculo del descriptor HOG se lleva a cabo para todos los vehículos que hayan sido identificados por la primera fase de la clasificación como camión o autobús. Como consecuencia, hay que multiplicar el valor estimado por el número máximo de camiones/autobuses que el sistema puede manejar. El objetivo es minimizar este tiempo para cumplir con el requisito de tiempo real.

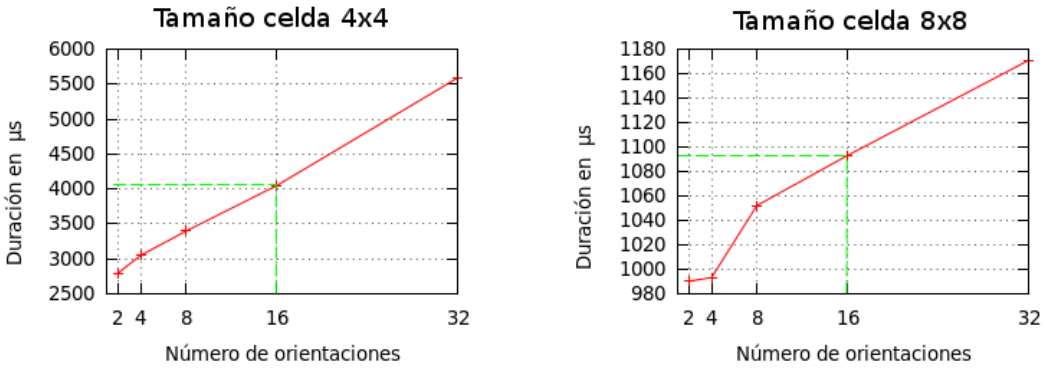


Figura 5.28: Tiempo necesario para calcular el descriptor HOG para distintas combinaciones de tamaño de celda y número de orientaciones.

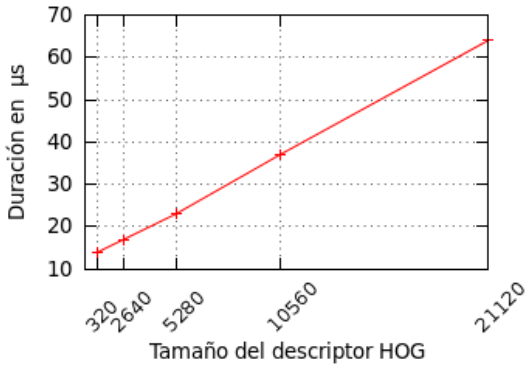


Figura 5.29: Tiempo necesario para predecir el resultado mediante SVM en función del tamaño del descriptor HOG. Los tamaños corresponden respectivamente a configuraciones con: 2, 4, 8, 16 y 32 orientaciones con un tamaño de celda fijo de $s = 8 \times 8$.

Dado que los resultados obtenidos con configuraciones basadas en un tamaño de celda $s = 4 \times 4$ son prácticamente parecidos a los obtenidos con $\beta = 16$, $s = 8 \times 8$ hemos optado al final por utilizar esta configuración, ya que su coste computacional para calcular el descriptor *HOG* es inferior. La configuración elegida para representar las categorías entonces es: $\beta = 16$, $s = 8 \times 8$, un tamaño de bloque de $s = 16 \times 16$ y un solapamiento entre bloques de dos celdas. Esta configuración da lugar a un descriptor *HOG* de 10560 elementos.

5.3. Experimentos sobre tráfico real

Para validar la etapa de clasificación se han llevado a cabo varios experimentos sobre vídeos con distinta composición de categorías de vehículos y con distintas condiciones de: densidad de tráfico, meteorología, iluminación, etc. La idea es sa-

car conclusiones sobre el comportamiento del sistema de clasificación de vehículos sobre distintas condiciones que se dan en la vida real.

5.3.1. Métricas

La medición del rendimiento del clasificador sobre los distintos casos de prueba (vídeos) se basa en varias métricas calculadas sobre matrices de confusión. Una matriz de confusión es una representación de los resultados obtenidos en un experimento de clasificación. El concepto original fue introducido por [Pearson, 1904], pero no llegó al campo de *Machine Learning* hasta las últimas décadas por el trabajo de [Kohavi and Provost, 1998]. Para generar una matriz de confusión para cada clase en este caso se crea una matriz con el resumen de las estadísticas de cada clase por separado frente a todas las restantes. Dado que en el contexto actual tenemos cinco categorías distintas se hará uso de esta técnica para derivar las métricas que se utilizan en medir el rendimiento del clasificador para cada clase.

De nuevo utilizaremos la precisión, exhaustividad y Valor-F como las métricas principales para medir el rendimiento del clasificador en las distintas condiciones de tráfico e iluminación (ver sección 4.3 para más detalles sobre las métricas).

5.3.2. Clasificación en distintas condiciones de iluminación y densidad de tráfico

La tabla 5.4 muestra la matriz de confusión para los resultados de clasificación en un escenario nublado. Analizando los resultados de esta tabla se puede ver que el clasificador ha cometido muy poquitos errores de clasificación. En general, todos los vehículos que han sido detectados han sido clasificados de forma correcta. Este hecho se ve reflejado también la tabla 5.5 que incluye los valores finales de clasificación para todas las categorías incluyendo los totales (verdad absoluta) de cada categoría. Si analizamos estos datos vemos que todas las categorías han sido clasificadas de forma correcta con un porcentaje de acierto mayor que 90 %. La única excepción es la clase de Motos, en este caso vemos que los resultados obtenidos no son tan buenos. El resultado del Valor-F de la clasificación para esta categoría es de 60 % aproximadamente. Esto se debe a dos razones:

1. Problemas de detección en escenarios con nubes

2. Falsos positivos producidos por coches totalmente oscuros

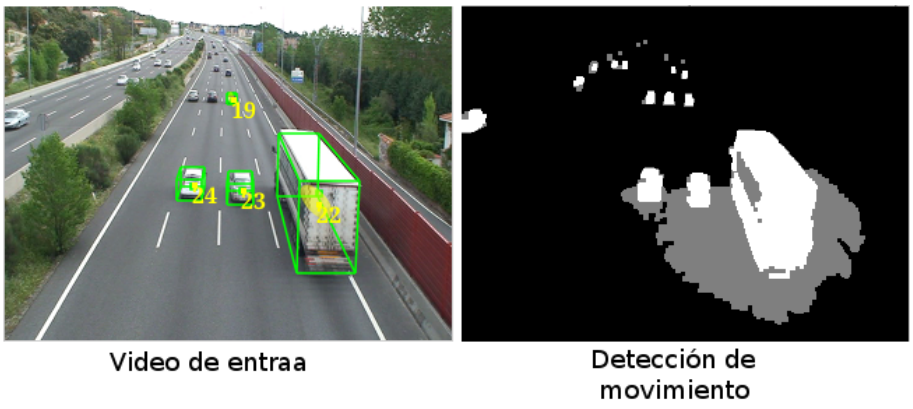


Figura 5.30: Ejemplo de detección de sombras en un escenario nublado. Los grises de la imagen de la derecha corresponden a las sombras detectadas por el módulo de detección de fondo.

	Motos	Coches	Furgonetas	Camiones	Autobuses
Motos	13	0	0	0	0
Coches	9	2148	1	0	0
Furgonetas	0	1	44	1	0
Camiones	0	0	0	41	2
Autobuses	0	0	0	3	10

Cuadro 5.4: Matriz de confusión para escenarios con nubes y bajas oclusiones

El primer problema ya ha sido descrito en detalle en la sección 4.1. Básicamente en estos escenarios el mecanismo de detección de sombras reduce la superficie del cuerpo de los blobs pequeños (como es el caso de las motos). Como consecuencia, éstos aveces quedan descartados por el mecanismo de filtrado de ruido descrito en la sección 4.1. Obviamente se trata de un efecto no deseado de este mecanismo, sin embargo dada la importancia que tiene para todo el sistema, en general trae más beneficios que perjuicios. Este mecanismo se activa siempre y no sólo en caso de escenarios soleados. Como se puede ver en la figura 5.31, incluso en un escenario con nubes el fondo de la escena puede detectar y reportar las sombras de los vehículos. Así que el mecanismo de detección de sombras ha de estar activo siempre, independientemente de la iluminación de la escena. La figura 5.30 muestra varias capturas de clasificación de los vídeos empleados para estos experimentos.

Categoría	Total	T_d	T_p	F_n	F_p	P(%)	R(%)	F_1(%)
Motos	22	22	13	9	9	59.09	59.09	59.09
Coches	2170	2152	2148	25	4	99.81	98.99	99.40
Furgonetas	50	46	44	6	2	95.65	88.00	91.67
Camiones	43	44	41	2	3	93.18	95.35	94.25
Autobuses	13	12	10	3	2	83.33	76.92	80.00

Cuadro 5.5: Resultados de clasificación en escenarios con nubes y bajas oclusiones

Figura 5.31: Ejemplos de clasificación en escenarios con nubes.

	Motos	Coches	Furgonetas	Camiones	Autobuses
Motos	24	0	0	0	0
Coches	2	1269	0	0	0
Furgonetas	0	1	84	0	0
Camiones	0	0	0	48	1
Autobuses	0	0	0	1	10

Cuadro 5.6: Matriz de confusión para escenarios soleados con bajas oclusiones

La matriz de confusión 5.6 muestra los resultados obtenidos en escenarios soleados con bajas oclusiones. Los datos reflejados en esta tabla y los valores de precisión y exhaustividad resumidos en la tabla 5.7 demuestran que el porcentaje de acierto en la clasificación de nuestro sistema es muy alto. En este caso no hay problemas con la categoría de motos ya que el subsistema de detección de fondo no tiene problemas para detectar y reportar de forma correcta blobs de pequeño tamaño. Esto da lugar a una tasa de acierto en la clasificación de motos de 92 %. El resultado para el resto de las categorías es igual de bueno. A excepción de la clase de autobuses se ha superado el 90 % para todas las demás categorías. El sistema roza la perfección para la clase de coches con un porcentaje de acierto en la clasificación cercano al 99 %. Los resultados obtenidos por nuestro sistema son parecidos a los reportados por [Marcos Nieto, Luis Unzueta, Javier Barandiaran, Andoni Cortés, Oihana Otaegui and Pedro Sánchez, 2011] que reporta un rango de precisión de 80-99 %, y superan otros trabajos como [Niluthpol Chowdhury Mithun, Nafi Ur Rashid, and S. M. Mahbubur Rahman, 2012] que reporta un rango de precisión de 88-91 %. La figura 5.32 muestra varias capturas de clasificación de los vídeos empleados para estos experimentos.

La matriz de confusión 5.8 muestra los resultados obtenidos en escenarios

Figura 5.32: Ejemplos de clasificación en escenarios soleados con pocas oclusiones.

Categoría	Total	T_d	T_p	F_n	F_p	P(%)	R(%)	F_1(%)
Motos	26	26	24	2	2	92.31	92.31	92.31
Coches	1302	1271	1269	32	2	99.84	97.47	98.64
Furgonetas	88	87	84	2	5	96.55	95.45	96.00
Camiones	55	51	48	6	2	94.12	87.27	90.57
Autobuses	11	10	9	2	1	90.00	81.82	85.71

Cuadro 5.7: Resultados de clasificación en escenarios soleados con bajas oclusiones

soleados con oclusiones. En general se puede apreciar una ligera bajada en el porcentaje de acierto en la clasificación. Analizando los resultados en detalle se puede ver que la categoría de coches mantiene valores similares a los de los escenarios anteriores mientras que el resto de las categorías experimentan una ligera bajada para situarse entorno al 90 %. Las categorías que mejores resultados registran son la de los camiones y coches. Analizando la matriz de confusión 5.8 para este escenario se puede ver que pese a ser un escenario con oclusiones el clasificador comete muy pocos errores de clasificación. El error de clasificación más común en este caso es clasificar de forma errónea un autobús como si fuera una furgoneta. Esto se debe al mecanismo de eliminación de sombras que en ciertos casos reduce demasiado el tamaño del blob. Esto hace que la primera fase de clasificación basada en plantillas 3D clasifica el blob como furgoneta. La figura 5.33 muestra varios ejemplos de clasificación en escenarios soleados con oclusiones. A diferencia de las figuras anteriores en este caso se ha deshabilitado la funcionalidad de mostrar las plantillas 3D para apreciar de forma más clara los distintos vehículos que están involucrados en una oclusión (vehículos con identificador en color rojo). La categoría mostrada en este caso corresponde a la que más similitud haya acumulado antes de que el vehículo entre en oclusión.

	Motos	Coches	Furgonetas	Camiones	Autobuses
Motos	12	0	0	0	0
Coches	2	753	1	0	0
Furgonetas	0	0	65	0	0
Camiones	0	0	0	52	2
Autobuses	0	0	4	1	14

Cuadro 5.8: Matriz de confusión para escenarios soleados con oclusiones

Categoría	Total	T_d	T_p	F_n	F_p	P(%)	R(%)	F_1(%)
Motos	14	13	12	2	1	92.31	85.71	88.89
Coches	767	756	753	14	3	99.60	98.17	98.88
Furgonetas	77	70	65	12	5	92.86	84.42	88.44
Camiones	56	54	52	4	2	96.30	92.86	94.55
Autobuses	19	14	14	5	2	100.00	73.68	84.85

Cuadro 5.9: Ejemplos de clasificación en escenarios soleados con oclusiones.

