



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

DETECCIÓN Y SEGUIMIENTO DE PERSONAS EN AMBIENTES DINÁMICOS USANDO UN ROBOT MÓVIL AUTÓNOMO

TESIS PARA OPTAR AL GRADO DE DOCTORA EN INGENIERÍA
ELÉCTRICA

WILMA PAIRO HUAYNOCA

PROFESOR GUÍA:
DR. JAVIER RUIZ DEL SOLAR SAN MARTÍN

PROFESOR CO-GUÍA:
DR. PATRICIO LONCOMILLA ZAMBRANA

MIEMBROS DE LA COMISIÓN:
DR. JORGE SILVA SÁNCHEZ
DR. MARÍA JOSÉ ESCOBAR
DR. PEDRO NÚÑEZ TRUJILLO

SANTIAGO DE CHILE
2019

RESUMEN DE TESIS PARA OPTAR
AL GRADO DE DOCTORA EN INGENIERÍA ELÉCTRICA
POR: WILMA PAIRO HUAYNOCA
FECHA: 2019
PROF. GUIA: JAVIER RUIZ DEL SOLAR

Desde que la humanidad comenzó a diseñar y usar máquinas inteligentes siempre ha deseado que estas imiten el comportamiento humano, que tengan inteligencia y autonomía. Para poder darles la habilidad de seguir personas, primero se debe implementar la detección de la persona a seguir. Sin embargo, los métodos de detección confiables existentes actualmente no se ejecutan en tiempo real en cpus. Por otro lado, existen innumerables métodos de seguimiento de personas, pero dichos métodos pueden fallar frente a oclusiones, grandes variaciones de iluminación o cuando el tiempo de seguimiento es largo. En este trabajo de tesis se propone un sistema de seguimiento libre de retardo (delay-free), que usa un sistema de detección confiable pero lento como base. El sistema es capaz de actualizar las detecciones retrasadas, de modo que la inicialización de los algoritmos de seguimiento sea adecuada.

El sistema propuesto debe considerar varios factores, tales como tener la capacidad de ser invariante a cambios bruscos en el aspecto y forma de la persona. Estas variaciones de aspecto se producen por las distintas posiciones relativas entre el sensor y la persona observada. Además, el sistema debe ser invariante a cambios de iluminación y debe ser capaz de recuperarse ante oclusiones y aglomeraciones de personas. El método de seguimiento propuesto se basa en el algoritmo KLT (Shi-Tomasi función de seguimiento de puntos o esquinas [1]) mezclado con RANSAC (RANDOM SAMPLE CONSENSUS [2]) que es usado para realizar la transformación libre de retardo. Es importante señalar que el procedimiento de transformación libre de retardo es independiente del algoritmo de detección, sólo requiere que la región del objeto tenga textura visual que permita realizar el seguimiento. Además, el método propuesto logra detectar falsas detecciones causadas por fallas en el algoritmo de detección primaria de forma explícita, usando clasificadores estadísticos (SVM - Support Vector Machine), así como oclusiones, que se modelan utilizando un modelo oculto de Markov (HMM) cuyo estado se estima utilizando el algoritmo de Viterbi.

El método de seguimiento propuesto en esta tesis se validó en la aplicación "seguimiento de personas", en la que un robot de servicio sigue a un ser humano bajo condiciones propias del mundo real sin restricciones: en ambientes internos y externos (al aire libre), con intensidad de iluminación variable, fondos desordenados, ambientes con aglomeraciones, y oclusiones causadas por personas. El método propuesto se comparó con otros métodos de detección y de seguimiento: HOG [3], CT - Real-time compressive tracking [4], TDK Tracking-by-detection with Kernels [5], Meanshift [6], y KCF Kernel Correlation Filter [7] y de acuerdo a los resultados obtenidos, el método propuesto tiene mejor rendimiento que los métodos mencionados.

Agradecimientos

En primer lugar, agradecer a Dios y mi familia, sobre todo a mis padres que siempre han estado y son un ejemplo para mí desde el día que llegué a este mundo. A mi madre Zenovia por su apoyo incondicional y por esa frase que siempre llevo en mi mente “¡querer es poder!”. A mi padre Sebastián que siempre fue ejemplo de trabajo y esfuerzo, a mi hermano Alex por brindarme esa mano cuando más la he necesitado.

Quiero agradecer de manera especial y sincera al Profesor Javier Ruiz del Solar por su apoyo y confianza en mi trabajo y por darme la oportunidad de trabajar junto a él en el desarrollo de esta tesis.

A mi co-guía y amigo Patricio Loncomilla, no conocí a una persona más brillante amable y sincera, gracias por tus consejos, apoyo y sugerencias.

Gracias a mis queridos compañeros del laboratorio, del trabajo y a los amigos que conocí aquí en Chile, por su apoyo y amistad, sin ustedes no lo hubiera logrado.

Dar un gran agradecimiento a mi amada hija Antonella y mi pareja Leonardo que son mi inspiración y apoyo para seguir adelante.

Finalmente agradecer al gobierno de Chile y FONDECYT ya que esta tesis fue financiada parcialmente por los proyectos 1130153 y 1161500.

Tabla de Contenido

1. Introducción.....	1
1.1. Fundamentación General	1
1.2. Objetivos	2
1.2.1. Objetivo General.....	2
1.2.2. Objetivos Específicos	3
1.3. Hipótesis.....	3
1.4. Estructura de la Tesis	4
2. Aportes de la Tesis.....	5
3. Revisión Bibliográfica.....	6
3.1. Seguimiento de Personas por Robots.....	6
3.2. Sistemas de Detección de Objetos	7
3.2.1. Sistema de Detección de Objetos Basado en Histogramas de Gradientes Orientados HOG.....	7
3.2.2. Sistema de Detección de Objetos con Entrenamiento Discriminativo Basado en un Modelo de Partes.....	9
3.2.3. Sistemas de detección basados en Deep Learning.....	12
3.2.3.1. YOLO (You Only Look Once).....	13
3.2.3.2. OpenPose	14
3.2.3.3. Openpifpaf	15
3.3. Sistemas de Seguimiento	16
3.3.1. Método Compressive Tracking.....	16
3.3.2. Método Tracking-by-Detection with Kernels.....	18
3.3.3. Método High-Speed Tracking with Kernelized Correlation Filters.....	20
3.3.4. Método Mean Shift	21
3.4. Detección y Seguimiento de Características Locales.....	22
3.5. Modelos Ocultos de Markov	23
3.5.1. Procesos Discretos de Markov.....	23
3.5.2. Elementos de un HMM.....	23
3.5.3. Algoritmo de Viterbi.....	24
3.6. Random Sample Consensus	25
4. Metodología Propuesta	28

4.1. Estimación y Transformación Libre de Retardo	31
4.2. Oclusiones y gestión de falsas detecciones	33
4.3. Gestión de Falsas Detecciones	34
4.4. Seguimiento de Objetos Libres de Retardo.....	38
5. Experimentos	41
5.1. Descripción de los experimentos a realizar.....	41
5.2. Métodos evaluados.....	44
5.3. Resultados	45
5.4. Posibles aplicaciones del método libre de retardo	51
5.4.1. Robot de servicio	51
5.4.2. Otras aplicaciones	52
6. Conclusiones.....	54
7. Bibliografía.....	56
Anexo A.....	61

Índice de Figuras

Figura 1: Pasos para calcular el descriptor HOG.	8
Figura 2: Proceso de <i>matching</i> para una escala.	10
Figura 3: Modelo mixto para la categoría bicicletas.	11
Figura 4: Arquitectura de la red neuronal YOLO [13]	14
Figura 5: Pipeline genérico de la estimación de a postura humana a) Imagen de entrada para la red. b) Mapa de confianza predictiva para la detección de las partes del cuerpo. c) afinidad de los campos por asociación de partes. d) usando match bipartita para asociar las partes candidatas. e) salida de todas las personas de la imagen.	15
Figura 6: Arquitectura del modelo: La entrada es una imagen de tamaño (H,W) con 3 canales de color. La red neuronal bsada en codificadores produce campos PIF y PAF con canales de 17x5 y 19x7. Una operación con paso dos se indica mediante “//2”. El decodificador es un programa que convierte llo campos PIF y PAF en estimaciones de pose que contienen 17 uniones cada una. Cada articulación esta representada por una coordenada x e y además de un puntaje de confianza.....	16
Figura 7: Componentes principales del algoritmo <i>Compressive tracking</i>	17
Figura 8. Ejemplos con los resultados obtenidos por el método <i>Tracking by Detection with kernels</i>	19
Figura 9:Resultados cualitativos para el high-speed tracking with kernelized correlation filters propuesto en [7], comparado con el Struck y TDL. El kernel elegido es gaussiano, en las características de HOG.	20
Figura 10: Tres iteraciones del Mean Shift.	21
Figura 11. La recta robusta es la formada por los puntos a y b. Todo el conjunto de puntos, salvo dos <i>outliers</i> (c y d), forman el denominado conjunto de consenso.....	26
Figura 12. Correspondencias entre dos conjuntos de puntos. Transformación de todos los puntos siguiendo el modelo de consenso calculado por RANSAC.	27
Figura 13. Diagrama del concepto de <i>grupo de imágenes</i> (GOF). El proceso de detección de objetos dentro de una imagen demora un tiempo considerable. El set de imágenes que existe entre detección y detección se denominará GOF.....	29
Figura 14: Módulos que componen el método de seguimiento propuesto.....	29
Figura 15: Ejemplo de la detección delay-free. Las imágenes de izquierda a derecha representan detecciones de imágenes consecutivas, con diferentes tracks de características calculados. La etiqueta de la posición real del objeto GT (groud-truth) está representado por el recuadro verde. El retraso de la detección está representado por el recuadro rojo. La detección libre de retardo está representada por el recuadro azul. Los tracks de características se visualizan como líneas en la imagen. Los tracks de características aceptados por RANSAC y usados para calcular la transformación libre de	

retardo (delay-free) se visualizan en líneas amarillas.	33
Figura 16: HMM usado para modelar las oclusiones. El modelo considera dos estados: ocluido y no-ocluido.	37
Figura 17: Secuencias de imágenes en el mundo real usadas para las evaluaciones del método. El recuadro azul representa la salida del método libre de retardo (Delay-free). Las líneas representan los tracks de características. Los tracks aceptados por RANSAC y usados para calcular la transformación libre de retardo se muestran en Amarillo.....	44
Figura 18: Fallas en el seguimiento causadas por cambios fuertes de iluminación (videos v10 y v22). Etiqueta de la posición real de la persona (GT) en verde, DPM-HOG detección de objetos en rojo y la transformación libre de retardo en azul.....	51
Figura 19: Imágenes del robot (Bender) siguiendo a una persona. Izquierda/Derecha: Exterior/Ambiente Interior.....	52

Capítulo 1

Introducción

1.1. Fundamentación General

Desde que el hombre vio la posibilidad de usar máquinas inteligentes en vez de personas se puede decir que la humanidad siempre ha deseado que estas imiten el comportamiento humano, que tengan cierto grado de inteligencia y autonomía. Generalmente esto ha sido inspirado por la televisión, el cine y la literatura. Por lo tanto, para que los robots, ya sean sociales o de servicio, puedan tener estas habilidades necesitan extraer información del entorno que los rodea y así poder tener una interacción natural con el usuario. Para esto necesita contar con habilidades tales como: detectar personas, comprender el comportamiento y las indicaciones de las personas, realizar análisis de expresiones faciales, miradas o movimientos de manos. En tal sentido si los robots van a formar parte de la vida de las personas necesitan tener la habilidad de seguir a una persona y hasta caminar junto a esta.

Para poder darle la habilidad de realizar el seguimiento de personas, primero se debe realizar la detección de la persona a seguir. Sin embargo, los métodos de detección confiables no se ejecutan en tiempo real en cpus. Por otro lado, existen innumerables métodos de seguimiento de personas, pero normalmente todos los métodos tienen fallas frente a ciertos factores como: oclusiones, grandes variaciones de iluminación o cuando el tiempo de seguimiento es largo.

La detección y seguimiento de personas por un robot en un ambiente dinámico involucra la solución de un conjunto de problemas, tales como: detección robusta frente a oclusiones parciales o totales por parte de otras personas, evasión de objetos que obstruyen el seguimiento, problemas relacionados a las perspectivas desde las cuales el robot observa a la persona, generando variaciones en la observación de la misma y problemas relacionados a cambios en las condiciones del ambiente como por ejemplo cambios en la iluminación y la distribución de los objetos.

En la actualidad se ha tratado de abordar el problema de seguimiento con métodos tradicionales como el seguimiento mediante un sensor láser, y también con tecnologías y *frameworks* que se ofrecen en el mercado. Un ejemplo es el sensor kinect, que cuenta con funcionalidades como las que fueron proporcionadas por OpenNi [8], el cual tiene un módulo de detección y seguimiento de personas. Sin embargo, estos sistemas no son robustos a la hora de integrarlos con un robot, tal como lo indican los resultados obtenidos en [9], donde se observa que el uso de este tipo de sensor no siempre es fiable.

Las capacidades de cada sensor para la detección dependen mucho del material en el que se refleja (por ejemplo, en muchos materiales negros no se refleja la luz infrarroja), así como la cantidad de luz solar recibida por la cámara en exteriores a través de las ventanas (la luz solar puede saturar al sensor IR y saturar los patrones proyectados).

Por otro lado, tratar de resolver el problema con un único sensor se hace casi imposible. Es por esto que muchos trabajos combinan más de un sensor [10], porque al integrar la capacidad de los distintos sensores, se puede ver el objeto desde diferentes perspectivas y de esta manera reaccionar adecuadamente ante diversas situaciones. Por ejemplo, cuando dos o más personas caminan juntas ambas caminan mirando hacia el frente y no hacia su acompañante, o si una de ellas se detiene frente a un objeto, las demás también se detienen a su lado. Por lo tanto, en este tipo de situaciones tanto el ojo humano como los sensores del robot solo pueden ver un fragmento del cuerpo de la persona seguida.

De acuerdo con lo explicado, la idea básica de este trabajo es generar un método de seguimiento y detección de personas que pueda estimar la posición de una persona con respecto al robot, manteniendo la persona en un rango observable por la cámara. De esta manera se puede lograr que el robot siga a la persona correcta en todo momento.

1.2. Objetivos

1.2.1. Objetivo General

Diseñar y construir un método que sea capaz de realizar la detección y seguimiento de personas en ambientes dinámicos e implementarlo en un robot móvil autónomo. El método debe ser invariante a cambios bruscos en el aspecto y forma de la persona producidos por las distintas posiciones relativas entre el sensor (cámara) y la persona observada e invariante a cambios de iluminación. El modelo de este método también debe ser capaz de recuperarse (invarianza) ante oclusiones y aglomeraciones de personas.

Invariancia:

- Invariancia a oclusiones: En caso de que la cámara del robot siga a una persona y otro sujeto se cruce entre él sensor y la persona, el sistema de detección debe ser capaz de recuperarse y de esta manera no perderla.
- Invariancia a la iluminación: El sistema de detección y seguimiento debe ser capaz de seguir a la persona en ambientes interiores y exteriores con tasas elevadas de reconocimiento, sin depender de las variaciones de iluminación ambiental.
- Invariancia frente a aglomeraciones: El sistema debe ser capaz de seguir a la persona, aún si hay un grupo o varios grupos de personas que se encuentran entre la persona y el robot.

1.2.2. Objetivos Específicos

- Generar un modelo de caracterización de personas que permita reconocer partes deformables del cuerpo o bien el cuerpo completo de la persona, en base a los datos del sensor (cámara) utilizado.
 - Se requiere diseñar un sistema que genere un modelo de la persona que considere la cercanía y observabilidad de la misma con respecto al robot, ya que en un momento se podría observar el torso completo, en otro sólo un brazo, un fragmento del brazo o sólo la cabeza.
- Evaluar distintas metodologías de detección y seguimiento de personas en bases de datos públicas y en bases específicamente diseñadas.
- Implementar un sistema de detección robusto de personas que, en base a los sistemas estudiados en la literatura, permitan seguir a la persona en entornos donde se encuentran múltiples personas.
- Implementar un sistema de seguimiento de personas que determine la pose de la persona en tiempo real basado en la caracterización de dicha persona.
- Realizar pruebas de funcionamiento del sistema utilizando bases de datos existentes, nuevas bases de datos en ambientes dinámicos y un robot de servicio.

1.3. Hipótesis

En este trabajo de tesis se plantean las siguientes hipótesis:

1. El uso de tracking de características locales permite que un sistema de detección de personas u objetos en movimiento que no funciona en tiempo real pueda entregar detecciones sin retrasos.
2. El uso de características visuales extraídas desde secuencias de imágenes permite que un sistema de tracking de personas u objetos funcione robustamente ante falsas detecciones y cambios abruptos de iluminación, además de permitir reanudar el tracking de objetos previamente ocluidos.
3. El uso combinado de tracking de características locales y características visuales permite el seguimiento en tiempo real de personas u objetos en movimiento en ambientes dinámicos.

1.4. Estructura de la Tesis

La tesis está estructurada de la siguiente forma: El capítulo 2 muestra los aportes de la tesis, el capítulo 3 presenta una revisión bibliográfica de los métodos de seguimiento de personas usando robots, considerando: sistemas de detección de personas y objetos, sistemas de seguimiento en imágenes, y sistemas de extracción de características, modelos ocultos de Markov los cuales fueron empleados para la detección de oclusiones y falsas detecciones. En el capítulo 4 se presentará la metodología con la cual se construye el sistema de seguimiento libre de retardo, la gestión de falsas detecciones y la detección de oclusiones. En el capítulo 5 se presentan la descripción de los experimentos a realizar, la validación y evaluación de diversos métodos de seguimiento comparados con el método propuesto en esta tesis, los resultados obtenidos y las posibles aplicaciones del método libre de retardo. Finalmente, el capítulo 6 muestra las conclusiones de este trabajo.

Capítulo 2

Aportes de la Tesis

En esta tesis se presenta un sistema que permite hacer tracking visual de objetos a partir de detecciones, enfocado particularmente en el caso de personas. A continuación, se describen los principales aportes realizados con el desarrollo de esta tesis:

- Se propone un procedimiento basado en tracking de características locales que permite que detecciones retrasadas generadas por un sistema que no funcione en tiempo real, queden libres de retardos.
- Se propone un método que permita realizar la gestión de falsas detecciones en un sistema de tracking mediante el uso de características visuales y un clasificador SVM.
- Se genera un método de detección de oclusiones que permite determinar el inicio y el fin de oclusiones. El uso del algoritmo de Viterbi ayuda a determinar secuencias de estado que se usan para representar posibles oclusiones de los objetos bajo tracking sin la necesidad de tener un modelo del objeto ocluidor.

Capítulo 3

Revisión Bibliográfica

En este capítulo se presenta la revisión bibliográfica utilizada para el desarrollo de este trabajo. Esta sección está dividida en diferentes áreas que se abordarán en esta tesis. Muchos de los textos citados servirán de referencia y se utilizarán en el desarrollo de este trabajo.

3.1. Seguimiento de Personas por Robots

A través de los años se ha tratado de que los robots sean capaces de seguir a las personas y tengan un comportamiento similar o igual a ellas, es por ello que en este apartado se describen las diferentes investigaciones realizadas respecto a esta tarea tan simple para los seres humanos, pero compleja para un robot.

Chen [11] señala que el objetivo es realizar seguimiento y detección de personas en ambientes interiores usando un láser y un robot de servicio, utilizando la odometría del robot junto a la mezcla de información con los barridos del láser usando un filtrado de *outliers*. Otro método que emplea es la detección y comparación de objetos en movimiento en un mapa local, para cada objeto en movimiento emplea un filtro de Kalman extendido EKF que es actualizado para realizar el seguimiento. Se usan tres técnicas para el scan matching: *Outlier filtering*, *Moving objects filtering* y *Multiple ICP (Iterative closest point)*.

Ansuategui [12] propone realizar el seguimiento de una persona basándose en los datos del láser y cámaras mono y estéreo. Este sistema está basado en un filtro de partículas en tiempo real mezclada con la información provista por los sensores (láser, imágenes 2D y 3D), y realiza el cálculo de la posición de los objetos propuestos usando la probabilidad de existencia de los patrones de las piernas, también trabaja con las características de las imágenes y finalmente aplica flujo óptico.

Bellotto [10] propone una solución para el seguimiento de personas con un robot móvil que implementa una fusión de datos multi-sensorial. El algoritmo está basado principalmente en el uso de un sensor láser. Este sistema se basa en la detección de patrones de piernas, extraído de los barridos del láser. Este sistema puede trabajar en entornos no estructurados. Estos patrones ayudan a detectar personas estáticas y personas que caminan, las posiciones de las personas son estimadas mediante un *Unscented Kalman Filter* (UKF), trabaja en ambientes interiores. Para la detección se adoptan técnicas de fusión de datos multi-sensorial, buscando dos tipos de información: la primera es la detección de piernas basada en los datos generados por los barridos del

láser SICK LRF y la otra es la detección de la cara usando una cámara monocular. Para la detección de piernas el algoritmo se divide en tres partes principales: procesamiento de datos, detección de los bordes verticales y extracción de patrones de piernas.

Wang [13] implementa un seguimiento de personas y de evasión de obstáculos en un robot móvil. El robot es llamado (W-1) el cual se mueve por medio de tres ruedas omnidireccionales y consta de un sistema de visión para reconocer a la persona que sigue, además de un sensor infrarrojo que estima la distancia entre la persona seguida y el robot. De esta manera, el robot puede seguir a la persona a una distancia fija. Un control fuzzy es usado para determinar las funciones de movimiento hacia atrás, adelante y giro del robot.

Xing [14] presenta un sistema de seguimiento de personas basado en Kinect. La habilidad de seguir el esqueleto de la persona es usada para identificarlas. La posición de la persona es directamente medida por el sensor con las imágenes de profundidad.

Hoshino [15] realiza seguimiento de personas basado en un sensor láser y en el sensor Kinect para identificar a la persona. El láser se usa para observar lo que hay alrededor y el Kinect se encarga de observar a la persona. El láser detecta un patrón de piernas y en el caso del seguimiento de personas se utiliza un filtro de partículas (*k-means clustering*).

3.2. Sistemas de Detección de Objetos

En la literatura existen muchos sistemas de detección, pero para esta tesis se revisarán solo dos, debido a que estos dos sistemas tuvieron gran trascendencia tanto en este trabajo como en trabajos anteriores. En la sección 3.2.1 se detalla el método de detección de objetos basado en histograma de gradientes orientados HOG que permite realizar la extracción de características que distinguen una imagen de otra. En la sección 3.2.2 se detalla el método de detección de objetos con entrenamiento discriminativo basado en el método de la sección 3.2.1 el cual usa un modelo basado en “gramática visual”, con el que cada parte del objeto puede ser definido directamente en términos de otras partes de un modelo gramatical. En la sección 3.2.3 se detalla los nuevos métodos de detección de objetos y personas basados en aprendizaje profundo (en inglés Deep Learning) estos métodos se encargan de detectar objetos y personas considerándose como un problema único de regresión y por otro lado reconocer poses humanas a través de la detección de puntos anatómicos del cuerpo. Cada uno de los métodos mencionados se detalla a continuación.

3.2.1. Sistema de Detección de Objetos Basado en Histogramas de Gradientes Orientados HOG

Los Histogramas de Gradientes Orientados (HOG) indican que la forma de un objeto en una imagen puede ser descrita por medio de la distribución de los gradientes [3]. El objetivo principal de esta técnica es la extracción de características distintivas de una imagen. La imagen se divide en pequeñas celdas cada una de las cuales acumula

direcciones del histograma de gradientes u orientaciones de los bordes dentro de cada celda. Este descriptor se creó para la detección de personas con un clasificador simple [3], y posteriormente se generalizó para la detección de todo tipo de objetos [16]. Se recomienda, para una mejor respuesta, normalizar el contraste en zonas grandes (denominadas bloques) y utilizar dicho resultado para normalizar las celdas del bloque. El conjunto de las celdas normalizadas se denomina descriptor HOG (ver Figura 1). Por último, se utilizan los descriptores HOG de la ventana de detección como entrada a un clasificador SVMLight [17], que es una implementación de un SVM (*Support Vector Machine*) lineal adecuada para trabajar con grandes conjuntos de datos. SVM es una técnica para entrenar clasificadores capaces de discernir entre patrones de la clase a detectar y cualquier otro patrón. El algoritmo SVM se encarga de encontrar entre todas las superficies de separación posibles, la que maximiza la distancia entre los elementos más próximos de dos clases, por medio de la programación cuadrática.

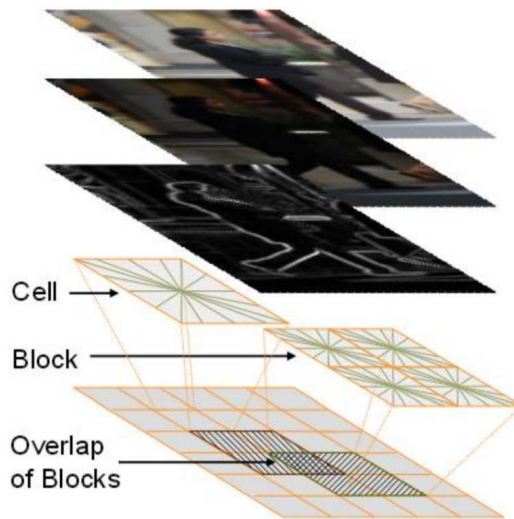


Figura 1: Pasos para calcular el descriptor HOG [18].

Los descriptores HOG están inspirados en las características SIFT [19] y en un contexto de formas [18]. Los autores remarcan como principal ventaja de estos descriptores, el hecho de que captan estructuras características de la forma a detectar y que, al trabajar con celdas pequeñas, estas características se mantienen invariantes ante cambios de posición, escala e iluminación, por lo que son adecuadas para detectar personas que se encuentren en diferentes posiciones.

3.2.2. Sistema de Detección de Objetos con Entrenamiento Discriminativo Basado en un Modelo de Partes

El reconocimiento de objetos es un proceso complejo, ya que los objetos pueden tener grandes variaciones en su apariencia de acuerdo con su forma, posición y desde los diferentes puntos de vista en las que se observa al objeto.

Felzenszwalb [16] describe un sistema de detección que puede trabajar con diversos objetos, usando una mezcla multi-escala de modelos de partes deformables, los cuales son entrenados usando un procedimiento discriminativo. Para realizar el entrenamiento solo se requiere marcar los objetos dentro del conjunto de imágenes. Para ello se usaron las bases de datos provistas por PASCAL VOC [20] y la base de datos de personas INRIA Person Data Set [3].

El enfoque que emplearon está basado en la construcción de un *framework* que contiene objetos compuestos por una colección de partes clasificadas en una configuración deformable donde cada parte de la colección captura las propiedades locales del objeto, mientras que las partes deformables son caracterizadas como conexiones de cadenas entre un par de partes. Es por ello que un modelo simple no es suficiente para representar las diversas categorías del objeto.

Por último, se usa un modelo basado en “gramática visual”, con el cual cada parte del objeto puede ser definido directamente en términos de otras partes de un modelo gramatical. Por otro lado, los modelos gramaticales permiten tener variaciones estructurales. Por ejemplo, diferentes modelos pueden compartir partes de otros objetos y pueden ser reutilizables.

El detector de Dalal-Triggs [3], que ganó la competencia de la detección de objetos en PASCAL 2006, usa un filtro sencillo sobre un histograma orientado de gradientes (HOG), estas características representan la categoría de un objeto. Este detector utiliza un enfoque de desplazamiento de ventanas, donde un filtro es aplicado a todas las posiciones y escalas de una imagen. El clasificador determina si existe o no una categoría del objeto dada la posición y la escala. Dado que el modelo es un filtro simple se puede calcular una puntuación como $\beta \cdot \Phi(x)$ donde β es el filtro, x es una imagen con la posición y la escala, y $\Phi(x)$ es un vector de características. Una de las principales innovaciones del detector de Dalal-Triggs fue la construcción de características particularmente eficaces.

El primer aporte de [16] consiste en robustecer el modelo Dalal-Triggs usando una estructura estrella del modelo basado en partes, definido por un filtro “raíz” (análoga a la del filtro de Dalal-Triggs) además de un conjunto de filtros de partes y modelos de deformaciones asociados. El puntaje de uno de los modelos estrella en una posición y escala particular dentro de una imagen es el resultado del filtro raíz dada una

localización más la suma de las partes en sus posiciones de mayor puntaje ver Ecuación (1), una vez realizado esto, se resta el puntaje de las partes respecto de su posición ideal.

$$score(p_0) = \max_{p_1, \dots, p_n} score(p_0, \dots, p_n) \quad (1)$$

Ambos resultados tanto del filtro raíz como sus partes son definidos por el producto punto entre un filtro que contiene el conjunto de pesos y la sub-ventana de una de las pirámides de características calculadas en base a la imagen de entrada. En la Figura 2 se puede ver el modelo estrella para la categoría de persona.

