

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA MULTIMODAL PARA LA
CLASIFICACIÓN DE VEHÍCULOS TERRESTRES



ANDRÉS FELIPE GÓMEZ MORENO
200738962
PABLO JULIÁN HERNÁNDEZ CADENA
200738963

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA
CALI
2013

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA MULTIMODAL PARA LA
CLASIFICACIÓN DE VEHÍCULOS TERRESTRES

ANDRÉS FELIPE GÓMEZ MORENO
PABLO JULIÁN HERNÁNDEZ CADENA

TRABAJO DE GRADO PARA OPTAR AL TÍTULO DE
INGENIERO ELECTRÓNICO

DIRECTOR
EVAL BLADIMIR BACCA CORTÉS, PHD.

UNIVERSIDAD DEL VALLE
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA
CALI
2013

Se aprueba el siguiente Trabajo de Grado
en cumplimiento de los Requisitos
expedidos por la institución para optar al
título de:

INGENIERO ELECTRÓNICO

EVAL BLADIMIR BACCA CORTÉS, PhD.
Director

Jurado 1

Jurado 2

Santiago de Cali, 04 de Enero de 2014

Nota de Aceptación

Dedicado a mi familia por brindarme el apoyo, el soporte, los principios y los valores a lo largo del desarrollo de la empresa que es mi vida. Dedicado especialmente a mis padres por su integridad y por su eterno buscar de la felicidad de sus seres amados.

Pablo Julián Hernández Cadena

*Le dedico este trabajo a mi papá y a mi
mamá... todo lo hago por ustedes,
ustedes son mi ejemplo y mi soporte.*

Andrés Felipe Gómez Moreno

Agradecimientos

Agradecimiento a nuestro director de trabajo de grado, por su propuesta de trabajo de grado, por sus valiosas contribuciones académicas conceptuales y materiales a nuestro conocimiento, y al desarrollo eficaz e integral de este proyecto, finalmente gracias por brindarnos el entusiasmo y la motivación en esta rama del conocimiento, que aprendimos a apreciar, a respetar y a la cual nos sentimos orgullosos y agradecidos de poder contribuir.

Agradecimiento especial a nuestro compañero y amigo Rafael Ospina, por brindarnos a lo largo de nuestro proyecto universitario, sus tan valiosas ideas, experiencias y principios de vida así como académicos, de igual manera las discusiones académicas y sobre el interés general de nuestras vidas que lograron enriquecernos como personas y como estudiantes.

Agradecimiento a los docentes, compañeros y amigos que contribuyeron de alguna forma diversa, a brindarnos, tanto conocimientos, ideas y experiencias, que permitieron la consecución exitosa de este proyecto.

Pablo Julián Hernández Cadena

Quiero agradecer a todas las personas que por largos o cortos períodos de tiempo me han regalado su agradable compañía, pues de todos he aprendido a vivir.

Quiero agradecer a mis profesores por su ardua y valiosísima labor de formar personas de bien, muy especialmente a mi director de trabajo de grado, el profesor Bladimir Bacca.

Quiero agradecer a mis padres, que se rompen el lomo todos los días por mí, no saben cuánto lo aprecio.

Quiero darle un muy tierno agradecimiento a esa mujer, que ha llegado en el momento justo y me ha alegrado los días desde entonces.

Sin embargo, mi más sentido agradecimiento es para Pablo Hernández, mi compañero de trabajo, por ser la rueda y el motor que ha hecho que este proyecto se pueda cumplir. Gracias por ser un buen amigo y por tenerme tanta paciencia.

Andrés Felipe Gómez Moreno

Resumen

Este documento registra el procedimiento teórico y práctico seguido para la implementación de un sistema de clasificación automática de vehículos terrestres que separa diez clases de vehículos diferentes (bicicletas, buses, camiones, camperos, furgonetas, hatchbacks, motos, pickups, sedanes y SUVs). El sistema implementado tiene un enfoque multimodal, al implementar técnicas de clasificación por medio de visión artificial y por medio de información de rango, usando una cámara Canon VC-C50i y un láser Sick LMS200, respectivamente. Sin embargo, el alcance de este trabajo no trasciende hasta la etapa de integración total de estos sistemas independientes, sino que se conciben como sistemas de clasificación autónomos y paralelos con potencial de ser combinados en un solo sistema mucho más robusto.

Los sistemas implementados integran enfoques de clasificación *Top-Down* y *Bottom-Up*. El enfoque *Top-Down* se evidencia al hacer uso de características estructurales de las instancias a clasificar, como la forma y el tamaño, mientras que el enfoque *Bottom-Up* se implementa haciendo uso de características de más bajo nivel basadas en partes, como descriptores tanto visuales como 3D. La tarea de clasificación se basa en el método de Bolsa de Características (Bag of Features) agrupando características locales. Con el fin de obtener las clases miembro se usa el método del K-vecino más cercano combinado con descriptores estructurales de los vehículos a clasificar.

Se presenta todo el marco conceptual necesario para la concepción y el diseño de un sistema de este tipo y se documenta el procedimiento seguido para su implementación. La implementación de este sistema se realizó en lenguaje C++ y mediante la herramienta de análisis computacional *Matlab*.

En este trabajo se realiza una comparación de las tasas de clasificación obtenidas usando los descriptores visuales SIFT, SURF, BRIEF, BRISK, ORB y FREAK para la clasificación visual y los descriptores 3D VFH, ESF, FPFH, SHOT y NARF para la clasificación por información de rango. De esta comparación de desempeño, se obtienen tasas de clasificación promedio ponderadas de entre 44.95% y 77.98% para el módulo de clasificación por información visual y de entre 64.50% y 84.88% para el módulo de clasificación con información de rango.

Adicionalmente, debido a la falta de un dataset integrado basado en información 2D-3D para vehículos terrestres variados, los estudiantes involucrados han elaborado un dataset propio, que cuenta con más de 500 datos. En este trabajo se presenta también el esquema de obtención del mismo basado en el láser y la cámara de video, la distribución de los sensores, las características de la información y los resultados obtenidos.

Abstract

This document presents the theoretical design and implementation work, for autonomously classify vehicles between ten different classes as follows: bikes, buses, trucks, vans, crossovers, SUVs, hatchbacks, motor bikes, pickups and sedan. The classification system implemented is focused on a multimodal strategy, since it implements image based methods and range information based methods. The sensors used in this work are a Canon VC-C50i camera and a SICK LMS200 laser range finder. It is worth noting that the scope of this work does not include the total integration of these two sub-systems, but they are designed and implemented as two independent classification threads. However, future works could include a total integration in order to improve the classification system performance.

The image and range information based classification systems implement both *Top-Down* and *Bottom-Up* methods. *Top-Down* methods use structural features into the classification system such as shapes and physical dimensions. *Bottom-Up* methods use low level local features such as image or 3D descriptors. The classification method implemented is based on the Bag of Features algorithm, grouping together local features. The member classes are obtained using the k-Nearest Neighbor method combined with vehicle's structural features.

This document presents a complete state of the art review in order to design and implement the vehicle classification system. C++ and Matlab were the programming languages used to implement the preprocessing, feature extraction and classification algorithms proposed in this work.

In addition, this work presents a comparison of classification performance between different image and 3D descriptors. The image descriptors used are: SIFT, SURF, BRIEF, BRISK, ORB and FREAK. The 3D descriptors used are: VFH, ESF, FPFH, SHOT and NARF. As a result of this comparison, this work achieve weighted mean classification rates between 44.95% and 77.98% for image based sub-system, and mean classification rates between 64.5% and 84.88% for 3D information based sub-system.

Finally, nowadays there is a lack of public datasets including 2D and 3D data information. For this reason, this work included building a dataset with more than 500 data (image and range information). This document presents the information concerning to the acquisition methodology such as: sensors used, sensor calibration and distribution, data format, preprocessing techniques used and obtained results.

Glosario

ADC: Analog to Digital Converter (Convertidor análogo – digital)

ARBOL KD: Estructura de organización de la información.

AVC: Automatic Vehicle Classification (Clasificación automática de vehículos)

BLOB: Región en una imagen digital en la que algunas propiedades son constantes o varían en un rango de valores preescrito.

BoF: Bag of Features (Bolsa de características)

BoW: Bag of Word (Bolsa de palabras)

BRIEF: Binary Robust Independent Elementary Features

BRISK: Binary Robust Invariant Scalable Keypoints

CENTROIDE: Centro geométrico. Posición promedio de todos los puntos en todas las coordenadas geométricas de un objeto (en un espacio n-dimensional)

CLÚSTER: agrupación o subconjunto de una nube de puntos mayor.

CONTENEDOR: Refiere al espaciamiento discreto, de la información en los descriptores.

DATASET: Colección de datos normalmente tabulada. Por cada elemento se indican varias características.

DoF: Degrees of Freedom (Grados de libertad)

DoG: Difference of Gaussians (Diferencia de gaussianas)

ERI: Enhanced Range Image (Imagen de rango mejorada)

ESF: Ensemble of Shape Functions

FAST: Features from Accelerated Segment Test

FLANN: Fast Library Approximate Nearest Neighbors

FPFH: Fast Point Feature Histograms

FREAK: Fast Retina Keypoint

GLOBAL: Referente a la totalidad de un elemento.

GMM: Gaussian Mixture Model (Modelo de mezclas de gaussianas)

HDPs: Hierarchical Dirichlet Processes

HOGs: Histogram of Oriented Gradients (Histograma de gradientes orientados)

ICP: Iterative Closest Point

K-NN: k-Nearest Neighbor. Elementos más cercanos en una vecindad multidimensional

LDA: Latent Dirichlet Allocation

LOCAL: Referente a subconjuntos de un elemento.

LQG: Linear Quadratic Gaussian

MARCO DE REFERENCIA: Referencia de medida del origen de cada dimensión.

MÓDULO: Cada uno de los subcomponentes de un sistema granular.

MoG: Mixture of Gaussians (Mezcla de gaussianas)

MST: Minimum Spanning Tree

MULTIMODAL: En estadística, distribución de probabilidad que presenta dos o más modas. En clasificación de patrones, sistema que integra varios modos de percepción.

NARF: Normal Aligned Radial Features

NORMAL: Vector perpendicular al plano tangente a una vecindad geométrica.

NUBE DE PUNTOS 3D: Estructura que contiene todos los puntos (x,y,z) de una captura de láser.

OEa: Organización de Estados Americanos.

ORB: Oriented FAST and Rotated BRIEF

PCA: Principal Component Analysis (Análisis de componentes principales)

PCL: Point Cloud Library

PDF: Probability Density Function (Función de densidad de probabilidad)

pLSA: probabilistic Latent Semantic Analysis

RANSAC: Random Sample Consensus

RBNN: Radially Bounded Nearest Neighborgraph

ROI: Region Of Interest (Región de interés)

SHOT: Signature of Histograms of Orientations

SIFT: Scale Invariant Feature Transform

SIT: Sistemas Inteligentes de Transporte.

SLAM: Simultaneous Localization And Mapping (Localización y mapeo simultáneos)

SPMODE: Symmetries and Perturbations Mode

SURF: Speeded-Up Robust Features

SUV: Sport Utility Vehicle

SVD: Singular Value Decomposition (Descomposición del valor singular)

SVM: Support Vector Machine (Máquinas de soporte vectorial)

TIC: Tecnologías de la Información y la Comunicación.

VECINDAD: subregión multidimensional, con la menor distancia a un ente de su misma dimensionalidad.

VFH: Viewpoint Feature Histograms

Contenido

1	Introducción.....	14
1.1	Descripción del problema de investigación	14
1.1.1	Justificación	14
1.1.2	Planteamiento del problema	15
1.2	Objetivos	17
1.2.1	Objetivo general	17
1.2.2	Objetivos específicos.....	17
1.3	Metodología.....	17
1.3.1	Esquema de desarrollo y realización del trabajo.....	17
1.3.2	Enfoque de la investigación.....	17
2	Marco teórico	18
2.1	Introducción.....	18
2.2	Estado del arte y antecedentes	19
2.2.1	Clasificación de objetos usando visión artificial	20
2.2.2	Estimación de velocidad mediante técnicas de análisis visual.....	25
2.2.3	Clasificación de objetos usando información de rango	32
2.2.4	Clasificación de patrones.....	40
2.3	Sensores y dispositivos.....	43
2.4	Librerías y software	44
3	Desarrollo de ingeniería	46
3.1	Elaboración del dataset.....	46
3.1.1	Descripción del esquema y montaje	46
3.1.2	Limitaciones y condiciones	48
3.1.3	Estructura de directorios.....	49
3.2	Calibración de la cámara Canon VC-C50i con OpenCV y Matlab	50
3.3	Calibración del sensor láser Sick LMS200 en Matlab y SickToolbox	52
3.3.1	Alineación eje central	52
3.3.2	Modelo lineal del sensor laser	52
3.4	Diseño y desarrollo del módulo de extracción de características y clasificación de vehículos con información visual.....	53
3.4.1	Selección de la región de interés (ROI) en un fotograma	55
3.4.2	Segmentación, extracción del foreground y formación de blobs en un fotograma	56
3.4.3	Detección y filtrado de contornos representativos	58
3.4.4	Proceso de seguimiento mediante flujo óptico y cálculo de la velocidad estimada del vehículo	59
3.4.5	Alisamiento offline mediante filtro de Kalman	61
3.4.6	Obtención de puntos clave en la región de interés y cálculo de descriptores y características visuales	63
3.4.7	Formación de vector de características y clasificación	67
3.5	Diseño y desarrollo del módulo de clasificación basado en información de rango	69

3.5.1	Estimación de la velocidad del vehículo y filtrado de Kalman.....	70
3.5.2	Rectificación de la información de rango a través del modelo calibrado y proceso de registro de la escena 3D	71
3.5.3	Corrección de ángulo de inclinación	72
3.5.4	Segmentación y extracción de la región de interés en la información de rango	74
3.5.5	Obtención de los puntos clave	79
3.5.6	Estimación de normales de superficie	82
3.5.7	Extracción de descriptores 3D en la nube de puntos de la información de rango	84
4	Pruebas y análisis de resultados.....	95
4.1	Introducción.....	95
4.2	Esquema, descripción y desarrollo de las pruebas.....	95
4.2.1	Módulo clasificación de vehículos con información visual	95
4.2.2	Modulo de información de rango.....	98
4.3	Resultados y análisis.....	103
4.3.1	Módulo clasificación de vehículos con información visual	103
4.3.2	Módulo clasificación de vehículos con información de rango.....	109
5	Conclusiones	117
5.1	Trabajos futuros.....	118
6	Referencias	119
7	Anexos.....	128
7.1	Módulo de clasificación de vehículos con información visual	128
7.1.1	Descriptores visuales	128
7.1.2	Árboles de resultados.....	134

1 INTRODUCCIÓN

En la actualidad el auge de la robótica y la necesidad de implementación de sistemas inteligentes han hecho que las investigaciones se enfoquen hacia el desarrollo de técnicas que permitan potencializar la capacidad de percepción y generar un alto nivel de discernimiento ante las características dinámicas de los entornos de trabajo. Dentro de las ramas de investigación relacionadas con los objetivos mencionados, se encuentran el modelamiento dinámico de ambientes y la visión computacional. La primera es aplicada generalmente en la navegación topológica de los robots en entornos desconocidos, en la localización y mapeo simultáneo SLAM (Hartmann & Forouher, 2012), en el modelamiento de objetos en dos o tres dimensiones (García & Prieto, 2011), análisis estructural de proyectos de construcción (Guarnieri, Pirotti, Pontin, & Vettore, 2006), e incluso en la interacción presente en los videojuegos de última generación. Por su parte, la rama de la visión computacional es más extensa en lo que se refiere a las aplicaciones y criterios de análisis. Se hace uso de ella en gran cantidad de elementos como son: las interfaces humano máquina, sistemas de seguridad y acceso, reconstrucción mediante el proceso de *mosaicing* de imágenes (Elibol, Gracias, Garcia, Gleason, & Gintert, 2011), y monitoreo satelital de diversa índole, etc.

Un nuevo enfoque hacia el que las investigaciones están dirigiendo sus esfuerzos es la integración multimodal de técnicas de percepción y análisis para generar sistemas más robustos. Un caso particular es la integración de las técnicas de análisis mediante visión artificial con las técnicas de análisis de información de rango. Estos sistemas son diseñados para que puedan aprovechar las virtudes de cada uno de los componentes y así mismo logren suplir las debilidades de cada técnica. Generalmente se configuran este tipo de sistemas para lograr que cada componente sea más fuerte cuando el otro presenta inconvenientes significativos, así pues la oclusión característica en las técnicas visuales puede ser superada mediante el uso de técnicas análisis de información de rango. Del mismo modo, ante limitaciones de modelamiento del ambiente por problemas con los índices de reflectividad, o discontinuidades bruscas en el entorno, se puede hacer uso de la información visual para obtener mejores desempeños.

Técnicas de inteligencia computacional pueden ser aplicadas en situaciones cotidianas como la regulación del transporte urbano, debido a la gran potencialidad de sus características y sus resultados satisfactorios. Todo esto con el objetivo de lograr un sistema planificado eficiente y dinámico ante las características en tiempo real del tráfico. La comercialización de vehículos terrestres generalmente progresa más rápido que la infraestructura vial de un país o región, por lo que no hay una regulación activa y se generan caos, retrasos, accidentes, deterioro de las vías, etc. Es por esto que se plantean técnicas como la inteligencia de enjambres para el control de tráfico en autopistas, monitoreo por visión computacional con fines de seguridad y estadísticas de tráfico, clasificación de vehículos para el cobro de tarifas y habilitación de vías auxiliares, asistencia de manejo en condiciones climáticas difíciles o para personas en situación discapacidades, etc. (N. E. Buch, 2010), (Hsieh, Yu, Chen, & Hu, 2006).

1.1 DESCRIPCIÓN DEL PROBLEMA DE INVESTIGACIÓN

1.1.1 JUSTIFICACIÓN

Los sistemas y las estrategias de identificación y clasificación de vehículos surgen como uno de las vertientes o áreas de trabajo de un macro-conjunto de técnicas y arquitecturas denominadas con el rótulo "Sistemas Inteligentes de Transporte (SIT)"; el enfoque que se brinda a este tipo de soluciones ingenieriles de carácter informático y electrónico se orienta

hacia el mejoramiento y la optimización del desempeño de las diversas actividades que componen un sistema de transporte en general. Los SIT abarcan en su denominación, cualquier tipo de transporte, ya sea aéreo, náutico o terrestre, sin embargo su aplicación se ha profundizado o fortalecido en este último.

Los sistemas SIT han sido establecidos para diseñar y aplicar soluciones que permitan realizar una gestión y administración robusta del transporte, principalmente urbano, con tendencias de crecimiento abruptas en la actualidad, esto debido a la globalización y a la alta competencia entre empresas y modalidades de medios de transporte, lo que conduce a la reducción de costos de adquisición de vehículos en general. Surge entonces la necesidad de generar esquemas que puedan controlar situaciones presentes en este tipo de ambiente, algunas de estas son: fluidez mediante habilitación de rutas o activación óptima de semáforos en intersecciones según el tipo de tráfico presente, tarificación automática en peajes que evite largas filas de espera o por extensión el alargue en los tiempos de viaje, estudios estadísticos para la apropiada gestión de mantenimiento de las vías, aplicando los recursos de manera óptima, monitoreo de vacantes en parqueaderos, seguridad vial, etc. Colombia no es ajena a estos incrementos significativos en las ventas de este sector, presentándose en septiembre de 2012 un incremento del 22% en las ventas con respecto al mismo mes en el año 2011, calculándose el cierre el año con aproximadamente 310 mil unidades vendidas ("Adelanto Boletín de Prensa Comité Automotor Colombiano," 2012), por lo que se observa que la concepción e implementación de estos sistemas es requerida también en el contexto nacional.

El área de enfoque en nuestro trabajo de grado corresponde a la identificación y clasificación de vehículos mediante técnicas de información de rango y visión artificial, lográndose así una aplicación multimodal que permita lograr un proceso de complementariedad y finalmente una mayor robustez. Como se explicó anteriormente para esta área su principal aplicación se lleva a cabo en los prototipos de sistemas de tarificación (peajes) automática, los cuales incluso podrían llegar a integrarse con una base de datos que reciba como entrada la identificación (tipo y matrícula) para realizarse el cobro directamente en una cuenta o perfil específico del usuario. También se mencionó para esta rama de aplicación la habilitación de rutas automática y la gestión de intersecciones viales.

En Colombia el flujo de transporte intermunicipal aumenta abruptamente en las fechas festivas de la nación, lo que genera lentitud e ineficiencia en el desempeño del tráfico en las denominadas operaciones retorno, por lo que un sistema que ayude a aliviar la congestión vehicular y los puntos de cuello de botella mediante técnicas de reconocimiento automático para tarificación, presentaría una solución muy apropiada e interesante.

Como factor agregado para la justificación del proyecto, se destaca el proyecto conjunto que se encuentra realizando el grupo de percepción y sistemas inteligentes de la universidad del valle PSI, para el diseño e implementación de estrategias electrónicas e informáticas bajo el esquema de trabajo de los SIT.

1.1.2 PLANTEAMIENTO DEL PROBLEMA

De acuerdo con el artículo 2 de la ley 105 de 1993 "por la cual se dictan disposiciones básicas sobre el transporte, se redistribuyen competencias y recursos entre la Nación y las Entidades Territoriales, se reglamenta la planeación en el sector de transporte y se dictan otras disposiciones", se establece en el literal b): "De la intervención del Estado: corresponde al estado la planeación, el control, la regulación y la vigilancia del transporte y de las actividades a él vinculadas". ("Ley 105 de 1993 de la República de Colombia", 1993)

La ley 1450 de 2011, en el artículo 84, les dio una base jurídica a los sistemas inteligentes de tránsito y transporte – SIT, definiéndolos como un conjunto de soluciones tecnológicas informáticas y de telecomunicaciones que recolectan, almacenan, procesan y distribuyen información, y que deben ser diseñados para mejorar la operación, la gestión y la seguridad del transporte y el tránsito. (“Ley 1450 de 2011 de la República de Colombia”, 2011)

En Colombia, las restricciones de tránsito en la malla vial (restricciones de circulación en carreteras y puentes), al igual que el cobro de peajes en las vías, se viene haciendo de la manera tradicional. Esto implica técnicas invasivas de sensado como la detección dinámica de carga en las vías, al igual que la necesidad de detener el flujo vehicular y generar tráfico en las mismas mediante el sistema Stop-and-Go utilizado para el cobro de peajes. Además de esto, el tránsito vehicular a través de vías y puentes en el territorio nacional no se restringe de acuerdo al tipo de vehículo, lo que influye de manera negativa en la vida útil de estas estructuras, haciendo más costoso e ineficiente el mantenimiento y reparación de las mismas. Todo esto es cobijado en el tema de la política de mantenimiento vial, área en la cual el país tiene un claro atraso, tanto en materia de desarrollos viales como en cuestión de infraestructura. Es por esto que se propone la aplicación de las TIC (Tecnologías de la Información y la Comunicación), en el área de la gestión de transportes en general, y en el monitoreo y control del transporte en las áreas urbanas en particular. Para tener una visión clara y objetiva del problema, sería válido citar el caso de la carretera panamericana, sistema colectivo de carreteras interconectadas que vincula a casi todos los países del continente americano, desde Alaska hasta Buenos Aires. Colombia, a pesar de haber firmado un compromiso en la 5ª Conferencia Internacional de la OEA en 1923, presenta un atraso de varias décadas en materia de desarrollo de esta vía, lo que da cuenta de la situación actual, una deficiente administración en cuestión de mantenimiento vial y una urgente necesidad de innovación en la materia.

Por otro lado, a partir de la puesta en marcha de la ley 1450 de 2011, se pretende modernizar el sistema vial colombiano, mediante el uso de las TIC. Uno de los puntos álgidos en este plan de modernización será el cobro eficiente de la tarifa de peajes que, de manera ideal, debe permitir el pago sin necesidad de detener o disminuir la velocidad de tránsito por la vía. Esto implica, entre otras cosas, la correcta clasificación de los vehículos dentro del rango de categorías que dicta el marco legal colombiano. En Colombia ya se viene implementando un sistema de cobro de peajes electrónico hace poco más de un año denominado *flypass* (“Cobro electrónico de peajes sigue creciendo en Colombia — Noticias Telemedellín,” n.d.), que permite la identificación electrónica de vehículos mediante el uso de un *TAG* o *transponder*, es decir, un dispositivo electrónico que se instala en el vehículo, el cual recibe y envía información, y un lector fijo en la vía. Esta aproximación puede ser, sin embargo, costosa y de difícil implementación, debido a la necesidad de instalación de estos dispositivos electrónicos en todos los vehículos que quieran ser incluidos en el sistema. Con respecto a la restricción de tránsito por las vías de acuerdo al tipo de vehículo, todavía no se implementa ningún tipo de sistema automatizado que permita llevar a cabo esta tarea, de modo que se hace necesaria la implementación de esta tecnología.

Es por esto que se busca el desarrollo de sistemas más simples, no invasivos y que permitan una implementación y una capacitación más sencilla desde el punto de vista tanto del usuario final, es decir, del conductor, ya que todo el proceso sería transparente para él; como del comprador del equipo, es decir, el encargado de implementar el sistema de AVC en las vías, que puede ser el Estado o un particular contratado por este. Uno de los puntos centrales de estos sistemas es la clasificación automática de vehículos. ¿Cómo es posible implementar un sistema de clasificación automática de vehículos usando la tecnología disponible y que se ajuste al contexto colombiano?

1.2 OBJETIVOS

1.2.1 OBJETIVO GENERAL

Diseñar e implementar un sistema de Clasificación Automática de Vehículos (AVC) terrestres usando imágenes de rango y descriptores visuales.

1.2.2 OBJETIVOS ESPECÍFICOS

- Realizar una revisión bibliográfica del estado del arte en técnicas de AVC, con el fin de seleccionar aquellas que más se adecuen al contexto del presente trabajo.
- Elaborar un *dataset* que incluya imágenes de rango, imágenes visuales y datos reales de clasificación de vehículos.
- Diseñar e implementar una técnica de clasificación que use principalmente las imágenes de rango y también la información visual contenidas en el *dataset* elaborado.
- Validar el sistema desarrollado con el *dataset* elaborado y presentar los resultados obtenidos.

1.3 METODOLOGÍA

1.3.1 ESQUEMA DE DESARROLLO Y REALIZACIÓN DEL TRABAJO

Una vez recopilada la información de estado del arte necesaria para iniciar la operación de desarrollo de nuestro propio sistema, se procedió a la toma de datos con el objetivo de disponer de un marco de trabajo concreto en el cual basarse para el diseño del mismo.

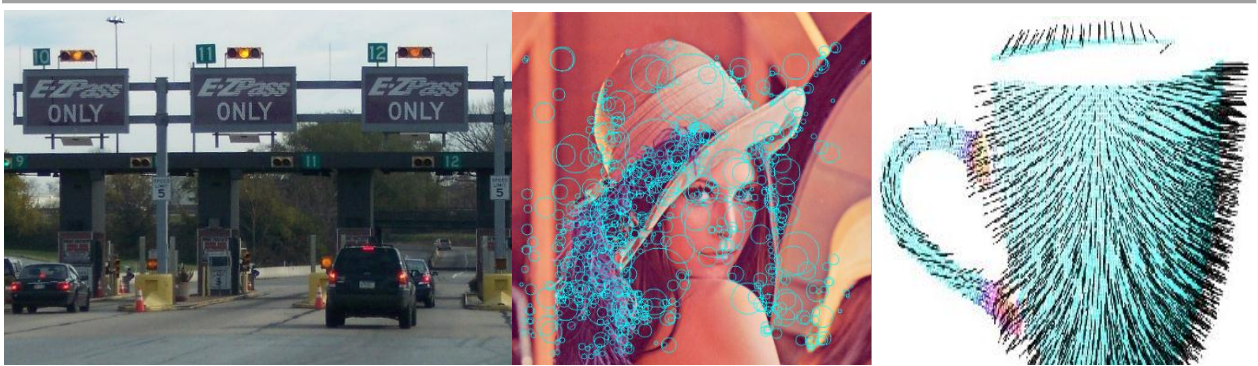
El *dataset* fue construido a partir de los datos obtenidos de un láser Sick LMS200 ubicado a un lado y perpendicular a la vía de tránsito de los vehículos observados, y de una cámara Canon VC-C50i situada en el lado opuesto de la vía y perpendicular a esta. Los datos obtenidos de esta manera son almacenados en una estructura de directorios particular, que permite una correcta y sencilla clasificación y manejo de los mismos.

Una vez adquirida una cantidad suficiente de datos para trabajar (aún sin disponer del *dataset* completo), se inicia la etapa de programación de los algoritmos para realizar la clasificación de vehículos. Estos algoritmos incluyen, entre otros, la estimación de la velocidad de tránsito de los vehículos, el filtrado (suavizado) de la velocidad, la segmentación y clusterización de las nubes de puntos, el filtrado de las nubes de puntos, la obtención de características 3D y visuales, el entrenamiento del clasificador y el algoritmo final de clasificación para la información visual y 3D.

Por último, se realiza una etapa de pruebas para evaluar el desempeño del sistema en la tarea de clasificación de vehículos terrestres. Se reportan los resultados obtenidos y se consignan las conclusiones obtenidas del proceso de concepción, diseño y pruebas del sistema multimodal de clasificación de vehículos terrestres.

1.3.2 ENFOQUE DE LA INVESTIGACIÓN

La presente investigación tiene un enfoque teórico-práctico, en el que se hace una búsqueda de técnicas que permitan la consecución de los objetivos planteados y, posteriormente, se llevan a la praxis los conocimientos adquiridos con el fin de lograr un comportamiento específico deseado. Comportamiento que será evaluado y cotejado para su uso práctico.



2 MARCO TEÓRICO

2.1 INTRODUCCIÓN

A continuación se da una breve revisión de las tendencias presentes en la bibliografía que cobijan el campo de la clasificación automática de vehículos (*Automatic Vehicle Classification –AVC–*). En este campo hay múltiples aproximaciones, tecnologías y técnicas que permiten llevar a cabo la tarea. Existen dos grandes grupos, en los cuales pueden ser clasificadas las técnicas de AVC: Técnicas invasivas y técnicas no invasivas.

Las técnicas invasivas de AVC requieren de la intrusión de un instrumento de medición ya sea en la vía, en el vehículo o ambos. Las principales técnicas invasivas que se pueden encontrar en la literatura incluyen los lazos de inducción o *loops*, que requieren de la instalación de un lazo conductor eléctrico debajo del pavimento y usan el perfil magnético del vehículo (Tok, 2008); la detección dinámica de peso, en la que se utilizan sensores piezoeléctricos instalados debajo del pavimento (Refai, 2009); la utilización de tubos neumáticos o la instalación de dispositivos en el pavimento que se comunican con TAGs electrónicos ubicados debajo de los automóviles.

Por otro lado están las técnicas no invasivas de clasificación, entre las que se incluyen sistemas de visión artificial, que incluyen múltiples métodos que se basan en la detección y cómputo de altura, anchura, longitud, número de vagones y en algunos casos diversos descriptores de imagen (Ng & Tay, 2011); técnicas basadas en ondas sonoras, es decir, basar la clasificación en los ruidos acústicos radiados por vehículos en movimiento, de los cuales se extraen vectores de características usados para distinguir el tipo de vehículo (Ji, Qi, Han, & Yuan, 2009); sistemas de radar Doppler, que realizan cálculos de longitud y velocidad (Bullimore and Hutchinson, 1996); técnicas basadas en sensores magnéticos y técnicas que utilizan imágenes de intensidad de láser o imágenes de rango (Abdelbaki, Hussain, Member, & Gelenbe, 2001).

Basados en la cantidad y tendencia de las publicaciones que tratan al respecto, se puede afirmar que los desarrollos recientes en sistemas de AVC se centran en técnicas de clasificación basadas en imágenes visuales y en información de rango. El uso de cámaras se ha vuelto popular debido a su bajo costo, fácil implementación y gran potencial, contando con desarrollos que permiten la detección y clasificación, con tasas de éxito satisfactorias, de objetos sin características de textura importantes, tal como los autos (D.G. Lowe, 1999) (David G. Lowe, 2004) (Ng & Tay, 2011). Por otro lado, el uso de sensores láser también ha tenido gran acogida en el campo de la AVC, debido a la gran precisión de las técnicas basadas en estos datos. Sistemas de clasificación con sensores láser ya han sido

implementados y probados, presentándose como un caso de éxito (Scholz & Koy-Oberthür, 2006).

En estos dos campos, existen diversas técnicas que permiten la clasificación de vehículos. En el campo de la visión es usual el uso de la longitud, ancho y altura del vehículo para realizar la clasificación. En algunos casos más sofisticados, son usados los *keypoints*, con los cuales se forman vectores que describen los objetos a ser identificados. En (Ng & Tay, 2011) se propone una aproximación que usa un descriptor de imagen similar al ADN humano, formado a partir de las características extraídas mediante *Scale-Invariant Feature Transform* (SIFT) (D.G. Lowe, 1999; David G. Lowe, 2004). En (Grimson & Ma, 2005) se muestra una modificación al algoritmo de detección de los *keypoints* para ser adaptado a la tarea de clasificación de vehículos. También en el uso de sensores láser es usual usar las características básicas de longitud, ancho y altura para clasificar vehículos (Abdelbaki et al., 2001). Sin embargo, debido a la información entregada por un sensor láser (de intensidad y de rango) y a la posibilidad de tener un perfil 3D del vehículo, otras características importantes pueden tenerse en cuenta, como por ejemplo el número de ejes, el número de vagones y la forma de los mismos (Harlow & Peng, 2001). Uno de los métodos utilizados para la generación del perfil 3D de un objeto es el cálculo de los vectores normales a la superficie, información que es obtenida a partir de una nube de puntos generada por el sensor. Esta nube de puntos representa la información en bruto obtenida y constituye los datos básicos sobre los que se debe trabajar en las distintas técnicas de procesamiento a partir de información de rango. El uso de las normales se ha hecho popular debido a su bajo costo computacional, a la vez que se conserva una alta robustez con respecto a discontinuidades en los datos de rango en bruto (Hinterstoisser et al., 2011).

Los sensores de rango poseen una característica especial que los hace una opción particularmente atractiva a emplear en sistemas de AVC y que consiste en la posibilidad de la obtención directa de un modelo 3D del objeto. Esta característica le ofrece una robustez extra a cualquier sistema de clasificación que haga uso de información de rango.

2.2 ESTADO DEL ARTE Y ANTECEDENTES

Se hará un recorrido teórico por dos grandes subcampos en el tema de la clasificación automática de objetos. El primero consiste en la clasificación por medio de técnicas de visión artificial, el cual es abordado en la sección 2.2.1. También mediante visión artificial se aborda un problema que, a pesar de no ser clasificación como tal, sí es usado con tal fin en este trabajo, y es el problema de estimación de velocidad de un objeto en una secuencia de video, tema que es abordado en la sección 2.2.2. El segundo subcampo consiste en la clasificación haciendo uso de información de rango, lo cual se cubre en la sección 2.2.3. Por último, en la sección 2.2.4 se exponen las bases teóricas de la clasificación y reconocimiento de patrones, al igual que los clasificadores.

La **Figura 2.1** muestra un diagrama conceptual de los ejes temáticos abordados en el presente trabajo de grado. La revisión de la bibliografía relacionada fue un requisito crucial para la correcta concepción y desarrollo del sistema de clasificación automática de vehículos descrito.

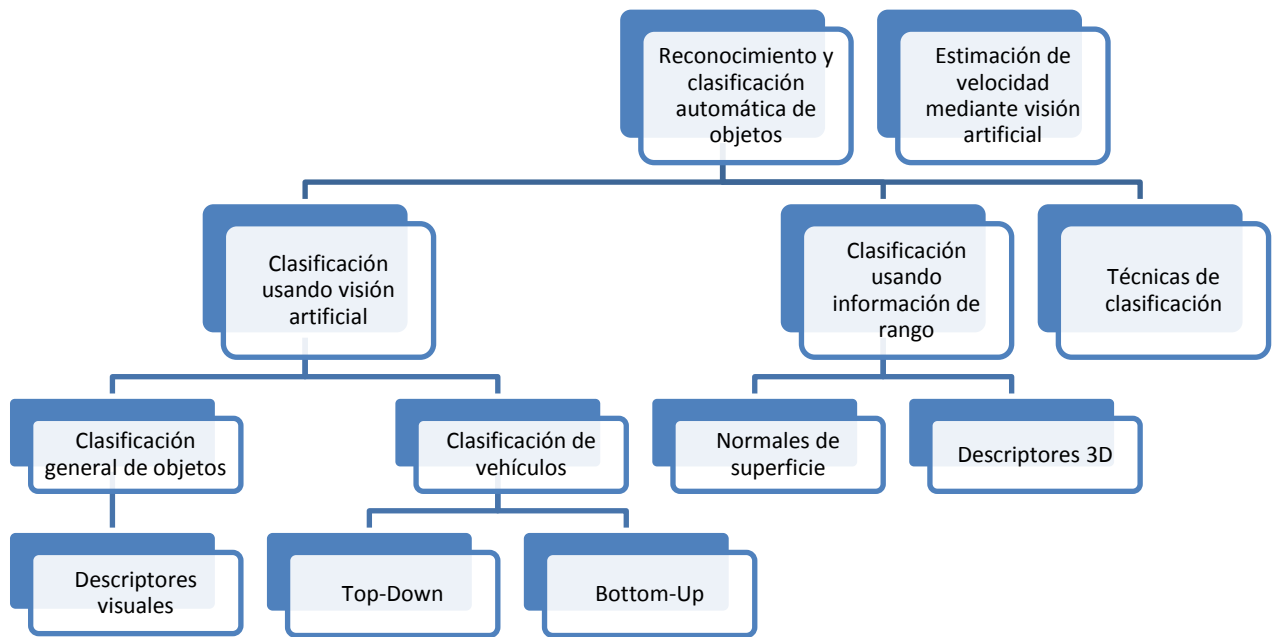


Figura 2.1 Diagrama de flujo de los ejes temáticos abordados en el trabajo

2.2.1 CLASIFICACIÓN DE OBJETOS USANDO VISIÓN ARTIFICIAL

La visión artificial, ha experimentado en los últimos años un vertiginoso aumento de su uso en el área de la clasificación de objetos. A continuación se hace una revisión bibliográfica de las técnicas y enfoques más usados en este campo.

2.2.1.1 Clasificación general de objetos

Las técnicas generales de clasificación de imágenes pueden apuntar a solucionar dos tipos diferentes de problemas (Kamiya, Takahashi, Ide, & Murase, 2010):

- Clasificar imágenes con los objetos de interés ocupando la mayor parte de la imagen, y las escalas de los objetos son similares (e.g. *Dataset Caltech 101/256*)
- Clasificar imágenes con los objetos de interés ocupando áreas parciales en la imagen, y las escalas de los objetos pueden diferir (e.g. *Graz, PASCAL*)

Para la tarea de reconocimiento de categorías semánticas con varios tipos de apariencias, como Carro, Silla o Perro, bajo cambios en el ambiente como dirección de los objetos, distancia de los objetos, iluminación o fondos, se emplean aproximaciones basadas en partes. En (Kamiya et al., 2010) se exponen dos métodos bien conocidos en este ámbito, los cuales son un esquema que usa un *Bag of Features (BoF)* y el *Fergus's Constellation Model*.

BoF es una analogía al modelo "*Bag of Words*" propuesto originalmente en el campo de procesamiento del lenguaje natural. *BoF* es usualmente utilizado con clasificadores como *SVM (Support Vector Machines)* y métodos de análisis de documentos como el *probabilistic Latent Semantic Analysis (pLSA)*, *Latent Dirichlet Allocation (LDA)* y *Hierarchical Dirichlet Processes (HDPs)*.

Se propone también un *Fergus's Constellation Model* modificado. El modelo propuesto apunta a solucionar imágenes del primer tipo, es decir, con los objetos de interés ocupando la mayor parte de la imagen. Mediante el modelo propuesto se obtienen tasas de

clasificación de $99.5\% \pm 0.10\%$ para la base de Datos *Caltech* y de $38.8\% \pm 1.00\%$ para el *PASCAL Visual Object Classes Challenge 2006*.

En (Azhar & Amer, 2012) se hace uso de descriptores visuales MPEG-7. Estos descriptores cubren las categorías básicas de características visuales, como son: color, textura, forma y movimiento. En el artículo se propone el uso de estos descriptores visuales, en conjunto con series caóticas para llevar a cabo la clasificación en 4 clases genéricas (*has_person*, *has_group_of_persons*, *has_vehicle* and *has_unknown*).

En (Hinterstoisser et al., 2012) se describe un sistema de detección de objetos sin textura mediante *template matching*, usando un método de extracción de gradientes en la imagen. Se consideran los gradientes de imagen porque se ha probado que son más discriminantes que otras formas de representación, y de igual manera son robustos ante cambios de iluminación y ruido. Además, los gradientes de imagen son a menudo los únicos descriptores de una imagen, cuando se trata de objetos sin textura, sobre los cuales es mucho más difícil obtener características *SIFT* o similares.

En el campo de la clasificación general de objetos, por medio de técnicas de visión artificial, se hace amplio uso de características de bajo nivel (parches, pixels e incluso subpixels) presentes en la imagen llamadas comúnmente *features* o *keypoints*. Es usual también, construir un descriptor que permita darle a las características una condición de unicidad, de manera que sea posible identificar una característica específica en distintas imágenes que la contengan y relacionarlas entre sí. La **Tabla 2.1** muestra un compendio de algunas de las características visuales más conocidas y utilizadas.

Tabla 2.1 Técnicas de extracción de puntos característicos y descriptores visuales.

Trabajo	Detección de puntos clave	Cálculo de descriptores	Tipo de descriptor	Tamaño del descriptor	Invariante ante escala	Invariante ante rotación
SIFT	Diferencia de gaussianas	Histogramas de gradientes alrededor del punto clave	Flotante	128 componentes	Sí	Sí
SURF	Determinante de la matriz Hessiana	Contenido de intensidad alrededor del punto clave	Flotante	64 componentes	Sí	Sí
FAST	Evaluación de un círculo de Bresenham alrededor de un pixel	-----	-----	-----	-----	-----
BRIEF	-----	Test binarios alrededor del punto clave, en patrones aleatorios	Binario	128, 256, 512 bits	No	No
BRISK	Supresión de no máximos basada en puntuación FAST	Test binarios alrededor del punto clave, en un patrón circular concéntrico	Binario	512 bits	Sí	Sí
ORB	Versión modificada de FAST con componente de orientación	Versión modificada de BRIEF. Test binarios rotados con menor correlación	Binario	256 bits	Sí	Sí
FREAK	-----	Test binarios alrededor del punto clave, en un patrón basado en la retina humana	Binario	512 bits	Sí	Sí

Los puntos clave *SIFT* –*Scale Invariant Feature Transform*- (D.G. Lowe, 1999; David G. Lowe, 2004) son obtenidos como mínimos y máximo locales en las imágenes resultantes del método de *DoG* (*Difference of Gaussians*, diferencia de gaussianas) para distintas escalas. A cada punto clave se le asigna un descriptor que se construye a partir de histogramas de las magnitudes y orientaciones de los gradientes en un parche alrededor del punto. La **Figura 2.2(a)** muestra un ejemplo de reconocimiento de objetos y su pose, bajo condiciones de oclusión, usando *SIFT*.

SURF –*Speeded-Up Robust Features*- (Bay, Ess, Tuytelaars, & Gool, 2008) está basado en

SIFT, pero con diferencias importantes, enfocadas en la velocidad de cómputo. Los puntos clave se obtienen mediante una supresión de no máximos en una vecindad de $3 \times 3 \times 3$ y haciendo uso del determinante de la matriz hessiana. El descriptor tipo *SURF* describe las respuestas Haar Wavelet en las direcciones x y y alrededor del punto. Para hacer el cálculo de los determinantes de la matriz hessiana y de las respuestas Haar Wavelet de manera eficiente se utiliza un concepto conocido como imagen integral. La **Figura 2.2(b)** muestra un ejemplo de puntos clave SURF y su respectivo parche para la construcción del descriptor.

FAST –Features from Accelerated Segment Test- (Rosten & Drummond, 2006) es un método de detección de esquinas. La ventaja más importante de FAST es su eficiencia computacional, ya que está diseñado con ese objetivo. El algoritmo utiliza, básicamente, un círculo de 16 pixels alrededor de un punto p para decidir si es una esquina, comparando sus intensidades con respecto a un umbral t . La **Figura 2.2(c)** muestra el conjunto de pixels utilizado para evaluar si un punto es una esquina, usando el criterio *FAST*.

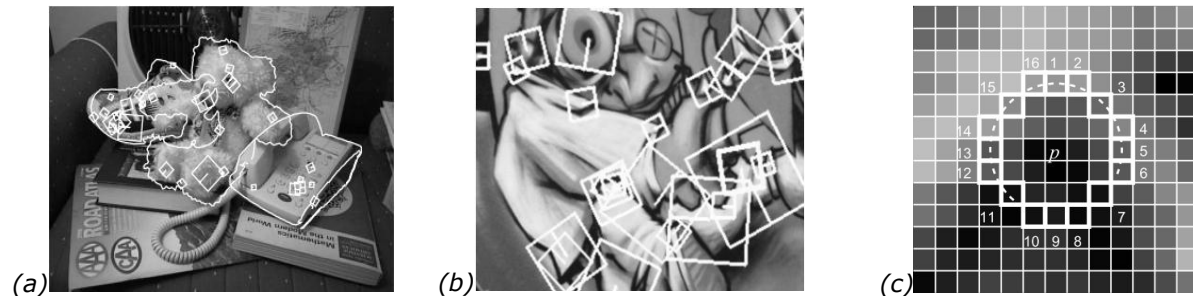


Figura 2.2 (a) Ejemplo de reconocimiento de objetos y su pose bajo condiciones de oclusión, usando *SIFT* (D.G. Lowe, 1999) (b) Regiones cuadradas corregidas con orientación para puntos representativos SURF (Bay et al., 2008) (c) Círculo de Bresenham de radio 3, usado para evaluar una esquina FAST (Rosten & Drummond, 2006)

BRIEF –Binary Robust Independent Elementary Features- (Calonder, Lepetit, Strecha, & Fua, 2010) tiene el mismo enfoque que FAST, pero orientado a la construcción de descriptores. Un descriptor BRIEF es una cadena binaria, en la que cada componente es el resultado de un test de intensidad entre una pareja de pixels, donde estas parejas son obtenidas aleatoriamente al interior de un parche centrado en el punto de interés. La **Figura 2.3**, muestra diferentes formas de generar los test binarios.



Figura 2.3 Diferentes aproximaciones para escoger las localizaciones de los test binarios. Todos, excepto el de más a la derecha, son muestreos aleatorios. Se muestran 128 tests en cada imagen. (Calonder et al., 2010)

Los puntos clave *BRISK –Binary Robust Invariant Scalable Keypoints-* (Leutenegger, Chli, & Siegwart, 2011) son obtenidos por medio de la puntuación *FAST* s , pero aplicada tanto al plano de imagen como a un espacio de escalas. Los descriptores se construyen a partir de una evaluación de intensidades entre parejas de pixels en un patrón circular concéntrico alrededor del punto de interés. La **Figura 2.4(a)** muestra el patrón de muestreo propuesto.

ORB –Oriented FAST and Rotated BRIEF- (Rublee, Rabaud, Konolige, & Bradski, 2011) está basado en BRIEF, pero agregando un componente de rotación. Para la obtención de puntos clave se hace uso del detector de esquinas *FAST* y se eligen los puntos más apropiados por medio de la medida de esquinas *Harris*. El descriptor se construye realizando tests de

intensidad binarios con distribución gaussiana aleatoria al interior de un parche alrededor del punto de interés. La **Figura 2.4(b)** muestra un ejemplo de correspondencia de puntos tipo ORB en una imagen con cambios en escala, rotación y punto de vista.

FREAK -Fast Retina Keypoint- (Alahi, Ortiz, & Vanderghenst, 2012) es un algoritmo para la construcción de descriptores binarios. Para esto se utiliza un patrón de muestreo circular, similar a BRISK, con la diferencia de que tiene una mayor densidad de puntos cerca del centro. Al interior de este patrón se realizan una serie de tests binarios entre parejas de pixels. La **Figura 2.4(c)** muestra el patrón de muestreo propuesto para *FREAK*, el cual está basado en la distribución de los campos receptivos en la retina humana.

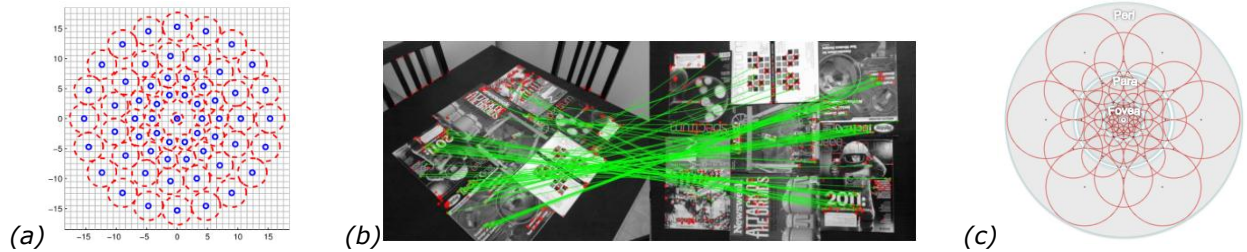
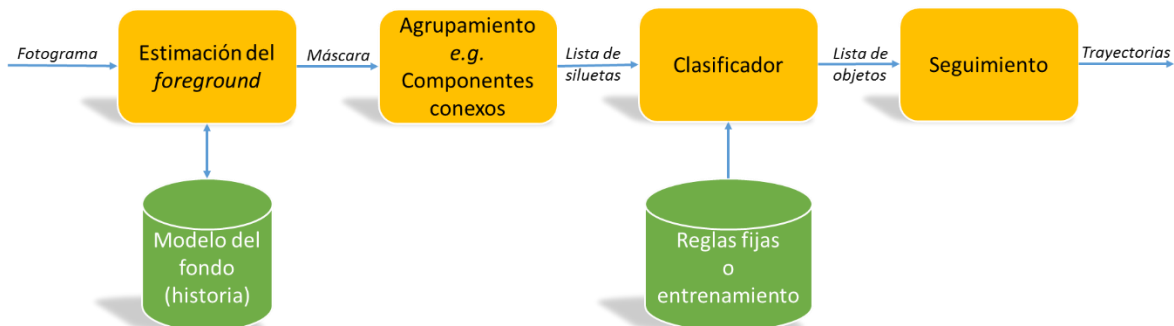


Figura 2.4 (a) Patrón de muestreo de la vecindad para un punto representativo BRISK. (Leutenegger et al., 2011) (b) Mapeo de puntos clave ORB en una imagen con modificaciones en escala, rotación y punto de vista. Las líneas verdes son correspondencias válidas, los círculos rojos indican puntos sin correspondencia (Rublee et al., 2011) (c) Patrón de muestreo de la vecindad de un punto clave para los descriptores FREAK. Los círculos rojos representan los campos receptivos, en donde la imagen se suaviza con su respectivo operador gaussiano. (Alahi et al., 2012)

2.2.1.2 Clasificación de vehículos

Existen dos aproximaciones básicas para la clasificación de vehículos basándose en técnicas de análisis visual: La aproximación *Top-Down* y la aproximación *Bottom-Up*. La **Figura 2.5** muestra un diagrama de bloques de una aplicación típica de análisis de tráfico por medio de video (N. Buch, Velastin, & Orwell, 2011)



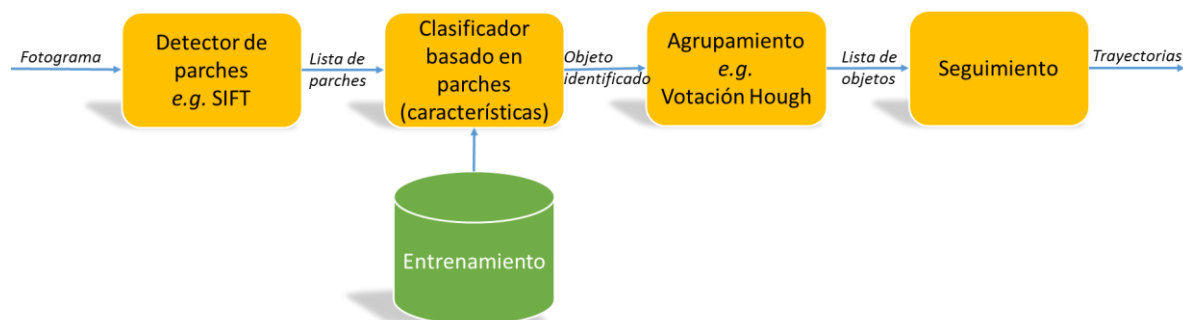


Figura 2.5 Diagrama de bloques de un sistema de clasificación y seguimiento mediante aproximación *Top-Down* (arriba) y *Bottom-Up* (abajo). (N. Buch et al., 2011) / Traducido del original

Tabla 2.2 Clasificación de vehículos *Top-Down* y *Bottom-Up*

Tipo de clasificación	Enfoque	Origen de características	Método	Problemas a resolver
Clasificación <i>Top-Down</i> (N. Buch et al., 2011)	De lo general a lo particular.	Características basadas en región o basadas en contorno.	Se utilizan <i>blobs</i> , bordes, contornos, características estructurales (área, <i>aspect-ratio</i> , momentos, etc.)	Clasificación inter-clase. Entornos de autopista.
Clasificación <i>Bottom-Up</i> (N. Buch et al., 2011)	De lo particular a lo general.	Características locales que describen todos los pixels de la imagen o un subconjunto de estos.	Se clasifican subpartes de los objetos, para posteriormente ser agrupadas en objetos completos.	Clasificación intra-clase. Entornos urbanos. Reconocimiento de objetos genéricos.

En el enfoque de clasificación *Top-Down* (N. Buch et al., 2011), el clasificador necesita información sobre una nueva instancia, lo que usualmente se denomina *features* o características. En esta aproximación, un algoritmo de *Machine-Learning* es entrenado con instancias de clases conocidas para extraer información discriminante a partir de las características. Esta información se utiliza para asignar una clase conocida o la instancia procesada.

Las características pueden ser basadas en región o basadas en contornos. Las características basadas en región usualmente son extraídas del área total que ocupa un objeto en una imagen. Muchas veces se utilizan los momentos de imagen para generar un vector de características de la silueta a procesar. Es posible también utilizar la forma de la silueta para realizar la comparación, siendo esta una de las técnicas más simples a implementar, aunque no por ello débil. (N. Buch, Orwell, & Velastin, 2008), (N. Buch, Orwell, & Velastin, 2010), (Johansson, Wiklund, Forssén, & Granlund, 2009), (Messelodi, Modena, & Zanin, 2004), (Song & Nevatia, 2007), (Gupte & Masoud, 2002) y (G. Zhang, Author, & Avery, 2007) hacen uso del ancho y el largo de la máscara binaria resultado de la segmentación del *foreground* para realizar la clasificación. Esta técnica es especialmente apropiada si se quiere realizar una clasificación inter-clase (bus, automóvil, minivan, etc.) y si se aplica en un entorno de autopistas. Sin embargo, puede no ser aplicable a entornos urbanos, en donde es más necesaria una clasificación intra-clase (entre automóviles con distintos perfiles, por ejemplo) y en donde la oclusión es un fenómeno mucho más común.

La aproximación de clasificación *Bottom-Up* (N. Buch et al., 2011) implica detectar partes de los objetos y clasificar dichas partes, antes de ser agrupadas en objetos. Esta aproximación usualmente se utiliza para reconocimiento de objetos genéricos. Los descriptores de puntos de interés, también llamados *keypoints*, son posiciones en la imagen de las cuales son extraídas características locales.

El vector de características más simple basado en parches es una colección de valores de los pixels en la imagen. Esta aproximación puede ser usada para generar un alfabeto de parches para la clasificación (Agarwal, Awan, & Roth, 2004). El uso de histogramas en lugar de los valores de los pixels permite una mejor invarianza espacial. Por otro lado, la detección de bordes binarios permite una entrada normalizada para descriptores de características, ya que las condiciones de iluminación son removidas durante este proceso.

También es muy usual hacer uso de los algoritmos de extracción de características descritos anteriormente como *SIFT*, *SURF*, u otros métodos como *HoGs* (*Histogram of Oriented Gradients*), el cual está basado en evaluar histogramas locales normalizados de las orientaciones de los gradientes de la imagen en una grilla densa (Dalal & Triggs, 2005)

2.2.2 ESTIMACIÓN DE VELOCIDAD MEDIANTE TÉCNICAS DE ANÁLISIS VISUAL

En el presente trabajo, el problema de estimar la velocidad de un objeto (en este caso, un vehículo) en una secuencia de video se abordó de la siguiente manera: En primer lugar, se estima un modelo del fondo que posteriormente se utilizará para segmentar el objeto de interés en la escena, es decir, obtener un *foreground* estimado. Una vez segmentado el objeto, se procede a aplicarle ciertas operaciones morfológicas que permitan tener un *foreground* mucho más estable. En el área demarcada por el *blob* obtenido de estos procesos se calculan características que permitan un fácil seguimiento, seguimiento que es realizado gracias al método conocido como *flujo óptico*. Haciendo el seguimiento de características de un fotograma al siguiente, y conociendo el tiempo transcurrido entre este par de fotogramas, es posible calcular la velocidad de movimiento de los pixels que pertenecen al objeto de interés, en unidades de $\frac{\text{pixels}}{\text{seg}}$. Es por esto que se hace necesaria una transformación de perspectiva que lleve estas medidas a su correspondiente valor en el mundo real. Finalmente, para obtener una velocidad más estable, se aplica un suavizado utilizando filtros de Kalman. A continuación se da la fundamentación teórica a estos procesos.