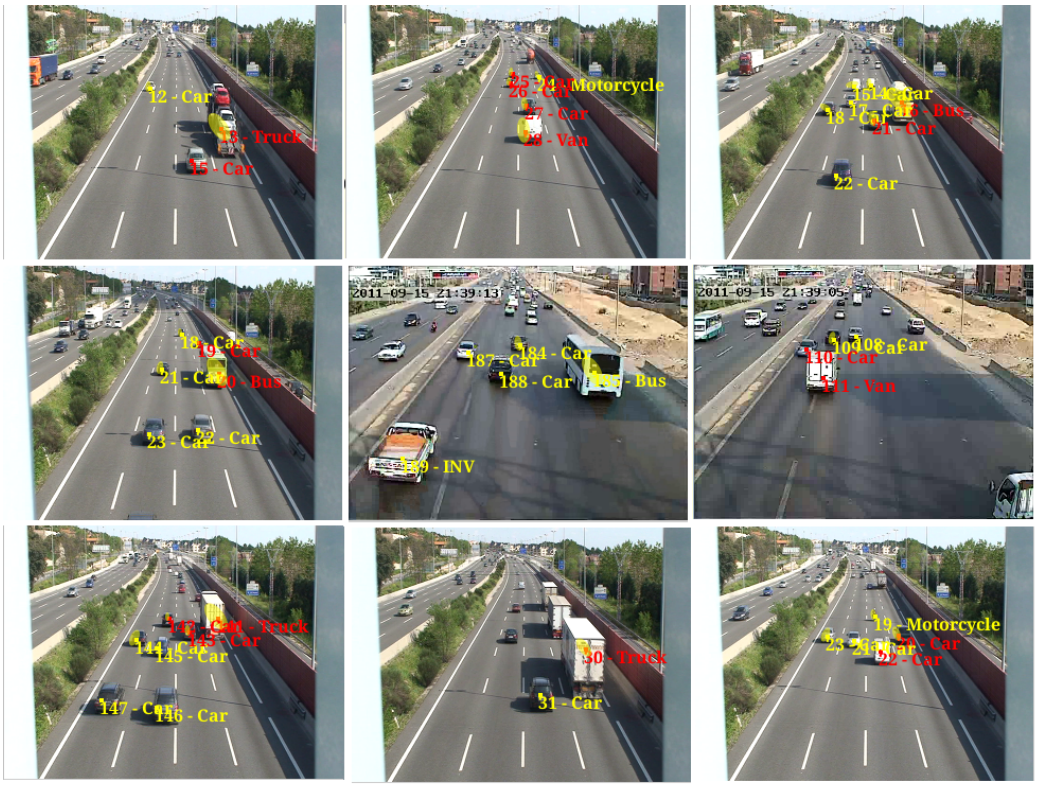


Figura 5.33: Ejemplos de clasificación en escenarios soleados con oclusiones

Figura 5.34: Ejemplos de detección de las categorías camión, autobús y camión cisterna mediante HOG/SVM en escenarios tanto con como sin sombras. La imagen en miniatura corresponde a la imagen rectificada del trasero de cada vehículo.

5.3.3. Clasificación basada en HOG/SVM

Hasta ahora se han presentado los resultados de clasificación para todas las categorías sin entrar en analizar en detalle la prestación del sistema para distinguir entre las categorías de camión y autobús. Como se ha visto en la sección 5.2 se trata de un caso especial dado que la clasificación se basa en HOG/SVM para determinar la categoría final del vehículo. La figura 5.34 muestra ejemplos de clasificación para las tres categorías: camión, autobús y camión cisterna con distintas condiciones de iluminación. En esta figura se puede ver la precisión con la que el sistema es capaz de extraer la parte trasera del vehículo mediante la rectificación explicada en detalle en la sección 5.2.1.

La tabla 5.10 muestra los resultados finales de clasificación para ambas categorías. Se puede ver que el sistema en general es capaz de clasificar ambas con una precisión alta que ronda el 95 % en el caso de los camiones y un 92 % en el caso de los autobuses.

Categoría	Total	T_d	T_p	F_n	F_p	P(%)	R(%)	F_1(%)
Camión	55	51	48	6	2			
Camión	43	44	41	2	3			
Camión	56	54	52	4	2			
Autobús	19	14	14	5	2			
Autobús	11	10	9	2	1			
Autobús	13	12	10	3	2			
Camión	154	149	141	12	7	94.63	91.56	93.07
Autobús	43	36	33	10	5	91.67	76.74	83.54

Cuadro 5.10: Resultados de clasificación para las categorías Camión/Autobús.

Por otro lado, los resultados de la exhaustividad son mejores en el caso de camiones frente al de autobuses con un porcentaje de 91 % frente a 77 % para el caso de los autobuses. Esta diferencia en general se debe a fallos de clasificación para la categoría de autobuses relacionados con el tamaño del blob o un fallo del SVM/HOG para determinar la categoría correcta. Los errores que se producen en el caso de camiones/autobuses están más bien relacionados con la forma de los objetos. En la figura 5.35 se puede ver dos ejemplos de fallos de clasificación. En el primer de ellos un autobús esta siendo clasificado de forma incorrecta por la técnica SVM/HOG como un camión. Si nos fijamos en este ejemplo en concreto se puede ver que este autobús en la parte trasera lleva una raya vertical negra. Como consecuencia, el descriptor HOG resultante se parece mucho al de un camión. En el segundo ejemplo un camión está siendo clasificado como autobús, en este caso el camión no tiene puertas en la parte trasera, además lleva un dibujo en la parte de arriba justo donde los autobuses suelen tener la ventana. Esto confunde al clasificador SVM/HOG y la categoría final asignada en este caso es incorrecta.

Figura 5.35: Ejemplos de fallos de clasificación mediante HOG/SVM

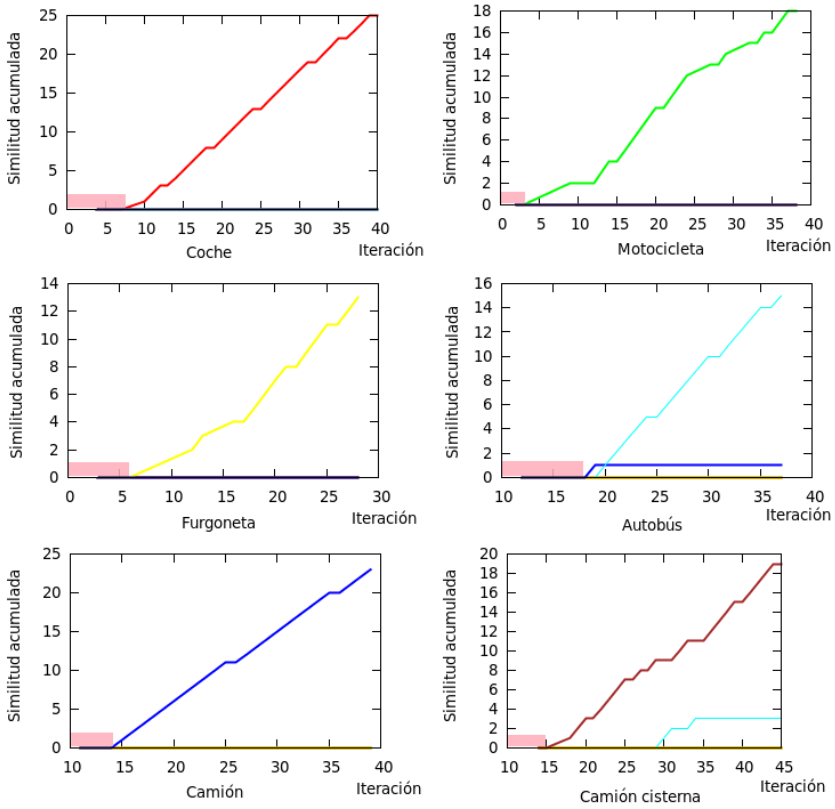


Figura 5.36: Análisis de la acumulación de similitud para cada categoría en condiciones favorables. El rectángulo que parte desde el origen de tiempos indica el tiempo que se tarde en iniciar la clasificación del vehículo.

5.3.4. Análisis de la acumulación de similitud

Como ya se había discutido en la sección 5.1.2 la idea es acumular información de clasificación de los vehículos a lo largo de toda la zona de seguimiento. Cada vez que el vehículo es clasificado el sistema acumula la similitud en la categoría correspondiente. De esta forma, si se produce un error puntual de clasificación en alguna iteración durante el transcurso de la clasificación, éste no afecta al resultado final de la clasificación dado que la similitud acumulada en condiciones normales termina siendo la de la categoría correcta. Para analizar el comportamiento de la función de similitud con cada categoría, se han llevado a cabo varios experimentos que consisten en medir la variación de la similitud que acumula un vehículo a lo largo de la zona de seguimiento para las distintas categorías soportadas bajo distintas condiciones de iluminación. Finalmente, en la misma figura se puede observar que el tiempo que tarda un vehículo en comenzar la clasificación (básicamente en traspasar la zona de entrada) depende la categoría. En general cuanto más pequeño es el tamaño de la categoría menos tiempo se tarda en arrancar la clasificación del vehículo.

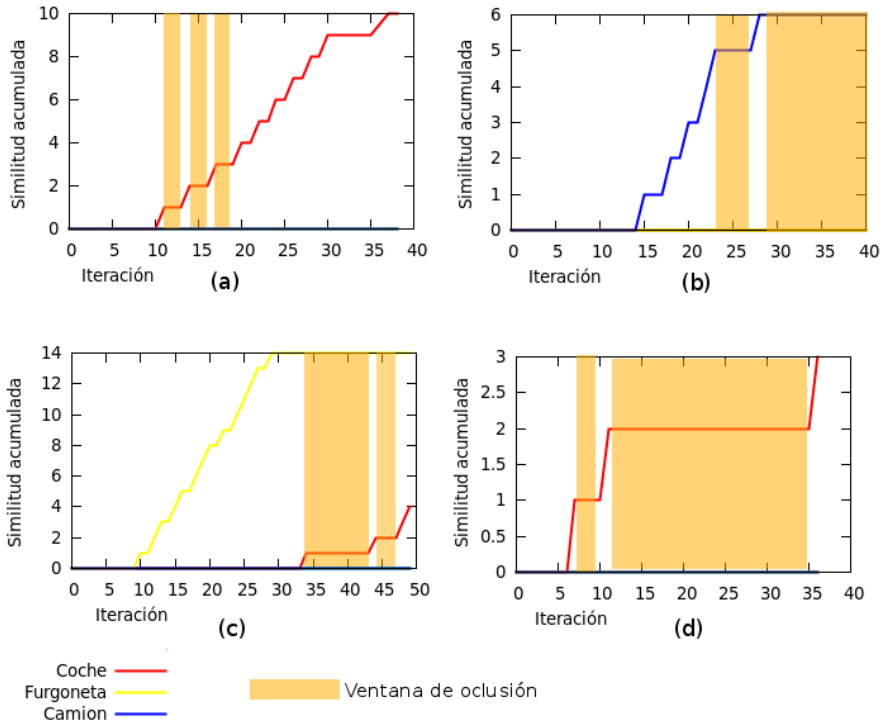


Figura 5.37: Análisis de la acumulación de similitud en presencia de oclusiones. Las ventanas de oclusión representan intervalos de tiempo donde el vehículo no es clasificado porque está involucrado en una oclusión

El primer experimento consiste en observar la acumulación de similitud para todas las categorías soportadas. Para ello, en un escenario normal (sin oclusiones) se ha medido la similitud acumulada para un vehículo representativo de cada categoría. La figura 5.36 muestra los resultados obtenidos. En esta figura el tiempo que tarda cada vehículo en iniciar su clasificación está señalado con un rectángulo que cubre desde el inicio de los tiempos hasta la primera acumulación de similitud. En los gráficos de similitud acumulada de esta figura se puede ver claramente que la acumulación de similitud de la categoría dominante es lineal creciente. En la mayoría de los casos la similitud acumulada para las demás categorías se mantiene en cero. En algunos casos como para las categorías autobús y camión cisterna se puede ver que hubo un par de errores de clasificación sin consecuencias en el resultado final.