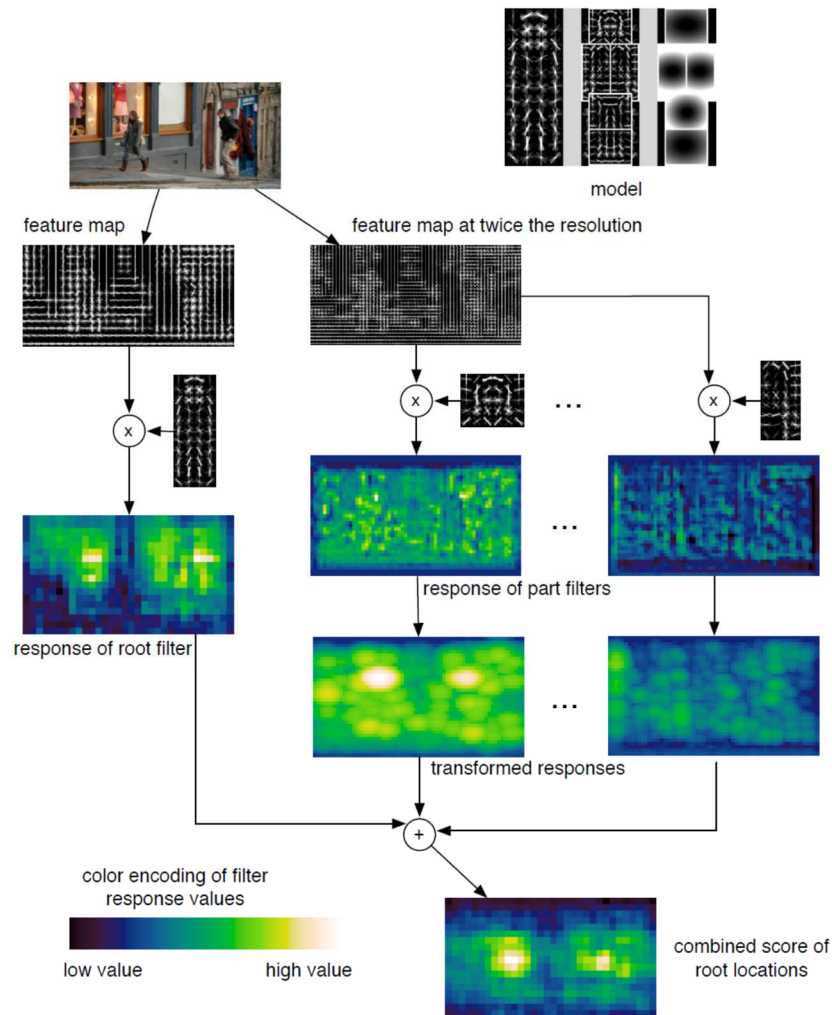


Figura 2: Proceso de *matching* para una escala [16].

En el modelo presentado, el filtro de partes captura características al doble de la resolución espacial en relación con las características capturadas por el filtro raíz. El algoritmo se aplica sobre múltiples escalas.

Al entrenar el modelo usando datos parcialmente etiquetados se usa una formulación de variable latente de MI-SVM [21] que puede ser llamado *latent SVM* (LSVM).

En un *latent SVM* cada ejemplo x es el puntaje para una función como se muestra en la Ecuación (2):

$$f_{\beta}(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z) \quad (2)$$

Donde β es un vector del modelo de parámetros, z es un valor desconocido, y $\Phi(x, z)$ es un vector de características. En el caso de que uno de los modelos estrella β es la concatenación del filtro raíz, el filtro de partes, la deformación y costo de los pesos, z es una especificación de la configuración del objeto, y $\Phi(x, z)$ es una concatenación de las sub-ventanas de la pirámide de características y las características de la deformación de partes.

La segunda clase de modelos representa una categoría de objetos mediante un modelo mixto de estrellas, cada estrella es denominada un componente. El puntaje del modelo mixto en una posición y una escala particular es el máximo sobre los componentes del puntaje de cada componente en la ubicación determinada. En este caso la información existente z , especifica una etiqueta y una configuración para ese componente. La Figura 3 muestra un modelo mixto para la categoría de bicicletas.

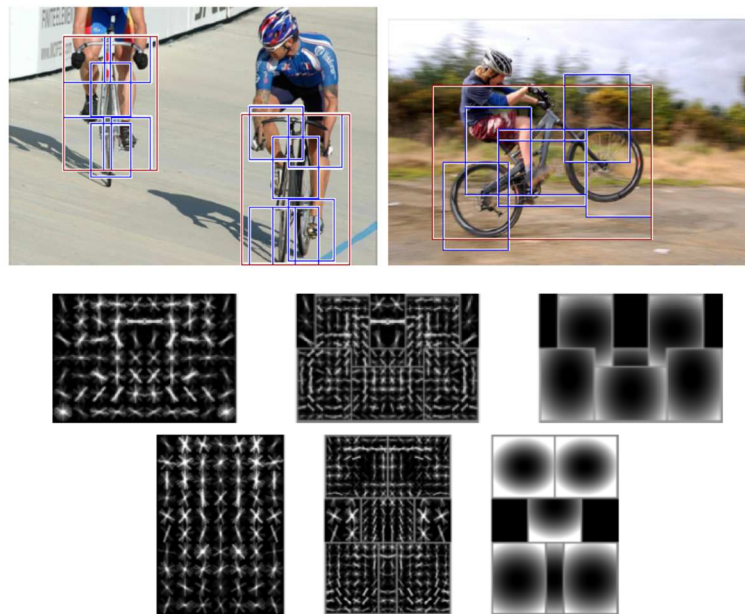


Figura 3: Modelo mixto para la categoría bicicletas [3].

Para obtener un alto rendimiento usando un entrenamiento discriminativo es importante la utilización de grandes conjuntos de entrenamiento, en el caso de la detección de objetos, el problema de entrenamiento es mayor ya que existen muchos más ejemplos del fondo que del objeto mismo. Esto motiva a un proceso de búsqueda a través de los

datos del fondo para encontrar ejemplos con un número relativamente pequeño de falsos positivos, o un gran número de ejemplos negativos.

La metodología de minería de datos para la detección de valores tipo “*hard negative*” que son difíciles de clasificar fue el adoptado por Dalal-Triggs [3]. Se analizan algoritmos de minería de datos para el entrenamiento del SVM y LSVM. Se demuestra que los métodos de extracción de datos pueden converger a un modelo óptimo que se define en términos de todo el conjunto de entrenamiento.

Los modelos empleados son definidos por puntajes para cada filtro aplicado a las sub-ventanas de la pirámide de características. Al hacer el análisis de componentes principales (PCA) en las características proporcionadas por HOG, la dimensión de las características puede ser reducida sin perder información relevante.

También se han considerado algunos problemas específicos que surgen en el desafío de la detección de objetos en PASCAL y conjuntos de datos similares. Se puede ver la ubicación de las partes como una hipótesis del objeto, la cual puede ser usada para predecir el *bounding box* del objeto. Esto se hace por medio del entrenamiento de un modelo predictor específico usando regresión de mínimos cuadrados. La idea básica es que algunas categorías de objetos provean evidencia a favor o en contra del objeto en otras categorías en la misma imagen.

3.2.3. Sistemas de detección basados en Deep Learning

Desde los años 50 hasta hace muy pocos años el terreno de la Inteligencia Artificial avanzada era solo en los laboratorios de investigación y la ciencia ficción, sin embargo, gracias al Big Data se ha revolucionado el entorno empresarial. Las organizaciones sometidas a la necesidad de la transformación digital se han convertido en criaturas sedientas de cantidades gigantescas de datos y por primera vez en la historia de la Inteligencia Artificial existe una demanda generalizada de sistemas con una inteligencia avanzada equivalente a la de un ser humano. Una de las claves de la Inteligencia Artificial está en el aprendizaje que cada vez solicita que las máquinas sean capaces de auto-programarse y en este punto es donde entra el Aprendizaje Máquina, que está al alcance cualquier programador, dentro de este aprendizaje automático está el aprendizaje supervisado y no supervisado. En este paradigma del aprendizaje supervisado los algoritmos son capaces de aprender sin intervención humana previa, sacando ellos mismos las conclusiones acerca de la semántica embebida en los datos. Entre estos algoritmos está el Deep Learning que es un conjunto de algoritmos de Aprendizaje Máquina que intenta modelar abstracciones de alto nivel en datos usando arquitecturas computacionales que admiten transformaciones no lineales múltiples e iterativas de datos expresados en forma matricial o tensorial, es por estas capacidades que el Deep Learning se acerca cada vez más a la potencia perceptiva humana.

Un caso particular de Deep Learning son las redes convolucionales o Convolutional Neural Networks (CNN) [22] [23], las cuales constituyen el estado del arte de varios problemas de visión computacional dado su buen desempeño a problemas de reconocimiento e interpretación en imágenes y video [24]. Su capacidad para actuar adecuadamente en estos contextos está basada en características fundamentales: conexiones locales, pesos compartidos, pooling y el uso de una gran cantidad de capas [22].

Los métodos presentados a continuación tienen alta precisión en las bases de datos de entrenamiento y en los dominios en los cuales fueron entrenados, sin embargo, eso no siempre se traduce en un buen desempeño en otro tipo de ambientes o dominios, además, estos algoritmos se ejecutan en tiempo real solo con la GPU proporcionada por las tarjetas de video.

3.2.3.1. YOLO (You Only Look Once)

En YOLO se toma la detección de objetos como un problema único de regresión, una única red convolucional predice simultáneamente múltiples cuadros delimitadores que enmarcan los objetos en la imagen y predice probabilidades condicionales por cada clase $p(Clase|Objeto)$ para cada uno de estos cuadros delimitadores [25]. La red neuronal puede lograr una velocidad de ejecución de 45 frames por segundo según [25].

YOLO trabaja globalmente sobre la imagen cuando hace predicciones, a diferencia de la técnica de ventana deslizante y las técnicas basadas en análisis de regiones en una imagen [25]. Por esto codifica implícitamente la información contextual, modela el tamaño y la forma de los objetos, así como su apariencia [25].

Como se puede apreciar en la Figura 4, YOLO en su primera versión tiene 24 capas convolucionales seguidas por 2 capas completamente conectadas, y en lugar de los módulos iniciales propuestos por GoogLeNet, YOLO utiliza capas de reducción de 1x1 seguidas de capas convolucionales de apariencia de 3x3 [25]. En la versión YOLO9000 [26], al agregar la normalización de lotes en todas las capas convolucionales, se obtiene una mejora de más del 2% en la precisión promedio (mAP, por sus siglas en inglés). La normalización por lote también ayuda a regularizar el modelo y se elimina el sobreajuste. YOLO9000 predice las detecciones en un mapa de características de 13x13. Si bien esto es suficiente para objetos grandes, debe beneficiarse de funciones de grano más fino para localizar objetos más pequeños [26].

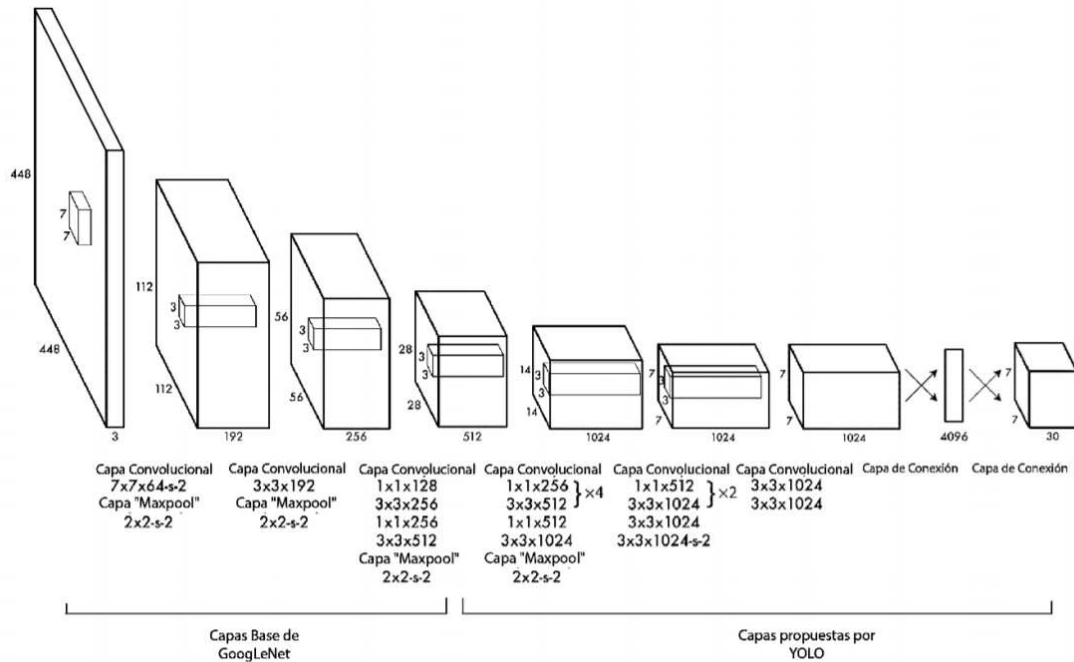


Figura 4: Arquitectura de la red neuronal YOLO [25].

3.2.3.2. OpenPose

OpenPose es una librería que implementa un sistema de detección de un cuerpo humano en tiempo real, para detectar puntos clave del cuerpo humano como son la mano, el rostro en imágenes simples. Es capaz de detectar la postura 2D de varias personas de manera eficiente. En este apartado se revisará el algoritmo y la implementación del sistema OpenPose.

Según [27], OpenPose consta de tres bloques diferente: detección de pie-cuerpo, detección de mano y detección de rostro. El bloque central es el detector de punto clave de cuerpo-pie combinado detallado en [27]. Alternativamente puede usar los detectores originales de cuerpo [28] entrenados en conjuntos de datos COCO y MPII. En la función de salida del detector de cuerpo, las propuestas de cajas de contorno facial se pueden estimar aproximadamente a partir de algunas ubicaciones de partes del cuerpo, en particular las orejas, los ojos, la nariz y el cuello.

Análogamente, las propuestas de cuadro delimitador manual se generan con los puntos clave del brazo. El algoritmo del detector de puntos clave de mano se explica con mas detalle en [29] mientras que el detector de puntos clave facial ha sido entrenado de la misma manera que el detector de puntos clave de mano La arquitectura de este método se ilustra en la Figura 5.

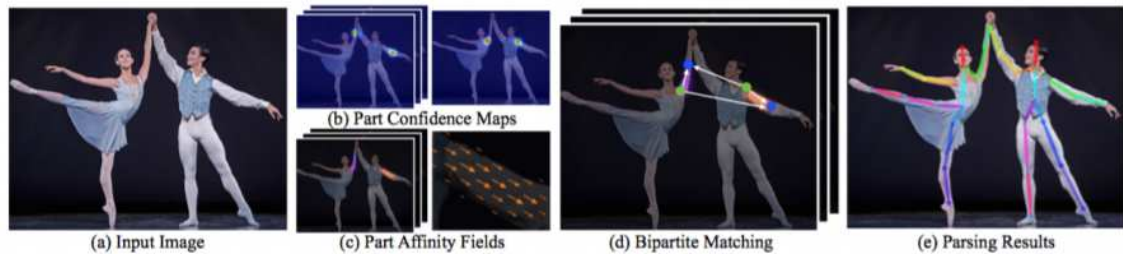


Figura 5: Pipeline genérico de la estimación de a postura humana a) Imagen de entrada para la red. b) Mapa de confianza predictiva para la detección de las partes del cuerpo. c) afinidad de los campos por asociación de partes. d) usando match bipartita para asociar las partes candidatas. e) salida de todas las personas de la imagen [29].

El tiempo de inferencia de OpenPose supera a todos los métodos de vanguardia, al tiempo que conserva resultados de alta calidad. Su modelo de cuerpo-pie combinado puede funcionar a aproximadamente 22 FPS en una máquina con una Nvidia GTX 1080 Ti, a la vez que conserva una alta precisión. OpenPose ya ha sido utilizado por la comunidad de investigación para muchos temas de visión y robótica, como la identificación de personas, el redireccionamiento de video basado en GAN de rostros humanos y cuerpos, Interacción persona-computadora. Estimación de la postura humana en 3D y generación de modelos de malla humana en 3D como se explica en [27]. Además, la biblioteca OpenCV [30] ha incluido OpenPose dentro de su módulo de red neuronal profunda (DNN)

3.2.3.3. Openpifpaf

En [31] se propone un nuevo método bottom-up para la estimación de posturas humanas en 2D en aglomeraciones y es adecuado para la movilidad urbana, como los autos autónomos y robots de servicio. Este método usa un Campo de Intensidad de Partes (PIF) para localizar las partes del cuerpo y un Campo de Asociación de Partes (PAF) para asociar las partes del cuerpo entre si para formar poses humanas completas. Este método supera a métodos como el OpenPose en bajas resoluciones y en escenas donde existe aglomeración de personas y ocultas gracias al nuevo campo compuesto PAF que codifica información de grano fino y la elección de pérdida de Laplace para regresiones que incorpora una noción de incertidumbre.

Los métodos top-down particularmente sufren cuando los peatones son ocluidos por otros peatones donde chocan las cajas de delimitación. Este método está libre de cualquier restricción basada en cuadrícula en la localización espacial de las uniones y tiene la capacidad de estimar múltiples posturas que se ocluyen entre sí. La Figura 6 presenta la arquitectura del modelo. Es una red de base ResNet [32] compartida con dos redes de cabecera: una red predice la confianza, la ubicación precisa y el tamaño de una articulación lo que se denomina Campo de intensidad de Partes (PIF), y la otra red

predice las asociaciones entre las partes, denominado Campo de Asociación de Partes (PAF) es por ello que el método es denominado PifPaf.

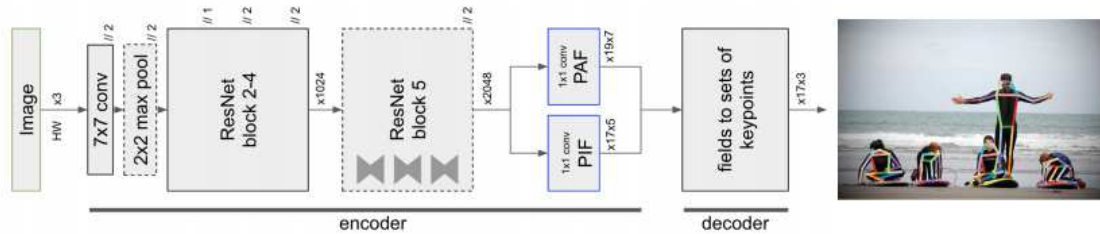


Figura 6: Arquitectura del modelo: La entrada es una imagen de tamaño (H,W) con 3 canales de color. La red neuronal basada en codificadores produce campos PIF y PAF con canales de 17x5 y 19x7. Una operación con paso dos se indica mediante “//2”. El decodificador es un programa que convierte los campos PIF y PAF en estimaciones de pose que contienen 17 uniones cada una. Cada articulación está representada por una coordenada x e y además de un puntaje de confianza [31].

El entrenamiento de 75 épocas en ResNet101 en dos tarjetas de video GTX1080Ti fue aproximadamente 95 horas. En las pruebas realizadas el método es superior a métodos como el OpenPose y el MaskR-CNN en más del 18%.

3.3. Sistemas de Seguimiento

Existen numerosos algoritmos propuestos en la literatura, sin embargo, el seguimiento de objetos en imágenes sigue siendo un problema difícil debido al cambio en la apariencia causada por cambios de pose, iluminación, oclusión y movimiento, entre otros. Los métodos que se describen a continuación demuestran que son robustos ante los problemas descritos con anterioridad y sobre todo el tiempo de respuesta de cada método permitirá ayudar a cumplir con los objetivos propuestos en esta tesis.

3.3.1. Método Compressive Tracking

En [4] se propone un algoritmo de seguimiento en base a un modelo de apariencia con características extraídas del espacio de características de la imagen multi-escala, donde hace uso de proyecciones aleatorias no adaptativas que preservan la estructura del espacio de características de los objetos de la imagen. Para extraer dichas características del modelo de apariencia de forma eficiente se utiliza una matriz de medida rara. Esta misma matriz es la encargada de comprimir las muestras del fondo del primer plano. La tarea de tracking se formula como una clasificación binaria a partir de un clasificador sencillo de Bayes con actualizaciones llevadas a cabo de forma online en el dominio comprimido. El algoritmo *Compressive Tracking* tiene muy buenos resultados cuando se producen oclusiones. Las principales componentes del algoritmo de *Compressive Tracking* se observan en la Figura 7.

El modelo de apariencia creado para [4] es de tipo generativo y discriminativo, generativo porque el objeto puede ser representado en base a características extraídas del

dominio comprimido y discriminativo por que se usan estas características para separar el objeto del fondo a través de un clasificador *naive Bayes*. En el modelo propuesto las características son seleccionadas mediante una reducción de dimensionalidad que preserva la información y es no adaptativa, a partir de las características de una imagen multi-escala basado en teorías de detección comprimidas explicadas en [33] y [34].

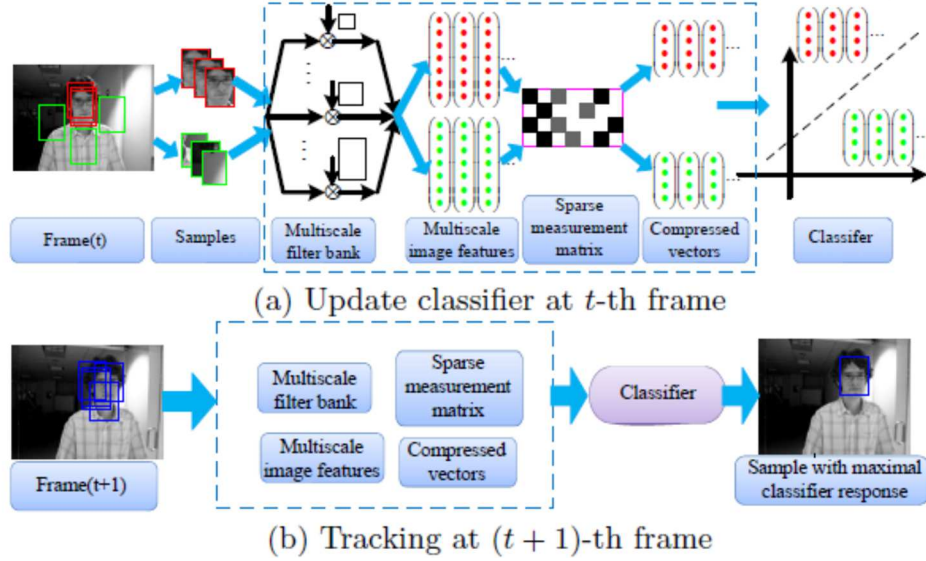


Figura 7: Componentes principales del algoritmo *Compressive tracking* [4].

Se ha demostrado que un pequeño número de mediciones generadas al azar puede preservar la mayor parte de la información de salida y permite casi una perfecta reconstrucción de la señal, si ésta es capaz de comprimirse, como las imágenes naturales o de audio [33] y [35]. Para esto se usa una matriz rala de mediciones que satisfacen la propiedad isométrica restringida (RIP) [36]; de esta manera, facilita la proyección del espacio de características de la imagen a un sub-espacio comprimido de baja dimensión.

A continuación, se presenta el algoritmo generado:

Algoritmo 1. *Compressive Tracking*

Entrada: t -th video frame

1. Muestras de un conjunto de sub-ventanas de la imagen, $D^Y = \{z \mid \|l(z) - l_{t-1}\| < \gamma\}$ donde l_{t-1} es la posición del *tracking* con el frame $(t-1)$, y se extraen las características de baja dimensionalidad.
2. Usa un clasificador basado en Naive Bayes del tipo

$$H(v) = \sum_{i=1}^n \log \left(\frac{p(v_i | y = 1)}{p(v_i | y = 0)} \right)$$

donde $p(v_i | y = 1)$ y $p(v_i | y = 0)$ en el clasificador $H(v)$ asumen que son distribuciones gaussianas con cuatro parámetros $(\mu_i^1, \sigma_i^1, \mu_i^0, \sigma_i^0)$ donde

$$p(v_i | y = 1) \sim N(\mu_i^1, \sigma_i^1), p(v_i | y = 0) \sim N(\mu_i^0, \sigma_i^0)$$

El cálculo se realiza para cada $v(z)$ y encuentra la ubicación l_t con la máxima

respuesta del clasificador.

3. Muestras de dos conjuntos de sub-ventanas de la imagen $D^\alpha = \{z \mid ||l(z) - l_t| < \alpha\}$ y $D^{\zeta, \beta} = \{z \mid \zeta < ||l(z) - l_t| < \beta\}$ con $\alpha < \zeta < \beta$.

4. Extraer las características a partir de estos dos conjuntos de muestras y actualizar los parámetros del clasificador de acuerdo a:

$$\mu_i^1 \leftarrow \lambda \mu_i^1 + (1 - \lambda) \mu^1$$

$$\sigma_i^1 \leftarrow \sqrt{\lambda (\sigma_i^1)^2 + (1 - \lambda) (\sigma^1)^2 + \lambda (1 - \lambda) (\mu_i^1 - \mu^1)^2}$$

Salida: posición del *tracking* l_t y los parámetros del clasificador.

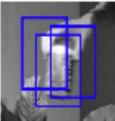
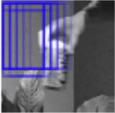
3.3.2. Método Tracking-by-Detection with Kernels

A pesar de que algunos parámetros permiten hacer supuestos sobre el objeto [37] y [38], a veces es deseable realizar el seguimiento de un objeto con un poco de conocimiento a priori. Los modelos de seguimiento se enfocan en el aprendizaje y adaptación de representaciones del objeto en línea.

Hay muchos enfoques exitosos basados en *tracking-by-detection* [39] y [40]. Muchos de estos algoritmos pueden ser adaptados para un entrenamiento en línea, donde cada detección exitosa provee más información del objeto.

Todos los métodos propuestos tienen algo en común: una estrategia de muestreo raro [39], [41] y [40]. En cada *frame*, varias muestras son obtenidas de la vecindad del objeto, en cada muestra se caracteriza una sub-ventana del mismo tamaño del objeto, estas diferencias entre métodos se observan en la Tabla 1. Claramente se observa una gran cantidad de redundancia, ya que la mayoría de las muestras presentan oclusiones entre sí.

Tabla 1. Principales diferencias entre los métodos estándar de *tracking-by-detection* y el enfoque [5].

		Almacenamiento	Cuello de botella	Velocidad
<p><i>Muestreo aleatorio</i> (sub-ventanas aleatorias p)</p> 	Características de las p sub-ventanas	Algoritmo de aprendizaje (Estructuras, SVM [34], Boost [32], [35],...)	10 - 25 FPS	
<p><i>Muestreo Densa</i> (todas las ventanas, método propuesto)</p> 	Características de una imagen	Transformada rápida de Fourier FFT	320 FPS	

En cambio, la mayoría de los métodos simplemente recogen un pequeño número de muestras, porque el costo de no hacerlo sería mayor.

El hecho de que el entrenamiento de datos tiene mucha redundancia significa que probablemente no se está explotando esta estructura eficientemente. En [5] se propone un nuevo *framework* teórico para abordar esto, basado en el aprendizaje con muestreo denso, donde el componente principal del *tracking-by-detection* es un clasificador, el cual es entrenado con todas las muestras. Se demuestra que el proceso de tomar sub-ventanas de imágenes induce a *estructuras circulantes* del tipo:

$$C(u) = \begin{bmatrix} u_0 & u_1 & u_2 & \dots & u_{n-1} \\ u_{n-1} & u_0 & u_1 & \dots & u_{n-2} \\ u_{n-2} & u_{n-1} & u_0 & \dots & u_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_1 & u_2 & u_3 & \dots & u_0 \end{bmatrix}$$

Donde la primera fila es el vector u , la segunda fila es u desplazado un elemento hacia la derecha, la siguiente es u desplazado dos elementos hacia la derecha y así sucesivamente. La motivación detrás de las matrices circulantes es que se puede codificar la convolución de vectores y ser calculadas en el dominio de Fourier usando la Ecuación (3):

$$C(u)v = \mathcal{F}^{-1}(\mathcal{F}^*(u) \odot \mathcal{F}(v)) \quad (3)$$

Así, se establece el uso de la Transformada rápida de Fourier (FFT) que incorpora rápidamente la información de todas las sub-ventanas, sin iterar sobre ellas.

Estos desarrollos permiten nuevos algoritmos de aprendizaje que pueden ser de mayor velocidad que los métodos estándar. También se ve que la clasificación no-lineal en el espacio de características puede ser resuelta de manera eficiente con el truco del *kernel*, en el espacio de la imagen. En la Figura 8 se observan los resultados obtenidos por el algoritmo desarrollado en [5].



Figura 8. Ejemplos con los resultados obtenidos por el método *Tracking by Detection with kernels*.

3.3.3. Método High-Speed Tracking with Kernelized Correlation Filters

Se trata de un filtro de correlación [7] que opera sobre características HOG. Las mejoras respecto a la versión anterior revisada en la sección 3.3.2 son soporte de múltiples escalas, estimación de peaks máximos por bloques o celdas y reemplazo de las actualizaciones del modelo mediante interpolación lineal con un esquema de actualización más robusto. El mayor problema de *Kernelized Correlation Filters* KCF está relacionado con la robustez del algoritmo. En términos de exactitud muestra buenos resultados para cambios de escala, de iluminación, de movimiento, oclusiones y movimiento de la cámara. Sin embargo, tanto en oclusiones como en cambios de escala, tiene problemas a la hora de seguir al objetivo. En la Figura 9 se muestra una evaluación del método KCF versus otros métodos explicados en [7].