2.2.2.1 Técnicas de extracción del fondo de imagen

En (N. Buch et al., 2011) se hace un análisis general del estado del arte de las técnicas de visión artificial para la aplicación en análisis de tráfico urbano. Una aproximación similar tiene (Hall & Nascimento, 2005), en el cual se realiza una comparación en el rendimiento de algoritmos de detección de objetivos basados en diferenciación de fondo adaptativa. La **Tabla 2.3** muestra un resumen de las técnicas de estimación de fondo abordadas en los artículos, seguido por una breve ampliación de cada una.

Tabla 2.3 Principales técnicas de extracción del fondo (*background*) en un fotograma de video. (N. Buch et al., 2011)

Método	Modelo de fondo	Ventajas	Desventajas	Asunciones
Averaging	Promedio de los n fotogramas de video anteriores.	- Muy fácil implementación y uso. - Muy bajo costo computacional. - Modelo de fondo dinámico.	- Aparición de estelas en el modelo. - Relativo alto uso de memoria. - No puede manejar fondos bimodales. - No puede manejar objetos de movimiento lento. - No puede manejar cambios en las condiciones de iluminación	Los pixels de <i>foreground</i> son datos atípicos en la escena.

Median filter	Mediana de los n fotogramas de video anteriores.	• Muy fácil implementación y uso. • Muy bajo costo computacional. • Modelo de fondo dinámico.	• Relativo alto uso de memoria. • No puede manejar fondos bimodales. • No puede manejar objetos de movimiento lento. • No puede manejar cambios en las condiciones de iluminación • Se ve afectado por la tendencia en color de los vehículos.	Probabilidad de aparición del fondo más alta que otros valores hacia arriba y hacia abajo.
Adaptive Single Gaussian	Modelo gaussiano simple por pixel.	• Modelo de fondo dinámico. • Capaz de manejar cambios en iluminación.	• Mayor costo computacional que con filtros media y mediana. • No puede manejar fondos bimodales.	Los pixels de <i>foreground</i> son datos atípicos en la escena.
Mode estimation	Moda del histograma temporal de cada pixel.	• Fácil implementación. • Estimación de fondo más precisa que con media y mediana. • Modelo de fondo dinámico.	• Relativo alto uso de memoria. • No puede manejar fondos bimodales. • No puede manejar cambios en iluminación. • Sensible al tamaño de los <i>bins</i> en el histograma.	Fondo de imagen visible de manera dominante durante el periodo de observación.
Kalman filter	Filtro de Kalman.	• Robusto ante cambios de iluminación. • Modelo de fondo dinámico.	• No maneja bien fondos bimodales. • Mayor costo computacional.	<i>Foreground</i> interpretado como un ruido para el estado del filtro.
Mixture of gaussians	Uno o más modelos gaussianos simples por pixel.	• Mayor robustez ante cambios en iluminación. • Maneja fondos bimodales.	• Alto consumo de memoria. • Complejidad computacional relativamente elevada. • Es importante una buena inicialización de las gaussianas.	Aparición en el fondo de múltiples superficies bajo cambios de iluminación.

La alta demanda de memoria, que supone el filtro media (*averaging*) descrito en la **Tabla 2.3**, puede ser manejada mediante el uso de una media móvil. La aparición de estelas en el modelo se debe a la sensibilidad del promedio frente a datos atípicos. La **Figura 2.6(a)** muestra un ejemplo típico del uso de esta técnica.

El *filtro mediana*, mostrado en la **Tabla 2.3**, es un método en el que se asume que el valor del pixel permanece en el *background* por más de la mitad de los fotogramas en el *buffer*. La **Figura 2.6(b)** muestra un ejemplo típico del uso de esta técnica.

En el método de *Single Gaussian*, que se muestra en la **Tabla 2.3**, se obtienen dos imágenes para el modelo del fondo: una imagen de medias y una imagen de varianzas. Cada pixel en un nuevo fotograma es clasificado como *background* o *foreground* dependiendo de su posición en la distribución gaussiana.

En el método basado en la *moda* (**Tabla 2.3**), si el tamaño de los *bins* en el histograma es muy pequeño y los valores de los pixels de entrada varían entre varios *bins*, no aparecerá un pico distintivo y el método fallará. La **Figura 2.6(c)** muestra un ejemplo de la distribución de valores en un histograma para un pixel de fondo en una toma de video de 2 minutos de duración.

En el método del *filtro de Kalman*, de la **Tabla 2.3**, a pesar de que los cambios en la iluminación son ruido no gaussiano y violan las asunciones básicas para el uso de los mismos, es posible ajustar los estados individuales de los filtros mediante un estimado de la distribución de iluminación en la imagen. Este ajuste da cuenta de una buena robustez del método frente a cambios en la iluminación.

En el método de *Mixture of Gaussians (MoG)*, descrito en la **Tabla 2.3**, si un pixel de fondo fluctúa entre, por ejemplo, dos valores, se guardan dos modelos gaussianos simples, cada uno de los cuales está descrito por los parámetros de media y varianza. Un nuevo valor de pixel es declarado como *foreground* sólo si no pertenece a ninguno de los modelos guardados.

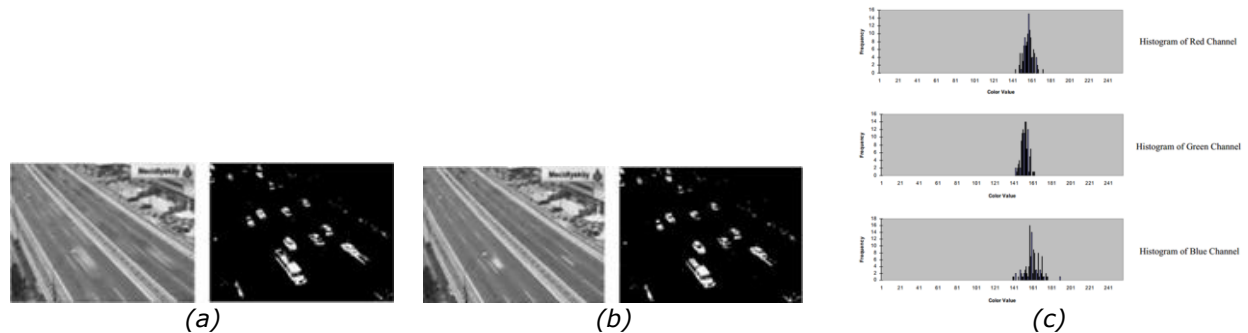


Figura 2.6 Fondo estimado y foreground obtenido mediante el filtro (a) media o averaging y (b) mediana, para 10 fotogramas en el video. (Tamersoy, 2009) (c) Histogramas para el canal rojo (arriba), verde (centro) y azul (abajo) en un pixel de una toma del fondo de 2 minutos de duración. (J. Zheng & Wang, 2006)

2.2.2.2 Técnicas de segmentación del objeto de interés

La **Tabla 2.4** muestra un compendio de algunas de las principales técnicas utilizadas para segmentar un objeto de interés en un fotograma de video.

Tabla 2.4 Principales técnicas de segmentación de objetos de interés (foreground) en un fotograma de video. (N. Buch et al., 2011) (Hall & Nascimento, 2005)

Método	Técnica usada	Observaciones
Frame differencing	Diferencia pixel-por-pixel entre dos fotogramas consecutivos y umbralización.	- Método muy simple. - Computacionalmente eficiente.
Background Subtraction	Resta entre el fotograma actual y un modelo del fondo y umbralización.	- Bajo costo computacional. - Complejidad del modelo del fondo dependiente del método usado.
Gaussian Mixture Model	Cada pixel es modelado en el tiempo como la combinación lineal de dos o más gaussianas.	- Relativamente alta complejidad computacional. - Alto consumo de memoria.
Alternative GMM	Una función de densidad de probabilidad (PDF) es usada para detectar los objetos de interés en la imagen.	- No se guarda una imagen explícita del fondo. - No requiere muchos recursos.
W4	El fondo es modelado por: intensidad mínima (min), intensidad máxima (max), y máxima diferencia de intensidad (D) para cada pixel entre fotogramas consecutivos.	- Bajo costo computacional. - Algoritmo simple y por tanto, simple implementación.
LOTS	Usa dos imágenes de fondo y dos umbrales por pixel.	Algoritmo simple y por tanto, simple implementación.

Diferenciación de fotogramas es un método que no puede lidiar con ruido, cambios abruptos de iluminación o movimientos periódicos en el fondo, como los árboles. Es muy dependiente del *framerate* y de la velocidad y estructura de los objetos a segmentar. (N. Buch et al., 2011)

El costo computacional del método de *sustracción del fondo* es muy bajo, sin embargo, la complejidad de la estimación del fondo depende del método que se utilice para este proceso (*mostrados más atrás*). (N. Buch et al., 2011)

En el modelo de *combinación de gaussianas* (*Gaussian Mixture Model* -GMM-), la estabilidad de las distribuciones gaussianas es evaluada para estimar si son el resultado de un proceso de fondo más estable o un proceso de *foreground* de corta duración. Este modelo puede lidiar con cambios en la iluminación y desorden repetitivo. Se usan dos imágenes, de medias y de varianzas, por cada distribución gaussiana (de dos a cinco, típicamente) y por esto tiene una alta demanda de memoria. (Stauffer & Grimson, 1999)

Una *alternativa al modelo anterior* (Tanaka, 2007) usa una función de densidad de probabilidad (*PDF*) para detectar objetos en la imagen. No se guarda una imagen explícita del fondo. Basado en la *PDF* estimada, se calcula la probabilidad de ocurrencia de un pixel observado. Si la probabilidad es alta, se asume que el pixel es de fondo. Si es baja, el pixel es *foreground*. El algoritmo es muy eficiente computacionalmente porque sólo se calcula una estimación para una *GMM*.

En W4 (Haritaoglu, Harwood, & Davis, 1998) se representa el fondo mediante: la intensidad mínima (*min*), la intensidad máxima (*max*) y la máxima diferencia de intensidades (*D*) para cada pixel entre fotogramas consecutivos, durante el periodo de entrenamiento. Un pixel $I(\phi)$ en será identificado como *foreground* si se cumple la *Ecuación 2.1* o la *Ecuación 2.2*

$$| \min(\phi) - I(\phi) | > D(\phi) \quad \text{Ecuación 2.1}$$

$$| \max(\phi) - I(\phi) | > D(\phi) \quad \text{Ecuación 2.2}$$

El método *LOTS* (Boult, Micheals, & Eckmann, 2001) usa dos imágenes de fondo y dos umbrales por pixel. Las dos imágenes de fondo modelan cambios periódicos en el ambiente, cómo árboles. La imagen umbralizada por pixel, puede tratar a cada pixel de manera diferenciada, permitiendo al detector ser robusto frente a ruido localizado en regiones de tamaño pequeño en la imagen.

2.2.2.3 Operaciones morfológicas

Las operaciones básicas, y por medio de las cuales se realizan todas las demás operaciones morfológicas, son la erosión y la dilatación.

La erosión de la imagen binaria A por el elemento estructurante B está definida como muestra la *Ecuación 2.3*

$$A \ominus B = \{z \in E \mid B_z \subseteq A\} \quad \text{Ecuación 2.3}$$

Siendo E un espacio euclidiano o una cuadrícula de enteros, A una imagen binaria en E y B_z la traslación de B por el vector z , es decir, $B_z = \{b + z \mid b \in B\}, \forall z \in E$

La dilatación de A por el elemento estructurante B está definida por la Ecuación 2.4

$$A \oplus B = \bigcup_{b \in B} A_b \quad \text{Ecuación 2.4}$$

Siendo E un espacio euclidiano o una cuadrícula de enteros, y A una imagen binaria en E .

2.2.2.4 Good Features to Track

(Shi & Tomasi, 1994) Es un método de extracción de características que es óptimo desde su construcción, ya que está basado en cómo funciona el *tracker*, y un método de monitoreo de características que es robusto ante oclusiones, desocclusiones y características que no corresponden a puntos en el mundo real. La idea principal de este trabajo es que las características correctas son aquellas que hacen que el *tracker* trabaje de la mejor manera.

2.2.2.5 Flujo óptico



Figura 2.7 Seguimiento de puntos característicos mediante flujo óptico desde el inicio de la secuencia de video –moto de izquierda a derecha- (a) y entre dos fotogramas consecutivos –hatchback de derecha a izquierda- (b). (Imagen tomada en la Universidad del Valle)

El flujo óptico es el patrón de movimiento aparente de objetos, superficies o bordes en una escena, producido por el movimiento relativo entre un observador (un ojo o una cámara) y la escena. Por esta razón, el flujo óptico puede dar información importante sobre la disposición espacial de los objetos vistos y la tasa de cambio de esta disposición. El cómputo del flujo óptico es un procedimiento de dos pasos:

- Medición de las derivadas espacio-temporales de la intensidad (que es equivalente a medir las velocidades normales a las estructuras de intensidad locales).
- Integrar las velocidades normales en velocidades totales, por ejemplo, localmente por medio del cálculo de mínimos cuadrados o globalmente por medio de regularización.

Asumiendo que un pixel que se mueve entre dos fotogramas consecutivos tiene la misma intensidad y resolviendo la ecuación diferencial $I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t)$, se obtiene la Ecuación 2.5

$$I_x V_x + I_y V_y = -I_t \quad \text{Ecuación 2.5}$$

Donde I_x , I_y e I_t son las derivadas de la intensidad en x , y y t respectivamente y V_x , V_y son las componentes x e y de la velocidad o flujo óptico de $I(x, y, t)$.

Este sistema infradeterminado se resuelve considerando que el movimiento es uniforme y el mismo para los pixels vecinos, obteniendo de esta manera un sistema sobredeterminado que puede ser resuelto mediante un proceso de minimización. La **Figura 2.7** muestra el método en dos secuencias de video.

2.2.2.6 Transformación de perspectiva

La transformación de perspectiva se refiere a la obtención de una relación entre medidas geométricas en el mundo real (metros, centímetros, etc.) y las correspondientes medidas mapeadas en el plano de imagen (pixels). En muchos casos, debido a la pérdida de la información de profundidad por el uso de cámaras monoculares, es preciso lidiar con situaciones en las que se presenta una distorsión de perspectiva a lo largo de alguna trayectoria, esto es, una variación en la escala de la relación metro-píxel en la imagen. Esta variación no es lineal, y para superar esta situación se hace uso de distintas técnicas. Una de las técnicas usadas, particularmente en el campo de estimación de velocidad de vehículos, cuando los mismos tienen la misma dirección que el eje focal de la cámara, consiste en obtener los bordes de demarcación de la carretera (transformada *Hough* u otras técnicas) y calcular los puntos de desvanecimiento para, con esta información, realizar una rectificación de la imagen. Este proceso se puede observar en la **Figura 2.8**.

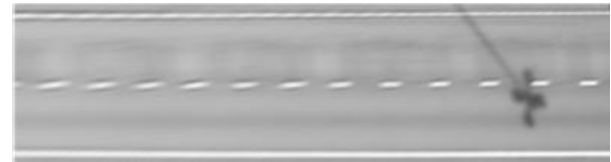


Figura 2.8 Transformación de perspectiva cuando la trayectoria del objeto de interés va en sentido del eje focal de la cámara (Cathey & Dailey, 2005)

2.2.2.7 Filtro de Kalman

Un filtro de Kalman es un estimador para lo que es llamado el *Linear-Quadratic-Gaussian (LQG) estimation problem*, que consiste en el problema de estimar el estado instantáneo de un sistema dinámico lineal perturbado por ruido blanco gaussiano, usando mediciones linealmente relacionadas al estado, pero degradadas por ruido blanco gaussiano. El estimador resultante es estadísticamente óptimo con respecto a cualquier función cuadrática de estimación de error (Grewal & Andrews, 2008)

En otras palabras, un filtro de Kalman infiere parámetros de interés de un sistema dinámico lineal a partir de observaciones indirectas (no se miden todas las variables de interés - variables a estimar), imprecisas (la medición está afectada por ruido -que se asume blanco y gaussiano-) e inciertas (no es posible conocer *a priori* el valor de las observaciones realizadas). (Kleeman, 1996) La **Figura 2.9(a)** muestra el esquema básico de funcionamiento de un filtro de Kalman. La **Tabla 2.5** muestra las ecuaciones en tiempo discreto para el cálculo recursivo de un filtro de Kalman.

Tabla 2.5 Ecuaciones del filtro de Kalman en tiempo discreto. (Grewal & Andrews, 2008)

Modelo dinámico del sistema:	$x_k = \Phi_{k-1} x_{k-1} + w_{k-1}$ $w_k \sim N(0, Q_k)$
Modelo de medición:	$z_k = H_k x_k + v_k$ $v_k \sim N(0, R_k)$
Condiciones iniciales:	$E\langle x_0 \rangle = \hat{x}_0$ $E\langle \tilde{x}_0 \tilde{x}_0^T \rangle = P_0$
Asunción de independencia:	$E\langle w_k v_j^T \rangle = 0$ para todo k y j
Extrapolación de la estimación del estado (predicción):	$\hat{x}_k(-) = \Phi_{k-1} \hat{x}_{k-1}(+)$
Extrapolación de la matriz de covarianza del error (predicción):	$P_k(-) = \Phi_{k-1} P_{k-1}(+) \Phi_{k-1}^T + Q_{k-1}$

Actualización de la estimación del estado (corrección):	$\hat{x}_k(+) = \hat{x}_k(-) + \bar{K}_k[z_k - H_k\hat{x}_k(-)]$
Actualización de la matriz de covarianza del error (corrección):	$P_k(+) = [I - \bar{K}_k H_k]P_k(-)$
Matriz de ganancias de Kalman:	$\bar{K}_k = P_k(-)H_k^T[H_k P_k(-)H_k^T + R_k]^{-1}$

Hay tres tipos de estimadores para el problema LQG. La relación entre estas clases de estimadores es ilustrada en la **Figura 2.9(b)** La distinción depende de cuándo un valor estimado del vector de estado $\hat{x}$ es válido y cómo ese tiempo está relacionado con los tiempos en los cuales las mediciones usadas para realizar esa estimación fueron tomadas.

- Los *filtros* ocupan la línea diagonal en la **Figura 2.9(b)** Hacen uso de observaciones hasta e incluyendo el tiempo en el cual se estima el estado del sistema.

$$t_{obs} \leq t_{est} \quad \text{Ecuación 2.6}$$

- Los *predictores* ocupan la región debajo de la línea diagonal en la **Figura 2.9(b)** Hacen uso de observaciones estrictamente anteriores al tiempo en el cual se estima el estado del sistema.

$$t_{obs} < t_{est} \quad \text{Ecuación 2.7}$$

- Los *suavizadores* ocupan la región por encima de la línea diagonal en la **Figura 2.9(b)** Hacen uso de observaciones en tiempos anteriores y posteriores al tiempo en el cual se estima el estado del sistema.

$$t_{obs} \leq t_{est} < t_{obs} \quad \text{Ecuación 2.8}$$

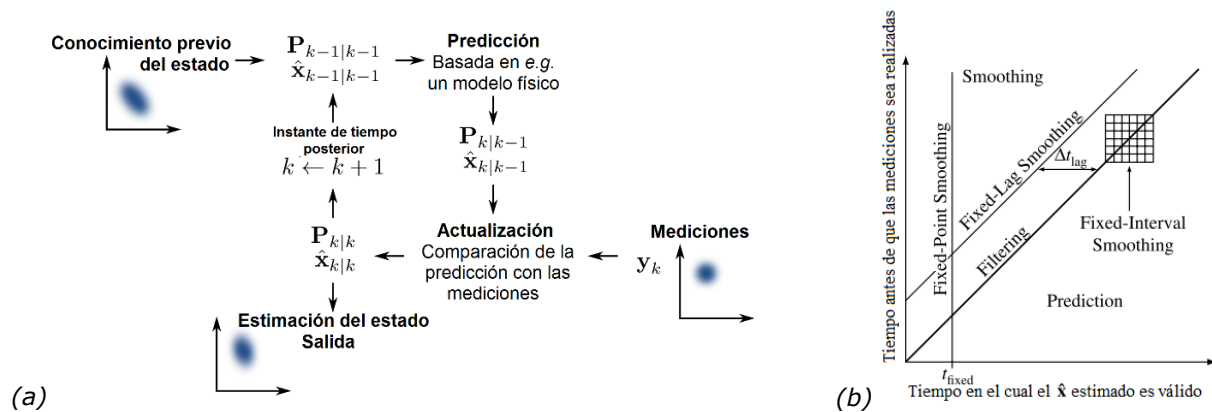


Figura 2.9 (a) Pasos básicos del filtrado de Kalman: predicción y corrección. El filtro calcula no sólo el valor estimado del estado del sistema, sino también la varianza estimada. "Basic concept of Kalman filtering" por Petteri Aimonen licenciada bajo CC0 1.0 / Traducida del original (b) Relaciones temporales entre estimación y mediciones para distintos tipos de estimadores. (Grewal & Andrews, 2008)

Dentro de los estimadores suavizadores hay, además, tres tipos distintos, de acuerdo a la dependencia entre los datos de medición y el vector de estado estimado.

- Los suavizadores de intervalo fijo (*Fixed-Interval Smoothers*) usan todas las mediciones hechas en el tiempo t_{meas} en un intervalo fijo de tiempo $t_{start} \leq t_{meas} \leq t_{end}$ para producir un vector de estado estimado $\hat{x}(t_{est})$ en tiempos $t_{start} \leq t_{est} \leq t_{end}$ en el

mismo intervalo fijo, como se ilustra con el área cuadrada sobre la diagonal en la **Figura 2.9(b)**

- Los suavizadores de retardo fijo (*Fixed-Lag Smoothers*) usan todas las mediciones hechas en un intervalo de tiempo $t_{start} \leq t_{meas} \leq t_{est} + \Delta t_{lag}$ para la estimación $\hat{x}(t_{est})$ en el tiempo t_{est} . Es decir, el estimado generado en un tiempo t es para el valor de x en el tiempo $t - \Delta t_{lag}$, donde Δt_{lag} es un tiempo de retardo fijo, como se ilustra con la línea paralela a la línea de "filtrado" en la **Figura 2.9(b)**
- Los suavizadores de punto fijo (*Fixed-Point Smoothers*) generan un estimado $\hat{x}(t_{fixed})$ de x en un tiempo fijo t_{fixed} basado en todas las mediciones $z(t_{meas})$ hasta el momento actual t (i.e., $t_{start} \leq t_{meas} \leq t$). Los suavizadores de punto fijo actúan como predictores cuando $t < t_{fixed}$, como filtros cuando $t = t_{fixed}$ y como suavizadores cuando $t > t_{fixed}$. Esto es ilustrado por la línea vertical en la **Figura 2.9(b)**

En el presente trabajo se usó el método de suavizado de intervalo fijo en tres pasos (Grewal & Andrews, 2008). El método opera en tiempo discreto y genera las estimaciones filtradas y suavizadas, así como también sus respectivas covarianzas de incertidumbre de estimación. Es por esto que se eligió este método como suavizador para el sistema implementado. El procedimiento consiste en ejecutar tres recorridos a través de la totalidad de las mediciones y los datos derivados de allí, así:

1. Un paso completo del filtro de Kalman en dirección hacia adelante (i.e. con el tiempo de medición en aumento), almacenando los valores *a posteriori* de la estimación del vector de estados $\hat{x}_{[f],k}(+)$ y la matriz de covarianza de incertidumbre de estimación asociada $P_{[f],k}(+)$, de acuerdo a las ecuaciones de la **Tabla 2.5**
2. Un paso completo del filtro de Kalman en dirección hacia atrás (i.e. con el tiempo de medición en disminución), almacenando los valores *a priori* de la estimación del vector de estados $\hat{x}_{[b],k-1}(-)$ y la matriz de covarianza de incertidumbre de estimación asociada $P_{[b],k-1}(-)$, de acuerdo a las Ecuación 2.9 y Ecuación 2.10

$$\hat{x}_{[b],k-1}(-) = \Phi_{[f],k-1}^{-1} \hat{x}_{[b],k}(+) \quad \text{Ecuación 2.9}$$

$$P_{[b],k-1}(-) = \Phi_{[f],k-1}^{-1} P_{[b],k}(+) \Phi_{[f],k-1}^{-T} + Q_k \quad \text{Ecuación 2.10}$$

3. Un tercer paso de suavizado combinando los datos hacia adelante y hacia atrás para obtener el estimado suavizado.

$$\hat{x}_{[s],k}(+) = \hat{x}_{[f],k}(+) + \underbrace{P_{[f],k}(+) [P_{[f],k}(+) + P_{[b],k}(-)]^{-1}}_{\tilde{K}} [\hat{x}_{[b],k}(-) - \hat{x}_{[f],k}(+)] \quad \text{Ecuación 2.11}$$

2.2.3 CLASIFICACIÓN DE OBJETOS USANDO INFORMACIÓN DE RANGO

En este apartado se presentan todas las técnicas relevantes y representativas actuales en el área de detección y el reconocimiento en 3 dimensiones, esto incluye las técnicas para la segmentación de planos a partir de información de rango, obtención de normales a la superficie, así como las técnicas para la descripción de los elementos. En primer lugar se presenta una tabla de resumen de las principales técnicas y posteriormente se ahonda en la descripción de las más relevantes.

2.2.3.1 Segmentación de Información de Rango usando Normales de Superficie

Debido a La gran cantidad de información a procesar en un nube de puntos, es necesario recurrir a técnicas de segmentación que permitan descomponer la nube en clústeres más pequeños a procesar e incluso detectar modelos geométricos primarios, que permitan una eliminación basada en parámetros, en esta sección se presenta un análisis de las principales técnicas de esta temática. Adicionalmente en esta sección se presenta el estado del arte de la extracción de las normales de superficie, dada su importancia para el proceso de descripción.

El estado del arte en esta temática, ofrece tres grandes ramas de enfoque para la estimación de las normales de superficie y la segmentación de las nubes de puntos; en primer lugar un grande grupo de trabajos utilizan el enfoque de área promediada, donde la información de rango es dividida en mallas de diferente geometría, posteriormente a través de los vértices de estas mallas se obtiene la información que describe la superficie, y a través de un promedio de vecindad se logra la obtención del vector normal para cada punto de interés. El segundo grande grupo, y en el más común entre los métodos con esta finalidad, realiza el ajuste de planos o elementos geométricos triviales a la información de rango (puntos de interés y sus vecinos) mediante el uso de *PCA (Principal Component Analysis)* y *SVD (Singular Value Decomposition)*, además del ajuste de parámetros de control y precisión. Finalmente un tercer grupo hace uso del ajuste iterativo con técnicas como *RANSAC* e *ICP*. En primer lugar en este apartado se presenta una descripción del concepto de normal de superficie, segundo en la Tabla 2.6 se resumen las técnicas del estado del arte para la estimación de las normales y finalmente se plantean sus características.

El vector normal (en un ámbito 3D) a una superficie en un punto determinado, es el vector perpendicular al plano tangente en ese punto de la superficie, esto es representado en la **Figura 2.10**, esta característica describe la orientación de cada una de las vecindades locales a el punto de interés, es una medida de la curvatura y tiene aplicaciones en la construcción de descriptores 3D, además en la creación de las gráficas computacionales permitiendo la aplicación de niveles de intensidad de brillo según su distribución y orientación.

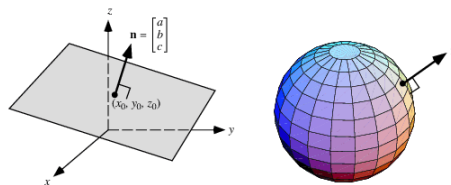


Figura 2.10 Concepto de Normal de superficie.

Para una superficie S determinada implícitamente satisfaciendo $F(x, y, z) = 0$, la normal puede ser estimada en cada punto sencillamente mediante el gradiente $\nabla F(x, y, z)$ evaluado en ese punto. Para una superficie S determinada explícitamente $F(x, y)$, puede hallarse la normal transformándose a una determinación implícita $F(x, y, z) = z - f(x, y) = 0$, y utilizando el gradiente $\nabla F(x, y, z)$; Otra forma es usando el gradiente de la forma explícita $\nabla F(x, y)$ para hallar el de la implícita como $\nabla F(x, y, z) = \hat{k} - \nabla f(x, y)$. Donde $\hat{k}$ es el vector unitario del eje Z.

Tabla 2.6 Resumen de estado del arte en la segmentación de planos y obtención de las normales de superficie.

Trabajo Relacionado	Enfoque de la técnica	Descripción
(Holzer, Rusu, Dixon, Gedikli, & Navab, 2012)	Método de promediado, y paramétrico.	• Uso de imágenes integrales. • Ventana adaptativa
(K. Klasing, Althoff, Wollherr, & Buss, 2009)	Método paramétrico.	• Índices de calidad para evaluación. • Diseño diversos modelos paramétricos.
(J. Weingarten & Siegwart, 2005)	Método paramétrico.	• Algoritmo de descomposición y fusión de celdas.
(Mitra, Nguyen, & Guibas, 2003)	Método paramétrico.	• Proceso complejo de minimización y selección del valor de soporte.
(Badino, Huber, Park, & Kanade, 2011)	Método basado en coordenadas esféricas.	• Independiente del valor de rango • Concepción de modelos esféricos. • Extremadamente rápido.
(Nüchter & Hertzberg, 2008)	Método de ajuste de modelos paramétricos aleatorios.	• Desarrollo de la segmentación mediante <i>RANSAC</i> e <i>ICP</i>.
(Xiao, Zhang, Zhang, Zhang, & Hildre, 2011)	Método paramétrico.	• Construcción, clasificación y fusión de mallas. • Algoritmo rápido.
(K. Klasing, Wollherr, & Buss, 2009)	Método paramétrico, diseñado para tiempo real.	• Segmentación incremental • controles de sensibilidad • Tiempo real.
(Goron, Tamas, Reti, & Lazea, 2010)	Método de ajuste de modelos paramétricos aleatorios.	• Filtrado de regiones. • Uso de <i>RANSAC</i> para estimación de planos.
(Klaas Klasing, Wollherr, & Buss, 2008)	Método paramétrico simple.	• Extremadamente sencillo • Técnica de grafo denominada <i>RBNN</i>.
(Zhan & Liang, 2010)	Método paramétrico y uso de espacio de color.	• Esquema general se clusterización de rango y estimación de modelo en vecindad • Segmentación en espacio de color.
(Hegde & Ye, 2011)	Método paramétrico.	• Uso de imágenes ERI • criterios de similitud
(L. Chen, Nguyen, & Lin, 2012)	Método paramétrico y sistema de clasificación	• Fusión del cálculo paramétrico y los histogramas locales para clasificación 3D.
(Hinterstoisser et al., 2012)	Método de estimación de normales con series de Taylor.	• Uso de nubes de puntos organizadas • Aproximación de Taylor para modelar la información de rango.
(J. W. Weingarten, Gruener, & Siegwart, 2004)	Método paramétrico.	• Utilización del método bajo el esquema <i>SPMODE</i> (algoritmo iterativo de estimación paramétrica).
(Hoppe, DeRose, Duchamp, McDonald, & Stuetzle, 1992)	Método paramétrico.	• Artículo base de la bibliografía, • Basado en distancias signadas y procesos de rectificación.
(D. Zheng, Xu, & Ren-xi, 2009)	Método de área promediada	• Generación de mallas de triángulos en las nubes de puntos. • Promediado vectorial

Los métodos en (Holzer et al., 2012) y (Hinterstoisser et al., 2012) están diseñado para nubes de puntos Organizadas, Solamente aplicable a datos organizados obtenidos con sensores como el *Kinect*. En el primer trabajo se obtienen primero obtiene imágenes integrales (reducen la carga computacional) a partir de estas se diseña un proceso de selección de ventana de procesamiento adaptativa. Finalmente se aplican tanto técnicas de promediado de área como paramétricas para la estimación de las normales de superficie. En el segundo trabajo se utiliza una aproximación de series de Taylor para modelar la función de profundidad, y a partir del producto cruz entre dos vectores directores, obtiene el vector normal a la superficie, trabaja con una grilla definida con un número predeterminado de contenedores para vectores normales de superficie, y aplica una medida de similitud que corresponde al producto punto entre vectores normalizados. Extremadamente sencillo, rápido y robusto, aplicación del concepto de reconocimiento multimodal de patrones.

Los trabajos realizados en (K. Klasing, Wollherr, et al., 2009), (K. Klasing, Althoff, et al., 2009), (Zhan & Liang, 2010) y (J. Weingarten & Siegwart, 2005) realizan la estimación de los vectores normales de los puntos entrantes es basada en las características paramétricas de las k -vecindades. El método utilizado es el de estimación *PlaneSVD* y *planePCA* de la matriz de vecindad aumentada, y la búsqueda de una orientación consistente. Ofrecen controles de sensibilidad. Métodos muy robustos. El tercero presenta adicionalmente un algoritmo de segmentación y clusterización en colores, los componentes de los vectores normales son normalizados y transformados a un espacio de color $XYZ \rightarrow RGB$, finalmente se realiza la segmentación con la distancia euclidiana en el espacio de color. En el último trabajo se realiza adicionalmente una previa descomposición de celdas. Similar a los trabajos anteriores En (J. W. Weingarten et al., 2004) Se realiza una estimación paramétrica, pero construye un modelo *SPmode* (*The Symmetries and Perturbations Mode*) siendo este es un marco para representar información geométrica errónea. Los autores en (Mittra et al., 2003) realizan la obtención de las normales de superficie mediante la estimación de planos aplicando el estudio de los efectos de la curvatura, el ruido, y la densidad de muestreo para la elección del tamaño de la vecindad de cálculo. Se obtienen resultados muy robustos y fieles a las características de las muestras. Los autores en (Badino et al., 2011) realizan la estimación de las normales de superficie a partir de información en coordenadas esféricas, el algoritmo prueba ser altamente eficiente y robusto, además muestra ser muy inmune al ruido. Es un algoritmo pionero, con documentación escasa. Se realiza un proceso extensivo de evaluación para variaciones de una gran variedad de parámetros.

En (Papazov & Burschka, n.d.) Se propone un algoritmo cuya finalidad es la obtención de la transformación T que mejor hace corresponder una serie de descriptores almacenados en una tabla hash, por medio de un proceso de iteración; estos descriptores se encuentran definidos para cada par de puntos escogidos por un proceso de selección uniforme. Este algoritmo propone un número de interacciones dependiente de la probabilidad de éxito de hallar la transformación correcta que el usuario ingrese como parámetro. La segmentación de planos en (Nüchter & Hertzberg, 2008) y (Goron et al., 2010) son implementada mediante la combinación del algoritmo *Iterative Closest Points (ICP)*, *PCA* y el algoritmo RANSAC. El primero utiliza una red de restricción para el etiquetado de planos e interpretación; además obtiene el gradiente local entre los datos en coordenadas esféricas para la interpretación semántica de planos segmentados. El segundo utiliza métodos estadísticos de filtrado y de eliminación de áreas con densidades bajas.

Un novedoso algoritmo llamado *grid-based región growing* es presentado para ambientes estructurados en (Xiao et al., 2011); la nube de puntos es dividida en regiones pequeñas basadas en información de vecindad cuando esta es vista como una imagen de rango. La pequeña región es llamada malla. Las mallas son clasificadas en diferentes categorías, de acuerdo a su apariencia local. Finalmente las mallas planares son agrupadas en grandes parches mediante crecimiento de regiones. La Estimación de planos es realizada con la matriz de covarianza y minimización de mínimos cuadrados.

El trabajo en (Klaas Klasing et al., 2008) presenta un método descrito como *Radially Bounded Nearest Neighborgraph (RBNN)*, está basado en un grafo con radio determinado, donde los nodos son los puntos de la nube y los conectores son las tripleta entre puntos y su distancia euclidiana, en contraste con KNN que se basa en los K -vecinos cercanos. Baja carga computacional, sin embargo no es muy robusto, y exige nubes de puntos densas. El trabajo (Hegde & Ye, 2011) Propone un método recursivo para la extracción de superficies planares en información con ruido. En primer lugar transforma la imagen de rango a una imagen de tipo *ERI* (*Enhanced Range Image*), la cual contiene características propias de la

geometría de la información, luego esta se agrupa en súper-píxeles usa un método de grafo basado en estos nuevos píxeles como nodos del mismo. Una función de similaridad es establecida para definir los costos de los conectores del grafo, el cual es particionado hasta lograr ciertas condiciones. Finalmente el algoritmo genera fusiones de subregiones basado en funciones de similaridad. Es un Método bastante robusto.

En (L. Chen et al., 2012) se presenta un sistema bastante robusto de segmentación y clasificación de elementos particulares (vehículos, personas) compuesto por tres etapas. En primer lugar se realiza un proceso de segmentación que permite eliminar el *background* del proceso (disminuyendo el número de regiones a analizar y clasificar, luego dos etapas de clasificación incluyentes: histogramas de orientación de vectores normales, además de la dimensión y densidad de puntos. Es extremadamente robusto y con respuestas de gran precisión.

En (Hoppe et al., 1992) obtienen las superficies de la nube de puntos, mediante el cálculo de una función de distancia geométrica signada desde puntos arbitrarios hasta la superficie, el set cero de esta función corresponde a la superficie deseada. Para la obtención de esta función de distancia se utiliza la estimación de planos para las subregiones de K-vecinos cercanos y además se realizan correcciones en la orientación y propagación de los planos mediante una gráfica *MST (Minimum Spanning Tree)*. Este trabajo es el Artículo Base de la mayoría de trabajos actuales en la estimación de los vectores normales a la superficie.

En (D. Zheng et al., 2009) la nube de puntos desorganizada es proyectada en un plano 2D a través de la superficie cilíndrica. Luego se realiza la estimación de mallas de triángulos, a las cuales se les calcula el vector normal aproximado y su área correspondiente, finalmente mediante un promedio se obtienen los vectores normales para cada punto. Método extremadamente sencillo, resultados muy precisos en la estimación de la orientación de los vectores normales. Costo computacional alto por el pre cálculo de las mallas de triángulos.

2.2.3.2 Descriptores 3D

A continuación es presentada una tabla comparativa de los principales descriptores 3D presentes en la bibliografía consultada, para cada uno se indica el tipo de enfoque, una descripción breve de sus características, el tamaño de descriptor, y se indica si retribuye la información completa de la pose del elemento (6DoF). Adicionalmente son incluidas dos técnicas que no generan propiamente descriptores, pero adicionan una etapa de verificación minimización a cualquiera de los otros descriptores locales.

Tabla 2.7 Resumen del estado del arte en el diseño de descriptores 3D y el reconocimiento 3D.

Trabajo Relacionado	Enfoque de la técnica	Descripción	Dimensión histograma (Contenedores)	Varianza al punto de vista y 6DOF
(Federico Tombari, Salti, & Stefano, 2010)	Método con enfoque local.	- Marco local repetible. - Ángulos entre normales.	401 contenedores	NO
(Wohlkinger & Vincze, 2011)	Método con enfoque global.	- Distancias D2, A3 y D3, - Rápido.	10 histogramas * 64 contenedores	NO
(Radu Bogdan Rusu, Marton, Blodow, & Beetz, n.d.; R. Rusu, 2010)	Método con enfoque local.	- Ángulos entre Normales - Descriptor robusto.	15 contenedores	NO
(Radu Bogdan Rusu, Blodow, & Beetz, 2009; R. Rusu, 2010)	Método con enfoque local.	- Versión optimizada de PFH - Descriptor robusto.	33 contenedores	NO

(R B Rusu, Holzbach, Beetz, & Bradski, 2009; R. Rusu, 2010)	Método con enfoque global.	• Versión global del FPFH • Árbol octree • Probabilidades de forma • Uso de SVM	$(n_c + 1)(n_c + 2)/2$ Donde n_c es el número de categorías.	NO
(R. Rusu, 2010; R.B. Rusu, Bradski, Thibaux, & Hsu, 2010)	Método con enfoque global.	• Ángulos entre eje óptico y normales	308 contenedores	SI
(Aldoma et al., 2011; R. Rusu, 2010)	Método con enfoque global.	• Obtención previa de regiones estables	308 contenedores	SI
(Aldoma, Tombari, Rusu, & Vincze, 2012)	Método con enfoque global.	• Marco de referencia local repetible • Filtrado gaussiano.	313 contenedores	SI
(Papazov & Burschka, n.d.)	Método de ajuste de correspondencias locales modelo-escena.	• Tabla hash. • Minimización de parámetros	NO APLICA (Longitud x de descriptor local)	NO
(Steder & Konolige, n.d.; Steder & Rusu, 2011)	Método con enfoque local.	• Extracción de bordes e índices de interés. • Imágenes rango 2D • Operaciones 2D/3D • Parches normales alineados	36 contenedores	SI
(A Velizhev, 2012; Alexander Velizhev, 2012)	Método de ajuste de correspondencias locales modelo-escena.	• Votaciones locales con respecto a la dirección del centro de masa de la nube de puntos.	NO APLICA (Longitud x de descriptor local usado)	NO
(H. Chen & Bhanu, 2007)	Método con enfoque local.	• Índices calidad • histogramas locales 2D • Tipo de superficie local. • Agrupación geométrica.	Definido por la selección del número de k vecinos.	SI
(Ramezani & Ebrahimnezhad, 2011)	Método con enfoque local.	• Histogramas Locales 2D • uso de redes neuronales	20 contenedores por dimensión.	SI
(L. Chen et al., 2012)	Método con enfoque local y global.	• Índices de calidad • Lógica difusa geométrica. • Histogramas globales de ángulos.	No especifica	SI
(F. Tombari & Stefano, 2010)	Método con enfoque local.	• Agrupación geométrica de correspondencias	NO APLICA (Longitud x de descriptor local)	NO
(Bielicki & Sitnik, 2013)	Método con enfoque local.	• Detección de bordes locales, • Descriptores C1 y C2 paramétricos. • Proceso de correlación.	Determinado por la tasa de muestreo de puntos en la implementación	NO
(Johnson, 1997)	Método con enfoque local.	• Estrategia de rechazo y umbralización de ángulos vectoriales • Ponderación por distancia.	153 contenedores	NO

El proceso de extracción de descriptor *SHOT* (*Signature of Histograms of Orientations*), (Tombari, Salti, & Stefano, 2010) lleva a cabo la construcción y optimización de un proceso para lograr la repetitividad de los marcos de referencia locales de las características en las nubes de puntos 3D; esto se realiza mediante la descomposición de vectores propios (EVD por sus siglas en inglés), obteniendo los vectores propios de la matriz de covarianza en un rango esférico de radio R desde el punto característico, adicionalmente asignando un peso según la distancia de los puntos vecinos en el rango; La obtención de características consiste en la superposición de una grilla esférica de 32 divisiones con centro en el punto característico (**Figura 2.11(a)**), y la creación de histogramas de 11 contenedores los cuales

acumulan la presencia de la medida del coseno del ángulo entre los vectores normales del punto característico y cada punto de la división, el histograma es normalizado para hacerlo robusto a la variación de la densidad de puntos.

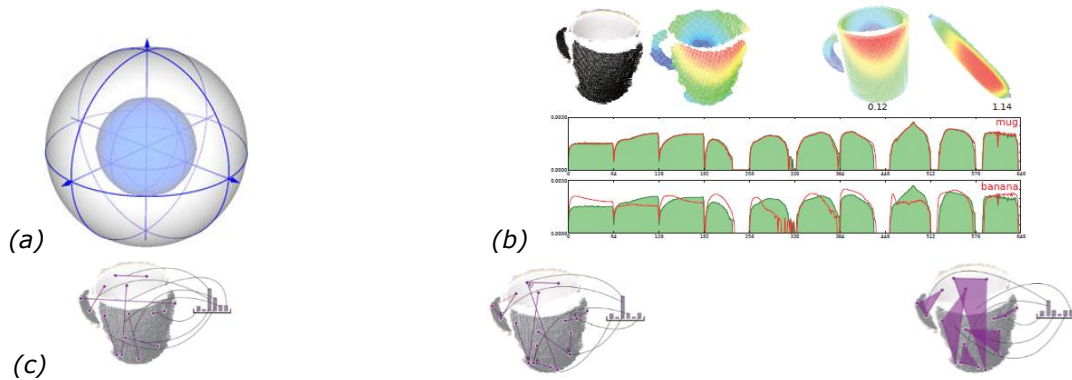


Figura 2.11 (a) Estructura de la grilla esférica para la extracción del histograma del descriptor SHOT. (Federico Tombari et al., 2010). b) Descriptor ESF y sus 10 Histogramas concatenados de 64 contenedores, para dos tipos de elementos diferentes (Mug, Banana); c) Funciones de forma D2, A3 y D3 del descriptor ESF, para un Mug. (Wohlkinger & Vincze, 2011).

El descriptor *ESF* (*Ensemble of Shape Functions*), (Wohlkinger&Vincze, 2011), está definido como un conjunto de 10 histogramas de 64 contenedores **Figura 2.11(b)**, los histogramas dependen de 3 tipos de funciones que definen la forma de la nube de puntos 3D, denominadas D2, A3 y D3, **Figura 2.11(c)** definidas en su orden como: las distancias entre un par de puntos seleccionados aleatoriamente, el ángulo entre dos líneas creadas para 3 puntos aleatorios, y el área del triángulo formado por 3 puntos aleatorios. Para cada función se elaboran 3 subhistogramas considerando si los puntos analizados se encuentran fuera, adentro o las dos, en la superficie del elemento que es envuelto en una grilla de 64 voxels de lado. Finalmente se considera un histograma adicional para la tasa entre líneas exteriores e interiores a la superficie.

El descriptor *PFH* (*Point Feature Histograms*), (R.B. Rusu, Marton, Blodow, & Beetz, 2008) tiene como meta representar las propiedades geométricas y la curvatura de la vecindad de cada punto componente de la nube de puntos en tres dimensiones, esto es logrado mediante la obtención de histogramas multidimensionales a partir de las relaciones entre las normales de superficie en cada punto, acumulado las características propias de dicha vecindad y generando una firma propia para la nube analizada. Para la obtención de este descriptor se procede en primer lugar con la determinación de un nuevo marco de referencia (u, v, w) , a partir de este marco se obtienen las relaciones entre cada par de vectores normales de dos puntos pertenecientes a la vecindad del punto de interés, estas relaciones están definidas por la tripleta de los ángulos α , φ , θ y la distancia d (*Ecuación 2.12*). Finalmente es generado un descriptor total que contiene todas las combinaciones posibles de los subhistogramas calculados para cada ángulo característico, obteniéndose un histograma de 125 contenedores; este procedimiento es computacionalmente lento con una complejidad $O(n * k^2)$, el esquema de las relaciones geométricas en la vecindad del punto de interés y la obtención del marco de referencia auxiliar son observadas en las **Figura 2.12(a) y (b)**.

$$\begin{cases} \alpha = v \cdot n_t \\ \varphi = u \cdot \frac{(p_t - p_s)}{d} \\ \theta = \tan^{-1}(w \cdot n_t, u \cdot n_t) \end{cases} \quad \text{Ecuación 2.12}$$

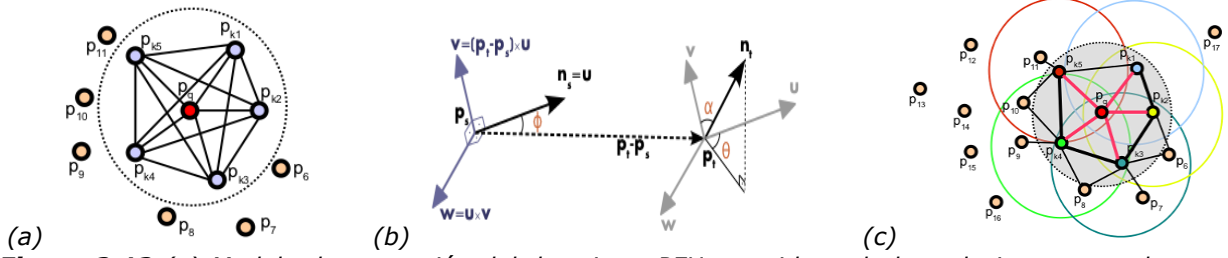


Figura 2.12 (a) Modelo de extracción del descriptor PFH, considerando las relaciones entre los pares de puntos de la vecindad del punto de interés, (b) Extracción de los ángulos y las distancias entre los vectores normales de cada dupleta de puntos. (Radu Bogdan Rusu, Marton, et al., n.d.). (c) Proceso de optimización del descriptor FPFH, considerando las relaciones del punto de interés con sus vecinos, y no entre cada par de puntos de la vecindad. (Radu Bogdan Rusu et al., 2009).

Este descriptor *FPFH* (*Fast Point Feature Histograms*), (Radu Bogdan Rusu et al., 2009) presenta un proceso de optimización de la técnica basada en histogramas de características de puntos con la finalidad de aplicarlo al procesamiento en tiempo real, reduce la complejidad a $O(n * k)$ mediante la obtención de características entre el punto característico del descriptor local y los puntos integrantes de su k -vecindad, a diferencia del método original que considera todas las combinaciones de pares en esa vecindad, el método se divide en 2 etapas, en primer lugar realiza la versión simplificada ya mencionada, la cual es denominada *SPFH*, posteriormente para cada k -vecino se obtiene su respectiva relación *SPFH* y se realiza un promedio ponderado teniendo en cuenta la distancia de cada vecino respectivo hacia el punto de interés (Ecuación 2.13). Es importante resaltar las diferencias entre el método original y esta optimización (**Figura 2.12(c)**), en primer lugar existen regiones exteriores utilizadas para la estimación de características debido a la realización de un proceso simplificado de procesamiento para cada vecino, además se presenta la sobre inclusión de ciertas interconexiones por este mismo fenómeno, el descriptor final contiene 11 contenedores por cada ángulo concatenados en un descriptor final de 33 contenedores.

$$FPFH(p) = SPFH(p) + \frac{1}{k} \sum_{i=1}^k \frac{1}{w_k} SPFH(p_k) \quad \text{Ecuación 2.13}$$

El descriptor *VFH* (*Viewpoint Feature Histogram*), (R.B. Rusu et al., 2010), añade información del punto de vista utilizando el ángulo entre las normales de superficie y el vector central del punto de vista trasladado a cada origen de los vectores normales. Debido a que este es un descriptor global se produce un único descriptor para toda la nube de puntos. El proceso de creación del descriptor consiste en 3 etapas, en primer lugar se realiza el cálculo del centroide de la nube de puntos original, posteriormente se lleva a cabo la generación de un marco de referencia único para cada punto de interés (u, v, w) y se realiza la medición de los ángulos α, ϕ, θ con la misma metodología que *FPFH* y *PFH*, pero entre cada vector normal y el vector desde el punto de vista hasta el centroide de la nube de puntos. Cada uno de los ángulos es empaquetado en un histograma de 45 contenedores y finalmente es calculado el componente de punto de vista (**Figura 2.13**), como fue definido anteriormente y es empaquetado en un histograma de 128 contenedores, dando como resultado un descriptor de 308 contenedores. Este tipo de descriptor es variante a la pose aunque invariante a la escala, lo que permite la recuperación de la pose 6DOF del elemento en la escena.

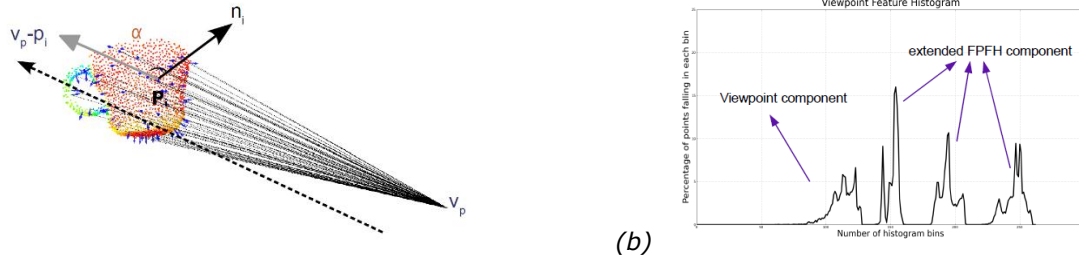


Figura 2.13 (a) Obtención del componente del punto de vista, (b) Histograma global final, donde se indica cada componente. (R.B. Rusu et al., 2010).

El descriptor NARF (Normal Aligned Radial Feature) (Steder & Konolige, n.d.; Steder & Rusu, 2011) hace uso de imágenes de rango y su codificación en píxeles, además utiliza puntos característicos definidos teniendo en cuenta los límites y bordes presentes en esta imagen. En esta técnica primero se lleva a cabo un proceso para la extracción de 3 tipos de bordes en la imagen de rango es decir bordes de objeto, sombra y bordes intermedios (**Figura 2.14(a)**), para esto se utiliza un algoritmo similar a un filtro bilateral y se obtienen ciertas puntuaciones para cada uno de los píxeles circundantes al píxel que se está evaluando (píxeles con cambios bruscos de profundidad en sus alrededores) en una vecindad M ; según estos valores, se clasifica cada píxel de este tipo como alguno de las 3 clases de borde. Una vez ha sido realizada la extracción de bordes, se define los puntos característicos según índices de interés por los autores, donde se tiene en cuenta el ángulo de proyección de las normales de superficie en un plano perpendicular al vector entre el sensor y el punto en cuestión, además la reducción del ruido y la búsqueda de cambios significativos en la curvatura. El cálculo del descriptor NARF se compone de 3 etapas, en primer lugar se obtiene un parche alineado con la normal, considerando un marco de referencia local y analizando la vecindad de puntos denominada tamaño de soporte utilizada en la extracción de puntos característicos, como siguiente paso, se proyecta una estrella de n rayos encima del parche (**Figura 2.14(b)**) y se obtienen las celdas que se encuentran bajo él, con estas celdas se calcula el valor de cada índice del descriptor. Finalmente, se realiza un histograma de orientación final con el que puede llevarse a cabo la desambiguación de la pose 6DOF del elemento en la escena.

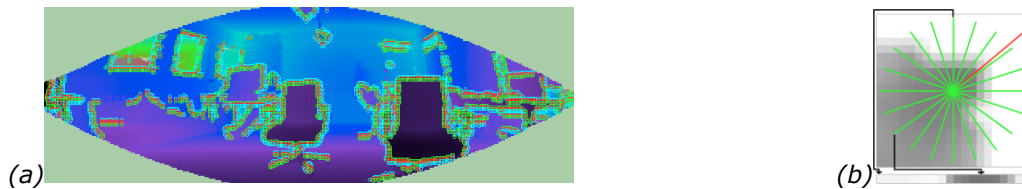


Figura 2.14 (a) Detección de bordes en una imagen de rango, (b) estrella proyectada sobre el punto de interés para la extracción del descriptor NARF. (Steder & Konolige, n.d.; Steder & Rusu, 2011).

2.2.4 CLASIFICACIÓN DE PATRONES

El aprendizaje automático (*machine learning*) y clasificación de patrones puede llevarse a cabo de distintas maneras. Este proceso siempre es realizado usando vectores de características que describen, de la mejor manera posible, los datos disponibles del problema. Normalmente, en el conjunto de datos del problema, se debe disponer de un subconjunto que será usado para el entrenamiento del clasificador y un segundo subconjunto que será usado para la validación del mismo. El aprendizaje automático puede dividirse en aprendizaje supervisado y aprendizaje no supervisado.

El aprendizaje supervisado se caracteriza por la presencia de un agente externo al sistema, al que se le conoce como supervisor o maestro. El problema en el mundo real se describe mediante un conjunto de características que se le presentan al sistema y al maestro simultáneamente. El maestro propone una salida a ese conjunto de datos de entrada (salida deseada), que se compara con la salida ofrecida por el sistema de clasificación. Basándose en esto, se calcula un error, que es usado para corregir los parámetros del sistema de clasificación de manera que su salida se ajuste mejor a la que se espera de él. Este proceso es repetido iterativamente hasta que se alcance un criterio de parada. (Caicedo & López, 2009)

En el aprendizaje no supervisado no existe este agente externo que ofrezca una salida deseada para ser comparada con la salida del sistema de clasificación. Este proceso de aprendizaje se realiza a partir de una serie de observaciones del entorno sobre el que se está trabajando, es decir, las características que describen el problema y que se le presentan como entradas al sistema de clasificación. (Caicedo *et al*, 2009)

Existen técnicas que utilizan distancias entre los vectores de características para realizar la clasificación, es decir, se calculan las distancias l_1 (distancia Manhattan), l_2 (distancia euclidiana), Mahalanobis o cualquier otra métrica para determinar qué tan alejado está el vector de características a evaluar del centroide de un clúster determinado con anterioridad. Durante el proceso de aprendizaje, estas informaciones son usadas para ajustar los clústeres que separan los datos de entrenamiento.

Algunas técnicas, denominadas técnicas de inteligencia computacional, apuntan a la generalización de comportamientos a partir de una información no estructurada que se ingresa en forma de ejemplos, es decir, se lleva a cabo un proceso de inducción del conocimiento. Son notables en este campo las redes neuronales artificiales y las máquinas de soporte vectorial.