En un segundo experimento se ha observado la acumulación de similitud para varios vehículos que entran en algún momento en oclusión con otros vehículos. Como hemos visto anteriormente, cada vez que se detecta una oclusión el sistema deja de acumular similitud para todos los vehículos involucrados en la misma para evitar el riesgo de introducir errores en los resultados ya acumulados. La figura

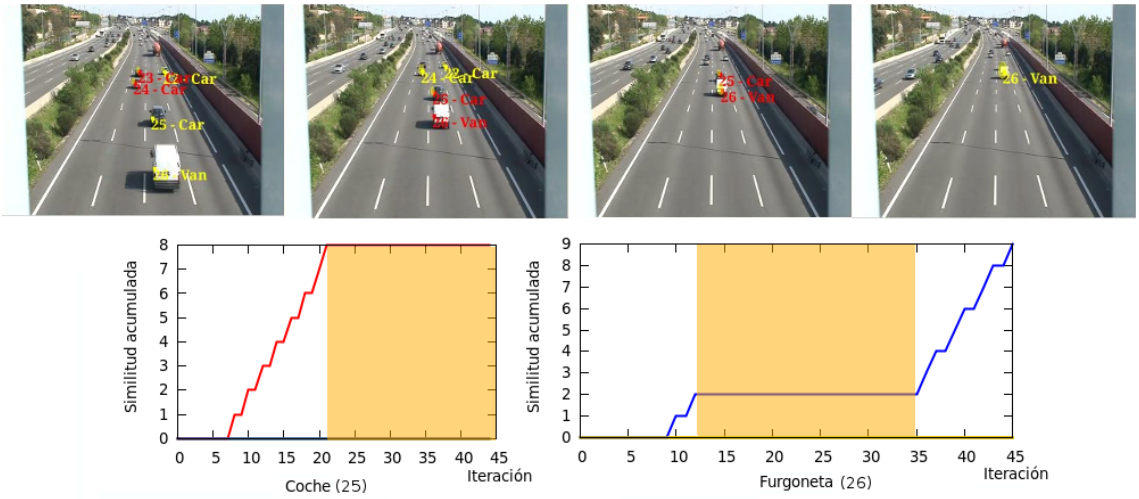


Figura 5.38: Comportamiento de la acumulación de similitud para dos vehículos que entran en oclusión

5.37 muestra varios ejemplos de la acumulación de similitud en estos escenarios. En la subfigura **(a)** el vehículo (un coche en este caso) entra en oclusión varias veces entre las iteraciones 10 y 20, a partir de este momento ya no entra más en oclusión y sigue acumulando similitud para su categoría *coche* hasta el final de seguimiento. En la subfigura **(b)** tenemos un caso similar, sin embargo en este caso el camión se recupera de la primera oclusión que sufre en la iteraciones 24-26 pero a partir del iteración 29 entra de nuevo en oclusión de la cual no sale hasta el final del seguimiento. El caso de la subfigura **(c)** es un poco distinto. En este caso la furgoneta va acumulando similitud para su categoría de forma correcta hasta la iteración 30, en es momento hay un error de clasificación (porque está entrando en oclusión), y es clasificada como coche (de ahí la curva de similitud para la categoría coche empieza a acumular valor) durante las iteraciones 34-42 el vehículo entra en oclusión, se recupera durante una sola iteración pero de nuevo vuelve a entrar otra vez. En todo momento la clasificación está sufriendo errores dado que la clase que está en este momento acumulando similitud es la de coche en vez de furgoneta. Aún así, la clase correcta ha acumulado tanta similitud que estos errores no alteran la categoría final estimada para el vehículo.

Finalmente, la subfigura **(d)** muestra el caso de un coche que pasa la mayor parte del tiempo de seguimiento en oclusión. Durante su trayectoria este coche tiene tres oportunidades de clasificación que se aprovechan e incrementan la similitud de la clase correcta. Al final del seguimiento aún cuando el vehículo haya tenido poquitas oportunidades de clasificación estas han sido aprovechadas para estimar su categoría de forma correcta.

La figura 5.38 muestra el comportamiento de la acumulación de similitud para dos vehículos que entran en oclusión. En esta figura el coche con el identificador 25 entra en oclusión con el de la furgoneta con el identificador 26. La figura muestra la secuencia de oclusión y al mismo tiempo cómo han ido acumulando la similitud ambos vehículos. En la gráfica correspondiente al coche podemos ver que éste ha empezado a acumular similitud en su categoría en su iteración 6 y ha seguido este curso hasta la iteración 21 donde entra en oclusión con la furgoneta. En el caso de la furgoneta ésta ha empezado a acumular similitud en su iteración 9 pero pronto (iteración 12) entra en oclusión con el coche. A partir de este instante ambos vehículos dejan de acumular similitud (zona naranja de las dos gráficas) hasta que se produce la oclusión total en la iteración 45 del coche (correspondiente a la iteración 35 de la furgoneta). En este momento el sistema detiene el seguimiento del coche, la situación de oclusión es resuelta y sigue acumulando similitud de forma correcta para la furgoneta. Cabe señalar que el origen de tiempos para ambos vehículos es distintos, puesto que entran en la escena en diferentes instantes.

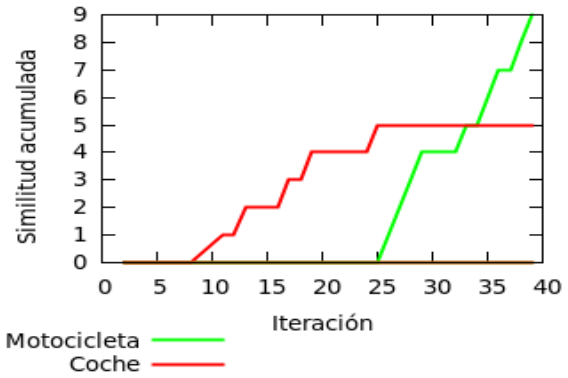


Figura 5.39: Fallo de clasificación debido a fallos en la detección de fondo

Finalmente, cabe destacar que al fin y al cabo el sistema de clasificación depende en gran medida de la etapa de detección de fondo. Cuanto más fiable es esta última menos errores de seguimiento y clasificación se obtendrán. Como ya se ha visto en la sección 5.3.2 en escenarios con nubes la capacidad de detección de la categoría de motocicletas es más baja en comparación con el resto de los escenarios. Esto se debe en parte a los fallos detección de esta categoría pero también a los falsos positivos en forma de coches que acaban siendo erróneamente clasificados como motocicletas. La figura 5.39 ilustra un ejemplo de estos fallos de clasificación. En este ejemplo se puede ver cómo un vehículo empieza la clasificación siendo clasificado como coche pero al final acaba siendo asignado de

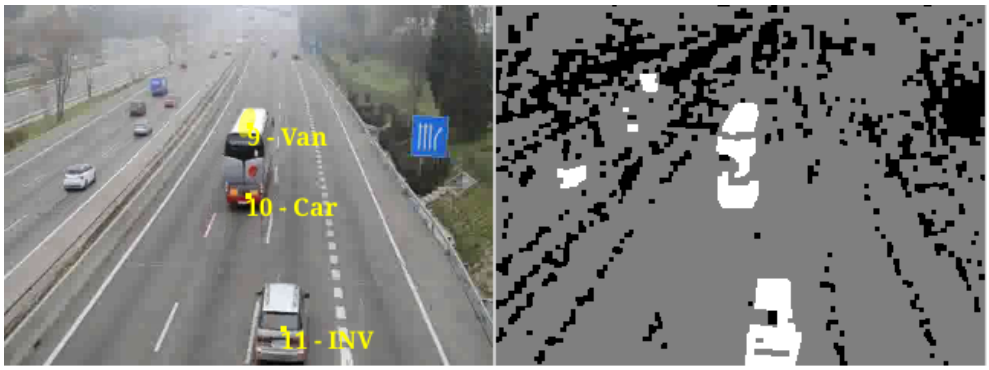


Figura 5.40: Errores de clasificación ocasionados por fallos en al detección de fondo en escenarios con niebla

forma incorrecta a la categoría de motocicletas. En este caso en concreto se trata de un coche con color oscuro, que en las primeras iteraciones al estar cerca de la cámara y tener un tamaño considerable el subsistema de detección de fondo no tiene problemas para identificarlo de forma correcta. A medida que el vehículo se aleja el tamaño de su blob es cada vez más pequeño tanto por el efecto de la proyección como por la detección de fondo que en estos escenarios no maneja bien blobs pequeños con color oscuro. Como consecuencia, el blob detectado es cada vez más pequeño y va acumulando más similitud en la categoría motocicletas. Esto se aprecia de forma clara en la figura 5.39 donde a partir de la iteración 25 la categoría motocicleta pasa a ser la dominante. En torno a la iteración 35 la similitud acumulada para esta categoría supera la acumulada por la clase coche y el vehículo acaba siendo incorrectamente clasificado como motocicleta.

5.3.5. Tráfico denso y distintas condiciones de meteorología

Aunque no forma parte de los objetivos de esta tesis construir un sistema que funcione en todas las condiciones meteorológicas, se han llevado a cabo varios experimentos bajo distintas condiciones con el fin de evaluar el rendimiento de los algoritmos implementados en cada caso y ver sus puntos fuertes y débiles.

La figura 5.41 muestra capturas de varios experimentos que se han llevado a cabo bajo distintas condiciones de meteorología. En general se ha podido comprobar que el sistema funciona muy bien cuando las condiciones son favorables (tiempo soleado). En escenarios con nubes el sistema se comporta bien en lo que se respecta a la clasificación de todas las categorías a excepción de las motocicletas donde el porcentaje de acierto es más bajo si lo comparamos con el resto

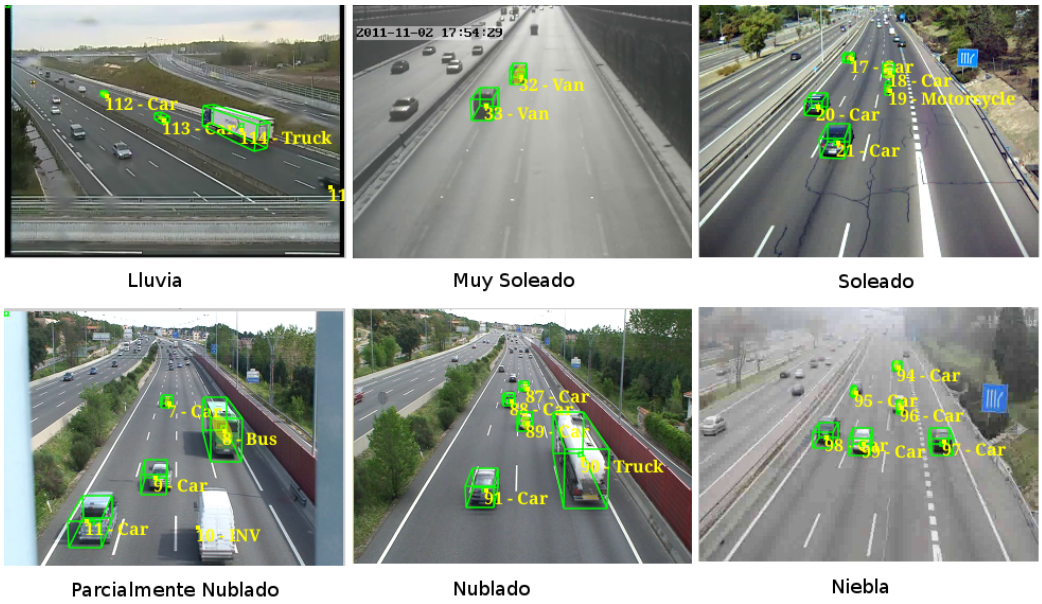


Figura 5.41: Clasificación con distintas condiciones meteorológicas

de las clases. En caso de lluvia el sistema se comporta de forma parecida a escenarios con nubes a excepción de casos extremos donde hay baja visibilidad o la cámara es salpicada por gotas de agua. La figura 5.42 muestra un ejemplo de estos casos. Por último, en escenarios con niebla la clasificación en general funciona razonablemente bien, sin embargo la baja visibilidad afecta a la calidad de los blobs detectados. Este hecho se acaba propagando a los resultados finales de la clasificación en forma de fallos de detección de la categoría de motocicletas y fallos de clasificación en el caso de coches que se detectar de forma incorrecta como motocicletas. La figura 5.40 muestra un ejemplo de fallo de clasificación en estos escenarios.

Por último, la figura 5.43 muestra varios ejemplos de clasificación en tráfico denso. En esta figura se puede apreciar cómo el sistema ha sido capaz de detectar y clasificar correctamente vehículos de distintas categorías.

5.4. Conclusiones

En este capítulo se han visto las técnicas y los algoritmos empleados en la clasificación de vehículos. En concreto, se han visto dos técnicas distintas: la primera basada en plantillas 3D de referencia. Esta técnica consiste en definir un tamaño 3D para cada una de las categorías soportadas. Aprovechando el conocimiento preciso de la información tridimensional de las escena (gracias a la cámara calibrada) esta técnica clasifica los vehículos comparando la huella producida por el

Figura 5.42: Fallos de clasificación bajo condiciones meteorológicas extremas. En (a) hay tres falsos positivos dado que no hay ningún vehículo en la escena. En (b) el subsistema de detección no detecta el vehículo que hay en la imagen.