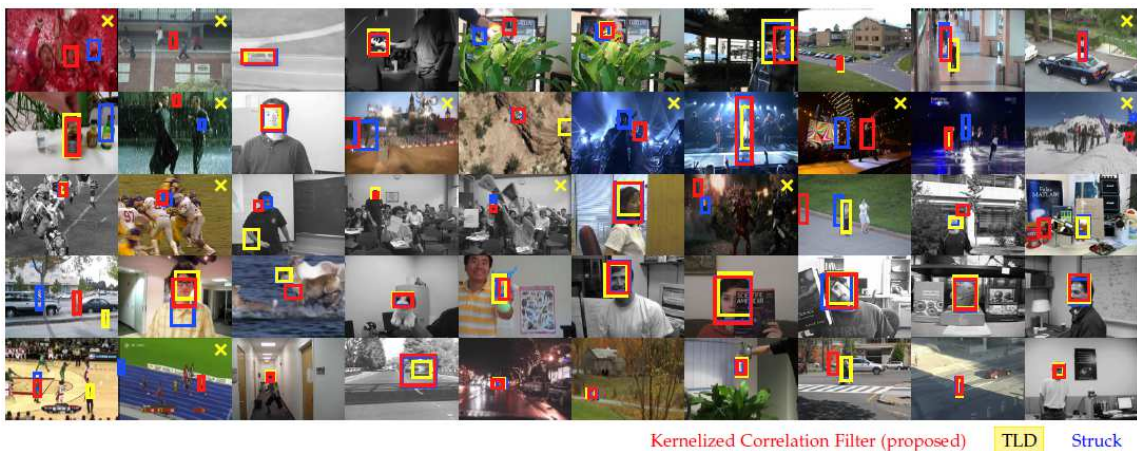


Figura 9: Resultados cualitativos para el high-speed tracking with kernelized correlation filters propuesto en [7], comparado con el Struck y TLD. El kernel elegido es gaussiano, en las características de HOG.

En el trabajo anterior [5], se demostró por primera vez la conexión entre Ride Regresión con muestras cíclicamente desplazadas y filtros de correlación clásicos. Esto permitió un aprendizaje rápido con un costo computacional de $O(n \log n)$, es decir, transformaciones rápidas de Fourier en lugar de álgebra matricial costosa, también se propuso el primer filtro de correlación kernelizado, aunque limitado a un solo canal.

En [7] se extiende el trabajo original para tratar con múltiples canales, lo que permite el uso de características de última generación que dan un impulso importante al rendimiento. También se amplía los experimentos originales de 12 a 50 videos y se agrega una nueva variante del seguidor KCF basado en las características del histograma de gradientes orientados HOG en lugar de píxeles sin procesar.

Por otro lado, se propone un filtro lineal multicanal con muy baja complejidad computacional que casi coincide con el rendimiento de kernels no lineales, el cual denominan Dual Correlation Filter (DCF).

Experimentalmente se demuestra que KFC funciona mejor que un filtro lineal sin extracción de características, en la que emplea características de HOG y se puede observar que tanto el DCF lineal como el KCF no lineal superan el rendimiento de los métodos de seguimiento como Struck [42] o el Tracking-Learning-Detect [43]. Y este método se ejecuta a cientos de imágenes por segundo.

3.3.4. Método Mean Shift

Es un método de proceso iterativo de punto fijo que converge a un máximo local, y en cada iteración se estima el gradiente normalizado de la función de densidad de probabilidad en el espacio de características elegido, por lo general la característica elegida es el color, podrían usarse otras variables como textura, borde o una mezcla de ambas.

Por lo tanto, si se considera un vector x , el método consiste en iterar. En cada una de estas iteraciones se deberá encontrar el nuevo vector x' . Si se considera esto como un incremento $x' = x + \Delta_x$, cada iteración se basará únicamente en la búsqueda de este incremento.

La búsqueda de este valor se basa en el acercamiento al modelo de densidad que define al propio objeto, mediante la siguiente ecuación:

$$\Delta_x = \frac{\sum_a K(a - x)w(a)a}{\sum_a K(a - x)w(a)} - x = \frac{\sum_a K(a - x)w(a)(a - x)}{\sum_a K(a - x)w(a)}$$

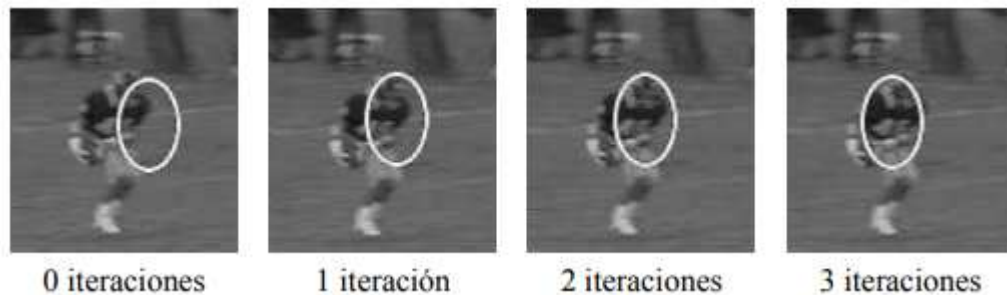


Figura 10: Tres iteraciones del Mean Shift [6].

Donde a son los pixeles que corresponden al área de búsqueda del Mean Shift la cual suele ser del mismo tamaño que el modelo del objeto, K es una función que ofrece la información del núcleo, normalmente una gaussiana, y $w(a)$ un peso asignado en función de la información cromática del pixel.

Se debe iterar en este método repetidas veces, efectuando la actualización pertinente $x' = x + \Delta_x$ a cada iteración. Los resultados obtenidos al realizar este proceso serán similares a los que muestra la Figura 10, para cada iteración del método. En dicha figura se puede observar cómo cada iteración el modelo se aproxima más a la solución correcta, llegando finalmente a ella.

El método es robusto, pero tiene sus limitaciones. El hecho de tener que trabajar en un área pequeña de búsqueda hace que únicamente sirva en aquellas ocasiones en las que los objetos no se solapan de un frame a otro, ya que en frames-rates bajos, se pierde el tracking.

3.4. Detección y Seguimiento de Características Locales

Durante el proceso de detección de características se busca detectar puntos que sean fácilmente distinguibles y comparables en una imagen. Los detectores de características se dividen en detectores de esquinas y detectores de blobs. Una esquina se define como un punto que se sitúa en la intersección de dos o más bordes, en cambio un blob es una región de la imagen que claramente se diferencia de su entorno en términos de intensidad, color y textura, por lo tanto, un blob no es una esquina ni tampoco un borde.

Las propiedades de un buen detector de características en una imagen deberían ser: precisión en la posición dentro de la imagen, repetitividad (una característica existente en dos imágenes consecutivas debería ser detectada en ambas imágenes), eficiencia en tiempos de ejecución, robustez ante ruido, ante imágenes desenfocadas, distinción (es decir que las características sean claramente identificables entre ellas), variación nula ante cambios en la luminosidad y ante cambios geométricos como puede ser rotación, escalado y distorsiones de perspectiva detallados en [44].

Para la extracción y seguimiento de características de las imágenes se ha utilizado el detector de características locales de la imagen de Shi-Tomasi [1]. Este algoritmo está basado en la detección de esquinas de Harris [45], en el cual las esquinas son regiones dentro de una imagen que contiene una gran variación en la intensidad, en todas las direcciones. El primer intento por conseguir detectar estas esquinas vino de la mano de [45] que dio nombre al detector “Harris Corner Detector” que se basa en una idea matemática sencilla, donde se busca la diferencia entre las intensidades que se generan en un desplazamiento de (u, v) en todas sus direcciones detallado en la Ecuación (4).

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2 \quad (4)$$

Para la detección de las esquinas se debe maximizar $E(u, v)$. Por ello, aplicando Taylor a la Ecuación (4) obtenemos la ecuación final (ver Ecuación (5)) como:

$$E(u, v) = [u \ v] M \begin{bmatrix} u \\ v \end{bmatrix} \quad (5)$$

Donde:

$$M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \quad (6)$$

Aquí, I_x e I_y son las derivadas de la imagen en las direcciones x e y , respectivamente.

Tras esto, viene la parte principal del algoritmo donde se crea la Ecuación (7), que determinará si una ventana puede contener una esquina o no:

$$R = \det(M) - k(\text{trace}(M))^2 \quad (7)$$

Donde:

- $\text{Det}(M) = \lambda_1 * \lambda_2$
- $\text{trace}(M) = \lambda_1 + \lambda_2$
- λ_1 y λ_2 son los valores eigen (valor propio) de M

Con los valores de λ_1 y λ_2 de M se puede decidir si una región es esquina, borde o plana:

- Cuando $|R|$ es pequeño, es decir λ_1 y λ_2 son pequeños, la región es plana.
- Cuando $R < 0$, es decir, $\lambda_1 \gg \lambda_2$, o viceversa, la región es borde.
- Cuando R es grande, es decir, λ_1 y λ_2 son grandes, la región es una esquina.

En 1994 J.Shi y C. Tomasi [1] realizan pequeñas modificaciones y consiguen mejores resultados, en comparación a Harris [45]. Este algoritmo propone los siguientes cambios con respecto al anterior ver Ecuación (8):

$$R = \min(\lambda_1, \lambda_2) \quad (8)$$

Si el valor que se obtiene es mayor que el umbral, es una esquina.

3.5. Modelos Ocultos de Markov

El Modelo Oculto de Markov (HMM) esta descrito en detalle en [46], sin embargo en esta sección se explicará la teoría de aquellos puntos que se consideraron en la construcción de esta tesis.

3.5.1. Procesos Discretos de Markov

Se considera que un sistema puede ser descrito en cualquier instante de tiempo, es decir que este puede encontrarse en N estados distintos S_1, S_2, \dots, S_n . En instantes discretos de tiempo, el sistema sufre un cambio de estado de acuerdo con un conjunto de probabilidades asociadas a este estado. Se conocen los instantes de tiempo asociados con los cambios de un estado como $t = 1, 2, \dots$, y se llama al estado en el tiempo t como q_t . Una descripción probabilística completa del sistema mencionado, generalmente requiere la especificación del estado actual (en el tiempo t) como de los estados anteriores.

3.5.2. Elementos de un HMM

Un HMM se caracteriza por lo siguiente:

1. N , es el número de estados en el modelo. A pesar de que los estados están ocultos, para muchas aplicaciones prácticas hay algún significado físico adherido a los estados o a conjuntos de estados del modelo. Generalmente los estados están interconectados entre ellos de tal manera que se puede alcanzar un estado desde cualquier otro. Se denominan los estados individuales como $S = \{S_1, S_2, \dots, S_N\}$.

2. M , es el número de símbolos de observaciones distintas por estado, por ejemplo, el tamaño discreto del alfabeto. Los símbolos de las observaciones corresponden a la salida física del sistema siendo modelado. Se denominan los símbolos individuales como $V = \{V_1, V_2, \dots, V_M\}$.

3. La distribución de las probabilidades de transición de estado $A = \{a_{i,j}\}$ donde,

$$a_{i,j} = P[q_{t+1} = S_j], 1 \leq i, j \leq N \quad (9)$$

Para el caso especial donde cualquier estado puede llegar a cualquier otro en un paso, tenemos que $a_{i,j} = 0$ para uno o más de un par i, j .

4. La distribución de las probabilidades de los símbolos de observaciones en el estado $j, B = \{b_j(k)\}$, donde:

$$b_j(k) = P[V_k \text{ at } t | q_t = S_j], \quad \begin{matrix} 1 \leq j \leq N \\ 1 \leq k \leq M \end{matrix} \quad (10)$$

5. La distribución del estado inicial $\pi = \{\pi_i\}$, donde,

$$\pi_i = P[q_1 = S_i], 1 \leq i \leq N \quad (11)$$

Dados los valores apropiados de N, M, A, B y π , el HMM puede ser utilizado para proveer una secuencia de observaciones,

$$O = O_1, O_2, \dots, O_t \quad (12)$$

Donde cada observación O_t es uno de los símbolos de V , y T es el número de observaciones en la secuencia.

Existe una estrategia formal para encontrar la mejor secuencia de estados y está basada en Programación Dinámica denominada Algoritmo de Viterbi [47] y [48].

3.5.3. Algoritmo de Viterbi

Para encontrar la mejor secuencia de estados $Q = \{q_1, q_2, \dots, q_T\}$, para la secuencia de observaciones dada $O = O_1, O_2, \dots, O_t$, se necesita definir:

$$\delta_t(i) = \max_{q_1, q_2, \dots, q_{t-1}} P[q_1 q_2 \dots q_t = i, O_1, O_2, \dots, O_t | \lambda] \quad (13)$$

Por ejemplo, si $\delta_t(i)$ es la mayor probabilidad a través de una trayectoria, en el tiempo t , que toma en cuenta las primeras t observaciones y acaba en el estado S_i .

Por lo que tenemos,

$$\delta_t(i) = \left[\max_j \delta_t(j) a_{j,i} \right] \cdot b_j(O_{t+1}) \quad (14)$$

Para obtener realmente la secuencia de estados hay que conocer el índice que maximizó a la Ecuación (14) para cada t y j . Esto se hace con la variable $\psi_t(j)$.

El procedimiento completo para encontrar la mejor secuencia de estados se plantea a continuación:

1. Inicialización

$$\begin{aligned}\delta_1(i) &= \pi_i b_i(O_1), \quad 1 \leq i \leq N \\ \psi_1(i) &= 0\end{aligned}\tag{15}$$

2. Recursión

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{i,j}] b_j(O_t), \quad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq M \end{matrix}\tag{16}$$

$$\psi_t(j) = \operatorname{argmax}_{1 \leq i \leq N} [\delta_{t-1}(i) a_{i,j}], \quad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq M \end{matrix}\tag{17}$$

3. Terminación

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)]\tag{18}$$

$$q_T^* = \operatorname{argmax}_{1 \leq i \leq N} [\delta_T(i)]\tag{19}$$

4. Backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T - 1, T - 2, \dots, 1\tag{20}$$

3.6. Random Sample Consensus

El algoritmo RANSAC (RANdom SAMple Consensus) introducido por [2] es el algoritmo de estimación robusta más utilizado. Consiste en realizar un ajuste robusto de un modelo, con un conjunto de datos S , que contiene valores atípicos u outliers.

Para comprender el funcionamiento de este algoritmo, primero se ilustrará un caso particular y posteriormente se mostrará el caso general.

El funcionamiento del algoritmo, para ajustar una línea recta a un conjunto de puntos, es el siguiente:

1. Se selecciona de forma aleatoria dos puntos del conjunto S . Estos dos puntos definen una recta. En la Figura 11 se seleccionaron los puntos a y b que definen la recta dibujada.
2. Se evalúa el consenso de la recta, es decir el número de puntos compatibles con esa recta. Para ello, se calcula cuántos de estos puntos están a una distancia de la recta menor que un umbral previamente fijado (línea discontinua). Estos puntos se denominan inliers o puntos de consenso. Los puntos quedan por encima del umbral se llaman outliers.

- Esta selección aleatoria se repite un número de veces y la recta con más inliers se considera la recta robusta y se elige como modelo final.

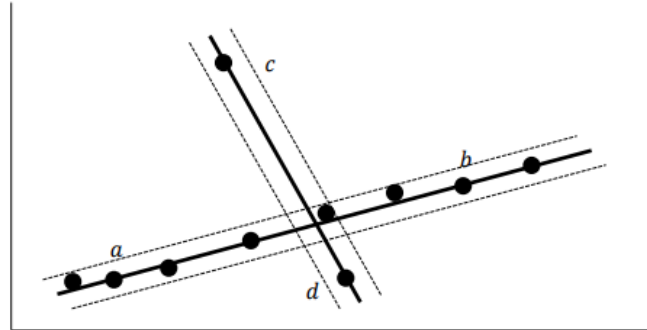


Figura 11. La recta robusta es la formada por los puntos a y b. Todo el conjunto de puntos, salvo dos *outliers* (c y d), forman el denominado conjunto de consenso.

En muchos casos en los que no es posible ajustar un modelo con dos únicos puntos, como en el caso anterior, se utiliza el caso general de RANSAC. El algoritmo general sigue los siguientes pasos [2]:

- Se selecciona aleatoriamente una muestra s de puntos, del conjunto total S , y con este subconjunto se calcula el modelo de transformación.
- Se determina el subconjunto de puntos S_i que está dentro del umbral $U1$ de distancia al modelo calculado en el punto anterior. El subconjunto S_i es el conjunto consenso y contiene los *inliers* del conjunto de puntos S .
- Si el número de inliers del subconjunto S_i es mayor que un segundo umbral $U2$ se vuelve a estimar el modelo utilizando todos los puntos de S_i y se termina el algoritmo.
- Si el número de inliers de S_i es menor que el umbral $U2$ se selecciona un nuevo subconjunto S_i y se repite el paso anterior.
- Después de N intentos se selecciona el subconjunto S_i con mayor consenso. El modelo se recalcula utilizando todos los puntos del subconjunto S_i .

En la Figura 12 se muestra una corrección del outliers con RANSAC, aplicada a correspondencia de puntos. En azul se puede ver un conjunto de puntos en el instante de tiempo i , y en rojo sus correspondencias en el instante de tiempo $i + 1$. Mediante RANSAC calculamos el modelo de consenso (la línea azul muestra los inliers y la línea roja muestra los outliers), y aplicamos esa transformación a todos los puntos.

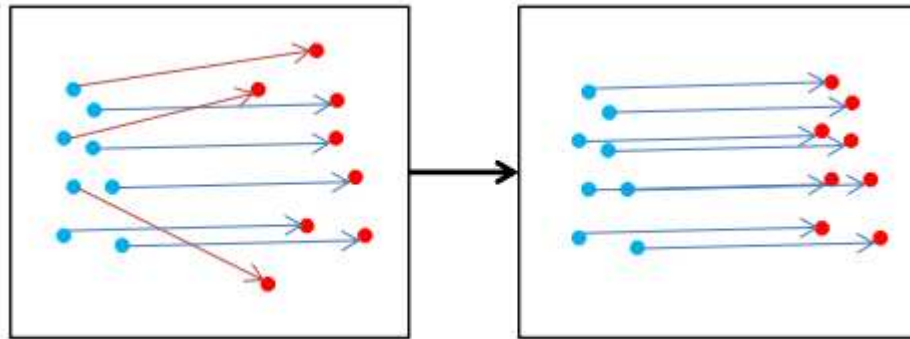


Figura 12. Correspondencias entre dos conjuntos de puntos. Transformación de todos los puntos siguiendo el modelo de consenso calculado por RANSAC.

Capítulo 4

Metodología Propuesta

En este trabajo se propone un método que es capaz de hacer tracking a una persona mientras se mueve en el sistema de referencia de la cámara. Este movimiento relativo puede ser producido por el movimiento del sensor (por ejemplo, la cámara de un robot) o por el movimiento de la persona. El sistema de tracking propuesto es usado para la aplicación de seguimiento de personas por medio de un robot móvil, pero también se puede usar en cámaras de vigilancia. El método es capaz de mejorar el rendimiento de un detector de personas de cuatro maneras:

1. Es capaz de proyectar la detección calculada en una imagen anterior a la actual.
2. Puede usar información temporal para rechazar detecciones falsas.
3. Es capaz de detectar oclusiones, descartando las detecciones generadas por un objeto que ocluye el objeto seguido.
4. Es capaz de propagar las detecciones en el tiempo y detectar el inicio y fin de las oclusiones.

Básicamente el método inicia con las imágenes capturadas por la cámara. Luego un detector de objetos se inicializa con la primera imagen capturada, sin embargo, este detector funciona a un frame rate muy bajo (0,5 fps). Una vez realizada la primera detección se inicializa un sistema de seguimiento con los parámetros de la primera detección. Aunque el detector realiza detecciones en intervalos grandes de tiempo (podrían pasar 25 frames entre una detección y la siguiente), el sistema de seguimiento funciona en tiempo real (23 fps), por lo que cada detección atrasada se puede usar para corregir y actualizar los parámetros del sistema de seguimiento.

Para la base del sistema de seguimiento se vio la posibilidad de usar métodos como SIFT o SURF, sin embargo las características generadas por estos métodos pueden aparecer y desaparecer entre secuencias de imágenes y dificultan realizar el seguimiento, por otro lado en el método Kanade-Lucas-Tomasi (KLT) [1] los tracks de características no desaparecen y permiten una estabilidad de las mismas, es por ello que se eligió el método KLT.

El sistema de seguimiento está basado en Kanade-Lucas-Tomasi (KLT) [1] que se usa para realizar el seguimiento de esquinas, esto lo hace a través de puntos generados en las imágenes durante el proceso de detección, los cuales, se utilizan para calcular una transformación libre de retardo que permita proyectar un cuadro delimitador de detección de la imagen anterior a la actual, el grupo de imágenes que se encuentran dentro del intervalo de detecciones y el resultado generado por el detector de objetos se

denomina *grupo de imágenes* GOF (group of frames) por sus siglas en inglés y se define como: $GOF^k = \{I_1^k, \dots, I_{n_k}^k\}$, siendo k el número correlativo de la detección (ver Figura 13). La cantidad de imágenes en cada GOF es variable esta puede ser menor o mayor dependiendo del retardo del detector.

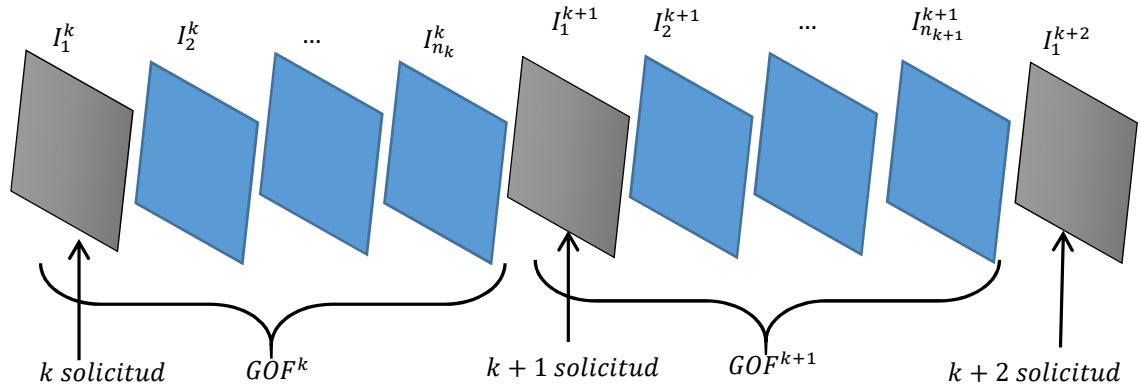


Figura 13. Diagrama del concepto de *grupo de imágenes* (GOF). El proceso de detección de objetos dentro de una imagen demora un tiempo considerable. El set de imágenes que existe entre detección y detección se denominará GOF.

La Figura 14 muestra las etapas que componen el método propuesto en la tesis:

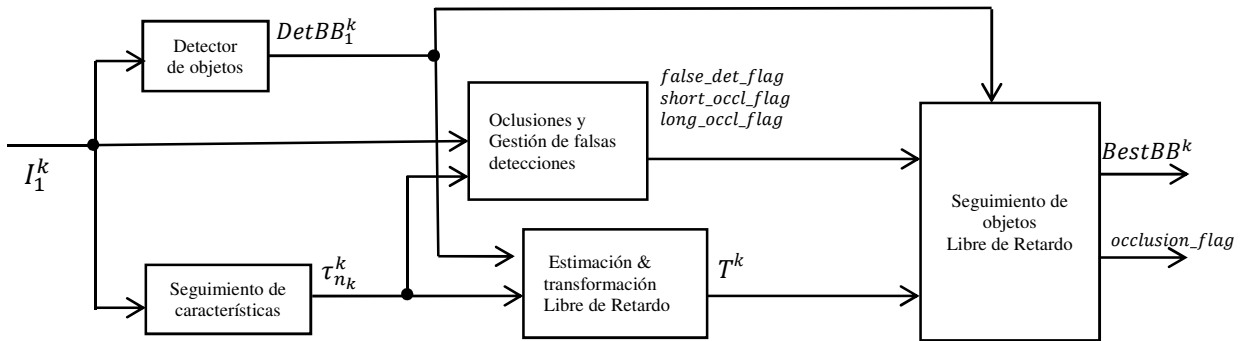


Figura 14: Módulos que componen el método de seguimiento propuesto.

La función de cada módulo mostrado en la Figura 14 se detalla a continuación:

Detección de objetos: Este módulo se encarga de analizar una imagen I_1^k en el orden que se va generando la secuencia de imágenes. En caso de que el objeto sea detectado se genera un cuadro delimitador denominado $DetBB_1^k$ como la salida de este bloque. El cuadro delimitador indica la posición del objeto en la imagen I_1^k , dado que el proceso de detección requiere algo de tiempo para generar el resultado el módulo ignora todas las

nuevas imágenes que se van generando durante el proceso de detección. Estas nuevas imágenes se denominarán intra-imágenes I_u^k . Para el propósito de esta tesis se usa los dos detectores HOG y HOG1/2.

Seguimiento de características: Este módulo toma como entrada la imagen I_1^k y calcula las características (esquinas) que pueden ser seguidas en la siguiente intra-imagen. Para cada intra-imagen se actualizan las posiciones de las esquinas detectadas en el último cuadro y se generan tracks (puntos de seguimiento) de características $\tau_{n_k}^k$ que contienen la trayectoria de las posiciones de cada imagen. Cuando se realiza la detección de objetos, las características $\tau_{n_k}^k$ que están contenidas en el cuadro delimitador se van transmitiendo a los siguientes módulos.

Estimación y transformación libre de retardo: Este módulo toma las resultantes de los módulos anteriores $DetBB_1^k$ generada para la imagen I_1^k y el set de tracks de características $\tau_{n_k}^k$ como entrada. $\tau_{n_k}^k$ contiene la posición de las esquinas seguidas a lo largo de las intra-imágenes, esto se puede usar para calcular una transformación T^k que mapea el cuadro delimitador de detección original $DetBB_1^k$ sobre uno libre de retardo $DetBB_{n_k}^k$. La salida de este módulo es la transformación T^k .

Oclusiones y gestión de falsas detecciones: Este módulo usa como entradas la imagen I_1^k , el track de características $\tau_{n_k}^k$ y el cuadro delimitador $DetBB_1^k$ que se encarga de estimar la presencia de falsas detecciones (*false_det_flag*) y oclusiones cortas (*short_occl_flag*) y largas (*long_occl_flag*).

Seguimiento de objetos libre de retardo: Este módulo se encarga de generar el cuadro delimitador final del objeto $BestBB^k$ considerando la transformación T^k que se necesita aplicar sobre $DetBB_1^k$ y el grado de confianza de la detección respecto a la presencia de falsas detecciones y oclusiones. La salida de este módulo es el mejor cuadro delimitador y una bandera que indica si el objeto está ocluido o no.

Por lo tanto, el sistema puede generar detecciones libres de retardo que son robustas ante los errores de detección y las oclusiones. En el caso de que se realice el seguimiento de múltiples objetos, el módulo de seguimiento de características debe ejecutarse solo una vez por imagen. El cálculo de la transformación libre de retardo es muy rápido (requiere 0.07 milisegundos en un Intel Core i5 usando solo un núcleo), por lo que el algoritmo se puede escalar fácilmente para seguir varios objetos. Los objetos sin textura pueden causar escasez de tracks de características. Sin embargo, este no es un problema importante, dado que los tracks de características se calculan y actualizan solo entre dos detecciones consecutivas. Además, los objetos sin textura se pueden seguir con éxito mediante el uso de las características de sus bordes.

4.1. Estimación y Transformación Libre de Retardo

Este módulo transforma una detección retrasada en una detección libre de retardo. La detección inicia en la primera imagen I_1^k del GOF^k y termina en la última imagen $I_{n_k}^k$. El superíndice indica el número de GOF y el subíndice es usado para indicar el orden de la imagen que está dentro del GOF como se observa en la Figura 13. Entonces cuando la imagen I_1^k está disponible el detector de objetos inicia en esa imagen. Adicionalmente el algoritmo detector de esquinas de shi-tomasi inicia con los parámetros entregados por el detector en la imagen I_1^k . El detector de objeto puede requerir un tiempo considerable para realizar su tarea, mientras la tarea de detección se realiza varias intra-ímagenes I_u^k están disponibles, por lo cual para cada intra-imagen la posición de las esquinas detectadas en I_1^k son actualizadas y denominadas *tracks de características* que contienen las trayectorias de cada posición de esquinas de las imágenes consecutivas que se van generando. El set de tracks de características está compuesto por N tracks de características individuales $\tau_u^k[i]$ definido como:

$$\tau_u^k[i] = (x_1^k, y_1^k, time_1^k, \dots, x_u^k, y_u^k, time_u^k) \quad (21)$$

Donde x_i, y_i representan la posición de la i – ésima esquina en el tiempo $time_i$.

Cuando el detector termina la tarea (en la imagen $I_{n_k}^k$), el track de características $\tau_{n_k}^k$ está contenido en el cuadro delimitador de detección $DetBB_1^k$ y estas características son usadas para obtener la transformación T^k que permite habilitar la proyección de $DetBB_1^k$ a un nuevo cuadro delimitador $DetBB_{n_k}^k$, referida a la última imagen $I_{n_k}^k$. Una transformación de escala y traslación es usada para relacionar los dos cuadros delimitadores. Esta transformación está determinada por la traslación t_x, t_y y un factor de escala s . La ecuación usada para calcular la transformación de una muestra mínima está compuesta por dos tracks de características $\tau_{n_k}^k[a]$ y $\tau_{n_k}^k[b]$ es:

$$s = \frac{\sqrt{(\tau_{n_k}^k[a].x_{n_k} - \tau_{n_k}^k[b].x_{n_k})^2 + (\tau_{n_k}^k[a].y_{n_k} - \tau_{n_k}^k[b].y_{n_k})^2}}{\sqrt{(\tau_{n_k}^k[a].x_1 - \tau_{n_k}^k[b].x_1)^2 + (\tau_{n_k}^k[a].y_1 - \tau_{n_k}^k[b].y_1)^2}} \quad (22)$$

$$t_x = \tau_{n_k}^k[a].x_{n_k} - s * \tau_{n_k}^k[a].x_1 \quad (23)$$

$$t_y = \tau_{n_k}^k[a].y_{n_k} - s * \tau_{n_k}^k[a].y_1 \quad (24)$$

La transformación T^k es calculada usando el algoritmo de RANSAC [2].