Por último, existen técnicas probabilísticas de clasificación de patrones. Estas técnicas están íntimamente relacionadas con el campo de estudio de la estadística y hacen uso de sus métodos para estimar las probabilidades de una instancia específica de pertenecer a una clase en particular. Una técnica notable en este campo es el *k-nearest neighbors (k-NN)* (Silverman & Jones, 1989)

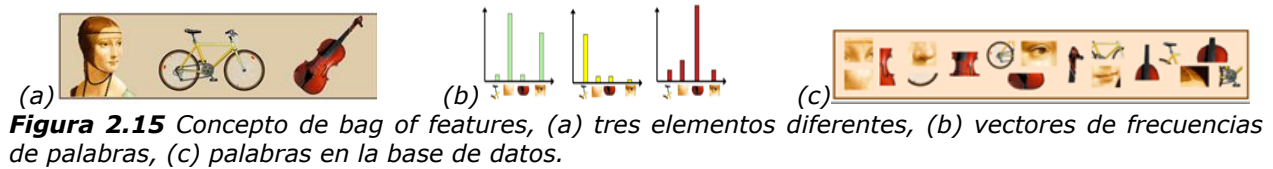
2.2.4.1 Bolsa de Características (Bag of Features)

Es un método de agrupación de características y búsqueda de elementos generales propuesto en (Sivic & Zisserman, 2003), tomando como ejemplo el estilo de búsqueda de la web *Google*. En esta técnica es realizada una cuantificación vectorial de las características locales, mediante la clusterización con *k-means*. Cada clúster es denominado una palabra visual (*visual Word*), y cada vector de características perteneciente a ese clúster es una instancia de esa palabra. Un elemento es obtenido y clasificado mediante el cálculo de su vector de frecuencias de palabras (para cada característica local, es buscado su vecino más cercano k-dimensional). El orden de las retribuciones puede ser utilizado para ajustar la calidad los elementos devueltos. (Sivic & Zisserman, 2003; Steder & Ruhnke, 2011; Toldo, Castellani, & Fusiello, 2009). El método es robusto ante cambios en la geometría y el punto de vista. Es usado en reconocimiento de imágenes y en aplicaciones *SLAM*. Recientemente ha sido utilizado con éxito en la integración de características locales 3D como Surf 3D (Knopp, Prasad, & Willems, 2010), en contraposición a agrupamientos de correspondencias locales geométricos.

En la **Figura 2.15** se observa el concepto de este método. Para cada uno de los 3 diferentes elementos, un único vector de frecuencias de aparición de palabras es calculado, finalmente cada vector se compara con los presentes en las bases de datos, para retribuir su clasificación. Por lo general se incluye también una etapa de ponderación, que permite dar más importancia a las palabras visuales más representativas. Esta ponderación se define como:

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{N_i} \quad \text{Ecuación 2.14}$$

Donde t_i es el componente i del vector de frecuencias de palabras t , n_{id} es el número de ocurrencias de la palabra i en el documento, N es el número de documentos en la base de datos y N_i el número de documentos que tienen la palabra i .



2.2.4.2 K-dimensional tree

Un árbol k-dimensional (Bentley, 1975) es una estructura de partición espacial para la organización de puntos en un espacio k-dimensional. Es un árbol binario en el que cada nodo es un punto k-dimensional. Cada nodo no principal puede ser pensado como la división implícita mediante un hiperplano que divide el espacio en dos partes, conocidas como medio-espacios. Son útiles para la búsqueda en información multidimensional. Los puntos a la izquierda de este hiperplano están representados por el subárbol izquierdo de ese nodo y puntos del lado derecho del hiperplano están representados por el subárbol derecho. La dirección hiperplano se elige de la siguiente manera: cada nodo en el árbol está asociado con una de las k-dimensiones, con el hiperplano perpendicular al eje de esa dimensión.

2.2.4.3 K-NN (K-nearest neighbors) con árboles k-dimensionales.

El algoritmo de búsqueda del vecino más cercano (KNN) tiene como objetivo encontrar el punto en el árbol que está más cerca de un punto de entrada dado. Esta búsqueda se puede hacer de manera eficiente mediante el uso de las propiedades de los árboles para eliminar rápidamente una gran parte del espacio de búsqueda. La búsqueda de un vecino más cercano en un árbol KD se desarrolla de la siguiente manera:

- 1) Comenzando con el nodo raíz, el algoritmo se mueve hacia abajo en el árbol de forma recursiva, de la misma manera que lo haría si se estuviera insertando el punto de búsqueda (es decir, se va a la izquierda o la derecha dependiendo de si el punto es menor o mayor que el nodo actual en la dimensión de división).
- 2) Una vez que el algoritmo llega a un nodo hoja, guarda ese punto como el "mejor actual"
- 3) El algoritmo desenrolla la recursividad del árbol, realizando los pasos siguientes en cada nodo:
 - a) Si el nodo actual está más cerca que la mejor actual, entonces se convierte en el mejor actual.

- b) El algoritmo comprueba si existe cualquier punto en el otro lado del plano de división que está más cerca del punto de búsqueda que el mejor actual. En concepto, esto se hace mediante la intersección del hiperplano de división con una hiperesfera alrededor del punto de búsqueda que tiene un radio igual a la distancia actual más cercana. Ya que los hiperplanos están todos alineados con el eje, este se implementa como una simple comparación para ver si la diferencia entre la división de coordenadas del punto de búsqueda y el nodo actual es menor que la distancia (coordenadas globales) desde el punto de búsqueda para la mejor actual.
- i) Si la hiperesfera cruza el plano, podría haber puntos más cercanos en el otro lado del plano, por lo que el algoritmo se debe mover hacia abajo en la otra rama del árbol desde el nodo actual en busca de puntos más cercanos, siguiendo el mismo proceso recursivo como toda la búsqueda.
- ii) Si la hiperesfera no interseca el plano de división, entonces el algoritmo continúa subiendo el árbol, y toda la rama en el otro lado de ese nodo es eliminada.
- 4) Cuando el algoritmo termina este proceso para el nodo raíz, entonces la búsqueda ha terminado.

2.3 SENSORES Y DISPOSITIVOS

En este capítulo se expone un resumen de los sensores y dispositivos utilizados en la elaboración del presente trabajo. El objetivo es que el lector tenga claridad con respecto a la naturaleza de la información recolectada en el *dataset* construido mediante el conocimiento del hardware del que se hizo uso. La **Tabla 2.8** muestra los sensores y dispositivos utilizados, junto a una breve descripción y algunas de sus características más importantes.

En el presente trabajo de grado, el convertidor USB a RS-232/422/485 es utilizado para la obtención sin pérdidas de los escaneos del láser Sick LMS200 en tiempo real. Esto es debido a los requerimientos de alta velocidad en la captura de datos para vehículos en movimiento. De igual modo, el convertidor ADC EasyCap se utiliza para la digitalización de la información entregada por la cámara Canon VC-C50i.

Tabla 2.8 Resumen de los sensores y dispositivos usados en el desarrollo del sistema de clasificación de vehículos terrestres

Dispositivo	Descripción	Características	
Sensor láser Sick LMS200 	Sistema de medición de información de rango basado en el cálculo del tiempo de vuelo.	Resolución angular, Tiempo de respuesta mínimo y número de escaneos	0.25°-53.33 ms-401 escaneos 0.5°-26.66 ms-181 escaneos 1°-13.33 ms-361 escaneos
		Angulo de escaneo	180° y 100° (A la resolución de 0.25°)
		Precisión de medida	±15 mm
		Electrónica del LMS200	24 Vcc ± 15 , Máximo consumo de corriente 1.8 A
		Interfaz	RS232 9.6 kBd, 19.2 kBd, 38.4 kBd RS422 9.6 kBd, 19.2 kBd, 38.4 kBd, 500 kBd
		Velocidad del motor	75 Hz
Communication Camera VC-C50i 	Cámara de comunicación especializada para conexión en cascada y alto rendimiento con poca iluminación.	Resolución Analógica	Horizontal: 460 líneas de televisor Vertical: 350 líneas de televisor
		Longitud focal	f=3,5 mm (angular) ~ 91 mm (tele) f/1.6 hasta f/4.0
		Rango de paneo	200° (-100° ~ +100°)
		Rango de inclinación vertical	120° (-30° ~ +90°)
		Electrónica de la cámara	13 V _{cc} , 120 ~ 240 V CA, 12,5 W
		Interfaz	RS232, hasta 9 cámaras
		Conectores de salida	BNC, salida de video compuesto

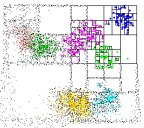
Convertidor USB a RS232/422/485 USB-COMi-M 	Rápida conversión entre los protocolos RS232, RS422/485 a USB (seleccionable).	Interfaz		USB 2.0
		Configuración del dip switch	Modo RS-232	0 1 1 1
			Modo RS-422: 4 cables con <i>handshaking</i>	1 1 1 1
			Modo RS-485: 4 cables <i>full-duplex</i>	1 0 1 1
			Modo RS-485: 2 cables <i>half-duplex</i> con <i>echo</i>	1 0 0 1
			Modo RS-485: 2 cables <i>half-duplex</i> sin <i>echo</i>	1 0 0 0
Convertidor ADC EasyCap 	Permite adquisición de video y audio a través del protocolo USB mediante la digitalización de información proveniente de salidas análogas RCA/BNC.	Interfaz		USB 2.0
		Captura de video		Hasta 720x576px @ 25fps (PAL) y 720x480px @ 30fps (NTSC)
		Entradas		CVBS, S-VIDEO, AUDIO(L), AUDIO(R)
		Audio		AC'97 audio (48 kHz tasa de muestreo), o micrófono mono de audio (8 kHz tasa de muestreo)

2.4 LIBRERÍAS Y SOFTWARE

Así como es necesario conocer el hardware utilizado en la concepción y diseño del presente sistema de clasificación, es igual de importante conocer el software del que se hizo uso. La **Tabla 2.9** muestra las librerías y herramientas software utilizadas en el presente trabajo.

Tabla 2.9 Librerías y herramientas software usadas en el desarrollo del sistema de clasificación de vehículos terrestres

Software / Librería	Descripción y referencia	Referencia
Lenguaje C++ 	Lenguaje de propósito general, situado entre el alto y bajo nivel de abstracción, diseñado por Bjarne Stroustrup.	(Stoustrup, 1997)
OpenCV (Open Computer Vision) 	Librería de visión computacional escrita en lenguaje C++ y ofrecida bajo la licencia BSD.	("OpenCV," 2013)
Eigen Library 	Librería de álgebra lineal implementada en C++ y liberada bajo la licencia <u>MPL2</u> . Posee matrices, vectores, solucionadores numéricos y algoritmos relacionados.	("Eigen," 2013)
PCL (Point Cloud Library) 	Proyecto independiente, elaborado en C++ y ofrecido con licencia BSD, para el procesamiento de imágenes 2D y 3D, así como nubes de puntos.	("About - Point Cloud Library (PCL)," 2013)
The SICK LIDAR MATLAB/C++ 	Proyecto, disponible bajo licencia BSD, enfocado en entregar controladores C++ estables y de fácil uso para los sensores laser Sick LMS2xx y Sick LD, además de una interfaz MEX para el trabajo y la adquisición en tiempo real en la herramienta software Matlab.	(Derenick & Miller, 2008)

ANN Library (Approximate Nearest Neighbors) 	Librería escrita en C++, para la búsqueda exacta y aproximada de vecinos cercanos en espacios de varias dimensiones.	("ANN - Approximate Nearest Neighbor Library," 2013; Mount, 2010)
OpenMP 	El API OpenMP soporta programación paralela multiplataforma de memoria compartida en C/C++ y Fortran.	("OpenMP.org » About the OpenMP ARB and OpenMP.org," 2013)
MPlayer 	Reproductor de video que permite la captura de video en tiempo real. Está disponible bajo licencia GPLv2.	("MPlayer - The Movie Player," 2013)
GCC, The GNU compiler collection 	La colección de compiladores de GNU incluye interfaces para C, C++, Objective-C, Fortran, Java, Ada y Go, así como bibliotecas para estos lenguajes (libstdc++, libgcj,...).	("GCC, the GNU Compiler Collection - GNU Project - Free Software Foundation (FSF)," 2013)
Boost C++ Libraries 	Librerías que proveen soporte para tareas como procesamiento multi-hilo, gestión de archivos del sistema y entorno de carpetas, generación de números aleatorios y procesamiento de imágenes. Se encuentra disponible bajo su propia licencia BSL.	("Boost C++ Libraries," 2013)



3 DESARROLLO DE INGENIERÍA

3.1 ELABORACIÓN DEL DATASET

Para realizar el entrenamiento y validación del sistema multimodal de clasificación de vehículos, se hace necesario el uso de un *dataset* en el que se incluyan los datos de un sensor de rango (láser Sick LMS200) y los datos del sensor de intensidad (cámara VC-C50i). Se decidió construir este dataset en una de las porterías del campus Meléndez de la Universidad del Valle, más específicamente en la Portería Carrefour o Ingreso vehicular Calle 16. La construcción de este *dataset* se hizo necesaria debido a la ausencia de un dataset integrado basado en información 2D-3D para vehículos terrestres variados.

La distribución física del espacio y ciertas restricciones y condiciones en el uso de esta portería fueron las razones para la escogencia de la misma. Con respecto a estas ventajas, puede puntualizarse lo siguiente:

- Se escogió el carril de entrada de vehículos en esta portería debido a la presencia de dos policías acostados sucesivos en el mismo, lo que restringe las velocidades de los vehículos a velocidades aproximadamente constantes.
- la presencia de árboles en esta zona no implica una pérdida de visibilidad sobre el carril de entrada desde la vía de salida, lo que permite disponer de una distancia considerable entre la cámara y el carril objetivo. Al mismo tiempo, la disposición de espacio permite disponer de una proximidad adecuada entre el láser y los vehículos.
- Hay presencia de dos casetas de vigilancia (la portería propiamente dicha y una segunda caseta, para el parqueadero de motocicletas), lo que da la posibilidad de dos puntos de conexión eléctrica bastante convenientes para el montaje realizado, detallado más adelante.
- Se presenta una interesante diversidad en el tipo de vehículos que ingresan a través de esta portería a la Universidad del Valle.

3.1.1 DESCRIPCIÓN DEL ESQUEMA Y MONTAJE

El montaje realizado para la toma de datos fue llevado a cabo en miras de obtener una plataforma de trabajo adecuada para la obtención de los datos necesarios en el presente trabajo de clasificación. La **Figura 3.1** muestra un diagrama de bloques del sistema de adquisición de datos para la construcción del *dataset*.

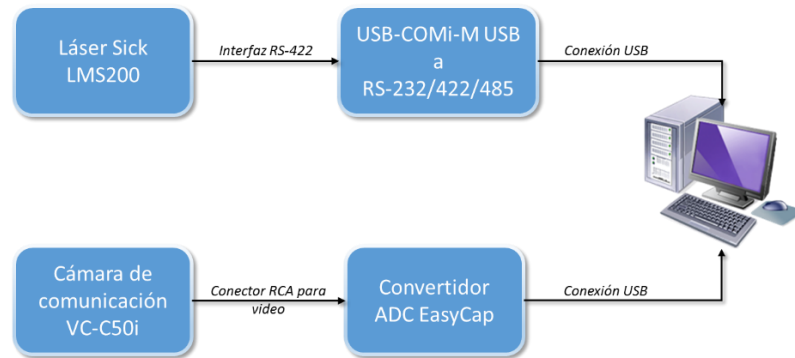


Figura 3.1 Diagrama de bloques del proceso de adquisición de datos.

En primer lugar, observando la **Figura 3.2**, se dispuso el sensor de rango a una distancia de aproximadamente 2.0 metros del centro del carril de entrada hacia el lado de la portería, apuntando hacia el centro y cuidando que el plano de escaneo del mismo fuera lo más perpendicular posible al plano de la carretera y que el vector normal al plano de escaneo del sensor fuera lo más paralelo posible a la dirección de desplazamiento de los vehículos. Esta disposición del láser se realizó con el objetivo de que los planos de escaneo que se realizaron a los vehículos fueran transversales a los mismos y abarcaran la mayor porción de la superficie posible. El láser se montó sobre un soporte compuesto de dos varas de hierro extensibles que permiten elevarlo una altura determinada que, para el caso del presente dataset, fue de 2.75 metros.

Luego, en el extremo opuesto, es decir, en el borde opuesto del carril de salida, se dispuso la cámara cuidando que su eje focal quedara lo más paralelo posible al plano de la carretera y lo más perpendicular posible a la dirección de desplazamiento de los vehículos, apuntando hacia el carril de entrada. La cámara se montó, a su vez, sobre un soporte compuesto de una vara de hierro extensible que permite elevarla una altura determinada que, para el caso del presente dataset, fue de 2.82. Esta disposición de la cámara permite obtener una toma de la escena bastante amplia, procurando, de esta manera, que los vehículos se encuentren al interior del plano de imagen durante un mayor número de fotogramas. .

La **Figura 3.2** muestra la configuración física del montaje del láser y la cámara y su distribución en el espacio donde se tomaron las muestras.

El láser se conecta al computador mediante una interfaz de alta velocidad RS-422, que se configura con ayuda del convertidor USB-COMi-M, el cual convierte la señal a USB para ser leída en el PC.

La cámara se conecta al computador mediante un cable de video común (conector RCA) que va al dispositivo ADC EasyCap, el cual convierte la señal de video a USB para ser leída en el PC.

Los datos se tomaron y procesaron en una computadora Asus N46VJ con un procesador Intel® Core™ i7-3630QM @ 2.4GHz (8 CPUs), con 8GB de memoria RAM DDR3. Las conexiones se realizan a través de puertos USB 3.0.

Aparte de estas conexiones, también deben ser tenidas en consideración las conexiones de potencia de los dispositivos. Estas fueron logradas utilizando extensiones industriales de 6 y 18 metros de largo.

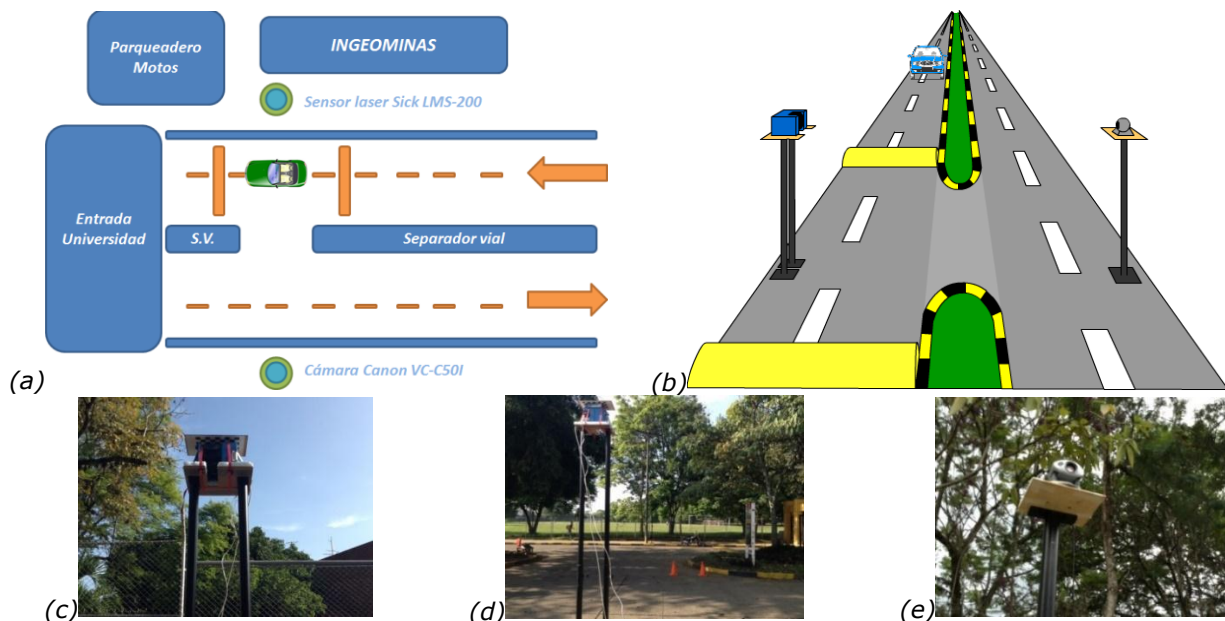


Figura 3.2 (a) Esquema de distribución de los sensores, para la toma del dataset, en la entrada de la universidad. (b) Esquema artístico del montaje. (c)(d)(e) fotografías reales de la configuración física de los sensores para la toma del dataset.

3.1.2 LIMITACIONES Y CONDICIONES

Hubo diversas situaciones que impusieron limitaciones y condiciones en el funcionamiento del sistema de adquisición de datos de rango y de intensidad. Estas situaciones pueden ser puntualizadas como sigue:

- Condiciones climáticas desfavorables (lluvia por peligro a los equipos). Por esta razón, el *dataset* no presenta datos que permitan evaluar el desempeño del sistema bajo estas condiciones.
- Oclusión de los objetos en la toma de video (peatones que se ubican entre la cámara y el vehículo o varios vehículos cruzando la escena uno al lado del otro). Esto puede generar errores en la etapa de la estimación de velocidad del vehículo y puede hacer que el dato sea inválido para la clasificación visual. Estos datos fueron descartados del *dataset* final por medio de una revisión manual.
- Oclusión de los objetos en la toma de rango (múltiples vehículos que cruzan el plano de escaneo al mismo tiempo). Estas situaciones provocan que el dato sea inválido para la clasificación mediante la información de rango. Estos datos fueron descartados del *dataset* final por medio de una revisión manual.
- Velocidades de movimiento de los vehículos muy altas. Esta limitación se mantuvo bajo control gracias a la disposición física del espacio, característica que se explicó anteriormente.
- No es posible ubicar los sensores sin inclinación alguna con respecto al plano de la carretera. Con respecto a la inclinación del láser, es corregida mediante software. En el caso de la cámara, se asumió que esta situación no producía un efecto apreciable en las mediciones realizadas (velocidad, clasificación).
- Variabilidad en las diferentes tomas de los datos, debido a diferentes posiciones de ubicación de los soportes para el láser y la cámara en el espacio. Estas ubicaciones fueron determinadas arbitrariamente en cada una de las sesiones de toma de datos. Se comprobó que esta condición no afecta, sin embargo, el desempeño del sistema.

3.1.3 ESTRUCTURA DE DIRECTORIOS

Los datos obtenidos se encuentran en una estructura de directorios, que define el número de la toma y especifica la información relacionada con el tiempo de captura, siguiendo el estándar internacional *ISO8601*. Cada directorio contiene seis archivos.

El primer archivo, **data(N).avi/.mp4**, contiene la información de los fotogramas de video en formato *AVI* o formato *MPGE-4*. El segundo archivo, **data(N).dat**, contiene la información de rango y los tiempos de captura de cada escaneo. El tercer archivo, **data(N)Vid.dat**, contiene los tiempos de adquisición exacto de cada fotograma de video, de nuevo, siguiendo el estándar *ISO8601*. El cuarto archivo, **data(N)Pos.dat**, contiene información sobre la distancia recorrida por los vehículos en los fotogramas en los que se encuentran en la escena. El quinto archivo, **data(N)Vel.dat**, contiene información sobre la velocidad de los vehículos en los fotogramas en los que se encuentran en la escena. Por último, el sexto archivo, **data(N)VelProm.dat**, contiene información sobre los promedios de velocidad de los vehículos en los fotogramas en los que se encuentran en la escena.

- *DatasetFinal*
 - *Data(N) (DD-MM-YYYY_ hh_mm_ss)*
 - *data(N).avi*
 - *data(N).dat*
 - *data(N)Vid.dat*
 - *data(N)Pos.dat*
 - *data(N)Vel.dat*
 - *data(N).VelProm.dat*
 - *Data(N+1)(DD-MM-YYYY_ hh_mm_ss)*
 - *data(N+1).avi*
 - *data(N+1).dat*
 - *data(N+1)Vid.dat*
 - *data(N+1)Pos.dat*
 - *data(N+1)Vel.dat*
 - *data(N+1).VelProm.dat*

La información en el archivo **data(N).dat**, que corresponde a la información de rango del *dataset* y sus características de obtención, es entregada en una matriz de **$N \times (M+6)$** , donde ***N*** corresponde al número total de escaneos y ***M*** el número de medidas por escaneo, adicionando 6 campos de características de la toma. Este archivo tiene la forma:

<i>Tipo de sensor</i>	<i>Ángulo de escaneo</i>	<i>Modo de medición</i>	<i>Resolución del sensor</i>	<i>Número de medidas</i>	<i>Tiempo del escaneo</i>	<i>Data1</i>	<i>...</i>	<i>DataN</i>
-----------------------	--------------------------	-------------------------	------------------------------	--------------------------	---------------------------	--------------	------------	--------------

El campo tipo de sensor define la clase a la que pertenece el dispositivo entre las opciones de sensado láser que provee la empresa **SICK**. El ángulo de escaneo puede ser 100° o 180°, dependiendo de la configuración que se elija. El modo de medición indica si el sensor debe medir en modo centímetros o en modo milímetros, además de permitir cambiar la distancia máxima permitida entre 8 y 80 metros. La resolución puede ser de 0.25°, 0.5° y 1° según se configure. El siguiente campo indica la cantidad de medidas por escaneo. Posteriormente se muestra el tiempo de toma del escaneo, este tiempo es relativo y cuenta la cantidad de segundos transcurridos desde el inicio de la ejecución del software de adquisición del dato. Finalmente, se consignan los valores de rango para cada rayo láser (medida).

El formato de **data(N)Vid.dat** consiste en un vector en el que se consigna, en cada posición, el tiempo de adquisición de cada fotograma de video del archivo con formato **AVI** o **MPGE-4**. Este archivo tiene la forma:

Tiempo de adquisición fotograma 1	Tiempo de adquisición fotograma 1	...	Tiempo de adquisición fotograma N
-----------------------------------	-----------------------------------	-----	-----------------------------------

El formato de **data(N)Pos.dat** consiste en un encabezado que demarca las columnas y posteriormente los datos consignados en forma de matriz, con tantas filas como fotogramas hayan sido detectados conteniendo al vehículo y 4 columnas. Cada fila contiene, de izquierda a derecha, el tiempo de adquisición relativo al inicio de la ejecución del software de adquisición, la posición recorrida por el vehículo en metros, la posición recorrida por el vehículo en kilómetros y el número de fotograma de aquellos fotogramas en los que se detecte el vehículo en la escena (i...j)

sec	m	km	frame
Tiempo de adquisición del fotograma i	Posición, en metros, recorrida por el vehículo en el fotograma i	Posición, en kilómetros, recorrida por el vehículo en el fotograma i	i
...	...	...	...
Tiempo de adquisición del fotograma j	Posición, en metros, recorrida por el vehículo en el fotograma j	Posición, en kilómetros, recorrida por el vehículo en el fotograma j	j

El formato de **data(N)Vel.dat** consiste en un encabezado que demarca las columnas y posteriormente los datos consignados en forma de matriz, con tantas filas como fotogramas hayan sido detectados conteniendo al vehículo y 4 columnas. Cada fila contiene, de izquierda a derecha, el tiempo de adquisición relativo al inicio de la ejecución del software de adquisición, la velocidad del vehículo en $\frac{m}{s}$, la velocidad del vehículo en $\frac{km}{h}$ y el número de fotograma de aquellos fotogramas en los que se detecte el vehículo en la escena (i...j)

sec	m/s	km/h	frame
Tiempo de adquisición del fotograma i	Velocidad, en $\frac{m}{s}$, del vehículo en el fotograma i	Velocidad, en $\frac{km}{h}$, del vehículo en el fotograma i	i
...	...	...	...
Tiempo de adquisición del fotograma j	Velocidad, en $\frac{m}{s}$, del vehículo en el fotograma j	Velocidad, en $\frac{km}{h}$, del vehículo en el fotograma j	j

El formato de **data(N)VelProm.dat** consiste en un encabezado que demarca las columnas y posteriormente los datos consignados en forma de matriz, con tantas filas como fotogramas hayan sido detectados conteniendo al vehículo y 3 columnas. Cada fila contiene, de izquierda a derecha, el tiempo de adquisición relativo al inicio de la ejecución del software de adquisición, el promedio acumulado de velocidad simple (en $\frac{km}{h}$) y el promedio acumulado de la velocidad suavizada con Kalman (en $\frac{km}{h}$) de aquellos fotogramas en los que se detecte el vehículo en la escena (i...j)

sec	mean	kalman
Tiempo de adquisición del fotograma i	Velocidad promedio acumulada, en $\frac{km}{h}$, del vehículo en el fotograma i	Velocidad suavizada promedio acumulada, en $\frac{km}{h}$, del vehículo en el fotograma i
...	...	...
Tiempo de adquisición del fotograma j	Velocidad promedio acumulada, en $\frac{km}{h}$, del vehículo en el fotograma j	Velocidad suavizada promedio acumulada, en $\frac{km}{h}$, del vehículo en el fotograma j

3.2 CALIBRACIÓN DE LA CÁMARA CANON VC-C50I CON OPENCV Y MATLAB

En las secciones 3.2 y 3.3 se desarrollan los procesos de calibración de los sensores. Esta

etapa es necesaria para realizar la rectificación y el ajuste de la información en bruto que proviene de los mismos. Vale aclarar que este proceso es realizado antes de proceder con el análisis de la información y de la extracción de descriptores.

El proceso de calibración fue realizado tanto en lenguaje C++ mediante la librería *OpenCV*, como en el software Matemático Matlab. Lo anterior con el fin de realizar una comparación en las matrices de parámetros intrínsecos y los parámetros de distorsión que mejor solucionaran el inconveniente de las imperfecciones de fabricación presente en el dispositivo. Las matrices de calibración obtenidas en cada caso son presentadas en la **Tabla 3.1**

El proceso de calibración permite atenuar los efectos de dos tipos de distorsiones en la información 2D, denominados Distorsión tangencial y Distorsión radial, además establece la relación entre la información 3D **xyz** y el plano de imagen **uv**. El proceso de minimización es basado en imágenes de patrones conocidos (en este caso un patrón de tipo ajedrez) tomadas desde distintas perspectivas y orientaciones. Con la calibración realizada se obtiene también un modelo extrínseco cuyo origen (marco global) está determinado por una de las esquinas del patrón de calibración.

Los resultados de calibración se presentan en la **Figura 3.3**. La **Figura 3.3(a)** muestra una imagen a rectificar, usando los parámetros obtenidos en los procedimientos en *OpenCV* y *Matlab*. La **Figura 3.3(b)** muestra el resultado de calibración con *OpenCV*. La **Figura 3.3(c)** muestra el resultado de calibración con *Matlab*. Los parámetros, resultado de los procesos de calibración, tanto en *OpenCV* como en *Matlab*, son mostrados en la **Tabla 3.1**

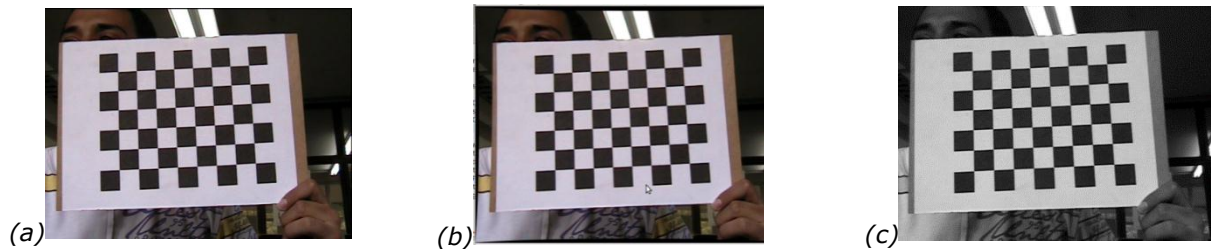


Figura 3.3 (a) Imagen original. (b) Resultado de la calibración usando *OpenCV*. (c) Resultado de la calibración usando *Matlab*.

Tabla 3.1 Información de los parámetros de calibración obtenidos con *OpenCV*.

PARÁMETROS OPENCV	VALOR		PARÁMETROS OPENCV	VALOR	
	OpenCV	Matlab		OpenCV	Matlab
f_x	8.7793014076447719 $e + 02$	849.26593 ± 2.75870	k_1	-1.7560241156970879 $e - 01$	-0.18342 ± 0.03253
f_y	8.7793014076447719 $e + 02$	879.84977 ± 2.84829	k_2	3.0101084991944321 $e - 00$	0.50427 ± 0.0407450
c_x	3.1950000000000000 $e + 02$	299.45146 ± 6.59032	p_1	0	0.00080 ± 0.0001340
c_y	2.3950000000000000 $e + 02$	209.08019 ± 4.84782	p_2	0	0.00000 ± 0.00000
k_3	-1.7325723540153990 $e + 00$	-0.00556 ± 0.00165			

Con el modelo en *OpenCV* se obtuvieron los mejores resultados, pues se elimina significativamente la distorsión radial presente en la información original. Esto se observa en la rectificación de los bordes laterales, que originalmente presentan un comportamiento curvo.

3.3 CALIBRACIÓN DEL SENSOR LÁSER SICK LMS200 EN MATLAB Y SICKTOOLBOX

El proceso de calibración del sensor laser es llevado a cabo siguiendo el esquema presentado en (Borenstein, 2002) y resumido en la **Figura 3.(a)**. Las imágenes de la disposición de la plataforma de calibración se observan en la **Figura 3.(b)**.

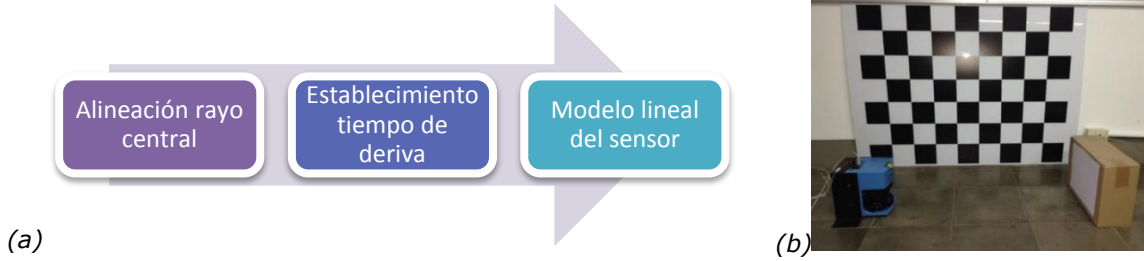


Figura 3.4 (a) Esquema del proceso de calibración del sensor laser Sick LMS200. (b) Plataforma de calibración del sensor laser Sick LMS200.

3.3.1 ALINEACIÓN EJE CENTRAL

En primer lugar es realizado un proceso de alineamiento del eje central sobre el cual se desplazara el láser para la ejecución de las siguientes pruebas y el rayo central del dispositivo; teóricamente de los 181 rayos provenientes del láser a una resolución de 1°, el rayo 91 debería ser la medida central, sin embargo en la práctica, el haz central tiende a encontrarse en otro número de rayo, para determinar el número correcto se procede de la siguiente manera.

1. Situar el láser a una distancia $d1$ y tomar el promedio de 1000 escaneos (guardar en vector $dm1$) para los rayos:

$$\frac{(\text{Número escaneos} + 1)}{2} - 10 \rightarrow \frac{(\text{Número escaneos} + 1)}{2} + 10$$

2. Situar el láser a una distancia $d2$ y tomar el promedio de 1000 escaneos (guardar en vector $dm2$) para los rayos:

$$\frac{(\text{Número escaneos} + 1)}{2} - 10 \rightarrow \frac{(\text{Número escaneos} + 1)}{2} + 10$$

3. Restar los dos vectores de 21 datos ($dm3 = dm2 - dm1$, $dm1$ y $dm2$ observados en la **Figura 3.5(a)** para un caso de 11 datos), y observar el resultado de $dm3(\text{longitud}(dm3)/2)$, si este es mayor que la distancia real desplazada ($d2 - d1$), cambiar la inclinación frontal del sensor hasta alcanzar una diferencia mínima.
4. Una vez logrado lo anterior, buscar en $dm3$ el valor mínimo para los 21 datos, obtener el índice y ese será el rayo central del láser, que se utilizará para el resto de pruebas. En nuestro caso se obtuvo que el número de rayo que corresponde al central es el 89.

3.3.2 MODELO LINEAL DEL SENSOR LASER

Existe una relación lineal entre la distancia medida y la distancia real hacia el patrón de medida, como es demostrado en (Borenstein, 2002); esta relación es definida por las ecuaciones *Ecuación 3.1*, *Ecuación 3.2*, *Ecuación 3.3*. Para la toma de datos que definan el

modelo, se realizaron mediciones cada 20 cm desde el origen del sensor hasta 2 metros de distancia, los resultados de la calibración se hacen evidentes en la **Figura 3.5(b)** donde se observa como el modelo calibrado presenta errores máximos de 5.5 mm en comparación al error máximo no calibrado de 20 mm.

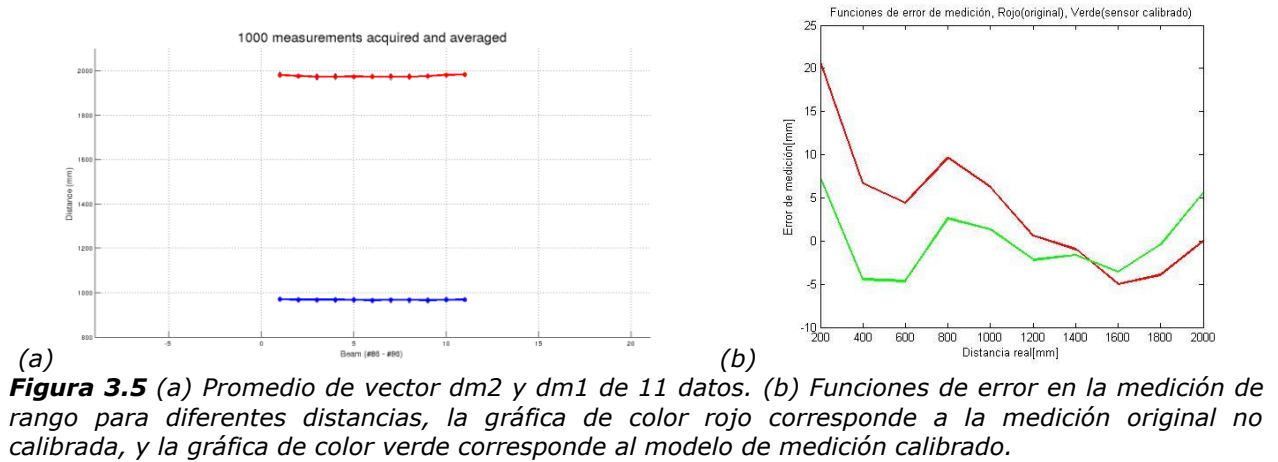


Figura 3.5 (a) Promedio de vector $dm2$ y $dm1$ de 11 datos. (b) Funciones de error en la medición de rango para diferentes distancias, la gráfica de color rojo corresponde a la medición original no calibrada, y la gráfica de color verde corresponde al modelo de medición calibrado.

$$\hat{y} = m\mu + b \quad \text{Ecuación 3.1}$$

$$m = \frac{\sum_{i=1}^n (y_i - \bar{y})(\mu_i - \bar{\mu})}{\sum_{i=1}^n (\mu_i - \bar{\mu})^2} = 0.98453 \quad \text{Ecuación 3.2}$$

$$b = \bar{y} - m\bar{\mu} = 15.233 \quad \text{Ecuación 3.3}$$

3.4 DISEÑO Y DESARROLLO DEL MÓDULO DE EXTRACCIÓN DE CARACTERÍSTICAS Y CLASIFICACIÓN DE VEHÍCULOS CON INFORMACIÓN VISUAL

El módulo de extracción de características visuales y clasificación por medio de imágenes de intensidad fue construido de manera totalmente modular, con ayuda de la librería *OpenCV*, descrita con anterioridad. De esta manera, cualquier subproceso que necesite ser cambiado, puede ser modificado individualmente sin afectar otros procesos independientes.

El esquema mostrado en la **Figura 3.6** describe los diferentes procesos que se siguen con el objetivo final de obtener una clasificación de una instancia de una clase (vehículo) en una secuencia de video.

Para dar claridad al lector, sobre el resultado final que se busca, la **Figura 3.7** presenta una instancia de cada una de las clases que se pretenden separar.

Se puede apreciar la similitud de varias de las clases (e.g. Sedán, Hatchback, Campero, SUV), lo que hace la tarea de separación entre estas un proceso bastante complejo. A continuación se describe, entonces, la metodología utilizada para llevar a cabo esta tarea.

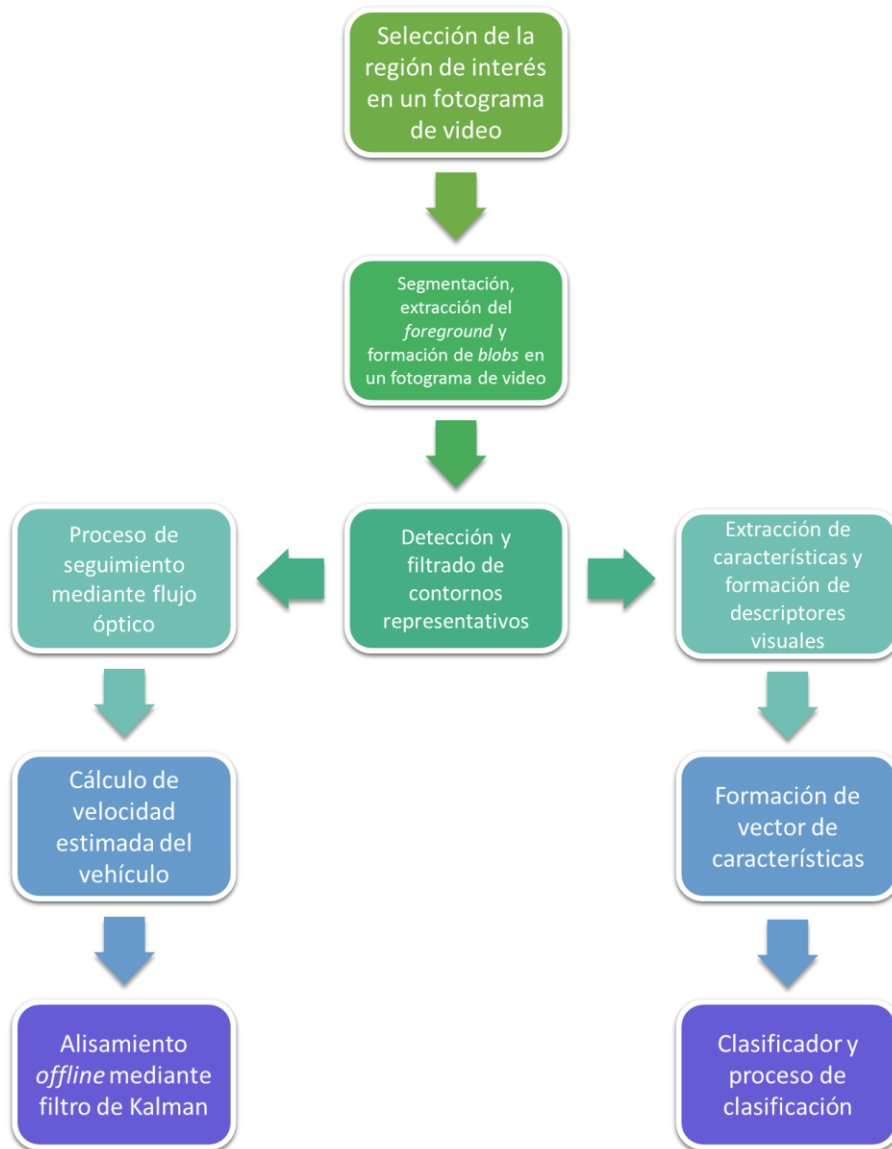


Figura 3.6 Diagrama de bloques del módulo de clasificación basado en información visual (Secuencias de video, imágenes de intensidad)

El software escrito está compuesto de distintos segmentos de código que implementan cada una de las anteriores etapas. En primer lugar, se tienen dos clases que encapsulan las funcionalidades de más alto nivel, *i.e.* El procesamiento de la secuencia de video y el procesamiento de cada fotograma. Estas clases fueron llamadas el *VideoProcessor* y el *FrameProcessor*.

El *VideoProcessor* da acceso a las funcionalidades que permiten la lectura y visualización adecuada de los fotogramas que componen el archivo de video obtenido del *dataset*. Esta clase tiene una interfaz que le permite la comunicación con el *FrameProcessor*.

El *FrameProcessor* encapsula las funcionalidades de más bajo nivel requeridas a nivel de fotogramas y pixels, que permiten el procesamiento cada una de las imágenes que componen la secuencia de video, es decir, aquellas mostradas en la **Figura 3.6**.

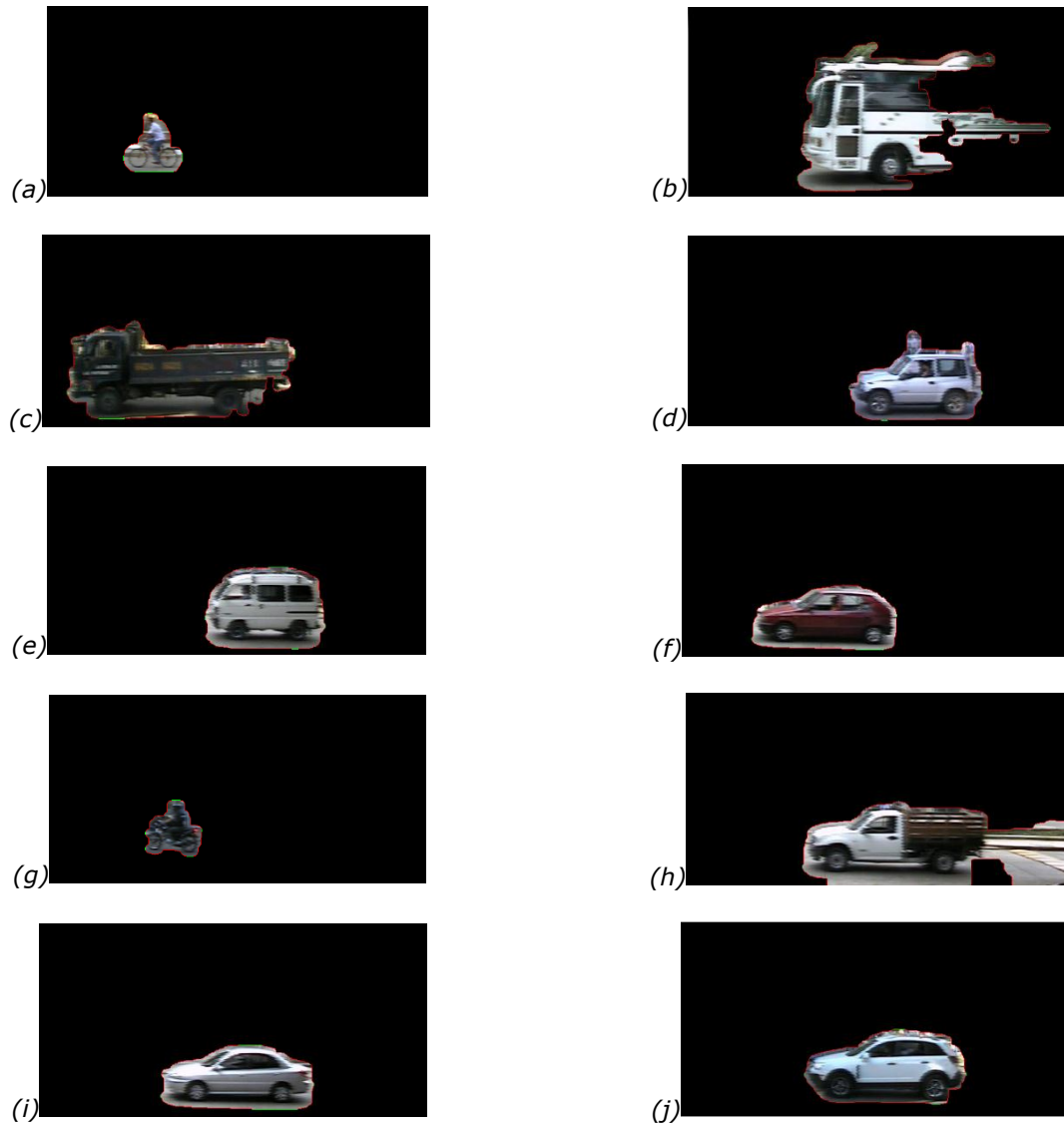


Figura 3.7 Clases a separar mediante el módulo visual: (a) Bicicleta. (b) Bus. (c) Camión. (d) Campero. (e) Furgoneta. (f) Hatchback. (g) Moto. (h) Pickup. (i) Sedan. (j) SUV.

3.4.1 SELECCIÓN DE LA REGIÓN DE INTERÉS (ROI) EN UN FOTOGRAMA

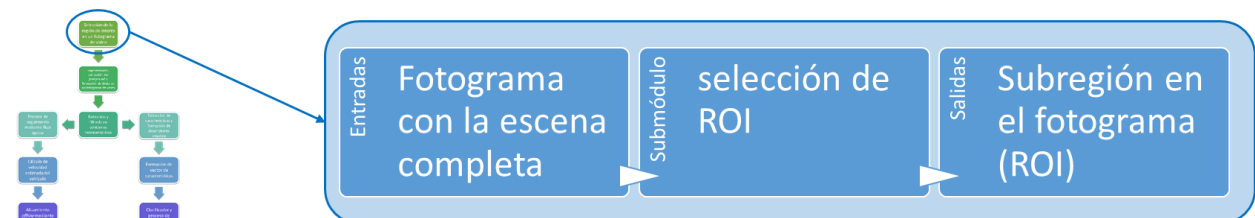


Figura 3.8 Diagrama de entradas y salidas para el submódulo de selección de ROI.

Debido a que a lo largo de todo el *dataset*, es decir, durante todo el proceso de obtención de los datos, la posición del sensor con respecto a la escena fue más o menos constante, las secuencia de video obtenidas tienen una relativa consistencia en relación a la región de la

escena en el mundo real registrada. La **Figura 3.9(a)** muestra una toma típica de la cámara hecha en la escena.



Figura 3.9 (a) Escena completa tomada por el sensor de intensidad. (b) Región de interés recortada del fotograma original.

Se puede observar que, conociendo de antemano y considerando la dimensión máxima de los vehículos que serán parte del *dataset*, es posible tomar una subregión de la escena que será el área efectiva del fotograma al que se le aplicarán los posteriores procesamiento. Se determinó de manera heurística que, de los fotogramas de tamaño 640x480 pixels, el área que contiene la Región de Interés (*Region Of Interest -ROI-*) estaba contenida en un rectángulo de tamaño 640x320 pixels con origen x, y en (0,60). Esto quiere decir que el área de interés se encuentra contenida entre las coordenadas $x=[0,640]$ y $y=[60,380]$, logrando de esta manera una reducción de 160 pixels en la dirección vertical, lo que representa una reducción total de 102400 pixels. Esto implica un importante ahorro en la carga computacional que implican los posteriores algoritmos. La región de interés está marcada mediante un rectángulo verde en la **Figura 3.9(a)** y el recorte final de la escena se muestra en la **Figura 3.9(b)**.

3.4.2 SEGMENTACIÓN, EXTRACCIÓN DEL FOREGROUND Y FORMACIÓN DE BLOBS EN UN FOTOGRAMA

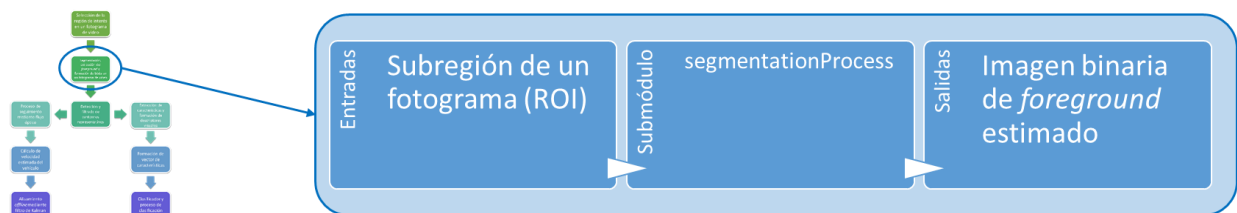


Figura 3.10 Diagrama de entradas y salidas para el submódulo *segmentationProcess*.

El submódulo *segmentationProcess* recibe como entrada una región (ROI) de un fotograma de la secuencia de video y entrega como salida de su procesamiento una imagen binaria de igual tamaño conteniendo la segmentación entre el fondo y el *foreground*, codificado en blobs o “manchas” que contienen los elementos detectados como tal.

El proceso de segmentación entre el fondo y el foreground es mostrado en la **Figura 3.11**.

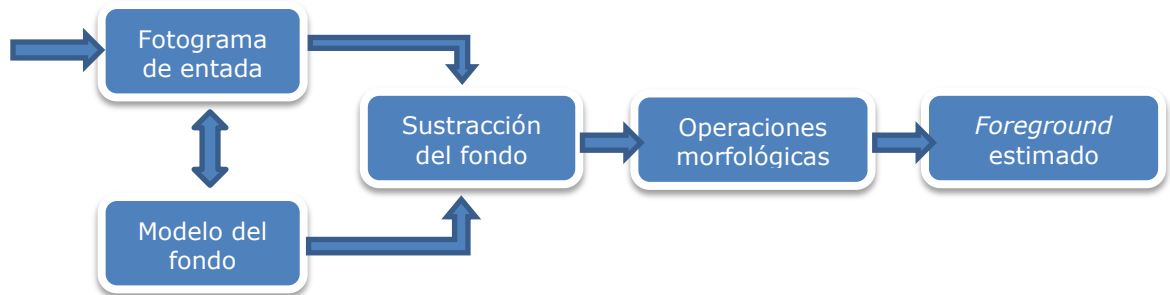


Figura 3.11 Diagrama de bloques que describe el proceso de segmentación entre el fondo de la imagen y el foreground.

En el presente trabajo se hizo uso de *Mixture of Gaussians* para realizar el modelado del fondo. Esta técnica se encuentra implementada en *OpenCV* y su utilización es bastante simple, requiriendo como único parámetro de control una tasa de aprendizaje que determina qué tan rápido son incorporadas o sustraídas las estimaciones gaussianas sobre el fondo. Esta tasa de aprendizaje se modificó de acuerdo a la clase procesada para obtener resultados satisfactorios y es constante para todas las instancias de cada clase contenidas en el *dataset*. Los valores utilizados se muestran en la **Tabla 3.2**.

Una vez llega un fotograma de entrada, se convierte la imagen a escala de grises, se actualiza el modelo del fondo con esta información y se realiza una resta entre estas dos imágenes. El resultado de esta operación se somete a una umbralización que da como resultado una imagen binaria con los pixels de *foreground* estimados en blanco (valor 255) y los pixels de fondo estimados en negro (valor 0), tal como se observa en la **Figura 3.12(a)**

A la imagen binaria obtenida se le realizan dos operaciones morfológicas con el fin de obtener *blobs* que representan los objetos de *foreground* estimados. Las operaciones morfológicas utilizadas fueron el *cierre* y la *dilatación* (véase *Capítulo 2*), en ese orden para favorecer la eliminación de pequeñas regiones aisladas. De igual modo que la tasa de aprendizaje para el algoritmo de modelado del fondo, los tamaños de los elementos estructurantes para estas operaciones fueron ajustados de acuerdo a la clase para obtener resultados satisfactorios. Los valores utilizados se muestran en la **Tabla 3.2**.

Tabla 3.2 Tamaño de los elementos estructurantes para el cierre y la dilatación, de acuerdo a la clase procesada.

Clase	Tasa de aprendizaje utilizada	Tamaño elemento para cierre	Tamaño elemento para dilatación
Bicicleta	0.02	10	8
Bus	0.00955	10	8
Camión	0.005	10	8
Campero	0.00955	15	8
Furgoneta	0.00955	10	8
Hatchback	0.009	15	7
Moto	0.02	7	7

El *foreground* obtenido después de seguir estos procesos es mostrado en la **Figura 3.12(b)**. Se puede observar que, además del verdadero objeto de interés, permanecen en la imagen algunos *blobs* remanentes que no representan información de interés. Este ruido será eliminado en la siguiente etapa del proceso.

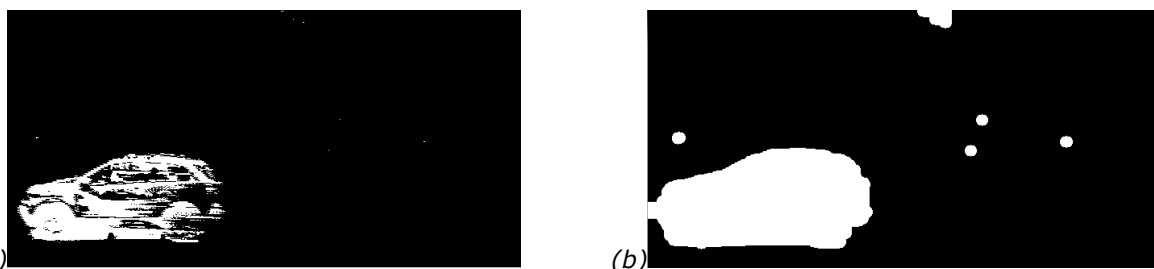


Figura 3.12 Foreground estimado en una escena (a) después de realizar la sustracción del fondo, (b) después de realizar operaciones morfológicas.