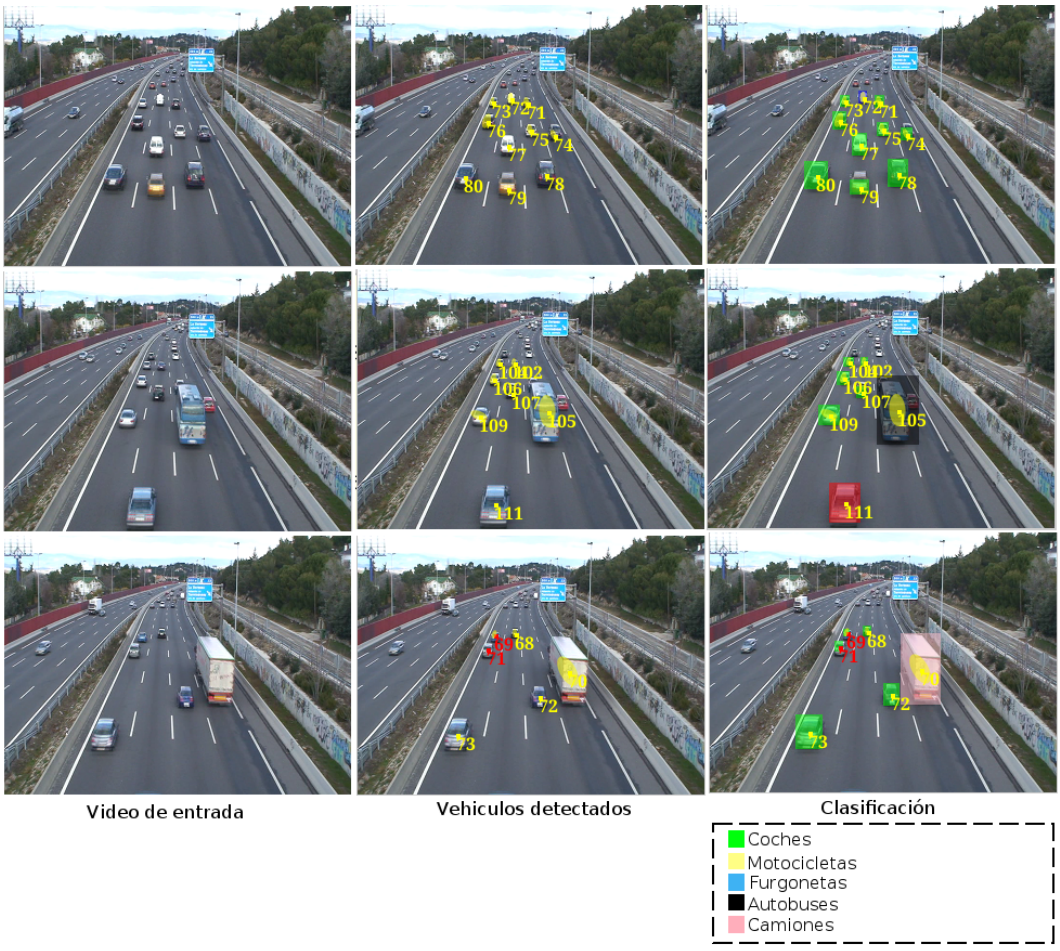


Figura 5.43: Resultados de clasificación en escenarios de tráfico denso.

blob detectado por el módulo de detección de vehículos contra la huella producida por la proyección de la plantilla sobre el plano de la carretera. Esta técnica a pesar de ser precisa y eficiente está limitada a la hora de clasificar dos categorías que tienen el mismo tamaño. Esto se da en nuestro caso para la categoría de autobuses y camiones que tienen un tamaño similar. Para solventar este problema se ha diseñado e implementado un segundo algoritmo de clasificación basado en *SVM* y *HOG* capaz de distinguir entre estas categorías. En ambos casos la etapa de clasificación aprovecha el conocimiento acerca de las oclusiones detectadas en la fase de seguimiento para mejorar los resultados de clasificación de los vehículos involucrados. La idea básica es evitar seguir acumulando la información de clasificación cuando dos o más vehículos entran en oclusión.

La idea detrás del clasificador *HOG*/*SVM* es combinar la información 3D de las plantillas para extraer en tiempo real la parte trasera de los vehículos cuando éstos son clasificados por la primera etapa como camión o autobús. La extracción precisa se lleva a cabo mediante una técnica de rectificación de planos. El descriptor HOG de la parte extraída es clasificado por un *SVM* que había sido anteriormente entrenado sobre una base de datos de 250 ejemplares aproximadamente. Para caracterizar el rendimiento del modelo generado se ha llevado a cabo un estudio en profundidad tanto del clasificador *SVM* como de los distintos parámetros del descriptor *HOG* con el objetivo de entender su efecto sobre el comportamiento del clasificador y elegir la configuración óptima para llevar a cabo la tarea de la clasificación en tiempo real. La parametrización final elegida para el descriptor HOG es de $\beta = 16$, $s = 8 \times 8$, un tamaño de bloque de $s = 16 \times 16$ y un solapamiento entre bloques de dos celdas. Esta configuración además de ofrecer un alto porcentaje de acierto en la clasificación tiene la ventaja de tener menor coste computacional frente a las otras alternativas evaluadas.

El capítulo es reforzado con una extensa sección de experimentos cuyo objetivo es validar los algoritmos diseñados sobre tráfico real bajo distintas condiciones de tráfico, iluminación y meteorología. De nuevo estos experimentos se han llevado a cabo sobre la base de datos propia además de los vídeos ofrecidos por la base de datos *GRAM-RTM*. Los resultados de estos experimentos han demostrado la validez del sistema de clasificación construido como parte de esta tesis. A excepción de la categoría de motocicletas donde el clasificador ha demostrado ciertos problemas en escenarios de baja visibilidad, el sistema ha reportado porcentajes de acierto muy altos para el resto de las categorías en todas las condiciones probadas en los experimentos.

Tanto el algoritmo de clasificación basada en plantillas 3D como el clasificador basado en HOG/SVM han dado altos porcentajes de clasificación para las categorías principales en la mayoría de los escenarios. La función de similitud ha demostrado ser una herramienta precisa para comparar los blobs con las plantillas de las distintas categorías soportadas. Finalmente, el mecanismo de acumulación de similitud a lo largo de la zona de seguimiento hace que la etapa de clasificación sea robusta frente a ruido, errores puntuales y oclusiones.

La técnica de extracción y rectificación en tiempo real de la parte trasera para clasificar las categorías Camión/Autobús ha sido probada con éxito sobre tráfico real. Los resultados obtenidos en este caso han sido similares a los obtenidos como parte de las pruebas de entrenamiento del clasificador HOG/SVM no obstante en las pruebas realizadas, los resultados finales de clasificación reflejan un tasa de precisión superior para el caso de los camiones frente a la categoría de autobuses.

Capítulo 6

Conclusiones

A lo largo de los capítulos anteriores hemos visto en detalle los problemas que esta tesis aborda, se ha repasado y clasificado las técnicas y la bibliografía más significativa relacionada con cada uno de ellos. Este capítulo repasa el trabajo realizado durante esta tesis así como las conclusiones y resultados finales de la misma. Se realizará un repaso de los objetivos que han sido planteados en el capítulo 1 verificando el grado de cumplimiento y satisfacción de cada uno de ellos. En paralelo, se describirán las soluciones aportadas y las principales contribuciones de esta tesis. Finalmente se hará un barrido de las líneas futuras que este trabajo puede tener.

6.1. Objetivos

Tal y como se ha visto en la sección 1.4, el objetivo principal de esta tesis es crear un sistema completo para la monitorización de tráfico rodado en una autopista de varios carriles en tiempo real basado en visión. Este objetivo ha sido cumplido mediante la implementación de un sistema compuesto por tres grandes bloques.

El capítulo 2 hace un barrido extenso de la bibliografía más significativa relacionada con los temas tratados en esta tesis, repasando las distintas técnicas utilizadas para abordar los problemas relacionados con la monitorización de tráfico y profundizando en aquellas que tienen una relación directa con esta tesis.

El capítulo 3 presenta el diseño global del sistema implementado y hace un estudio detallado de la composición física de la escena, sus elementos y los factores que hay que tener en cuenta. Este capítulo además presenta el sistema de calibración semi-automática de la cámara basado en la técnica *VVH*. Esta técnica

ayuda a cumplir el objetivo de un sistema flexible capaz de soportar y adaptarse a distintas configuraciones de la cámara.

El capítulo 4 presenta los dos subsistemas de detección y seguimiento de vehículos. En el primero de ellos se hace una presentación de la técnica de detección automática de carretera y del subsistema que se encarga de la detección y segmentación de los vehículos que pasan por la carretera. El núcleo de este bloque se basa en la detección de fondo mediante una versión mejorada de *GMM* apoyada por un mecanismo de filtrado de ruido basado en información 3D de la escena. En el mismo contexto, este capítulo presenta los detalles de una técnica de eliminación de sombras que ayuda a mejorar los procesos posteriores sobre todo la etapa de clasificación donde es necesario realizar *cortes* precisos de la parte trasera del vehículo en caso de camiones y autobuses.

Este capítulo presenta en detalle también los algoritmos que han sido implementados como parte de esta tesis para resolver el problema de seguimiento de varios vehículos de forma simultánea. Se han presentado dos algoritmos distintos: el primero basado en proximidad espacio-temporal mientras que el segundo combina este criterio con un seguimiento basado en *KLT* para resolver el problema de las oclusiones. Se han estudiado en detalle las oclusiones, cómo se originan y cómo son detectadas y resueltas con el algoritmo propuesto.

La fase de seguimiento es validada experimentalmente mediante una sección entera dedicada a varias pruebas cuyo objetivo es evaluar el rendimiento y el comportamiento de los distintos algoritmos implementados bajo distintas configuraciones de carretera, condiciones de iluminación y tráfico. Se ha realizado un estudio detallado de la respuesta del sistema frente a las oclusiones y cómo afectan éstas a la precisión de seguimiento para todos los algoritmos soportados. Como conclusión, a la vista de los resultados experimentales obtenidos se puede decir que la precisión de seguimiento y conteo de vehículos del sistema es muy alta (mayor que 95 % en la mayoría de los casos). Los algoritmos de detección y resolución de oclusiones han dado buenos resultados ayudando a conseguir estos niveles tan altos de precisión. El algoritmo basado en *KLT* ha demostrado ser una solución eficiente que ha dado muy buenos resultados de seguimiento en presencia de oclusiones. Los resultados obtenidos en este capítulo ayudan en parte a cumplir el tercer objetivo que consiste en seguir de forma simultánea varios vehículos sobre una autopista.

El capítulo 5 presenta los detalles de las técnicas de clasificación de vehículos implementadas en esta tesis. La primera de ellas se basa en plantillas 3D. En

este contexto cada categoría de vehículos es representada por una plantilla que representa su tamaño 3D. Mediante el uso de la cámara calibrada el sistema compara la huella producida por el movimiento del vehículo contra la proyección de las distintas plantillas que representan las categorías soportadas por el sistema. Esta comparación es utilizada para calcular la similitud del vehículo respecto a cada una de las categorías soportadas. El vehículo es asignado a la categoría que más similitud acumula a lo largo de la zona de seguimiento. La segunda técnica de clasificación se basa en un clasificador *SVM*/HOG y su objetivo es distinguir entre las categorías de camión/autobús dado que la primera técnica basada en tamaños no es capaz de diferenciar entre estas dos categorías porque tienen un tamaño similar. Esta técnica se basa en la información 3D de las categorías para recortar de forma precisa la parte trasera del vehículo (camión o autobús), rectificar el área recortada a un tamaño fijo, calcular su descriptor HOG y clasificar el resultado mediante un *SVM* que ha sido previamente entrenado para diferenciar entre tres categorías de vehículos: Camiones, Autobuses y Camiones cisterna.

El capítulo 5 concluye con una sección experimental donde los algoritmos creados son validados con tráfico real bajo distintas condiciones de iluminación, densidad de tráfico y oclusiones. Los dos algoritmos implementados como parte de esta tesis han demostrado altos valores de precisión en la clasificación de todas las categorías en la mayoría de las condiciones. El único caso donde la precisión de la clasificación es más baja en comparación con la media obtenida por el resto de las categorías, es en el caso de las motocicletas en escenarios con nubes. El problema en realidad no proviene de la etapa de clasificación como tal sino del subsistema de detección de fondo que confunde parte del cuerpo de los blobs con sombras. Dicho esto, la categoría de las motocicletas en el resto de escenarios es detectada con una fiabilidad alta (con una media del 90 %). Las demás categorías en principio son clasificadas de forma satisfactoria en todas las condiciones alcanzando valores de precisión superiores al 90 % para la mayoría de las categorías. Los resultados conseguidos con los algoritmos implementados en este capítulo ayudan a conseguir el tercer objetivo (relacionado con clasificación de vehículos).

El último objetivo ha sido cumplido con el diseño global del sistema capaz de monitorizar de forma simultánea varios carriles en tiempo real. Este criterio se ha tenido en cuenta durante el diseño de todos los algoritmos implementados como parte de esta tesis.