El procedimiento para realizar este cálculo corresponde al Algoritmo 2 ilustrado en la Figura 15. En cada iteración del algoritmo un par del tracks de características es seleccionado aleatoriamente y usado para calcular una hipótesis de transformación T . Si más tracks de características son compatibles con la transformación T la hipótesis es aceptada, sino un nuevo par de características es seleccionado, el proceso se repite. La variable *maxNumIter* corresponde al número máximo de iteraciones en RANSAC. En esta tesis el seguimiento libre de retardo (*delay-free*) es usado para el seguimiento de personas. Sin embargo, el proceso de transformación libre de retardo puede ser aplicado a cualquier problema de detección de objetos.

Algoritmo 2: Algoritmo basado en RANSAC usado para el proceso de estimación y transformación libre de retardo (Delay-free).

Función *ransac_estimator*($DetBB_1^k, \tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N]$):

1. Calcular Q como el subconjunto de tracks de características que son contenidos dentro de $DetBB_1^k$
2. **Para** $i \leftarrow 1$ **a** *maxNumIter*
3. Elegir dos tracks de características $\tau_{n_k}^k[a]$ y $\tau_{n_k}^k[b]$ de Q
4. Calcular una hipótesis de transformación $T = (s, t_x, t_y)$ que relacione $\tau_{n_k}^k[a], \tau_{n_k}^k[b]$ entre ambas imágenes usando las ecuaciones (22), (23), (24)
5. Calcular $Q_{consensus}$, i.e., los elementos en Q que son compatibles con T
6. **Si** $size(Q_{consensus}) > threshold$ **entonces**
7. **Para** $j \leftarrow 1$ **a** *maxNumIter*
8. Elegir dos tracks de características random $\tau_{n_k}^k[a]$ y $\tau_{n_k}^k[b]$ de $Q_{consensus}$
9. Calcular una hipótesis de transformación $T = (s, t_x, t_y)$ que relacione $\tau_{n_k}^k[a], \tau_{n_k}^k[b]$ entre ambas imágenes usando las ecuaciones (22), (23), (24)
10. $S[j] \leftarrow s, \quad T_x[j] \leftarrow t_x, \quad T_y[j] \leftarrow t_y$
11. **fin**
12. seleccionar $S[imed]$ como la media de S
13. **retornar** $T = (S[imed], T_x[imed], T_y[imed])$
14. **fin**
15. **fin**
16. RANSAC falló; **retornar** $T = identity_transform$



Figura 15: Ejemplo de la detección delay-free. Las imágenes de izquierda a derecha representan detecciones de imágenes consecutivas, con diferentes tracks de características calculados. La etiqueta de la posición real del objeto GT (ground-truth) está representado por el recuadro verde. El retraso de la detección está representado por el recuadro rojo. La detección libre de retardo está representada por el recuadro azul. Los tracks de características se visualizan como líneas en la imagen. Los tracks de características aceptados por RANSAC y usados para calcular la transformación libre de retardo (delay-free) se visualizan en líneas amarillas.

4.2. Oclusiones y gestión de falsas detecciones

El método propuesto estima la ocurrencia de las siguientes situaciones:

1. Una falsa detección del objeto en I_1^k ocurre cuando el objeto se encuentra junto a objetos de la misma categoría.
2. El objeto trackeado esta ocluido por un segundo objeto de la misma categoría (e.g. una persona esta ocluida por una segunda persona) en I_1^k y el segundo objeto es detectado. Esta situación se denomina una oclusión larga.
3. El objeto seguido está ocluido por un segundo objeto en una intra-imagen de la secuencia de imágenes I_u^k , por lo tanto, el track de características está corrupto. A esta situación se denomina oclusión corta.

En otras palabras, el caso 1 corresponde a una falsa detección, el caso 2 corresponde a una oclusión larga donde el objeto seguido desaparece y la cámara solo observa al objeto ocluidor, esto ocurre mientras se realiza la detección en la imagen y el caso 3 corresponde a una oclusión corta donde el objeto ocluido se encuentra dentro del intervalo de detecciones. En los primeros dos casos el resultado de la detección de objetos es descartada y se usa la detección previa es decir $BestBB^k = BestBB^{k-1}$. En el tercer caso los tracks de características son inválidos. Por lo tanto, la transformación libre de retardo no puede ser calculada. Entonces la detección en I_1^k no es proyectada y se usa como la detección final $BestBB^k = DetBB_1^k$. Tomar en consideración que el sistema de detección inicializa con la persona o el objeto en frente de la cámara.

4.3. Gestión de Falsas Detecciones

El método estima la ocurrencia de una falsa detección mediante la comparación de detecciones correctas, después de aplicar la transformación libre de retardo, $DetBB_1^k$ con la detección previa $DetBB_1^{k-1}$. En caso de que la diferencia entre ambos (en apariencia o posición) es larga entonces $DetBB_1^k$ es considerado como una falsa detección y se usa la detección anterior.

El problema se modela como un problema de clasificación de dos clases: donde dos detecciones consecutivas pueden ser similares o dos detecciones consecutivas pueden ser diferentes. Un clasificador estadístico es entrenado usando ejemplos reales de pares de detecciones similares y pares de detecciones diferentes. Para esta tesis se usa como clasificador el SVM con un kernel RBF porque tiene mayor estabilidad y su entrenamiento es relativamente simple, pero se podría usar cualquier otro clasificador.

Inicialmente se consideraron 12 características: el vector de características que incluye dos LBP (distancia entre dos frames), el centro de masa (x, y) del primer frame, y el centro de masa del segundo frame (x, y) , por lo cual se evaluaron varios subconjuntos de características usando la validación cruzada (10-fold) y el clasificador RBF SVM. Se usaron las 7 características que dieron el mejor subconjunto probado en estas validaciones y se descartaron las otras características ya que no aportan a la tarea de detectar falsos positivos.

El vector final de características con 7 componentes se usa para entrenar el clasificador estadístico, los primeros dos componentes están relacionados al contenido del color de los cuadros delimitadores de $DetBB_1^{k-1}$ y $DetBB_1^k$, y los otros 5 componentes están basados en la información de posición, el clasificador está disponible para usar la información de ambas fuentes para discriminar entre una verdadera y falsa detección. Los 7 componentes/características son calculados por una función denominada $comp_feat_{falseDet}()$ y los componentes son:

- distancia euclideana entre el histograma RGB de $DetBB_1^{k-1}$ y $DetBB_1^k$,
- distancia euclideana entre el histograma HSV de $DetBB_1^{k-1}$ y $DetBB_1^k$ (solo se usan los canales H y S),
- diferencia de posición en el eje x entre $DetBB_1^{k-1}$ y $DetBB_1^k$,
- diferencia de posición en el eje y entre $DetBB_1^{k-1}$ y $DetBB_1^k$,
- ratio entre las áreas de $DetBB_1^{k-1}$ y $DetBB_1^k$,
- el área del recuadro BB (Bounding Box) correspondiente a $DetBB_1^k$, y
- el puntaje de la intersección entre $DetBB_1^{k-1}$ y $DetBB_1^k$:

$$score(BB_1, BB_2) = \frac{intersection(BB_1, BB_2)}{union(BB_1, BB_2)} \quad (25)$$

En caso de que exista más de una detección disponible en la imagen, el módulo de detección de falsas detecciones se aplica a todas las detecciones. La gestión de falsas detecciones es un paso muy importante ya que, si no el sistema realizaría el seguimiento a otro objeto, los resultados obtenidos demuestran que las detecciones falsas se pueden detectar de forma robusta utilizando el procedimiento descrito.

Gestión de oclusiones cortas

Las oclusiones cortas ocurren dentro de dos detecciones consecutivas, es decir dentro de un GOF^k (grupo de imágenes), donde la primera imagen I_1^k no se encuentra ocluida., pero una intermedia si lo está. Este tipo de eventos hacen que el seguimiento sea poco confiable para la detección libre de retardo.

El hecho de que se realice el seguimiento de tracks a través de las intra-imágenes hace que sea posible analizar y predecir oclusiones cortas, es decir, a medida que se ocluye el objeto seguido los tracks de seguimiento desaparecen y se generan nuevas para el ocluidor dependiendo de la dirección de movimiento del ocluidor. Por lo tanto la posición horizontal x y el tiempo t en el que cada uno desaparece pueden ser almacenados como pares (x, t) y mediante el uso de mínimos cuadrados se puede generar una línea recta donde la pendiente m , el coeficiente de correlación r y la relación de las características perdidas p con respecto al inicio pueden ser utilizados en la construcción de un vector de características (m, r, p) , el parámetro de la línea recta n no se utiliza, ya que se relaciona únicamente con la posición de la caja de contorno. Para calcular el modelo de mínimos cuadrados, se utilizan las siguientes ecuaciones:

$$m = \frac{\sum(x_i - \bar{x})(t_i - \bar{t})}{\sum(x_i - \bar{x})^2} \quad (26)$$

$$r = \frac{\sum(x_i - \bar{x})(t_i - \bar{t})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(t_i - \bar{t})^2}} \quad (27)$$

$$p = \frac{\text{size}\{(x, y, t) \text{ with } t = t_{last}\}}{\text{size}\{(x, y, t) \text{ with } t \geq 1\}} \quad (28)$$

Un clasificador lineal asume que la detección es correcta cuando $\|m\| > m_{umbral}$, $\|r\| > r_{umbral}$ and $p < p_{umbral}$. Después de realizar varios experimentos en esta tesis, los umbrales que mejor resultado dieron se determinaron como 0,005, 0,3 y 0,3 respectivamente. Este procedimiento se denomina *threshold_classifier*, y se detalla en el Algoritmo 3.

Algoritmo 3: Threshold-based clasificador para detectar oclusiones cortas.

Función $threshold_classifier(\tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N])$:

1. Definir $nLastFrame = 0$
2. **Para** $i \leftarrow 1$ **a** N
3. $x[i] \leftarrow \tau_{n_k}^k[i].x_{last}$
4. $t[i] \leftarrow \tau_{n_k}^k[i].time_{last} - \tau_{n_k}^k[i].time_1$
5. **Si** $\tau_{n_k}^k[i].time_{last} = lastFrame$ **entonces**
6. $nLastFrame \leftarrow nLastFrame + 1$
7. **fin**
8. **fin**
9. $(m, r) \leftarrow least_squares(x, t)$
10. $p \leftarrow nLastFrame/N$
11. **Si** $\|m\| > m_{umbral}$ **y** $\|r\| > r_{umbral}$ **y** $p < p_{umbral}$ **entonces**
12. $short_{occlflag} \leftarrow true$
13. **Si no**
14. $short_{occlflag} \leftarrow false$
15. **fin**
16. **retornar** $short_{occlflag}$

Gestión de oclusiones largas

Se denomina oclusión larga cuando un objeto diferente al que se sigue es detectado en la imagen I_1^k , esto puede provocar que el sistema siga al ocluidor en lugar del objeto original. Este problema se puede modelar como un problema de dos estados: ocluido y no ocluido; para determinar este estado se usa un modelo de Markov oculto (HMM) con probabilidad de transición $p(x_t|x_{t-1})$ y probabilidad de observación $p(z|x_t)$, donde la probabilidad de transición se asume constante (0.5) y la probabilidad de observación depende del estado actual y el vector de observación, la máquina de estados usada para este trabajo se muestra en la Figura 16.

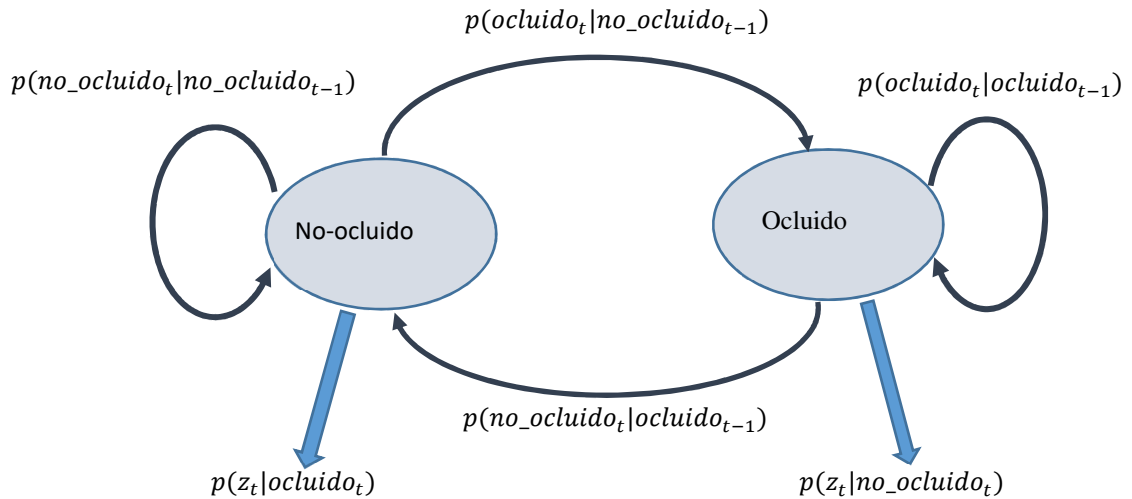


Figura 16: HMM usado para modelar las oclusiones. El modelo considera dos estados: ocluido y no-ocluído.

La observación del vector z está compuesta por tres variables binarias: *patch_change*, *features_entering* y *features_leaving*:

$$z = (\textit{patch_change}, \textit{features_entering}, \textit{features_leaving}) \quad (29)$$

La variable *patch_change* es una bandera que indica si la región seleccionada de la imagen está relacionada con la detección, es decir si es similar o diferente a la región de la detección anterior. En base a esto se pueden detectar cambios abruptos que pueden ser utilizados para detectar las oclusiones. Se entrena un clasificador estadístico usando ejemplos reales de detecciones consecutivas correspondientes al mismo objeto (*patch_change* = 0) y detecciones consecutivas correspondientes a diferentes objetos (*patch_change* = 1). Se calculan los siguientes componentes para caracterizar la similitud entre las regiones seleccionadas de cada detección:

- Distancia Euclideana entre el histograma LBP de ambos recuadros (BB), y
- Distancia Euclideana entre el histograma HSV de ambos recuadros (BB) (solo se consideran los canales H y S).

Las personas con ropa similar pueden provocar fallas y hacer que el sistema se confunda de persona, sin embargo, las funciones relacionadas con el movimiento (basados en tracks de características) ayudan a definir la persona a ser seguida. Para lograr esto, se definen variables como *features_entering* y *features_leaving* que indican si existe un número significativo de tracks que ingresan (*nin*) o salen (*nout*) del cuadro de detección. Estas medidas son usadas en caso de que existiera oclusión al objeto seguido, ya que cuando se produce la oclusión, la ruta de los tracks van hacia el interior del recuadro de detección y cuando la oclusión desaparece los tracks van hacia fuera del recuadro de detección. Por lo tanto, la cantidad de tracks que van fuera o dentro del recuadro se pueden utilizar para discriminar el inicio o fin de la oclusión. Esta medida es

robusta frente a traslaciones porque, la cantidad de tracks que van dentro del recuadro son casi igual al número que abandonan el recuadro. Las variables se definen así:

$$features_entering = \begin{cases} 1 & \text{Si } nin > 15 \text{ y } nin > 2 * nout \\ 0 & \text{Si no} \end{cases} \quad (30)$$

$$features_leaving = \begin{cases} 1 & \text{Si } nout > 15 \text{ y } nout > 2 * nin \\ 0 & \text{Si no} \end{cases} \quad (31)$$

El algoritmo de Viterbi [47] es un óptimo estimador de estados para HMMs, por lo que se utiliza para detectar oclusiones en este trabajo. La estimación de estado de una oclusión larga se implementa en una función de *viterbi()* que realiza el cálculo de las características z y la estimación del nuevo estado mediante el uso de $p(x_t|x_{t-1})$ y $p(z|x_t)$.

El método utilizado para resolver el problema de asociación de datos compara características obtenidas de dos detecciones diferentes, y decide si ambas detecciones corresponden a la misma persona o no. El procedimiento de entrenamiento considera pares de detecciones de coincidencia y no coincidencia de personas en la base de datos de entrenamiento. Por lo tanto, si bien el método requiere que la persona vista la misma ropa durante cada secuencia de video, no es necesario tener un modelo a priori de la ropa. En consecuencia, el método no depende de la ropa específica de la persona en cada video.

Debe tenerse en cuenta que los errores en la estimación final de la oclusión pueden provocar que el sistema se confunda y provoque el seguimiento del objeto que está realizando la oclusión. Sin embargo, en los experimentos realizados, no se produjo confusión de seguimiento.

4.4. Seguimiento de Objetos Libres de Retardo

La transformación libre de retardo T se aplica a una detección del objeto $DetBB_1^k$, donde se solicita estimar un conjunto de tracks τ_u^k para obtener la transformación y junto con la información sobre detecciones falsas y oclusiones (cortas o largas) se determina si la detección es válida y si será aplicada para proyectar la nueva detección. El funcionamiento se detalla en el Algoritmo 4.

El sistema completo funciona procesando una secuencia de imágenes, donde se ejecutan dos procesos a la vez: un proceso de seguimiento y el otro proceso se encarga de realizar la detección de objetos. El proceso de seguimiento recibe como entrada una secuencia de imágenes: en el caso del detector la entrada siempre es una nueva imagen mientras el detector de objetos está funcionando, en cambio, el proceso de seguimiento genera los tracks de puntos característicos y a estos puntos se realiza el seguimiento. Cuando el detector de objetos genera una respuesta de detección, el cuadro delimitador de detección se actualiza utilizando los tracks de características, aplicando una transformación libre de retardo. Se detectan oclusiones cortas cuando un objeto ocluidor se mueve dentro del objeto en seguimiento y genera la pérdida de sus puntos seguidos, lo que imposibilita la aplicación de una transformación libre de retardo. Además, las oclusiones largas se detectan cuando las características relacionadas con la respuesta de detección de objetos son diferentes de las esperadas del objeto seguido. Un HMM se usa para estimar el inicio y el final de las oclusiones, considerando tanto los tracks de características como las características de la detección actual de objetos (cuadros de delimitación e histogramas de color de los trozos de imagen correspondientes).

Algoritmo 4: Procedimiento de tracking y detección libre de retardo.

1. Dado $k \leftarrow 1$
2. **Mientras** (existan imágenes)
3. Captura I_1^k
4. Iniciar función *object_detect*(I_1^k) en un segundo proceso
5. Inicializar el track de características τ_1^k para detector esquinas en I_1^k
6. Definir $u \leftarrow 1$
7. **Mientras** (*object_detect*) no ha terminado
8. $u \leftarrow u + 1$
9. Captura I_u^k
10. $\tau_u^k \leftarrow \text{update_tracks_using_KLT}(\tau_{u-1}^k, I_u^k)$
11. **fin**
12. $n_k \leftarrow u$
13. $\text{DetBB}_1^k \leftarrow \text{object_detection_result}()$
14. $\text{long_occl_flag} \leftarrow$
 $\text{viterbi}(\tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N], \text{DetBB}_1^k, I_1^k, \text{DetBB}_1^{k-1}, I_1^{k-1})$
15. **Si** (long_occl_flag es falso) **entonces**
16. $\text{feat}_{\text{falseDet}}^k \leftarrow \text{comp_feat}_{\text{falseDet}}(\text{DetBB}_1^k, I_1^k, \text{DetBB}_1^{k-1}, I_1^{k-1})$
17. $\text{false_det_flag} \leftarrow \text{classifier}_{\text{falseDet}}(\text{feat}_{\text{falseDet}}^k)$
18. **Si** (false_det_flag es falso) **entonces**
19. $\text{short_occl_flag} \leftarrow \text{threshold_classifier}(\tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N])$
20. **Si** (short_occl_flag es falso) **entonces**

```

21.            $T^k \leftarrow \text{ransac\_estimator}(\text{DetBB}_1^k, \tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N])$ 
22.            $\text{BestBB}^k \leftarrow T^k(\text{DetBB}_1^k)$ 
23.            $\text{occlusion\_flag} \leftarrow 0$ 
24.       Si no
25.            $\text{BestBB}^k \leftarrow \text{DetBB}_1^k$ 
26.            $\text{occlusion\_flag} \leftarrow 1$ 
27.       fin
28.   fin
29.        $\text{BestBB}^k \leftarrow \text{BestBB}^{k-1}$ 
30.        $\text{occlusion\_flag} \leftarrow 0$ 
31.   fin
32. Si no
33.      $\text{BestBB}^k \leftarrow \text{BestBB}^{k-1}$ 
34.      $\text{occlusion\_flag} \leftarrow 1$ 
35.   fin
36.    $k \leftarrow k + 1$ 
37. Fin

```

Capítulo 5

Experimentos

El método propuesto e implementado se valida en la aplicación “seguimiento de personas” en la cual un robot de servicio sigue a una persona bajo un entorno completamente dinámico donde existe cambios de intensidad variable, fondos desordenados, ambientes concurridos (varias personas alrededor) y oclusiones por parte de otras personas. Por otro lado, el seguimiento de personas por parte de robots de servicio se considera una habilidad relevante que interactúan con seres humanos diariamente.

5.1. Descripción de los experimentos a realizar

Para realizar las pruebas de entrenamiento y validación del sistema, se construyó una base de datos que contiene secuencias de vídeo capturadas bajo diferentes situaciones; en los videos se sigue a una persona con una cámara que está apuntando siempre hacia adelante aproximadamente a una distancia de 1.5 [m] del sujeto que sigue. La cámara se mueve libremente, por lo que se supone que sus parámetros intrínsecos y extrínsecos no son conocidos, es decir, el sistema no depende de ningún tipo de calibración de la cámara. Las imágenes capturadas tienen una resolución de 640x480; sin embargo, se reducen a 320x240 para ser procesados.

En la Tabla 2, se detallan los diferentes videos que conforman la base de datos usadas, tanto para entrenamiento como para prueba. Las secuencias de vídeo incluyen diferentes condiciones en el ambiente (véase la Figura 17). Un total de 44.069 imágenes se utilizaron para el entrenamiento del sistema de detección y 41.289 imágenes se utilizaron como la base de datos de prueba. Las bases de datos de entrenamiento y prueba se componen de las siguientes secuencias de vídeo:

Conjunto de videos usados para entrenamiento:

Set 1: 20 videos de entrenamiento sin oclusión (v1, v3, v5, v7, v9, v11, v13, v15, v17, v19, v21, v23, v25, v27, v29, v31, v33, v35, v37, v39).

Set 2: 11 videos de entrenamiento con oclusiones (vo_4, vo_5, vo_12, vo_13, vo_14, vo_15, vo_16, vo_17, vo_18, vo_19, vo_20).

Conjunto de video usados para pruebas:

Set 3: 17 videos de pruebas sin oclusiones (v2, v4, v6, v8, v10, v12, v14, v16, v18, v20, v22, v24, v26, v28, v30, v32, v34).

Set 4: 9 videos de pruebas con oclusiones (vo_1, vo_2, vo_3, vo_6, vo_7, vo_8, vo_9, vo_10, vo_11).

Set 5: 7 videos de pruebas del trabajo anterior [9].

Set 6: 4 videos de prueba con tiempos de duración variable con respecto a las oclusiones.

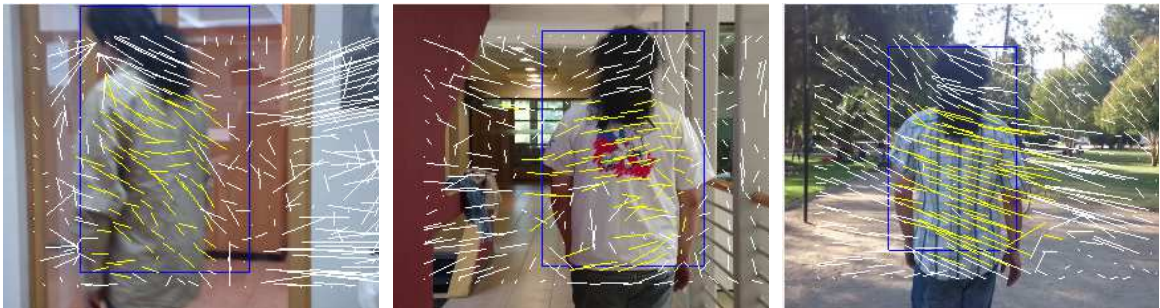
Tabla 2: Videos usados para entrenamiento y pruebas de los diferentes métodos de seguimiento de personas.

Videos	Set	Descripción	Oclusión
v1, v2,v3, v4, v5, v6, v7	1, 3	Buena iluminación, una persona caminando en frente de la cámara a 1.5[m] de distancia.	No
v8, v9, v10, v11 v12	1, 3	Escenario complejo, se debe a que la iluminación cambia abruptamente además de la existencia de reflejos de sol alrededor de la persona.	No
v13, v14, v15, v16, v17	1, 3	Escenario complejo, se inicia con buena iluminación y posteriormente existen variaciones en el cambio de iluminación ya que existe reflejos de sol alrededor de la persona.	No
v18, v19, v20, v21, v22	1, 3	Escenario complejo, esto debido a la existencia de mala iluminación.	No
v23, v24, v25, v26, v27, v28, v29, v30, v31, v32, v33, v34, v35, v37, v39	1, 3	Escenario complejo, se debe a que la persona camina en un escenario público (un parque) en el que la iluminación es totalmente variable y además existe gente en el fondo.	No
vo_1, vo_2, vo_3, vo_4	2, 4	Escenario complejo, se debe a que la persona camina en un parque público donde la iluminación es totalmente variable existen oclusiones porque la persona camina alrededor de los árboles y de diferentes objetos.	Yes
vo_5, vo_6, vo_7, vo_8, vo_9, vo_10, vo_11, vo_12, vo_13, vo_14, vo_15, vo16, vo_17, vo_18, vo_19, vo_20	2, 4	Escenario complejo, se debe a que la persona camina en una calle donde la iluminación es totalmente variable y además existen personas alrededor y se producen oclusiones por parte de otras personas hacia la persona seguida.	Yes
V4, V5, V6, V7, V8	5	Buena iluminación y el escenario es complejo ya que existe interacción de la persona seguida con otras personas, además de que existen muchas personas en el fondo y se dan oclusiones. En el video V8 la persona ingresa a un ascensor.	Yes

VD2, VD3	5	Escenario complejo, existen personas en el fondo y hay interacción con otras personas además de la existencia de oclusiones.	Yes
vo1, vo2, vo3, vo4	6	La persona se encuentra a una distancia fija y existen oclusiones con distintos tiempos de duración de la oclusión.	Yes

Todas las secuencias de video se encuentran etiquetadas. Una de cada 10 imágenes del video fue etiquetado a mano y las imágenes intermedias fueron etiquetados usando interpolación lineal, las imágenes con oclusiones también fueron etiquetadas con un segundo conjunto de etiquetas. La persona a ser seguida debe encontrarse en frente de la cámara cuando se empieza a grabar el video, esto, para fines de inicialización. De esta manera no se genera una ambigüedad en cuál es la persona a seguir. Los tracks de características se muestran en la Figura 15 y Figura 17, donde se observa que siempre se sigue a la persona correcta independiente de si hay personas alrededor. Por otro lado, se debe considerar que las oclusiones se pueden detectar a medida que desaparecen los tracks de características relacionadas con la persona ocluida.

Existen dos indicadores utilizados para comparar el rendimiento de los diferentes métodos: la tasa de detección correcta (DR), definido como el número de detecciones positivas verdaderas dividido por la cantidad de detecciones reales del (GT) y el número de falsos positivos (FP). Se considera que una detección es correcta si el recuadro que el sistema genera es similar al recuadro etiquetado ($IoU > 0.5$) y el objeto no se encuentra ocluido.



Interior.

Interior con cambios de iluminación.

Exterior.



Exterior con gente alrededor.

Exterior con oclusiones.

Exterior con oclusiones.

Figura 17: Secuencias de imágenes en el mundo real usadas para las evaluaciones del método. El recuadro azul representa la salida del método libre de retardo (Delay-free). Las líneas representan los tracks de características. Los tracks aceptados por RANSAC y usados para calcular la transformación libre de retardo se muestran en Amarillo.

5.2. Métodos evaluados

En este trabajo, se evalúan los métodos propuestos (DF-Simple, DF-Full), con 1 método de tracking basado en detecciones (HoG1/2) y 4 métodos de tracking directo (CT, TDK, Meanshift y KCF). Los siguientes métodos de detección y seguimiento serán evaluados y comparados en el método propuesto en esta tesis:

- *HOG 1/2*: Este término se usa en [9] para describir un DPM-HOG detector de torsos basado en [16]. Este método es el mismo que el HOG estándar, pero para cumplir con el objetivo se realizó una modificación al método ya que usa solo la parte superior del torso para entrenar al detector. De esta forma se pueden detectar personas de pie cerca de la cámara del robot donde normalmente solo sería visible el torso superior. Los ejemplos positivos y negativos para el clasificador SVM se extraen del conjunto de datos de entrenamiento para el seguimiento de personas. Este método de detección genera una detección con retardo $DetBB_1^k$ referida a la imagen I_1^k que es anterior a la imagen actual $I_{n_k}^k$.

- *HOG 1/2-I*: Este es un detector ideal (I) y está basado en el método anterior HoG 1/2, calculado off-line, frame a frame. Es decir que para cada imagen de la secuencia de imágenes se va calculando su respectiva detección.

- *DF-Simple (Seguimiento libre de retardo)*: Este método está basado en HoG 1/2. Sin embargo, este método proyecta $DetBB_1^k$ a una detección libre de retardo $DetBB_{n_k}^k$ usando los tracks de características disponibles.

- *DF-Full*: Este método es similar al anterior, pero usa un clasificador SVM para descartar las falsas detecciones y las posibles oclusiones, además de esto usa un modelo HMM para estimar la presencia de oclusiones.