3.4.3 DETECCIÓN Y FILTRADO DE CONTORNOS REPRESENTATIVOS

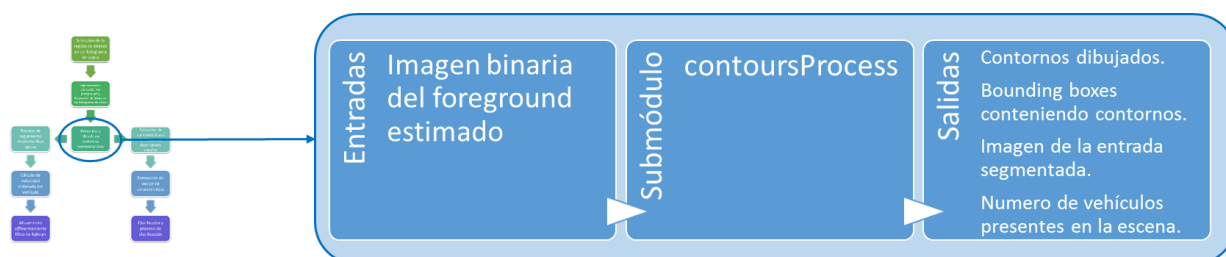


Figura 3.13 Diagrama de entradas y salidas del submódulo contoursProcess.

El submódulo *contoursProcess* recibe como entrada la imagen binaria generada por el módulo *segmentationProcess* y entrega como salidas, los contornos que corresponden a objetos de interés (vehículos) dibujados en el fotograma de entrada (ROI), así como una región rectangular que los contiene, una imagen de la entrada segmentada y el número de vehículos presentes en la escena.

A partir de los *blobs* generados en la etapa anterior se extraen los contornos presentes en la escena. A estos contornos se les pueden extraer una serie de características que permiten su análisis estructural. Una de estas características es el área del contorno, que para el caso del presente trabajo fue utilizada con el objetivo de filtrar aquellos *blobs* que no contienen información sobre el objeto de interés. Una vez más, el valor de esta característica fue ajustado para cada clase, obteniéndose así los mejores resultados en el filtrado. Los valores utilizados son mostrados en la **Tabla 3.3**.

Tabla 3.3 Valores del área de los contornos conservados de acuerdo a la clase procesada.

Clase	Área del contorno
Bicicleta	5000
Bus	35000
Camión	35000
Campero	20000
Furgoneta	17000
Hatchback	14000
Moto	5000
Pickup	22000
Sedan	16000
SUV	20500



Figura 3.14 Demarcación en el fotograma de entrada de los contornos clasificados como vehículos por medio de una línea roja y un rectángulo verde.

Los contornos que permanecen después del filtrado por área son considerados como vehículos en la escena. Estos vehículos son demarcados en el fotograma de entrada con una

línea roja y delimitados con un rectángulo de color verde, como se observa en la **Figura 3.14**. De igual modo, se genera una imagen de la entrada segmentada (Ver **Figura 3.15**) y un conteo de los vehículos presentes en la escena, que será utilizado para posterior procesamiento.

Este procedimiento mostró buenos resultados en la mayoría de los casos como se puede observar en la **Figura 3.15(a)**. Sin embargo, en algunos casos el resultado final no es tan perfecto debido a fenómenos de oclusión en la escena (problema que no es considerado en esta solución) o un resultado no tan bueno del proceso de segmentación y formación de *blobs*, tal como se aprecia en la **Figura 3.15(b)**.

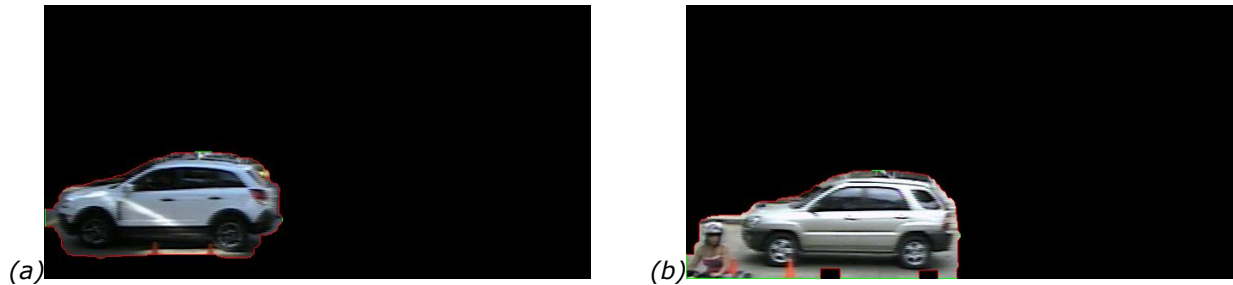


Figura 3.15 Resultados del proceso de filtrado de contornos: (a) Contorno bien filtrado. (b) Defecto en el filtrado del contorno debido a oclusión.

3.4.4 PROCESO DE SEGUIMIENTO MEDIANTE FLUJO ÓPTICO Y CÁLCULO DE LA VELOCIDAD ESTIMADA DEL VEHÍCULO

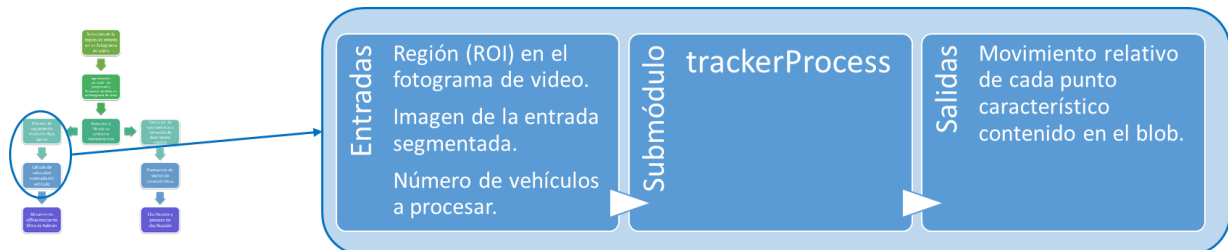


Figura 3.16 Diagrama de entradas y salidas para el submódulo *trackerProcess*.

El submódulo *trackerProcess* recibe como entradas una región (*ROI*) en el fotograma de video, una imagen de la entrada segmentada y el número de vehículos a procesar, y entrega como salidas el movimiento relativo de cada uno de los puntos característicos contenidos en el *blob* en cuestión y la velocidad total estimada del *blob* en el mundo real, en unidades de $\frac{m}{s}$ y $\frac{km}{h}$.

El algoritmo seguido en el submódulo *trackerProcess* es mostrado en **Algoritmo 3.1**, en pseudocódigo. El algoritmo de estimación de velocidad es mostrado en **Algoritmo 3.2** en pseudocódigo. La **Figura 3.17** muestra el resultado de este proceso, con las líneas blancas indicando la dirección y magnitud del movimiento de cada uno de los puntos hallados mediante *Good Features to Track* desde el fotograma inicial, hasta el actual.



Figura 3.17 Movimiento de cada punto hallado mediante Good Features to Track al interior del blob clasificado como vehículo, desde su posición inicial hasta su posición actual, indicado con las flechas blancas.

Algoritmo 3.1 Algoritmo de seguimiento de características mediante flujo óptico (trackerProcess)

1.	$i \leftarrow 0;$	// Iniciar el contador
2.	$points; points_{new};$	// Vectores que contienen puntos en la imagen
3.	$N_t \leftarrow numVehiculos;$	// Número de vehículos en la escena
4.	$img_{new} \leftarrow entrada segmentada;$	// Entrada del submódulo
5.	Mientras $i < N_t$ Hacer	// Recorre todos los vehículos de la escena
6.	Si vacío(img) Entonces	// Si la imagen anterior está vacía
7.	$img \leftarrow entrada segmentada;$	// Asigna la entrada a la imagen anterior
8.	Fin Si	
9.	Si tamaño($points$) ≤ 10 entonces	// Si no hay suficientes puntos que seguir
10.	$aux \leftarrow goodFeaturesToTrack(img_{new});$	// Detectar nuevos puntos en la entrada
11.	$points \leftarrow points + aux;$	// Agregar puntos detectados
12.	Fin Si	
13.	$aux \leftarrow opticalFlow(img, img_{new}, points);$	// Seguimientos de puntos entre el fotograma actual y el anterior
14.	$points_{new} \leftarrow aux;$	// Puntos seguidos en el fotograma actual
15.	Para cada $p_j \in points_{new}$ Hacer	// Recorre los puntos seguidos
16.	Si $points_{new}[j] - points[j] \leq 1$ Entonces	// Si el movimiento relativo es insuficiente
17.	descartar(p_j);	// Descartar el punto
18.	Fin Si	
19.	Fin Para cada	
20.	Estimacion de velocidad;	// Algoritmo de estimación de velocidad
21.	$img \leftarrow img_{new};$	// Fotograma anterior = fotograma actual
22.	$points \leftarrow points_{new};$	// Puntos anteriores = puntos actuales
23.	Fin Mientras	

Algoritmo 3.2 Algoritmo de estimación de la velocidad a partir de la información suministrada por el algoritmo de flujo óptico.

1.	$i \leftarrow 0;$	// Iniciar el contador
2.	$points; points_{new};$	// Vectores que contienen puntos en la imagen
3.	$mov \leftarrow 0; vel \leftarrow 0;$	// Velocidad y movimiento de un punto
4.	$vel_{prom} \leftarrow 0; mov_{prom} \leftarrow 0;$	// Velocidad y movimiento promedios
5.	Para cada $p_i \in points_{new}$ Hacer	// Recorre todos los puntos seguidos con flujo óptico
6.	flecha(inicial(p_i), p_i);	// Dibuja una flecha entre la posición inicial y actual del cada punto
7.	$mov \leftarrow movimiento(points[i], p_i);$	// Calcula el desplazamiento del punto entre dos fotogramas consecutivos
8.	$vel \leftarrow velocidad(mov, tiempo);$	// Calcula la velocidad conociendo el desplazamiento y el tiempo entre fotogramas
9.	Si $vel \leq 4 \frac{m}{s} = 14.4 \frac{km}{h}$ Entonces	// Si la velocidad del punto es menor a $4 \frac{m}{s}$
10.	$mov_{prom} \leftarrow mov;$	// Agrega el desplazamiento del punto al promedio
11.	$vel_{prom} \leftarrow vel;$	// Agrega la velocidad del punto al promedio
12.	Fin Si	
13.	Fin Para cada	

14.	$mov_{prom} \leftarrow \frac{mov_{prom}}{t};$	// Calcula el promedio de desplazamiento
15.	$vel_{prom} \leftarrow \frac{vel_{prom}}{t};$	// Calcula el promedio de velocidad
16.	$archivo_externo_{mov} \leftarrow mov_{prom};$	// Se escribe el desplazamiento promedio en un archivo externo
17.	$archivo_externo_{vel} \leftarrow vel_{prom};$	// Se escribe la velocidad promedio en un archivo externo

Como se puede observar, los datos de velocidad y distancia recorrida por el vehículo estimadas en el mundo real son exportados a dos archivos externos con nombres **data(N)Pos.dat** y **data(N)Vel.dat**

3.4.5 ALISAMIENTO OFFLINE MEDIANTE FILTRO DE KALMAN

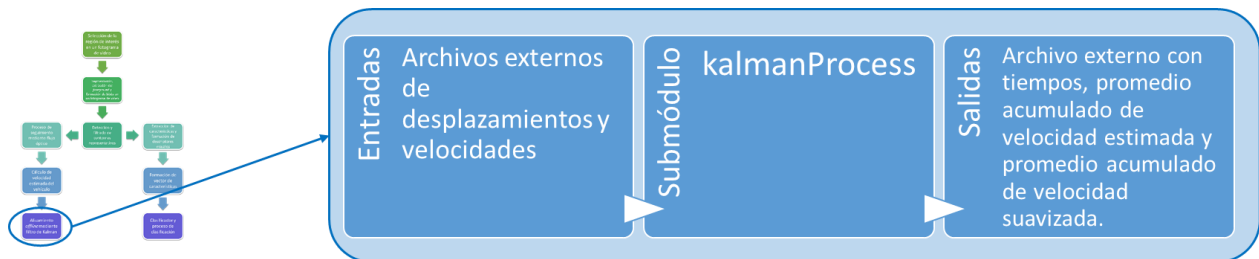


Figura 3.18 Diagrama de entradas y salidas para el submódulo *kalmanProcess*.

El submódulo *kalmanProcess* recibe como entradas los archivos externos de desplazamientos y velocidades entre fotogramas generados por el submódulo *trackerProcess* y entrega como salida un archivo externo **data(N)VelProm.dat** que contiene los tiempos, un promedio acumulado de la velocidad estimada y un promedio acumulado de la velocidad suavizada mediante un filtro de kalman suavizador de los fotogramas de la secuencia de video en los cuales el vehículo analizado se encuentra presente en la escena. El uso del suavizador es necesario debido a que los datos presentan grandes fluctuaciones, como se observa en la **Figura 3.19**, y es necesario tener una medición más estable.

Tabla 3.4 Modelo de estados para un sistema con movimiento uniforme rectilíneo con velocidad constante.

Modelo	Tiempo continuo	Tiempo discreto
Planta	$\dot{x}(t) = F(t)x(t) + w(t)$	$x_k = \Phi_{k-1}x_{k-1} + w_{k-1}$
Ruido de la planta	$E\langle w(t) \rangle = 0$ $E\langle w(t)w^T(s) \rangle = \delta(t-s)Q(t)$	$E\langle w_k \rangle = 0$ $E\langle w_k w_k^T \rangle = \Delta(k-i)Q_k$
Matriz de transición de estados	$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$	$F_k = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix}$
Matriz de covarianzas del ruido de la planta	Mediciones del sistema	$Q_k = q * \begin{bmatrix} \frac{\delta t^3}{3} & \frac{\delta t^2}{2} \\ \frac{\delta t^2}{2} & \delta t \end{bmatrix}$
Mediciones	$z(t) = H(t)x(t) + v(t)$	$z_k = H_k x_k + v_k$
Ruido de medición	$E\langle v(t) \rangle = 0$ $E\langle v(t)v^T(s) \rangle = \delta(t-s)R(t)$	$E\langle v_k \rangle = 0$ $E\langle v_k v_k^T \rangle = \Delta(k-i)R_k$
Matriz de acople de medición	$H = [0 \ 1]$	$H = [0 \ 1]$
Matriz de covarianzas del ruido de medición	Obtenido de las mediciones	Obtenido de las mediciones

Para el cálculo del suavizado de velocidad mediante el filtro de kalman se hizo uso de la técnica de *Suavizado de intervalo fijo en tres pasos* descrita en la Sección 2.2.2.7. Para esto, es necesario disponer de un modelo del sistema, que para el caso del presente trabajo corresponde a un movimiento uniforme rectilíneo con velocidad constante (Reid & Term, 2001) con mediciones de velocidad. Para realizar el proceso de suavizado con filtro de Kalman es preciso disponer de todos los datos de velocidad, de principio a fin de la toma. Es decir, este es un proceso que se realiza de manera *offline* una vez se haya estimado las

velocidades para todos los fotogramas en los que un vehículo se encuentra presente en la escena. Para un sistema de esta naturaleza se tiene un modelo de estados como muestra la **Tabla 3.4**.

Para realizar el primer paso del filtro de Kalman, hacia adelante en el tiempo, se ajustan unos valores iniciales como muestra la **Tabla 3.5**. Se realiza el primer recorrido del filtro, guardando los valores *a posteriori* del vector de estados estimado y de la matriz de covarianzas de la incertidumbre de estimación en sendas matrices, para ser utilizados posteriormente en el suavizado. El valor de δt se actualiza en cada iteración del cálculo del filtro, ya que los tiempos entre los fotogramas de video no son uniformes y son leídos de los archivos externos. La actualización se hace por medio de un promedio acumulado de los tiempos entre fotogramas. De igual modo, con la actualización del δt , se deben actualizar también los valores de las matrices F y Q .

Para realizar el segundo paso del filtro de kalman, hacia atrás en el tiempo, se ajustan unos valores iniciales como muestra la **Tabla 3.5**. Se realiza el segundo recorrido del filtro, guardando los valores *a priori* del vector de estados estimado y de la matriz de covarianzas de la incertidumbre de estimación en sendas matrices, para ser utilizados posteriormente en el suavizado. El valor de δt se actualiza en cada iteración del cálculo del filtro, ya que los tiempos entre los fotogramas de video no son uniformes y son leídos de los archivos externos. La actualización se hace por medio de un promedio acumulado de los tiempos entre fotogramas. De igual modo, con la actualización del δt , se deben actualizar también los valores de las matrices F y Q .

Tabla 3.5 Valores iniciales para el segundo recorrido del filtro de Kalman en el proceso de suavizado, hacia atrás en el tiempo

Variable	Primer recorrido	Segundo recorrido
δt	0.033	0.033
x_{est}	$[pos_{inicial}, vel_{inicial}]$	$[pos_{inicial}, vel_{inicial}]$
$pos_{inicial}$	0	pos_{end}
$vel_{inicial}$	$prom(vel_1, vel_2, vel_3, vel_4, vel_5)$	vel_{end}
F_k	$\begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix}$
R_k	$var(vel)$	$var(vel)$
q	1	1
Q_k	$q * \begin{bmatrix} \frac{\delta t^3}{3} & \frac{\delta t^2}{2} \\ \frac{\delta t^2}{2} & \delta t \end{bmatrix}$	$q * \begin{bmatrix} \frac{\delta t^3}{3} & \frac{\delta t^2}{2} \\ \frac{\delta t^2}{2} & \delta t \end{bmatrix}$
P_k	Q_k	Q_k

Posteriormente, se hace un tercer recorrido, esta vez a través de todos los datos guardados durante los dos primeros recorridos, integrando de manera óptima (mediante filtro de Kalman) estos valores utilizando la *Ecuación 2.13*.

Por último se hace un recorrido a través de todos los datos de velocidad en bruto y los datos de velocidad filtrada mediante el suavizador de kalman, realizando un promedio acumulado de los mismos y consignando, en un archivo externo ***data(N)VelProm.dat***, en tres columnas, los tiempos, el promedio de velocidad en bruto y el promedio de velocidad filtrada con kalman.

Adicionalmente, el submódulo *kalmanProcess* realiza, usando *gnuplot*, 4 gráficos que permiten visualizar el proceso de filtrado hacia adelante comparado con los datos medidos, filtrado hacia atrás comparado con los datos medidos, suavizado comparado con los datos medidos y los promedios acumulados comparados con los datos de velocidad suavizados. Estos gráficos, calculados para el *dato75* de clase SUV son mostrados en la **Figura 3.19**.

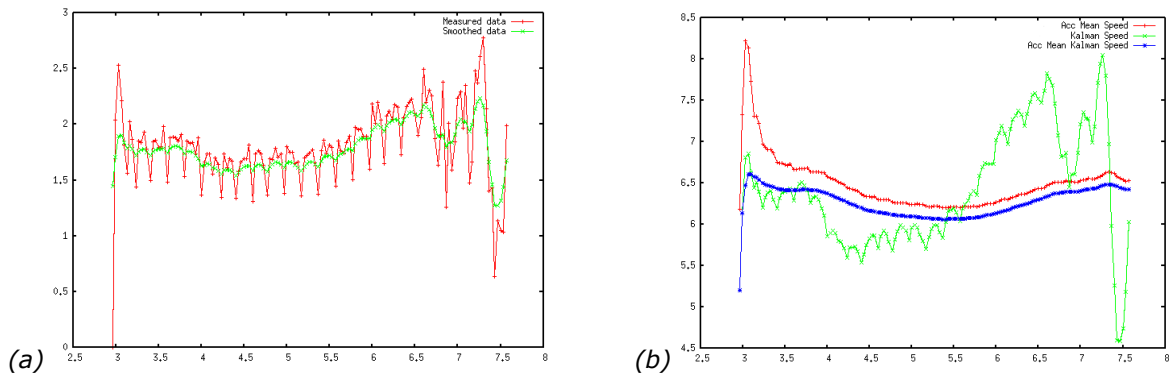


Figura 3.19 (a) Resultados de medición de velocidad –rojo- y suavizado óptimo mediante Kalman –verde-, en $\frac{m}{s}$. (b) Resultados de suavizado óptimo mediante Kalman –verde-, promedio acumulado de medición de velocidad simple –rojo- y promedio acumulado de velocidad suavizada con Kalman –azul-, en $\frac{km}{h}$

3.4.6 OBTENCIÓN DE PUNTOS CLAVE EN LA REGIÓN DE INTERÉS Y CÁLCULO DE DESCRIPTORES Y CARACTERÍSTICAS VISUALES

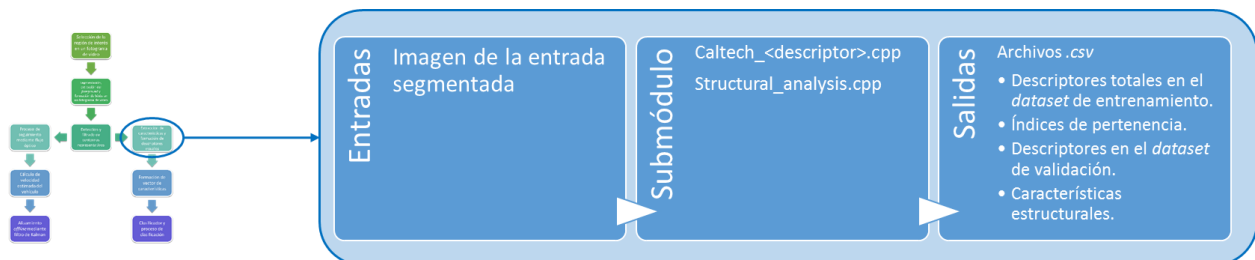


Figura 3.20 Diagrama de entradas y salidas del submódulo de obtención de puntos clave y cálculo de descriptores y características visuales.

El submódulo de obtención de puntos clave y formación de descriptores visuales se encuentra escrito en un archivo fuente aparte con nombre *Caltech_<descriptor>.cpp* y, por consiguiente, se ejecuta de manera independiente a los anteriores módulos. Este submódulo toma como entrada la imagen de entrada segmentada, obtenida en el submódulo *contoursProcess* y entrega como salidas una serie de archivos de valores separados por comas (.csv).

Para la obtención de puntos clave y descriptores visuales se hizo uso de las técnicas descritas en la sección 2.2.1.1. Hay métodos que sólo son aplicables al cálculo de descriptores, más no a la detecciones de puntos clave (BRIEF, FREAK). Para estos métodos se hizo uso de los puntos clave *FAST*.

La **Figura 3.21** muestra una misma imagen de entrada para el *dato75*, correspondiente a la clase *SUV*, y los puntos clave obtenidos. La representación de su información de escala y orientación se encuentra codificada en los círculos blancos concéntricos a cada punto clave. El radio del círculo corresponde a la escala en la cual el punto fue hallado y la línea que une el centro del círculo con su perímetro está dibujada en la dirección de la orientación del punto clave. Las salidas de este módulo se diseñaron en miras de su uso para el *toolbox Caltech Large Scale Image Search Toolbox* (Aly, Munich, & Perona, 2011) de Matlab. De esta manera, en miras de satisfacer las entradas requeridas, las salidas fueron 3 tipos de archivos .csv diferentes. Estos tres archivos fueron generados para todos y cada uno de los

conjuntos de descriptores visuales considerados, esto con el fin de llevar a cabo un proceso de comparación, que será detallado más adelante.

Para hacer uso del *toolbox* de *Caltech*, se requieren de unos datos de entrenamiento y unos datos de validación. Para generar estos dos conjuntos se decidió hacer uso de los datos contenidos en el *dataset* y separarlos mediante un módulo 2, es decir, empezando por el primero, se hace uso de todos los datos con índice impar (*i.e.* 1º, 3º, 5º, etc.). De igual modo se procedió para los datos de validación, pero esta vez empezando por el segundo dato, obteniéndose todos los datos con índice par (*i.e.* 2º, 4º, 6º, etc.). Este procedimiento de separación se realizó de manera manual, generando dos directorios llamados "*imagenes training*" e "*imagenes testing*". La cantidad de datos, para entrenamiento y validación, obtenidos de esta manera se muestran en la **Tabla 3.6**.

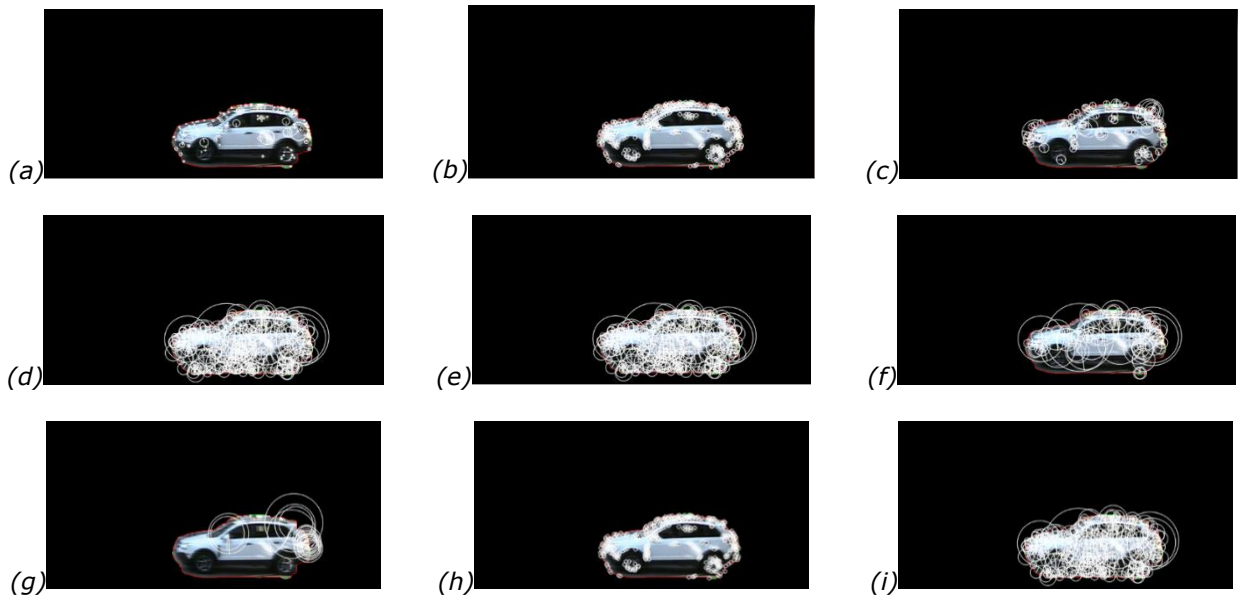


Figura 3.21 Imagen de entrada segmentada y los respectivos puntos clave obtenidos mediante distintas técnicas: (a) Puntos clave SIFT para calcular descriptores SIFT. (b) Puntos clave FAST con umbral de 20 para calcular descriptores BRIEF. (c) Puntos clave BRISK para calcular descriptores BRISK. (d) Puntos clave SURF con umbral de 0 para calcular descriptores SURF. (e) Puntos clave SURF con umbral de 20 para calcular descriptores SURF. (f) Puntos clave SURF con umbral de 2500 para calcular descriptores SURF. (g) Puntos clave ORB con umbral de 20 para calcular descriptores ORB. (h) Puntos clave FAST con umbral de 20 para calcular descriptores FREAK. (i) Puntos clave SURF con umbral de 20 para calcular descriptores FREAK.

Tabla 3.6 Cantidad de datos totales del dataset destinados a entrenamiento y validación.

Clase	No. de datos de entrenamiento	No. de datos de validación	TOTAL
Bicicleta	13	13	26
Bus	3	2	5
Camión	14	14	28
Campero	13	13	26
Furgoneta	7	6	13
Hatchback	13	12	25
Moto	13	12	25
Pickup	13	13	26
Sedan	13	12	25
SUV	13	12	25
TOTAL	115	109	224

En primer lugar, se genera un archivo denominado *descriptors.csv*, en el que se consignan todos los descriptores de todos los puntos clave de las imágenes contenidas en el *dataset* de entrenamiento. Este archivo tiene la forma de una matriz, en la que las columnas están separadas por comas y las filas están separadas por saltos de línea. Esta matriz tiene tantas columnas como componentes tenga el descriptor obtenido (Véase la sección 2.2.1.1), y tantas filas como el número de datos totales en el *dataset* de entrenamiento multiplicado por el número de puntos clave obtenidos en cada dato. Esto es mostrado en la **Figura 3.22**

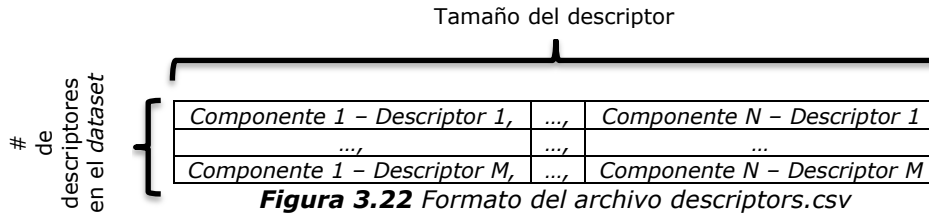


Figura 3.22 Formato del archivo *descriptors.csv*

En segundo lugar, se genera un archivo denominado *labels.csv*, que se compone de una sola fila, y tantas columnas como el número de datos totales en el *dataset* de entrenamiento multiplicado por el número de puntos clave obtenidos en cada dato, separadas por comas. Este archivo contiene, por cada columna, el índice de la imagen del *dataset de entrenamiento* al cual pertenece el descriptor correspondiente en el archivo *descriptors.csv*. En otras palabras, lo que se logra con este vector es, básicamente, asociar cada uno de los descriptores del *dataset* de entrenamiento con la imagen a la que pertenece. Este formato se puede observar en la **Figura 3.23**

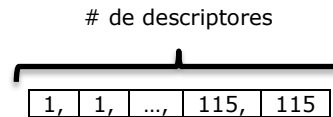


Figura 3.23 Formato del archivo *labels.csv*

Por último, se generan 109 archivos con nombres *n.csv*, donde $n = 1, 2, \dots, 109$. Cada uno de estos archivos tiene la forma de una matriz en la que las columnas están separadas por comas y las filas están separadas por saltos de línea. Cada una de estas matrices tiene tantas columnas como componentes tenga el descriptor utilizado y tantas filas como número de puntos clave obtenidos en el dato en cuestión. Los datos utilizados para este proceso son aquellos que pertenecen al *dataset* de validación. Esto se puede observar en la **Figura 3.24**

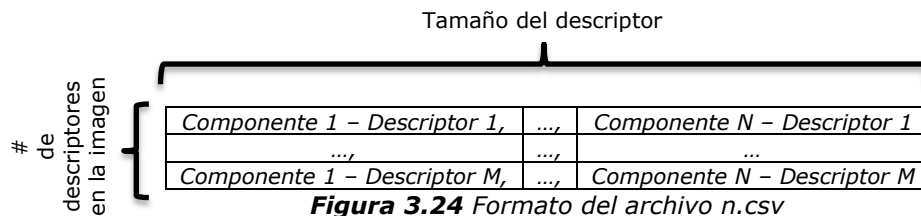


Figura 3.24 Formato del archivo *n.csv*

Los archivos *dsecriptors.csv* y *labels.csv* son guardados en un directorio llamado *training* ubicado en una carpeta designada para cada tipo de descriptor visual específico. Para efectos de claridad, la **Figura 3.25(a)** muestra una instantánea del directorio general utilizado para almacenar los archivos resultantes de este módulo. De igual modo, se observa en **Figura 3.25(b)** el interior de una de estas carpetas, en este caso, la carpeta que contiene los archivos para los descriptores tipo BRIEF, en donde se explica que el número que acompaña a la palabra *caltech* designa el umbral utilizado por el detector de puntos clave para desempeñar su tarea. A su vez, la **Figura 3.25(c)** muestra el interior de una de

estas carpetas, en este caso la carpeta que contiene los archivos para descriptores tipo BRIEF calculados de puntos clave tipo FAST con umbral de 20.

Los archivos *n.csv* son guardados en un directorio llamado *testing* ubicado en una carpeta designada para cada tipo de descriptor visual específico. También es posible visualizar este directorio en la **Figura 3.25(c)**.

Además de este submódulo de obtención de puntos clave y descriptores visuales, también fue escrito un submódulo de obtención de características estructurales que permitieran una mejor distinción entre las clases consideradas. De igual modo que el anterior, la obtención de características estructurales se encuentra en un archivo fuente adicional, llamado *Structural_analysis.cpp*, que debe ser ejecutado independientemente del módulo de visión.

Este submódulo recibe como entrada las imágenes de la entrada segmentada y entrega como salidas dos archivos conteniendo 5 características estructurales del *blob* procesado. Estos archivos corresponden cada uno al *dataset* de entrenamiento y de validación y son llamados "*stats.csv*". Estos archivos son guardados en sendas carpetas con nombres *training* y *testing*, de acuerdo a los datos utilizados para su formación. Las carpetas *training* y *testing* son, a su vez, guardadas en una carpeta superior denominada *stats*, que puede visualizarse también en la **Figura 3.25(c)**.

Las características estructurales obtenidas se detallan a continuación, en la *Ecuación 3.4*, *Ecuación 3.5*, *Ecuación 3.6*, *Ecuación 3.7* y *Ecuación 3.8*. (Huang, Chen, & Hu, 2012)

$$Compacticidad = \frac{\text{perímetro}^2}{\text{área}} \quad \text{Ecuación 3.4}$$

$$\text{relación de aspecto} = \frac{\text{altura}}{\text{ancho}} \quad \text{Ecuación 3.5}$$

$$\text{relación de área} = \frac{\text{área del contorno}}{\text{área del bounding box}} \quad \text{Ecuación 3.6}$$

$$\text{Distancia horizontal del centroide al centro} \quad \text{Ecuación 3.7}$$

$$d_{cc} = | \text{centroide.x} - \text{centro.x} |$$

$$\text{Relación distancia centroide – centro con largo} \quad \text{Ecuación 3.8}$$

$$\text{norm}_{d_{cc}} = \frac{| \text{centroide.x} - \text{centro.x} |}{\text{ancho}}$$

Los archivos *stats.csv* tienen forma de matriz con tamaños:

- 5 filas y 115 columnas para los datos de entrenamiento, correspondientes a 5 características estructurales y 115 datos totales en el *dataset*.
- 5 filas y 109 columnas para los datos de validación, correspondientes a 5 características estructurales y 109 datos totales en el *dataset*.

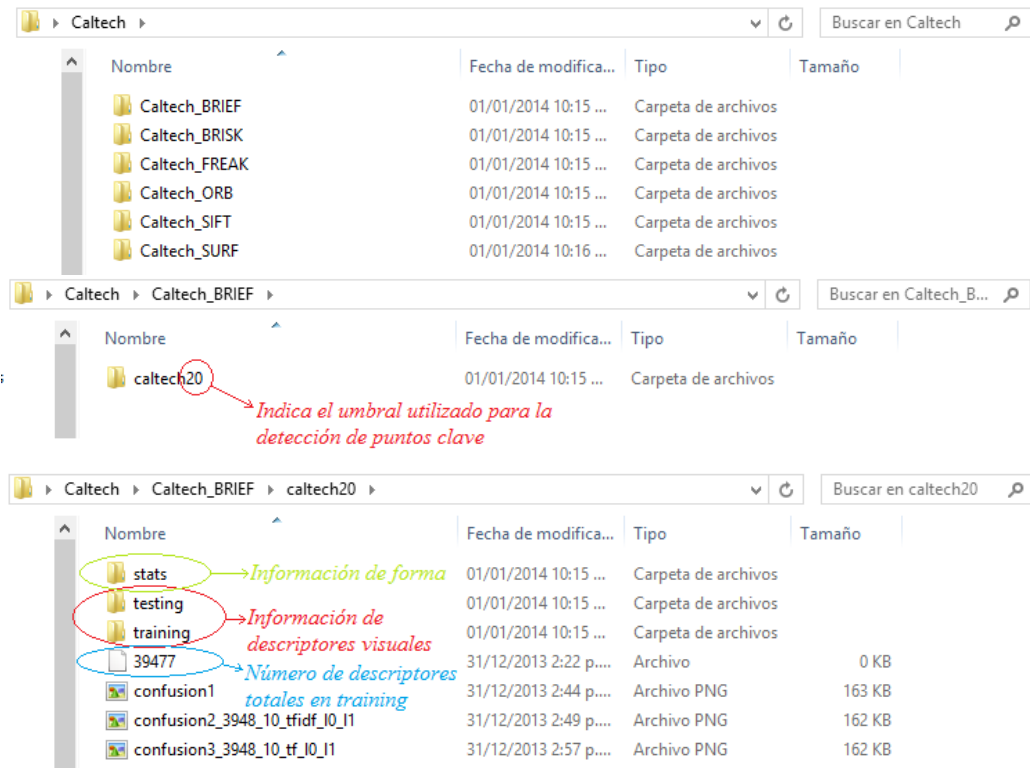


Figura 3.25 (a) Carpeta general que contiene los archivos obtenidos para cada uno de los 6 descriptores visuales considerados en el trabajo. (b) Interior de la carpeta Caltech_BRIEF. (c) Interior de la carpeta Caltech_BRIEF\caltech20.

3.4.7 FORMACIÓN DE VECTOR DE CARACTERÍSTICAS Y CLASIFICACIÓN

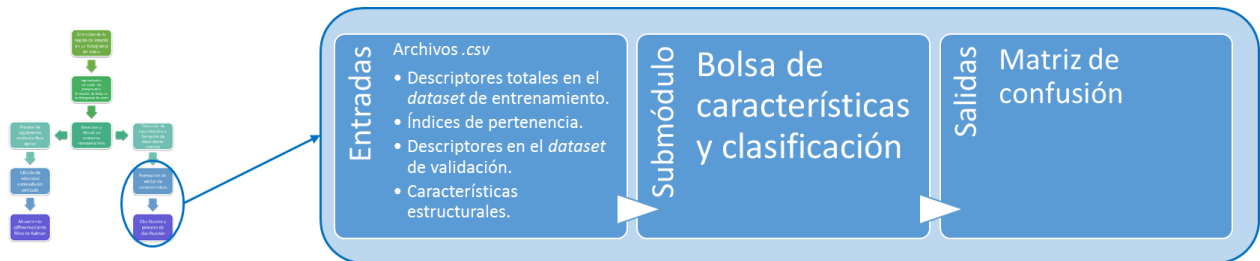


Figura 3.26 Diagrama de entradas y salidas del submódulo de formación de vectores de características y clasificación.

El módulo de clasificación con información de intensidad hace uso de los descriptores visuales obtenidos y la técnica *Bag of Features* para formar los vectores de características. Esta técnica se escogió por su apropiada utilización con características de tipo descriptores visuales. Como se mencionó, para la formación del vector de características y, de hecho, para el proceso de clasificación final (Detallado en el apartado siguiente), se hizo uso del toolbox de *Caltech* para Matlab. El esquema mostrado en la **Figura 3.27** detalla los pasos seguidos en el *toolbox* con el objetivo de lograr la clasificación final de los datos.

Este proceso recibe como entradas los archivos de valores separados por comas correspondientes a los descriptores totales en el *dataset* de entrenamiento, los índices a los cuales pertenece cada descriptor del *dataset* de entrenamiento, los descriptores correspondientes a cada uno de los datos en el *dataset* de validación y las características

estructurales de los datos contenidos en los *datasets* de entrenamiento y validación, y entrega como salida una matriz de confusión que permite evaluar el desempeño del sistema con el conjunto de descriptores usado.

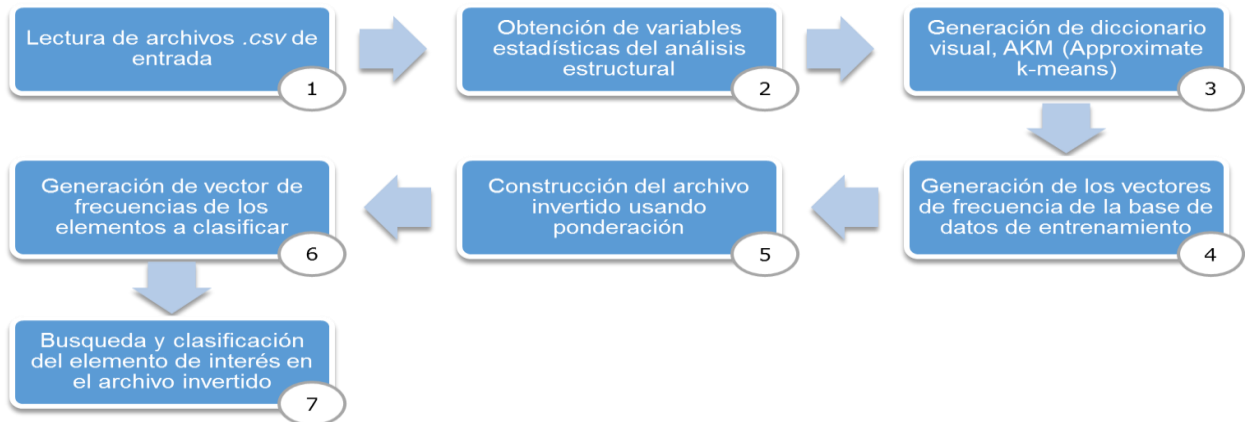


Figura 3.27 Esquema de trabajo para la clasificación con Bolsa de Características usando el toolbox de Caltech en Matlab.

Una vez leídos los archivos, haciendo uso del comando `csvread` de Matlab, es sólo cuestión de almacenar el resultado en una variable para disponer de la información necesaria en el formato adecuado (matriz de vectores columna, conteniendo los descriptores totales contenidos en el *dataset* de entrenamiento y los índices de las instancias a las cuales pertenece cada uno de ellos). Esto constituye el primer paso del flujo mostrado en la **Figura 3.27**

Tabla 3.7 Mediciones estadísticas de las variables estructurales obtenidas de los blobs contenidos en las imágenes de entrenamiento.

	Compacticidad		Relación de aspecto		Relación de área		Distancia centroide-centro		Distancia centroide-centro normalizada	
	Media	Des. Est.	Media	Des. Est.	Media	Des. Est.	Media	Des. Est.	Media	Des. Est.
Bicicleta	19.6544	1.9539	0.9830	0.0804	0.6493	0.0466	2.4962	3.1238	0.0224	0.0242
Bus	43.6681	15.6708	0.5017	0.2103	0.6468	0.0454	35.0425	31.3671	0.0790	0.0668
Camión	25.5127	10.5307	0.5470	0.0838	0.7679	0.1002	9.6387	7.2000	0.0278	0.0221
Campero	20.3134	2.0194	0.5013	0.0608	0.7504	0.0693	9.2532	9.0969	0.0338	0.0301
Furgoneta	20.0368	3.0784	0.5183	0.0621	0.8329	0.0498	6.7349	3.2881	0.0252	0.0123
Hatchback	19.2997	1.2679	0.4737	0.0426	0.7946	0.0347	4.3557	1.8424	0.0190	0.0078
Moto	22.0767	5.3101	0.8145	0.1343	0.6451	0.0500	3.9025	3.1716	0.0278	0.0197
Pickup	22.6406	2.3739	0.4163	0.0630	0.7620	0.0369	4.7612	4.4467	0.0148	0.0130
Sedan	22.0487	4.0920	0.4203	0.0288	0.7688	0.0534	3.3119	1.8948	0.0129	0.0073
SUV	20.1645	1.5799	0.4400	0.0289	0.8040	0.0323	5.5352	3.1456	0.0202	0.0109

En segundo lugar, se leen también los archivos que poseen las mediciones estructurales de las imágenes de entrenamiento, es decir, el archivo *stats.csv* contenido en la carpeta *training*. Una vez hecho esto, y conociendo la forma del *dataset* de entrenamiento, se genera una tabla que contiene mediciones estadísticas de estos datos. Estas mediciones son útiles para obtener un modelo estadístico del comportamiento de la información estructural, de modo que pueda ser usada con todas las instancias de una clase específica, e.g. el valor de compacticidad de una instancia de una bicicleta debe encontrarse al interior del intervalo 19.6544 ± 1.9539 . La tabla generada es igual para todos los descriptores, ya que las mediciones estructurales son iguales, independientemente del descriptor usado, y sus valores se muestran en la **Tabla 3.7**.

Posteriormente, el programa genera un diccionario de palabras visuales, o características, presentes en el *dataset* de entrenamiento teniendo en cuenta el número de palabras especificado por el usuario, parámetro que fue ajustado heurísticamente empezando por un valor de 10% del número de descriptores totales hallados en el *dataset* de entrenamiento (Knopp et al., 2010). El método utilizado para generar este diccionario es *Approximate k-means*. Se finaliza así el tercer paso del proceso mostrado en la **Figura 3.27**

Luego, como cuarto paso, para cada imagen en el *dataset* de entrenamiento se genera un vector de palabras, que contiene tantas componentes como puntos clave se hayan encontrado en una imagen específica, y que detalla el *id* de palabra (clúster en el diccionario) para cada descriptor visual presente. Estos constituyen los vectores de frecuencias que describen cada imagen, es decir, su vector de características usado para la clasificación.

Una vez se dispone de un vector de frecuencias para una imagen, este puede ser usado por cualquier tipo de clasificador para realizar la tarea de clasificación. Con el fin de hacer esta tarea de una manera eficiente, en (Sivic & Zisserman, 2003) se propone la elaboración de un *inverted-file*. Este archivo es análogo al índice de un libro. Establece los elementos (clases de vehículos) en los cuales aparece alguna instancia de cada palabra visual. Se procede, entonces, a construir el archivo invertido usando la ponderación y normalización elegida por el usuario de las opciones detalladas en la **Tabla 3.8** Cumpliendo así con la quinta componente del proceso mostrado en la **Figura 3.27**

Tabla 3.8 Modos de ponderación y normalización para la construcción del archivo invertido en el toolbox de Caltech en Matlab.

Ponderación	'none'	No se ponderan los valores del conteo.
	'bin'	Se binariza el conteo.
	'tf'	Se divide cada conteo por el número de palabras en el documento.
	'tfidf'	Usa la ponderación <i>tf-idf</i> (Sivic & Zisserman, 2003)
Normalización	'none'	Sin normalización.
	'l0'	Normalizar por el número de palabras.
	'l1'	Normalizar de modo que la suma del histograma = 1
	'l2'	Normalizar de modo que la suma de los cuadrados del histograma = 1

Posterior a esto, se generó un *loop* que itera a través de todos los datos de validación, generando sus vectores de frecuencias y buscando, por medio del uso del archivo invertido, los *k* vecinos más cercanos del *dataset* de entrenamiento, donde *k* es un parámetro elegido por el usuario, que para el caso del presente trabajo es 5. De estos *k*-vecinos más cercanos, se elige el primero como clase a la que pertenece la instancia procesada. Esto abarca los pasos seis y siete del diagrama de flujo mostrado en la **Figura 3.27**

Ya teniendo una clase asociada al elemento a clasificar, se procede a modificar la matriz de confusión y, una vez se haya iterado por todos los elementos del *dataset* de validación, el programa muestra la matriz de confusión para hacer la validación del sistema. Los resultados obtenidos de todo este proceso son expuestos en el *Capítulo 4*.

3.5 DISEÑO Y DESARROLLO DEL MÓDULO DE CLASIFICACIÓN BASADO EN INFORMACIÓN DE RANGO

El módulo de clasificación basado en descriptores 3D fue desarrollado con un esquema de pequeños procesos que permite la sustitución o modificación de los algoritmos en cada etapa, sin afectar a las demás. El esquema mostrado en la **Figura 3.28** presenta los diferentes procesos que se siguen con el objetivo final de obtener una clasificación de una instancia de una clase (vehículo) en nubes de puntos (clústeres) extraídas de las capturas totales; en esta sección se realiza un análisis de cada uno de los componentes y se describe

el proceso de implementación de cada etapa. Básicamente se puede resumir este proceso en 4 macro etapas: Construcción de la nube, segmentación, descripción y clasificación. Ciertas etapas como la extracción de puntos claves y descriptores fueron implementadas con diferentes técnicas, con la finalidad de realizar un proceso de comparación y evaluación del desempeño del sistema en cada caso. Es importante informar en este punto que internamente la etapa de estimación de descriptores 3D está dividida en una sección para descriptores locales y otra para descriptores globales. Finalmente es importante resaltar que las etapas de la 4 a la 9 del esquema de la **Figura 3.28**, sirven como pre procesamiento, para el núcleo de extracción de descriptores 3D.



Figura 3.28 Diagrama de bloques del módulo de clasificación basado en información de rango.

3.5.1 ESTIMACIÓN DE LA VELOCIDAD DEL VEHÍCULO Y FILTRADO DE KALMAN

El primer paso que es necesario para la elaboración consistente de una nube de puntos en 3 dimensiones (en el contexto de este trabajo), corresponde a la estimación de la velocidad con la cual los vehículos atraviesan el haz de escaneo, ya que contando con esta información y el tiempo entre cada escaneo puede determinarse la distancia real de separación entre cada uno. Haciendo uso de las virtudes de un sistema multimodal, y pretendiendo el uso de un esquema sencillo y sin requerimientos de sensores adicionales, se decidió la realización de un proceso de estimación de velocidad basado en la información visual entregada por la cámara, a través de técnicas de análisis de flujo óptico. Todos los detalles de la implementación de este sub módulo son presentados en la sección 3.4.4 y la descripción de estado del arte en esta temática en la sección 2.2.2. Los resultados obtenidos presentan una muy buena aproximación de la velocidad del vehículo en cada fotograma de la información de imagen. Sin embargo, es necesario un proceso de filtrado-suavizado debido a la presencia de diversas fuentes de ruido en el sistema como lo son: la confusión o pérdida en el seguimiento de características fotograma a fotograma, el registro de velocidades ajenas al *blob* de interés, fusión de diferentes elementos del *foreground* segmentado. La implementación del filtro suavizador de kalman se presenta en la sección 3.4.5.

3.5.2 RECTIFICACIÓN DE LA INFORMACIÓN DE RANGO A TRAVÉS DEL MODELO CALIBRADO Y PROCESO DE REGISTRO DE LA ESCENA 3D

Una vez, es obtenido el modelo apropiado de calibración del sensor, el valor de rango de cada punto $P(\theta, \rho)$, es transformado según las ecuaciones del modelo lineal, determinadas en la sección 3.3; el siguiente paso consiste en la unión de los escaneos 2D obtenidos en la captura del dataset, para la creación de los perfiles en 3D dimensiones de los vehículos. Existen dos formas de registrar o concatenar los planos consecutivos obtenidos en la captura de la información laser 3D:

- El primer método consiste en asumir el valor final del promedio acumulado, del filtro suavizador de kalman de velocidades, y utilizar este único valor para separar cada par de planos. Tiene como ventaja su inmunidad al ruido, su facilidad de implementación, y una mejor distribución de la densidad de puntos, lo que evita resultados inesperados o indeseables en la extracción de los descriptores 3D. Tiene problemas cuando el vehículo tiene cambios abruptos reales en la velocidad, mientras atraviesa el haz de escaneo.
- El segundo método, que por definición es más realista (esto es, que describe mejor la estructura de la nube), realiza un filtrado ponderado adicional para definir la separación entre pares de escaneos, considerando la distancia temporal entre los dos escaneos de láser y el fotograma de cámara más cercano para cada uno. En este enfoque el desplazamiento es actualizado en el tiempo mismo del registrado, lo que permite la detección de cada cambio en velocidad. Este enfoque exige mayor robustez en el sistema ante los cambios en densidades intra vehículo. Fue diseñado teniendo en cuenta que la cantidad máxima de escaneos entre dos fotogramas de cámara es 2 o 3 para tasas de 30 y 25 imágenes por segundo (13ms laser, 33ms y 40ms cámara).

La **Figura 3.29(a)** presenta un ejemplo de la distribución temporal de los escaneos y los fotogramas de la cámara, y la **Figura 3.29(b)** identifica las distancias usadas para la ponderación de la velocidad asignada entre cada par de escaneos de láser.

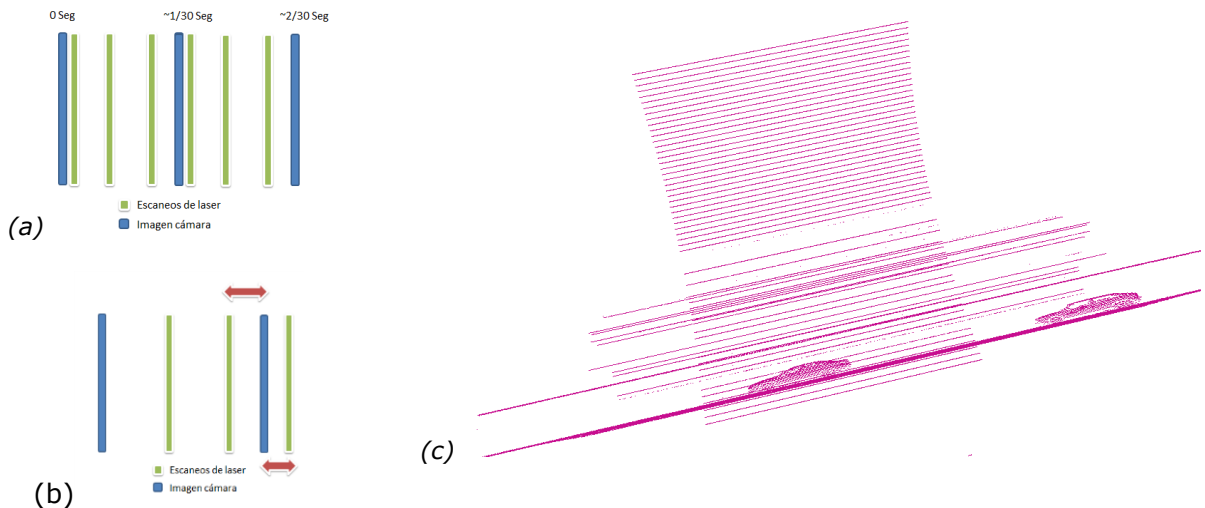


Figura 3.29 Esquema de asignación de ponderaciones, para las velocidades entregadas en cada fotograma (a) distribución temporal de los escaneos y los fotogramas, (b) distancias utilizadas en la ponderación; c) Nube de puntos original capturada por el sistema y corregida mediante el modelo lineal del sensor laser.

En la **Figura 3.29(c)** es presentado el resultado de una nube de puntos completa registrada, al fondo se percibe el arco generado por la distancia máxima de lectura del

sensor, aproximadamente 80 metros, se perciben un par de vehículos, las cuerdas de alta tensión, y otras líneas generadas por el perfil estático de un árbol enfrente del láser, pero más allá de donde pasan los vehículos.

3.5.3 CORRECCIÓN DE ÁNGULO DE INCLINACIÓN

En la etapa anterior se registró una nube 3D completa, a partir de escaneos 2D originalmente independientes; esta nube de puntos, incluye en su información tanto elementos relevantes, (vehículos) como no deseados (árboles, cuerdas de energía, y un plano dominante de carretera y un arco de rango máximo). Los primeros elementos no deseados, serán eliminados gracias a un proceso de clusterización y umbralización (sección 3.5.4.3) , pero el plano principal y la información fuera del rango deseado, deben ser eliminados a través de un mecanismo de filtrado de distancias en la nube de puntos o mediante consenso de muestras (para el plano, sección 3.5.4.2); para que el mecanismo de filtrado de distancias funcione correctamente debe el ángulo de inclinación presentado en los escaneos, ocasionado debido a que el haz central del sensor laser *Sick LMS200*, no es situado perfectamente paralelo a la superficie de la carretera. En la práctica, es muy difícil lograr esta condición, por lo que se necesita una etapa adicional que elimine la inclinación. En las **Figura 3.30** (a) y (b), se observa una nube de puntos con la inclinación debida a la posición incorrecta del sensor. El procedimiento para corregir este problema, se describe a continuación:

La corrección de la inclinación es realizada, sumándole al eje vertical de la nube de puntos, los valores obtenidos de una función de recta con variable independiente en el eje horizontal de la misma, la cual es determinada por dos puntos en un mismo haz de escaneo, pertenecientes a la carretera, lo que permite parametrizar y solucionar completamente, la orientación errónea de la nube de puntos, con respecto a su orientación en el mundo real. Con la finalidad de eliminar o atenuar la incidencia del ruido, los valores de los puntos clave seleccionados en la carretera, son promediados, a lo largo de los escaneos, tomados desde el inicio, hasta el cumplimiento del tiempo muerto (tiempo que pasa entre el inicio de toma de datos, y el paso del primer vehículo u objeto que obstruya el haz de laser), el procedimiento completo es sintetizado en la **Figura 3.31**, y detallado inmediatamente después.

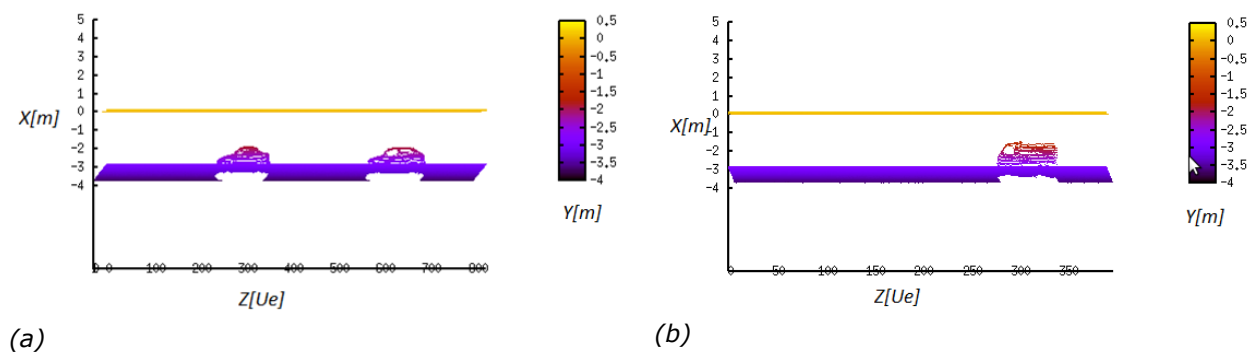


Figura 3.30 (a) Nube de puntos de un sedán y un hatchback con orientación inclinada, por posición del sensor laser incorrecta. (b) Nube de puntos inclinada de un camión.