6.2. Contribuciones

Como parte del desarrollo de esta tesis, además de hacer uso de otras técnicas clásicas se han hecho varias contribuciones con el fin de mejorar las técnicas existentes de seguimiento, clasificación y en general la monitorización de tráfico basada en visión. Un resumen del trabajo realizado ha sido publicado en el siguiente articulo:

- Kachach, R., Cañas J. M. "Hybrid three-dimensional and support vector machine approach for automatic vehicle tracking and classification using a single camera". Journal of Eletronic Imaging (**JCR 2016**). [Kachach and Cañas, 2016].

A continuación un resumen de las contribuciones más importantes de esta tesis:

- Algoritmo de seguimiento con soporte para la detección y resolución de oclusiones basado en *KLT*
- Técnica de clasificación automática de vehículos en tiempo real basada en plantillas 3D apoyada en un mecanismo robusto de acumulación de similitud
- Técnica de clasificación que emplea un *SVM* y la información 3D de las plantillas para recortar la parte trasera, rectificarla y clasificarla en base a su descriptor *HOG*
- Amplia base de datos de vídeos de tráfico real, etiquetados y clasificados según las condiciones de: iluminación, densidad de tráfico y meteorología. Este conjunto de vídeos es puesto a disposición de la comunidad cinética para resolver la escasez de este tipo de *data sets*.
- Técnica de calibración semi-automática de la cámara aplicada al contexto de tráfico

Además de los aportes principales esta tesis presenta una serie de contribuciones menores que consisten en:

- Mecanismo de filtrado de ruido adaptativo que mejora sustancialmente la calidad de los *blobs* detectados eliminando ruido de la escena mediante el empleo de información 3D de las plantillas.

- Algoritmo para la eliminación de sombras basado en la geometría del blob mediante la aplicación de una secuencia controlada de operadores morfológicos.

- Sistema para la detección automática de la carretera empleando información del fondo.

6.3. Trabajos futuros

Una vez descrito todo el trabajo llevado a cabo como parte de esta tesis, los puntos fuertes y débiles de los algoritmos implementados como parte de la misma, esta sección traza varias líneas futuras de investigación como posible continuación directa y realista de este trabajo.

Quizás la mayor limitación que tiene el sistema descrito en este trabajo frente a otras alternativas clásicas como bucles por inducción es no poder funcionar de noche. Es un problema clásico que sufre la mayoría de los sistemas basados en visión. En este sentido una de las posibles líneas de investigación es adaptar los algoritmos implementados, sobre todo el basado en plantillas 3D para procesar información proveniente de una cámara térmica. Esta adaptación en principio tiene que ser posible dado que el algoritmo basado en plantillas ha sido diseñado para no depender de información como el color o la textura de los vehículos.

Otro de los problemas que no han sido abordado en esta tesis es la clasificación de vehículos en escenarios complejos como el caso de atascos, condiciones meteorológicas extremas o con una cámara en movimiento. Todos estos problemas plantean retos que requieren de otro tipo de algoritmos (no basados en detección de fondo). En este sentido se podría adoptar un mecanismo basado en *deep learning* para una detección más robusta de vehículos. Esta capacidad hará posible utilizar el sistema en escenarios urbanos y no sólo en autopistas, además, el sistema podría procesar vídeos provenientes de una cámara en movimiento.

Por último el sistema se puede mejorar y ampliar añadiendo más inteligencia y capacidad para procesar y entender la semántica de lo que está pasando en el escenario. Esto incluye detectar siniestros, anomalías o comportamientos sospechosos de vehículos sobre la carretera.

Apéndice A

Modelo pinhole de cámara

Dada una cámara cualquiera, ésta se puede modelar con los siguientes parámetros:

- **Parámetros intrínsecos:** Dependen del modelo utilizado para representar la cámara. En el modelo *Pinhole* son:
  - f_x: distancia focal multiplicada por el factor de tamaño de píxeles en el eje X, S_x
  - f_y: distancia focal multiplicada por el factor de tamaño de píxeles en el eje Y, S_y
  - (U_0, V_0): punto principal
- **Parámetros extrínsecos:** Estos representan la posición y orientación de la cámara en el mundo. En general se representan con dos matrices genéricas de rotación y traslación RT. Pero en el caso de rotación hay varias maneras de representarla (cuaterniones, ángulos de Euler, foco de atención + rol, etc). En esta tesis en concreto las librerías utilizadas hacen uso de de una matriz genérica de rotación.

Como se puede observar en la figura A.1, en el triángulo (C,P,Z), un punto P_w^{cam} expresado en el sistema de coordenadas de la cámara con coordenadas $[X_p, Y_p, Z_p]$ se proyecta en un punto del plano de imagen p de coordenadas (x, y). Aplicando el teorema de *Tales* en este triángulo obtenemos:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{A.1}$$

El siguiente paso es convertir el punto (x, y) en (u, v) que viene a ser los píxeles correspondientes en el sensor de la imagen. Para ello, tenemos que saber

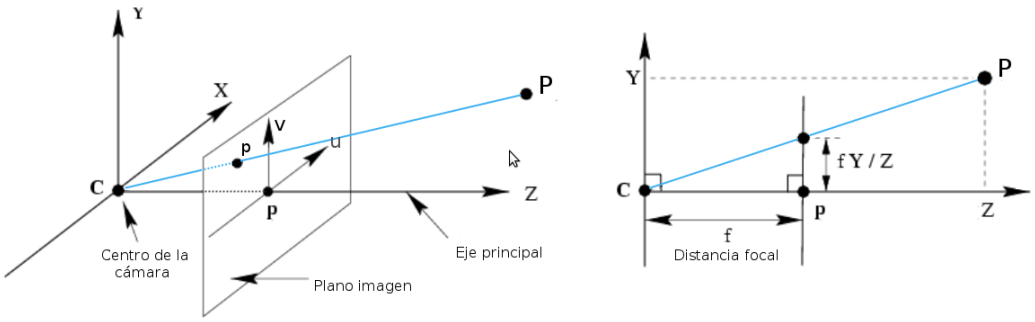


Figura A.1: Modelo pinhole

el tamaño de los píxeles en horizontal y vertical. La conversión se lleva a cabo mediante las siguientes ecuaciones:

$$u = S_x \cdot x + u_0 \tag{A.2}$$

$$v = S_y \cdot y + v_0 \tag{A.3}$$

Donde (u_0, v_0) son las coordenadas del punto principal en píxeles.

La correspondencia de un punto 3D P_w^{cam} a otro punto 2D P_{im} no es única. Dado un punto P_{im} (u, v) todos los puntos que pertenecen a la recta que une el centro de proyección C con el P_w^{cam} y P_{im} impactan en el mismo punto P_{im}. Utilizando coordenadas homogéneas, la ecuación A.1 se puede expresar:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_w^{cam} \tag{A.4}$$

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & U_0 & 0 \\ 0 & f_y & V_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_w^{cam} \tag{A.5}$$

El punto P_w^{cam} está expresado en el sistema de coordenadas de la cámara. En la vida real las coordenadas vienen expresadas respecto a otro sistema de referencia absoluto que no tiene porqué ser el de la cámara. Dado un punto P_w^{abs} expresado en un sistema de referencia en el universo, para hallar el punto P_{im} correspondiente a este punto lo primero que tenemos que hacer es expresar el mismo en el sistema de referencia de la cámara. Sólo entonces podemos aplicar la ecuación A.5 para hallar el punto P_{im}. De modo genérico, para pasar del sistema de coordenadas absoluto al sistema de referencia de la cámara, tenemos que aplicar

una rotación y una traslación (alguna de ellas podría ser identidad o nula). Las matrices correspondientes a este cambio de base se denominan matriz de rotación y traslación extrínseca, RT_{ext}. Este cambio de coordenadas se puede expresar de forma matricial en coordenadas homogéneas con la siguiente ecuación:

$$P_w^{cam} = R \cdot T \cdot P_w^{abs} \tag{A.6}$$

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_{cam} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_w \tag{A.7}$$

Recapitulando lo anterior, para calcular el punto P_{im} correspondiente a un punto P_w^{abs} cualquiera tenemos que hacer los siguientes pasos:

- Trasladar P_w^{abs} al punto P_w^{cam} expresado en el sistema de referencia de la cámara utilizando la ecuación A.7
- Proyectar el punto P_w^{cam} sobre la imagen utilizando la ecuación A.5

Combinando estas dos ecuaciones obtenemos A.9, la ecuación general para proyectar cualquier punto 3D del universo sobre el plano imagen. En la vida real esto no es siempre posible, puesto que el sensor de la cámara es de tamaño limitado y habrá ciertos puntos que salen de su campo de visión.

$$P_{im} = K \cdot R \cdot T \cdot P_w^{abs} \tag{A.8}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}_{im} = \begin{bmatrix} f_x & 0 & U_0 & 0 \\ 0 & f_x & V_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_w \tag{A.9}$$

Apéndice B

Algoritmo Kanade-Lukas-Tomasi

KLT es un algoritmo de seguimiento de características en imágenes. Es el resultado de la combinación del trabajo de [Lucas and Kanade, 1981] que desarrolló la idea de utilizar gradientes ponderados con una aproximación de la segunda derivada para emparejar dos imágenes y el trabajo de [Tomasi and Kanade, 1991] que mejoraron el seguimiento mediante la selección de las características apropiadas para esta tarea.

La implementación piramidal de este algoritmo está descrita en detalle en el trabajo de [yves Bouguet, 2000a]. La novedad de esta implementación radica en la aplicación de KLT de forma recursiva sobre una pirámide de imágenes. Dadas dos imágenes I y J en escala de grises, definimos $I(p) = I(x, y)$ y $J(p) = J(x, y)$ como los valores de los píxeles en la posición correspondiente al punto $p = [x, y]^T$. Dado un punto $u = [u_x, u_y]^T$ en la imagen I el objetivo del algoritmo es encontrar el punto $v = u + d = [u_x + d_x, v_y + d_y]^T$ en la segunda imagen J de tal forma que $I(u)$ y $J(v)$ sean similares. Dicho en otras palabras, estamos buscando el desplazamiento $d = [d_x, d_y]^T$ que ha sufrido el punto u entre las imágenes I y J. Nos queda por definir el concepto de *similitud* en un área 2D. Dados dos valores positivos w_x y w_y definimos la velocidad d de la imagen como el vector que minimiza la función ϵ:

$$\epsilon(d) = \epsilon(d_x, d_y) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (I(x, y) - J(x + d_x, y + d_y))^2 \tag{B.1}$$

La función de similitud se mide alrededor del punto en una ventana $(2w_x + 1) \times (2w_y + 1)$. Esta ventana también se conoce como la ventana de integración. La elección del tamaño idóneo de la ventana es un compromiso entre precisión y robustez. Un valor pequeño de ventana permite encontrar el valor idóneo alrededor del punto de partida suponiendo que el punto no haya efectuado un gran

desplazamiento. Un valor grande de ventana permitirá cubrir estos casos pero al mismo tiempo aumenta el riesgo de calcular de forma incorrecta el desplazamiento ya que naturalmente pasamos a tener más candidatos. La implementación piramidal trata de resolver este problema.

Dada una imagen I de tamaño $n_x \times n_y$, definimos $I^0 = I$ como la imagen de nivel cero. Básicamente es la imagen con la misma resolución que la imagen de entrada. El tamaño de esta imagen es igual que la imagen de partida. El resto de las imágenes en la pirámide son construidas de forma recursiva: I^1 es construida en base a I^0, I^2 desde I^1 y así de forma recursiva. Cada píxel $I^L(x, y)$ de la imagen correspondiente al nivel L se calcula interpolando los valores alrededor del píxel $I^{L-1}(2x, 2y)$. La profundidad de la pirámide m es un valor escogido de forma heurística, los valores típicos para m son 2,3 y 4. Por ejemplo para una imagen de entrada de tamaño 640×480, las imágenes I^1, I^2, I^3 tiene respectivamente los tamaños 320×240, 160×120 y 80×60. La profundidad de la pirámide afecta a la capacidad del algoritmo para manejar desplazamientos mayores que el tamaño de la ventana (w_x, w_y) así que el valor de m se debe escoger pensando en el desplazamiento máximo que un píxel puede efectuar en el caso particular.