- *CT (Compressive Tracking)* [4]: En este método, las características Haar-like son usadas para seguir los trozos de imagen dentro de la secuencia de imágenes. Este método está inspirado en Compressive Sensing.

-TDK (Tracking by Detection with kernels) [5]. Este método entrena un clasificador en cada imagen dentro de la transformada de Fourier. Este método trabaja a una alta tasa de imágenes por segundo.

- Meanshift [6]: Este método calcula el histograma de color de un trozo de una imagen a ser seguido, posteriormente busca la zona con el histograma más parecido alrededor de la posición anterior.

- KCF Kernel Correlation Filter [7]: Similar al TDK, pero usa canales de características similares al HoG.

Para el detector de personas HoG 1/2, la implementación original de los autores descrita en [16] fue usado para la detección de torso. CT [4] y TDK [5] también tiene pública la implementación que fue usada. Además se usó la implementación libre de Meanshift de [6]. El algoritmo KLT fue integrado en el sistema para calcular el track de características, OpenCV y SVM son las herramientas que se usan para realizar el procesamiento de imágenes y el entrenamiento respectivamente, los parámetros RBF óptimos de SVM fueron seleccionados por OpenCV usando la validación cruzada de la data de entrenamiento.

Para HoG 1/2, HoG 1/2-I, CT se ejecutó la implementación estándar de MATLAB, en cambio Meanshift, KLT y OpenCV se implementaron en C++.

5.3. Resultados

La Tabla 3 muestra el desempeño de los diferentes métodos correspondientes al Subconjunto 3 de videos sin oclusiones, como se puede observar el uso del método de transformación libre de retardo (DF-Simple) supera al procedimiento original del HoG 1/2 incrementando la tasa de detecciones y bajando la tasa de falsas detecciones. El uso de la transformación libre de retardo incrementa la tasa de detección de un 0.88 a un 0.90 que es muy cercano al HoG 1/2 -I de 0.93. La transformación DF-Full puede realizar la gestión de falsas detecciones y detectar oclusiones, por lo que puede mejorar el sistema al disminuir el número de falsas detecciones, sin embargo, el costo de disminuir las falsas detecciones implica descartar algunas detecciones buenas. En los videos que se muestran en la Tabla 3 el uso de la transformación DF-full genera una mejora en la reducción de la tasa de falsos positivos de 8,18 a 2.24, es decir disminuye en un 73% los falsos positivos, sin embargo, la tasa de detecciones correctas disminuye de 0.88 a 0.83. Por otro lado, los sistemas CT y TDK tienen una tasa de detección de 0.7 y una tasa de falsos positivos mucho más alta que de las metodologías basada en HOG. Meanshift tiene una detección de 0.15 y 58 de falsos positivos por lo cual se determina que no es confiable para el seguimiento de personas. KFC tiene una tasa de detecciones de 0.77 que es más alta que los métodos de seguimiento mencionados anteriormente, pero también es superada por el método propuesto es esta tesis.

Tabla 3: Tasa de detección para los videos del subconjunto (Set 3) (sin oclusiones) para DF, CT, TDK, KCF and Meanshift. Nframes: Número de imágenes. NGOFS: Número de grupos de imágenes. DR: Tasa de detecciones correctas. FP: Número de falsos positivos.

video	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
v2	1951	64	0.90	6	0.10	58	0.16	54	0.54	29	0.15	55	0.92	5	0.74	3	1.00	0
v4	1901	63	1.00	0	0.85	9	0.29	45	0.81	12	0.11	56	0.98	1	0.97	1	1.00	0
v6	1931	61	0.90	6	0.39	37	0.29	43	0.99	1	0.16	51	0.90	6	0.92	1	0.93	4
v8	2281	73	0.90	7	0.17	60	0.12	64	0.60	29	0.25	55	0.90	7	0.86	2	0.92	6
v10	1901	63	0.85	9	0.47	33	0.04	61	0.99	1	0.17	52	0.78	13	0.65	5	0.85	9
v12	1931	64	0.89	7	0.02	63	0.02	63	0.61	25	0.15	54	0.92	5	0.59	5	1.00	0
v14	3621	118	0.88	14	0.11	105	0.02	116	0.97	4	0.23	90	0.97	4	0.97	3	1.00	0
v16	3621	114	0.83	19	0.77	26	0.29	82	0.98	2	0.18	94	0.85	16	0.82	2	0.86	15
v18	1861	57	0.89	6	0.85	9	0.52	27	0.37	36	0.10	51	0.91	5	0.89	3	0.93	4
v20	1881	61	0.86	8	0.45	34	0.31	42	0.19	49	0.15	52	0.90	6	0.81	3	0.92	5
v22	1931	64	0.89	7	0.58	27	0.16	54	0.42	37	0.12	57	0.92	5	0.57	3	0.95	3
v24	1941	64	0.90	6	0.70	19	0.96	3	0.72	18	0.09	58	0.98	1	0.95	0	1.00	0
v26	1861	62	0.98	1	0.11	55	0.15	53	0.99	1	0.17	52	0.98	1	0.98	1	1.00	0
v28	1861	62	0.94	4	0.10	56	0.03	60	0.99	1	0.11	55	0.98	1	0.97	2	1.00	0
v30	1911	61	0.65	19	0.21	48	0.75	15	0.99	1	0.10	55	0.70	16	0.69	2	0.74	14
v32	1991	66	0.89	7	0.79	14	0.16	55	0.99	1	0.13	57	0.97	2	0.97	2	0.98	1
v34	1841	49	0.72	13	0.08	45	0.89	5	0.99	0	0.16	41	0.77	11	0.74	0	0.79	10
Average	2130.41	68.59	0.88	8.18	0.40	41.11	0.30	49.54	0.77	14.47	0.15	58.00	0.90	6.18	0.83	2.24	0.93	4.18

La Tabla 4, muestra los resultados correspondientes de los diferentes métodos empleados en el subconjunto de videos denominado Set 4 con oclusiones. Como se puede observar en dicha tabla, el procedimiento libre de retardo DF-Simple supera al registrado por el método original HoG ½, ya que tiene un rendimiento mejor tanto en tasa de detecciones correctas como en la disminución de falsos positivos. El uso del método de transformación libre de retardo permite aumentar las tasas de detecciones correctas de 0.84 a 0.89 que es más cercana a la del HoG ½ Ideal de 0.96. El uso de los clasificadores SVM y HMM para oclusiones del DF-full disminuyó la tasa de falsos positivos de 10.44 a 5.33 pero esto también provoco la disminución de la tasa de detecciones correctas de 0.84 a 0.79 porque, como se explicó en párrafos anteriores, existe una compensación entre la tasa de detecciones correctas y la disminución de falsos positivos. Los sistemas de seguimiento clásicos (CT, TDK, Meanshift, KCF) funcionan mal en videos con oclusiones en comparación al método propuesto. En esta tabla también se puede observar que el video vo_1 con respecto al método DF-FULL tiene una tasa de detección de 0.33 siendo más baja que el método DF-Simple y esto se debe a la complejidad de las oclusiones ya que en este video el sujeto siempre se está escondiendo detrás de los árboles. Sin embargo, los falsos positivos se mantienen en ambos métodos.

Tabla 4: Tasa de detecciones correctas para los videos del Set 4 (con oclusiones) para DF, CT, TDK, KCF y Meanshift. Nframes: Número de imágenes. NGOFS: Número de grupos de imágenes. DR: Tasa de detecciones correctas. FP: Numero de falsos positivos.

video	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
vo_1	1281	28	0.63	10	0.04	27	0.06	27	0.13	24	0.10	26	0.96	1	0.33	1	0.89	3
vo_2	1151	36	0.59	14	0.16	31	0.02	35	0.24	27	0.10	33	0.78	9	0.67	1	0.96	4
vo_3	1441	46	0.74	12	0.04	44	0.03	45	0.96	2	0.14	40	0.77	11	0.70	4	0.91	5
vo_6	1871	62	0.93	8	0.00	62	0.24	48	0.39	38	0.11	56	0.93	8	0.93	5	0.98	5
vo_7	1981	66	0.90	11	0.00	66	0.00	66	0.43	38	0.07	62	0.93	9	0.93	7	0.95	8
vo_8	1961	65	0.93	8	0.00	65	0.00	65	0.30	46	0.09	60	0.89	10	0.91	7	0.98	5
vo_9	1911	64	0.97	5	0.28	48	0.36	43	0.36	41	0.12	57	0.95	6	0.95	4	1.00	3
vo_10	1921	63	0.96	11	0.13	56	0.02	62	0.12	55	0.18	53	0.90	14	0.90	14	0.98	10
vo_11	1881	62	0.89	15	0.53	37	0.08	58	0.55	28	0.24	50	0.93	13	0.82	5	1.00	10
Average	1711.00	54.67	0.84	10.44	0.13	48.43	0.09	49.96	0.39	33.20	0.13	48.48	0.89	9.00	0.79	5.33	0.96	5.89

La Tabla 5 muestra el rendimiento de los diferentes métodos en los videos correspondientes al Set 5 videos con oclusiones. Los videos se capturaron con una cámara kinect, por lo que la calidad de las imágenes es menor que las imágenes de las otras bases de datos y los tracks de características son menos confiables que las otras bases de datos. El uso de la transformación libre de retardo DF-Simple mejora la tasa de detección a 0.87 y disminuye la tasa de falsos positivos a 7.71. Sin embargo, la tasa de detecciones correctas no es cercana a las detecciones de HOG 1/2-I de 0.96 esto es debido a la falta de confiabilidad en los tracks de características. El uso de la transformación libre de retardo DF-full disminuye la tasa de detección de falsos positivos de 9.57 a 4.43 pero la tasa de detecciones correctas disminuye de 0.84 a 0.64, lo que se debe a que los clasificadores etiquetan erróneamente algunas detecciones correctas. Nuevamente en el conjunto de datos los clasificadores CT, TDK, Meanshift y KCF tienen un rendimiento deficiente.

Tabla 5: Tasa de detecciones correctas para los videos del Set 5 (con oclusiones) para DF, CT, TDK, KCF y Meanshift. Nframes: Número de imágenes. NGOFS: Número de grupos de imágenes. DR: Tasa de detecciones correctas. FP: Numero de falsos positivos.

video	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
V4	3344.00	92	0.96	5	0.35	80	0.35	80	0.49	47	0.09	112	0.98	3	0.75	2	0.99	2
V5	1737.00	38	0.87	5	0.36	41	0.20	51	0.86	5	0.07	59	0.89	4	0.92	2	0.97	1
V6	5004.00	64	0.66	23	0.04	177	0.04	177	0.06	60	0.07	172	0.77	16	0.63	11	0.92	6
V7	3011.00	35	0.72	12	0.01	110	0.00	111	0.40	21	0.08	102	0.81	9	0.56	2	0.91	6
V8	4436.00	118	0.91	16	0.20	115	0.04	132	0.05	118	0.06	130	0.92	15	0.63	12	0.96	10
VD2	1140.00	24	0.86	5	0.36	27	0.20	33	0.20	24	0.06	39	0.86	5	0.14	1	0.95	3
VD3	1900.00	14	0.93	1	0.09	64	0.05	67	0.06	14	0.05	66	0.86	2	0.86	1	1.00	0
Average	2938.86	55.00	0.84	9.57	0.20	87.86	0.13	93.17	0.30	39.69	0.07	97.27	0.87	7.71	0.64	4.43	0.96	4.00

Se midió el rendimiento de los diferentes métodos en los videos correspondientes al conjunto de datos Set 6, está compuesto de cuatro videos en los que la persona seguida permanece estática mientras una segunda persona camina y se para delante de la persona seguida. Entre la persona seguida y la cámara, se generaron varias oclusiones en cada uno de los videos con duraciones crecientes de 2, 4, 8, 16 y 32 segundos. Las oclusiones se agrupan en cinco subconjuntos, cada uno con varias oclusiones y con una duración similar. Por lo tanto, la Tabla 6 muestra los promedios de todos los videos evaluados según la duración de la oclusión. Se puede observar que los tiempos de oclusión elevados incrementan el número de falsos positivos tanto para HoG $\frac{1}{2}$ como para HoG $\frac{1}{2}$ -I. Los falsos positivos se reducen en un 65%, además se debe tener en cuenta que el sistema propuesto supera a los sistemas de seguimiento anteriores por un amplio margen, tanto en la tasa de detecciones correctas como en la reducción de la cantidad de falsos positivos.

Tabla 6: Tasa de detecciones correctas para los videos del Set 6 (con oclusiones) para DF, CT, TDK, KCF y Meanshift. Nframes: Número de imágenes. NGOFS: Número de grupos de imágenes. DR: Tasa de detecciones correctas. FP: Número de falsos positivos.

Oocl Time	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
2[s]	190.50	8.25	1.00	3.25	0.44	6.41	0.44	6.41	0.58	3.44	0.76	5.07	0.96	3.50	0.71	1.50	1.00	3.25
4[s]	303.25	9.75	0.94	5.25	0.64	6.72	0.64	6.72	0.72	2.71	0.78	6.37	0.88	5.50	0.88	1.25	0.94	5.25
8[s]	398.25	11.75	0.75	7.00	0.59	8.55	0.59	8.55	0.75	2.97	0.80	8.12	0.71	7.25	0.71	4.75	0.75	7.00
16[s]	677.25	20.75	1.00	14.25	0.74	16.47	0.82	16.21	0.86	3.01	0.75	16.69	0.97	14.50	0.94	1.00	1.00	14.25
32[s]	1184.00	32.50	1.00	26.75	0.60	28.62	0.72	28.39	0.69	10.08	0.72	28.32	0.96	27.00	0.96	8.75	1.00	26.75

Con el fin de caracterizar mejor las capacidades del detector propuesto para oclusiones de larga duración, se usa una segunda medida para medir la calidad de las detecciones: el número de los verdaderos positivos y la cantidad de falsos negativos en todos los videos. Del número total de oclusiones largas en el conjunto 4 y 6 el 79% se detectaron correctamente y solo se obtuvo una tasa de detección de falsos positivos del 3%. Las fallas de detección de oclusión son causadas por tres situaciones posibles: la persona ocluida no se detecta antes de que ocurra la oclusión, se produce una detección incorrecta de la persona o el color y los histogramas LBP del ocluidor son muy similares a los de la persona ocluida. Además, las personas demasiado cerca de la cámara pueden hacer que el detector de personas falle antes de que ocurra la oclusión, estos resultados indican que el detector de oclusiones largas es seguro de usar y que es capaz de rechazar detecciones incorrectas que podrían provocar un error en el proceso del seguimiento de personas.

La presencia de múltiples personas, oclusiones y la falta de textura en la ropa y la falta o presencia de objetos oclusores en las secuencias de video son factores que afectan negativamente el rendimiento del sistema. El impacto generado por estos tres factores se analiza usando los datos de los Set 3 al Set 6. Se analizan tres casos: i) cuando los videos incluyen a varias personas, la tasa de detección no se ve afectada, pero los falsos positivos disminuyen en un 50% con respecto a los videos sin varias personas. ii) cuando los videos incluyen oclusiones, la tasa de detección no se ve afectada pero los falsos positivos se incrementan en un 50%. iii) cuando en los videos la ropa de la persona seguida no tiene textura la tasa de detección disminuye en un 17% y los falsos positivos aumentan en un 40%. A partir de estos resultados, se concluye que, si bien la presencia de múltiples objetos no disminuye el rendimiento del sistema, las oclusiones y la falta de textura en el objeto hacen que el sistema disminuya su rendimiento. Se debe tomar en cuenta que, a pesar de estas difíciles condiciones, el incremento de falsos positivos no supera el 50%, por lo tanto, el sistema puede funcionar con éxito en estos casos. El procedimiento de asociación de datos está entrenado para discriminar entre dos personas diferentes.

En las tablas anteriores los resultados obtenidos son un poco difíciles de entender, es por ello que se genera una tabla resumen (ver Tabla 7). Se puede concluir que el método propuesto DF-Full es el mejor método de seguimiento al ser comparado con los otros métodos como son CT, TDK, KCF y MeanShift. Se puede observar que la tasa de detecciones aumenta y las falsas detecciones disminuyen considerablemente.

Tabla 7: Promedio de resultados de los sistemas de seguimiento propuesto DF-Full, CT, TDK, KCF y Meanshift sobre los Sets de videos 3, 4, 5 y 6. DR: Tasa de detecciones correctas. FP: Número de falsos positivos. En total 37 videos fueron considerados. El método propuesto DF-Full es el mejor tanto en tasa de detecciones correctas (DR) y falsos positivos (FP) en todo el conjunto de datos probados.

Video Set	DF-Full		CT		TDK		KCF		Meanshift	
	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
Set 3 (sin oclusiones)	0.83	2.24	0.40	41.11	0.30	49.54	0.77	14.47	0.15	58
Set 4 (con oclusiones)	0.79	5.33	0.13	48.43	0.09	49.96	0.39	33.20	0.13	48.48
Set 5 (con oclusiones)	0.64	4.43	0.20	87.86	0.13	93.17	0.30	39.69	0.07	97.27
Set 6 (con tiempo incremental)	0.84	3.45	0.60	13.35	0.64	13.25	0.72	4.44	0.76	12.91

Finalmente se analizan dos videos relevantes que muestran fallas (secuencias de videos v10 y v22 del Set 1). Aunque se han descrito las ventajas comparativas del método propuesto en esta tesis (capacidad de seguimiento a largo plazo y fuerte asociación de datos), el análisis de fallas puede ser útil para comprender las debilidades del método, pero también para mostrar sus capacidades de recuperación ante fallas. En los dos videos analizados del conjunto de datos etiquetados (GT) que se muestran en verde, la detección del objeto DPM-HOG se muestra en rojo y la estimación propuesta se seguimiento libre de retardo se muestra en azul, esto se puede observar en la Figura 18. La tasa de detección en el video v10 es baja (DF-full 78% y DF-Simple 85%) incluso cuando no contiene ningún tipo de oclusión. Esto se debe a que la persona se acerca demasiado a la cámara, por lo tanto, el detector de personas falla y genera varios falsos positivos consecutivos, lo que hace que el sistema de seguimiento falle. Sin embargo, el sistema se recupera cuando la persona se encuentra a la distancia de 1.5 m de la cámara. El video v22 contiene cambios abruptos de iluminación, que hace que el detector de oclusiones falle y pierda el objetivo en aproximadamente 18 segundos. Sin embargo, el sistema se recupera al final del video. De acuerdo con el análisis realizado, se puede decir que el sistema falla cuando el objeto se encuentra muy cerca de la cámara o cuando existen fuertes cambios de iluminación notar que el sistema es capaz de recuperarse en ambos videos.

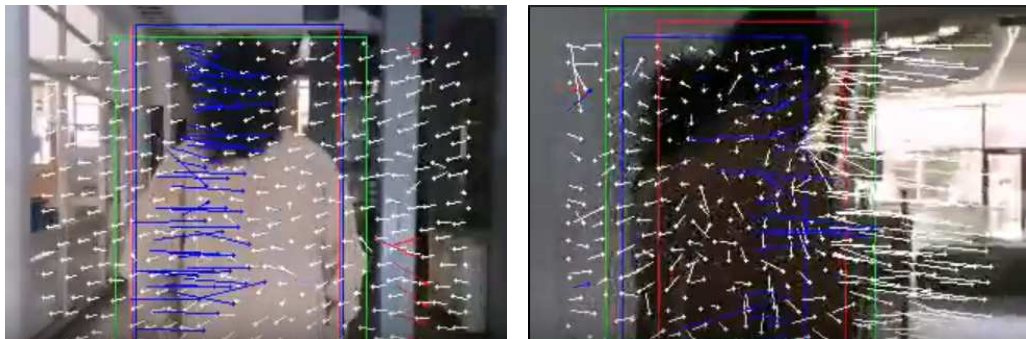


Figura 18: Fallas en el seguimiento causadas por cambios fuertes de iluminación (videos v10 y v22). Etiqueta de la posición real de la persona (GT) en verde, DPM-HOG detección de objetos en rojo y la transformación libre de retardo en azul.

5.4. Posibles aplicaciones del método libre de retardo

5.4.1. Robot de servicio

Para el robot de servicio Bender se puede implementar un caso de uso de seguimiento de personas desarrollado en esta tesis. Bender posee una plataforma de investigación abierta y flexible que proporciona capacidades de comunicación similares a las de los humanos. Bender tiene la parte superior del cuerpo antropomórfica (cabeza, brazos, pecho) y una plataforma de accionamiento diferencial que le proporciona movilidad. El robot usa dos computadoras Alienware, una dedicada a la visión por computador y otra

dedicada a la planificación y navegación. Los componentes de hardware electrónico y mecánico del robot, así como la arquitectura de software se describen en [28].

El comportamiento de “seguimiento de personas” se ha probado en entornos de la vida diaria tanto en interiores como exteriores, como los que se muestran en la Figura 19. En estos entornos, Bender puede seguir a las personas de forma robusta, pudiendo gestionar oclusiones producidas por humanos y cambios en la iluminación. Este comportamiento tiene características que los diferencian del seguimiento de personas típico en el dominio de la imagen: primero, la persona seguida se encuentra cerca del robot y luego cubre un área amplia de la imagen. En segundo lugar, solo se debe seguir a una persona para que el comportamiento funcione. Tercero, la tarea de seguir a una persona es crítica para generar ordenes de control que puedan mantener a la persona seguida dentro del campo de visión de la cámara inclusive cuando la persona gira de repente. La tarea inicia con la persona en frente del robot, esto para no generar ambigüedades al iniciar el sistema. El sistema completo funciona a 23fps usando solo un núcleo en un procesador Intel core i5. Además, el procedimiento libre de retardo basado en RANSAC requiere solo 0.07 milisegundos para procesar una detección. Por lo tanto, el enfoque propuesto basado en la transformación libre de retardo es una herramienta útil cuando se aplica al seguimiento de personas u otras tareas de visión activa.



Figura 19: Imágenes del robot (Bender) siguiendo a una persona. Izquierda/Derecha: Exterior/Ambiente Interior.

Un video que muestra el seguimiento de persona por Bender está disponible en <http://www.amtc.cl/DFTracker/benderfollow.mp4> y de acuerdo a los videos analizados para los resultados de la sección 5.3 los movimientos que realizan las personas a las que se sigue son naturales dados los ambientes en los que se realizaron estas pruebas. Además, el movimiento del robot también se ve natural.

5.4.2. Otras aplicaciones

El método propuesto en esta tesis se probó en dos aplicaciones adicionales: seguimiento facial y seguimiento de automóviles. Estas aplicaciones se seleccionaron porque los detectores básicos requeridos están disponibles como fuente abierta: MTCNN [49] para la detección de rostros y YOLOv3 [50] para la detección de automóviles. Se

seleccionaron dos secuencias de video de rostro y una secuencia de videos de automóvil del conjunto de datos usado en KCF [7] para propósitos de comparación. Debe tenerse en cuenta que los videos tienen una resolución de 320x240, es decir, esa resolución es suficiente para resolver problemas de seguimiento. KCF fue seleccionado para la comparación ya que es el mejor seguidor alternativo en las tablas 3, 4, 5 y 6. Los resultados de estas pruebas se muestran en la Tabla 8.

De la Tabla 8, se puede observar que DF-Full es capaz de seguir con éxito caras y automóviles del conjunto de datos KCF [7]. Además, DF-full tiene un mejor rendimiento en promedio que KCF en los videos probados. La tasa de detecciones correctas de KCF en el conjunto de datos “David” es pequeña debido a que el etiquetado de la posición original del rostro esta errónea y por ende se tuvo que etiquetar de nuevo.

Tabla 8: Tasa de detecciones correctas (DR) y numero de falsos positivos (FP) para seguimiento de rostros y vehículos usando el sistema propuesto (DF-Full) v/s detección usando la tasa de detecciones correctas de KCF. Nframes: Número de imágenes.

Videos	NFrames	DF-Full		KCF	
		DR	FP	DR	FP
FaceOccl	870	0.99	9	1.00	0
Car5	656	0.96	26	0.34	433
David	750	0.54	345	0.22	585
Average					
Average	758.6	0.83	127	0.52	339

Capítulo 6

Conclusiones

En esta tesis, se propuso un sistema de seguimiento basado en una transformación libre de retardo que puede funcionar con un detector de objetos lento. La transformación libre de retardo es estimada en base a la detección de un conjunto de tracks de características contenidas en el recuadro detectado usando RANSAC. La transformación es usada para proyectar el recuadro detectado en una imagen anterior a la imagen actual de acuerdo a la línea de tiempo, lo que permite la generación de detecciones libres de retardo. El sistema también puede detectar oclusiones y falsas detecciones. Las oclusiones se detectan usando un novedoso modelo HMM que considera los cambios en los trozos de la imagen como el número de tracks de características que entran o salen de los recuadros de detección, es así como trabaja sin necesidad de modelar o detectar el objeto ocluidor. Se usa un SVM para rechazar falsas detecciones al extraer las características de dos trozos de imágenes consecutivas. De esta manera el sistema puede generar detecciones actuales y es robusto frente a errores.

El sistema se probó en la tarea de seguimiento de personas, usando un detector de torsos de personas (HoG 1/2) y el algoritmo KLT para generar tracks de características. El detector de torsos HoG 1/2 se usa para permitir la detección de personas que se encuentran cerca de la cámara. El método completo incluye la transformación libre de retardo, la detección de oclusiones y la detección de falsas detecciones, el método se comparó con el método HOG (modelo basado en partes) y con algoritmo de seguimiento actuales descrito en la revisión bibliográfica.

De acuerdo con los resultados obtenidos se concluye que el método DF-Simple que incluye solamente la transformación libre de retardo mejora los resultados con respecto a los otros métodos de seguimiento y en el peor de los casos (una persona estática con oclusiones) genera una pequeña degradación en los resultados, por lo que el uso de este algoritmo se considera seguro. El método DF-Full puede detectar oclusiones sin necesidad de seguir o modelar al objeto ocluidor, también se observa que el método reduce la cantidad de falsos positivos en un 73% en videos sin oclusiones y también es capaz de detectar oclusiones en un 79% de los casos. Sin embargo, los procedimientos de rechazo de falsas detecciones hacen que la tasa de detecciones correctas disminuya. Por lo tanto, existe una compensación entre la tasa de detecciones correctas versus los falsos positivos. Por otro lado, los métodos de seguimiento de objetos clásicos y nuevos como el CT, TDK, Meanshift y KCF fallan cuando el objeto a seguir esta ocluido y el desempeño de estos sistemas en seguimientos de larga duración falla rotundamente. También se ha podido comprobar mediante las pruebas realizadas que la calidad de las imágenes mejora los resultados de los métodos propuestos en esta tesis.

Los cambios abruptos de iluminación (de la sombra a la luz o viceversa) pueden hacer que el sistema de seguimiento de características falle, tal como se observa en los videos v10, v30 y v34, ya que en este método influye el flujo óptico local y la asociación de datos de las detecciones depende de los histogramas de color. También se puede observar que el método propuesto dio resultados exitosos en dos tareas extras de seguimiento: el seguimiento de vehículos y el seguimiento de caras, lo que permite concluir que el método de seguimiento desarrollado en esta tesis es independiente de los objetos, sin embargo, requiere un detector de objetos adecuado como base para funcionar. Por otro lado, el sistema también puede ser usado para inicializar otros algoritmos de seguimiento de objetos con una hipótesis de detección libre de retardo. El sistema completo puede funcionar a 23fps usando solo un núcleo en un procesador Intel Core i5 sin GPU.

Finalmente, el uso de una metodología para estimar en que imágenes los tracks de características son confiables, analizando la nitidez de la imagen y detectando cambios abruptos en la iluminación, podría reducir la pérdida en la tasa de detecciones correctas mientras se conserva la capacidad de detectar oclusiones y detección de falsas detecciones.

Bibliografía

- [1] J. Shi and C. Tomasi, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, 1994, pp. 593-600.
- [2] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381-395, 1981.
- [3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886-893, 2005.
- [4] K. Zhang, L. Zhang and M.-H. Yang, "Real-time compressive tracking," *Computer Vision–ECCV 2012*, pp. 864-877, 2012.
- [5] J. F. Henriques, R. Caseiro, P. Martins and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," *Computer Vision–ECCV 2012*, pp. 702-715, 2012.
- [6] D. Comaniciu, V. Ramesh and P. Meer, "Real-time tracking of non-rigid objects using mean shift," *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, pp. 142-149, 2000.
- [7] J. F. Henriques, R. Caseiro, P. Martins and J. Batista, "High-speed tracking with kernelized correlation filters., 37(3), 583-596.," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 583-596, 2015.
- [8] OpenNi, "OpenNi User Guide".
- [9] W. Pairo, J. Ruiz-del-Solar, R. Verschae, M. Correa and P. Loncomilla, "Person following by mobile robots: analysis of visual and range tracking methods and technologies," *RoboCup 2013: Robot World Cup XVII*, pp. 231-243, 2014.
- [10] N. Bellotto and H. Hu, "Multisensor-based human detection and tracking for mobile service robots," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 39, no. 1, pp. 167-181, 2009.
- [11] C.-L. Chen, C.-C. Chou and F.-L. Lian, "Detecting and tracking of host people on a slave mobile robot for service-related tasks," in *SICE Annual Conference (SICE)*,

2011 Proceedings of, 2011, pp. 1326-1331.

- [12] A. Ansuategui, A. Ibarguren, J. M. Martínez-Otzeta, C. Tubó and E. Lazkano, "Particle filtering for people following behavior using laser scans and stereo vision," *International Journal on Artificial Intelligence Tools*, vol. 20, no. 2, pp. 313-326, 2011.
- [13] W.-J. Wang and J.-W. Chang, "Implementation of a mobile robot for people following," in *System Science and Engineering (ICSSE), 2012 International Conference on*, 2012, pp. 112-116.
- [14] G. Xing, S. Tian, H. Sun, W. Liu and H. Liu, "People-following system design for mobile robots using kinect sensor," in *Control and Decision Conference (CCDC), 2013 25th Chinese*, 2013, pp. 3190-3194.
- [15] F. Hoshino and K. Morioka, "Human following robot based on control of particle distribution with integrated range sensors," in *System Integration (SII), 2011 IEEE/SICE International Symposium on*, 2011, pp. 212-217.
- [16] P. F. Felzenszwalb, R. B. Girshick, D. McAllester and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627-1645, 2010.
- [17] T. Joachims, "Making large scale SVM learning practical," 1999.
- [18] S. a. M. J. a. P. J. Belongie, "Matching shapes," *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1, pp. 454-461, 2001.
- [19] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 2, no. 91-110, p. 60, 2004.
- [20] M. Everingham, L. Van Gool, C. K. Williams, J. Winn and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303-338, 2010.
- [21] S. Andrews, I. Tsochantaridis and T. Hofmann, "Support vector machines for multiple-instance learning," in *Advances in neural information processing systems*, 2003.
- [22] K. Fukushima, "Neocognitron: A hierarchical neural network capable of visual pattern recognition," *Neural networks*, vol. 1, no. 2, pp. 119-130, 1988.