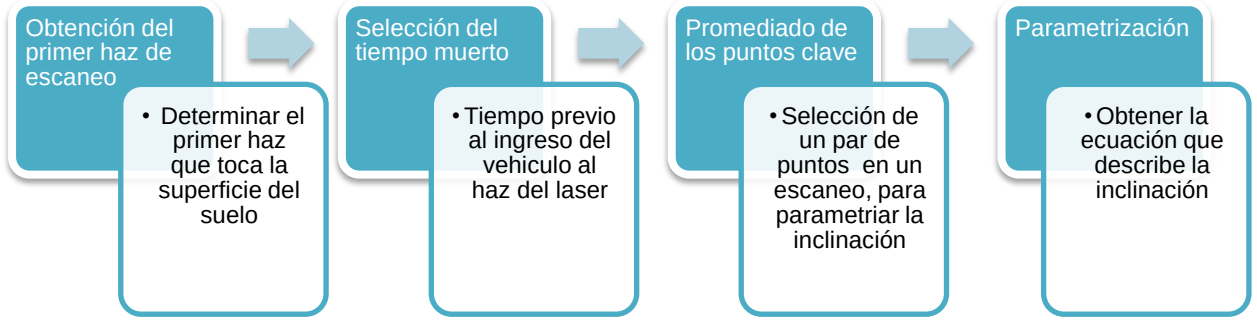


Figura 3.31 Esquema del proceso de corrección del ángulo de inclinación en la nube de puntos.

Se deben determinar puntos claves en la nube de puntos, que se encuentren en la carretera, que es la parte que permite parametrizar la inclinación, en este trabajo dado las condiciones de toma del dataset, esto se logra tomando los valores de la coordenada Y de cualquiera de los primeros 20 °grados de escaneo, en teoría se podría realizar tomando cualquiera de los primeros 180 °, pero en algunos de estos rayos impactarían la carretera, en regiones fuera del alcance máximo del láser por lo que sus medidas no son válidas. Para lograr este objetivo, en primer lugar se determina si el primer o el último haz de escaneo es el que primero impacta la superficie de la carretera, esto cambia dependiendo del lado del sensor que quede recostado sobre la base en la que reposa (en la toma de datos, esta orientación no fue siempre igual), esto se determina evaluando la condición presente en la *Ecuación 3.9*.

$$Ef(x) = \begin{cases} \text{impacto más cercano en primer rayo,} & \text{Escaneo[numrayo} - x] > \text{Escaneo}[x] \\ \text{impacto más cercano en último rayo,} & \text{Escaneo[numrayo} - x] < \text{Escaneo}[x] \end{cases}$$

Ecuación 3.9

Donde x puede variar entre 0 y 20, dependiendo del resultado de esta relación puede determinarse si el primer o último rayo impacta en la superficie de la carretera, y cual se dirige hacia arriba y devuelve el valor fuera de rango. Con la intención de que los puntos seleccionados se encuentren en la carretera y no en un vehículo (que pase dentro del rango de los primeros 20 °grados), se establece que al momento de la toma de datos de rango, se iniciara la adquisición 1 segundo antes de que el vehículo atravesara el haz de escaneo, este tiempo es denominado el tiempo muerto, y se garantizó en cada toma realizada en el dataset.

Una vez son adquiridos los puntos que definirán el modelo paramétrico, se realiza un promedio de sus valores, a lo largo del set de escaneos dentro del tiempo muerto, esto se traduce en un promedio de 77 valores, (cada escaneo se obtiene cada 13.33ms, y el tiempo muerto es de 1 segundo) esto se define en la *Ecuación 3.10* y la *Ecuación 3.11*.

$$puntoclave1 = \frac{\sum_{i=1}^{77} \text{Escaneo}(i)[\text{numrayo1}]}{77}$$

Ecuación 3.10

$$puntoclave2 = \frac{\sum_{i=1}^{77} \text{Escaneo}(i)[\text{numrayo2}]}{77}$$

Ecuación 3.11

Finalmente es obtenido el modelo de la recta con la inclinación definida por la pendiente, y esta función es adicionada a la nube de puntos, obteniéndose con una inclinación nula como se observa en la **Figura 3.32**

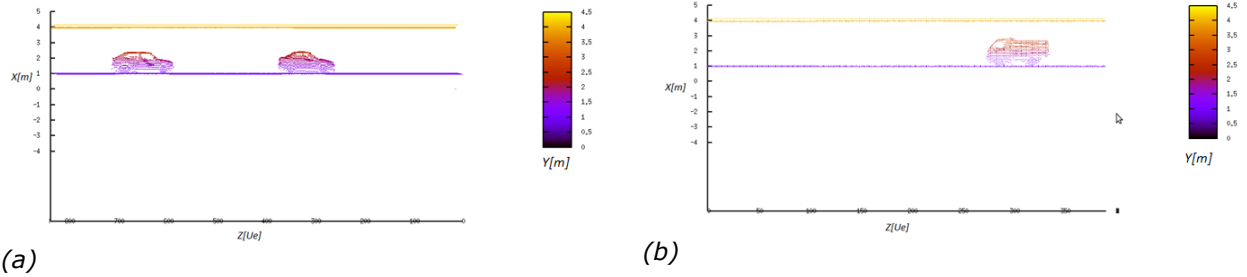


Figura 3.32 (a) Nube de puntos de un sedán y un hatchback con orientación corregida, eliminando el ángulo de inclinación. (b) Nube de puntos con inclinación corregida de un camión.

3.5.4 SEGMENTACIÓN Y EXTRACCIÓN DE LA REGIÓN DE INTERÉS EN LA INFORMACIÓN DE RANGO

En esta etapa del trabajo es importante la realización de un proceso de extracción de la información relevante presente en las nubes de datos originales; lo anterior es necesario por varios motivos que se enlistan a continuación: reducción de la carga computacional, comparación de elementos más precisa y la separación de los vehículos presentes en cada captura. En las siguientes subsecciones son presentados cada uno de los procesos realizados, así como ejemplos resultantes de cada etapa.

3.5.4.1 Filtrado de distancias

Como se mencionó anteriormente, en la nube de puntos original producida por el registro de los escaneos independientes, hay presencia de elementos no deseados (diferentes a los que se eliminarán mediante clusterización-umbralización), o zonas que no representan ningún interés para el proceso de clasificación, esto es, otros carriles, zonas alejadas de la región de interés etc. Siempre y cuando se haya eliminado el ángulo de inclinación de la nube de puntos causado por la posición incorrecta del sensor, todas estas zonas innecesarias pueden ser eliminadas mediante un filtro de distancias, es decir, una etapa de umbralización de las distancias en cada dirección de la nube de puntos. El proceso anterior es posible debido a que se conoce con anterioridad las dimensiones y ubicaciones de las regiones que atraviesan los vehículos terrestres al momento de la captura del dataset. El marco de referencia en la nube de puntos es el siguiente: **x** corresponde al valor de rango *r*, **y** es perpendicular a la superficie de la carretera y **z** es paralelo al desplazamiento de los vehículos. En la **Figura 3.29(c)** se presenta una nube de puntos conteniendo dos vehículos con el ángulo lateral de inclinación corregido (la inclinación es por el punto de vista en el visualizador), el proceso de filtrado de distancias es llevado a cabo siguiendo el esquema básico presentado en la **Figura 3.33**.



Figura 3.33 Filtrado de distancias para la nube de puntos con información del sensor calibrado y sin inclinación.

En la **Figura 3.34** es posible observar el resultado, del filtrado de distancias, se aprecia como la información elimina regiones no deseadas y deja el camino preparado para la segmentación exacta de los vehículos presentes en la nube de puntos.

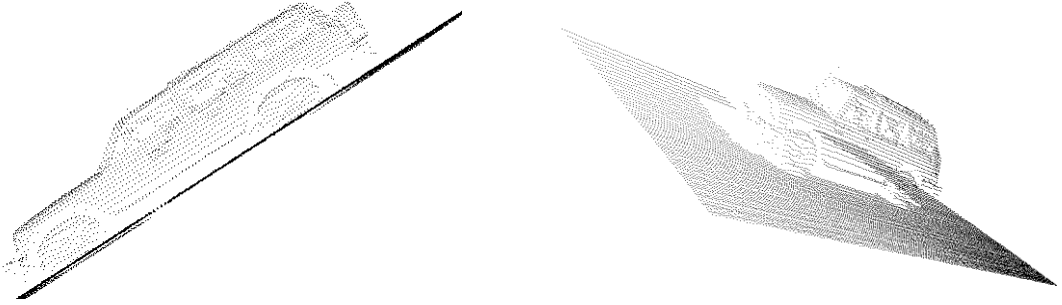


Figura 3.34 (a, b) Nubes de puntos resultantes del proceso que utiliza el filtro de distancias, como puede observarse, esta etapa elimina las regiones innecesarias que no contribuyen al proceso de clasificación, logrando extraer solo los vehículos en el carril principal, y la superficie de la carretera, donde reposan.

3.5.4.2 Extracción de planos

Uno de los procesos más comunes en la segmentación de nubes de puntos 3D, es la estimación de modelos de forma (planos, cilindros, rectángulos, círculos, esferas, conos, etc.) que sean fácilmente parametrizables, facilitando la detección de elementos y la generación de hipótesis sobre los tipos de modelos presentes en la escena. El caso más común, es el ajuste e identificación de modelos planares (referirse a la sección 2.2.3.1 para mayores detalles en estas técnicas), con la intención de eliminar la superficie donde reposan los elementos de interés en gran mayoría de aplicaciones de clasificación (mesas, carreteras, suelo en general, tablas). En este trabajo, la extracción de los modelos planares principales, es necesaria por dos razones, en primer lugar, la necesidad de eliminar gran cantidad de información sin valor presente en la carretera, en segundo lugar los descriptores globales (VFH, ESF) requieren que la nube de puntos contenga exacta y únicamente el objeto de interés, Ya que por ejemplo un mismo vehículo, pero en dos entornos diferentes (diferentes nubes de puntos en el exterior vehículo) representan dos firmas globales totalmente distintas. Teniendo en cuenta las necesidades de nuestro proyecto en el cual las capturas poseen una superficie planar dominante horizontal, se lleva a cabo la implementación de la técnica presente en el trabajo de (Radu Bogdan Rusu, Blodow, Marton, & Beetz, 2009), la cual funciona mediante la realización de un consenso de muestras (RANSAC3D); esta idea es presentada en (Nüchter & Hertzberg, 2008). El procedimiento es descrito en el **Algoritmo 3.3** para una nube P .

Algoritmo 3.3 Extracción de modelo planares mediante RANSAC 3D (Consenso de muestras aleatorias).

1.	$Normal3DEstimation(P);$	// Estima vectores normales
2.	Si $(n_i Y_{axis})$ Entonces	// ¿Son Paralelos?
3.	$ClusterizacionEuclidiana(P);$	// Agrupa K-vecindades
4.	$T^i = \{t_1^i, \dots, t_n^i\};$	// Puntos hipótesis de planos
5.	$RansacConsensoMuestras(T);$	// (Nüchter & Hertzberg, 2008)
6.	Sino Entonces	// No paralelos
7.	$ClusterizacionEuclidiana();$	// Agrupa K-vecindades
8.	$C^i = \{c_1^i, \dots, c_n^i\}$	// Puntos exteriores al plano
9.	$ExtraeClusteres(C);$	// Obtiene Clústeres vehículos
10.	Fin si	

El primer componente del algoritmo $Normal3DEstimation(P)$, estima las normales de superficie para cada una de las k-vecindades centradas en cada punto $p_i \in P$, donde P representa la nube de puntos, la orientación de estas normales es comparada con la orientación del eje central vertical (consideración que obtiene solamente planos

perpendiculares a este eje, en este trabajo Y axis), el paso anterior permite llevar a cabo un proceso de clusterización de dos tipos diferentes de conjuntos denominados T y C , el primero refiere a los puntos pertenecientes a los elementos planares a descartar, y el segundo a los elementos de interés a clasificar más adelante. La clusterización es realizada mediante el proceso descrito en la sección 3.5.4.3 3.5.6 .

1. Seleccionar tripletas de puntos ($p_1, p_2, p_3 \in R^3$) o un punto 3D (p_1, n , con $\|n\| = 1, p_1, n \in R^3$).
2. Para detectar una superficie planar, el algoritmo selecciona aleatoriamente un punto y estima un plano a través de 2 puntos vecinos, que cumplan $|(x - p_1) \cdot n| < Eth$, donde Eth equivale a la máxima diferencia de orientación aceptada, entre la normal del plano (modelo) y el vector entre dos puntos hipótesis en el plano,
3. Si el número de puntos encontrados excede un número determinado por el usuario, se prepara la optimización ICP (*Iterative Closest Point*); los puntos obtenidos son proyectados en el modelo del plano y se realiza la optimización, obteniéndose finalmente el ajuste correcto del modelo planar.

El resultados obtenido de la estimación de planos en nubes de puntos aplicando RANSAC3D se aprecia en la **Figura 3.35**, se observa, la correcta detección del modelo planar principal dominante, en una nube de puntos con un vehículo SUV.

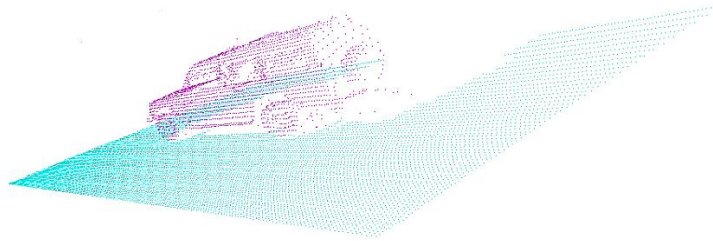


Figura 3.35 Segmentación del modelo planar dominante en una nube de puntos, el plano determinado se aprecia en color azul claro, y el elemento restante a reconocer (vehículo SUV) en morado.

3.5.4.3 Clusterización

En los sistemas de procesamiento de información 3D, uno de los principales factores que inciden negativamente en la complejidad computacional y el tiempo de computación de los algoritmos, es la gran dimensión de la información que debe ser procesada (nubes de puntos), pues cada proceso debe ser aplicado a cada punto $p_i(x, y, z)$ perteneciente a la nube P ; en este trabajo el proceso de la clusterización es necesario por dos razones adicionales, en primer lugar, las nubes de puntos obtenidas en el dataset tienen longitudes diferentes en cada caso (esto es, el número de escaneos por cada dato o captura del *dataset*), debido a que la adquisición es diferente para dato, pues depende de condiciones como la cantidad de vehículos que se deseaban capturar por toma, o el tiempo que se ejecutó el sistema de adquisición en cada caso; lo anterior ocasiona vehículos presentes en cada captura se encuentran distribuidos en diferentes regiones de la nube de puntos original, y realizar una comparación o clasificación directa con los datos originales, llevaría a la obtención de resultados erróneos, además de un tiempo de procesamiento demasiado alto. En segundo lugar, es importante llevar a cabo la clusterización, porque los descriptores globales (definidos adelante en la sección de descriptores 3D) requieren que los elementos a describir (y posteriormente a clasificar) se encuentren totalmente identificados y separados

de la nube de puntos original, pues su concepción incluye un único histograma descriptor que es variante ante la distribución de los puntos en la nube de entrada.

Uno de los algoritmos básicos en el proceso de clusterización, es el denominado *clusterización por distancia euclidiana* definido en (R. Rusu, 2010), este algoritmo, obtiene buenos resultados si los elementos a identificar, se encuentren claramente separados por una distancia mínima conocida aproximada, y si las partes que lo componen como elemento se encuentran distanciadas una a otra una distancia menor, a la que se para los clústeres; además requiere que el plano principal donde reposan los elementos, haya sido previamente segmentado o extraído del proceso. En el enfoque de este trabajo, todas las condiciones previamente definidas son satisfechas, por lo que no es necesario el uso de procesos de clusterización más complejos como los definidos en (Sareen, Knopf, & Canas, 2010; Strom, Richardson, & Olson, 2010; J. X. Zhang & Lin, 2012), otros basados en colores de segmentación como (Zhana, Liangb, & Xiaoa, 2009), métodos de grafos (Strom et al., 2010), u otros también definidos en (R. Rusu, 2010), como *clusterización basada en crecimiento de regiones* o la *clusterización por distancia euclidiana condicional*.

Los componentes del proceso realizado para la clusterización euclidiana se presentan y describen a continuación en la **Algoritmo 3.4**.

Algoritmo 3.4 Esquema del proceso para la clusterización euclidiana.

1.	CreaArbolKD(P);	// Organización de los puntos en un esquema de árbol k-dimensional
2.	$C \leftarrow \{0 \dots 0\}; Q \leftarrow \{0 \dots 0\};$	// Crea lista de clústeres y Cola de procesamiento
3.	$P_k \leftarrow \{0 \dots 0\};$	// Crea Conjunto K-vecinos
4.	Para Cada $p_i \in P$ Hacer	// Itera por cada punto de la nube
5.	$Q \leftarrow p_i;$	// Agrega el punto a la cola
6.	Para Cada $p_i \in Q$ Hacer;	// Itera por cada punto de la cola
7.	$P_k \leftarrow \text{BuscaKvecinos}(D_{\max});$	// Busca conjunto k-vecinos en una esfera de radio $D_{\max}$
8.	Si ($\neg P_k \text{OK}(P_k)$) entonces	// Si el punto no ha sido procesado
9.	$Q \leftarrow p_i;$	// Adicione el punto a la cola
10.	Fin Si	
11.	Fin Para Cada	
12.	Si(QOK(Q)) entonces	// Si todos los puntos de la cola han sido procesados
13.	$C \leftarrow Q; Q \leftarrow \{0 \dots 0\};$	// Adicione la cola a la lista de clústeres, y limpie la cola
14.	Fin Si	
15.	Fin Para Cada	
16.	POK(P)	// Revisa Si todos los puntos han sido asignados a un clúster

Los resultados exitosos de la aplicación de la clusterización por distancia euclidiana a las nubes de puntos con los datos corregidos, filtradas en distancia y con los planos principales segmentados o removidos, son presentados en la **Figura 3.36** a y b, en la que se observa cómo en una nube de puntos con dos vehículos (un sedán y un hatchback), donde el plano de la carretera ha sido previamente segmentado, los dos vehículos son detectados y separados en dos clústeres diferentes (dos colores distintos); en la **Figura 3.37** se observa un caso aún más complejo donde se presentan 6 clústeres distintos correctamente detectados (dos motos, un hatchback, una SUV, y un campero). Es importante resaltar que esta etapa cumple una función adicional de filtrado de elementos no deseados, al ser combinada con un proceso de umbralización, que elimina los clústeres obtenidos con muy pocos puntos, o con densidades muy bajas, así pues, se logra eliminar personas, ramas u objetos sin ningún valor para el trabajo. Cada uno de los elementos clústeres finales

sobrevivientes al proceso unido de clusterización-umbralización, es extraído en nubes de puntos independientes como se observa en la **Figura 3.38**, y su centroide de masa es localizado, para realizar una traslación que ubique el centroide en el origen del sensor, por cuestiones de visualización y facilidad de manipulación; el último paso es también útil para la realización de una ponderación basada en la distancia al centroide de cada punto clave del vehículo al momento de la clasificación.

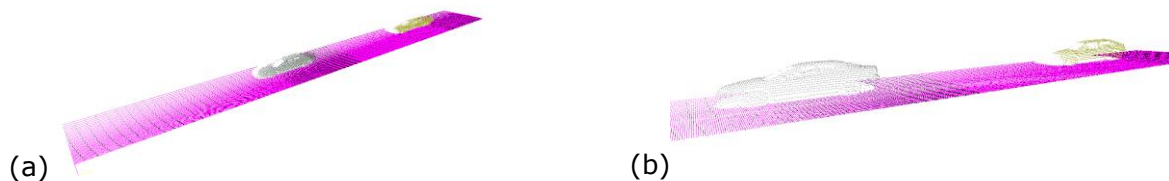


Figura 3.36 (a, b) Clusterización euclidiana para una nube de puntos con dos vehículos (a.1, b.1) sedan gris, (a.2, b.2) hatchback rosa, el plano principal ha sido previamente segmentado (a.3, b.3) color rosa.

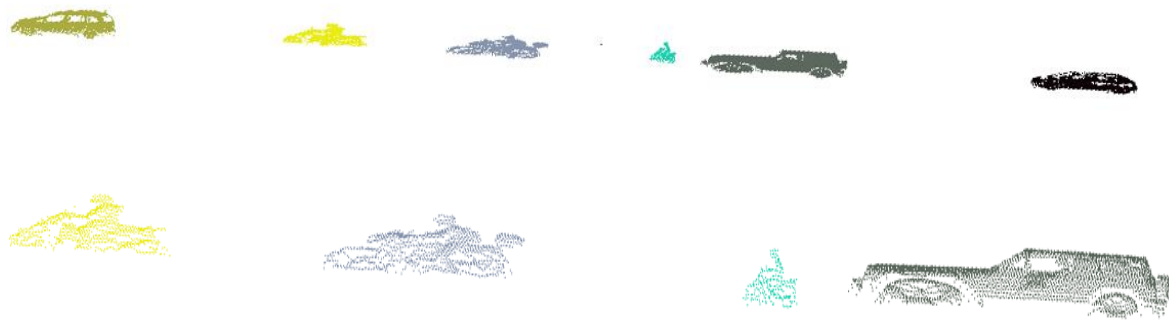
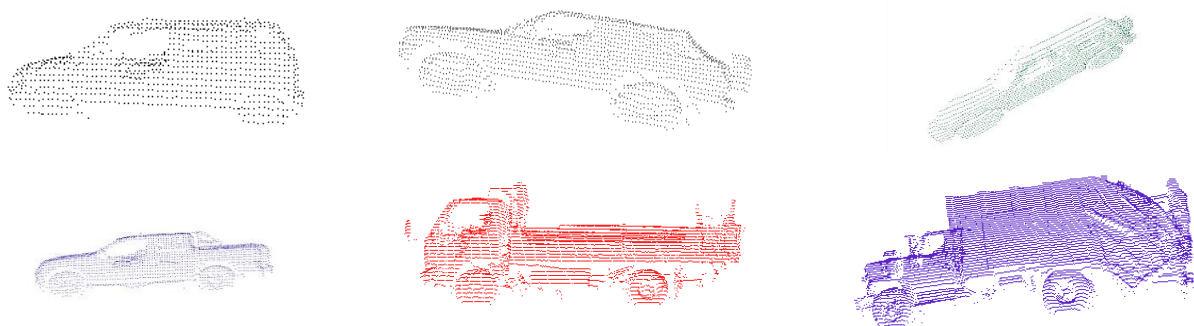


Figura 3.37 (a, b) Clusterización euclidiana de un caso más complejo con 6 clústeres distintos (dos motos, un hatchback, una SUV, y un campero) el plano ha sido removido, por cuestiones de visualización.



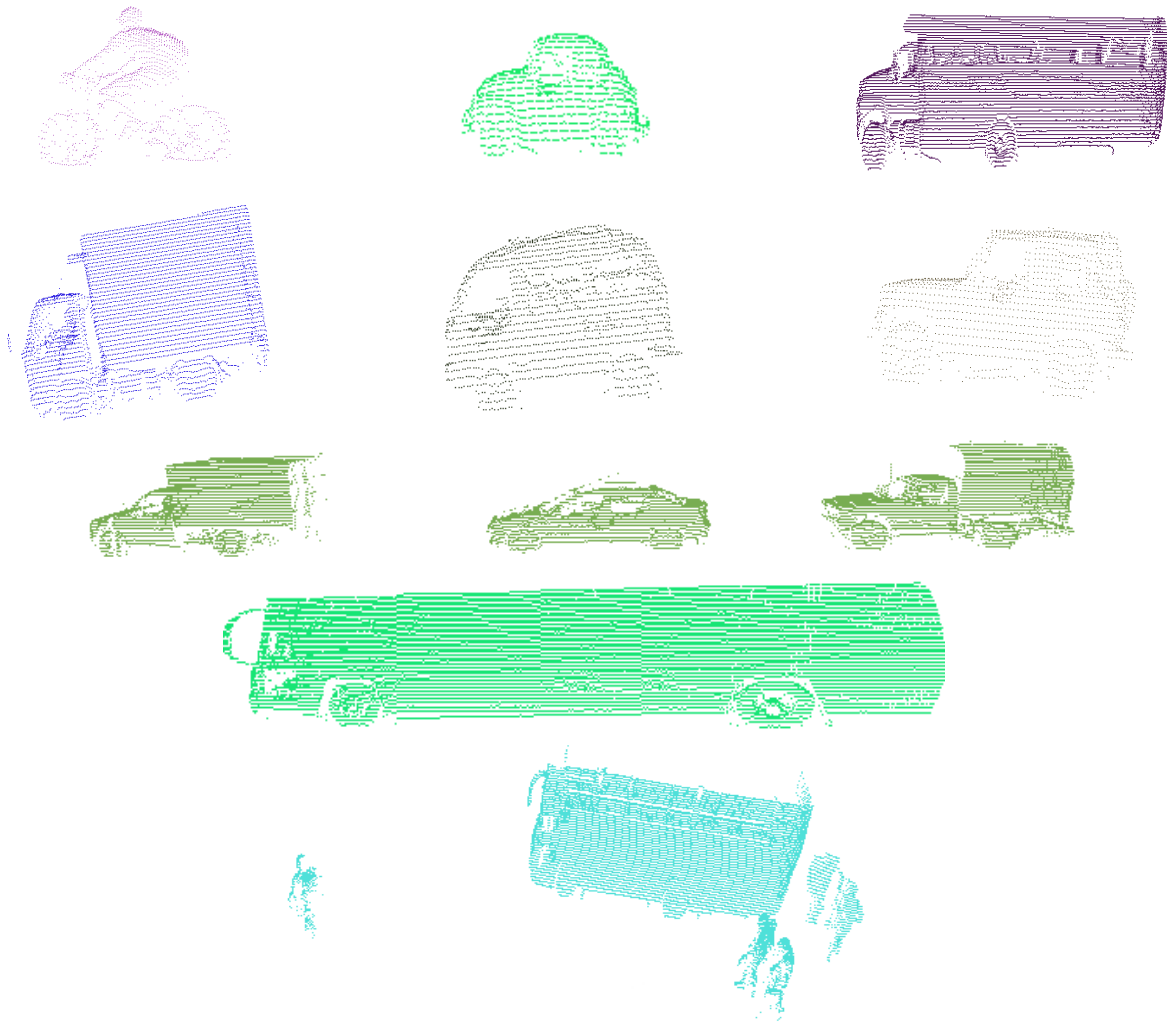


Figura 3.38 Ejemplos de Clústeres correspondientes a vehículos extraídos de nubes de puntos diferentes.

3.5.5 OBTENCIÓN DE LOS PUNTOS CLAVE

Esta etapa del proceso consiste en la selección de los puntos de la nube segmentada que serán sometidos a los algoritmos de extracción de descriptores 3D; esto es realizado con la finalidad de alivianar la carga computacional buscando características solo en puntos interesantes (geométricamente y descriptivamente), o sencillamente muestreando la información original teniendo en cuenta la densidad de cada clúster correspondiente a un vehículo. Siguiendo las ideas presentadas la tarea de esta etapa es abordada en este trabajo a través de dos métodos diferentes, y dada la modularidad del sistema, puede seleccionar cualquiera de los dos métodos según se desee, dadas las características del entorno de trabajo.

3.5.5.1 Selección de puntos clave mediante muestreo uniforme de la información.

El primero de los métodos aplicados en este trabajo, (Aldoma et al., 2011, 2012; R. Rusu, 2010; Radu Bogdan Rusu et al., 2009; Radu Bogdan Rusu, Marton, Blodow, Dolha, & Beetz,

2008; Radu Bogdan Rusu, Marton, et al., n.d.)(Aldoma et al., 2012; Federico Tombari et al., 2010; Wohlking & Vincze, 2011), busca reducir la cantidad de información original mediante un proceso de muestreo uniforme teniendo en cuenta la resolución y densidad de la nube de puntos original, este enfoque realiza una descomposición de la nube de puntos original en voxels (pequeñas cajas 3D), realizado este paso se puede aproximar los puntos contenidos en el voxel, mediante el centro del voxel, o mediante el centroide de los puntos contenidos en el voxel, la última opción siendo más lenta pero más precisa. Los resultados de este proceso de selección se observan en la **Figura 3.39**, es claro que este método tiene en cuenta la distribución de los puntos a lo largo de la nube y reduce significativamente la carga computacional.

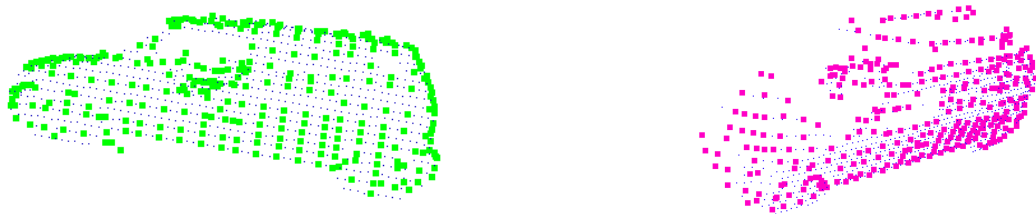


Figura 3.39 Resultado de la etapa de selección de puntos claves en la información de rango de un vehículo (a, b, visto desde diferentes perspectivas) mediante la técnica de muestreo uniforme.

3.5.5.2 Selección de puntos clave teniendo en cuenta los bordes presentes en la información de rango.



Figura 3.40 a)Traslación al centroide desplazado de la nube de puntos; b)Imagen de rango obtenida mediante la librería PCL, con una resolución de 1 grado por pixel, y un marco de referencia trasladado a el centroide de la imagen (c_x, c_y, c_z), más una traslación en el eje x (profundidad, eje perpendicular al lado del carro).

Esta técnica (Steder & Konolige, n.d.; Steder & Rusu, 2011) fue diseñada basada en las características de la información 2.5D, esto se refiere a información 3D de vistas parciales de los elementos y no a la nube de la forma completa del elemento; esta condición es característica de nuestro trabajo también, pues la información es capturada desde un único sensor de rango, logrando vistas laterales y superiores del vehículo. Así pues los puntos clave son buscados en los bordes detectados en la información de rango, definidos estos como la presencia de puntos consecutivos cuyas vecindades, presentan cambios bruscos pero bien definidos en la medida de rango. Es importante resaltar que para la realización de este método es necesario primeramente, la creación de una imagen de rango a partir de la nube de puntos, estableciendo la resolución por pixel y los rangos de escaneo; una imagen de rango es una matriz 2D que codifica la profundidad desde un único punto de vista mediante intensidades o colores en la imagen. Es importante definir un punto de vista adecuado para su creación, lo que en este trabajo se logró, mediante la traslación del origen del sensor hacia el centroide de la nube de puntos pero desplazado en el eje x (profundidad) hacia el exterior del vehículo, un ejemplo de una imagen de rango obtenida mediante la librería PCL es mostrada en la **Figura 3.40** (Steder, 2013.). La extracción de los bordes y los posteriores puntos clave es definida en el esquema del **Algoritmo 3.5**.

Algoritmo 3.5 Esquema resumen del procedimiento de obtención de bordes y puntos claves.

1.	$P_{Claves} \leftarrow \{0 \dots 0\}; P_{Bordes} \leftarrow \{0 \dots 0\};$	// Inicializar Conjunto de puntos clave y bordes
2.	$FactorClave = x; FactorBorde = y;$	// Definir Umbral borde y punto clave
3.	Para Cada $p(u, v)_i \in IRango$ Hacer	// Itera Para los puntos en la imagen de rango
4.	$S_{dir} = \max(0, 1 - \frac{\delta}{\delta_{dir}}).$	// Obtención del Índice de calidad y probabilidad de Bordes en las vecindades de la distancia máxima 3D δ
5.	$S'_{dir} = \max(0.9, 1 - (1 - S_{sombra})^3) S_{dir}.$	// Modificación de los índices teniendo en cuenta la distancia al punto de vista
6.	Si ($S'_{dir} > FactorBorde$) entonces	
7.	$P_{Bordes} \leftarrow p$	
8.	Fin Si	
9.	$I_1(p) = \min_i(1 - w_{n_i} \max(1 - \frac{10 \cdot \ p - n\ }{\sigma}, 0))$	// Factor de calidad de punto clave 1
10.	$f(n) = \sqrt{w_n(1 - \left \frac{2 \cdot \ p - n\ }{\sigma} - \frac{1}{2} \right)}$	// Itera por cada punto de la nube
11.	$I_2(p) = \max_{i,j}(f(n_i)f(n_j)(1 - \cos(\alpha'_{n_i} - \alpha'_{n_j})))$	// // Factor de calidad de punto clave 2
12.	$I(p)$	// Factor de calidad final de los puntos en los bordes
13.	Si ($I(p) > FactorClave$) entonces	
14.	$P_{Claves} \leftarrow p$	
15.	Fin Si	
16.	Fin Para Cada	

En el esquema del **Algoritmo 3.5** se observa una primera etapa en la que se obtienen factores de calificación direccionados (independientemente calculados para cada dirección) basados en la distancia máxima 3D δ , desde el punto en particular, hacia sus vecinos en una grilla 2D, en la imagen de rango (selección de vecindad en 2D, distancias en 3D) y en la distancia 3D desde el punto hacia la ubicación promedio de los puntos en la vecindad direccionada. Posteriormente se obtiene un factor de calificación de bordes adicional que indica la presencia de un borde de sombra (marcas de oclusión generadas por el elemento) o de velo (intermedio) basado en las distancias desde el punto de vista, teniendo en cuenta este nuevo valor S_{sombra} , se recalcula el factor de calificación S_{dir} y se umbraliza para determinar, si se es un punto en el borde del elemento, o es un borde intermedio de velo. (**Figura 3.41**) Finalmente se realiza un proceso de estimación de factor de calidad $I(p)$ que después de una umbralización permita reducir correctamente el número de puntos claves teniendo en cuenta 2 factores: a) tenga en cuenta los bordes y b) sean puntos visibles desde otros puntos de vista no iguales pero similares (donde es visible el lado del carro). Este procedimiento necesita cuantificar las normales de superficie locales para los puntos, presentes en un tamaño de soporte 2D, que corresponde a una vecindad cuadrada (el cálculo de las normales es hecho en 3D) de cada punto, para esta vecindad es estimada las direcciones de las normales de superficie y se obtiene el factor de curvatura λ , para calcular el peso de cada punto $w = 1 - (1 - \lambda)^3$, finalmente se proyectan las normales de los vecinos en un soporte 3D σ en un plano perpendicular a la dirección del punto de vista y se cuantifican los ángulos α de cada vector normal unidimensional; con todos estos elementos puede cuantificarse el factor de calificación y realizar la umbralización para obtener los puntos claves, como los observados en la **Figura 3.42** para una mayor descripción en detalle se recomienda la lectura de trabajo citado.



Figura 3.41 Ejemplos de los bordes detectados resultantes en los vehículos siguiendo el esquema del trabajo de (Steder & Rusu, 2011). Los bordes verdes son los bordes del plano del elemento de interés, los rojos son los de velo o intermedios, y los azules los de oclusión.



Figura 3.42 Puntos claves detectados en los vehículos gracias a los índices o factores de calidad determinados en (Steder & Rusu, 2011).

3.5.6 ESTIMACIÓN DE NORMALES DE SUPERFICIE

En la sección 3.5.5 se presentó el proceso de extracción de puntos clave, ahora es necesario asignar a cada punto información sobre la características de su vecindad local, esto se requiere para la posterior extracción de los descriptores 3D (a excepción de ESF, ver sección 3.5.7.2.2). El enfoque implementado en este trabajo es similar al desarrollado en los trabajos (Ciocarlie et al., n.d.; Holzer et al., 2012; Hoppe et al., 1992; K. Klasing, Althoff, et al., 2009; König & Gumhold, n.d.; Mitra et al., 2003; R. Rusu, 2010; Teichman, Levinson, & Thrun, 2011).

Una vez se ha logrado la correcta segmentación y clusterización de los vehículos en cada escena es necesario someter cada elemento a los procesos de extracción de características tanto locales, como globales. A excepción del método global *ESF* (ver sección en el capítulo 2), todos los descriptores hacen uso en cierta medida de las características geométricas de las superficies a describir denominada normal de superficie, esto es, la orientación en un marco de referencia, de la superficie local definida para la vecindad p^k obtenida para el punto p_i (origen del vector normal); análogamente el vector normal al plano tangente estimado en la vecindad p^k .

Aunque actualmente existen diferentes técnicas, para la estimación de las normales de superficie véase (K. Klasing, Althoff, et al., 2009) y el capítulo 2, la técnica más común en la bibliografía y con los resultados más consistentes, es basada en el enfoque de *PCA* (*Principal Component Analysis*) y es descrita como sigue: el problema de estimar una normal de superficie es aproximado por la estimación de un plano tangente a la superficie, lo cual se convierte en un estimación de mínimos cuadrados de un plano en la vecindad p^k . El plano es representado como un punto x y un vector normal $\vec{n}$, y una distancia desde un punto $p_i \in P^k$

al plano, definida como $d_i = (p_i - x) \cdot \vec{n}$. Los valores de x y $\vec{n}$ son calculados en el sentido de mínimos cuadrados, minimizando para obtener $d_i = 0$ tomando:

$$x = \bar{p} = \frac{1}{k} \cdot \sum_{i=1}^k p_i \quad \text{Ecuación 3.12}$$

Como el centroide de P^k , la solución para $\vec{n}$ es dada analizando los valores y vectores propios de la matriz de covarianza $C \in R^{3 \times 3}$ de P^k expresada como:

$$C = \frac{1}{k} \cdot \sum_{i=1}^k \alpha_i (p_i - \bar{p}_i) \cdot (p_i - \bar{p}_i)^T, \quad C \cdot \bar{v}_j = \lambda \cdot \bar{v}_j, j \in \{0,1,2\} \quad \text{Ecuación 3.13}$$

El termino α_i representando un posible peso para p_i y es usualmente 1, C es simétrica y Semi-definida positivamente, y sus valores propios son números reales. Los vectores propios $\bar{v}_j$ forman el marco ortogonal y $\bar{v}_0$ correspondiente al más pequeño de los valores propios, es la aproximación del vector normal $\vec{n} = \{nx, ny, nz\}$, (R. Rusu, 2010); el signo del vector es ambiguo, pero su desambiguación es trivial si se conoce el punto de vista v_p satisfaciendo $\vec{n} \cdot (v_p - p_i) > 0$.

Las implementaciones del cálculo de las normales de superficie y los posteriores descriptores 3D requiere un paso adicional de obtención de la resolución o densidad de la nube de puntos, esto es debido a que dadas las diferentes velocidades con que cada vehículo atraviesa el haz de escaneo, la distribución métrica de los planos de puntos es variante de un vehículo a otro.

El procedimiento para obtener la resolución de la nube de puntos es descrito en el diagrama de la **Figura 3.43**.

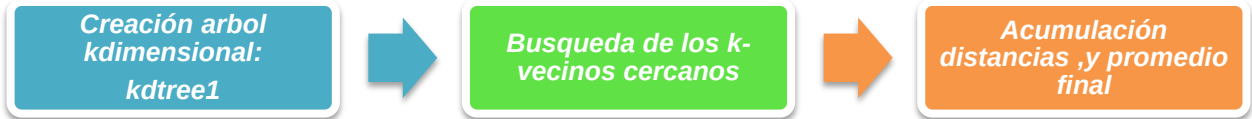


Figura 3.43 Esquema para la obtención de la resolución de la nube de puntos.

En primer lugar se realiza la creación de un árbol k-dimensional con el objetivo de estructurar la distribución de los datos de tal manera que la búsqueda de las k-vecindades cercanas de cada punto sea efectiva y rápida (optimización), en segundo lugar se busca el vecino más cercano de cada punto y se acumula esta distancia en un acumulador global, finalmente se divide el acumulador global entre el número de puntos, obteniéndose la resolución de cada nube que permitirá definir los ángulos de soporte de búsqueda de cada descriptor 3D y de la vecindad P^k a la que se le estimara un modelo planar apropiado

El procedimiento de la extracción de las normales de superficie comienza entonces con la obtención de la densidad de la nube de puntos, para establecer el rango de búsqueda de las k-vecindades, y luego procede con la estimación como fue descrita previamente, los resultados exitosos son apreciados en la **Figura 3.44**, donde se observa los vectores paralelos en las superficies planas del vehículo y los cambios de orientación en los bordes del mismo.

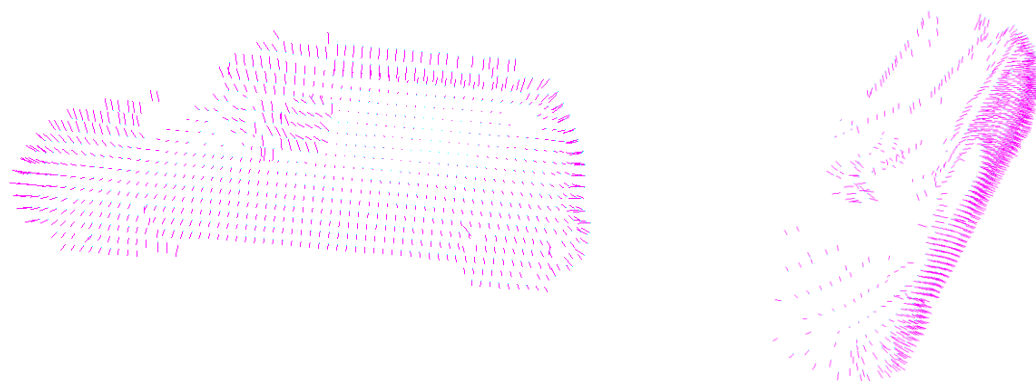


Figura 3.44 Resultado de la extracción de las normales de superficie, desde dos puntos de vista diferentes, para un vehículo hatchback.

3.5.7 EXTRACCIÓN DE DESCRIPTORES 3D EN LA NUBE DE PUNTOS DE LA INFORMACIÓN DE RANGO

Los pasos desarrollados hasta ahora, han sido diseñados como etapas de ajuste y complementación de la información original para la extracción exitosa y discriminante de diversos tipos de descriptores 3D que sirva como entrada para un clasificador final, o como elemento de interés en las estrategias de búsqueda y retribución. Existen dos tipos de líneas de procesamiento basadas en dos clases diferentes de clasificadores 3D, el esquema de procesamiento local y el global; a pesar de que las dos estrategias pueden entregar solución a problemas genéricos como lo es el reconocimiento 3D, cada línea ha sido diseñada adicionalmente para la solución a ciertas tareas y aplicaciones específicas, a continuación se establecen las características de cada línea y las etapas de su cadena de procesamiento, posteriormente se observan los resultados en este trabajo, para la extracción de descriptores locales y globales mediante distintos métodos pertenecientes a cada línea. Es importante mencionar que en este trabajo la clasificación es realizada para los dos tipos de esquemas, tanto global (evaluación individual de los descriptores VFH y ESF) como local (evaluación individual de los descriptores FPFH, NARF y SHOT), para cada uno son desarrollados los módulos adicionales de clasificación según corresponde, en esta sección.

3.5.7.1 Descriptores 3D locales

Han sido desarrollados para aplicaciones específicas, como lo son: el registro de múltiples nubes de puntos (integración y construcción de nubes complejas, basadas en nubes más pequeñas), categorización local de superficie (detección de distintos tipos de geometrías en nubes con múltiples clústeres, sin segmentar) y también para aplicaciones generales como el reconocimiento 3D. Para las aplicaciones específicas mencionadas, se necesitan en ambos casos las características en el nivel mínimo de granularidad de la información ya que es necesario realizar el pareo o la generación de correspondencias que permitan intersectar las nubes en los puntos correctos, y describir las geometrías locales mediante la intersección de vecindades similares, respectivamente. Concluyendo lo expresado este enfoque se tiene que la información de cada punto perteneciente a la nube original es complementada con información sobre la geometría local (esto es, de la k -vecindad definida del punto de interés). Si bien, se tiene acceso a un mayor nivel local de la información y se es independiente de un proceso previo de segmentación, aplicaciones más generales como el reconocimiento 3D requieren etapas adicionales de agrupación y generación de correspondencias, que permitan indicar la clasificación de cierta información 3D y en algunos

casos la pose 6DoF en la escena donde se encuentra el elemento. Resumiendo los pasos necesarios para la exitosa implementación de una clasificación basada en descriptores de tipo local, se presenta el esquema de la **Figura 3.45**

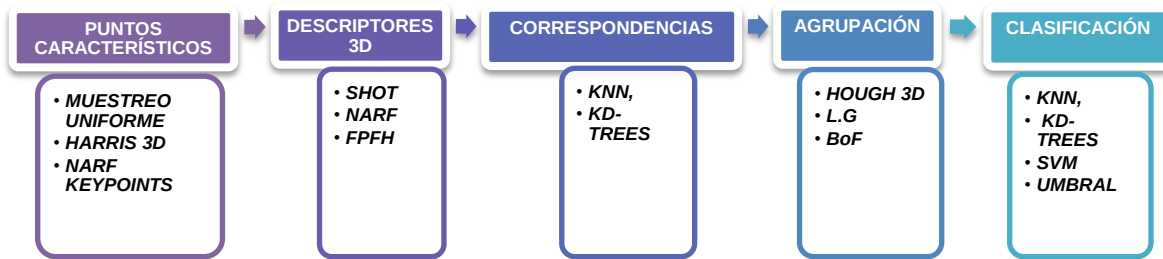


Figura 3.45 Diagrama de bloques de un sistema de clasificación basada en descriptores de tipo local.

En primer lugar, el hecho de que las nubes de puntos presenten dimensiones diferentes, y que la distribución de los vehículos y su número presente no sea igual para cada captura, exige una etapa de segmentación previa; esta es desarrollada la misma etapa de segmentación-clusterización para las dos estrategias, local y global; esto es útil también para generar una reducción en la carga computacional y aportar mayor robustez del sistema. Dicho lo anterior, una vez obtenidos los clústeres de los vehículos se extraen los puntos clave de la manera mencionada en las anteriores etapas (en este trabajo realizado en base a dos técnicas *muestreo uniforme* y *puntos clave NARF*), y este procedimiento es seguido con la extracción de los diferentes descriptores que se presentaran en breve (*SHOT*, *NARF*, *FPFH*); los descriptores locales, entregan igual número de histogramas acumuladores que el número de puntos en la nube del clúster (cada histograma con una longitud finita, definida en la siguiente sección). Después de que se han calculado los descriptores para la nube de puntos es necesaria la realización de dos etapas que actúan como una sola combinada y simultánea para la agrupación o la integración de correspondencias (relación de pares similares entre los descriptores locales), este proceso puede ser afrontado mediante 3 técnicas diferentes Condiciones geométricas (H. Chen & Bhanu, 2007), Hough 3D (F. Tombari & Stefano, 2010) y bolsa de características (Sivic & Zisserman, 2003), en este trabajo se decidió utilizar la última de ellas, debido a su robustez significativa, su potencialidad en clasificación intracalse, y a los niveles de ponderación de frecuencias que ofrece la técnica (lo que brinda importancia a las características más escasas, pero posiblemente más discriminantes).

Construcción de correspondencias e integración mediante bolsas de características.

1. Selección de nubes de bases de datos de diferentes modelos (conocidas sus categorías, y esto es indicado en sus campos de información).
2. Elaboración de diccionario visual, mediante k-means.
3. Obtención de los parámetros de ponderación
4. Selección nubes de base de datos de datos de diferentes modelos (conocidas sus categorías, y esto es indicado en sus campos de información).
5. Detección de correspondencias mediante KNN y arboles k-dimensionales
6. Integración de correspondencias mediante vectores ponderados de frecuencias de palabras visuales.

Finalmente la última etapa de esta rama de clasificación global lleva a cabo la clasificación, e puede ser llevada a cabo mediante búsquedas de los vectores más cercanos a través de KNN con árboles k-dimensionales o mediante clasificadores entrenados como *SVMs* o *redes neuronales*. En este trabajo, con finalidad de comparación y de búsqueda de la mayor robustez posible, son aplicados 3 descriptores locales diferentes, FPFH, SHOT y NARF, la descripción del descriptor y de la implementación en el trabajo, así como los histogramas resultantes, son presentados a continuación.

3.5.7.1.1 FPFH

Desarrollado en los trabajos (Radu Bogdan Rusu et al., 2009; Radu Bogdan Rusu, Marton, et al., n.d.), este descriptor es una extensión del descriptor (*Point Feature Histograms*, caracterizado por ser el histograma más descriptivo de la geometría local), optimizada para el uso en tiempo real reduciendo la complejidad computacional de $O(n * k^2)$ a $O(n * k)$, conservando casi totalmente la calidad descriptiva de *PFH*, y agregando información adicional externa a la vecindad del punto, mediante su complementación final de sumas aproximadas.

El proceso de obtención de este descriptor a partir de la información 3D de los vehículos es como sigue (referirse al capítulo 2):

1. Por cada punto p_q un conjunto de duplas $(\alpha, \varphi, \theta)$ entre el punto y sus vecino es calculado y este resultado es denominado SPFH.
2. En un segundo paso los k -vecinos en el radio r son determinados de nuevo y mediante los valores SPFH de ellos, se realizan sumas ponderadas mediante el inverso del factor $\sqrt{\exp\|p - pk\|}$ entre el punto y cada vecino para obtener el descriptor FPFH.

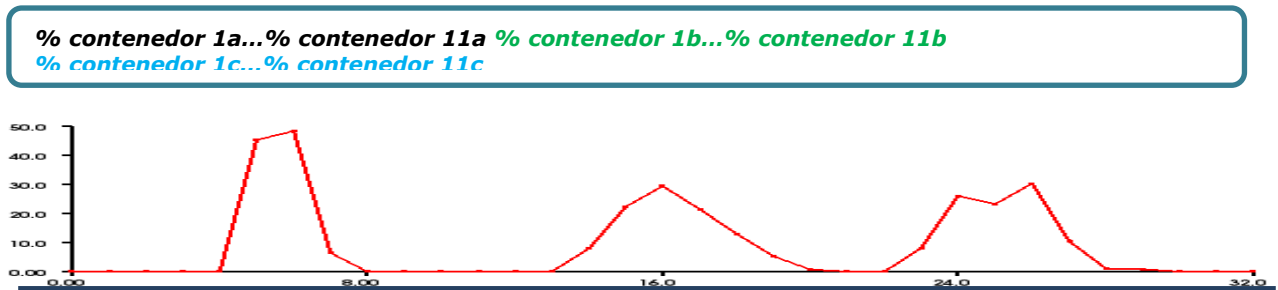


Figura 3.46 Ejemplo del descriptor FPFH para 1 punto en un clúster de un vehículo hatchback, se observa la presencia de 3 subhistogramas concatenados, uno para cada ángulo o característica, y los porcentajes por cada contenedor.

En la **Figura 3.46**, se presentan ejemplos de los resultados obtenidos en este trabajo para 3 diferentes puntos clave aleatorios en la nube, para un vehículo hatchback, así pues se presenta el macro histograma de 33 espacios, con 11 contenedores para acumular la cantidad de puntos de cada característica (3 ángulos), para la vecindad de cada punto, y los porcentajes de puntos en cada contenedor, en este trabajo fue asignado un radio de búsqueda de entre 5 y 10 veces la resolución de la nube. Concluyendo, este descriptor fue seleccionado por tres razones, en primer lugar al hacer parte del grupo de descriptores de geometría de superficie local, es el que entrega la mayor calidad de información sobre los comportamientos, en la orientación y curvatura de cada región del elemento de interés, describiendo muy bien cada cambio y forma del vehículo, en segundo lugar las diferentes optimizaciones que presentan lo hacen atractivo para un sistema de funcionamiento en tiempo real, donde la complejidad computacional es un factor relevante en el diseño. Finalmente en la bibliografía su uso es recomendado para información 2.5D como la de este trabajo, donde el rango es adquirido a través de un único punto de vista. A continuación se presenta el esquema básico de descriptor para cada punto, seguido de un ejemplo de archivo PCD (formato nube de puntos y descriptores en PCL) con formato FPFH (solo se muestra un descriptor de los 320 en el archivo).

3.5.7.1.2 SHOT

Este descriptor (Federico Tombari et al., 2010) ha sido diseñado basado en 4 principios o ideales que deben satisfacerse en las características que lo compongan, estos son: eficiencia de representación, descriptividad robusta, robustez ante el ruido y el desorden, además inmunidad ante las variaciones en la densidad de puntos de los elementos a describir. Este descriptor realiza una construcción de una firma de histogramas.

El descriptor es calculado a partir de las nubes de puntos 3D de los vehículos gracias a la serie de pasos que se describen a continuación (referirse al capítulo 2):

1. Superposición de una grilla esférica 3D isotrópica, en el soporte (vecindad de los k -vecinos).
2. Establecimiento de las particiones iguales a lo largo de los ejes radial, azimuth y de elevación, en total 32.
3. Por cada uno de los histogramas, acumular puntos en los contenedores a partir del ángulo entre cada uno de los vectores normales de la vecindad y el normal del punto clave.
4. Eliminación de los efectos de límites y fronteras, mediante una interpolación cuadrilinear de cada punto acumulado en los histogramas.

Para lograr robustez a las variaciones de la densidad de puntos, el descriptor es normalizado para sumar en su totalidad hasta uno, a diferencia de la normalización de cada contenedor con la inversa de la densidad de puntos y el volumen por contenedor. Este tipo de descriptor asume que las variaciones de densidad son globales o por lo menos regionales, manteniendo las diferencias muy locales de densidad como características discriminativas adicionales. A continuación se presenta el esquema básico de descriptor para cada punto, seguido de un ejemplo de archivo PCD (formato nube de puntos y descriptores en PCL) con formato SHOT (solo se muestra un descriptor de los 320 en el archivo). Al ser un descriptor local, se obtienen n_k histogramas en total por cada clúster a describir, uno por cada punto clave de la nube, en la **Figura 3.47**, se presentan ejemplos de los resultados obtenidos en este trabajo para 3 diferentes puntos clave aleatorios en la nube, para un vehículo *hatchback*, así pues se presenta el macro histograma de 352 espacios, con 11 contenedores para acumular la cantidad de puntos con diferentes ángulos entre las normales, en cada subdivisión (32 divisiones) de la grilla esférica isotrópica 3D para la vecindad de cada punto, y los porcentajes de puntos en cada contenedor, en este trabajo fue asignado un radio de búsqueda de entre 5 y 10 veces la resolución de la nube.

Las razones que motivaron al uso de este descriptor son: la invariancia ante las transformaciones rígidas, la consecución de un marco repetible y consecuente que permita el registrado congruente de las características geométricas y la invariancia a la distribución de densidad de puntos en la nubes, esta última es la más importante debido a que es una característica propia de nuestro sistema, dada las diferentes velocidades con las que los vehículos atraviesan el haz de escaneo 2D del láser,

% contenedor 1a...% contenedor 11a % %contenedor 1b...% contenedor 11b
% contenedor 1c...% contenedor 11c %... %contenedor 1k...% contenedor 11k marco de
referencia1... marco de referencia9

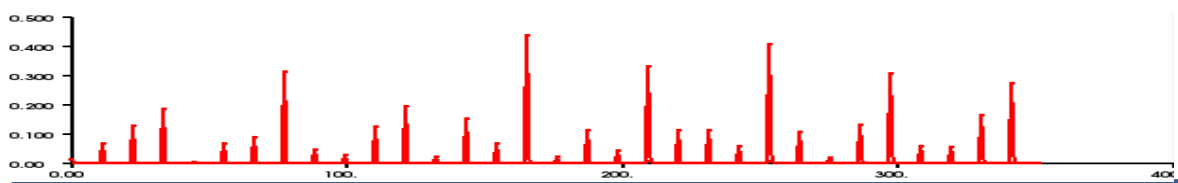


Figura 3.47 Ejemplo del descriptor SHOT para 1 punto en un clúster de un vehículo hatchback, se observa la presencia de 11 histogramas concatenados (uno para cada uno de las 32 divisiones de la grilla esférica) y los porcentajes por cada contenedor.

3.5.7.1.3 NARF

Los descriptores NARF (Steder & Konolige, n.d.; Steder & Rusu, 2011) describen el área alrededor de un punto interés de forma que permita la comparación eficiente teniendo en cuenta la máxima similaridad. Los criterios de diseño son: que capture la presencia del espacio ocupado y el vacío, permitiendo la obtención de la forma exterior de los elementos en la escena, robusto ante la presencia de ruido o elementos indeseados y que permita obtener un marco de referencia único y repetible.