El algoritmo recursivo de KLT piramidal empieza calculando el flujo óptico en el nivel más alto en la pirámide L_m. Este resultado se propaga al nivel inferior y así de forma sucesiva hasta llegar al nivel base L_0 (correspondiente a la imagen original). De forma genérica si $g^L = [g_x^L, g_y^L]^T$ es el valor del flujo óptico calculado en el nivel $L + 1$ entonces para calcular el valor del flujo óptico en el nivel L tenemos que calcular el vector $d^L = [d_x^L, d_y^L]$ que minimiza la función ϵ^L:

$$\epsilon^L(d^L) = \epsilon^L(d_x^L, d_y^L) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (I^L(x, y) - J^L(x + g_x^L + d_x^L, y + g_y^L + d_y^L))^2 \tag{B.2}$$

En esta ecuación el valor del flujo óptico calculado en el nivel anterior g^L se utiliza para desplazar la imagen original I respecto a la imagen de origen J. De esta manera el vector $d^L = [d_x^L, d_y^L]$ es pequeño y por lo tanto más sencillo de calcular mediante un KLT estándar. Una vez calculado el valor de d^L entonces es propagado de forma recursiva al siguiente nivel $L - 1$. El nuevo valor en este nivel es calculado con la siguiente fórmula:

$$g^{L-1} = 2(g^L + d^L) \tag{B.3}$$

Siguiendo el mismo procedimiento, el valor de g^{L-1} se propagará al siguiente nivel y así hasta llegar al nivel base L_0 donde el valor final del desplazamiento es

calculado con la misma fórmula:

$$d = g^0 + d^0 \tag{B.4}$$

El algoritmo piramidal empieza calculando el desplazamiento en el nivel más alto de la pirámide L_m. El resultado es propagado hacia el siguiente nivel L_{m-1} y así sucesivamente hasta alcanzar el nivel L_0 (que corresponde a la imagen original). En general, dado el valor $g^L = [g_x^L, g_y^L]^T$ del flujo óptico calculado en el nivel $L+1$, para calcular el valor del flujo óptico en el nivel L tenemos que calcular el vector $d^L = [d_x^L, d_y^L]$ que minimiza la función residual:

$$\epsilon^L(d^L) = \epsilon^L(d_x^L, d_y^L) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (I^L(x,y) - J^L(x + g_x^L + d_x^L, y + g_y^L + d_y^L))^2 \tag{B.5}$$

En esta función el valor del flujo óptico g^L (calculado en el nivel anterior) es utilizado para pre-trasladar la imagen de entrada en la segunda imagen J. Una vez calculamos el valor d^L, éste es propagado de forma recursiva al próximo nivel $L-1$. El nuevo valor de desplazamiento es calculado con la siguiente fórmula:

$$g^{L-1} = 2(g^L + d^L) \tag{B.6}$$

Siguiendo la recesión, el valor final del desplazamiento es calculado en el nivel L_0 utilizando la siguiente fórmula:

$$d = g^0 + d^0 \tag{B.7}$$

Hasta aquí se ha visto como el algoritmo piramidal recursivo que propaga el cálculo entre los distintos niveles. En cada nivel L de la pirámide, la versión iterativa del algoritmo de Lucas-Kanade es utilizada para hallar el vector d^L que minimiza la función residual ϵ^L. Si reescribimos la ecuación B.5 cambiando el vector de desplazamiento d^L con $\overline{v} = [v_x, v_y]$ y la posición con $p = [p_x, p_y]^T$ obtenemos:

$$\epsilon(\overline{v}) = \epsilon(v_x, v_y) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (A(x,y) - B(x + v_x, y + v_y))^2 \tag{B.8}$$

En el caso óptimo la primera derivada de ϵ con respecto a $\overline{v}$ es cero. Si desarrollamos la derivada mediante el método de Taylor entorno al punto $\overline{v} = [0,0]^T$ (esto es posible porque el desplazamiento previsto es muy pequeño) el valor óptimo del flujo óptico es calculado con la siguiente fórmula:

$$\overline{v}_{opt} = G^{-1}\overline{b} \tag{B.9}$$

Donde:

$$G \doteq \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{B.10}$$

y

$$b \doteq \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{bmatrix} \delta I I_x \\ \delta I I_y \end{bmatrix} \tag{B.11}$$

La ventaja de la implementación piramidal del algoritmo es que mientras calculamos el valor final del vector desplazamiento d, el valor del flujo óptico d^L de cada nivel intermedio se mantiene pequeño. Asumiendo que cada nivel de la pirámide puede manejar un desplazamiento máximo de d_{max}, entonces el valor final del desplazamiento que puede manejar una pirámide de m niveles es $d = (2^{L+1} - 1)d_{max}$. Esto nos permite manejar largos desplazamientos con un tamaño de ventana relativamente pequeño.

Apéndice C

Maquinas de Vectores Soporte

SVM es un clasificador lineal que intenta maximizar el margen entre dos clases basándose en un número determinado de muestras que se denominan vectores de soporte. En otras palabras, dado un conjunto de muestras de entrada SVM intenta hallar el hiperplano que lo divide en dos regiones disjuntas maximizando el área libre de muestras entre ellas. En el caso particular de dos dimensiones la solución es una línea recta. Para hallar la línea de separación que maximiza el margen, SVM se basa en un número limitado de elementos que se denominan vectores soporte y que son escogidos de ambas clases. Cabe señalar que el conjunto de líneas que separan ambos conjuntos podría ser infinito sin embargo, el objetivo es encontrar la línea que separa las dos clases generando la mayor separación entre ambas. En el ejemplo de la figura 5.11 podemos ver dos clases de muestras, los vectores soporte de cada una de ellas y dos soluciones distintas. Se puede ver claramente que el margen generado por la segunda solución es mucho mayor que el generado por la primera. Dado que la solución encontrada por SVM es la óptima, cualquier otra solución nos dará un margen más estrecho y por lo tanto con menos seguridad a la hora de clasificar muestras adyacentes a la frontera entre las dos clases.

Si analizamos de nuevo la figura 5.11 podemos ver que el resto de los elementos de cada clase no intervienen en el cálculo de la solución. Ésta se basa sólo en los vectores soporte. Se trata de una característica importante de SVM que le hace robusto estadísticamente hablando ya que cualquier cambio en la distribución de elementos que no son vectores soporte no afecta a la solución final y por tanto a la capacidad discriminatoria del clasificador. Adicionalmente esta característica hace que el SVM sea robusto frente al problema del *overfitting* (sobreentrenamiento) ya que la solución final no depende del conjunto empleado durante la fase de entrenamiento.

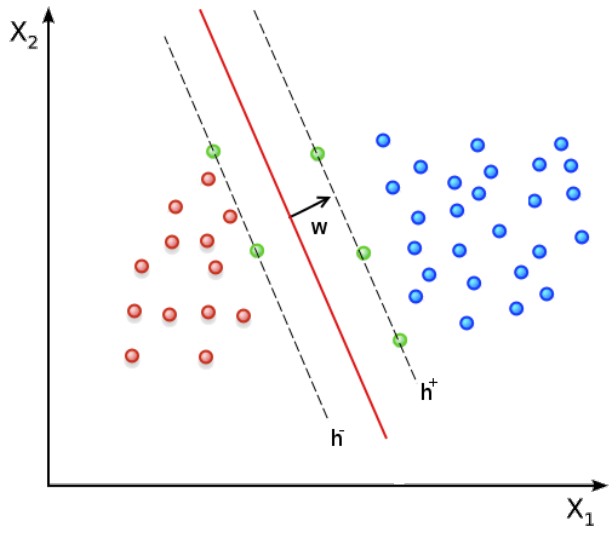


Figura C.1: Elementos geométricos relacionados con SVM

Matemáticamente hablando, el objetivo principal de SVM es hallar la solución en forma de hiperplano que maximiza el margen entre las dos clases. Al ser lineal, tiene que haber un vector w ortogonal al hiperplano solución con un coeficiente de intersección b tal que:

$$w^T x + b = 0 \tag{C.1}$$

Este hiperplano se obtiene como el intermedio entre h^- y h^+ que son los hiperplanos que contienen los vectores de soporte. Estos se definen con las siguientes expresiones:

$$h^+ \rightarrow w^T x_i + b = +1 \qquad i = 1, .., n$$
$$h^- \rightarrow w^T x_i + b = -1 \qquad i = 1, .., n$$

Dicho en otras palabras, cuando aplicamos la expresión $w^T x_i + b$ a muestras positivas obtenemos un $+1$ y cuando la aplicamos a muestras en el lado negativo obtenemos un -1. Las dos expresiones se pueden resumir en la siguiente fórmula siendo y_i una etiqueta cuyo valor es $+1$ para las muestras en el lado positivo y -1 para las muestras que caen en el lado negativo.

$$y_i(w^T x_i + b) \geq 1 \qquad i = 1, .., n$$

Con estos datos en la mano, el margen (región amarilla en la figura 5.11) se puede calcular como la distancia entre los hiperplanos h^- y h^+ con la siguiente

fórmula:

$$d^+ = d^- = \frac{|wx + b|}{\|w\|}$$
$$margen = d^+ + d^- = 2\frac{1}{\|w\|}$$

Maximizar el margen equivale a minimizar la función $\Phi(w)$ sujeto a la condición que habíamos establecido anteriormente: $y_i(w^T x_i + b) \geq 1$. Este proceso es equivalente al problema original sin embargo permitirá simplificar la notación posterior y obtener expresiones más compactas.

$$\Phi(w) = \frac{1}{2}\|w\|^2 = \frac{1}{2}w^T w$$

Para resolver este problema de optimización cuadrática, se hace uso del Lagrangiano L formulado de la siguiente manera:

$$\begin{aligned} &L(x, \alpha) = f(x) + \sum_i \alpha_i g_i(x) \qquad \forall \alpha_i \geq 0 \\ &Donde: \\ &\quad f(x) \rightarrow \Phi(w) \\ &\quad g_i(x) \rightarrow y_i(w^T x_i + b) \geq 1 \end{aligned} \tag{C.2}$$

En la construcción del Lagrangiano $f(x)$ es la función por optimizar mientras que $g_i(x)$ son las restricciones. De forma general, la solución al problema de optimización cuadrática se obtiene minimizando el Lagrangiano respecto a los parámetros w, b y maximizando el resultado respecto a los multiplicadores de Larnage α_i. En términos formales:

$$max_{\alpha_i}(min_{w,b} L(w, b, \alpha)) \tag{C.3}$$

Si desarrollamos el Lagrangiano de la ecuación C.2 obtenemos lo siguiente:

$$L(x, b, \alpha_i) = \frac{1}{2}w^T w - \sum_i \alpha_i [y_i(w^T x_i + b) - 1] \tag{C.4}$$

Antes de aplicar la primera condición KKT (Karush-Kuhn-Tucker) para minimizar L respecto a w, b, primero tenemos que calcular las derivadas parciales respecto a ambas variables. Esto implica:

$$\begin{aligned} &\frac{\partial L}{\partial w} = w - \sum_i \alpha_i y_i x_i = 0 \Rightarrow w = \sum_i \alpha_i y_i x_i \\ &\frac{\partial L}{\partial b} = -\sum_i \alpha_i y_i = 0 \Rightarrow \sum_i \alpha_i y_i x_i = 0 \quad \forall \alpha_i \geq 0 \end{aligned} \tag{C.5}$$

Si sustituimos el valor de w en la ecuación C.4 y desarrollamos el sumatorio obtenemos:

$$\Theta(\alpha) = \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T xj + \sum_i \alpha_i$$

$$Sujeto\, a: \tag{C.6}$$

$$\sum_i \alpha_i y_i = 0 \quad \forall \alpha_i \geq 0 \tag{C.7}$$

Aplicando la condición complementaria de KKT obtenemos:

$$\alpha_i[1 - y_i(w^T x_i + b)] = 0, \qquad i = 1, ..., n \tag{C.8}$$

Dado que por la restricción C.7 afirma que $\alpha_i > 0$, entonces la segunda parte de la expresión C.8 tiene que ser cero y por lo tanto:

$$y_i(w^T x_i + b) = 1, \qquad i = 1, ..., n$$

En otras palabras, las muestras que cumplen esta igualdad tienen un valor de $\alpha_i > 0$. Por definición, estas muestras son los vectores de soporte. De la misma manera si observamos el valor de w en la ecuación C.5 podemos concluir que el hiperplano solución es una combinación lineal de vectores de soporte.

Siguiendo con el análisis, hemos transformado el problema original en un nuevo problema de maximización de $\Theta(\alpha)$. Esta formulación del SVM se conoce como la formulación dual. Cabe señalar que la solución obtenida para la versión básica del problema (donde suponemos que los datos son linealmente separables) no es paramétrica. No habrá que ajustar ningún parámetro duran la fase de aprendizaje.