- [23] Y. a. B. L. a. O. G. a. M. K. LeCun, "Efficient backprop in neural networks: Tricks of the trade (orr, g. and m{\u}ller, k., eds.)," *Lecture Notes in Computer Science*, vol. 1524, no. 98, p. 111, 1998.
- [24] Q. a. L. H. a. L. X. a. D. L. a. L. H. a. R. T. M. a. A. W. Fang, "Detecting non-hardhat-use by a deep learning method from far-field surveillance videos," *Automation in Construction*, vol. 85, pp. 1-9, 2018.
- [25] J. a. D. S. a. G. R. a. F. A. Redmon, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.
- [26] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *arXiv preprint*, 2017.
- [27] Z. a. H. G. a. S. T. a. W. S.-E. a. S. Y. Cao, "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields," *arXiv preprint arXiv:1812.08008*, 2018.
- [28] Z. a. S. T. a. W. S.-E. a. S. Y. Cao, "Realtime multi-person 2d pose estimation using part affinity fields," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7291-7299.
- [29] T. a. J. H. a. M. I. a. S. Y. Simon, "Hand keypoint detection in single images using multiview bootstrapping," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 1145-1153.
- [30] G. Bradski, "The opencv library," *Dr Dobb's J. Software Tools*, vol. 25, pp. 120-125, 2000.
- [31] S. a. B. L. a. A. A. Kreiss, "Pifpaf: Composite fields for human pose estimation}," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11977-11986.
- [32] K. a. Z. X. a. R. S. a. S. J. He, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.
- [33] D. L. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289-1306, 2006.
- [34] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *Information Theory, IEEE Transactions on*, vol. 52, no. 12, pp. 5406-5425, 2006.

- [35] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry and Y. Ma, "Robust face recognition via sparse representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210-227, 2009.
- [36] E. J. Candes and T. Tao, "Decoding by linear programming," *Information Theory, IEEE Transactions on*, vol. 51, no. 12, pp. 4203-4215, 2005.
- [37] J. F. Henriques, R. Caseiro and J. Batista, "Globally optimal solution to multi-object tracking with merged measurements," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011.
- [38] A. R. Zamir, A. Dehghan and M. Shah, "Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs," in *Computer Vision--ECCV 2012*, Springer, 2012, pp. 343-356.
- [39] B. Babenko, M.-H. Yang and S. Belongie, "Robust object tracking with online multiple instance learning," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 8, pp. 1619-1632, 2012.
- [40] A. a. L. C. a. S. J. a. G. M. a. B. H. Saffari, "On-line random forests," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, 2009, pp. 1393-1400.
- [41] S. Avidan, "Support vector tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 4, pp. 1064-1072, 2004.
- [42] S. Hare, A. Saffari and P. H. Torr, "Struck: Structured output tracking with kernels," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 263-270.
- [43] Z. Kalal, K. Mikolajczyk and J. Matas, "Tracking-learning-detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 7, p. 1409, 2012.
- [44] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: a survey," *Foundations and trends® in computer graphics and vision*, vol. 3, no. 3, pp. 177-280, 2008.
- [45] C. Harris and M. Stephens, "A combined corner and edge detector," in *In Fourth Alvey Vision Conference*, Manchester, 1988.
- [46] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286, 1989.

- [47] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *Information Theory, IEEE Transactions on*, vol. 13, no. 2, pp. 260-269, 1967.
- [48] G. D. FORNEY, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268-278, 1973.
- [49] K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499-1503, 2016.
- [50] J. a. F. A. Redmon, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

Anexo A

A Delay-Free and Robust Object Tracking Approach for Robotics Applications

Wilma Pairo, Patricio Loncomilla and Javier Ruiz del Solar

Advanced Mining Technology Center, Universidad de Chile, Chile

Dept. of Electrical Engineering, Universidad de Chile, Chile

wpairo@ing.uchile.cl, ploncomi@ing.uchile.cl, jruizd@ing.uchile.cl

Abstract: In many robotic applications there is the need for detecting and tracking moving and/or static objects while the robot moves, in order to interact with them. High quality detection methods require considerable computational time when the number of objects to be detected is high, or when operating within dynamic, real-world environments. Then, when an object detection result is available, it is referred to a previous frame and not to the current one. A method for obtaining delay-free detections is introduced in this present article. It consists of projecting a delayed detection onto the current frame by using a set of feature tracks generated by using the KLT (Kanade-Lucas-Tomasi) tracker. The proposed method is shown to improve detection accuracy when the tracked object is moving with respect to the camera. In addition, the method is able to detect and manage false detections and occlusions using statistical classifiers (Support Vector Machine) and the Viterbi algorithm [26]. The method is validated in a person-following task, and compared against a part-based HOG person detector, and four performant tracking methods (Meanshift, Compressive Tracking, Tracking-by-detection with Kernels and Kernelized Correlation Filter). Additionally, the method is validated in two additional tasks: face tracking and car tracking. In all reported experiments, the proposed method obtains the best performance among all compared methods.

Keywords: delay-free detections, human tracking, human detection, person following by robot, “follow-me” behavior.

1. Introduction

In many robotics applications there is the need of detecting moving objects (e.g. people, balls, other robots), and/or static objects while the robot moves (e.g. mobile manipulation applications), in order to interact with them. As reliable object detection methods do not run in real time, there is a delay between the object detection request and its result. Thus, the object detection is delayed with respect to the current frame.

There are two main approaches to addressing these kinds of applications, which balance the tradeoff of robustness and delay-free detection in different ways. The first

approach consists of detecting the object in each frame. The main drawback of this approach is that when a robust detector is used (e.g. DPM-HOG – Deformable Part Models [1]), delayed detections are obtained due to the high computational complexity of the process. A delayed detection is one obtained at a time when the object is in a different position from the original one. The second approach consists of tracking the object after initializing the tracking region using an object detector. State-of-the-art tracking methods are very fast, but they normally fail when large variations of illumination occur or when long occlusions happen.

In this article a third approach is proposed that transforms delayed detections into delay-free ones by using a fast feature tracker that estimates the relative movement in the image. Delayed detections are obtained by a robust detection algorithm, DPM-HOG in our case, which is not able to process the image sequence in real-time. The delay-free transformation procedure is based on the KLT feature tracker (Kanade-Lucas-Tomasi feature tracker [2]), and uses the RANSAC (RANdom SAMple Consensus [3]) algorithm for estimating the delay-free transformation. It is important to note that the delay-free transformation procedure is independent of the detection algorithm, and requires only that the object region have some visual textures that can be tracked. In addition, the proposed method manages false detections caused by failures in the primary detection algorithm explicitly, using statistical classifiers (SVM – Support Vector Machine) as well as occlusions, which are modeled using a Hidden Markov Model (HMM) whose state is estimated using the Viterbi algorithm [26].

The method is validated in the “person-following” application, in which a service robot follows a human under unconstrained, real-world conditions: in indoor and/or outdoor environments, with variable illumination, cluttered backgrounds, crowded environments (several persons in the environment), and occlusions caused by people. The method is compared with other detection and tracking methods: DPM-HOG [1], CT - Compressive Tracking [4], TDK - Tracking-by-detection with Kernels [5], Meanshift [6] and KCF - Kernelized Correlation Filter [48] and obtains the best performance among them. Also, the method is tested in two extra tasks: face tracking, and car tracking, obtaining also the best performance in these tasks. It must be noted that the proposed system is able to work with object detectors working at a very low frame rate, but it is also agnostic respect to the object detector being used. Then, the use of high-quality object detectors aimed to the specific task to be solved is encouraged.

The contributions of this paper are three: In the first place, a procedure for transforming a delayed object detection into an up-to-the-minute one by using a delay-free transformation is proposed and implemented. The delay-free detection can be used for initializing a tracker, or directly as the current object detection. The second contribution is the use of a statistical classifier (SVM) that uses features extracted from a pair of images for detecting and rejecting false detections when using a mobile camera. The final contribution is the use of an HMM and the Viterbi algorithm [26] for detecting occlusions (including its start and end) without both needing to have a model of the occluding object and without needing the camera to be static. Then, the system is able to work reliably even when the detectors have high delays and a very low frame rate (less

than 2 fps). Note that even when the delay-free procedure is applied to object detections, the global task to solve is object tracking.

This paper is structured as follows: In Section 2, relevant related work is presented. In Section 3, the proposed delay-free tracking method is described. In Section 4, the method is validated and evaluated against the previously mentioned detection and tracking methods. Finally, in Section 5 conclusions are drawn.

2. Related Work

2.1. Object Detection

There is a wide variety of algorithms for detecting and recognizing objects and people. Some of them are based on local descriptions of the object's patches, while others are based on the sliding window approach. There are also other methods that combine the strengths of both types of methodologies, such as part-based models.

Object detection using local descriptors. This approach consists of selecting salient points, named interest points, from the image of the object to be detected. At each interest point, an oriented patch is selected and a feature vector, named a local descriptor, is computed. Different patches can be differentiated by comparing their local descriptors. An object can be represented by a set of local descriptors. When the same object is present in two images, a significant number of local descriptors are matched between both images. Then, by analyzing the matches, the presence of an object in a test image can be predicted. This kind of method requires that the object contain a significant amount of visual texture, and that the object does not contain repetitive patterns. A descriptor-based object recognition pipeline requires the following components: (i) Detection of keypoints, and generation of local descriptors, (ii) Generation of correspondences between local descriptors from two images, (iii) Clustering of correspondences for generating object detection hypotheses, and (iv) Verification of the object detection hypotheses. State-of-the-art systems use fast generators of binary descriptors (like ORB [7] or FREAK [8]); Hamming distance for generating correspondences (or Flann¹ when the number of training objects is high); Hough transform [9] for detecting sets of coherent correspondences; and a hypothesis verification system such as the L&R system [10]. This last step is needed for reliable object recognition as shown in [11]. For a more detailed explanation of object detection using local descriptors, see [12] and [13]. The main limitation of these methods is the requirement of textured objects, and also that they are not able to achieve robustness and real-time operation at the same time.

Object detection using the sliding-window approach. Feature vectors computed by using sliding windows can be used for detecting the object as a whole. There are several families of algorithms for computing these features, including color histograms, local

¹ FLANN - <http://www.cs.ubc.ca/research/flann/>

binary patterns and histograms of oriented gradients. Different approaches used for detecting objects and/or people by using global descriptors are shown in Table 1.

Object detection using object proposals and convolutional neural networks. Algorithms like Selective Search [29] and EdgeBoxes [30] can be used for generating bounding boxes with high probability of containing an object. Then, convolutional neural networks (CNNs) [36][37][38][39] can be used for classifying the bounding boxes as being background or containing an object. However, this kind of approaches have some weaknesses: (i) the algorithms used for generating object proposals by using the computer’s CPU are very slow (several seconds per image), (ii) approaches based on CNNs both for generating candidate bounding boxes and/or for classifying them (e.g., Faster R-CNN [38], YOLOv2 [46]) require the use of high-end GPUs which are not available in several robotic platforms, and (iii) approaches based on object proposals / CNNs are aimed at detecting a wide variety of object classes, while for some robotics tasks, like person following, only one or a few object classes need to be detected. As an example MT-CNN [47] is a very performant face detector based on CNNs, and is able to run about 2 fps using a CPU. Then, the methodology used for solving a specific task must consider all the constraints involved for obtaining an efficient solution.

Table 1: Methods used for Generating Features from an Object Patch.

Method	Description
Color histogram	Generation of color-based features from an image patch.
LBP	Generation of texture-based features from an image patch [14].
HoG	Generation of gradient-based features from an image patch [15].
DPM-HOG detector	Generation of gradient-based features for deformable models from an image patch [1].

The common approach for obtaining delay-free detections is to speed up the frame rate of the object detector. However, for obtaining faster detectors, some accuracy must be lost as there is an associated trade-off between accuracy and speed. The delay-free transformation procedure proposed in this work is able to produce delay-free and reliable detections from detectors such as DPM-HOG[31], FPD[32] or ACF[33], which are able to run on CPU with minimal memory requirements. Also, machine-learning algorithms are used for rejecting unreliable detections and managing occlusions, which are common in some tracking applications (e.g. person following). Thus, the proposed method is able to complement and to improve the state-of-the-art in object detection.

2.2. Visual Tracking

Visual tracking techniques make following objects in real-time or near real-time possible. Given an initial position of an object in an image, visual tracking is able to predict the movement of the object frame to frame. There are many visual tracking methods. Eight high-performing methods are described in the following paragraphs.

- *Mean Shift*: This method [6] computes a color histogram from a training patch. In the following frames, the system estimates a new position for the center of the bounding box by maximizing the number of pixels whose color distribution is compatible with the training color histogram.

- *Visual tracking using Tracking-by-Detection with Kernels* (TDK) [5]. During operation, the method (re)-trains a classifier at every frame of the video. During the training it uses samples from the previous frame of the object being tracked (thus it is called Tracking-by-Detection). The main difference from similar existing methods is that the training and the classification are done in the Fourier-space thanks to the use of a circulant matrix representation that allows performing the training and the detection at a very high frame rate. The method performs a temporal averaging of the classifier's parameters for including temporal information.

- *High-Speed Tracking with Kernelized Correlation Filters* (KCF) [48]: This method is based also on the use of a circulant matrix representation and Fourier-space analysis. However, it is able to manage multiple channels, and to use a Kernelized Correlation Filter based on HoG features instead of raw pixels.

- *Visual tracking using Compressive Tracking* (CT) [4]. This method extracts Haar-like features from the image and projects them using a sparse matrix for obtaining the features to be used by a classifier. The sparse matrix is selected using a procedure inspired by Compressive Sensing. Features are classified using a naïve Bayes classifier that is updated online over time. As in the case of TDK, a classifier is trained using positive and negative samples obtained from the previous frame.

- *Tracking-learning-detection* [34]: This method decomposes the long-term tracking task into tracking, learning and detection. The tracker follows the object from frame to frame. The detector localizes all appearances that have been observed so far, and corrects the tracker if necessary. The object to be tracked is represented as a set of exemplar patches, which enables the system to learn variations in the aspect of the object.

- *Struck* [49]: This method frames the overall tracking problem as one of structured output prediction, in which the task is to directly predict the change in object configuration between frames. It uses a loss function that considers both bounding boxes and its features jointly, instead of first sampling bounding boxes and then labeling them.

- *Visual tracking by oversampling local features* [35][41]: This family of methods works by aligning a set of local features between two consecutive frames. Local features are computed by using SIFT-like descriptors. Then, methods based on RANSAC are used for obtaining a robust transformation between two consecutive images. While this

family of methods performs well, it requires the object to be textured in order to be described by local features.

- *Direct Visual tracking* [42][43][44][45]: These methods minimize iteratively an error function that depends on the similarity between a template and its projection in the image. Then, an optimal transformation is estimated. Different methods use different formulations for the error function, which can be formulated to be invariant to linear illumination changes [42]. These methods achieve a high accuracy, but given that they assume the object to be rigid or even planar, they are not designed for being robust to partial occlusions and they can fail due to convergence to local minima [43][44][45]. These problems can be overcome by using random forests for learning the relation between the parameters that defines the object's motion and the changes they induce on the image intensities of the template [43]. To track the template when it moves, changes on the image intensities are used to predict the parameters of the motion. This method is able to run up to 2 milliseconds per frame using one core of a CPU [43]. Direct visual tracking methods can also be used for tracking objects from depth images [43][44][45]. Latest methods use hybrid approaches combining feature-based and direct methods for exploiting their comparative advantages [45].

One of the main drawbacks of visual tracking algorithms is that they are not robust to occlusions. Given that visual tracking algorithms update the detection's bounding-box in the current image by searching locally for a patch similar to the original one, when an occlusion is present, none of the patches is similar to the searched one. Then, the bounding box moves randomly, going away from the real position of the occluded object. If the distance between the center of the bounding box and the occluded object is great, the tracker will be unable to recover the target. Consequently, tracking procedures must be complemented with a detector for recovering from tracking losses. If a detector with a high delay is used, and the person moves, then the tracking window will be initialized at the wrong position. Therefore, a delay-free transformation procedure would be useful for initializing trackers that have lost the person to be searched for.

The performance of the method proposed here (based on object detection) will be compared with the ones of four frame-to-frame trackers: CT, TDK, Meanshift and KCF.

2.3.Occlusion detection

In [16], [17], [18], and [19], multiple object tracking is done by detecting multiple objects in consecutive frames, and then estimating occlusions. The main contribution in [16] is the use of projected gradient optimization for tracking ellipse-shaped kernels by using a mean-shift approach. This technique enables tracking each object by selecting a close patch with a similar appearance. The matching score is used for detecting occlusions. In [17], a Gaussian mixture probability hypothesis, density-based visual tracking system with game-theoretical occlusion handling is proposed. A color-based appearance model is generated for generating occlusion hypotheses, which are confirmed or rejected by using game-theoretical algorithms. The system, for working

properly, requires localizing all of the objects. In [18], a Kalman filter is used for tracking each of the persons. Occlusions are detected by analyzing whether patches merge or split. In [19], a part-based model is used for generating multiple detections and then predicting occlusions. In [20] background subtraction is used for detecting body patches. The trajectory for each patch is determined, and then occlusions are detected. In [21] mean-shift is used for multiple object tracking. Each object has a position and a scale, but scale changes are not considered when objects are occluded. Occlusions are recognized by detecting overlapping of patches. In [22], background subtraction is used for detecting patches, and three possible states are considered for managing occlusions: the states are *normal*, *occluded*, and *split*. In the *occluded* state, a constant velocity model is used for predicting the position of the object. In the *split* state, a matching of the patches against the available models is performed. In [23] and [24], a detector of patches containing pairs of persons with different levels of occlusions is trained by using a part-based model. A tracker based on continuous energy minimization is used for predicting the location of the objects. In [25], a particle filter is used for tracking each of the persons, including the occluding objects. The persons can be re-identified by using both color histograms and the distance between the predicted and detected objects for reactivating the tracking hypothesis.

All of the previous occlusion handling methods requires the camera to remain static or to have a model of the occluding objects. The occlusion handling system proposed in this work enables the robot to detect occlusions while moving, without needing to track possible occluding objects. Also, it is able to work using low frame rate detectors (less than 2 fps).

3. Delay-free Tracking

3.1.General Description

The proposed method is able to detect an object in real-time while it moves in the camera reference system. This relative movement can be produced by the sensor (e.g. robot) movement, or by the object movement. The method is able to improve the performance of an object detector in three ways: in the first place, it is able to project the detection computed on a past frame onto the current one. In the second place, it is able to use temporal information for rejecting false detections. And thirdly, it is able to detect occlusions, discarding detections generated by an occluding object.

The system works as follows: First, the object detector starts processing a frame image. The object detector runs at a low or medium frame rate. While the object detector is working, new frames become available. A feature tracker, (KLT [2]), is used for generating and tracking points across these frame images. When the result of the object detection process is available, it is delayed, i.e., it is referred to a previous frame and not to the current one. Then, the feature tracks are used for computing a transformation that is used for projecting the detection's bounding box from the old frame onto the current

one. The set of images contained between an object detection request and the object detector result is named a *Group of Frames*, and it is defined as $GOF^k = \{I_1^k, \dots, I_{n_k}^k\}$, with k being the correlative number of the detection request (see Figure 2).

Thus, the following modules are defined (the block diagram is shown in Figure 1):

- Object Detector: This module analyses an image frame I_1^k in order to detect a given object. In case an object is detected, it generates a bounding box $DetBB_1^k$ as output. The bounding box indicates the position of the object in the image frame I_1^k . Given that the detection process requires some time for generating its result, the module ignores new image frames (named as intra-frames I_u^k) until it finishes the object detection request.
- Feature Tracking: This module gets an image frame I_1^k as input, and compute features (corners) that will be tracked in the next intra-frames. Then, for each intra-frame, the positions of the detected corners in the last frame are updated, and feature tracks τ_u^k that contain the trajectory of the corners' positions through consecutive frames are generated. When the object detection is made, the feature tracks $\tau_{n_k}^k$ that are contained in the detection-bounding box are transmitted to the succeeding modules.
- Delay-free Transformation Estimation: This module gets a bounding box $DetBB_1^k$ referred to the image frame I_1^k , and a set of feature tracks $\tau_{n_k}^k$ as input. As $\tau_{n_k}^k$ contains the position of the tracked corner through the intra-frames, it can be used for computing a transformation T^k that maps the original detection bounding box $DetBB_1^k$ onto a delay-free one $DetBB_{n_k}^k$. The output of this module is the transformation T^k .
- Occlusions & False Detections Management: This module uses the image frame I_1^k , the feature tracks $\tau_{n_k}^k$ and the bounding box $DetBB_1^k$ as input, and it estimates the presence of false detections (*false_det_flag*), short occlusions (*short_occl_flag*) and long occlusions (*long_occl_flag*).
- Delay-free Object Tracking: This module generates the final object bounding box $BestBB^k$ by considering the transformation T^k that needs to be applied over $DetBB_1^k$, and the reliability of the detections (presence of false detections and occlusions). Its output is the best bounding box, and a flag that indicates whether the object is occluded.

Thus, the system is able to generate delay-free detections that are robust against both detection errors and occlusions. In the case that multiple objects are tracked, the feature tracking module must be run only once per frame. The computation of the delay-free transformation is very fast (requires 0.07 milliseconds in an Intel Core i5 using only one core), so the algorithm can scale easily for tracking several objects. Untextured objects can cause sparseness in feature tracks. However, this is not a major problem, given that feature tracks are computed and updated only between two consecutive detections. In addition, untextured objects can be tracked successfully by using features from their boundaries.

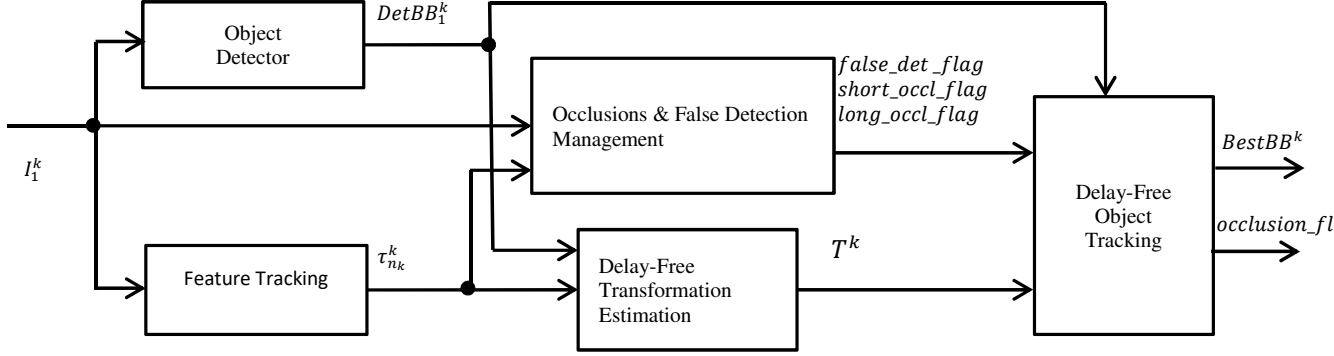


Figure 1: Block diagram for the proposed algorithm. An image stream is fed into an object detector and a feature tracker (KLT). The object detector needs time for processing, generating a delayed detection. The tracked features can be used for projecting the delayed object detection onto the current frame. Also, the system is able to detect occlusions and recover the target (see main text for details).

3.2. Delay-free Transformation Estimation

The delay-free transformation estimation procedure transforms a delayed detection onto a delay-free one. A detection starts in the first image frame I_1^k of the GOF^k and finishes in the last frame $I_{n_k}^k$. Super-indices indicate the GOF number, and sub-indices are used to indicate the frame order within the GOF , as shown in Figure 2. Thus, when a frame I_1^k is available, an object detector is started in that frame. In addition, corner features are detected inside I_1^k by using a Shi-Tomasi corner extractor that is used for initializing the KLT algorithm [2]. The object detector requires several frames for executing its task, hence intra-frames I_u^k are available. For each intra-frame, the positions of the corners detected in I_1^k are updated, and so-called *feature tracks* that contain the trajectory of each corner-position through consecutive frames are generated. The set of feature tracks is composed of N individual feature tracks $\tau_u^k[i]$, defined as:

$$\tau_u^k[i] = (x_1, y_1, time_1, \dots, x_u, y_u, time_u) \quad (1)$$

where x_i, y_i represent the position of the i -th corner in time step $time_i$.

When the detector finishes its task (in the frame $I_{n_k}^k$), the feature tracks $\tau_{n_k}^k$ that are contained in the detection bounding box $DetBB_1^k$ are used for obtaining a transformation T^k that enables projecting $DetBB_1^k$ onto a new bounding box $DetBB_{n_k}^k$, referred to the last frame $I_{n_k}^k$. A translation and scale transformation is used for relating the two bounding boxes. This transformation is determined by a translation t_x, t_y and a

scale factor s . The equations used for computing the transformation for a minimal sample composed of two feature's tracks $\tau_{n_k}^k[a]$ and $\tau_{n_k}^k[b]$, are:

$$s = \frac{\sqrt{(\tau_{n_k}^k[a] \cdot x_{n_k} - \tau_{n_k}^k[b] \cdot x_{n_k})^2 + (\tau_{n_k}^k[a] \cdot y_{n_k} - \tau_{n_k}^k[b] \cdot y_{n_k})^2}}{\sqrt{(\tau_{n_k}^k[a] \cdot x_1 - \tau_{n_k}^k[b] \cdot x_1)^2 + (\tau_{n_k}^k[a] \cdot y_1 - \tau_{n_k}^k[b] \cdot y_1)^2}} \quad (2)$$

$$t_x = \tau_{n_k}^k[a] \cdot x_{n_k} - s * \tau_{n_k}^k[a] \cdot x_1 \quad (3)$$

$$t_y = \tau_{n_k}^k[a] \cdot y_{n_k} - s * \tau_{n_k}^k[a] \cdot y_1 \quad (4)$$

The transformation T^k is computed by using the RANSAC algorithm.

The procedure is presented in Algorithm 1, and illustrated in Figure 3. In each iteration of the algorithm, a pair of feature tracks are selected randomly, and used for computing a transformation hypothesis T . If most of the other tracks are compatible with T , the hypothesis is accepted. If not, a new pair of features is selected and the process repeated. The variable *maxNumIter* corresponds to the maximum number of iterations in RANSAC. It must be noted that in this work, the delay-free tracker is used for person tracking. However, the delay-free transform can be applied to any object detection problem.

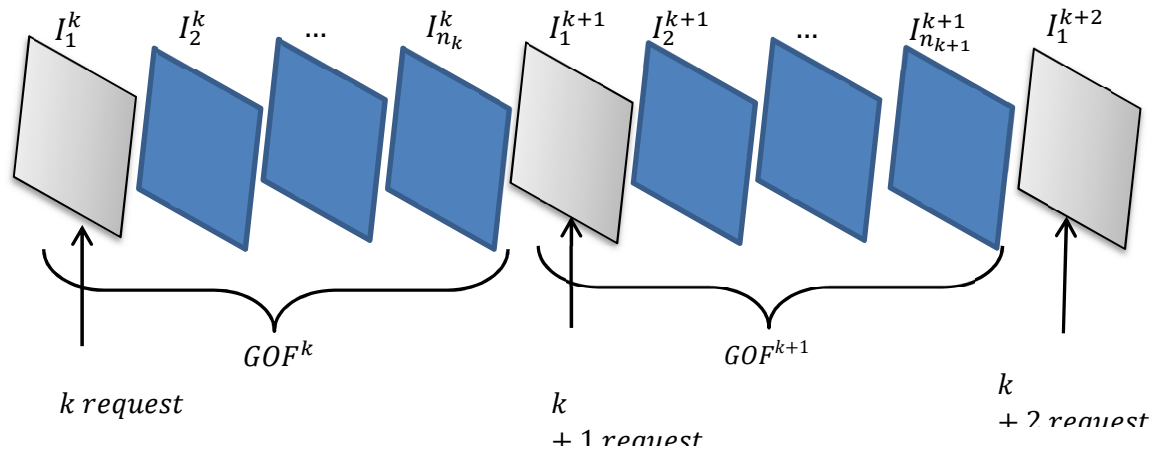


Figure 2: Diagram of the *Group of Frames (GOF)* Concept. The object detector processes an image, but it requires time for computing the detection. The set of frames between the object detection request and the object detection response is named a Group of Frames. A Group of Frames can be used for undelaying the object detection.

Algorithm 1: RANSAC-based algorithm used for the delay-free transformation estimation procedure.