El proceso de obtención combinado a partir de imágenes de rango y nubes de puntos 3D es como sigue (referirse al capítulo 2):

1. Calcula un parche normalmente alineado en el punto de interés, el cual es una pequeña imagen de rango, con el vector normal apuntando hacia el observador.
2. Establecer un patrón estrellado encima del parche normal alineado, donde cada rayo corresponde a un valor en el descriptor final, que captura el nivel de cambio de los pixeles de la imagen de rango bajo el rayo en cuestión. Para cada rayo son extraídas las celdas sobre las cuales este es proyectado, de esta forma es generado un set de celdas $\{c_0, c_1, \dots, c_m\}$, el valor de cada celda es basado en la transformación de los puntos en el soporte definido σ , mediante una proyección en el parche local, tomando como valor el rango (eje z, hacia el origen del sensor). El valor del descriptor es finalmente calculado siguiendo el esquema de ponderación siguiente.

$$\begin{cases} w(c_j) = 2 - \frac{2 * \|c_j - c_0\|}{\sigma} \\ D'_i = \frac{\sum_{j=0}^{m-1} (w(c_j) \cdot (c_{j+1} - c_j))}{\sum_{j=0}^{m-1} (w(c_j))} \\ D_i = \frac{\text{atan2}\left(D'_i, \frac{\sigma}{2}\right)}{180} \end{cases} \quad \text{Ecuación 3.14}$$

La última expresión permite la normalización del descriptor en un rango de (-0.5 a 0.5).

3. Extracción de una orientación principal y única del descriptor
4. Desplazar el descripto de acuerdo a la orientación principal para hacerlo invariante a la rotación

Al ser un descriptor local, se obtienen n_k histogramas en total por cada clúster a describir, uno por cada punto clave de la nube, en la **Figura 3.48**, se presentan ejemplos de los resultados obtenidos en este trabajo para 3 diferentes puntos clave aleatorios en la nube, para un vehículo hatchback, así pues se presenta el macro histograma de 36 contenedores, donde se registra las variaciones de rango en cada una de las direcciones del patrón estrellado súper impuesto en el parche local. En este trabajo fue asignado un radio de búsqueda de entre 5 y 10 veces la resolución de la nube.

Las razones que motivaron al uso de este descriptor son: Un proceso de extracción de distintos tipos de bordes, que permite la extracción de las siluetas de los elementos, y descriptores propios NARF como es desarrollado en las secciones anteriores, en el capítulo 2 y en la bibliografía; lo anterior es particularmente útil para este trabajo debido a que los elementos más discriminantes de los vehículos se encuentran en la silueta, más que en su interior. La invariancia ante las transformaciones rígidas, la consecución de un marco repetible y consecuente que permite la descripción robusta de los elementos. Otra

característica es su diseño e implementación específica y robusta para trabajar con imágenes en 2.5D una vez más dando relevancia a las características geométricas 3D de las siluetas, a diferencia de otros descriptores, que requieren modelos completos elaborados a partir de las vistas desde diferentes perspectivas. A continuación se presenta el esquema básico de descriptor para cada punto, seguido de un ejemplo de archivo PCD (formato nube de puntos y descriptores en PCL) con formato NARF (solo se muestra un descriptor de los 96 puntos clave descritos en el archivo).

Valor del cambio normalizado del contenedor del haz del patrón estrellado 1... valor de cambio normalizado del contenedor del haz del patrón estrellado 36

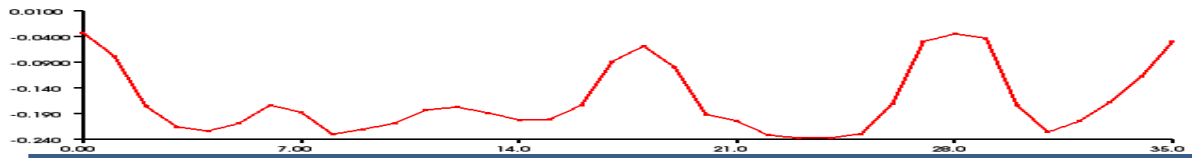


Figura 3.48 Ejemplo del descriptor NARF para 1 punto en un clúster de un vehículo hatchback, se observa la presencia de 36 contenedores en el descriptor NARF, uno por cada rayo del patrón estrellado súper impuesto, el valor representan la intensidad normalizada de cambio en el rango de las celdas bajo el haz en particular.

3.5.7.2 Descriptores 3D globales

Son representaciones multidimensionales de la geometría de los elementos que describen, han sido desarrollados para aplicaciones específicas, como lo son el reconocimiento de objetos, la categorización geométrica y la detección, segmentación y devolución de formas y no para relaciones y comparaciones punto a punto. En este enfoque se tiene que la información de cada descriptor representa un elemento en su totalidad como tal, y no ofrece descripciones locales o de subregiones. Esto facilita la comparación de correspondencias (1 comparación multidimensional), pero exige la segmentación del objeto de interés. Debido a que este tipo de descriptor no requiere etapas de agrupación de correspondencias, evita una de las etapas que puede contribuir negativamente en el reconocimiento de objetos debido a agrupaciones erróneas y geometrías parecidas en el interior de los elementos. Este descriptor presenta la desventaja de que es menos robusto a la oclusión, pues depende demasiado de la calidad de la segmentación y separación en escenas con gran concentración de elementos, además requiere generalmente, etapas adicionales para conseguir estimaciones robustas y precisas de la pose 6DOF. Resumiendo los pasos necesarios para la exitosa implementación de una clasificación basada en descriptores de tipo local, se presenta el esquema de la **Figura 3.49**.



Figura 3.49 Diagrama de bloques de un sistema de clasificación basada en descriptores de tipo global.

Como se han mencionado, el primer paso esencial en un proceso que involucre los descriptores globales es la segmentación y separación de los elementos a describir, presentes en la escena.

En primer lugar, son realizadas las etapas de las secciones anteriores, esto incluye segmentación y obtención de puntos claves. Este procedimiento es seguido con la extracción de los diferentes descriptores globales que se presentaran en breve (*ESF*, *VFH*); los descriptores globales, entregan un único histograma acumulador por cada nube del clúster (cada histograma con una longitud finita, definida en la siguiente sección). Después de que se han calculado los descriptores para la nube de puntos se procede con dos etapas que corresponden a la búsqueda-clasificación mediante la obtención de vectores correspondientes o los vectores vecinos más cercanos, es decir los que poseen mayor similitud en el marco modelo-escena.

La clasificación puede ser llevada a cabo mediante búsquedas de los vectores más cercanos a través de KNN con árboles k-dimensionales o mediante clasificadores entrenados como *SVMs* o *redes neuronales*. Para la clasificación en este trabajo se procede de la siguiente manera (se aclara que a cada nube ya se le ha aplicado el proceso de extracción de descriptores en esta etapa):

1. Selección nube de base de datos de modelo (conocida su categoría), o construcción de escena artificial con varias categorías conocidas.
2. Selección nube de base de datos de escena
3. Detección de similitud entre los dos descriptores mediante KNN y arboles k-dimensionales, clasificación.

En este trabajo, con la finalidad de la comparación y de la búsqueda de la mayor robustez posible dada las características del trabajo, son aplicados 2 descriptores globales diferentes VFH y ESF, la descripción del descriptor y de la implementación en el trabajo, así como los histogramas resultantes, son presentados a continuación.

3.5.7.2.1 VFH

Este descriptor (R.B. Rusu et al., 2010) es basado en las características extremadamente descriptivas del descriptor local FPFH pero adicionando variación al descriptor desde el punto de vista. Este nuevo descriptor extiende el descriptor local FPFH para ser estimado una sola vez para todo el clúster conteniendo en si información de la totalidad del elemento (vecindad absoluta) y computa estadísticas adicionales entre el vector V_c (entre el centroide y el punto de vista) trasladado a cada punto y las normales estimadas en cada punto. La característica de varianza ante el punto de vista no es particularmente útil para el trabajo, pues nuestros datos son obtenidos desde el mismo punto, sin embargo la consecución global de un único descriptor si es atractiva pues facilita la etapa de reconocimiento, por lo tanto el entrenamiento es realizado para elementos observados de la misma forma.

El proceso para la extracción de este descriptor es como sigue:

1. Encontrar el centroide de la nube de puntos p_c con el origen posicionado p_o .
2. Extraer el ángulo $\beta = \cos^{-1}(n_p \cdot (\frac{V_c + T}{\|V_c + T\|}))$ que el vector del punto de vista central trasladado a cada punto hace con la normal en cada punto, codificarlo en un histograma de 128 contenedores.
3. Para cada punto en la nube, construir un marco local (*Ecuación 3.15*).

$$\begin{cases} u = n_j \\ v = u \times \frac{(p_c - p_j)}{\|(p_c - p_j)\|_2} \\ w = u \times v \end{cases}$$

$$\begin{cases} \alpha = v \cdot n_c \\ \varphi = u \cdot \frac{(p_c - p_j)}{d} \\ \theta = \tan^{-1}(w \cdot n_c, u \cdot n_c) \end{cases} \quad \begin{array}{l} \text{Ecuación 3.16} \\ \text{y} \\ \text{Ecuación 3.17} \end{array}$$

4. Establecer los ángulos entre las dos normales (*Ecuación 3.17*).
5. Acumular el valor de cada ángulo en la codificación de un histograma de 45 contenedores.
6. Adicionar información sobre la distancia de cada punto y el centroide, y codificarlo en un histograma de 45 contenedores.
7. Concatenar los histogramas para la obtención de un histograma total de 308 contenedores.

Al ser un descriptor global, se obtiene 1 histograma en total por cada clúster a describir, en la **Figura 3.50**, se presentan ejemplos de los resultados obtenidos en este trabajo para un vehículo *hatchback*, así pues se presenta el macro histograma de 308 contenedores, donde se registra el componente de punto de vista y el componente extendido de FPFH. En este trabajo fue asignado un radio de búsqueda de entre 5 y 10 veces la resolución de la nube. Las razones que motivaron al uso de este descriptor son: Descriptor potentemente descriptivo como FPFH, pero con un único histograma por clúster, robusto ante cambio de densidad, independencia de etapas de agrupación de correspondencias, rapidez de obtención, comparación fácil, e información sobre el tamaño del objeto pues codifica la distancia al centroide de cada punto. A continuación se presenta el esquema básico de descriptor para cada clúster, seguido de un ejemplo de archivo PCD (formato nube de puntos y descriptores en PCL) con formato VFH.

Valor contenedor 1 del Angulo de Vc con n_k ... Valor contenedor 128 del Angulo de Vc con n_k %
contenedor 1a...% contenedor 45a % contenedor 1b...% contenedor 45b% contenedor 1c...%
contenedor 45c % contenedor 1d...% contenedor 45d



Figura 3.50 Ejemplo del descriptor VFH para un clúster de un vehículo *hatchback*, se observa la presencia de 308 contenedores en el descriptor VFH, 128 para el componente de punto de vista, 45x3 para los ángulos y 45 para las distancias al centroide.

3.5.7.2.2 ESF

Este descriptor (Wohlkinger & Vincze, 2011) es una integración de 10 subhistogramas de 64 contenedores cada uno, describiendo propiedades características de la nube de puntos. La función de forma D2 es creada mediante el muestreo de pares de puntos de la nube segmentada y formando un histograma de las distancias entre los dos puntos. Cada línea (distancia) es clasificada según si esta está dentro del modelo, afuera o una mezcla de los dos. Estas líneas son generadas con el algoritmo 3D de Bresenham en una grilla 3D voxel aproximada. Las líneas clasificadas como mixtas, son usadas para la realización de uno de los 10 histogramas, donde cada parte (adentro, fuera) es almacenada en contenedores diferentes. La función A3 codifica el Angulo entre dos líneas creadas, mediante el muestreo aleatorio de 3 puntos en la nube segmentada. La función D3 acumula en un subhistograma el área formada por 3 puntos aleatorios.

Al ser un descriptor global, se obtiene 1 histograma en total por cada clúster a describir, en la **Figura 3.51**, se presentan ejemplos de los resultados obtenidos en este trabajo para para un vehículo hatchback, así pues se presenta el macro histograma de 640 contenedores, donde se registra el cada una de las funciones de forma en 64x3 (adentro, fuera, mixto) subhistogramas y un histograma adicional de 64 contenedores, acumulando las tasas de los segmentos de recta mixtos de D2. Este tipo de descriptor no requiere las normales de superficie como parámetro de entrada.

Las razones que motivaron al uso de este descriptor son: Descriptor sencillo, descriptor supremamente eficaz y rápido pues no requiere las normales de superficie, con un único histograma por clúster, independencia de etapas de agrupación de correspondencias, comparación fácil, resultados satisfactorios en la bibliografía para bases de datos muy grandes. A continuación se presenta el esquema básico de descriptor para cada clúster, seguido de un ejemplo de archivo PCD (formato nube de puntos y descriptores en PCL) con formato ESF.

% contenedor 1D2in...% contenedor 64D2in % contenedor 1D2out...% contenedor 64D2out %
 contenedor 1D2mix...% contenedor 64D2mix % contenedor 1A3in...% contenedor 64A3in %
 contenedor 1A3out...% contenedor 64A3out % contenedor 1A3mix...% contenedor 64A3mix %
 contenedor 1D3in...% contenedor 64D3in % contenedor 1D3out...% contenedor 64D3out %
 contenedor 1D3mix...% contenedor 64D3mix % contenedor 1ratio...% contenedor 64ratio

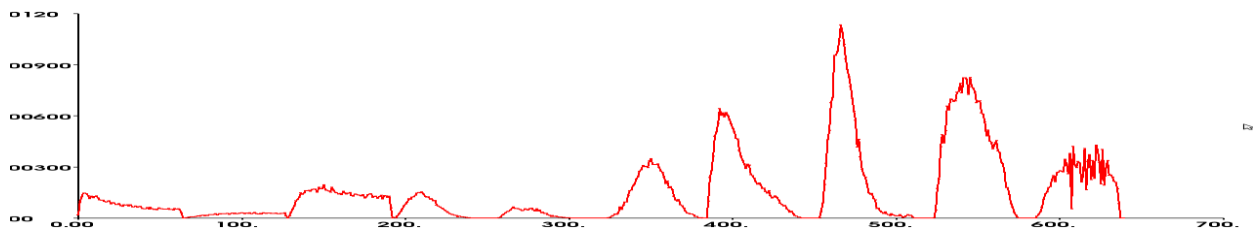


Figura 3.51 El histograma total del descriptor ESF calculado en un vehículo hatchback. Este descriptor contiene 10 subhistogramas de 64 contenedores de funciones de formas. En su orden de izquierda a derecha las funciones que aparecen en este histograma son: A3: Angle (in, out, mixed), D3: Area (in, out, mixed), D2:Distance (in, out, mixed) y la tasa de distancia de las líneas.

Algoritmo 3.6 Resumen del esquema de cálculo de descriptores 3D y clasificación

1.	Ndescriptores← {0.0}, Sdescriptores← {0.0};	// Crea estructuras para los descriptores
2.	Fdescriptores← {0.0}, Vdescriptor← {0.0};	
3.	Edescriptor← {0.0};	
4.	Pk← {pk1, ..., pkn};	// Define Nube de puntos claves
5.	Frec1← {0.0}, Frec2← {0.0}, Frec3← {0.0};	// Define la estructura de frecuencia
6.	Normal3DEstimation(Pk);	// Estima vectores normales
7.	Si (TipoDescriptor == "Local") Entonces	// ¿Local o global?
8.	Ndescriptores←ObtieneNARF(Pk);	// Calcula descriptores
9.	Sdescriptores←ObtieneSHOT(Pk);	
10.	Fdescriptores←ObtieneFPFH(Pk);	
11.	Frec1←BOW(Ndescriptors);	// Agrupa con bolsa de características
12.	Frec2←BOW(Sdescriptors);	
13.	Frec3←BOW(Fdescriptors);	
14.	BuscaKvecinos(Frec1);.. BuscaKvecinos(Frec3).	// Clasifica basado en distancias
15.	Sino Si (TipoDescriptor == "Global") Entonces	// No paralelos
16.	Vdescriptor←ObtieneVFH(Pk);	// Calcula descriptores

17.	<i>Edescriptor←ObtieneESF(Pk);</i>	
18.	<i>BuscaKvecinos(Vdescriptor);</i>	<i>//Clasifica basado en distancias</i>
19.	<i>BuscaKvecinos(Edescriptor);</i>	
20-	<i>Fin si</i>	

3.5.7.3 Agrupación de descriptores locales

Los descriptores locales tienen como principal ventaja su independencia de los procesos previos como la segmentación y la clusterización, pues al trabajar al nivel de cada punto y su vecindad, es posible la extracción de pares de correspondencias (relaciones de descriptores modelo-escena dada su similitud), que cumplan con las características de un elemento en particular; sin embargo la etapa de agrupación de correspondencias no es trivial, pues diferentes elementos pueden tener regiones similares. Para afrontar el problema de la unión, integración o agrupación de correspondencias existen principalmente tres enfoques, en primer lugar el más simple de ellos denominado *geometric constraints (condicionamiento geométrico)* (H. Chen & Bhanu, 2007), agrupa pares de correspondencias que cumplan ciertas restricciones o condiciones espaciales geométricas entre ellos. En segundo lugar uno de los más elaborados es denominado *Hough 3D* (F. Tombari & Stefano, 2010), este es basado en una extensión del algoritmo de votación de Hough para elementos 3D, acumulando en un marco de votación las transformaciones entre pares de correspondencias modelo-escena con respecto al centroide del modelo y de la escena. Finalmente el enfoque más robusto es denominado *Bag of Words* o más genéricamente *Bag of Features* (Sivic & Zisserman, 2003), este método primero realiza la construcción de un diccionario de características, realizando una clusterización mediante *k-means* de todos los descriptores multidimensionales de la base de datos, y posteriormente elaborando vectores de frecuencias para cada elemento.

En este trabajo el método de bolsa de características es implementado mediante el uso de dos librerías diferentes *Caltech Large Scale Image Search Toolbox* (Aly et al., 2011; Perona, 2010) y *PCL* (Radu Bogdan Rusu & Cousins, 2011), explícitamente la clase de *Implicit Shape Model*, que es un método muy similar a *Hough 3D* en su esquema de votación, pero que basa las ponderaciones de los votos en diccionarios visuales y palabras visuales. El esquema de trabajo para la integración de descriptores locales en *toolbox* de Caltech se muestra en la **Figura 3.52**

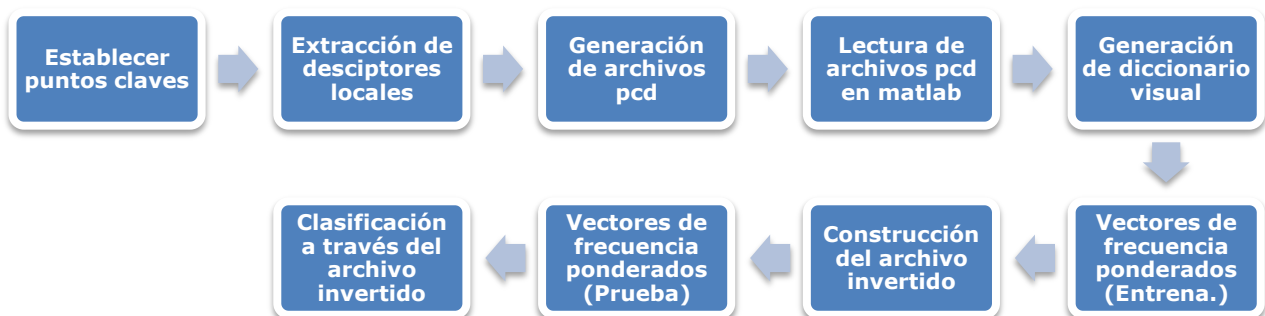


Figura 3.52 Esquema de trabajo para bolsa de características en MATLAB.

Siguiendo el esquema de la **Figura 3.52** primero se determina el número de características por elemento dado que si se trabaja con modalidad constante, es necesario que se limiten el número de descriptores por cada clúster resultante de las etapas anteriores de segmentación clusterización; la extracción de puntos clave por muestreo uniforme es basada en la densidad de cada nube, por lo que el mínimo número de puntos clave está relacionado con la nube menos densa, una vez se obtiene este valor, se aplica una voxelización a las nubes restantes y se obtiene el centroide de los elementos en los voxels hasta lograr un

número igual de puntos clave en todas las nubes. Puede trabajarse con diferentes número de características, si se desea, esto añade la ventaja de elegir más puntos clave en cada clase lo que aumenta la calidad de la descripción. Inmediatamente se tienen el número de puntos clave, se procede con la extracción de los descriptores locales y la generación de archivos con el formato PCD en PCL con la información de cada descriptor, (la longitud del histograma de cada descriptor es diferente, igual que su interpretación). En este trabajo se implementó un módulo software que permite la lectura escritura de este tipo de archivo propio de PCL mediante el software MATLAB. Después de que cada descriptor es leído y almacenado en MATLAB, cada característica es etiquetada con información sobre el clúster de donde proviene y su tipo de descriptor. Se procede entonces con la elaboración de 3 diccionarios visuales diferentes, uno para cada tipo de descriptor, y con la elaboración de los vectores de frecuencia ponderados mediante *tf-idf* (véase (Sivic & Zisserman, 2003) y la sección 2.2.4.1).

Un aspecto importante en el diseño de un sistema de bolsa de características, es que el proceso de retribución o clasificación de elementos sea eficaz y rápido, por lo que en el trabajo original de esta técnica se propone la elaboración de un *inverted-file* que es análogo a una especie de índice de un libro, en este caso estableciendo los elementos presentes (clústeres de los diferentes vehículos) en los cuales aparece alguna instancia de cada palabra visual (esto es un descriptor que es agrupado con un clúster específico), así pues siguiendo el esquema de trabajo para este *toolbox*, se procede con la creación de un archivo invertido índice de palabras visuales para lograr una retribución rápida en la etapa de clasificación. Finalmente para cada elemento a clasificar, esto es el 30-50% del dataset de pruebas, son obtenidos los vectores de frecuencias de palabras y se realiza la búsqueda mediante el archivo invertido, lográndose así la clasificación automática de vehículos.

La segunda implementación de bolsa de características es realizada directamente en el lenguaje C++, en la librería *PCL*, estrictamente no hay una implementación exacta de esta técnica de este tipo de herramienta, sin embargo existe una técnica denominada *modelo implícito de forma* (Chang, Leymarie, & Kimia, 2009)(Knopp et al., 2010; A Velizhev, 2012), que utiliza el concepto de diccionario visual, para una posterior ponderación en un algoritmo de votación, basado en las instancias, de palabras visuales en el diccionario. Así pues es posible extraer la sección de la construcción del diccionario visual y adaptarla a las necesidades del trabajo. En la **Figura 3.53** es presentado el esquema de trabajo para la implementación de este enfoque.

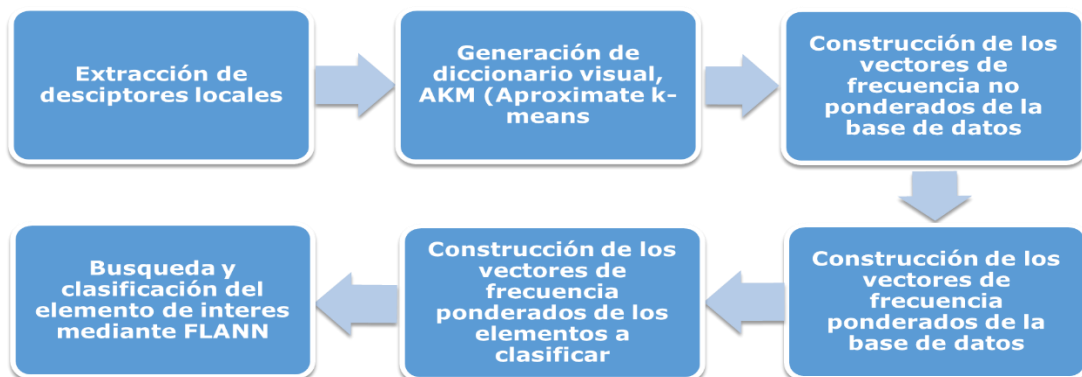


Figura 3.53 Esquema de trabajo para bolsa de características en PCL

Existen varias diferencias en el proceso, en primer lugar es necesaria la construcción manual de los vectores de frecuencias de palabras, y es necesario un paso más para la obtención de la ponderación *tf-idf*, la búsqueda y clasificación no hace uso de los archivos invertidos, por lo contrario hace uso de la librería FLANN (Fast Library Approximate Nearest Neighbors).



4 PRUEBAS Y ANÁLISIS DE RESULTADOS

4.1 INTRODUCCIÓN

Este capítulo busca describir la elaboración del esquema de pruebas para la validación del desempeño de cada módulo del trabajo, lo que es presentado en la *sección 4.2*. A pesar de que el sistema es capaz de realizar clasificaciones individuales se plantean aquí esquemas de evaluación globales y únicos para toda la información disponible. Además, se consignan en este capítulo los resultados obtenidos bajo cada esquema y se realiza un análisis de las características de respuesta del sistema diseñado. Esto último es expuesto en la *sección 4.3*

4.2 ESQUEMA, DESCRIPCIÓN Y DESARROLLO DE LAS PRUEBAS

Dadas las características multimodales de este sistema, se realiza una selección amplia de categorías de vehículos, esto es posible debido a que la multimodularidad brinda gran robustez y un mayor potencial en clasificaciones intraclase, a diferencia de sistemas más sencillos. Estos últimos solo permiten la diferenciación entre clases diferentes de elementos o, elementos de clases iguales pero muy bien diferenciados. La subdivisión de macro categorías en categorías más pequeñas es también motivada por la presencia de vehículos muy similares en una entrada universitaria, a diferencia de la variedad presente en carreteras importantes. En este trabajo se trabaja con 9 clases para el módulo de información 3D y 10 para el módulo de información de imagen 2D. En trabajos con clasificadores más sencillos esta gran cantidad de clases es agrupada en tan solo 2, 3 o máximo 4 clases diferentes. (Valverde Pedraza & Ossa Torres, 2011). A continuación son presentados todos los supuestos y condiciones, y son explicados los esquemas de desarrollo e implementación para la evaluación de los resultados del sistema.

4.2.1 MÓDULO CLASIFICACIÓN DE VEHÍCULOS CON INFORMACIÓN VISUAL

Gracias a la adecuada separación de tareas del módulo, cuando se aborda la etapa de realización de pruebas con el sistema ya se dispone de toda la información requerida para la realización de una o varias clasificaciones, pudiéndose así automatizar el proceso para todos los datos pertenecientes al *dataset* de validación y generar automáticamente las matrices de confusión que permiten calificar su desempeño. Esta sección da una descripción del esquema de pruebas llevado a cabo para evaluar el desempeño del módulo de clasificación de vehículos por medio de información visual.

Existen múltiples parámetros libres que permiten modificar de algún modo el desempeño del sistema de clasificación y penetran distintos niveles en el proceso. Los parámetros libres a

los que el programador tiene acceso, que se detallan en la **Tabla 4.1**, tienen una gran influencia en los porcentajes de clasificación que presenta el sistema.

Tabla 4.1 Posibles parámetros libres a considerar en el diseño del módulo de clasificación mediante información de intensidad.

Etapa	Parámetro	Posibilidades
Segmentación, extracción del <i>foreground</i> y formación de <i>blobs</i> en un fotograma de video	Tamaño del elemento estructurante de <i>dilatación</i>	Números naturales $\mathbb{N}$
	Tamaño del elemento estructurante de <i>cierre</i>	Números naturales $\mathbb{N}$
Obtención de puntos clave en la región de interés y formación de descriptores visuales	Técnica para la detección de puntos clave	SIFT
		SURF
		FAST
		BRISK
		ORB
	Umbral para el algoritmo de detección de puntos clave (Tiene distinto significado, de acuerdo a la técnica, e.g. En FAST denota la diferencia en intensidad entre pixels, en SURF se aplica a la hessiana de una imagen)	SIFT: no disponible
		SURF: Números reales $\mathbb{R}$
		FAST: Números reales $\mathbb{R}$
		BRISK: no disponible
		ORB: Números naturales $\mathbb{N}$
	Técnica para el cálculo de descriptores visuales	SIFT
		SURF
		BRIEF
		BRISK
		ORB
	Tamaño de los descriptores visuales	FREAK
		Solo disponible en BRIEF: 16, 32 y 64 bytes.
Formación del diccionario de características visuales	Número de palabras (clústeres)	Números naturales $\mathbb{N}$
	Tipo de diccionario	ak-means (Approximate k-means) hk-means (Hierarchical k-means)
	Número de árboles en el tipo de diccionario ak-means	Números naturales $\mathbb{N}$
Formación del vector de características	Método de ponderación de las palabras en el documento (imagen)	Sin ponderación ("none")
		Binarización ("bin")
		Frecuencia del término ("tf")
		tf-idf
	Método de normalización del vector de frecuencias	Normalización con el número de palabras ("l0")
		Suma del histograma = 1 ("l1")
		Suma de cuadrados del histograma = 1 ("l2")
Búsqueda de k-vecinos más cercanos	Tipo de distancia usada	Distancia $L1$ ("l1")
		Distancia $L2$ ("l2")
		Distancia <i>Hamming</i> ("ham")
		Producto punto ("cos")
		Distancia <i>Jacquard</i> ("jac")

Debido a la gran cantidad de parámetros libres y, por consiguiente, la inmensa cantidad de combinaciones posibles, se decidió afectar solo aquellos parámetros de más alto nivel en el proceso que afectarían más notablemente el comportamiento de la clasificación y, por consiguiente, el desempeño del sistema. Los parámetros elegidos fueron los siguientes:

- Técnica para la detección de puntos clave.
- Umbral para el algoritmo de detección de puntos clave.
- Técnica para el cálculo de los descriptores visuales.
- Número de palabras en el diccionario.
- Número de árboles en el diccionario ak-means.
- Método de ponderación de las palabras en el documento.

- Método de normalización del vector.
- Tipo de distancia usada para hallar los k-vecinos más cercanos.

Las pruebas se realizaron usando todos los tipos de descriptores visuales considerados en la revisión bibliográfica, siguiendo, para cada uno, un procedimiento sistemático que se detalla en el **Algoritmo 4.1**. Para cada descriptor, se obtuvieron los puntos clave con las técnicas y los umbrales que se detallan en la **Tabla 4.2**.

Tabla 4.2 Técnicas de obtención de puntos clave y umbral de detección

Descriptor	Obtención de puntos clave	Umbral
SIFT	SIFT	-----
SURF	SURF	0
		20
		2500
		4-5
		6-6
BRIEF	FAST	20
		30
		40
BRISK	BRISK	-----
ORB	ORB	20
		200
		500
FREAK	FAST	20
		30
		40

Algoritmo 4.1 Procedimiento seguido para la evaluación del desempeño del sistema ante variaciones en los parámetros libres.

1.	Procedimiento elegir_umbral(umbral)	// Elegir umbral para el algoritmo de detección de puntos clave
2.	mejor(desempeño(umbral));	//Se evalúan los umbrales disponibles para el descriptor en cuestión
3.	Elegir umbral;	// Se elige el umbral con mejor desempeño
4.	Fin Procedimiento	
5.	Procedimiento elegir_diccionario(numero)	// Elegir número de palabras en el diccionario
6.	numero ₀ ← 10% del # de descriptores en dataset de entrenamiento;	// Se inicia con 10% del número total de descriptores en el dataset de entrenamiento
7.	mejor ← desempeño(numero ₀);	// Se evalúa el desempeño inicial
8.	numero ← centena_anterior(numero ₀);	// Se decrementa a la centena anterior
9.	aux ← desempeño(numero);	// Se evalúa el desempeño
10.	Mientras aux ≥ mejor	// Si el desempeño es mejor
11.	mejor ← aux;	
12.	numero ← numero – 100;	// Se sigue decrementando por centenas
13.	aux ← desempeño(numero);	// Hasta que el aumento en desempeño no aumente más
14.	Fin Mientras	
15.	numero ← centena_posterior(numero ₀);	// Se incrementa a la centena posterior
16.	aux ← desempeño(numero);	// Se evalúa el desempeño
17.	Mientras aux ≥ mejor	// Si el desempeño es mejor
18.	mejor ← aux;	
19.	numero ← numero + 100;	// Se sigue incrementando por centenas
20.	aux ← desempeño(numero);	// Hasta que el aumento en desempeño no aumente más
21.	Fin Mientras	
22.	Elegir numero;	// Se elige el número de palabras de mejor desempeño
23.	Fin Procedimiento	
24.	Procedimiento elegir_arboles(numero)	// Elegir número de árboles en el diccionario ak-means.
25.	mejor(desempeño(numero));	// Se evalúa el desempeño con número de árboles

		<i>desde 1 hasta 10. numero = [1,...,10]</i>
26.	<i>Elegir numero;</i>	<i>// Se elige el número de árboles de mejor desempeño</i>
27.	<i>Fin Procedimiento</i>	
28.	<i>Procedimiento elegir_ponderacion(ponderacion)</i>	<i>// Elegir el método de ponderación de las palabra</i>
29.	<i>mejor(desempeño(ponderacion));</i>	<i>// ponderacion = [none, bin, tf, tfidf]</i>
30.	<i>Elegir ponderacion;</i>	<i>// Se elige el método de ponderación de mejor desempeño</i>
31.	<i>Fin Procedimiento</i>	
32.	<i>Procedimiento elegir_normalizacion(normalizacion)</i>	<i>// Elegir el método de normalización del vector de frecuencias</i>
33.	<i>mejor(desempeño(normalizacion));</i>	<i>// normalizacion = [l1, l0, l2]</i>
34.	<i>Elegir normalizacion;</i>	<i>// Se elige el método de normalización de mejor desempeño</i>
35.	<i>Fin Procedimiento</i>	
36.	<i>Procedimiento elegir_distancia(distancia)</i>	<i>// Elegir la distancia para búsqueda de los k-vecinos</i>
37.	<i>mejor(desempeño(distancia));</i>	<i>// distancia = [l1, l2, ham. cos. jac]</i>
38.	<i>Elegir distancia;</i>	<i>// Se elige la distancia de mejor desempeño</i>
39.	<i>Fin Procedimiento</i>	

4.2.2 MODULO DE INFORMACIÓN DE RANGO

Para evaluar el desempeño del sistema específicamente en el módulo basado en información 3D de rango, se realizó la selección previa de un set más pequeño a partir del *dataset* original, denominado Dataset de pruebas, esto se decidió debido a dos razones principales, como siguen:

- Reducir la carga computacional en los algoritmos de entrenamiento y búsqueda, para poder realizar gran diversidad de pruebas y la variación de los distintos parámetros del sistema en tiempos cortos (lo que llevaría demasiado tiempo para el Dataset completo de más de 500 datos), observando los cambios en los resultados y las afectaciones locales de cada variación más fácilmente.
- Evaluar similares cantidades de vehículos pertenecientes a cada clase, pues no todas las categorías poseen cantidades parecidas de datos, lo que contribuiría al sobre entrenamiento de ciertas clases, lo que podría incidir en las probabilidades de obtención de resultados acertados.

Las clases poseen un número de instancias que oscila entre 5(buses, escasos en la entrada a la universidad) y 150 (Hatchback, la clase más abundante en la población automotriz de la universidad) datos, sin embargo la mayoría registra números entre 20 y 30 datos, por lo que por fines de evaluación se decidió el uso de 25 datos en total por clase, siendo menor en casos como furgonetas(13) y buses(5), dada su escases, y mayor en algunos como camiones(31), camperos(29), pickup(26), dada la importancia de ciertas clases.

El módulo de trabajo con información de rango 3D, ha sido descompuesto en 5 sub módulos más pequeños lo que permitió, la validación y verificación de labores más puntuales y explícitas, a diferencia de una gran cadena de instrucciones de difícil depuración. Estos sub módulos son:

1. Selección de carpetas de información, consolidación de las nubes de puntos, segmentación, clusterización y etiquetado de instancias de clústeres.
2. Estimación de las normales de superficie, estimación de los descriptores globales VFH y ESF, junto con los locales NARF, SHOT y FPFH, separación entre datos de entrenamiento y datos de validación.

3. Creación de árboles k-dimensionales, índices y matrices FLANN de entrenamiento.
4. Proceso de clasificación y validación global.
5. Proceso de clasificación y validación local.

A continuación se describen cada uno de estos sub módulos y se presenta un diagrama de bloques que resume el proceso y las características de su implementación.

Sub Módulo de extracción y etiquetado de clústeres de vehículos

El objetivo de esta etapa es la construcción de clústeres plenamente etiquetados de instancias de vehículos de cada clase; en primer lugar se procede con la selección aleatoria o verificada de un número determinado de carpetas pertenecientes a cada categoría en el Dataset de pruebas, idealmente este valor es igual para cada clase, pero dada la asimetría en el número de datos por clase en este trabajo, este número es variable, pero estandarizado en 25. Se continua con la elaboración consistente o registrado de las nubes de puntos 3D mediante el acople de planos 2D consecutivos con la información de velocidad alisada de kalman de los vehículos atravesando el láser, esta nube es transformada de matrices estándar a la clase de trabajo de PCL. El siguiente paso es un proceso consecutivo de segmentación clusterización, que utiliza eliminación de componente planar principal, filtro de distancias y clusterización euclidiana. Finalmente, para todos los clústeres extraídos de cada nube de puntos, se realiza un proceso de etiquetado automático (basado en la carpeta contenedora) y manual (revisión de clústeres de diferentes clases en una carpeta de una clase determinada) de la clase a la que pertenece el dato, para permitir el proceso de validación clasificación automática más adelante. La última parte adiciona a cada dato un sufijo determinado por la clase a la que pertenece, como sigue en la **Tabla 4.3**.

Tabla 4.3 Etiquetado de clase y de número para los clústeres extraídos

CLASES	<i>a->bicicleta; b->bus; c->camión; d->campero; e->furgoneta; f->hatchback; g->malos; h->moto; i->moto utilitaria; j->pickup; k->sedan; l->station wagon; m->suv; n->volkswagenbeatle</i>
NÚMERO CLUSTER POR NUBE	<i>s->1er cluster; x->2do cluster; y->3er cluster; z->4to cluster; w->5to cluster</i>

Aunque en la tabla anterior aparecen 13 categorías, 3 son fusionadas en una sola (hatchback, station, wagon, volkswagen beetle), 2 más también para el módulo de rango 3D (motos, bicicletas) dadas las extremas similitudes intraclase entre ellas y una descartada (malos). El segundo campo indica el orden de aparición de un clúster en una nube capturada por toma, es importante aclarar que en el dataset no existen en ningún caso más de 5 clústeres correctos por toma. En el **Algoritmo 4.2** se presenta el pseudocódigo de este sub módulo.

Algoritmo 4.2 Algoritmo del sub módulo 1 del módulo de clasificación basado en información de rango 3D.

1.	$[I_t, N_T, Data] \leftarrow TomaData(Ruta); escaneos \leftarrow \{0 \dots 0\};$ $Velocidades \leftarrow \{0 \dots 0\}; Clusters \leftarrow \{0 \dots 0\};$ $n \leftarrow 0;$	<i>// Obtiene la información del Dataset de pruebas, e indica el número de clases N_T y el número por clase I_t. Iniciación</i>
2.	Mientras $n < N_T$ Hacer	<i>//para cada clase</i>
3.	$n++; i \leftarrow 0;$	<i>// Inicialización</i>
4.	Mientras $i < I_t$ Hacer	<i>// Para cada dato de la clase</i>
5.	$i++;$	
6.	$escaneos \leftarrow TomaRango(Data);$	<i>// Lectura escaneos data[i].dat,</i>
7.	$Velocidades \leftarrow TomaVel(Data);$	<i>// Lectura velocidad data[i]velprom.dat</i>
8.	$Registra3D(escaneos, Velocidades); xyz2Pcl();$	
9.	$FiltraDistancias("x", 0, 5);$	<i>// FiltraDistancias(dim, dimmin, dimmax);</i>

10.	<i>FiltraDistancias("y",0,5);</i>	
11.	<i>EstimaPlano("Ransac",30,"Y",300,0.05);</i>	<i>// EstimaPlano(método, desviación</i> <i>,ejenormal, iteraciones,</i> <i>distanciaumbral);</i>
12.	<i>[Clusters,numc]←ClusterEu(0.5,100,500k,kd1);</i>	<i>ClusterEuclidiana(distanciaumbral,</i> <i>puntosmin,puntosmax,arbolkd);</i>
13.	<i>cindex ← 0;</i>	
14.	<i>Mientras cindex < numc Hacer</i>	<i>// para cada clúster</i>
15.	<i> cindex ++;</i>	
16.	<i> Según clase(Clusters[cindex]) Hacer</i>	<i>// Obtiene clase según carpeta y</i> <i>evalúa</i>
17.	<i> Caso clase 1</i>	
18.	<i> Asignaclase1();</i>	<i>// Asigna clase a cada cluster</i>
	<i> .</i>	
	<i> .</i>	
	<i> .</i>	
19.	<i> Caso clase n</i>	
20.	<i> Asignaclase n();</i>	
21.	<i> Fin Según</i>	
22.	<i> Fin Mientras</i>	
23.	<i>Fin Mientras</i>	
24.	<i>Fin Mientras</i>	

Sub Módulo de obtención de normales de superficie y descriptores 3D

El paso más importante para la exitosa consecución de tasa altas de clasificación, corresponde a la etapa de extracción de descriptores 3D y geométricos que ofrezcan una descripción discriminante de cada una de las clases a separar o a reconocer mediante el desempeño del sistema. En este trabajo se optó con fines de comparación y evaluación con la extracción de 5 descriptores 3D (Descritos previamente) y 2 descriptores de altura y área adicionales presentados en la **Figura 4.1**, que son evaluados en un esquema de cascada y para ciertas clases que lo ameriten, con la finalidad de una discriminación intracase aún mejor (Discriminación Hatchbacks-SUV por altura, y Hatchbacks-sedan por área lateral derecha); como se observa estos descriptores umbrales están basados en centroides de los datos del plano (y, z), con la finalidad de obtener robustez ante el ruido.

El procedimiento de extracción de descriptores 3D consiste en el cálculo de la densidad espacial de la nube de puntos, la extracción de las normales de superficie, y el ajuste de cada uno de los parámetros para cada descriptor. Finalmente se computan los descriptores geométricos adicionales y se etiquetan los clústeres según el resultado de los mismos. Todo el proceso de este módulo es presentado en el **Algoritmo 4.3**.



Figura 4.1 Descriptores geométricos adicionales.

Algoritmo 4.3 Algoritmo para la extracción de los 7 descriptores totales, 5 3D (globales y locales) y 2 en dos dimensiones (globales).

1.	<i>[I_t, N_T, Data] ← TomaData(Ruta);</i> <i>n ← 0;</i> <i>Quaternion ← (−0.1379, −0.6935, 0.1379, −0.6935)</i>	<i>// Obtiene la información de</i> <i>los clústeres, e indica el</i> <i>número de clases N_T y el</i> <i>número por clase I_t.</i>
----	---	--

		Inicialización
2.	Mientras $n < N_T$ Hacer	//para cada clase
3.	$n++$; $i \leftarrow 0$;	// Inicialización e incremento
4.	Mientras $i < I_T$ Hacer	// Para cada dato de la clase
5.	$i++$;	// Incremento
6.	$Clusters \leftarrow \{0 \dots 0\}$;	// Reinicia
7.	$Ndescriptores \leftarrow \{0..0\}$, $Sdescriptores \leftarrow \{0..0\}$;	// Crea estructuras para los descriptores
8.	$Fdescriptores \leftarrow \{0..0\}$, $Vdescriptor \leftarrow \{0..0\}$;	
9.	$Edescriptor \leftarrow \{0..0\}$;	
10.	$Clusters \leftarrow TomaCluster(Data)$;	
11.	$P \leftarrow Clusters$; $Nor \leftarrow Normal3DEstimation(Pk)$;	// Estima normal e inicializa nube
12.	$Pk \leftarrow ObtieneClaves(Clusters)$; $Bus \leftarrow Kdtree1$;	// Define Parametros
13.	$Ra \leftarrow ObtieneResolucion(P)*10$; $vis \leftarrow (-5,0,0)$;	
14.	Si (TipoDescriptor == "Local") Entonces	// ¿Local o global?
15.	$Sdescriptores \leftarrow ObtieneSHOT(P,Pk,Nor,Bus,RA)$;	//Obtiene cada descriptor
16.	$Fdescriptores \leftarrow ObtieneFPFH(P,Pk,Nor,Bus,RA)$;	
17.	$T \leftarrow Traslación(Clusters)$;	// ajustes y preprocesamiento NARF
18.	$Aj \leftarrow AjustePuntodeVista(Quaternion)$;	
19.	$Ir \leftarrow ImgRango(Clusters)$;	
20.	$B \leftarrow DetectaBorde()$;	
21.	$Pn \leftarrow PuntosClaveBorde()$;	
22.	$S=0.4$;	
23.	$R=0.6$;	
24.	$V=0.35$;	
25.	$Ndes \leftarrow ONARF(S,R,V,T,Aj,Ir,B,Pn)$;	
26.	Sino Si (TipoDescriptor == "Global") Entonces	
27.	$Vdescriptor \leftarrow ObtieneVFH((P,Pk,Nor,Bus,RA,vis))$;	
28.	$Edescriptor \leftarrow ObtieneESF(P,Bus)$;	
29.	Fin si	
30.	$AgregaAltura(clusters)$; $AgregaArea(clusters)$;	//Agrega descripción adicional
31.	$TestOrTrain()$;	// ¿ Entrenamiento o prueba?
32.	Fin Mientras	
33.	Fin Mientras	

Sub Módulo de construcción de la base de datos y realización del entrenamiento del sistema

Algoritmo 4.4 Algoritmo para la creación de árboles, matrices y listas de información para el conjunto de datos de entrenamiento.

1.	[Data] $\leftarrow TomaData(Ruta)$;	// Lectura de la data en la ruta de entrenamiento Global.
2.	Modelos $\leftarrow CargaModelos(Data)$;	// Carga los modelos ESF y VFH
3.	Maflann $\leftarrow CreaMatrizFLANN(Modelos)$;	// Crea matriz con formato FLANN
4.	GuardaMatrizFLANN(Maflann);	// Guarda en sistema
5.	Kdind $\leftarrow CreaIndiceArbolKd(Modelos)$;	// Crea Índice para el árbol Kd.
6.	GuardaIndiceArbolKd (Kdind);	// Guarda en sistema

Inmediatamente se tienen todos los descriptores locales y globales para cada clúster, es necesaria la construcción de un esquema organizado de toda esta información, lo cual posibilite la obtención rápida de los resultados de comparación entre los elementos de testeo y los presentes en la base de datos (esta etapa es desarrollada con fines de implementación práctica, si bien no es mencionada en el desarrollo de ingeniería, es importante para la consecución del sistema automático de clasificación), por lo tanto para los descriptores globales es elaborada una matriz FLANN de entrenamiento, un índice basado en árboles k-

dimensionales y una lista con los nombres de cada dato de entrenamiento. En el **Algoritmo 4.4** se presenta el pseudocódigo de este proceso. Para los locales, el sistema almacena en un archivo invertido los vectores de frecuencia de cada dato de entrenamiento (**Algoritmo 4.5**);

Algoritmo 4.5 Algoritmo para el entrenamiento local.

1.	[Data] ← TomaData(Ruta);	// Lectura de la data en la ruta de entrenamiento Local.
2.	Modelos ← CargaModelos(Data);	// Carga los modelos NARF, SHOT y FPFH
3.	dictio ← CreaDiccionario(Modelos);	// Crea Diccionario
4.	palabras ← computaPalabras(dictio);	// Crea Palabras
5.	invf ← CreaArchivoInvertido(palabras);	// Crea Archivo invertido

Sub Módulo de clasificación y validación Local

Esta sección consiste en la clasificación global para los descriptores de este tipo, simplemente mediante la búsqueda de los k-vecinos más cercanos multidimensionales, debido a que cada clúster queda definido por un único vector general, esta búsqueda entrega los primeros k vecinos, donde k es un parámetro definido por el usuario, y en este trabajo en particular es 10; la lista obtenida es reclasificada u organizada según los resultados de los descriptores geométricos adicionales, lo que aumenta la robustez del sistema. En el caso de que ninguno de los 10 resultados retribuidos cumpla con los criterios adicionales de su clase, simplemente se retorna el más cercano. Los valores de umbral definidos por los descriptores adicionales, varían según las características observadas en los datos y si algunas clases presentan conflictos debido a su ambigüedad y similitud. El pseudocódigo de módulo es presentado en el **Algoritmo 4.7**.

Sub Módulo de clasificación y validación global

Esta sección consiste en la clasificación local para los descriptores de este tipo, integrando todos los descriptores locales, mediante bolsas de características, las cuales entregan a su salida vectores ponderados de frecuencias según el esquema *tfidf* (Sivic & Zisserman, 2003) y retribuyen las respuestas a través de archivos invertidos de frecuencia que cumplen la labor de índices de búsqueda. La anterior búsqueda entrega los primeros k vecinos, donde k es un parámetro definido por el usuario, y en este trabajo en particular es 5; la lista obtenida es reclasificada u organizada según los resultados de los descriptores geométricos adicionales, lo que aumenta la robustez del sistema. En el caso de que ninguno de los 5 resultados retribuidos cumpla con los criterios adicionales de su clase, simplemente se retorna el más cercano. Los valores de umbral definidos por los descriptores adicionales, varían según las características observadas en los datos y si algunas clases presentan conflictos debido a su ambigüedad y similitud. El pseudocódigo del módulo es presentado en el algoritmo **Algoritmo 4.6**.

Algoritmo 4.6 Módulo de clasificación Global

1.	$N_T \leftarrow \text{NumTests}$; clase ← 0; $K \leftarrow 10$; $n \leftarrow 0$;	// Obtiene la información del Dataset de evaluación y la base de datos
		// Inicialización
2.	desT1 ← LeeTestSet("ESF"); desT2 ← LeeTestSet("VFH");	// Lectura de los descriptors de evaluación.
3.	desB1 ← LeeFLANNdata("ESF"); desB2 ← LeeFLANNdata("VFH");	// Lectura de la base de datos FLANN.
4.	Resultado ← {0..0}; clasificado ← {0..0};	
5.	Mientras $n < N_T$ Hacer	// para cada dato de evaluación
6.	$n++$; $k \leftarrow 0$;	// Inicialización
7.	[Resultado, clase] ← BuscaKvecinos(desT1, desB1);	// Buscar los 10 más cercanos en la base de datos
8.	[Resultado, clase] ← BuscaKvecinos(desT2, desB2);	

9.	Mientras $k < K$ Hacer	// Para cada uno de los 10
10.	$k++$;	
11.	Si (CompruebaGeométrica()) Entonces	// Realice comprobaciones adicionales con los descriptores geométricos extras
12.	$Clase \leftarrow ExtraeclaseNum(Resultado[k])$	
13.	Break;	// Es ok, este es el resultado más apropiado, salir
14.	Fin Si	
15.	$Clase \leftarrow ExtraeclaseNum (Resultado[0]);$	// Si termina (ninguno cumplió) tome el primero.
16.	Fin Mientras	
17.	$clasificado \leftarrow Resultado(Clase);$	// Entregue el resultado
18.	Fin Mientras	

1.	$N_T \leftarrow NumTests; clase \leftarrow 0; K \leftarrow 10; n \leftarrow 0;$	// Obtiene la información del Dataset de evaluación y la base de datos // Inicialización
2.	$desT1 \leftarrow LeeTestSet("NARF"); desT2 \leftarrow LeeTestSet("SHOT");$	//Lectura de los descriptors de evaluación.
3.	$desT3 \leftarrow LeeTestSet("FPFH");$	
4.	$Resultado \leftarrow \{0..0\}; clasificado \leftarrow \{0..0\};$	
5.	Mientras $n < N_T$ Hacer	//para cada dato de evaluación
6.	$n++; k \leftarrow 0;$	// Inicialización
7.	$[Resultado, clase] \leftarrow BuscaEnIF(desT1);$	// Buscar los 10 más cercanos gracias a los vectores de frecuencia codificados en el archivo invertido
8.	$[Resultado, clase] \leftarrow BuscaEnIF(desT2);$	
	$[Resultado, clase] \leftarrow BuscaEnIF(desT3);$	
9.	Mientras $k < K$ Hacer	// Para cada uno de los 10
10.	$k++$;	
11.	Si (CompruebaGeométrica()) Entonces	// Realice comprobaciones adicionales con los descriptores geométricos extras
12.	$Clase \leftarrow ExtraeclaseNum(Resultado[k])$	
13.	Break;	// Es ok, este es el resultado más apropiado, salir
14.	Fin Si	
15.	$Clase \leftarrow ExtraeclaseNum (Resultado[0]);$	// Si termina (ninguno cumplió) tome el primero.
16.	Fin Mientras	
17.	$clasificado \leftarrow Resultado(Clase);$	// Entregue el resultado
18.	Fin Mientras	

Algoritmo 4.7 Módulo de clasificación Local.

4.3 RESULTADOS Y ANÁLISIS

4.3.1 MÓDULO CLASIFICACIÓN DE VEHÍCULOS CON INFORMACIÓN VISUAL

Para cada uno de los descriptores considerados se construyó un árbol de resultados, que muestra la evolución de las pruebas a medida que se evalúan los distintos parámetros libres posibles. Estos árboles, por no tratarse de resultados de clasificación directos, son mostrados en la sección de anexos. También se obtienen las matrices de confusión que corresponden a la combinación de parámetros que mejor desempeño mostró para cada descriptor.

En las matrices de confusión, mostradas en las **Tabla 4.4** a **Tabla 4.9**, las filas representan las clases objetivo, y las columnas representan las salidas del sistema. En la primera columna (clases objetivo) se muestra la clase y entre paréntesis el número total de datos de esa clase contenidos en la matriz de validación. En la celda (i, j) se muestra el número de instancias de la clase i que fueron clasificadas como clase j . También se muestra, en

formato de porcentaje, la relación entre el total de instancias clasificadas como clase j y el total de datos del *ground truth* pertenecientes a la clase i . Los elementos de la diagonal, resaltados en color verde, corresponden a las instancias de los datos de validación que fueron correctamente clasificadas.

Tabla 4.4 Matriz de confusión del sistema con descriptores SIFT de mayor desempeño probado.

	Bicicleta	Bus	Camión	Campero	Furgoneta	Hatchback	Moto	Pickup	Sedan	SUV
Bicicleta (13)	12 92.31%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 7.69%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (14)	1 7.14%	0 0%	9 64.29%	1 7.14%	0 0%	0 0%	0 0%	2 14.29%	1 7.14%	0 0%
Campero (13)	0 0%	0 0%	2 15.38%	6 46.15%	0 0%	0 0%	1 7.69%	1 7.69%	1 7.69%	2 15.38%
Furgoneta (6)	0 0%	0 0%	1 16.67%	0 0%	2 33.33%	1 16.67%	0 0%	0 0%	1 16.67%	1 16.67%
Hatchback (12)	1 8.33%	0 0%	0 0%	0 0%	0 0%	4 33.33%	1 8.33%	0 0%	5 41.67%	1 8.33%
Moto (12)	2 16.67%	0 0%	0 0%	0 0%	0 0%	3 25%	7 58.33%	0 0%	0 0%	0 0%
Pickup (13)	0 0%	0 0%	0 0%	1 7.69%	0 0%	0 0%	0 0%	8 61.54%	1 7.69%	3 23.08%
Sedán (12)	0 0%	0 0%	0 0%	0 0%	0 0%	1 8.33%	0 0%	0 0%	10 83.33%	1 8.33%
SUV (12)	0 0%	0 0%	0 0%	1 8.33%	0 0%	3 25%	0 0%	0 0%	2 16.67%	6 50%

Tabla 4.5 Matriz de confusión del sistema con descriptores SURF de mayor desempeño probado.

	Bicicleta	Bus	Camión	Campero	Furgoneta	Hatchback	Moto	Pickup	Sedan	SUV
Bicicleta (13)	13 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (14)	0 0%	0 0%	14 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Campero (13)	0 0%	0 0%	0 0%	8 61.54%	1 7.69%	1 7.69%	0 0%	2 15.38%	0 0%	1 7.69%
Furgoneta (6)	0 0%	0 0%	1 16.67%	0 0%	5 83.33%	0 0%	0 0%	0 0%	0 0%	0 0%
Hatchback (12)	0 0%	0 0%	0 0%	1 8.33%	0 0%	6 50%	0 0%	1 8.33%	3 25%	1 8.33%
Moto (12)	6 50%	0 0%	0 0%	0 0%	0 0%	0 0%	6 50%	0 0%	0 0%	0 0%
Pickup (13)	0 0%	0 0%	1 7.69%	0 0%	0 0%	0 0%	0 0%	12 92.31%	0 0%	0 0%
Sedán (12)	0 0%	0 0%	0 0%	0 0%	0 0%	1 8.33%	0 0%	0 0%	8 66.67%	3 25%
SUV (12)	0 0%	0 0%	0 0%	1 8.33%	0 0%	0 0%	0 0%	2 16.67%	0 0%	9 75%

Tabla 4.6 Matriz de confusión del sistema con descriptores BRIEF de mayor desempeño probado.