La solución para el segundo problema parte de la formulación dual de SVM desarrollada en la ecuación C.6. Si analizamos bien esta última veremos que los términos correspondientes a la muestra aparecen en forma de productos escalares $x_i^T x_j$. Dicho esto, la idea básica consiste en trasladar las muestras a otro espacio de mayor dimensión mediante una función ϕ con la esperanza de que en este nuevo espacio los datos sean linealmente separables. Dado que la ecuación C.6 sólo se basa en productos escalares entre vectores no tenemos que saber exactamente la función ϕ ni calcularla ya que este proceso podría ser demasiado complejo computacionalmente hablando o incluso imposible en caso de espacios infinitos. Según el teorema de Mercer, una función kernel se puede expresar en forma de un producto escalar en otro espacio. Así que basándose en este teorema, bastaría con encontrar un kernel $K(x_i, x_j)$ que sustituya el producto escalar en el nuevo espacio

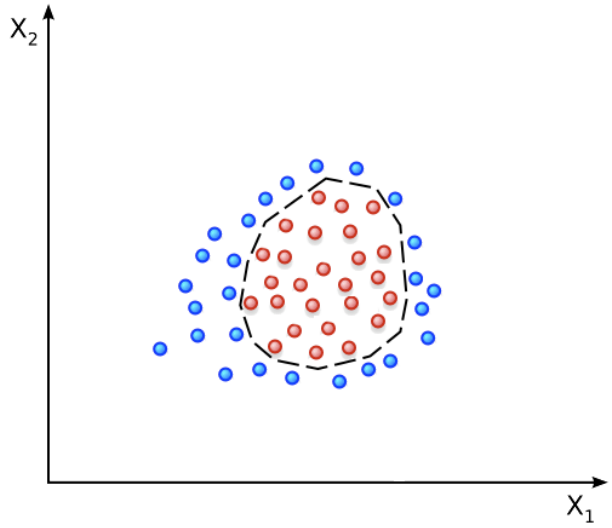


Figura C.2: Conjunto de datos que no se puede separar de forma lineal

escogido. De esta manera logramos el objetivo final que consiste en trasladar los cálculos a otro espacio sin ni siquiera tener que calcular o saber la función de mapeo a este espacio. Una vez elegimos el kernel que vamos a utilizar, éste se puede integrar de forma sencilla en la formulación dual del SVM. Esto da lugar a la nueva formulación:

$$\Theta(\alpha) = \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{K}(\mathbf{x_i}, \mathbf{x_j}) + \sum_i \alpha_i$$
$$b = y_i - w^T \phi(x_i) = y_i - \sum_j y_i \alpha_j K(x_j, x_i) \tag{C.9}$$

Resumiendo, el resultado final de la clasificación queda determinado por la función *signo* de la siguiente función:

$$f(x) = sgn(\sum_i y_i \alpha_i K(x_i, x_j) + b)$$

En la práctica, un kernel en realidad es una función que mide la similitud entre dos muestras (dos vectores en general). El producto escalar geométricamente es la proyección de un vector sobre el otro en el espacio correspondiente, así que de nuevo podemos interpretar el resultado como una medida para calcular la similitud entre dos vectores. A la hora de escoger el Kernel tenemos que aprovechar cualquier conocimiento sobre el dominio o la naturaleza de las muestras. En el caso de no disponer de dicha información se puede hacer uso de algún kernel estándar. Dependiendo de la función kernel que se utiliza puede ser necesario ajustar más parámetros durante la fase de entrenamiento.

En la vida real las muestras de entrada suelen ser más complejas que el caso básico que hemos empleado durante el análisis matemático y en general podemos

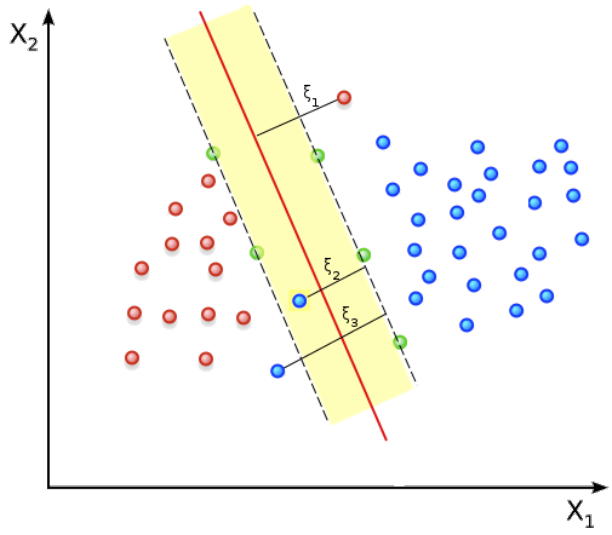


Figura C.3: Variables de holgura

encontrar problemas con clases cuyos elementos son cuasi-separables (como el caso de la figura C.3) o casos más extremos donde las clases no son linealmente separables como el ejemplo de la figura C.2. La solución para el primer problema es introducir *variables de holgura* (*slack variables* en inglés). La idea principal detrás de esta técnica es *suavizar* la condición del margen de tal forma que éste pueda contener algunos elementos de la muestra como podemos ver en la figura C.3, al mismo tiempo mantenemos el objetivo original de maximizar el margen entre las dos clases.

Las variables de holgura son incorporadas a la formulación del SVM en forma de coeficientes ξ_i de tal forma que un valor de ξ_i distinto de cero permite a la variable x_i no cumplir con el restricción del margen a un coste de penalización proporcional al valor de $C\xi_i$, siendo C un factor de regularización. La nueva formulación del problema de minimización después de incorporar las variables de holgura ξ_i queda:

$$\begin{aligned} &\Theta(\alpha) = \frac{1}{2} w^T w + C \sum_i \xi_i \\ &Sujeto\, a: \\ &y_i(w^T \phi(x_i) + b) >= 1 - \xi_i \end{aligned} \tag{C.10}$$

Así que con la nueva situación tenemos dos objetivos encontrados, cuando aumentamos el margen corremos el riesgo de incorporar a elementos indeseados y aumentar así los errores en la clasificación. Si estrechamos el margen estos elementos quedarán fuera pero como consecuencia perdemos seguridad en el resultado

final de la clasificación. Así que hay que ajustar las variables de holgura para minimizar los errores de clasificación y maximizar el margen. Éste puede ser menor que 1 para una variable x_i eligiendo un valor ξ_i mayor que cero pero esta decisión va acompañada de un factor de penalización de $C\xi_i$. La suma de las variables de holgura ξ_i supone un límite máximo para el número de errores que se pueden cometer durante el entrenamiento. El parámetro C es un factor de regularización que controla el sobre-entrenamiento. Este término básicamente refleja la prioridad que tiene clasificar los valores de entrada de forma correcta frente a maximizar el margen resultante. Un valor grande de C hará que el margen final sea estrecho minimizando el número de muestras que se clasifican de forma incorrecta. De lo contrario, un valor pequeño de C permite al clasificador cometer más errores de clasificación para obtener el máximo margen posible. Normalmente este parámetro se ajusta de forma dinámica durante la fase de entrenamiento hasta encontrar el valor óptimo que maximiza la precisión del clasificador.

Apéndice D

Resultados de clasificación sobre tráfico real

Las siguientes tablas contienen los resultados de clasificación detallados obtenidos con el sistema desarrollado en esta tesis sobre varios vídeos la base de datos propia.

Video	Motocicletas	Coches	Furgonetas	Camiones	Autobuses
video-0042	1	29	4	2	0
resultados	1	29	4	2	0
video-0062	1	127	4	0	0
resultados	1	123	4	0	0
video-0063	2	117	0	4	0
resultados	1	117	0	4	0
video-0064	2	127	0	0	1
resultados	1	127	0	0	1
video-0065	0	236	3	4	2
resultados	2	236	3	4	2
video-0066	1	130	1	5	0
resultados	0	128	1	5	0
video-0067	2	152	4	3	3
resultados	2	156	4	2	4
video-0069	0	82	4	2	0
resultados	1	82	3	2	0
video-0070	1	112	3	6	1
resultados	1	114	3	6	0
video-0071	0	70	0	0	0
resultados	0	70	0	0	0
video-0072	2	116	2	5	0
resultados	2	116	2	5	0
video-0074	0	42	0	1	1
resultados	0	41	0	1	1
video-0075	1	132	1	2	0
resultados	1	131	0	2	1
video-0076	0	112	1	2	0
resultados	0	110	1	2	0
video-0077	1	128	7	1	0
resultados	3	124	5	1	0
video-0079	2	125	6	1	5
resultados	0	124	5	3	3
video-0080	3	96	1	1	0
resultados	2	94	2	1	0
video-0081	1	63	3	2	0
resultados	1	59	3	2	0
video-0082	1	110	2	2	0
resultados	2	110	2	2	0
video-0083	1	64	4	0	0
resultados	1	61	4	0	0

Cuadro D.1: Resultados de clasificación en escenarios con nubes y bajas oclusiones

Video	Motocicletas	Coches	Furgonetas	Camiones	Autobuses
video-0102	14	234	19	1	0
resultados	14	225	23	0	1
video-0006	2	93	2	0	0
resultados	2	93	2	0	0
video-0008	1	14	0	0	0
resultados	1	14	0	0	0
video-0052	0	88	6	3	1
resultados	0	85	4	3	1
video-0050	1	108	9	2	1
resultados	1	107	8	2	0
video-0055	1	120	11	3	0
resultados	1	116	11	3	0
video-0056	1	44	5	5	0
resultados	1	43	5	5	0
video-0059	2	100	9	12	2
resultados	2	98	9	8	3
video-0060	3	96	5	9	1
resultados	3	94	4	9	1
video-0061	0	151	8	11	2
resultados	0	145	10	10	1
video-0047	0	78	7	5	3
resultados	0	75	4	7	3
video-0046	0	63	6	3	1
resultados	0	63	6	3	0
video-0005	1	113	1	1	0
resultados	1	113	1	1	0

Cuadro D.2: Resultados de clasificación en escenarios soleados con bajas oclusiones

Video	Motocicletas	Coches	Furgonetas	Camiones	Autobuses
video-0044	1	66	10	5	1
resultados	1	64	10	5	1
video-0045	1	90	12	1	1
resultados	1	89	10	1	1
video-0047	0	78	7	5	3
resultados	0	75	4	7	3
video-0049	6	89	6	7	3
resultados	6	88	9	7	2
video-0051	1	84	9	9	0
resultados	2	81	9	7	0
video-0053	1	75	10	7	2
resultados	1	73	10	7	2
video-0054	2	50	3	5	1
resultados	0	51	3	3	0
video-0057	1	71	3	4	3
resultados	1	71	3	4	1
video-0058	1	77	6	6	3
resultados	1	77	2	7	3
Video-0048	0	87	11	7	2
resultados	0	87	10	6	1

Cuadro D.3: Resultados de clasificación en escenarios soleados con oclusiones

Bibliografía

A. Elgammal, D. Harwood, and L. Davis (2000). Non-parametric model for background subtraction. *Proc. Eur. Conf. Comput. Vis. Part II*, LNCS 1843:751–767.

Amit Pal, G. S. and Celebi, M. E. (2010). Robust codebook-based video background subtraction. *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference*, pages 1146–1149.

Asaidi, H., Aarab, A., and Bellouki, M. (2014). Shadow elimination and vehicles classification approaches in traffic video surveillance context. *J. Vis. Lang. Comput.*, 25(4):333–345.

Bailing Zhang, Yifan Zhou and Hao PanTammam Tillo (2013). Hybrid model of clustering and kernel autoassociator for reliable vehicle type classification. *Machine Vision and Applications*, page 437–450.

Baker, K.D Sullivan, G.D. (1992). Performance assessment of model-based nacking. *Applications of Computer Vision, Proceedings, 1992., IEEE Workshop on*, pages 28–35.

Bjorn Johansson, Johan Wiklund, Per-Erik Forssén, Gösta Granlund (2009). Combining shadow detection and simulation for estimation of vehicle size and position. *Pattern Recognition Letters*.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA. ACM.

Buch, N., Orwell, J., and Velastin, S. A. (2009). 3d extended histogram of oriented

gradients (3dhog) for classification of road users in urban scenes. In *BMVC*. British Machine Vision Association.

C. Cortes and V. Vapnik (1995). Support-vector network. *Machine Learning*, 20:273–297.

C. Stauffer and W. E. Grimson (1999). Adaptive background mixture models forreal-time tracking,. *Proc. IEEE ICCV*, 2:256–261.

C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland (1997). Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Anal. Mach.Intell.*, 19:780–785.

Chang, Y., Su, Z., and Qian-Yu, L. (2012). A new traffic incident detection method under low-volume condition based on automatic vehicle identification. In *Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on*, pages 2853–2859.

Coifman, B., Beymer, D., McLauchlan, P., Malik, J., and B, J. M. (1998). A real-time computer vision system for vehicle tracking and traffic surveillance.

D. HAN, D. B. COOPER2 and H.-S. HAHN (2013). Bayesian vehicle class recognition using 3-d probe. *International Journal of Automotive Technology*, 14:747–756.

Dalal N. and Triggs B (2005). Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference*, 1:886–893.

Dorin Comaniciu, V. R. and Meer, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575.

Dubska, M., Herout, A., and Sochor, J. (2014). Automatic camera calibration for traffic understanding. *Proceedings of the British Machine Vision Conference*.

Duffner, S. ; LIRIS, Univ. de Lyon, Lyon, France ; Garcia, C. (2013). Pixeltrack: A fast adaptive algorithm for tracking non-rigid objects. *Elseiver, Real time imaging*, pages 2480–2487.

Fabrízia Medeiros de Sousa Matos , Renata Maria Cardoso Rodrigues de Souza (2012). An image vehicle classification method based on edge and pca applied to blocks. *IEEE International Conference on Systems, Man, and Cybernetics*.