Function $ransac_estimator(DetBB_1^k, \tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N])$:

17. Compute Q as the subset of feature tracks that are contained inside $DetBB_1^k$
18. **for** $i \leftarrow 1$ **to** $maxNumIter$
19. Draw two random feature tracks $\tau_{n_k}^k[a]$ and $\tau_{n_k}^k[b]$ from Q
20. Compute a transformation hypothesis $T = (s, t_x, t_y)$ that relates $\tau_{n_k}^k[a], \tau_{n_k}^k[b]$ between both images by using equations (2), (3), (4)
21. Compute $Q_{consensus}$, i.e., the elements in Q that are compatible with T
22. **if** $size(Q_{consensus}) > threshold$ **then**
23. **for** $j \leftarrow 1$ **to** $maxNumIter$
24. Draw two random feature tracks $\tau_{n_k}^k[a]$ and $\tau_{n_k}^k[b]$ from $Q_{consensus}$
25. Compute a transformation hypothesis $T = (s, t_x, t_y)$ that relates $\tau_{n_k}^k[a], \tau_{n_k}^k[b]$ between both images by using equations (2), (3), (4)
26. $S[j] \leftarrow s, T_x[j] \leftarrow t_x, T_y[j] \leftarrow t_y$
27. **end**
28. select $S[imed]$ as the median of S
29. **return** $T = (S[imed], T_x[imed], T_y[imed])$
30. **end**
31. **end**
32. RANSAC failed; **return** $T = identity_transform$

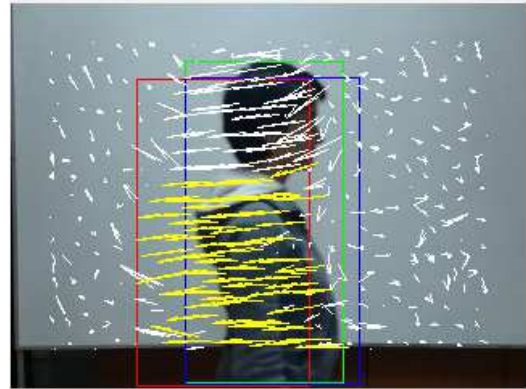
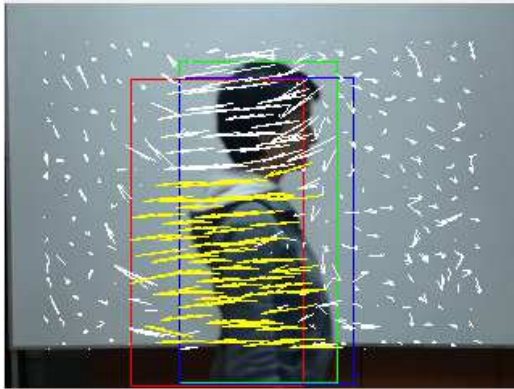


Figure 3: Example of the delay-free detection procedure. Images on the left and the right represent consecutive detection frames, with different feature tracks computed. The current ground-truth bounding box is shown in green. The delayed bounding box is shown in red. The delay-free bounding box is shown in blue. Feature tracks are shown as lines. Feature tracks accepted by RANSAC and used for computing the delay-free transform are shown in yellow.

3.3. Occlusions & False Detections Management

The proposed method estimates the occurrence of the following three situations: (i) a false detection of the object in I_1^k occurs (e.g. the tracked object is mixed up with a different object); (ii) the tracked object is occluded by a second object of the same category (e.g. a tracked person is occluded by a second person) in I_1^k , and the second object is detected; this situation is called a *long-occlusion*, and (iii) the tracked object is occluded by a second object in an intra-frame image I_u^k , and the feature tracks are corrupted; this situation is called a *short-occlusion*. In other words, case (i) corresponds to a false detection, case (ii) corresponds to a long occlusion (object keeps occluded after a detection frame), and case (iii) corresponds to a short occlusion (object occluded between two detection frames). In the first two cases, the current object detection is rejected, and the previous detection is used ($BestBB^k = BestBB^{k-1}$). In the third case, the obtained tracks are invalid. Therefore, the delay-free transformation cannot be computed. Hence, the detection in I_1^k is not projected and it is used as the final one ($BestBB^k = DetBB_1^k$).

The system must be initialized properly because, if the first detection is incorrect, the system will not be able to recover. In the case of the person following task, the person to be tracked stands in front of the camera when the following task starts.

False Detections Management

The method estimates the occurrence of a false detection by comparing the current detection, before applying the delay-free transformation, $DetBB_1^k$ with the previous one $DetBB_1^{k-1}$. In case the difference between them (in appearance or position) is large enough, $DetBB_1^k$ is considered to be a false detection, and the previous detection is used.

The problem is modeled as a two-class classification problem, with the following two classes: two consecutive detections are similar, and two consecutive detections are different. A statistical classifier is trained using real examples of similar detection-pairs and different detection-pairs. (In our case we use an SVM with a RBF Kernel, but any other classifier could be used.)

A feature vector with seven components is used to train the statistical classifier. The first two components are related to the color content of the bounding boxes of

$DetBB_1^{k-1}$ and $DetBB_1^k$, and the other five are based on the position information. Then, the classifier is able to use information from both sources for discriminating between true and false detections. The seven components/features, which are computed by a function named $comp_feat_{falseDet}()$ are:

- the Euclidean distance between the RGB histograms of $DetBB_1^{k-1}$ and $DetBB_1^k$,
- the Euclidean distance between the HSV histograms of $DetBB_1^{k-1}$ and $DetBB_1^k$ (only the H and S channels are used),
- the difference of x position between $DetBB_1^{k-1}$ and $DetBB_1^k$,
- the difference of y position between $DetBB_1^{k-1}$ and $DetBB_1^k$,
- the ratio between the areas of $DetBB_1^{k-1}$ and $DetBB_1^k$,
- the area of the BB (Bounding Box) corresponding to $DetBB_1^k$, and
- the IoU score between $DetBB_1^{k-1}$ and $DetBB_1^k$:

$$score(BB_1, BB_2) = \frac{intersection(BB_1, BB_2)}{union(BB_1, BB_2)} \quad (5)$$

Initially 12 features were considered, as the original feature vector included also LBP distance between the two bounding boxes, center of mass (x,y) of the first bounding box, and center of mass (x,y) of the second bounding box. Several possible subsets of features were evaluated by using 10-fold cross validation and a RBF SVM classifier. The seven selected features were the best subset tested in these evaluations. Then, the discarded features were not useful for the task of detecting false positives from the detector.

In case more than one detection is available in the current image, the error rejection procedure is applied to all of them. Rejection of false detections is a very important step because they can cause the system to get confused and to track another object. Results show that false detections can be robustly detected by using the described procedure.

Short Occlusions Management

A group of frames GOF^k contains a short occlusion when some of its intra-frames are occluded, but the first image frame I_1^k is not occluded, i.e., the occlusion occurs between two consecutive detections. In this case the occlusions do not affect the bounding boxes in the detection frames. However, this kind of event causes the feature tracks to be unreliable for generating the delay-free transformation. Short occlusions are characterized by analyzing the behavior of the feature tracks. When the tracked object is completely occluded, its feature tracks disappear. This fact can be used for obtaining features that describe the feature tracks, and they can be used for predicting short occlusions. As the object is being occluded, its feature tracks disappear along the occluder movement direction. This fact can be used for both detecting occlusions and estimating the movement of the occluder by analyzing the set of positions and the time in which each feature disappears. In the case of the person-following task, the feature

tracks are lost from left to right, or from right to left, depending on the movement direction of the occluding object. Then, the horizontal position x and the time t in which each track disappears can be stored as pairs (x, t) . A straight line can be fitted to this data by using least squares. Then, the line slope m , the correlation coefficient r and the ratio of the lost features p with respect to the initial ones can be used in building a feature vector (m, r, p) . The straight line parameter n is not used because it is related only to the position of the bounding box. For computing the least squares model, the following equations are used:

$$m = \frac{\sum(x_i - \bar{x})(t_i - \bar{t})}{\sum(x_i - \bar{x})^2} \quad (6)$$

$$r = \frac{\sum(x_i - \bar{x})(t_i - \bar{t})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(t_i - \bar{t})^2}} \quad (7)$$

$$p = \frac{\text{size}\{(x, y, t) \text{ with } t = t_{last}\}}{\text{size}\{(x, y, t) \text{ with } t \geq 1\}} \quad (8)$$

A linear classifier assumes the detection to be correct when $\|m\| > m_{threshold}$, $\|r\| > r_{threshold}$ and $p < p_{threshold}$. In our work, the thresholds were determined as 0.005, 0.3 and 0.3 respectively.

The procedure is named *threshold_classifier*, and it is shown in Algorithm 2. In a resumed description, the procedure works by collecting pairs (x, t) of features inside the tracked object that disappear at different times t . A linear model $x = mt + n$ with correlation coefficient r is fit onto the data. The term m is related to the speed of the occluding object, as it is the cause of the loss of feature tracks. If the term $\|m\|$ is over a threshold, the correlation coefficient $\|r\|$ is high, and the amount of surviving tracks p is low, then a short occlusion is detected.

Algorithm 2: Threshold-based classifier for detecting short occlusions.

Function *threshold_classifier*($\tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N]$):

```

17. Set nLastFrame = 0
18. for  $i \leftarrow 1$  to  $N$ 
19.    $x[i] \leftarrow \tau_{n_k}^k[i].x_{last}$ 
20.    $t[i] \leftarrow \tau_{n_k}^k[i].time_{last} - \tau_{n_k}^k[i].time_1$ 
21.   if  $\tau_{n_k}^k[i].time_{last} = lastFrame$  then
22.      $nLastFrame \leftarrow nLastFrame + 1$ 
23.   end
24. end

```

```

25.  $(m, r) \leftarrow \text{least\_squares}(x, t)$ 
26.  $p \leftarrow n\text{LastFrame}/N$ 
27. if  $\|m\| > m_{\text{threshold}}$  and  $\|r\| > r_{\text{threshold}}$  and  $p < p_{\text{threshold}}$  then
28.    $\text{short}_{\text{occl}_{\text{flag}}} \leftarrow \text{true}$ 
29. else
30.    $\text{short}_{\text{occl}_{\text{flag}}} \leftarrow \text{false}$ 
31. end
32. return  $\text{short}_{\text{occl}_{\text{flag}}}$ 

```

Long Occlusions Management

When a long occlusion occurs, a different object from the one being tracked is detected in the image frame I_1^k . The detection of occlusions is important for tasks like person following because undetected occlusions can cause the system to follow the occluding object (person) instead of the original one.

The occlusion problem is modeled as having two states: occluded and non-occluded. As the state (occluded or non-occluded) is unknown, the state machine corresponds to a Hidden Markov Model (HMM). There are transition probabilities $p(x_t|x_{t-1})$ and observation likelihoods $p(z_t|x_t)$. In this work, the transition probabilities are assumed to be constant (0.5). The observation likelihoods depend on both the current state and the observation vector. The machine state model used in this work is shown in Figure 4.

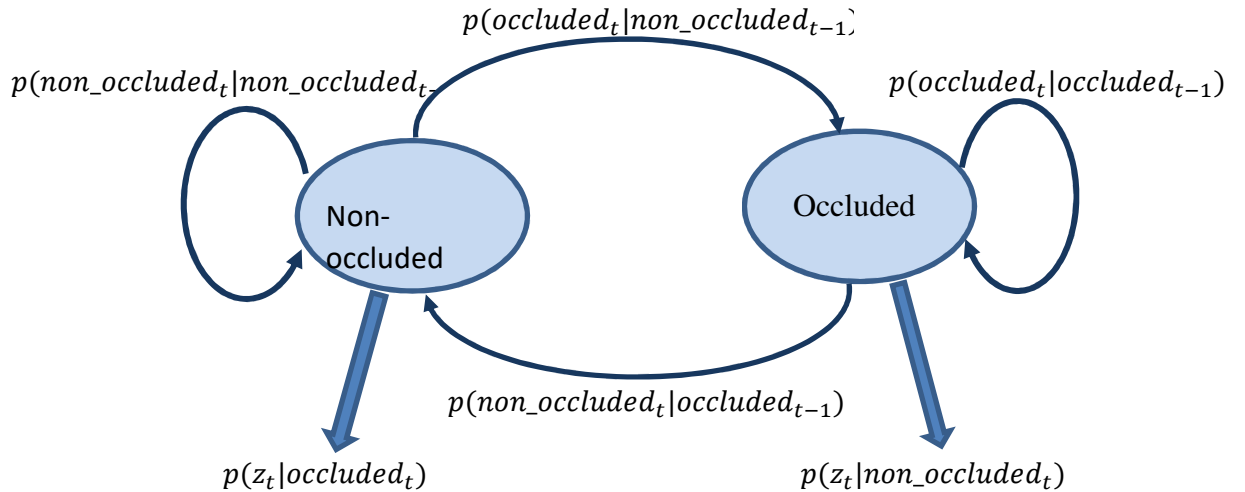


Figure 4: HMM used for Modeling Occlusions. The model considers two states: non-occluded and occluded. By considering both state transition probabilities and observations related to the available images, the system can predict if the object being tracked is or is not occluded.

The observation vector z is composed of three binary variables: *patch_change*, *features_entering* and *features_leaving*:

$$z = (\textit{patch_change}, \textit{features_entering}, \textit{features_leaving}) \quad (9)$$

The variable *patch_change* is a flag that indicates if the image patch related to the current detection is similar or different from the patch related to the previous one. The start and end of occlusions generate important changes in the graphical content of the patch; then abrupt patch changes can be detected and used for occlusion detection. A statistical classifier is trained using real examples of consecutive detections corresponding to the same object (*patch_change* = 0) and consecutive detections corresponding to different objects (*patch_change* = 1). The following features are used to characterize the visual similarity of the detected objects:

- the Euclidean distance between the LBP histograms of both BB, and
- the Euclidean distance between the HSV histograms of both BB (only the H and S channels are used).

Persons with similar clothes can lead the system to mismatch the person to be followed. Then, features related to motion (based on feature tracks) are also used. The variables *features_entering* and *features_leaving* indicate if there is a significant number of feature tracks entering (*nin*) or leaving (*nout*) the detection bounding boxes. The *nin* and *nout* measures are used because when the tracked object is being occluded, feature tracks from the occluding object go inside the detection bounding box. Also, when the occluding object goes out, its feature tracks go outside the detection bounding box. Then, the number of track features that go inside or outside the bounding box can be used for discriminating between the starts and ends of occlusions. This measure is robust against translations because, in this case, the track features that go inside the bounding box on one side are almost equal to the number that leave the bounding box on the other side. The variables are defined as follows:

$$\textit{features_entering} = \begin{cases} 1 & \text{if } nin > 15 \text{ and } nin > 2 * nout \\ 0 & \text{else} \end{cases} \quad (10)$$

$$\textit{features_leaving} = \begin{cases} 1 & \text{if } nout > 15 \text{ and } nout > 2 * nin \\ 0 & \text{else} \end{cases} \quad (11)$$

The Viterbi algorithm [26] is an optimal state estimator for HMMs, so it is used for detecting occlusions in this work. The long occlusion state estimation is implemented in a function *viterbi()* that implements the computation of the features z and the estimation of the new state by using $p(x_t|x_{t-1})$ and $p(z|x_t)$.

The method used for solving the data association problem compares features obtained from two different detections, and it decides whether both detections correspond to the same person or not. The training procedure considers both coincident and non-coincident pairs of detections from persons in the training database. Then, the method is not dependent on the specific clothing being used by the person.

It must be noted that errors in the estimation of the ending of the occlusion could cause the system to confuse and track the occluding object. However, in the performed experiments, no tracking confusion occurred.

3.4.Delay-Free Object Tracking

As already explained, the delay-free tracker is based on estimating a delay-free transform T for applying it to an object detection $DetBB_1^k$. This algorithm requires estimating a set of feature tracks τ_u^k for computing the transformation upon request. Also, information about false detections, short occlusions and long occlusions is used for defining if the detection is assumed to be valid, and for defining if the delay-free transformation will be applied to the detection or not. The algorithm is detailed in Algorithm 3.

The full system works by processing a stream of images. It uses two threads: a tracker thread and an object detector thread. The tracker thread receives a stream of images as input. When a new image arrives and the object detector is not activated, the object detection thread is started. While the object detector is working, the tracking thread tracks feature points in the image. When the object detector generates a detection response, the detection bounding box is updated by using the feature tracks, by applying a delay-free transformation. Short occlusions are detected when an occluding object moves into the tracked object and generate loss of its tracked points, making impossible the application of a delay-free transform. Also, long occlusions are detected when the features related to the object detection response are different from those expected from the tracked object. An HMM is used for estimating starts and endings of occlusions, by considering both the feature tracks and the features of the current object detection (bounding boxes and color histograms from the corresponding image patches).

Algorithm 3: Final delay-free detection and tracking procedure.

```

38. Set  $k \leftarrow 1$ 
39. while(images remaining)
40.   Capture  $I_1^k$ 
41.   Start process  $object\_detect(I_1^k)$  in a secondary thread
42.   Initialize feature tracks  $\tau_1^k$  by detecting corners in  $I_1^k$ 
43.   Set  $u \leftarrow 1$ 
44.   while( $object\_detect$  has not finished)
45.      $u \leftarrow u + 1$ 
46.     Capture  $I_u^k$ 
47.      $\tau_u^k \leftarrow update\_tracks\_using\_KLT(\tau_{u-1}^k, I_u^k)$ 
48.   end
49.    $n_k \leftarrow u$ 

```

```

50.    $DetBB_1^k \leftarrow object\_detection\_result()$ 
51.    $long\_occl\_flag \leftarrow$ 
       $viterbi(\tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N], DetBB_1^k, I_1^k, DetBB_1^{k-1}, I_1^{k-1})$ 
52.   if ( $long\_occl\_flag$  is false) then
53.        $feat_{falseDet}^k \leftarrow comp\_feat_{falseDet}(DetBB_1^k, I_1^k, DetBB_1^{k-1}, I_1^{k-1})$ 
54.        $false\_det\_flag \leftarrow classifier_{falseDet}(feat_{falseDet}^k)$ 
55.       if ( $false\_det\_flag$  is false) then
56.            $short\_occl\_flag \leftarrow threshold\_classifier(\tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N])$ 
57.           if ( $short\_occl\_flag$  is false) then
58.                $T^k \leftarrow ransac\_estimator(DetBB_1^k, \tau_{n_k}^k[1], \dots, \tau_{n_k}^k[N])$ 
59.                $BestBB^k \leftarrow T^k(DetBB_1^k)$ 
60.                $occlusion\_flag \leftarrow 0$ 
61.           else
62.                $BestBB^k \leftarrow DetBB_1^k$ 
63.                $occlusion\_flag \leftarrow 1$ 
64.           end
65.       else
66.            $BestBB^k \leftarrow BestBB^{k-1}$ 
67.            $occlusion\_flag \leftarrow 0$ 
68.       end
69.   else
70.        $BestBB^k \leftarrow BestBB^{k-1}$ 
71.        $occlusion\_flag \leftarrow 1$ 
72.   end
73.    $k \leftarrow k + 1$ 
74. End

```

4. Experimental Evaluation

The proposed method is validated in the “person-following” application, in which a service robot follows a human under unconstrained, real-world conditions: indoor and/or outdoor environments, variable illumination, cluttered background, crowded environment (several persons in the environment), and people occlusions. Person-following is considered a relevant ability of service robots that will interact with humans in daily life environments.

4.1. Experimental Setup

A database containing video sequences captured under different situations was built². In the videos, the setup consists of a camera following a person. The camera is pointing forward, and it is at a height of approximately 1.50 mt. The camera moves freely, so it is assumed that its intrinsic and extrinsic parameters are not known, i.e., the system is not dependent on any kind of camera calibration. Captured images have a resolution of 640x480; however, they are reduced to 320x240 when being processed. This resolution is enough for solving the person-following task as it is shown in Section 4.3. The person to be tracked stands in front of the camera when a video starts.

As it can be seen in Table 2, the video sequences include different real-time conditions (see Figure 5). A total of 44,069 frames were used for training the detection system, and 41,289 frames were used as the test database. The training and testing databases are composed of the following video sequences:

Sets of videos used for training:

Set 7: 20 training videos without occlusions (v1, v3, v5, v7, v9, v11, v13, v15, v17, v19, v21, v23, v25, v27, v29, v31, v33, v35, v37, v39).

Set 8: 11 training videos with occlusions (vo_4, vo_5, vo_12, vo_13, vo_14, vo_15, vo_16, vo_17, vo_18, vo_19, vo_20).

Sets of videos used for testing:

Set 9: 17 testing videos without occlusions (v2, v4, v6, v8, v10, v12, v14, v16, v18, v20, v22, v24, v26, v28, v30, v32, v34).

Set 10: 9 testing videos with occlusions (vo_1, vo_2, vo_3, vo_6, vo_7, vo_8, vo_9, vo_10, vo_11).

Set 11: 7 testing videos from an earlier work [27].

Set 12: 4 testing videos with increasing occlusion times.

Table 2: Videos used for Training and Evaluating the Different Person-tracking Methods.

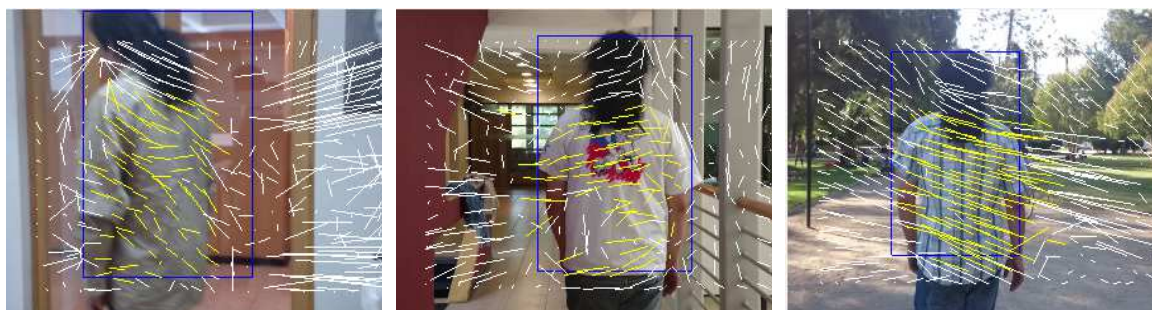
Videos	Set	Description	Occlusion
v1, v2, v3, v4, v5, v6, v7	1, 3	Good illumination, one person walking in front of the camera with a 1.5m of distance.	No
v8, v9, v10, v11 v12	1, 3	Complex scenario, because the illumination changing and exist sun reflection around the person.	No
v13, v14, v15, v16, v17	1, 3	Complex scenario, initializing with good illumination, exist lighting changing and exist sun reflection around the person.	No
v18, v19, v20, v21, v22	1, 3	Complex scenario, because exist a bad illumination.	No
v23, v24, v25, v26, v27, v28, v29, v30,	1, 3	Complex scenario, because the person walking in a public park and the illumination is totally variable,	No

² This database will be made public, after finalizing the reviewing process.

v31, v32, v33, v34, v35, v37, v39		people in the background.	
vo_1, vo_2, vo_3, vo_4	2, 4	Complex scenario, because the person walking in a public park and the illumination is totally variable and exist occlusion, because the person walks around of trees and objects.	Yes
vo_5, vo_6, vo_7, vo_8, vo_9, vo_10, vo_11, vo_12, vo_13, vo_14, vo_15, vo16, vo_17, vo_18, vo_19, vo_20	2, 4	Complex scenario, because the person walk in a street, the illumination is totally variable, people in the background and exists occlusions.	Yes
V4, V5, V6, V7, V8	5	Good illumination and complex scenario exists interaction w/ people, exits a lot of people in the background and occlusions, and in V8 subject enters elevator.	Yes
VD2, VD3	5	Complex scenario, people in the background, exist interaction w/people and exist occlusions.	Yes
vo1, vo2, vo3, vo4	6	Subject remains at a fixed distance, exist occlusions with different time of duration.	Yes

All the video sequences were labeled with their ground truth. One in every 10 frames was hand-labeled and the intermediate frames were marked using linear interpolation. Frames with occlusions were also labeled with a second set of tags. The person to be tracked should be in front of the camera when the video begins, for initialization purposes. Thus, there is no ambiguity in which person to track. Note that feature tracks (shown in Figures 3 and 5) follow the right person even when multiple persons are present. Also note that occlusions can be detected as the tracks related to the occluded person disappear.

There are two indicators used for comparing the performance of the different methods: the detection rate (DR), defined as the number of true positive detections divided by the total amount of ground-truth detections, and the number of false positives (FP). A detection is considered correct if it is compatible with the ground truth bounding box ($IoU > 0.5$), and the object is not being occluded.



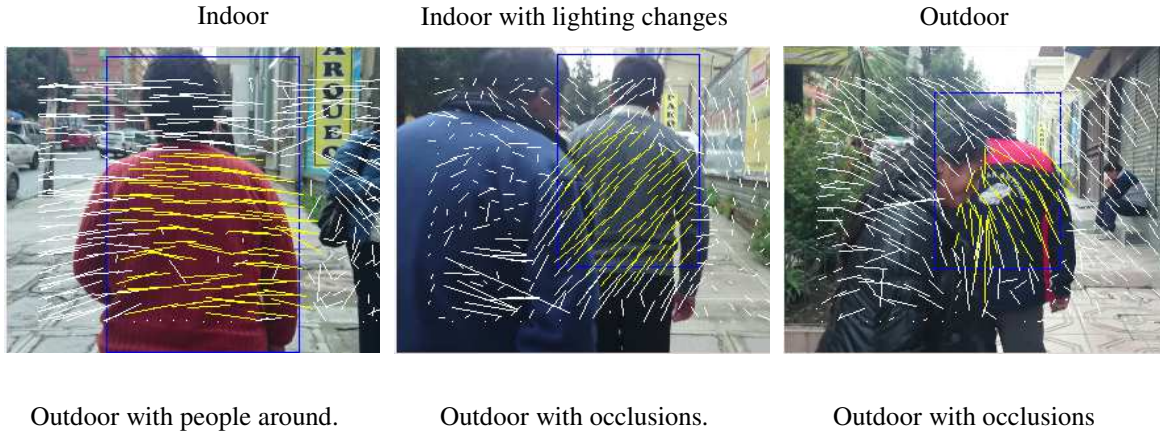


Figure 5: Image examples of the real-world video sequences used for the experimental evaluation. The delay-free bounding box is shown in blue. Feature tracks are shown as lines. Feature tracks accepted by RANSAC and used for computing the delay-free transform are shown in yellow.

4.2. Methods under Evaluation

The following detection and tracking methods are compared:

- HoG1/2: This term is used in [27] for describing a DPM-HOG human-torso detector based on [1]. This is the same as the standard HOG, but only the upper parts (torsos) of the subjects are used for the training of the detector. This allows the system to be able to detect subjects standing close to the robot --where only the torso is visible--, but also in cases when the subject is far away from the camera. Positive and negative examples for the SVM classifier are extracted from the person-following training dataset. This method generates a delayed detection $DetBB_1^k$ referred to the frame I_1^k that is older than the current frame $I_{n_k}^k$.

- HoG1/2-I: This is an ideal detector (the I stands for ideal) based on HoG1/2, computed offline. It generates a detection HoG_1^k instantly, i.e., when the current frame is I_1^k . It is used as an upper bound for the delay-free transformation procedure because the result from that procedure cannot be better than HoG1/2-I.

- DF-Simple (Delay-free Tracking): This method is based on HoG1/2. However, it projects the detection $DetBB_1^k$ onto a delay-free detection $DetBB_{n_k}^k$ by using the available feature tracks.

- DF-Full: This method is similar to DF-Simple Tracking, but it also uses SVM classifiers for rejecting false detections and occlusion candidates, and an HMM model for estimating the presence of occlusions.

- CT: Compressive tracking [4]. In this method, Haar-like features are used for tracking a moving patch inside an image sequence. It is inspired by Compressive Sensing.

- TDK: Tracking-by-detection with Kernels [5]. This method trains a classifier in each frame in Fourier-space. It has a very high frame-rate.

- Meanshift [6]: This method computes a color histogram from an image. In subsequent images, it searches for a window with maximal similarity between the training histogram and the current one.

- KCF: Kernel Correlation Filter [48]: Similar to TDK, but can use HoG-like features as channels.

For the HoG1/2 person detector, the original implementation of Felzenszwalb [1]³ was used and retrained with images containing torsos. CT [4]⁴ and TDK [5]⁵ also have publicly available implementations. The free Meanshift implementation from [6]⁶ was used. The KLT [2] algorithm⁷ was integrated in the system for computing the feature tracks. OpenCV was used for processing images and for training and using SVMs. The optimal RBF SVM parameters were selected automatically by OpenCV by using cross-validation on the training data over a parameter grid.

HoG1/2, HoG1/2-I, CT were executed by using their standard MATLAB based implementations. Meanshift, KLT and OpenCV are implemented and tested in C++.

4.3.Results

Table 3 shows the performance of the different methods on the videos corresponding to Set 3 (no occlusions). As it can be observed, the use of the delay-free transform (DF-simple) outperforms the original HoG1/2 procedure, as the detection rate is increased and the number of false detections is decreased. The use of the delay-free transform enables increasing the detection rate from 0.88 up to 0.90, which is close to that of the HoG1/2-I (0.93). The HoG1/2-I method is an upper bound for the detection rate of DF-Simple. The DF-full procedure is able both to reject false detections and manage occlusions, and so it can improve the system by decreasing only the number of false detections, but at the cost of rejecting some good detections. In the videos shown in Table 3, the use of DF-Full generates an improvement in the number of false positives that decreases from 8.18 down to 2.24, i.e., it decreases in a 73%. However, the detection rate decreases from 0.88 to 0.83. The tracking systems, CT and TDK, have a low detection rate (below 0.7), and a false positive rate much higher than that of the

³ Implementation used: <http://www.cs.berkeley.edu/~rbg/latent/index.html>

⁴ Implementation used: <http://www4.comp.polyu.edu.hk/~cslzhang/CT/CT.htm>

⁵ Implementation used: <http://www2.isr.uc.pt/~henriques/circulant/index.html>

⁶ Implementation used: <http://www.cs.bilkent.edu.tr/~ismaila/MUSCLE/ObjectTracker.cpp>

⁷ Implementation used: <https://www.ces.clemson.edu/~stb/klf/>

HoG-based methodologies. Meanshift has a 0.15 detection rate, and 58 false positives, making it unreliable for person tracking. KCF has a 0.77 detection rate, which is higher than that of the previous trackers, but it is also outperformed by the proposed method in both detection rate and false positives.