	Bicicleta	Bus	Camión	Campero	Furgoneta	Hatchback	Moto	Pickup	Sedan	SUV
Bicicleta (13)	12 92.31%	0 0%	0 0%	0 0%	0 0%	0 0%	1 7.69%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (14)	0 0%	1 7.14%	10 71.43%	1 7.14%	0 0%	0 0%	0 0%	2 14.29%	0 0%	0 0%
Campero (13)	0 0%	0 0%	0 0%	7 53.85%	0 0%	2 15.38%	0 0%	4 30.77%	0 0%	0 0%
Furgoneta (6)	0 0%	0 0%	0 0%	0 0%	6 100%	0 0%	0 0%	0 0%	0 0%	0 0%
Hatchback (12)	0 0%	0 0%	0 0%	0 0%	0 0%	8 66.67%	0 0%	0 0%	4 33.33%	0 0%
Moto (12)	1 8.33%	0 0%	0 0%	0 0%	0 0%	0 0%	11 91.67%	0 0%	0 0%	0 0%
Pickup (13)	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	12 92.31%	0 0%	1 7.69%
Sedán (12)	0 0%	0 0%	0 0%	0 0%	0 0%	2 16.67%	0 0%	0 0%	10 83.33%	0 0%
SUV (12)	0 0%	0 0%	0 0%	2 16.67%	0 0%	1 8.33%	0 0%	1 8.33%	1 8.33%	7 58.33%

Tabla 4.7 Matriz de confusión del sistema con descriptores BRISK de mayor desempeño probado.

	Bicicleta	Bus	Camión	Campero	Furgoneta	Hatchback	Moto	Pickup	Sedan	SUV
Bicicleta (13)	9 69.23%	0 0%	0 0%	1 7.69%	0 0%	0 0%	1 7.69%	1 7.69%	1 7.69%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (14)	0 0%	2 14.29%	8 57.14%	1 7.14%	0 0%	0 0%	2 14.29%	1 7.14%	0 0%	0 0%
Campero (13)	0 0%	0 0%	5 38.46%	4 30.77%	1 7.69%	0 0%	0 0%	2 15.38%	1 7.69%	0 0%
Furgoneta (6)	0 0%	1 16.67%	0 0%	1 16.67%	2 33.33%	0 0%	1 16.67%	1 16.67%	0 0%	0 0%
Hatchback (12)	0 0%	0 0%	0 0%	0 0%	1 8.33%	5 41.67%	1 8.33%	0 0%	3 25%	2 16.67%
Moto (12)	3 25%	0 0%	0 0%	1 8.33%	0 0%	0 0%	7 58.33%	0 0%	1 8.33%	0 0%
Pickup (13)	0 0%	0 0%	2 15.38%	1 7.69%	3 23.08%	0 0%	1 7.69%	5 38.46%	0 0%	1 7.69%
Sedán (12)	3 25%	0 0%	1 8.33%	0 0%	0 0%	5 41.67%	0 0%	0 0%	2 16.67%	1 8.33%
SUV (12)	2 16.67%	0 0%	0 0%	1 8.33%	1 8.33%	0 0%	1 8.33%	0 0%	2 16.67%	5 41.67%

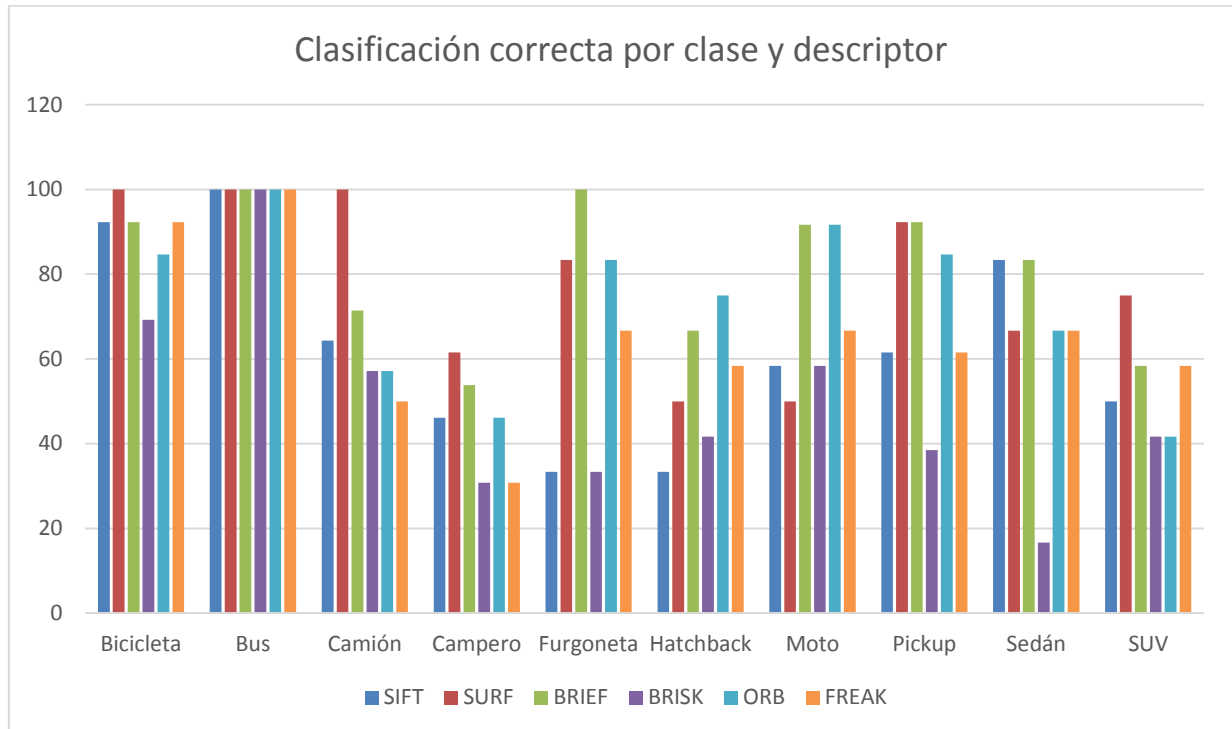
Tabla 4.8 Matriz de confusión del sistema con descriptores ORB de mayor desempeño probado.

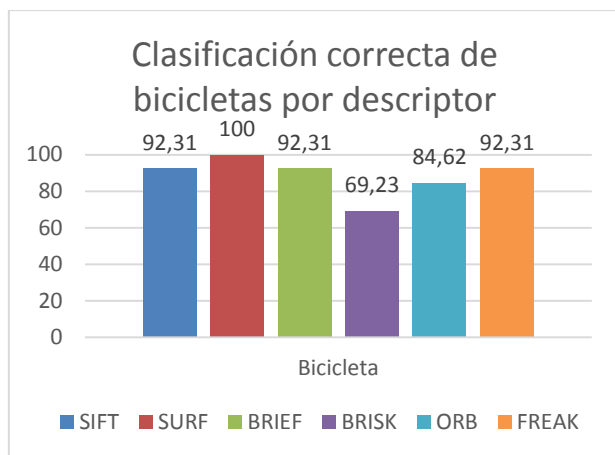
	Bicicleta	Bus	Camión	Campero	Furgoneta	Hatchback	Moto	Pickup	Sedan	SUV
Bicicleta (13)	11 84.62%	0 0%	1 7.69%	0 0%	0 0%	0 0%	1 7.69%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (14)	0 0%	0 0%	8 57.14%	2 14.29%	1 7.14%	0 0%	0 0%	3 21.43%	0 0%	0 0%
Campero (13)	0 0%	0 0%	1 7.69%	6 46.15%	0 0%	2 15.38%	2 15.38%	0 0%	0 0%	2 15.38%
Furgoneta (6)	0 0%	0 0%	0 0%	1 16.67%	5 83.33%	0 0%	0 0%	0 0%	0 0%	0 0%
Hatchback (12)	0 0%	0 0%	0 0%	0 0%	0 0%	9 75%	0 0%	0 0%	3 25%	0 0%
Moto (12)	1 8.33%	0 0%	0 0%	0 0%	0 0%	0 0%	11 91.67%	0 0%	0 0%	0 0%
Pickup (13)	0 0%	0 0%	0 0%	0 0%	0 0%	1 7.69%	0 0%	11 84.61%	0 0%	1 7.69%
Sedán (12)	0 0%	0 0%	0 0%	0 0%	0 0%	2 16.67%	0 0%	1 8.33%	8 66.67%	1 8.33%
SUV (12)	0 0%	0 0%	0 0%	2 16.67%	1 8.33%	1 8.33%	0 0%	2 16.67%	1 8.33%	5 41.67%

Tabla 4.9 Matriz de confusión del sistema con descriptores FREAK de mayor desempeño probado.

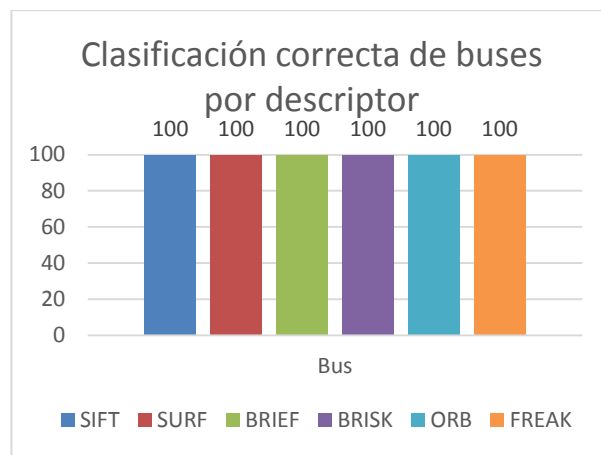
	Bicicleta	Bus	Camión	Campero	Furgoneta	Hatchback	Moto	Pickup	Sedan	SUV
Bicicleta (13)	12 92.31%	0 0%	0 0%	0 0%	0 0%	0 0%	1 7.69%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (14)	0 0%	2 14.29%	7 50%	1 7.14%	1 7.14%	0 0%	0 0%	2 14.29%	0 0%	1 7.14%
Campero (13)	0 0%	0 0%	5 38.46%	4 30.77%	0 0%	1 7.69%	1 7.69%	2 15.38%	0 0%	0 0%
Furgoneta (6)	0 0%	0 0%	0 0%	1 16.67%	4 66.67%	0 0%	0 0%	1 16.67%	0 0%	0 0%
Hatchback (12)	0 0%	0 0%	0 0%	2 16.67%	0 0%	7 58.33%	0 0%	0 0%	1 8.33%	2 16.67%
Moto (12)	1 8.33%	0 0%	2 16.67%	0 0%	0 0%	1 8.33%	8 66.67%	0 0%	0 0%	0 0%
Pickup (13)	0 0%	1 7.69%	0 0%	1 7.69%	1 7.69%	1 7.69%	0 0%	8 61.54%	0 0%	1 7.69%
Sedan (12)	0 0%	0 0%	0 0%	1 8.33%	1 8.33%	0 0%	0 0%	0 0%	8 66.67%	2 16.67%
SUV (12)	0 0%	0 0%	1 8.33%	2 16.67%	1 8.33%	1 8.33%	0 0%	0 0%	0 0%	7 58.33%

Los gráficos de barras presentados en la **Figura 4.2**, muestran los resultados de los elementos en la diagonal de las matrices de confusión, es decir, las tasas de clasificación correcta para cada uno de los descriptores visuales considerados.

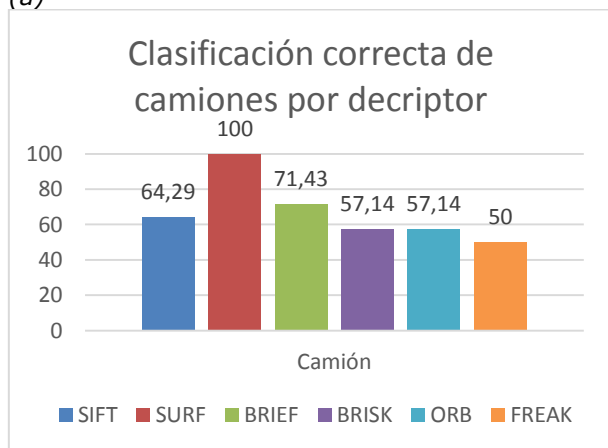




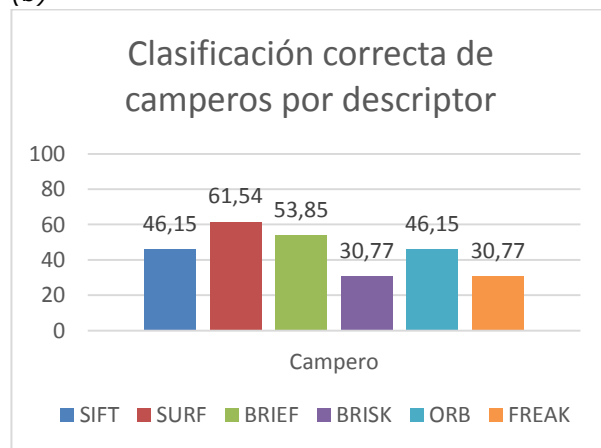
(a)



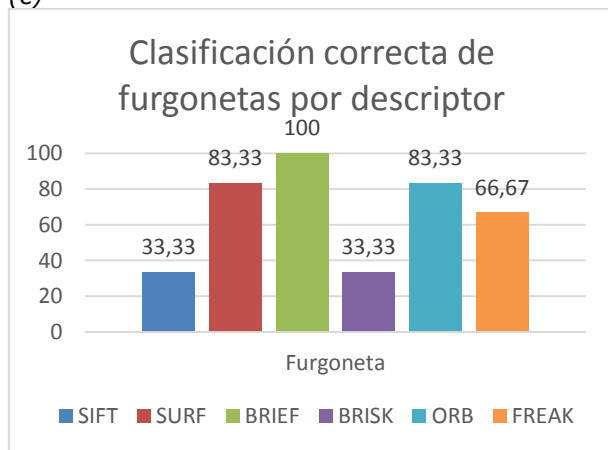
(b)



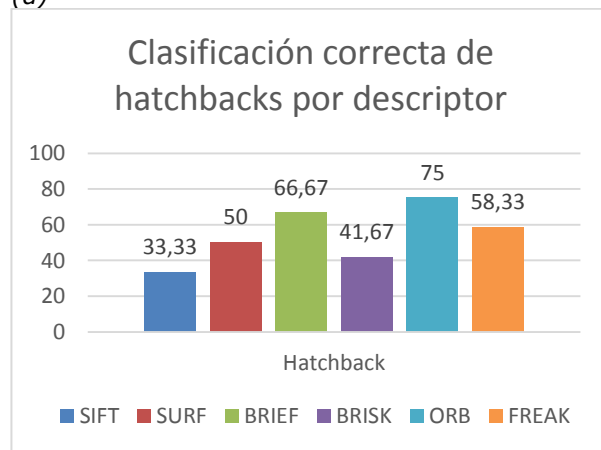
(c)



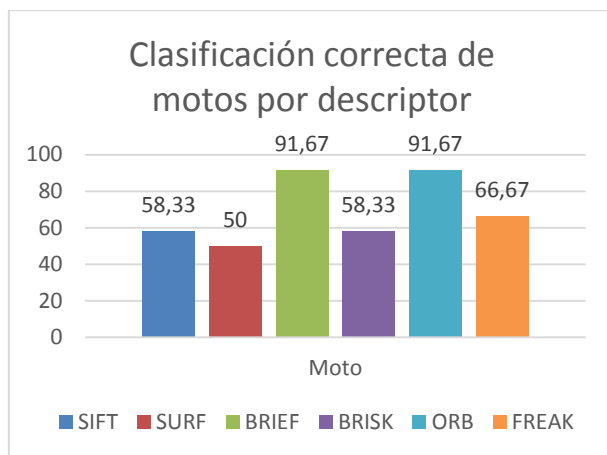
(d)



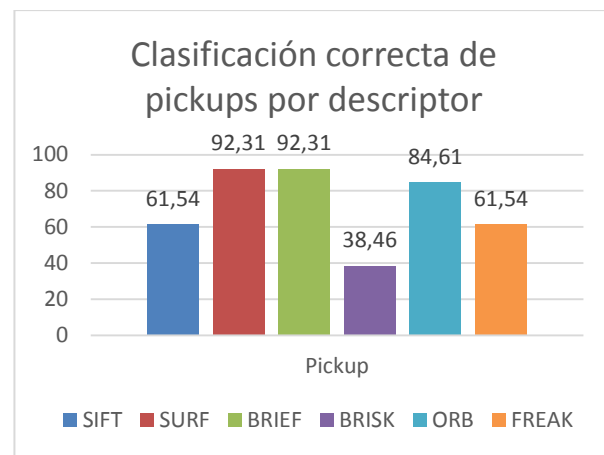
(e)



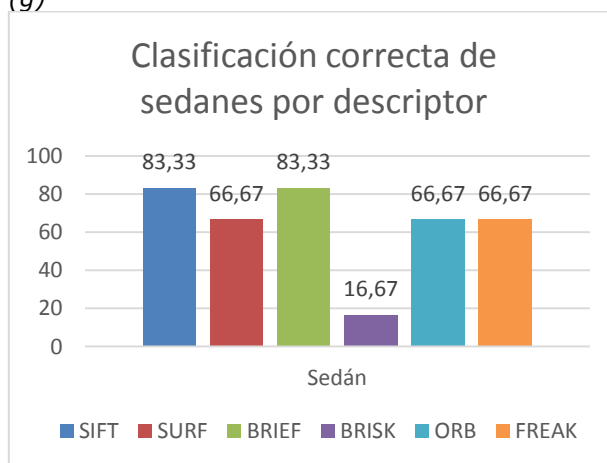
(f)



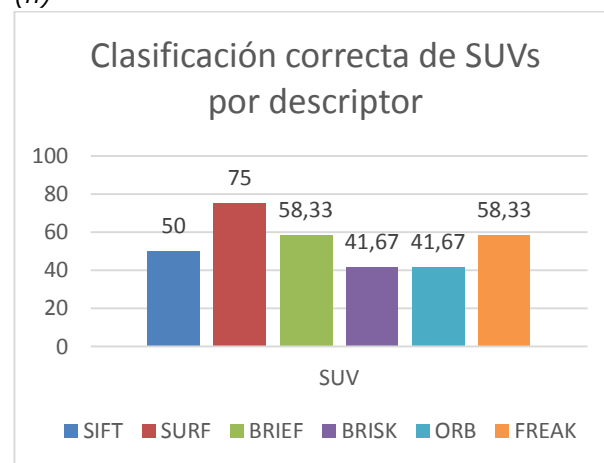
(g)



(h)



(i)



(j)

Figura 4.2 Tasas de clasificación correcta por cada tipo de vehículo y cada descriptor visual. (a)Bicicleta (b)Bus (c)Camión (d)Campero (e)Furgoneta (f)Hatchback (g)Moto (h)Pickup (i)Sedán (j)SUV

En la **Tabla 4.10** se presenta un compendio de los resultados obtenidos para el sistema de clasificación por medio de información visual. Los promedios presentados son los promedios, simple y ponderado, de clasificación correcta de cada descriptor considerando todas las clases (vehículos). Para el promedio simple se presenta también la desviación estándar de las tasas de clasificación. En la **Tabla 4.11** se presentan las cantidades de datos por clase y el factor de ponderación de cada una, en donde el factor de ponderación se calcula como el número de datos de una clase dividido el total de datos de validación.

Tabla 4.10 Promedio, desviación estándar y promedio ponderado de los sistemas con los diferentes descriptores visuales

	SIFT	SURF	BRIEF	BRISK	ORB	FREAK
Promedio	62.261	77.89	80.99	48.73	73.09	65.13
Desv. Est.	22.0795	19.1346	16.2637	22.3706	18.5798	18.6458
Prom. Pond.	60.5497	76.1474	77.982	44.9541	69.7247	61.4686

Tabla 4.11 Cantidad de datos por clase y factor de ponderación de cada clase

CLASE	Bici	Bus	Camión	Camp.	Furg.	Hatchb.	Moto	Pickup	Sedán	SUV
Datos/clase	13	2	14	13	6	12	12	13	12	12
Ponderación	0.1192	0.0183	0.1284	0.1192	0.0550	0.1100	0.1100	0.1192	0.1100	0.1100

Se puede observar que, en el caso de algunos descriptores, se obtuvieron tasas de clasificación bastante satisfactorias teniendo en cuenta una serie de factores:

- La diversidad del *dataset* es supremamente alta, conteniendo 10 clases en total, algunas de las cuales presentan similitudes muy marcadas. Debido a que la técnica utilizada (*Bag of Features*) se basa en pequeñas porciones de la imagen que, se asume, se repiten en todas las instancias de una misma clase, la posibilidad confusión entre clases como SUV y campero o hatchback y sedan, es alta, ya que estas clases presentan subestructuras muy parecidas.
- Una posible fuente de error fue identificada después de realizar las pruebas con el módulo de clasificación visual. En la **Figura 3.21** se puede observar un ejemplo de un dato de entrada típico para la obtención de puntos clave. El software de clasificación visual, a lo largo de su proceso, dibuja una línea roja alrededor de los contornos que representan vehículos, al igual que un rectángulo verde que los encierra. Estas líneas aparecen en la segmentación final, justo antes de extraer características y atendiendo a los ejemplos presentados en la figura, se puede observar que muchos de los puntos clave son arrastrados por esta anomalía en la escena. Ya que TODOS los datos contienen la misma información, esto puede estar causando confusiones en el sistema.
- Otra posible fuente de error es la presencia de una parte de la escena al interior del área de donde se extraen características. Esto también es común a todos los datos y puede causar confusiones en el sistema. Esto es fácilmente reparable por medio de una operación morfológica de erosión, que elimine parte del borde antes de pasar al proceso de extraer puntos clave.
- La baja resolución del sensor, de tan solo 640x480 pixeles, lo que, sumado a la distancia entre el objeto de interés y la cámara, da como resultado imágenes con calidades bastante bajas.

4.3.2 MÓDULO CLASIFICACIÓN DE VEHÍCULOS CON INFORMACIÓN DE RANGO

En la sección previa quedó definida la evaluación del módulo basado en información visual y se se presentan los resultados y su análisis. En esta etapa se procede de la misma forma para el módulo el módulo basado en información de rango, se presentan las matrices de confusión del sistema para sistema para cada descriptores las tablas (**Tabla 4.12-**

Tabla 4.16) y los resultados de las tasas de reconocimiento correctas para cada clase con cada tipo de descriptor (**Figura 4.3**), y un análisis para el desempeño de cada descriptor.

Tabla 4.12 Matriz de confusión del sistema con descriptores VFH de mayor desempeño probado.

	Bici, Moto.	Bus	Camión	Campero	Furgoneta	Hatch, SW, WB.	Pickup	Sedan	SUV
Bici, Moto. (38)	38 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (21)	0 0%	1 4.76%	18 85.71%	2 9.52%	0 0%	0 0%	0 0%	0 0%	0 0%
Campero (21)	0 0%	0 0%	0 0%	13 61.90%	1 4.76%	0 0%	2 9.52%	0 0%	5 23.81%
Furgoneta (9)	0 0%	0 0%	0 0%	0 0%	8 88.89%	0 0%	0 0%	0 0%	1 11.11%
Hatch, SW, WB. (17)	0 0%	0 0%	0 0%	1 5.88%	0 0%	13 76.47%	0 0%	1 5.88%	2 11.76%
Pickup (16)	0 0%	0 0%	1 6.25%	3 18.75%	1 6.25%	0 0%	7 43.75%	1 6.25%	3 18.75%
Sedán (17)	0 0%	0 0%	0 0%	0 0%	0 0%	3 17.65%	0 0%	14 82.35%	0 0%
SUV (18)	0 0%	0 0%	0 0%	1 5.56%	0 0%	0 0%	0 0%	0 0%	17 94.44%

Tabla 4.13 Matriz de confusión del sistema con descriptores ESF de mayor desempeño probado.

	Bici, Moto.	Bus	Camión	Campero	Furgoneta	Hatch, SW, WB.	Pickup	Sedan	SUV
Bici, Moto. (38)	38 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (21)	0 0%	1 4.76%	12 57.14%	4 19.05%	3 14.29%	0 0%	1 4.76%	0 0%	1 4.76%
Campero (21)	0 0%	0 0%	0 0%	14 66.67%	1 4.76%	0 0%	0 0%	0 0%	5 23.81%
Furgoneta (9)	0 0%	0 0%	2 22.22%	4 44.44%	2 22.22%	0 0%	0 0%	0 0%	1 11.11%
Hatch, SW, WB. (17)	0 0%	0 0%	0 0%	1 5.88%	1 5.88%	11 64.71%	2 11.76%	1 5.88%	1 5.88%
Pickup (16)	0 0%	0 0%	0 0%	3 18.75%	1 6.25%	1 6.25%	6 37.50%	1 6.25%	5 31.25%
Sedán (17)	0 0%	0 0%	0 0%	0 0%	0 0%	3 17.65%	3 17.65%	11 64.71%	0 0%
SUV (18)	0 0%	0 0%	0 0%	4 22.22%	0 0%	0 0%	0 0%	0 0%	13 72.22%

Tabla 4.14 Matriz de confusión del sistema con descriptores FPFH de mayor desempeño probado.

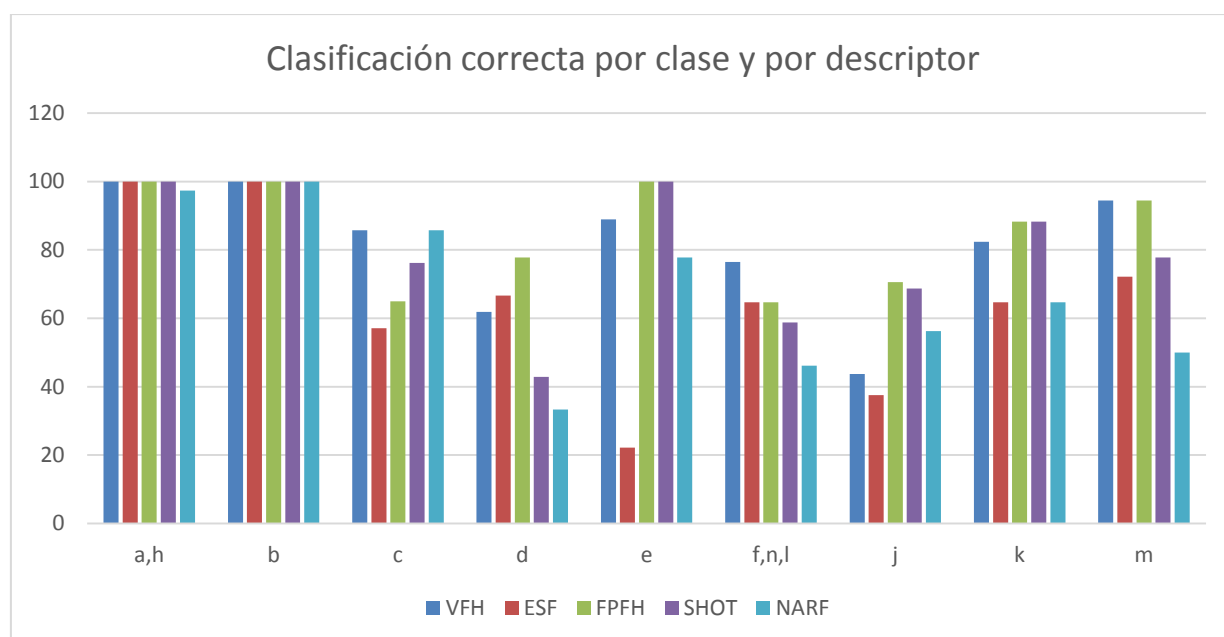
	Bici, Moto.	Bus	Camión	Campero	Furgoneta	Hatch, SW, WB.	Pickup	Sedan	SUV
Bici, Moto. (38)	38 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Bus (3)	0 0%	3 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (20)	0 0%	4 20%	13 65%	0 0%	2 10%	0 0%	0 0%	0 0%	1 5%
Campero (18)	0 0%	0 0%	0 0%	14 77.78%	1 5.56%	0 0%	0 0%	0 0%	3 16.67%
Furgoneta (9)	0 0%	0 0%	0 0%	0 0%	9 100%	0 0%	0 0%	0 0%	0 0%
Hatch, SW, WB. (17)	0 0%	0 0%	0 0%	0 0%	0 0%	11 64.71%	0 0%	6 35.29%	0 0%
Pickup (17)	0 0%	0 0%	0 0%	4 23.55%	0 0%	0 0%	12 70.59%	1 5.88%	0 0%
Sedán (17)	0 0%	0 0%	0 0%	0 0%	0 0%	2 11.76%	0 0%	15 88.24%	0 0%
SUV (18)	0 0%	0 0%	0 0%	1 5.56%	0 0%	0 0%	0 0%	0 0%	17 94.44%

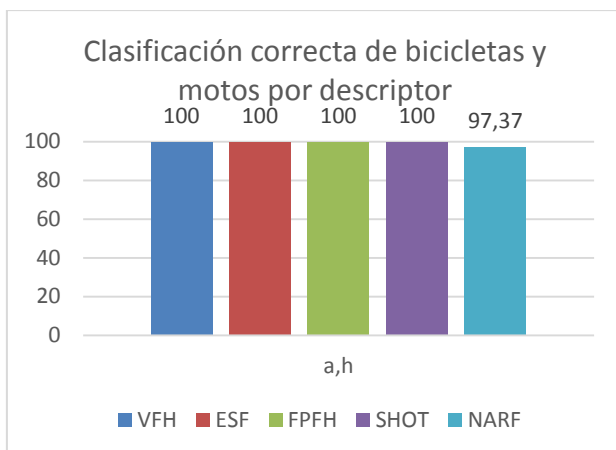
Tabla 4.15 Matriz de confusión del sistema con descriptores SHOT de mayor desempeño probado

	Bici, Moto.	Bus	Camión	Campero	Furgoneta	Hatch, SW, WB.	Pickup	Sedan	SUV
Bici, Moto. (38)	38 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (21)	0 0%	4 19.05%	16 76.19%	1 4.76%	0 0%	0 0%	0 0%	0 0%	0 5%
Campero (21)	0 0%	0 0%	2 9.52%	9 42.86%	3 14.29%	0 0%	2 9.52%	0 0%	5 23.81%
Furgoneta (9)	0 0%	0 0%	0 0%	0 0%	9 100%	0 0%	0 0%	0 0%	0 0%
Hatch, SW, WB. (17)	0 0%	0 0%	0 0%	0 0%	2 11.76%	10 58.82%	1 5.88%	3 17.65%	1 5.88%
Pickup (16)	0 0%	0 0%	1 6.25%	0 0%	1 6.25%	0 0%	11 68.75%	0 0%	3 18.75%
Sedán (17)	0 0%	0 0%	0 0%	0 0%	0 0%	2 11.76%	0 0%	15 88.24%	0 0%
SUV (18)	0 0%	0 0%	0 0%	1 5.56%	2 11.11%	1 5.56%	0 0%	0 0%	14 77.78%

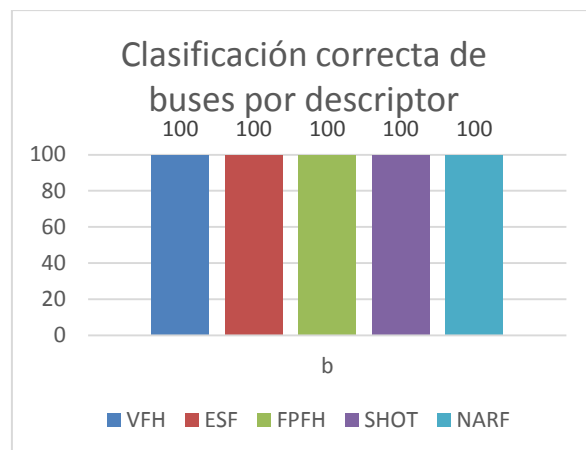
Tabla 4.16 Matriz de confusión del sistema con descriptores NARF de mayor desempeño probado.

	Bici, Moto.	Bus	Camión	Campero	Furgoneta	Hatch, SW, WB.	Pickup	Sedan	SUV
Bici, Moto. (38)	37 97.37%	0 0%	1 2.63%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Bus (2)	0 0%	2 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
Camión (21)	0 0%	2 9.52%	18 85.71%	1 4.76%	0 0%	0 0%	0 0%	0 0%	0 0%
Campero (21)	0 0%	0 0%	3 14.29%	7 33.33%	4 19.05%	0 0%	2 9.52%	0 0%	5 23.81%
Furgoneta (9)	0 0%	0 0%	1 11.11%	1 11.11%	7 77.78%	0 0%	0 0%	0 0%	0 0%
Hatch, SW, WB. (26)	0 0%	0 0%	1 3.85%	2 7.69%	2 7.69%	12 46.15%	6 23.08%	2 7.69%	1 3.85%
Pickup (16)	0 0%	0 0%	0 0%	2 12.5%	1 6.25%	0 0%	9 56.25%	0 0%	4 25%
Sedán (17)	0 0%	0 0%	0 0%	0 0%	1 5.88%	3 17.65%	2 11.76%	11 64.71%	0 0%
SUV (18)	0 0%	0 0%	1 5.56%	2 11.11%	5 27.78%	0 0%	1 5.56%	0 0%	9 50%

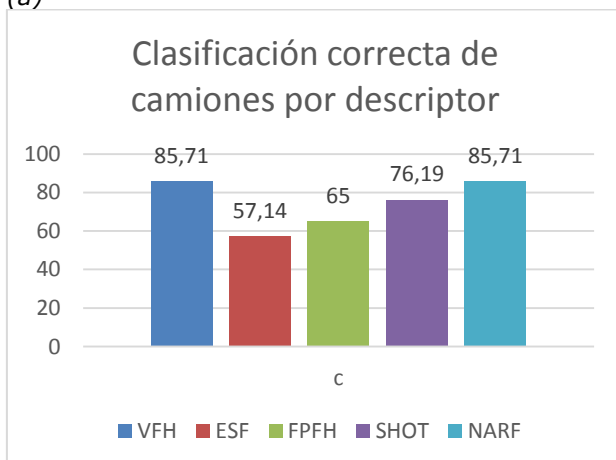




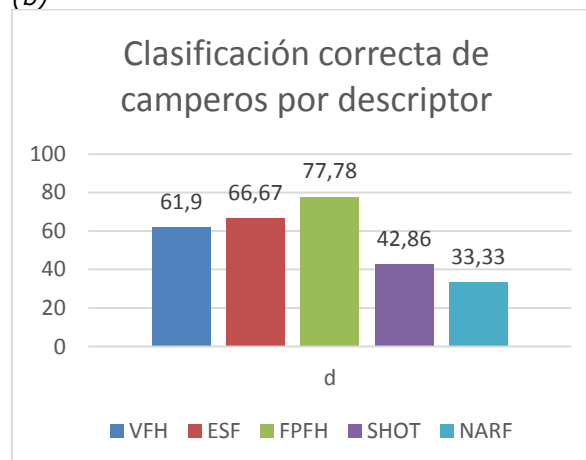
(a)



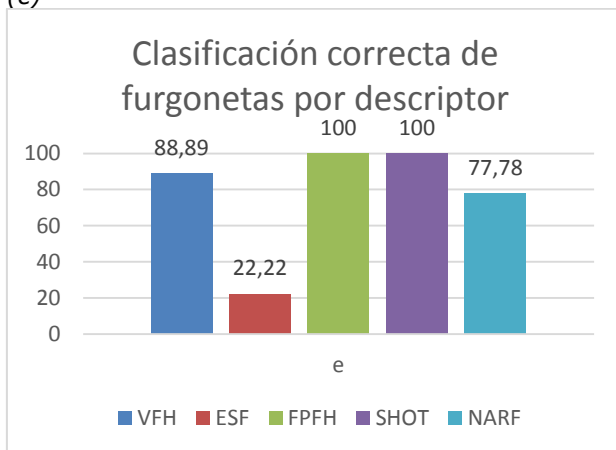
(b)



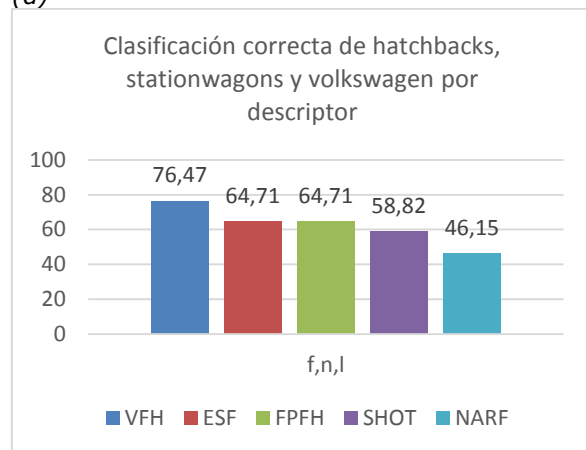
(c)



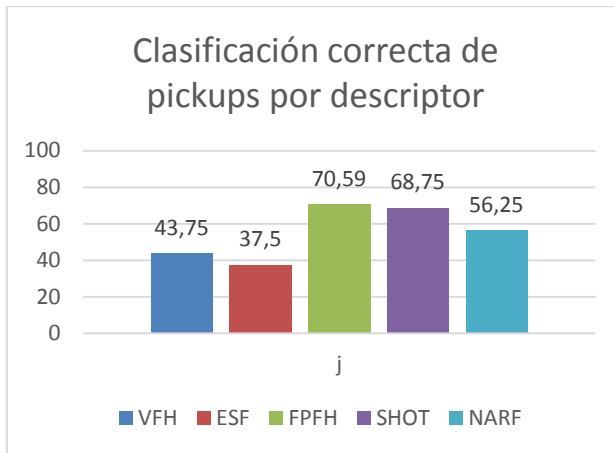
(d)



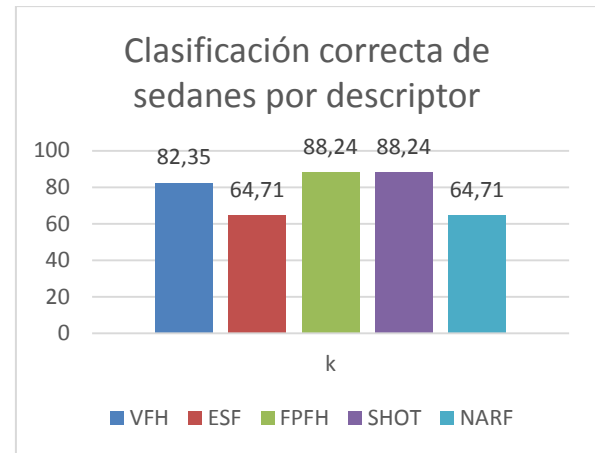
(e)



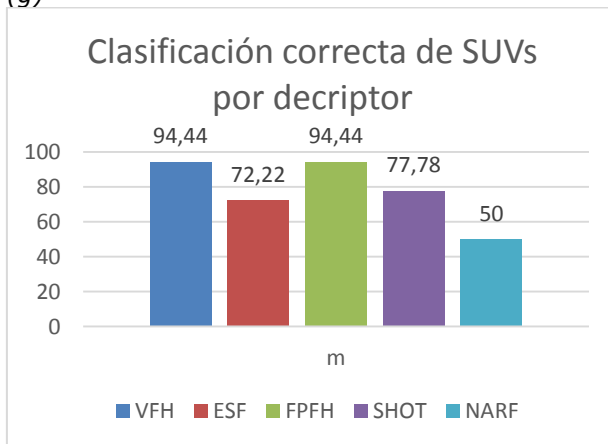
(f)



(g)



(h)



(i)

Figura 4.3 Tasas de clasificación correctas por cada clase de vehículo y para cada descriptor. (a)Bicicletas y motos (b)Buses (c)Camiones (d)Camperos (e)Furgonetas (f)Hatchback, Stationwagon y Volkswagen (g)Pickups (h)Sedanes (i)SUVs

Las medidas estadísticas del promedio, promedio ponderado y desviación estándar de los resultados de las tasas correctas de clasificación se consignan en la **Tabla 4.17**. Adicionalmente se consigna en la **Tabla 4.18** el número de elementos por clase y la ponderación de cada clase. Los promedios ponderados representan una tasa de reconocimiento global del sistema teniendo en cuenta, la cantidad de valores que contribuyen en los porcentajes por cada clase.

Tabla 4.17 Medidas estadísticas de los resultados.

	VFH	ESF	FPFH	SHOT	NARF
Promedio	81.501111	65.018889	84.528889	79.182222	67.922222
Desv. Est	18.57258	25.38021	15.182532	20.111743	23.580058
Prom. Pond.	81.987389	68.665223	84.077325	78.752166	68.503758

Tabla 4.18 Número de datos y ponderaciones por clase.

CLASE	a,h	b	c	d	e	f,n,l	j	k	m
Datos/clase	38	3	20	18	9	17	17	17	18
Ponderación	0.24203	0.01910	0.12738	0.11464	0.05732	0.10828	0.10828	0.10828	0.11464

La clasificación con **VFH** tiene un promedio final acumulado ponderado resultante de **81.987389%** presenta resultados altos para la mayoría de las clases, sin embargo la categoría pickup es bastante problemática pues estos vehículos se encuentran en un punto

intermedio de tamaño entre los grandes y los pequeños, por lo tanto la umbralización de altura no es lo suficientemente diciente o robusta, además la parte delantera de estos es muy similar al contorno cuadrado de los camperos, y los que tienen una bandeja con telón, asemejan en el modelo 3D con el de las SUV. La separación campero vs SUV es ambigua incluso para los seres humanos pero se incluyó, buscando la mayor clasificación intraclase posible. Consideramos que los resultados de 70% en adelante son buenos dada las similitudes tan grandes existentes entre algunas clases, pues son separaciones difíciles incluso para los seres humanos. Es importante rescatar que aunque este descriptor es global y es considerado para una macro vecindad centrada en el punto promedio del clúster (característica no muy apreciada en la clasificación intraclase), entrega buenos resultados y valores acumulados altos para este tipo de clasificación tan exigente, esto es posible gracias a que en principio está basado en un descriptor local altamente robusto (pero lento) denominado *Point Feature Histograms* (PFH) (Radu Bogdan Rusu, Marton, et al., n.d.) Que le brinda mucha información sobre los cambios en curvatura y las normales de superficie de la nube.

ESF es el descriptor con el porcentaje acumulado más bajo en todo el sistema **68.665223%**, se incluye con fines de comparación y de evaluación. Su resultado tan bajo es determinado por la ausencia en su creación del uso de las normales de superficie y de la información de curvatura de los objetos, su concepción conceptual es útil para elementos con distribuciones de densidad y organización de puntos muy similar (modelos computarizados 3D) o capturas con condiciones muy estrictas altamente repetibles (donde logra tasas extremadamente altas), situación que obviamente es difícil de cumplir en este trabajo, por la variación en velocidad de los vehículos, ruido del sensor, desplazamiento inclinado de los vehículos no perpendicular al eje central del láser. En condiciones difíciles como las mencionadas este sensor tiene resultados buenos o aceptables en las clases más diferentes, pero claramente no es apropiado para una clasificación robusta intraclase.

FPFH es el descriptor local más valioso de todos los empleados en el sistema, esto es debido a que es el que mejor describe cada cambio geométrico, de curvatura, de comportamiento en las vecindades locales de cada punto, además el hecho de ser local permite el uso de una integración por bolsa de características, lo que termina entregando un vector de frecuencias altamente discriminante incluso para las difíciles clasificaciones intraclase. Otra ventaja que posee es la gran cantidad de parámetros de control de entrada tanto en su estimación como en la etapa de agrupación de correspondencias, por lo que permite adaptarlo diferentes condiciones de trabajo, diferentes set de datos, y diferentes ponderaciones de clasificación, esto indica que puede ajustarse para que sea muy robusto diferenciando ciertas clase importantes, y menos discriminante en otras. Dada la gran cantidad de condiciones y parámetros que pueden brindársele al sistema para este descriptor, se consignan aquí 6 resultados diferentes, unos con promedios acumulados muy altos para clasificación intraclase (9 a 12 subclases, para máximo 3 clases en la mayoría de trabajos en la literatura, como ejemplo (Valverde Pedraza & Ossa Torres, 2011)) como **83.360 % y 84.077 %**, otros con promedios acumulados más bajos **82.712%**, pero, con porcentajes altos en todas las 9 clases y casos excepcionales con porcentajes más bajos acumulados **76.82142 %** pero con varias clases(4) en **100% y 95%**. Este descriptor demuestra la importancia de considerar los cambios en cada localidad de la vecindad de cada punto, más que en una vecindad global (lo que es generalmente rápido pero apropiado para clasificación extra clase). Es importante mencionar que la categoría pickup no es problemática con este tipo de descriptor, las dos que más se ven comprometidas son camión (debido a que no se implementó al momento de la toma de resultados una separación entre vehículos grandes y extra grandes, lo que confunde muchos camiones con cuerpos rectangulares con los buses) y Hatchbacks, pues es atacada en el caso de los pequeños por los sedan, que comparten una similitud del 80% de su cuerpo excepto la parte trasera, y en

el caso de los grandes por las SUV, que son extremadamente similares pero de gran tamaño(aunque existen nuevos vehículos que se encuentran en el punto medio de tamaño de las dos clases pero, con la misma forma, ejemplo: Renault stepway).

SHOT es el precursor de gran cantidad de los descriptores más modernos, es uno de los más elaborados conceptualmente pues establece marcos de referencia locales repetibles, para la acumulación consiente y coherente de histogramas de orientación de normales, y además realiza procesos de interpolación para alisar los cambios bruscos entre las vecindades locales definidas, entre los puntos pertenecientes a una vecindad esférica. Ofrece un resultado muy bueno de **78.7521%** para una clasificación intraclase, presenta problemas en la clase más complicada para los seres humanos, esto es los camperos, es una clase altamente ambigua y muy subjetiva, sin embargo es capaz de entregar resultados buenos incluso sin hacer uso de descriptores geométricos adicionales, pues entiende muy bien las características de una superficie. Como desventaja tiene su dimensionalidad muy grande de 352 contenedores, lo que aumenta la complejidad en las etapas de integración de bolsa de características, si bien esto no afecta su uso en tiempo real pues esta demora es en el entrenamiento, limita un poco la variación de parámetros y la prueba de distintas configuraciones, por el tiempo que toma cada vez.

NARF es el descriptor más elaborado y complejo en su diseño e implementación pues hace uso de información 3D como nube de puntos y de imágenes de rango, realiza detección de bordes 3D, además realiza la determinación de sus propios puntos característicos, basado en índices de importancia en las regiones de bordes 3D, sin embargo su correcto funcionamiento requiere de la adquisición de en gran medida una plataforma idónea como un sensor dual calibrado (Kinect) que entregue nubes de puntos organizadas e imágenes de rango bien elaboradas, este descriptor es útil para la clasificación intraclase, pues brinda importancia a cada borde o pliegue 3D en el interior además de los bordes exteriores, sin embargo, para los carros en particular, su textura lateral (tanto textura 2D como su textura 3D, que es la mayor parte observable en los clúster capturados) es demasiado similar, por lo que para esta aplicación en particular su uso no es discriminante, su porcentaje ponderado acumulado es de **68.5037%**, solo presenta un resultado extremadamente bajo en campero por la razones previamente mencionadas de su ambigüedad con las SUV.

Se observa que el resultado ponderado acumulado más alto del módulo basado en información de rango es de **84.077 %**, esto es logrado utilizando el máximo nivel de descripción por cada clúster, esto implica la extracción de características para la vecindad local de cada punto (si no hay muestreo de puntos claves) o de una versión reducida de cada nube que utiliza toda la información disponible, pero solo hace el cálculo para los puntos seleccionados. Este tipo de descripción es significativamente más completa y representativa, pero requiere pasos adicionales de integración y agrupación que permitan una comparación uno a uno entre cada elemento.

A primera vista, se puede juzgar muy duramente el porcentaje máximo alcanzado, sin embargo existen varias razones que se mencionan a continuación, por lo cual se considera que este porcentaje es bastante alto y da una buena impresión de la robustez del sistema; en primer lugar el proceso de toma del dataset es extremadamente libre, permitió comportamientos muy variados y arbitrarios en los vehículos, como, distintos y abruptos cambios en velocidad en la región de toma (mediante la verificación escaneo a escaneo de la relación entre tiempos de cámara, laser, y velocidades suavizadas de kalman en cada instante de tiempo), incursión irregular en el haz de escaneo a diferentes distancias o inclinaciones del origen del sensor, no controladas, pero incluidas en las características de los descriptores, y la no imposición de prácticamente ninguna restricción en la forma en la que los vehículos deben ser registrados (solo exige una velocidad inferior a los 60km debido a la limitación en velocidad del sensor laser, aunque esto puede ser solucionado si se utiliza

un sensor que capture toda la información del elemento en una sola toma 3D, como un Kinect o algún sensor con este principio, o con láseres 2D de mayor velocidad presentes en el mercado 100Hz-150Hz).

En segundo lugar según es del cocimiento de los autores, este trabajo es pionero en la clasificación extremadamente intracase para vehículos (9 subclases a partir de máximo 2 o 3 clases en la distribución común de tipos de vehículos), según la bibliografía consultada, por lo que una comparación con sistemas con clases bien diversificadas no es apropiada; lo mencionado tiene una ventaja significativa en la percepción en entornos robóticos donde generalmente las clases magnas de elementos son pocas, pero con muchas variaciones internas; se considera entonces que los resultados para este tipo de sistema son bastante satisfactorios y robustos.

Como última razón que se considera este resultado como muy satisfactorio, es el hecho de que solo se haya usado una nube de puntos lateral-superior del vehículo, donde la parte superior del vehículo es poco registrada debido a que, a menos que el sensor laser tenga su eje central perpendicular al techo del vehículo (o al plano tangente), el desplazamiento del mismo ocluye, los puntos contiguos a los puntos vecinos en escaneos consecutivos (generando un efecto arco), si se puede adicionar-registrar una nube de puntos tomada superiormente, se obtendría un perfil completo 3D (mediante un espejo del otro lado) que sin duda entregaría aún más información diciente de cada clase de vehículo. Y podría aportar una decena adicional a los porcentajes acumulados. En este trabajo se utilizaron 5 descriptores distintos con fines de evaluación y análisis, se resumen los porcentajes máximos acumulados como sigue:

<i>VFH(Global)</i> → 81.987389%		<i>ESF(Global)</i> → 68.665223%,	
<i>FPFH(Local)</i> → 84.077325%	<i>SHOT(Local)</i> → 78.7521%	<i>NARF(Local)</i> → 68.5037%	

Un punto importante a mencionar es que si bien no todos los modelos entrenados ofrecen resultados acumulados muy altos, si presentan capacidades excepcionales para el reconocimiento casi perfecto de cierto tipo de clases en la información de prueba, esto puede llegar a ser útil, para sistemas de reconocimiento en general, que exijan separaciones muy robustas para clases específicas (automatización industrial). Se concluye en este trabajo que la descripción local (al estilo de los descriptores 2D locales SIFT, SURF etc.) es más informativa y robusta si bien exige etapas adicionales y el ajuste de parámetros que afectan significativamente en el éxito de la agrupación de correspondencias, mientras que la global presenta un entrenamiento más rápido, y un proceso menos complicado de ajuste.



5 CONCLUSIONES

En este trabajo se realizó el diseño conceptual y la implementación software (módulos de procesamiento multimodal) y hardware (ajuste y distribución de los sensores, circuitería de adquisición de información) de un sistema inteligente y automático para la detección y el reconocimiento de 9 y 10 clases diferentes de vehículos terrestres, en su mayoría pequeños y medianos. El sistema desarrollado cuenta con 2 macro módulos integrados en ciertos aspectos de su implementación, para la clasificación basada en información 2D y 3D. Se elaboró este sistema buscando robustez en la clasificación intraclase, problema que puede ser abordado por sistemas con varios tipos de información presente.

Se demostró la importancia de un buen filtrado de la información entregada por cualquier sistema que contenga ruido y que pueda ser modelado mediante un esquema de estados. De esta manera, en este trabajo se logró un registrado correcto y suavizado de la velocidad de los vehículos en el momento del ingreso al sistema, lo que permitió la reconstrucción 3D correcta de cada modelo sin inconvenientes y con tasas de correlación muy altas.

Se realizó la construcción de un *dataset* propio con información de las características físicas y visuales de los vehículos terrestres que ingresan a la Universidad del Valle mediante el diseño de un esquema de toma de datos definido según las características propias del entorno de trabajo. Este dataset logró contener gran variedad de clases principales y variaciones subclase lo que permite realizar buenas estimaciones de lo que sería el comportamiento del sistema una vez sea totalmente implementado. Es importante añadir que este *dataset* exige un muy buen proceso de etiquetado (para el entrenamiento) y descripción de cada instancia (para entrenamiento y validación), pues su adquisición fue bastante libre y poco restrictiva, tratando de exigir al máximo el sistema diseñado y hacerlo funcionar en las condiciones más reales posibles, que a su vez representan las de mayor utilidad.

Dada la naturaleza de la información recolectada y gracias a ciertas condiciones que siempre se cumplieron, en este caso, el hecho de que los vehículos siempre transitaban a una distancia más o menos igual de la cámara, siempre en la misma dirección y siempre en sentido de derecha a izquierda en la toma, no se hace necesaria la invarianza con respecto a la escala y a la rotación de los descriptores visuales. Esto se evidencia en el hecho de que la mayor tasa de clasificación correcta fue obtenida con los descriptores tipo BRIEF, descriptores que no presentan características de invariabilidad en escala y rotación y, por el contrario, se ven fuertemente afectados por cambios en estas propiedades.

Si bien el sistema puede ser ajustado para un comportamiento completamente automático, se diseñó un esquema de pruebas semiautomático para la validación del comportamiento total y las tasas de éxito del sistema multimodal, haciendo uso de diferentes configuraciones

y distintos descriptores para cada módulo consignando sus virtudes y debilidades. Estas pruebas entregaron máximas tasas acumuladas promedio por prueba de **84.5280%** para el descriptor 3D más robusto y discriminante **FPFH**, así como resultados similarmente buenos para **VFH 81.50287778%** y **SHOT 79.1816%**, para la información de rango 3D. De modo similar, para el módulo de información visual 2D, se obtuvieron tasas de **77.98%** para el descriptor BRIEF calculado sobre puntos clave tipo FAST y **76.14%** para el descriptor SURF, ambos con la combinación de parámetros libres adecuada, detallada más arriba en la sección de pruebas y resultados. Estos resultados son bastantes satisfactorios dada la alta variabilidad presente en nuestro dataset, y el esfuerzo por no sobre entrenar el sistema, conservando una buena generalización.

Se realizó con éxito la integración de las características (en cada caso) locales visuales y de rango, mediante el uso de una técnica muy robusta de bolsa de características, esto permitió la extracción libre de diferentes números descriptores del mismo tipo en cada caso, y la realización de un proceso de consolidación de las mejores opciones de modelos similares en la base de datos presente.

5.1 TRABAJOS FUTUROS

Posibles mejoras

Proceso de entrenamiento más preciso y menos aleatorio, que permita la selección de las instancias más descriptivas en los elementos a incluir en la base de datos.

Ajuste automático a través de iteraciones múltiples del mejor valor para cada uno de los parámetros en la integración de características

Etapas de filtrado o ajuste (en software, no en el montaje físico y en la reglas de captura de información) del dataset obtenido, donde se reduzca las perturbaciones generadas por los problemas más evidentes en la toma de datos, que puedan confundir al sistema o que le generen ambigüedades innecesarias en el proceso de clasificación.

Evaluación de mayor variedad de técnicas de agrupación de descriptores locales, que permitan una consolidación de un modelo de respuesta aún más preciso.

A pesar de que son calculadas, aún no se han integrado las características estructurales 2D al módulo de clasificación visual. Es probable que, tomando en cuenta estas restricciones al momento de la clasificación, el desempeño del módulo aumente ostensiblemente.

Posibles proyectos

Realizar la integración total de los dos módulos desarrollados en este trabajo, mediante una calibración extrínseca de los modelos de los sensores, y diseñando un esquema de descripción, clasificación de la información combinada y no independiente. Esto también abre la puerta al uso de otros tipos de sensores presentes en la actualidad, que entregan la información ya organizada de matrices de imagen y de profundidad, por lo que solo sería necesario la concepción de un esquema de clasificación de la información integrada.

Hacer uso de los esquemas diseñados en este trabajo para cada módulo, en la detección, y reconocimiento de otro tipo de elementos en diferentes áreas de trabajo, ya sea en el área de la automatización industrial, o en el área de la inteligencia computacional y la robótica, mediante la percepción inteligente de cada uno de los elementos en el entorno.