G.D. Sullivan, K.D. Baker, A.D. Worral, C.I. Attwood, P.M. Remagnino (1997). Model-based vehicle detection and classification using orthographic approximations. *IVC 15*, 15:649–654.

Guerrero-Gomez-Olmedo, R. and Lopez-Sastre, R. J. and Maldonado-Bascon, S. and Fernandez-Caballero, A. (2013). Vehicle tracking by simultaneous detection and viewpoint estimation. *IWINAC 2013, Part II, LNCS 7931*, pages 306–316.

Guohui Zhang, R. P. A. and Wang, Y. (2007). A video-based vehicle detection and classification system for real-time traffic data collection using uncalibrated video cameras. TRB 2007 Annual Meeting.

Guohui Zhang, Ryan P. Avery, Y. W. (2007). A video-based vehicle detection and classification system for real-time traffic data collection using uncalibrated video cameras. Transportation Research Board of the National Academies:138–147.

H. Asaidi a,n, A. Aarab b, M. Bellouki c (2014). Shadow elimination and vehicles classification approaches in traffic video surveillance context. *Elsevier*.

haoxiang Zhang, Tieniu Tan, Kaiqi Huang, and Yunhong Wan (2012). Three-dimensional deformable-model-based localization and recognition of road vehicles. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 21:1–13.

Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition.

Hilbert, E. E., C. C. G. W. H. G. R. O. M. J. J. A. R. (1978). Wide area detection system. Report - Federal Highway Administration no. FHWA-RD-77-86.

Hoose, N (1989). Queue detection using computer image processing. *Road Traffic Monitoring, 1989., Second International Conference on*, pages 94–98.

J. Hu, T. Su, and S. Jeng (2006). Robust background subtraction with shadow and highlight removal for indoor surveillance. *In Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, pages 4545–4550.

James W. Perry, A. K. a. M. B. (1955). Machine literature searching x. machine language; factors underlying its design and development. *Journal of the association for information science and technology*, 6(4):242–254.

Jin-Cyuan Lai and Shih-Shinh Huang and Chien-Cheng Tseng (2010). Image-based vehicle tracking and classification on the highway. *Green Circuits and Systems (ICGCS), 2010 International Conference on*, pages 666–670.

J.M Blosseville, C. Krafft, F. Lenior, V. Motyka, S. Beucher (1989). New traffic measurement by image processing. *IFAC Control, Computers, Communications in Transportation*, pages 35–42.

K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis (2004). Background modeling and subtraction by codebook construction,. *Proc. ICIP*, 5:3061–3064.

K. Kim, T. H. Chalidabhongse, D. Harwood, L. Davis, (2005). Real-time foreground–background segmentation using codebook model. *Elseiver, Real time imaging*, pages 172–185.

Kachach, R. and Cañas, J. M. (2016). Hybrid three-dimensional and support vector machine approach for automatic vehicle tracking and classification using a single camera. *Journal of Electronic Imaging*, 25(3):033021.

Kanwal Yousaf, Arta Iftikhar, A. J. (2007). Comparative analysis of automatic vehicle classification techniques: A survey. *International Journal of Image, Graphics and Signal Processing*, 4(9):52–59.

Keerthi, S. S. and Lin, C.-J. (2003). Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Comput.*, 15(7):1667–1689.

Kohavi, R. and Provost, F. (1998). Glossary of terms. *Machine Learning*, 30(2-3):271–274.

Koller, D., Weber, J., Malik, J. (1994). Robust multiple car tracking with occlusion reasoning. *European Conference on Computer Vision*, 1:189–196.

Li-Chih Chen a, Jun-Wei Hsieh b,n, Yilin Yan b, Duan-Yu Chen (2015). Vehicle make and model recognition using sparse representation and symmetrical surfs. *Elsevier*.

Liebowitz, D. and Zisserman, A. (1998). Metric rectification for perspective images of planes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 482–488.

Lili Huang, Barth, M. (2010). Real-time multi-vehicle tracking based on feature detection and color probability model. *Intelligent Vehicles Symposium (IV)*, pages 981–986.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, pages 674–679.

Luis Unzueta– Marcos Nieto– Andoni Cortés– Javier Barandiaran– Oihana Otaegui– and Pedro Sánchez (2012). Adaptive multicue background subtraction for robust vehicle counting and classification. 13:527–540.

M. Mazaheri, S. M. (2011). Real time adaptive background estimation and road segmentation for vehicle classification. *IEEE*, pages 1–6.

Mandellos, N. A., Keramitsoglou, I., and Kiranoudis, C. T. (2011). A background subtraction algorithm for detecting and tracking vehicles. *Expert Syst. Appl.*, 38(3):1619–1631.

Marcos Nieto, Luis Unzueta, Javier Barandiaran, Andoni Cortés, Oihana Otaegui and Pedro Sánchez (2011). Vehicle tracking and classification in challenging scenarios via slice sampling. *EURASIP Journal on Advances in Signal Processing.*

Matthew J. Leotta and Joseph L. Mundy (2011). Vehicle surveillance with a generic, adaptive, 3d vehicle model. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 33:1457 – 1469.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme.

Michael Hodlmoser, Branislav Micusik, Marc Pollefeys, Ming-Yu Liu, Martin Kampel (2013). Model-based vehicle pose estimation and tracking in videos using random forests. *International Conference on 3D Vision*, pages 430–437.

Michael Hodlmoser, Branislav Micusik, Ming-Yu Liu, Marc Pollefeys, Martin Kampel (2012). Classification and pose estimation of vehicles in videos by 3d modeling within discrete-continuous optimization. *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference*, pages 198–205.

N. Buch, S.A. Velastin and J. Orwell (2011). A review of computer vision techniques for the analysis of urban traffic. *IEEE Transactions on Intelligent Transportation Systems*, 12:920–939.

Nateghinia, E. and Moradi, H. (2014). Video-based multiple vehicle tracking at intersections. In *Robotics and Mechatronics (ICRoM), 2014 Second RSI/ISM International Conference on*, pages 215–220.

Neeraj K. Kanhere; Stanley T. Birchfield (2010). A taxonomy and analysis of camera calibration methods for traffic monitoring applications. *Intelligent Transportation Systems, IEEE Transactions*, 11:441–452.

Niluthpol Chowdhury Mithun, Nafi Ur Rashid, and S. M. Mahbubur Rahman (2012). Detection and classification of vehicles from video using multiple time-spatial images. *IEEE Transactions on intelligent transportation systems*, 13.

Ninad S. Thakoor, Member, IEEE, and Bir Bhanu (2013). Structural signatures for passenger vehicle classification in video. *IEEE Transactions on intelligent transportation systems*, 14.

Onoe M, Ohba k (1976). Digital analysis of traffic flow. *Proc Int Joint Conf Pattern Recognition*, pages 803–804.

Orhan Bulan and, Robert P. Loce and, W. W. a. Y. W. a. E. A. B. a. Z. F. (2013). Video-based real-time on-street parking occupancy detection system. *J. Electronic Imaging*, 22(4):041109.

Pablo Barcellos, Christiano Bouvié, F. L. E. J. S. (2014). A novel video based system for vehicles at user-defined virtual loops. *Expert Systems with Applications*, 42(4):1845–1856.

Pang, C.C.C. ; Lam, W. . Y. N. (2007). A method for vehicle count in the presence of multiple-vehicle occlusions in traffic images. *Intelligent Transportation Systems*, 8:441–459.

Pang, C.C.C. Lam, W. . Y. N. (2004). A novel method for resolving vehicle occlusion in a monocular traffic-image sequence. *Intelligent Transportation Systems*, 5:129–141.

Papageorgiou, C. and Poggio, T. (2000). A trainable system for object detection. *Int. J. Comput. Vision*, 38(1):15–33.

Papageorgiou, C. P., Oren, M., and Poggio, T. (1998). A general framework for object detection. In *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, pages 555–.

Pearson, K. (1904). Mathematical contributions to the theory of evolution. XII. on a generalised theory of alternative inheritance,with special reference to Mendel's laws. *Philosophical Transactions. Royal Society of London.Series A. Mathematical and Physical Sciences.*, 203:53–86.

Quinlan, J. R. (1993). C4.5: Programs for machine learning. morgan kaufmann publishers, 1993.

Rijsbergen, C. J. V. (1979). *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition.

Robert, K. (2009). Night-time traffic surveillance: A robust framework for multi-vehicle detection, classification and tracking. Advanced Video and Signal Based Surveillance:1–6.

Shi, J. ; Tomasi, C. (1994). Good features to track. *Proc. IEEE CVPR*, pages 593–600.

Shih-Hao Yu, Jun-Wei Hsieh, Y.-S. C. and Hu, W.-F. (2003). An automatic traffic surveillance system for vehicle tracking and classification. Lecture Notes in Computer Science Volume 2749:379–386.

Shirazi, M. S. and Morris, B. T. (2016). Vision-based turning movement monitoring:count, speed amp; waiting time estimation. *IEEE Intelligent Transportation Systems Magazine*, 8(1):23–34.

Sirovich, L., K. M. (1987). Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America*, 4:519–524.

sun Shin, W., heon Song, D., and hun Lee, C. (2009). Vehicle classification by road lane detection and model fitting using a surveillance camera.

T. Chalidabhongse, K. Kim, D. Harwood, and L. Davis (2003). A perturbation method for evaluating background subtraction algorithms. *In Proc. IEEE Joint Int. Workshop VS-PETS*, pages 1–7.

Tao Wang, Zhigang Zhu, C. N. T. (2013). A multimodal temporal panorama approach for moving vehicle detection, reconstruction and classification. Computer Vision and Image Understanding:1724–1735.

Tomasi, C. and Kanade, T. (1991). Detection and tracking of point features. Technical report, International Journal of Computer Vision.

Tomás Rodríguez, Narciso García (2010). An adaptive, real-time, traffic monitoring system. *Machine Vision and Applications*, 21(4):555–576.

Vermeulen, E. (2014). Automatic incident detection (aid) with thermal cameras. *IEEE*, IET:1–5.

Versavel, J. ; Lemaire, F. ; Van der Stede, D. (1989). Camera and computer-aided traffic sensor. *Road Traffic Monitoring, 1989., Second International Conference on*, pages 66–70.

Wang, Y. (2009). Real-time moving vehicle detection with cast shadow removal in video based on conditional random field. *IEEE Trans. Cir. and Sys. for Video Technol.*, 19(3):437–441.

wei Hsu, C., chung Chang, C., and jen Lin, C. (2010). A practical guide to support vector classification.

Wei Wang, Yulong Shang, J. G. Z. Q. (2011). Real-time vehicle classification based on eigenface. Consumer Electronics, Communications and Networks (CECNet):4292–4295.

Xia, Y., Shi, X., Song, G., Geng, Q., and Liu, Y. (2014). Towards improving quality of video-based vehicle counting method for traffic flow estimation. *Signal Process.*, 120(C):672–681.

Yang, Y., Ming, Y., Gang, Y., and Yandong, Z. (2011). Length-based vehicle classification in multi-lane traffic flow. Transactions of Tianjin University, Vol 17:362–368.

Yang Lv, Benjamin Yao, Y. W. and Zhu, S.-C. (2012). Reconfigurable templates for robust vehicle detection and classification. Applications of Computer Vision (WACV):321–328.

Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey.

yves Bouguet, J. (2000a). Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs.*

yves Bouguet, J. (2000b). Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, pages 2480–2487.

Z. Zivkovic and F. van der Heijden (2006). Efficient adaptive density estimapion per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27:773–780.

Zezhi Chen, Tim Ellis (2011). Multi-shape descriptor vehicle classification for urban traffic. *International Conference on Digital Image Computing: Techniques and Applications*, pages 465–461.

Zezhi Chen, Tim Ellis, Sergio A Velastin (2011). Vehicle type categorization: A comparison of classification schemes. *2011 14th International IEEE Conference on Intelligent Transportation Systems*, pages 74–79.

Zezhi Chen, Tim Ellis, Sergio A Velastin (2012). Vehicle detection, tracking and classification in urban traffic. *15th International IEEE Conference on Intelligent Transportation Systems*.

Zhang, B., Zhou, Y., Pan, H., and Tillo, T. (2014). Hybrid model of clustering and kernel autoassociator for reliable vehicle type classification. *Mach. Vision Appl.*, 25(2):437–450.

Zhang, W., Wu, Q. J., and Yin, H. b. (2010). Moving vehicles detection based on adaptive motion histogram. *Digit. Signal Process.*, 20(3):793–805.

Zhou Zhu and Xiaobo Lu (2010). An accurate shadow removal method for vehicle tracking. *Artificial Intelligence and Computational Intelligence (AICI), 2010 International Conference on*, 2:59–62.

Zoran Zivkovic (2004). Improved adaptive gaussian mixture model for background subtraction. *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, 2:28–31.

Álvarez, S., Llorca, D., and Sotelo, M. (2014). Hierarchical camera autocalibration for traffic surveillance systems. *Expert Systems with Applications*, 41(4, Part 1):1532 – 1542.