Table 3: Detection Rate for Videos of Set 3 (no occlusions) for DF, CT, TDK, KCF and Meanshift. Nframes: Number of frames. NGOFS: Number of Group of Frames (see main text for details). DR: Detection Rate. FP: Number of False Positives.

video	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
v2	1951	64	0.90	6	0.10	58	0.16	54	0.54	29	0.15	55	0.92	5	0.74	3	1.00	0
v4	1901	63	1.00	0	0.85	9	0.29	45	0.81	12	0.11	56	0.98	1	0.97	1	1.00	0
v6	1931	61	0.90	6	0.39	37	0.29	43	0.99	1	0.16	51	0.90	6	0.92	1	0.93	4
v8	2281	73	0.90	7	0.17	60	0.12	64	0.60	29	0.25	55	0.90	7	0.86	2	0.92	6
v10	1901	63	0.85	9	0.47	33	0.04	61	0.99	1	0.17	52	0.78	13	0.65	5	0.85	9
v12	1931	64	0.89	7	0.02	63	0.02	63	0.61	25	0.15	54	0.92	5	0.59	5	1.00	0
v14	3621	118	0.88	14	0.11	105	0.02	116	0.97	4	0.23	90	0.97	4	0.97	3	1.00	0
v16	3621	114	0.83	19	0.77	26	0.29	82	0.98	2	0.18	94	0.85	16	0.82	2	0.86	15
v18	1861	57	0.89	6	0.85	9	0.52	27	0.37	36	0.10	51	0.91	5	0.89	3	0.93	4
v20	1881	61	0.86	8	0.45	34	0.31	42	0.19	49	0.15	52	0.90	6	0.81	3	0.92	5
v22	1931	64	0.89	7	0.58	27	0.16	54	0.42	37	0.12	57	0.92	5	0.57	3	0.95	3
v24	1941	64	0.90	6	0.70	19	0.96	3	0.72	18	0.09	58	0.98	1	0.95	0	1.00	0
v26	1861	62	0.98	1	0.11	55	0.15	53	0.99	1	0.17	52	0.98	1	0.98	1	1.00	0
v28	1861	62	0.94	4	0.10	56	0.03	60	0.99	1	0.11	55	0.98	1	0.97	2	1.00	0
v30	1911	61	0.65	19	0.21	48	0.75	15	0.99	1	0.10	55	0.70	16	0.69	2	0.74	14
v32	1991	66	0.89	7	0.79	14	0.16	55	0.99	1	0.13	57	0.97	2	0.97	2	0.98	1
v34	1841	49	0.72	13	0.08	45	0.89	5	0.99	0	0.16	41	0.77	11	0.74	0	0.79	10
Average	2130.41	68.59	0.8	8.1	0.4	41.11	0.3	49.54	0.77	14.47	0.15	58.00	0.90	6.18	0.83	2.24	0.93	4.18

video	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
			8	8	0		0											

Table 4 shows the performance of the different methods on the videos corresponding to Set 4 (occlusions). As can be observed, the delay free procedure (DF-Simple) outperforms the original HoG1/2 tracker, as it has better performance both in terms of detection rate and number of false positives. The use of the delay-free transform enables increasing the detection rate from 0.84 up to 0.89, which is closer to that of the HoG1/2-I (0.96). The use of the SVM classifiers and the HMM model for occlusions (DF-full) is able to decrease the number of false positives from 10.44 to 5.33, but its DR decreases from 0.84 to 0.79 because there is a tradeoff between the detection rate and the number of false positives. The classical tracking systems (CT, TDK, Meanshift) perform very poorly in videos with occlusions. Also, KCF is clearly beaten by the proposed method.

Table 5 shows the performance of the different methods on the videos corresponding to Set 5 (occlusions). The videos from set 5 were captured by using a Kinect camera, so the quality of the images is lesser than that of the other databases, and the feature tracks are less reliable than in the other databases. The use of the delay-free transform (DF-Simple) improves the detection rate (0.87) and the number of false positives (7.71). However, its detection rate is not close to HoG1/2-I (0.96) because of the unreliability of the feature tracks. The use of the SVM classifiers and the HMM model (DF-Full) decreases the FP from 9.57 to 4.43, but the DR decreases from 0.84 to 0.64 as the classifiers mislabel some correct detections. Again in this database, the classical tracking algorithms (CT, TDK, Meanshift) have very deficient performance. KCF performs better than the previous trackers, but it is largely outperformed by the proposed system.

Table 4: Detection Rate for Videos of Set 4 (occlusions) for DF, CT, TDK, KCF and Meanshift. Nframes: Number of frames. NGOFs: Number of Group of Frames (see main text for details). DR: Detection Rate. FP: Number of False Positives.

video	Nframes	NGOFs	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
vo_1	1281	28	0.63	10	0.04	27	0.06	27	0.13	24	0.10	26	0.96	1	0.33	1	0.89	3
vo_2	1151	36	0.59	14	0.16	31	0.02	35	0.24	27	0.10	33	0.78	9	0.67	1	0.96	4
vo_3	1441	46	0.74	12	0.04	44	0.03	45	0.96	2	0.14	40	0.77	11	0.70	4	0.91	5
vo_6	1871	62	0.93	8	0.00	62	0.24	48	0.39	38	0.11	56	0.93	8	0.93	5	0.98	5
vo_7	1981	66	0.90	11	0.00	66	0.00	66	0.43	38	0.07	62	0.93	9	0.93	7	0.95	8
vo_8	1961	65	0.9	8	0.0	65	0.0	65	0.3	46	0.0	60	0.8	10	0.9	7	0.9	5

video	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
			3		0		0		0		9		9		1		8	
vo_9	1911	64	0.97	5	0.28	48	0.36	43	0.36	41	0.12	57	0.95	6	0.95	4	1.00	3
vo_10	1921	63	0.96	11	0.13	56	0.02	62	0.12	55	0.18	53	0.90	14	0.90	14	0.98	10
vo_11	1881	62	0.89	15	0.53	37	0.08	58	0.55	28	0.24	50	0.93	13	0.82	5	1.00	10
Average	1711.00	54.67	0.84	10.44	0.13	48.43	0.09	49.96	0.39	33.20	0.13	48.48	0.89	9.00	0.79	5.33	0.96	5.89

Table 5: Detection Rate for Videos of Set 5 (occlusions) for DF, CT, TDK, KCF and Meanshift. Nframes: Number of frames. NGOFS: Number of Group of Frames (see main text for details). DR: Detection Rate. FP: Number of False Positives.

video	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
V4	3344.00	92	0.96	5	0.35	80	0.35	80	0.49	47	0.09	112	0.98	3	0.75	2	0.99	2
V5	1737.00	38	0.87	5	0.36	41	0.20	51	0.86	5	0.07	59	0.89	4	0.92	2	0.97	1
V6	5004.00	64	0.66	23	0.04	177	0.04	177	0.06	60	0.07	172	0.77	16	0.63	11	0.92	6
V7	3011.00	35	0.72	12	0.01	110	0.00	111	0.40	21	0.08	102	0.81	9	0.56	2	0.91	6
V8	4436.00	118	0.91	16	0.20	115	0.04	132	0.05	118	0.06	130	0.92	15	0.63	12	0.96	10
VD2	1140.00	24	0.86	5	0.36	27	0.20	33	0.20	24	0.06	39	0.86	5	0.14	1	0.95	3
VD3	1900.00	14	0.93	1	0.09	64	0.05	67	0.06	14	0.05	66	0.86	2	0.86	1	1.00	0
Average	2938.86	55.00	0.84	9.57	0.20	87.86	0.13	93.17	0.30	39.69	0.07	97.27	0.87	7.71	0.64	4.43	0.96	4.00

The performance of the different methods on the videos corresponding to Set 6 was measured. Set 6 contains four videos in which the tracked person remains static while a second person walks and stands between the tracked person and the camera. Several occlusions were produced in each video, with increasing durations of 2, 4, 8, 16, and 32 seconds. The occlusions were grouped into five subsets, each one containing several occlusions with a similar duration. Thus, Table 6 shows the averages of all the videos evaluated according to the duration of the occlusion. It can be observed that high occlusion times increase the number of false positives for both HoG1/2 and HoG1/2-I.

The false positives for these two methods increase from 3.25 (2[s]) to 26.75 (32[s]). The proposed DF-Full method does not improve the DR of HoG1/2 or HoG1/2-I, but the number of false positives are reduced by 65%. Also, note that the proposed system outperforms previous trackers by a large margin, both in detection rate and false positives.

Table 6: Detection Rate for Videos of Set 6 (occlusions) for DF, CT, TDK, KCF and Meanshift. Nframes: Number of frames. NGOFS: Number of Group of Frames (see main text for details). DR: Detection Rate. FP: Number of False Positives.

Occlusion Time	Nframes	NGOFS	HoG1/2		CT		TDK		KCF		Meanshift		DF-Simple		DF-Full		HoG-I	
			DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
2[s]	190.50	8.25	1.00	3.25	0.44	6.41	0.44	6.41	0.58	3.44	0.76	5.07	0.96	3.50	0.71	1.50	1.00	3.25
4[s]	303.25	9.75	0.94	5.25	0.64	6.72	0.64	6.72	0.72	2.71	0.78	6.37	0.88	5.50	0.88	1.25	0.94	5.25
8[s]	398.25	11.75	0.75	7.00	0.59	8.55	0.59	8.55	0.75	2.97	0.80	8.12	0.71	7.25	0.71	4.75	0.75	7.00
16[s]	677.25	20.75	1.00	14.25	0.74	16.47	0.82	16.21	0.86	3.01	0.75	16.69	0.97	14.50	0.94	1.00	1.00	14.25
32[s]	1184.00	32.50	1.00	26.75	0.60	28.62	0.72	28.39	0.69	10.08	0.72	28.32	0.96	27.00	0.96	8.75	1.00	26.75

In order to better characterize the capabilities of the proposed long occlusion's detector, a second measure of the quality of the detections is used: the number of true positive and false negative occlusions detected across all of the videos. From the total number of long occlusions in Set 4 and Set 6, 79% were correctly detected, and only a 3% false positive detection rate was obtained. Occlusion detection failures are caused by three possible situations: the occluded person is not detected before the occlusion happens, an incorrect person detection occurs, or the color and LBP histograms of the occluder are too similar to the one from the occluded person. Also, persons too close to the camera can cause the person detector to fail before the occlusion. These results indicate that the long occlusion detector is safe to use, and that it is able to reject wrong detections that could cause a person-following procedure to fail.

The presence of multiple persons, occlusions and the lack of texture in clothes and/or objects in the video sequences are factors that affect negatively the system's performance. The impact generated by these three factors is analyzed using data from Tables 3-6. Here three cases are analyzed: (i) When videos include multiple persons, the detection rate is not affected but false positives diminish about 50% respect to videos without multiple persons. (ii) When videos include occlusions, the detection rate is not affected, but false positives are incremented about 50%. (iii) When videos include lack of texture in clothes and/or objects, the detection rate diminishes 17%, and false positives increment in about 40%. From these results, it is concluded that, while

presence of multiple objects does not diminish the system's performance, occlusions and the lack of texture in object and/or clothes cause the system to diminish its performance. Note that, even under these challenging conditions, the increment of false positives does not surpass 50%, therefore the system is able to work successfully in that cases. The data association procedure is trained for discriminating between two different persons. Then, the system is better at comparing two persons that at comparing a person with a random crop. This fact causes the videos with multiple persons to behave better than videos with only one person. Then, training of data association considering random background crops is a possible way to improve the performance of the system.

Raw results from tables in this section can be hard to interpret. Then, they are summarized in Table 7. It can be concluded that the proposed method DF-Full is the best tracker available with respect to both detection rate and false positives, in all of the video sets. Note that 37 video sequences were used in total for evaluating the methods.

Table 7: Average results from the proposed tracker DF-Full, CT, TDK, KCF and Meanshift over video sets 3, 4, 5 and 6. DR: Detection Rate. FP: Number of False Positives. In total, 37 videos are considered. The proposed method DF-Full is the best respect to both DR and FP in all of the video sets.

Video Set	DF-Full		CT		TDK		KCF		Meanshift	
	DR	FP	DR	FP	DR	FP	DR	FP	DR	FP
Set 3 (no occlusions)	0.83	2.24	0.40	41.11	0.30	49.54	0.77	14.47	0.15	58
Set 4 (occlusions)	0.79	5.33	0.13	48.43	0.09	49.96	0.39	33.20	0.13	48.48
Set 5 (occlusions)	0.64	4.43	0.20	87.86	0.13	93.17	0.30	39.69	0.07	97.27
Set 6 (increasing time)	0.84	3.45	0.60	13.35	0.64	13.25	0.72	4.44	0.76	12.91

Finally, two relevant videos showing failures (video sequences v10 and v22 from video set 1) are analyzed. Although comparative advantages from our method (long-term tracking ability and strong data association) have been described, failure analysis can be useful for understanding the weaknesses of our method, but also for showing its ability to recover from failures. In the two analyzed videos from our dataset, the ground truth is displayed in green, DPM-HOG-object detection is displayed in red, and the proposed delay-free tracker estimation is displayed in blue, as shown in Figure 6. Detection rate in video sequence v10 is low (78% DF-Full / 85% DF-Simple) even when it does not contain any kind of occlusion. This is caused because of the person becoming too near to the camera. Then, the person detector fails and generates several consecutive false positives, which cause the tracking system to fail. However, the tracker recovers when the person is far enough from the camera again. Video sequence v22 contains abrupt illumination changes, which cause the long occlusion detector to fail and loss the target by about 18 seconds. However the system recovers near the end of the video. Then, the system can fail when the object to be tracked is too near to the camera (which causes lack of detections), or when strong illumination changes occur (detections considered outliers/occlusors). Note that the system is able to recover in these two trials.

The video sequences showing failure cases are available at: <http://www.amtc.cl/DFTracker/FailModes/>

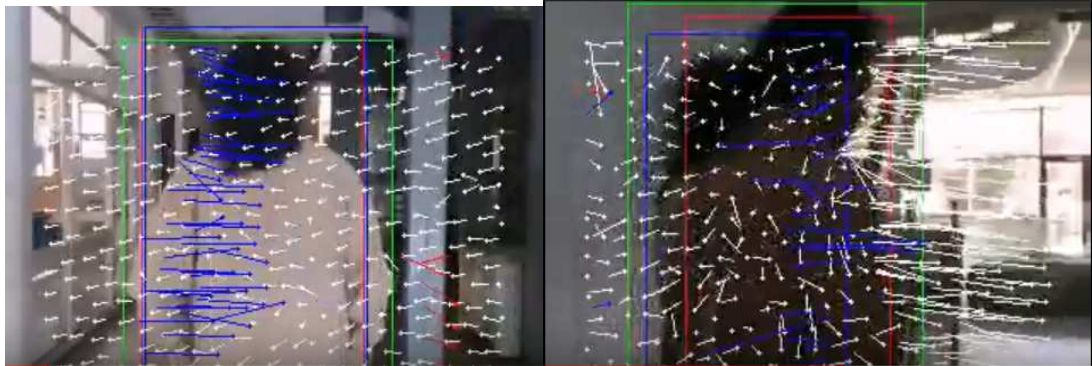


Figure 6: Tracking failure caused by strong illumination changes in video v22. The system is able to recover from this error. Ground truth displayed on green, DPM-HOG object detection on red and delay-free estimation in green.

4.4. Application in a Service Robot

A “person-following” behavior based on the here-proposed object/person tracking approach has been developed for Bender, a general-purpose social robot. Bender is an open and flexible research platform that provides human-like communication capabilities, as well as empathy. Bender has an anthropomorphic upper body (head, arms, chest), and a differential-drive platform that provides mobility. The robot uses three computers: a tablet for providing an user interface, and two Alienware computers, one dedicated to computer vision and the other dedicated to planning and navigation. The electronic and mechanical hardware components of the robot, as well as its software architecture, are described in [28].

The “person-following” behavior has been tested in indoor and outdoor daily life environments, such as the ones shown in Figure 7. In these environments, Bender is able to follow persons robustly, being able to manage occlusions produced by humans as well as changes in illumination. This behavior has characteristics that make it different from typical person tracking in the image domain: First, the tracked person is near the robot, and then it covers a wide area in the image. Second, only one person needs to be tracked for the behavior to work. Third, in the person following task it is critical to generate control orders that are able to keep the followed person inside the field of view of the camera, even when the person turns suddenly. The person-following task starts with the person standing in front of the robot, and then there is not ambiguity in which person to follow. The full system is able to run at 23 fps using only one core on an Intel Core i5 processor. Also, the delay-free procedure based on RANSAC requires only 0.07 milliseconds for processing a detection. Then, the proposed approach, based on the

delay-free transform, is a useful tool when applied to person following or other active vision tasks. A video showing person following trials is available at: <http://www.amtc.cl/DFTracker/benderfollow.mp4>



Figure 7: Images of the robot (Bender) following a person. Left/Right: Outdoor/Indoor setup.

4.5. Other Applications

The proposed method was tested in two additional applications: face tracking, and car tracking. These applications were selected because the required base detectors are available as open source: MTCNN [47] for face detection and YOLOv2 [46] for car detection. Two face video sequences and one car video sequence were selected from the dataset used in KCF [48] for comparison purposes. It must be noted that the videos have a resolution of 320x240, i.e., that resolution is enough for solving tracking tasks. KCF was selected for comparison as it is the best alternative tracker in Tables 3, 4, 5 and 6. Results from these tests are shown in Table 8.

From Table 8, it can be noted that DF-Full is able to track successfully faces and cars from the dataset of KCF [48]. Also, DF-Full performs better in average than KCF on the tested videos. The detection rate of KCF on the “David” dataset is small because of the original ground truth is mislabeled, and then it had to be labeled again. The videos are available for download at <http://www.amtc.cl/DFTracker/OtherObjects/>

Table 8: Detection Rate and False Positives for Face and Car Tracking Using the Proposed System (DF-Full) v/s Detection Rate Using KCF. Nframes: Number of frames.

Videos	NFrames	DF-Full		KCF	
		DR	FP	DR	FP
FaceOccl	870	0.99	9	1.00	0
Car5	656	0.96	26	0.34	433

Videos	NFrames	DF-Full		KCF	
		DR	FP	DR	FP
David	750	0.54	345	0.22	585
Average	758.6	0.83	127	0.52	339

5. Conclusions

In this work, a visual tracking system based on a delay-free transformation of object detections is proposed. A delay-free transformation is estimated by detecting coherent sets of feature tracks contained in the detection bounding box by using RANSAC. The transformation is used for projecting a delayed detection bounding-box across time, enabling the generation of delay-free detections. The system is also able to detect and reject occlusions and false detections. Occlusions are detected by using a novel HMM model that considers both changes in the image patches and the number of feature tracks entering/leaving the detection bounding boxes, and thus it is able to work without the need of modeling or detecting the occluding object. False detections are rejected by extracting features from two consecutive detection image patches and by using an SVM. In this way, the system is able to generate up-to-date and error-robust detections.

The proposed system was tested in the person-following task, using a part-based person torso detector (HOG1/2) and the KLT algorithm for generating feature tracks. The HOG1/2 torso detector is used for enabling detection of persons close to the camera. The full method (including delay-free transformation and detection of false detections and occlusions) was compared with a part-based HOG person detection and with state-of-the-art image tracking algorithms.

The results indicate that DF-Simple (including only the delay-free transform) improves the results, and in the worst case (static person with occlusions) it produces only a small degradation in the results, so it is a safe-to-use algorithm. The DF-Full system is able to detect occlusions without the need for tracking or modeling the occluding object. It reduces the number of false positives by 73% in videos without occlusions, and it is also capable of detecting occlusions in 79% of the cases, but the detection rejection procedures cause the detection ratio to decrease. Thus, a tradeoff between detection ratio and false positives exists. Classical object tracking algorithms (CT, TDK, Meanshift, KCF) fail when the tracked object is occluded. Also, all trackers tested (CT, TDK, Meanshift, KCF) are unable to track the target in the long term. Sharp images improve the results of DF-Simple and DF-Full as more accurate feature tracks can be computed. Abrupt illumination changes (from shadow to sunlight or vice versa) can cause the feature tracking system to fail (videos v10, v30 and v34), since it is based on local optical flow, and data association of detections are dependent on color histograms. Additionally, the proposed method was successfully tested on two more

tracking tasks: car tracking, and face tracking, which enables to conclude that the tracker is object-agnostic, but requires an appropriate object detector as base to work. The system can also be used for initializing other object tracking algorithms with a delay-free object detection hypothesis. The full system is able to run at 23 fps using only one core in an Intel Core i5 processor.

Future work is needed to recognize and to track several people by creating a distinctive model for each person, and for adapting other multi-tracking systems for working with delayed and low frame rate detectors, by using the delay-free transform. Also, the use of multi-hypothesis systems for integrating people detections (particle filters) is a possible goal. Finally, a methodology for estimating in which frames the feature tracks are trustworthy (e.g. by analyzing image sharpness and detecting abrupt illumination changes) could reduce the loss in detection rate while preserving the capacity of rejecting occlusions and incorrect detections.

Acknowledgments

This work was partially funded by FONDECYT Projects 1130153 and 1161500.

References

1. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1-8 (2008).
2. Tomasi, C., Kanade, T.: *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon Univ. Pittsburgh (1991).
3. Fischler, M.A., Bolles, R.C., *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. *Communications of the ACM*, vol. 24, no. 6, pp. 381-395 (1981).
4. Zhang, K., Zhang, L., Yang, M.H. *Real-time compressive tracking*. *Computer Vision–ECCV 2012*, pp. 864-877 (2012).
5. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: *Exploiting the circulant structure of tracking-by-detection with kernels*. *Computer Vision–ECCV 2012*, pp. 702-715 (2012).
6. Comaniciu, D., Ramesh, V., Meer, P.: *Real-time tracking of non-rigid objects using mean shift*. *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, pp. 142-149, 2000.

7. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: An Efficient Alternative to SIFT or SURF. Proceedings of the 2011 International Conference on Computer Vision, pp. 2564-2571, (2011).
8. Alahi, A., Ortiz, O. , Vandergheynst, P.: Freak: Fast retina keypoint. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pp. 510-517 (2012).
9. Ballard, D.H.: Generalizing the Hough transform to detect arbitrary shapes. Pattern recognition, vol. 13, no. 2, pp. 111-122 (1981).
10. Ruiz-Del-Solar, J., Loncomilla, P.: Robot head pose detection and gaze direction determination using local invariant features. Advanced Robotics, vol. 23, no. 3, pp. 305-328, (2009).
11. Loncomilla, P., Ruiz-del-Solar, J., Martínez, L.: Object Recognition using Local Invariant Features for Robotic Applications: A Survey. Pattern Recognition, Vol. 60, Dec. 2016, pp. 499-514. (2016)
12. Tuytelaars, T., Mikolajczyk, K.: Local invariant feature detectors: a survey. Foundations and Trends® in Computer Graphics and Vision, vol. 3, no. 3, pp. 177-280 (2008).
13. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 27, no. 10, pp. 1615-1630 (2005).
14. Guo, Z., Zhang, D.: A completed modeling of local binary pattern operator for texture classification. Image Processing, IEEE Transactions on, vol. 19, no. 6, pp. 1657-1663 (2010).
15. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1, pp. 886-893 (2005).
16. Chu, C.T., Hwang, J.N., Pai, H.I. and Lan, K.M.: Tracking human under occlusion based on adaptive multiple kernels with projected gradients. Multimedia, IEEE Transactions on, vol. 15, no. 7, pp. 1602--1615 (2013).
17. Zhou, X., Li, Y., He, B.: Tracking Humans in Mutual Occlusion based on Game Theory. (2013).
18. Jeong, J.M., Yoon, T.S., Park, J.B.: Kalman filter based multiple objects detection-tracking algorithm robust to occlusion. SICE Annual Conference (SICE), 2014 Proceedings of the, pp. 941-946 (2014).

19. Rahmatian, S., Safabakhsh, R.: Online multiple people tracking-by-detection in crowded scenes. *Telecommunications (IST), 2014 7th International Symposium on*, pp. 337-342 (2014).
20. Suresh, S., Chitra, K., Deepack, P.: Patch based frame work for occlusion detection in multi human tracking. *Circuits, Power and Computing Technologies (ICCPCT)*, pp. 1194-1196, 2013.
21. Li, Z., Tang, Q.L., Sang, N.: Improved mean shift algorithm for occlusion pedestrian tracking. *Electronics Letters*, vol. 44, no. 10, pp. 622--623 (2008).
22. Yan, J., Ling, Q., Zhang, Y., Li, F., Zhao, F.: A novel occlusion-adaptive multi-object tracking method for road surveillance applications. *Control Conference (CCC), 2013 32nd Chinese*, pp. 3547--3551 (2013).
23. Tang, S., Andriluka, M., Milan, A., Schindler, K., Roth, S., Schiele, B.: Learning people detectors for tracking in crowded scenes. *Computer Vision (ICCV), 2013 IEEE International Conference on*, pp. 1049-1056 (2013).
24. Tang, S., Andriluka, M., and Schiele, B.: Detection and tracking of occluded people. *International Journal of Computer Vision*, vol. 110, no. 1, pp. 58-69 (2014).
25. Guan, Y., Chen, X., Yang, D., and Wu, Y.: Multi-person tracking-by-detection with local particle filtering and global occlusion handling. *Multimedia and Expo (ICME), 2014 IEEE International Conference on*, pp. 1-6 (2014).
26. Viterbi, A.J.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, vol. 13, no. 2, pp. 260-269, 1967.
27. Pairo, W., Ruiz-del-Solar, J., Verschae, R., Correa, M., and Loncomilla, P.: Person following by mobile robots: analysis of visual and range tracking methods and technologies. *RoboCup 2013: Robot World Cup XVII*, pp. 231-243 (2014).
28. Ruiz-del-Solar, J., Correa, M., Verschae, R., Bernuy, F., Loncomilla, P., Mascaró, M., Riquelme, R., Smith, F.: Bender – A General-Purpose Social Robot with Human-Robot Interaction Abilities. *Journal of Human – Robot Interaction*, vol. 1, no. 2, pp. 54-75 (2012).
29. Uijlings, R., van de Sande, A., Gevers, T., Smeulders, M.: Selective search for object recognition. *International journal of computer vision* 104.2 (2013): 154.
30. Zitnick, C. L., and Dollár P.: Edge Boxes: Locating Object Proposals from Edges. *ECCV (5)*. (2014).

31. Felzenszwalb, P., Girshick, R., McAllester, D., and Ramanan, D.: Visual object detection with deformable part models. *Communications of the ACM* 56.9 (2013): 97-105.
32. Dollár, P., Belongie, S. J., and Perona, P.: The Fastest Pedestrian Detector in the West. *BMVC*. Vol. 2. No. 3. (2010).
33. Dollár, P., Appel, R., Belongie, S., and Perona, P.: Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.8 (2014): 1532-1545.
34. Kalal, Z., Mikolajczyk, K., and Matas, J.: Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.7 (2012): 1409-1422.
35. Pernici, F., and Del Bimbo, A.: Object tracking by oversampling local features. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.12 (2014): 2538-2551.
36. Krizhevsky, A., Sutskever, I., Hinton G.E.: Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 1097-1105 (2012)
37. Girshick, R.: Fast R-CNN. arXiv:1504.08083 [cs.CV] (2015)
38. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems* 28 (NIPS 2015).
39. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778
41. Nebehay, G., Pflugfelder, R.: Clustering of Static-Adaptive Correspondences for Deformable Object Tracking. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2784-2791 (2015)
42. Chen, L., Zhou, F., Shen, Y., Tian, X., Ling, H., & Chen, Y. Illumination insensitive efficient second-order minimization for planar object tracking. *ICRA* (2017)
43. Tan, D. J., & Ilic, S. Multi-forest tracker: A chameleon in tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1202-1209) (2014).

44. Tan, D. J., Tombari, F., Ilic, S., & Navab, N.. A versatile learning-based 3d temporal tracker: Scalable, robust, online. In Proceedings of the IEEE International Conference on Computer Vision (pp. 693-701) (2015).
45. Tan, D. J., Navab, N., & Tombari, F.. Looking Beyond the Simple Scenarios: Combining Learners and Optimizers in 3D Temporal Tracking. IEEE Transactions on Visualization and Computer Graphics (2017).
46. Redmon, J., Farhadi, A. YOLO9000: Better, Faster, Stronger. arXiv preprint arXiv:1612.08242, 2016
47. Zhang, K., Zhang, Z., Li, Z., Qiao, Y. Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. IEEE Signal Processing Letters, volume 23, number 10, pp. 1499-1503, year 2016.
48. Henriques, J.F., Caseiro, R., Martins, P., Batista, J. High-Speed Tracking with Kernelized Correlation Filters, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, pp. 583-596, year 2015.
49. Hare, S., Golodetz, S., Saffari, A., Vineet, V., Cheng, M.M., Hicks, S.L. and Torr, P.H.S. Struck: Structured Output Tracking with Kernels. IEEE Trans. Pattern Anal. Mach. Intell. vol 38, number 110, October 2016