6 REFERENCIAS

- Abdelbaki, H. M., Hussain, K., Member, S., & Gelenbe, E. (2001). A Laser Intensity Image Based Automatic Vehicle Classification System, 462–467.
- About - Point Cloud Library (PCL). (2013). Retrieved from <http://pointclouds.org/about/>
- Adelanto Boletín de Prensa Comité Automotor Colombiano. (2012), 1.
- Agarwal, S., Awan, A., & Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. *Pattern Analysis and Machine ...*, 26(11), 1475–90. doi:10.1109/TPAMI.2004.108
- Alahi, a., Ortiz, R., & Vandergheynst, P. (2012). FREAK: Fast Retina Keypoint. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 510–517. doi:10.1109/CVPR.2012.6247715
- Aldoma, A., Tombari, F., Rusu, R. B., & Vincze, M. (2012). OUR-CVFH – Oriented , Unique and Repeatable Clustered Viewpoint Feature Histogram for Object Recognition and 6DOF Pose Estimation, 113–122.
- Aldoma, A., Vincze, M., Blodow, N., Gossow, D., Gedikli, S., Rusu, R. B., & Bradski, G. (2011). CAD-model recognition and 6DOF pose estimation using 3D cues. *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 585–592.
- Aly, M., Munich, M., & Perona, P. (2011). Indexing in large scale image collections: Scaling properties and benchmark. *2011 IEEE Workshop on Applications of Computer Vision (WACV)*, 418–425. doi:10.1109/WACV.2011.5711534
- ANN - Approximate Nearest Neighbor Library. (2013). Retrieved from <http://www.cs.umd.edu/~mount/ANN/>
- Azhar, H., & Amer, a. (2012). Classification of surveillance video objects using chaotic series. *IET Image Processing*, 6(7), 919. doi:10.1049/iet-ipr.2011.0269
- Badino, H., Huber, D., Park, Y., & Kanade, T. (2011). Fast and accurate computation of surface normals from range images. *2011 IEEE International Conference on Robotics and Automation*, 3084–3091. doi:10.1109/ICRA.2011.5980275
- Bay, H., Ess, A., Tuytelaars, T., & Gool, L. Van. (2008). Speeded-up robust features (SURF). *Computer vision and image ...*, (September). Retrieved from <http://www.sciencedirect.com/science/article/pii/S1077314207001555>
- Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9), 509–517. doi:10.1145/361002.361007
- Bielicki, J., & Sitnik, R. (2013). A method of 3D object recognition and localization in a cloud of points. *EURASIP Journal on Advances in Signal Processing*, 2013(1), 29. doi:10.1186/1687-6180-2013-29

- Boost C++ Libraries. (2013). Retrieved from <http://www.boost.org/>
- Borenstein, J. (2002). Characterization of a 2D laser scanner for mobile robot obstacle negotiation. *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, 3(May), 2512–2518. doi:10.1109/ROBOT.2002.1013609
- Boult, T. E., Micheals, R. J., & Eckmann, M. (2001). Into the woods: visual surveillance of noncooperative and camouflaged targets in complex outdoor settings. *Proceedings of the IEEE*, 89(10), 1382–1402. doi:10.1109/5.959337
- Buch, N. E. (2010). Classification of vehicles for urban traffic scenes. Retrieved from http://dilnxsrv.king.ac.uk/norbert/main/papers/thesis_buch_low_resolution.pdf
- Buch, N., Orwell, J., & Velastin, S. a. (2008). Detection and classification of vehicles for urban traffic scenes. *Visual Information Engineering, ...*, 182–187. doi:10.1049/cp:20080305
- Buch, N., Orwell, J., & Velastin, S. a. (2010). Urban road user detection and classification using 3D wire frame models. *IET Computer Vision*, 4(2), 105. doi:10.1049/iet-cvi.2008.0089
- Buch, N., Velastin, S. a., & Orwell, J. (2011). A Review of Computer Vision Techniques for the Analysis of Urban Traffic. *IEEE Transactions on Intelligent Transportation Systems*, 12(3), 920–939. doi:10.1109/TITS.2011.2119372
- Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). Brief: Binary robust independent elementary features. *Computer Vision–ECCV 2010*. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-15561-1_56
- Cathey, F. W., & Dailey, D. J. (2005). A novel technique to dynamically measure vehicle speed using uncalibrated roadway cameras. *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, (2), 777–782. doi:10.1109/IVS.2005.1505199
- Chang, M.-C., Leymarie, F. F., & Kimia, B. B. (2009). Surface reconstruction from point clouds by transforming the medial scaffold. *Computer Vision and Image Understanding*, 113(11), 1130–1146. doi:10.1016/j.cviu.2009.04.001
- Chen, H., & Bhanu, B. (2007). 3D free-form object recognition in range images using local surface patches. *Pattern Recognition Letters*, 28(10), 1252–1262. doi:10.1016/j.patrec.2007.02.009
- Chen, L., Nguyen, X., & Lin, S. (2012). \$ XWRPDWLF REMHFW GHWHFWLRQ HPSOR \ LQJ YLHZLQJ DQJOH KLVWRJUDP IRU UDQJH LPDJHV\$, 196–201.
- Ciocarlie, M., Hsiao, K., Jones, E. G., Chitta, S., Rusu, R. B., & Ioan, A. S. (n.d.). Towards Reliable Grasping and Manipulation in Household Environments, 1–12.
- Cobro electrónico de peajes sigue creciendo en Colombia — Noticias Telemedellín. (n.d.). Retrieved from <http://noticias.telemedellin.tv/cobro-electronico-de-peajessigue-creciendo-en-colombia/>

- Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. ... *and Pattern Recognition, 2005. CVPR 2005* Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1467360
- Derenick, J. C., & Miller, T. H. (2008). *The Sick LIDAR Matlab / C ++ Toolbox : A Quick Start Guide*, 1–32.
- Eigen. (2013). Retrieved from http://eigen.tuxfamily.org/index.php?title=Main_Page
- Elibol, A., Gracias, N., Garcia, R., Gleason, A., & Gintert, B. (2011). Efficient autonomous image mosaicing with applications to coral reef monitoring. Retrieved from <http://yyy.rsmas.miami.edu/groups/reidlab/publications/Elibol2011.pdf>
- García, L. A., & Prieto, F. A. (2011). Adquisición de entornos tridimensionales empleando un Captor laser bidimensional.
- GCC, the GNU Compiler Collection - GNU Project - Free Software Foundation (FSF). (2013). Retrieved from <http://gcc.gnu.org/>
- Goron, L. C., Tamas, L., Reti, I., & Lazea, G. (2010). 3D Laser scanning system and 3D segmentation of urban scenes. *2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 1–5. doi:10.1109/AQTR.2010.5520872
- Grewal, M. S., & Andrews, A. P. (2008). *Kalman Filtering: Theory and Practice Using Matlab. Vasa* (3rd ed.). Retrieved from <http://medcontent.metapress.com/index/A65RM03P4874243N.pdf>
- Grimson, W. E. L., & Ma, X. (2005). Edge-based rich representation for vehicle classification. *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, 1185–1192 Vol. 2. doi:10.1109/ICCV.2005.80
- Guarnieri, A., Pirotti, F., Pontin, M., & Vettore, A. (2006). 3D Surveying for Structural Analysis Applications. *3rd IAG / 12th FIG Symposium, Baden*.
- Gupte, S., & Masoud, O. (2002). Detection and classification of vehicles. *Intelligent ...*, 3(1), 37–47. doi:10.1109/6979.994794
- Hall, D., & Nascimento, J. (2005). Comparison of target detection algorithms using adaptive background models. ... *of Tracking and ...*, (October). Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1570905
- Haritaoglu, I., Harwood, D., & Davis, L. S. (1998). W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People. *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231)*, 962–962. doi:10.1109/CVPR.1998.698720
- Harlow, C., & Peng, S. (2001). Automatic vehicle classification system with range sensors, 9, 231–247.

- Hartmann, J., & Forouher, D. (2012). Real-time visual SLAM using FastSLAM and the Microsoft Kinect Camera. ... of *ROBOTIK 2012*; ..., 458–463. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6309551
- Hegde, G. M., & Ye, C. (2011). A recursive planar feature extraction method for 3D range data segmentation. *2011 IEEE International Conference on Systems, Man, and Cybernetics*, 3119–3124. doi:10.1109/ICSMC.2011.6084139
- Hinterstoisser, S., Cagniard, C., Ilic, S., Sturm, P., Navab, N., Fua, P., & Lepetit, V. (2012). Gradient response maps for real-time detection of textureless objects. *IEEE transactions on pattern analysis and machine intelligence*, 34(5), 876–88. doi:10.1109/TPAMI.2011.206
- Hinterstoisser, S., Holzer, S., Cagniard, C., Ilic, S., Konolige, K., Navab, N., & Lepetit, V. (2011). Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. *2011 International Conference on Computer Vision*, 858–865. doi:10.1109/ICCV.2011.6126326
- Holzer, S., Rusu, R. B., Dixon, M., Gedikli, S., & Navab, N. (2012). Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2684–2689. doi:10.1109/IROS.2012.6385999
- Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., & Stuetzle, W. (1992). Surface reconstruction from unorganized points. *Proceedings of the 19th annual conference on Computer graphics and interactive techniques - SIGGRAPH '92*, 71–78. doi:10.1145/133994.134011
- Hsieh, J.-W., Yu, S.-H., Chen, Y.-S., & Hu, W.-F. (2006). Automatic Traffic Surveillance System for Vehicle Tracking and Classification. *IEEE Transactions on Intelligent Transportation Systems*, 7(2), 175–187. doi:10.1109/TITS.2006.874722
- Huang, D., Chen, C., & Hu, W. (2012). Feature-based vehicle flow analysis and measurement for a real-time traffic surveillance system. *Journal of Information ...*, 3(3), 282–296. Retrieved from <http://bit.kuas.edu.tw/~jihmsp/2012/vol3/JIH-MSP-2012-03-005.pdf>
- Ji, J., Qi, X., Han, X., & Yuan, Z. (2009). An Approach of Automatic Vehicle Classification by Acoustic Wave Based on PCA-RBF. *2009 International Conference on Information Engineering and Computer Science*, (1), 1–4. doi:10.1109/ICIECS.2009.5364602
- Johansson, B., Wiklund, J., Forssén, P.-E., & Granlund, G. (2009). Combining shadow detection and simulation for estimation of vehicle size and position. *Pattern Recognition Letters*, 30(8), 751–759. doi:10.1016/j.patrec.2009.03.005
- Johnson, A. E. (1997). Spin-Images : A Representation for 3-D Surface Matching.
- Kamiya, Y., Takahashi, T., Ide, I., & Murase, H. (2010). A multimodal constellation model for object image classification. ... *Image and Video Processing, 2010*, 1–10. doi:10.1155/2010/426781

- Klasing, K., Althoff, D., Wollherr, D., & Buss, M. (2009). Comparison of surface normal estimation methods for range sensing applications. *2009 IEEE International Conference on Robotics and Automation*, 3206–3211. doi:10.1109/ROBOT.2009.5152493
- Klasing, K., Wollherr, D., & Buss, M. (2009). Realtime segmentation of range data using continuous nearest neighbors. *2009 IEEE International Conference on Robotics and Automation*, 2431–2436. doi:10.1109/ROBOT.2009.5152498
- Klasing, Klaas, Wollherr, D., & Buss, M. (2008). A clustering method for efficient segmentation of 3D laser data. *2008 IEEE International Conference on Robotics and Automation*, 4043–4048. doi:10.1109/ROBOT.2008.4543832
- Kleeman, L. (1996). Understanding and applying Kalman filtering. *Second Workshop on "Perceptive Systems", Curtin ...* Retrieved from http://www.cs.cmu.edu/~motionplanning/papers/sbp_papers/integrated3/kleeman_kalman_basics.pdf
- Knopp, J., Prasad, M., & Willems, G. (2010). Hough transform and 3D SURF for robust three dimensional classification. *Computer Vision–ECCV ...*, 1–14. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-15567-3_43
- König, S., & Gumhold, S. (n.d.). Consistent Propagation of Normal Orientations in Point Clouds.
- Leutenegger, S., Chli, M., & Siegwart, R. Y. (2011). BRISK: Binary Robust invariant scalable keypoints. *2011 International Conference on Computer Vision*, 2548–2555. doi:10.1109/ICCV.2011.6126542
- Lowe, D.G. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1150–1157 vol.2. doi:10.1109/ICCV.1999.790410
- Lowe, David G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2), 91–110. doi:10.1023/B:VISI.0000029664.99615.94
- Messelodi, S., Modena, C. M., & Zanin, M. (2004). A computer vision system for the detection and classification of vehicles at urban road intersections, 1–38.
- Mitra, N. J., Nguyen, A., & Guibas, L. (2003). Estimating Surface Normals in Noisy Point Cloud Data. *Proceedings of the nineteenth conference on Computational geometry - SCG '03, 2003*, 322. doi:10.1145/777837.777840
- Mount, D. M. (2010). ANN Programming Manual.
- MPlayer - The Movie Player. (2013). Retrieved from <http://www.mplayerhq.hu/design7/info.html>
- Ng, J. Y., & Tay, Y. H. (2011). Image-based Vehicle Classification System, 1–11.

- Nüchter, A., & Hertzberg, J. (2008). Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11), 915–926. doi:10.1016/j.robot.2008.08.001
- OpenCV. (2013). Retrieved from <http://opencv.org/>
- OpenMP.org » About the OpenMP ARB and OpenMP.org. (2013). Retrieved from <http://openmp.org/wp/about-openmp/>
- Papazov, C., & Burschka, D. (n.d.). An Efficient RANSAC for 3D Object Recognition in Noisy and Occluded Scenes, (ii).
- Perona, P. (2010). Indexing in Large Scale Image Collections : Scaling Properties , Parameter Tuning , and Benchmark, 1–35.
- Ramezani, M., & Ebrahimnezhad, H. (2011). 3D Object Categorization Based on Histogram of Distance and Normal Vector Angles on Surface Points. *2011 7th Iranian Conference on Machine Vision and Image Processing*, 1–5. doi:10.1109/IranianMVIP.2011.6121545
- Refai, H. (2009). Accurate Vehicle Classification Including Motorcycles using Piezoelectric Sensors. Retrieved from <http://rip.trb.org/view/2009/P/1232374>
- Reid, I., & Term, H. (2001). Estimation II Discrete-time Kalman filter, 1–44.
- Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. *Computer Vision–ECCV 2006*, 1–14. Retrieved from http://link.springer.com/chapter/10.1007/11744023_34
- Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. *2011 International Conference on Computer Vision*, 2564–2571. doi:10.1109/ICCV.2011.6126544
- Rusu, R B, Holzbach, a, Beetz, M., & Bradski, G. (2009). Detecting and segmenting objects for mobile manipulation. *Iccvw*, 47–54.
- Rusu, R. (2010). Semantic 3D object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*.
- Rusu, R.B., Bradski, G., Thibaux, R., & Hsu, J. (2010). Fast 3D recognition and pose using the Viewpoint Feature Histogram. *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*.
- Rusu, R.B., Marton, Z. C., Blodow, N., & Beetz, M. (2008). Learning informative point classes for the acquisition of object model maps. *2008 10th International Conference on Control, Automation, Robotics and Vision*.
- Rusu, Radu Bogdan, Blodow, N., & Beetz, M. (2009). Fast Point Feature Histograms (FPFH) for 3D registration. *2009 IEEE International Conference on Robotics and Automation*, 3212–3217. doi:10.1109/ROBOT.2009.5152473

- Rusu, Radu Bogdan, Blodow, N., Marton, Z. C., & Beetz, M. (n.d.). Close-range Scene Segmentation and Reconstruction of 3D Point Cloud Maps for Mobile Manipulation in Human Environments.
- Rusu, Radu Bogdan, & Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). *2011 IEEE International Conference on Robotics and Automation*, 1–4. doi:10.1109/ICRA.2011.5980567
- Rusu, Radu Bogdan, Marton, Z. C., Blodow, N., & Beetz, M. (n.d.). Persistent Point Feature Histograms for 3D Point Clouds.
- Rusu, Radu Bogdan, Marton, Z. C., Blodow, N., Dolha, M., & Beetz, M. (2008). Towards 3D Point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11), 927–941. doi:10.1016/j.robot.2008.08.005
- Sareen, K. K., Knopf, G. K., & Canas, R. (2010). Rapid clustering of colored 3D point cloud data for reconstructing building interiors. *2010 International Symposium on Optomechatronic Technologies*, (c), 1–6. doi:10.1109/ISOT.2010.5687331
- Scholz, D., & Koy-Oberthür, R. (2006). TOLLCHECKER : SYSTEM DESIGN FOR HIGH PERFORMANCE TOLL ENFORCEMENT. *PROCEEDINGS OF THE 13th ITS WORLD CONGRESS, LONDON, 8-12 OCTOBER 2006*, 1–7.
- Shi, J., & Tomasi, C. (1994). Good features to track. ... , 1994. *Proceedings CVPR'94., 1994 IEEE ...*, 593–600. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=323794
- Silverman, B. W., & Jones, M. C. (1989). E. Fix and J. L Hodges (1951): An Important Contribution to Nonparametric Discriminant Analysis and Density Estimation: Commentary on Fix and Hodges (1951). *International Statistical Review*, 57, 233–238. Retrieved from <http://www.jstor.org/stable/1403796>
- Sivic, J., & Zisserman, A. (2003). Video Google: a text retrieval approach to object matching in videos. *Proceedings Ninth IEEE International Conference on Computer Vision*, (Iccv), 1470–1477 vol.2. doi:10.1109/ICCV.2003.1238663
- Song, X., & Nevatia, R. (2007). Detection and tracking of moving vehicles in crowded scenes. ... and Video Computing, 2007. *WMVC'07. ...*, (Mcmc). Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4118800
- Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, 246–252. doi:10.1109/CVPR.1999.784637
- Steder, B. (n.d.). The Point Cloud Library PCL.
- Steder, B., & Konolige, K. (n.d.). NARF : 3D Range Image Features for Object Recognition.
- Steder, B., & Ruhnke, M. (2011). Place recognition in 3D scans using a combination of bag of words and point feature based relative pose estimation. *Intelligent Robots and ...*

- Steder, B., & Rusu, R. (2011). Point feature extraction on 3D range scans taking into account object boundaries. ... *and Automation (ICRA)*, ..., 2601–2608.
- Stoustrup, B. (1997). The C++ programming language.
- Strom, J., Richardson, a, & Olson, E. (2010). Graph-based segmentation for colored 3D laser point clouds. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2131–2136. doi:10.1109/IROS.2010.5650459
- Tamersoy, B. (2009). Background Subtraction.
- Tanaka, T. (2007). A fast algorithm for adaptive background model construction using parzen density estimation. *2007 IEEE Conference on Advanced Video and Signal Based Surveillance*, 528–533. doi:10.1109/AVSS.2007.4425366
- Teichman, A., Levinson, J., & Thrun, S. (2011). Towards 3D object recognition via classification of arbitrary object tracks. *2011 IEEE International Conference on Robotics and Automation*, 4034–4041. doi:10.1109/ICRA.2011.5979636
- Tok, Y. (2008). Commercial Vehicle Classification System using Advanced Inductive Loop Technology, (161). Retrieved from <http://escholarship.org/uc/item/92x23786.pdf>
- Toldo, R., Castellani, U., & Fusiello, A. (2009). A bag of words approach for 3d object categorization. *Computer Vision/Computer Graphics ...*, 116–127.
- Tombari, F., & Stefano, L. Di. (2010). Object Recognition in 3D Scenes with Occlusions and Clutter by Hough Voting. *Image and Video Technology (PSIVT), 2010 Fourth Pacific-Rim Symposium on*, 349–355. doi:10.1109/PSIVT.2010.65
- Tombari, Federico, Salti, S., & Stefano, L. Di. (2010). Unique Signatures of Histograms for Local Surface Description, 347–360.
- Valverde Pedraza, C. felipe, & Ossa Torres, J. D. (2011). Detección, clasificación y conteo de vehículos en vías por medio de visión artificial. *Trabajo de rado Ingeniería electrónica*.
- Velizhev, A. (2012). Implicit shape models for object detection in 3D point clouds. *International ...*, 1.
- Velizhev, Alexander. (2012). Implicit shape models for object detection in 3D point clouds. *International ...*
- Weingarten, J., & Siegwart, R. (2005). EKF-based 3D SLAM for structured environment reconstruction. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3834–3839. doi:10.1109/IROS.2005.1545285
- Weingarten, J. W., Gruener, G., & Siegwart, R. (2004). Probabilistic plane fitting in 3D and an application to robotic mapping. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 927–932 Vol.1. doi:10.1109/ROBOT.2004.1307268

- Wohlkinger, W., & Vincze, M. (2011). Ensemble of shape functions for 3D object classification. *2011 IEEE International Conference on Robotics and Biomimetics*, 2987–2992. doi:10.1109/ROBIO.2011.6181760
- Xiao, J., Zhang, J., Zhang, J., Zhang, H., & Hildre, H. P. (2011). Fast plane detection for SLAM from noisy range images in both structured and unstructured environments. *2011 IEEE International Conference on Mechatronics and Automation*, 1768–1773. doi:10.1109/ICMA.2011.5986247
- Zhan, Q., & Liang, Y. (2010). A point cloud segmentation method based on vector estimation and color clustering. *The 2nd International Conference on Information Science and Engineering*, 3463–3466. doi:10.1109/ICISE.2010.5691038
- Zhana, Q., Liangb, Y., & Xiaoa, Y. (2009). Color-based segmentation of point clouds, *XXXVIII*, 248–252.
- Zhang, G., Author, C., & Avery, R. P. (2007). A Video-based Vehicle Detection and Classification System for Real-time Traffic Data Collection Using Uncalibrated Video Cameras.
- Zhang, J. X., & Lin, X. G. (2012). SVMs-BASED CLASSIFICATION OF SEGMENTED AIRBORNE LIDAR POINT CLOUDS IN URBAN AREAS, 1–6.
- Zheng, D., Xu, J., & Ren-xi, C. (2009). Generation method of normal vector from disordered point cloud. *2009 Joint Urban Remote Sensing Event*, 1–5. doi:10.1109/URS.2009.5137560
- Zheng, J., & Wang, Y. (2006). Extracting roadway background image: Mode-based approach. ... *Record: Journal of the ...*, (206), 1–15. Retrieved from <http://trb.metapress.com/index/D27XM7W58728G3H6.pdf>

7 ANEXOS

7.1 MÓDULO DE CLASIFICACIÓN DE VEHÍCULOS CON INFORMACIÓN VISUAL

7.1.1 DESCRIPTORES VISUALES

7.1.1.1.1 Puntos representativos y descriptores SIFT

Los puntos representativos SIFT son obtenidos como mínimos y máximo locales en las imágenes resultantes del método de *DoG* (*Difference of Gaussians*, diferencia de gaussianas) para distintas escalas de la imagen. El escalamiento en las imágenes se hace por medio de la convolución de la imagen original y un operador gaussiano de tamaño σ , posteriormente se realiza un remuestreo de la imagen y se aplica una operación idéntica. Cada remuestreo define un *octave* y para cada *octave* se realizan igual número de difuminados con el operador gaussiano a diferentes escalas (ver Figura 7.1). Un punto se elige como representativo, si y solo si, es mayor o menor a todos sus 8 vecinos en la escala en que se está evaluando, a sus 9 vecinos de la escala superior y a sus 9 vecinos de la escala inferior. Esto se muestra en la Figura 7.2

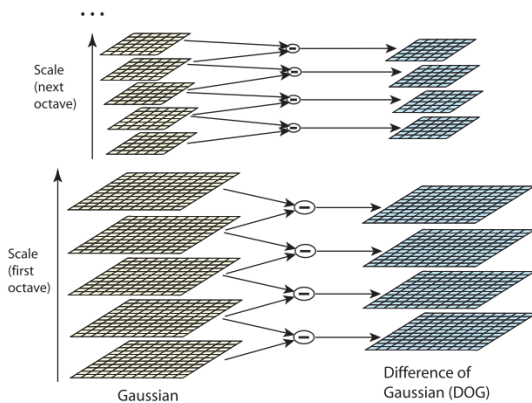


Figura 7.1 División en octaves y Diferencia de Gaussianas para puntos representativos SIFT (David G. Lowe, 2004)

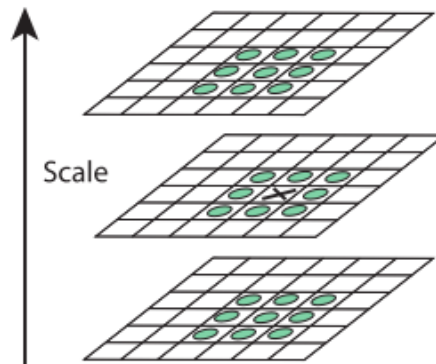


Figura 7.2 Evaluación de máximo local para puntos representativos SIFT (David G. Lowe, 2004)

A cada punto de interés se le asigna una o más orientaciones (caso para el cual se crean copias del punto de interés) basadas en las orientaciones de gradientes locales al punto.

Para cada punto de interés se calcula un vector de características de 128 componentes compuesto de un arreglo de tamaño 4x4 que contiene histogramas de las magnitudes y orientaciones de los gradientes en una vecindad de 16x16 pixels alrededor del punto de interés, donde cada histograma de orientaciones contiene 8 direcciones. El vector es normalizado a la unidad para reducir el efecto de cambios en la iluminación. Posteriormente se le aplica un umbral superior de 0.2 y se normaliza de nuevo a la unidad para reducir el efecto de cambios no lineales en la iluminación.

7.1.1.1.2 Puntos representativos y descriptores SURF

Para la obtención de puntos representativos SURF (Bay, Ess, Tuytelaars, & Gool, 2008) se realiza una representación en espacio de escalas, de forma similar a SIFT. Sin embargo, en lugar de utilizar el método de *DoG*, se utiliza el determinante de la matriz hessiana mediante una aproximación lineal de las derivadas parciales gaussianas de segundo orden para formar un mapa de respuestas tipo *blob* en diferentes escalas (aumentando el tamaño del filtro gaussiano). De igual forma que en SIFT, el espacio de escalas es dividido en *octaves* y en cada *octave* se realizan igual número de convoluciones con operadores gaussianos a diferentes escalas. La Figura 7.3 da una visión general de los tamaños de los filtros para los 3 primeros *octaves*. Este cálculo se realiza de manera eficiente haciendo uso de la imagen integral y aproximando el determinante de la matriz hessiana en un punto mediante la Ecuación 7.1

$$\det(H_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2 \quad \text{Ecuación 7.1}$$

Donde D_{xx} , D_{yy} y D_{xy} son las aproximaciones de las derivadas parciales gaussianas de segundo orden en la dirección x , y y xy , respectivamente.

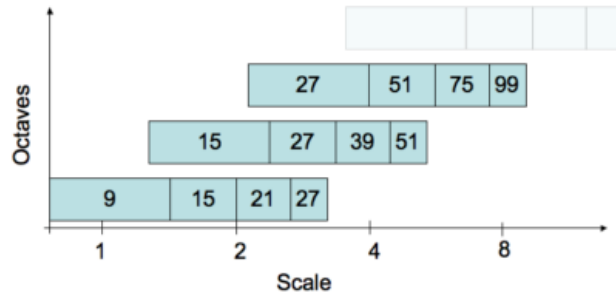


Figura 7.3 Tamaños de los filtros gaussianos en los tres primeros octaves para puntos representativos SURF (Bay et al., 2008)

Para localizar puntos de interés en la imagen y entre las escalas, se aplica una supresión de no-máximos en una vecindad de $3 \times 3 \times 3$.

Una vez localizados los puntos de interés, se identifica una orientación que sea reproducible para los mismos. Para esto, primero se calculan las respuestas Haar wavelet en las direcciones x e y al interior de una vecindad circular de radio $6s$ alrededor del punto, donde s es la escala en la cual el punto fue detectado. Para calcular las respuestas wavelet se puede hacer uso, una vez más, de la imagen integral. Después de calcularse las respuestas wavelet y ser ponderadas con una función gaussiana ($\sigma = 2s$) centrada en el punto de interés, estas son representadas como puntos en un espacio con la respuesta horizontal sobre el eje de las abscisas y la respuesta vertical sobre el eje de las ordenadas. La orientación dominante es estimada calculando la suma de todas las respuestas al interior de una ventana deslizante de orientación de tamaño $\frac{\pi}{3}$.

Luego de determinar la orientación de cada punto, se procede a construir una región cuadrada centrada en el punto y orientada a lo largo de esta dirección, con tamaño $20s$, tal como se observa en la Figura 7.4. Esta región cuadrada es dividida en 16 subregiones cuadradas (4 filas x 4 columnas) en cada una de las cuales se calculan las respuestas Haar wavelet (con tamaño del filtro $2s$) en la dirección horizontal x y vertical y (con respecto a la

orientación del punto). Estas respuestas se ponderan con una función gaussiana ($\sigma = 3.3s$) centrada en el punto de interés y posteriormente se obtienen las siguientes características:

- $\sum d_x$ Suma de las respuestas Haar wavelet en la dirección horizontal.
- $\sum d_y$ Suma de las respuestas Haar wavelet en la dirección vertical.
- $\sum |d_x|$ Suma de los valores absolutos de las respuestas Haar wavelet en la dirección horizontal.
- $\sum |d_y|$ Suma de los valores absolutos de las respuestas Haar wavelet en la dirección vertical.

De esta manera se construye un vector descriptor de 64 componentes para cada punto de interés SURF.



Figura 7.4 Regiones cuadradas corregidas con orientación para puntos representativos SURF (Bay et al., 2008)

7.1.1.1.3 Puntos representativos FAST

FAST (*Features from Accelerated Segment Test*) es un método de detección de esquinas, que puede ser usado en la tarea de extracción de puntos representativos, al igual que para realizar seguimiento y mapeo de objetos en varias imágenes o en video. La ventaja más importante de FAST es su eficiencia computacional. FAST es más rápido que varios métodos populares de extracción de características, como la *diferencia de Gaussianas* usado por SIFT, SUSAN y Harris, por ejemplo. El detector de esquinas FAST fue desarrollado por Edward Rosten y Tom Drummond (Rosten & Drummond, 2006).

El algoritmo utiliza, básicamente, un círculo de 16 pixels (un círculo Bresenham de radio 3) para clasificar si un punto p es una esquina. Se le da a cada píxel en el círculo una etiqueta con un número entre 1 y 16, en el sentido de las manecillas del reloj. Si un set de N pixels contiguos en el círculo son más brillantes que la intensidad del pixel candidato, I_p , más un umbral t , o son más oscuros que la intensidad del pixel candidato menos el valor umbral t , entonces el pixel p es clasificado como una esquina. Normalmente N es 12.

7.1.1.1.4 Descriptores BRIEF

BRIEF es un método de construcción de descriptores que apunta principalmente a la eficiencia computacional de los mismos. Este objetivo se logra construyendo un vector de

características binario en el que cada componente es el resultado de un test de intensidades hecho entre parejas de pixels, obtenidos aleatoriamente, al interior de un parche centrado en el punto de interés en una versión suavizada de la imagen. Un test τ en un parche $\mathbf{p}$ de tamaño $S \times S$ se define como:

$$\tau(\mathbf{p}; x, y) := \begin{cases} 1 & \text{si } \mathbf{p}(x) < \mathbf{p}(y) \\ 0 & \text{en otro caso} \end{cases} \quad \text{Ecuación 7.2}$$

En donde $\mathbf{p}(x)$ y $\mathbf{p}(y)$ son las intensidades de los pixels en un par de puntos x y y al interior del parche $\mathbf{p}$ de la imagen suavizada.

El documento seminal muestra diferentes resultados de rendimiento para tamaños del vector de características de 128, 256 y 512 bits. Estas versiones son llamadas BRIEF-16, BRIEF-32 y BRIEF-64, respectivamente, refiriéndose a la cantidad de bytes necesarios para almacenar cada descriptor.

7.1.1.1.5 Puntos representativos y descriptores BRISK

BRISK (Leutenegger, Chli, & Siegwart, 2011) es un algoritmo de obtención de puntos representativos y descriptores visuales. Para la obtención de los puntos representativos BRISK se hace uso del detector de esquinas FAST tanto en el plano de la imagen como en un espacio de escalas dividido en n octaves y n suboctaves, con n normalmente 4.

Inicialmente, se aplica un detector FAST 9-16 (9 pixels contiguos en el círculo de 16 pixels deben ser suficientemente brillantes u oscuros) con un umbral t constante para identificar potenciales regiones de interés. Posteriormente, se realiza un proceso de supresión de no máximos: primero, el punto debe cumplir la condición de máximo con respecto a sus 8 puntuaciones FAST s vecinas; segundo, las puntuaciones en las capas superior e inferior deben ser menores también. La puntuación s se define como el umbral máximo para el cual el punto en cuestión sigue siendo considerado una esquina.

Los descriptores visuales BRISK se basan en la evaluación de intensidades en un patrón usado para muestrear la vecindad del punto representativo. El patrón, ilustrado en la XXX, define N ubicaciones igualmente espaciadas en círculos concéntricos con el punto representativo. A cada punto p_i en el patrón se le aplica un suavizado gaussiano con desviación estándar σ_i , representada por los círculos rojos en la XXX, proporcional a la distancia entre puntos en el círculo concéntrico. De todas las parejas de puntos posibles (p_i , p_j) en el patrón, se saca un subconjunto de puntos con distancias cortas S y puntos con distancias largas L de la siguiente manera:

$$S = \{(p_i, p_j) \mid \|p_j - p_i\| < d_{max}\} \quad \text{Ecuación 7.3}$$

$$L = \{(p_i, p_j) \mid \|p_j - p_i\| < d_{max}\} \quad \text{Ecuación 7.4}$$

La dirección característica del punto representativo se obtiene mediante el gradiente de los puntos que pertenecen a L . Mientras que la cadena binaria que representa al punto se obtiene mediante la comparación de las intensidades de las parejas de puntos que pertenecen a S , rotados de acuerdo a la dirección del punto representativo. Finalmente se obtiene un vector descriptor con 512 componentes binarios.

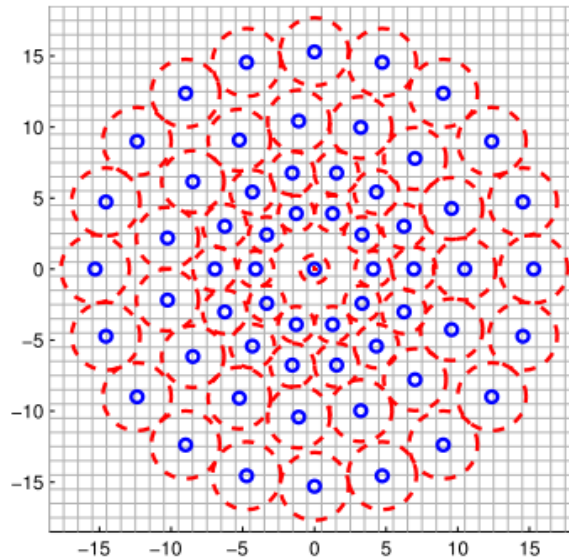


Figura 7.5 Patrón de muestreo de la vecindad con $N=60$ para un punto representativo BRISK. Los círculos azules representan las ubicaciones de muestreo y los círculos rojos en línea discontinua tienen un radio correspondiente a la desviación estándar del núcleo gaussiano utilizado para el suavizado. (Leutenegger et al., 2011)

7.1.1.1.6 Puntos representativos y descriptores ORB

Los puntos representativos y descriptores ORB (Rublee, Rabaud, Konolige, & Bradski, 2011) hacen uso tanto del detector de esquinas FAST, como de los descriptores BRIEF, ambos modificados para tener en cuenta el componente de rotación.

Para la obtención de N puntos característicos, primero se ajusta el umbral del algoritmo, de manera que se obtengan más de N esquinas. Posteriormente se ordenan estos puntos de acuerdo a la medida de esquinas Harris y se eligen los N puntos superiores. A los puntos obtenidos se les agrega un componente de orientación por medio de la técnica del *centroide de intensidad*, que asume que la información de intensidad de una esquina está corrida de su centro. Se calcula el centroide en una región circular de radio r alrededor de la esquina mediante el cómputo de los momentos del parche. Luego, se construye un vector entre el centro del parche y su centroide, que determina la orientación del punto representativo.

La construcción del descriptor se realiza en un parche alrededor del punto representativo en el que se toman n test binarios entre parejas de puntos con una distribución gaussiana aleatoria. Para agregar el componente de orientación, las parejas de puntos x_i y y_i son rotadas mediante una matriz de rotación R_θ usando la orientación del punto representativo θ . Este procedimiento disminuye significativamente el rendimiento con relación a la varianza y correlación de los puntos. Para superar esta limitación, se eligen los test binarios que tengan una media más cercana a 0.5 (mayor varianza posible para una variable binaria) y que tengan una correlación no mayor a un umbral establecido, para cada punto representativo.

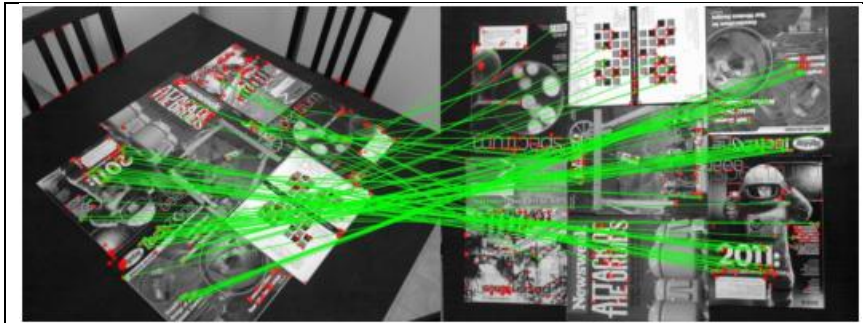


Figura 7.6 Mapeo de puntos representativos ORB en una imagen con modificaciones en escala, rotación y punto de vista. Las líneas verdes son correspondencias válidas; los círculos rojos indican puntos sin correspondencia. (Rublee et al., 2011)

7.1.1.1.7 Descriptores FREAK

Para la obtención de descriptores visuales, FREAK (Alahi, Ortiz, & Vanderghenst, 2012) utiliza un patrón de muestreo circular (similar a BRISK), con la diferencia de que tiene una mayor densidad de puntos cerca del centro (punto representativo descrito). Para reducir la sensibilidad al ruido cada punto de muestra se suaviza con un operador gaussiano con tamaño de núcleo diferente, similar a BRISK, con la diferencia de que el cambio de tamaño en el núcleo es exponencial y no lineal y que se presenta superposición de los campos receptivos, tal como se observa en la Figura 7.7

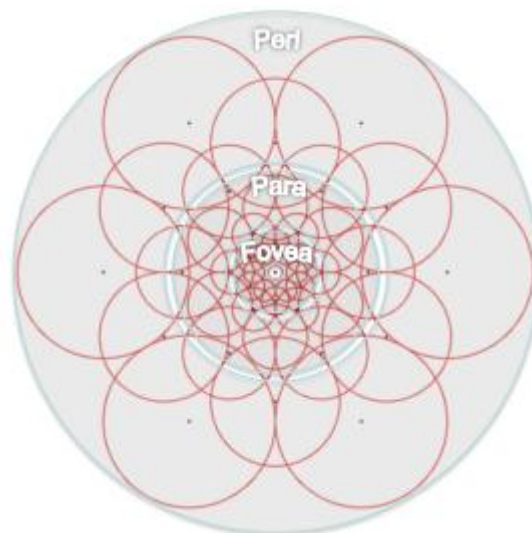


Figura 7.7 Patrón de muestreo de la vecindad de un punto representativo para los descriptores visuales FREAK. Los círculos rojos representan los campos receptivos, en donde la imagen se suaviza con su respectivo operador gaussiano. (Alahi et al., 2012)

El descriptor binario se construye umbralizando la diferencia en intensidades entre pares de campos receptivos suavizados por el operador gaussiano. Debido a que una pequeña cantidad de campos receptivos pueden producir varios miles de parejas posibles a comparar, se lleva a cabo un proceso de selección de pares similar al seguido en ORB, seleccionando

en primera instancia el test binario con la media más cercana a 0.5 (mayor varianza posible en una distribución binaria) y, a partir de este, se eligen los test binarios que presenten menos correlación hasta completar una cadena binaria de 512 bits. Este procedimiento se llevó a cabo para conocer las mejores parejas de campos receptivos a partir de los datos de entrenamiento, y se obtuvo como resultado una estructura de grueso a fino en los pares seleccionados (de la periferia al centro del patrón de muestreo).

7.1.2 ÁRBOLES DE RESULTADOS

En los siguientes árboles de resultados se pueden visualizar los parámetros libres modificados en cada caso, de acuerdo al nivel (niveles mostrados en los rectángulos anaranjados); los valores asignados a los parámetros, mostrados en blanco en los óvalos azules; y los porcentajes de clasificación correcta promedio total de sistema, mostrados abajo en los óvalos azules, en negro y en amarillo el de mejor desempeño.

Para el descriptor *SIFT*, la combinación que mejores resultados entregó fue:

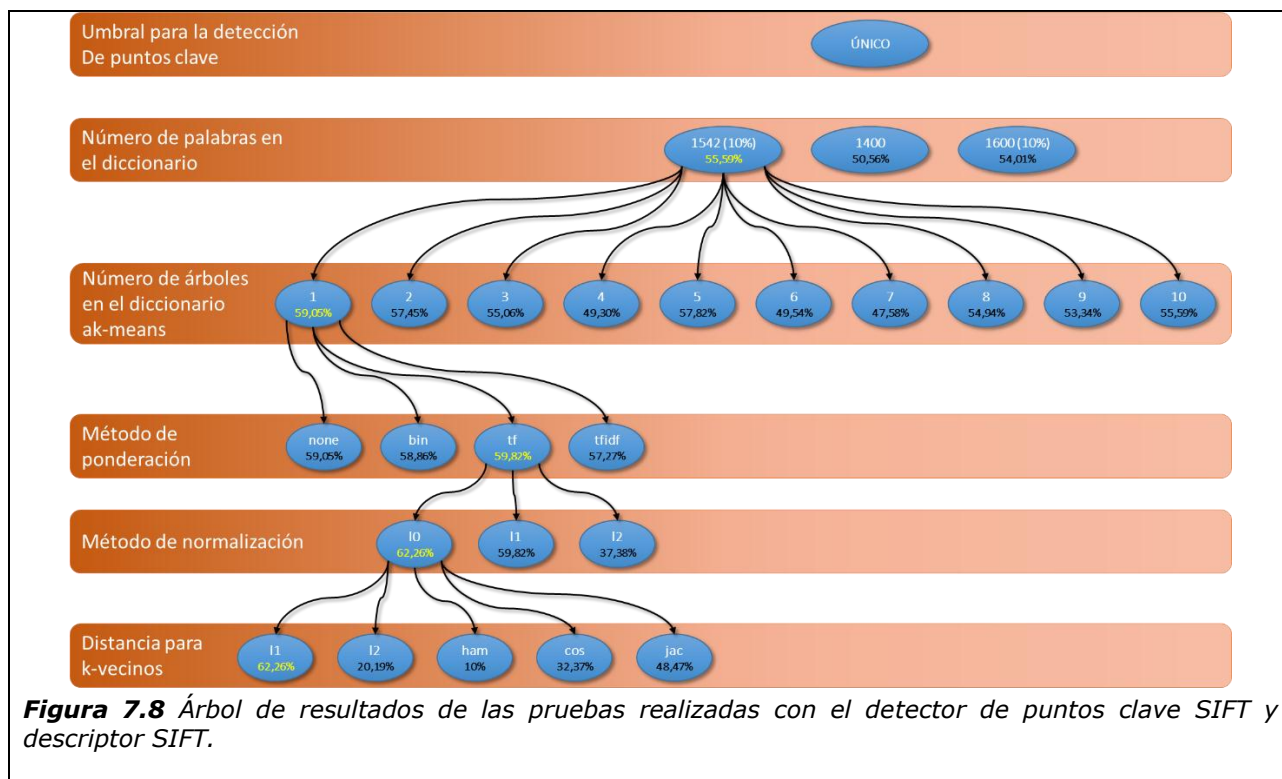
Palabras en el diccionario: 1542 (Criterio de 10%)

Árboles en el diccionario: 1

Ponderación: "tf"

Normalización: "l0"

Distancia: "l1"



Para el descriptor *SURF*, la combinación de parámetros que mejores resultados entregó fue:

Umbral para detección: 4-5
 Palabras en el diccionario: 15550
 Árboles en el diccionario: 4
 Ponderación: "bin"
 Normalización: "l0", "l1"
 Distancia: "l1"

En el nivel del parámetro de "umbral para la detección de puntos clave", los valores 4-5 y 6-6 se refieren a un muestreo cada 4 pixels en la dirección vertical y cada 5 pixels en la dirección horizontal, y un muestreo cada 6 pixels tanto en la dirección vertical como horizontal.

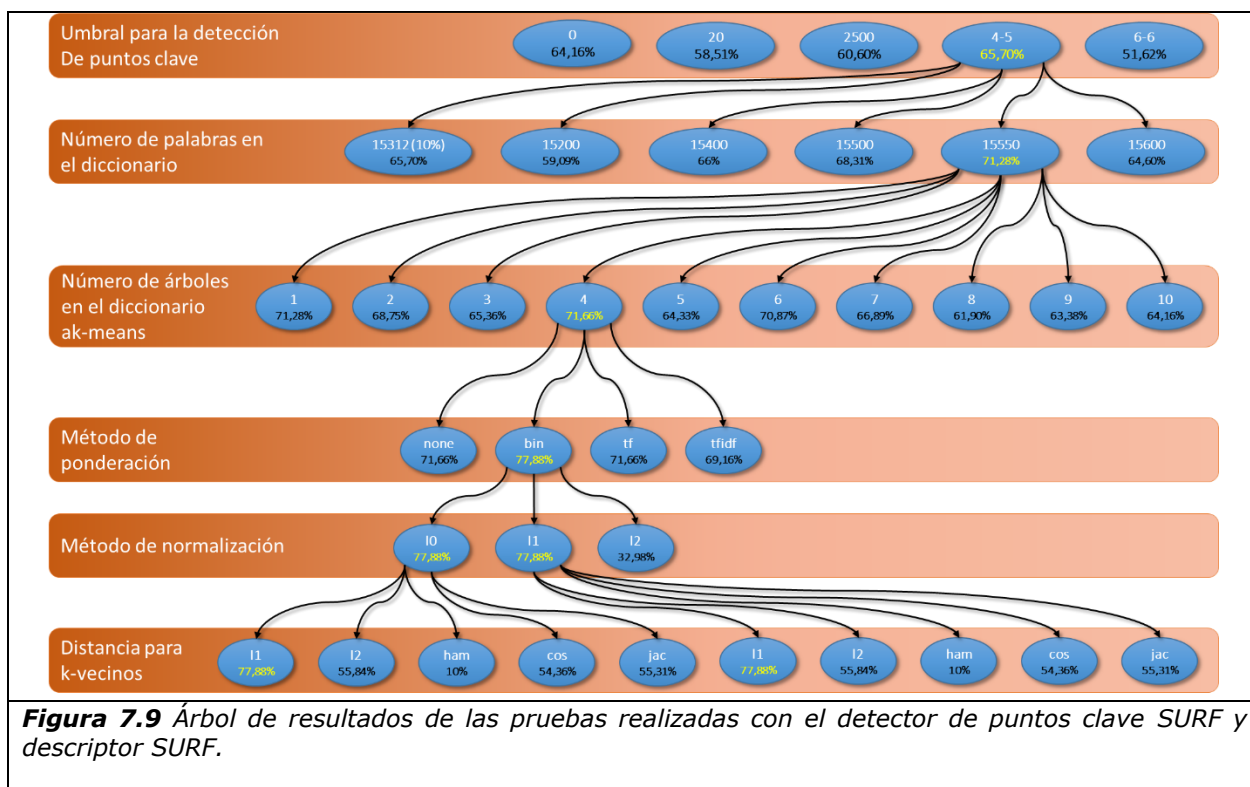
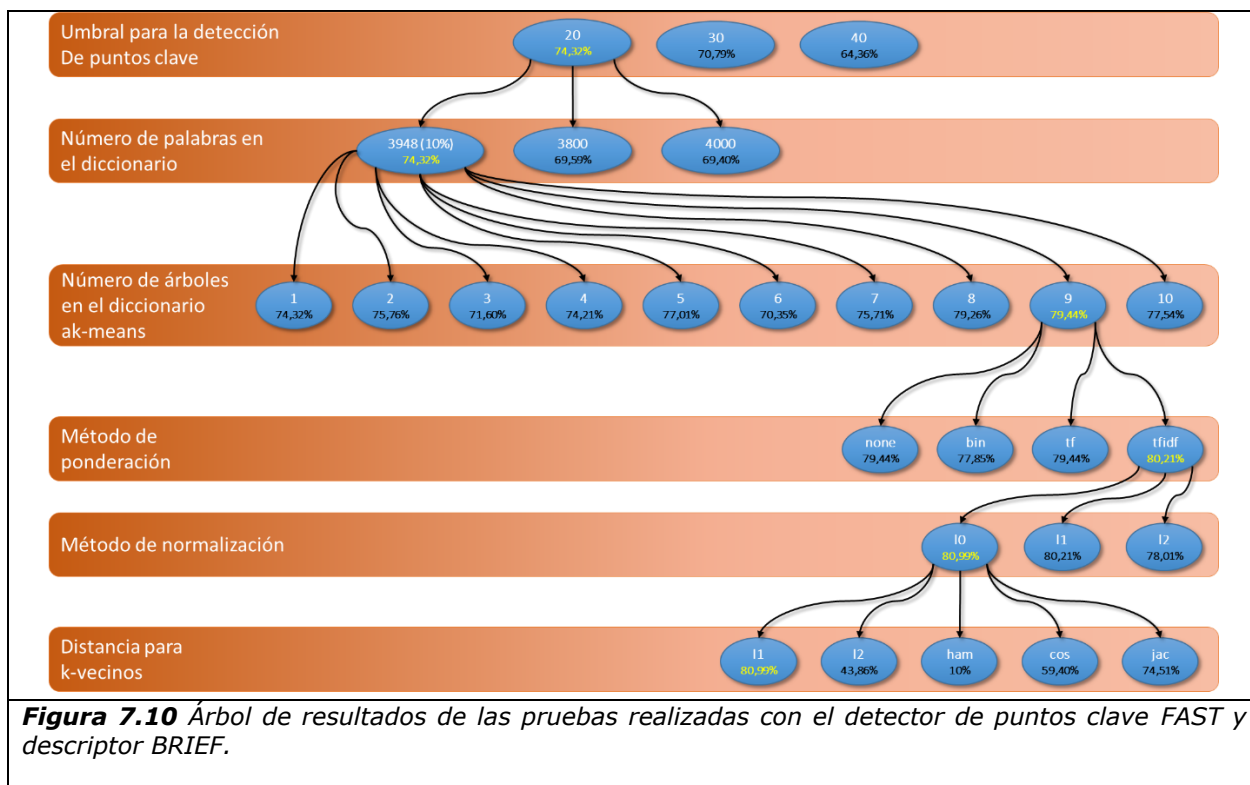


Figura 7.9 Árbol de resultados de las pruebas realizadas con el detector de puntos clave SURF y descriptor SURF.

Para el descriptor *BRIEF*, la combinación que mejores resultados entregó fue:

Umbral para detección: 20
 Palabras en el diccionario: 3948
 Árboles en el diccionario: 9
 Ponderación: "tfidf"
 Normalización: "l0"
 Distancia: "l1"



Para el descriptor *BRISK*, la combinación que mejores resultados entregó fue:

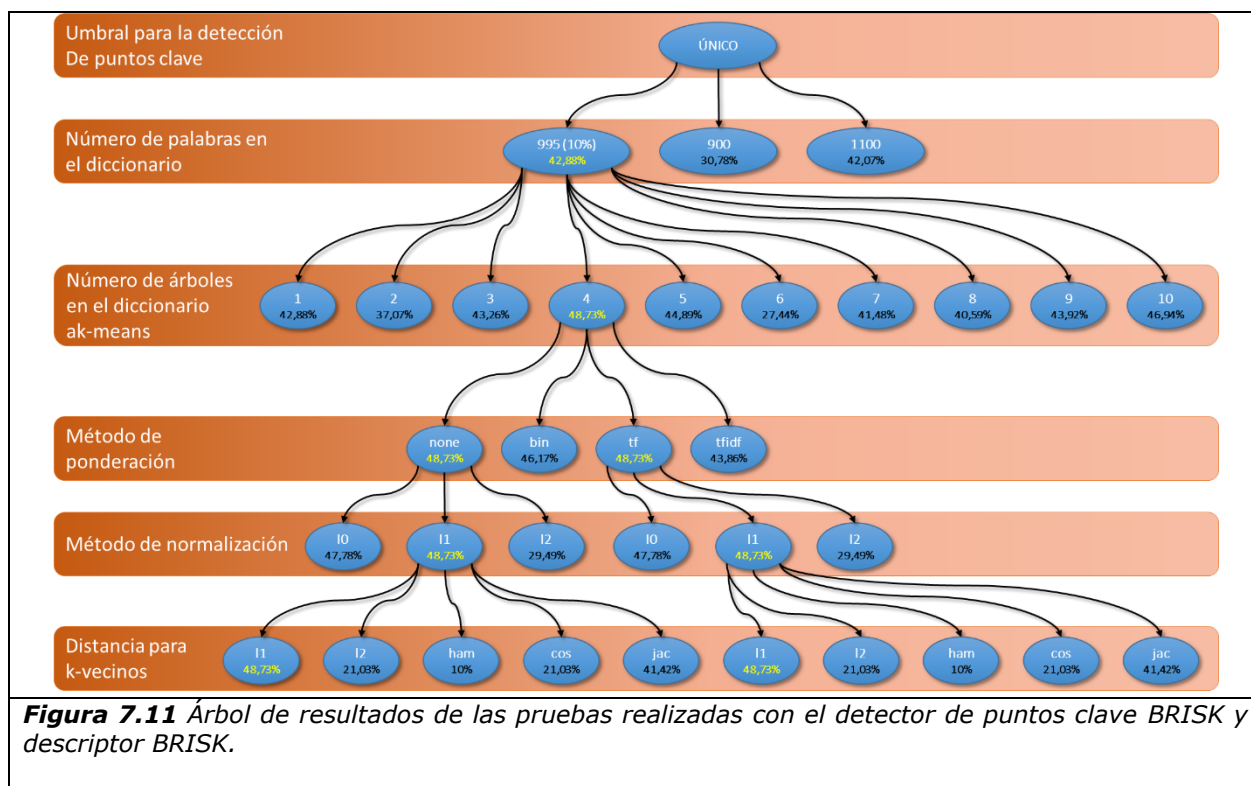
Palabras en el diccionario: 995

Árboles en el diccionario: 4

Ponderación: "none", "tf"

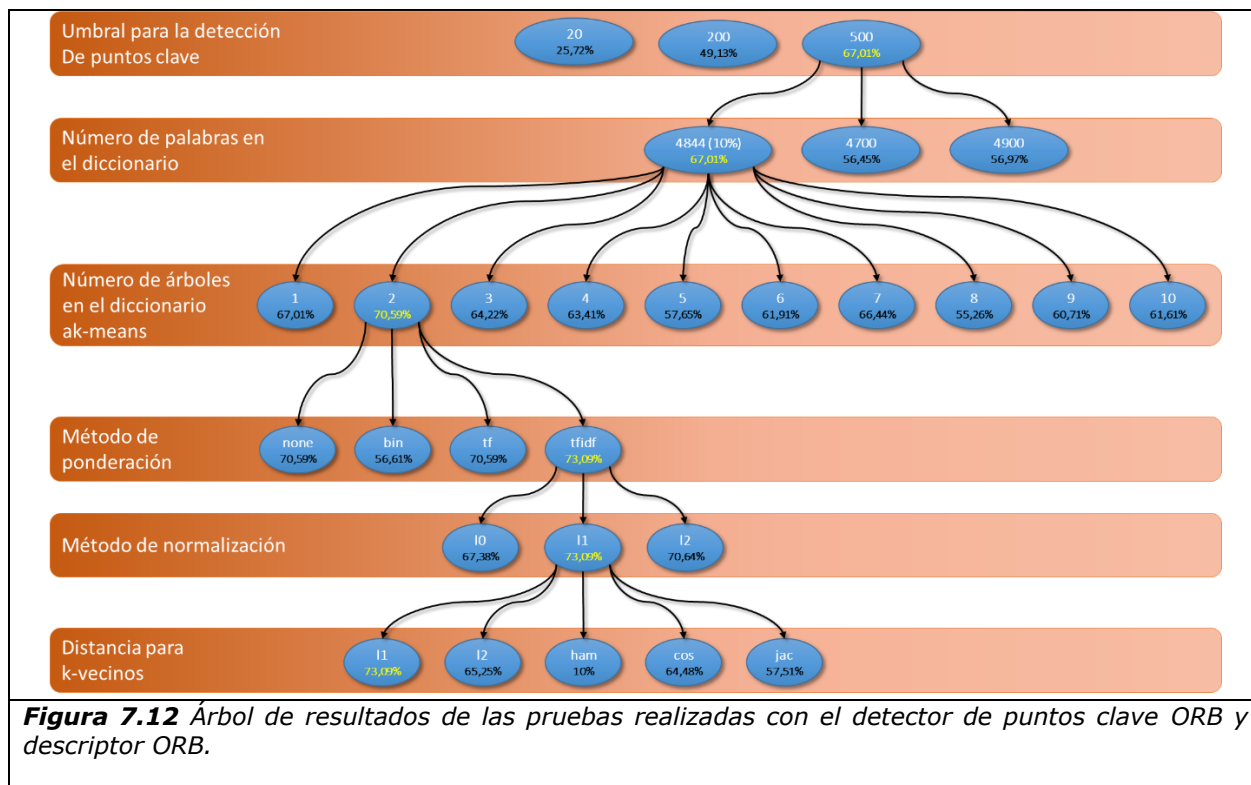
Normalización: "l1"

Distancia: "l1"



Para el descriptor *ORB*, la combinación que mejores resultados entregó fue:

Umbral para detección: 500
 Palabras en el diccionario: 4844
 Árboles en el diccionario: 2
 Ponderación: "tfidf"
 Normalización: "l1"
 Distancia: "l1"



Para el descriptor *FREAK*, la combinación que mejores resultados entregó fue:

Umbral para detección: 30
 Palabras en el diccionario: 2767
 Árboles en el diccionario: 7
 Ponderación: "none", "tf"
 Normalización: "l1"
 Distancia: "l1"

