PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

# MEJORAMIENTO DE ALGORITMOS DE SEGUIMIENTO UTILIZANDO MODELOS DE SALIENCIA

**CRISTÓBAL ALEJANDRO UNDURRAGA RIUS**

Tesis para optar al grado de
Magister en Ciencias de la Ingeniería

Profesor Supervisor:
DOMINGO MERY

Santiago de Chile, Octubre 2011

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

# MEJORAMIENTO DE ALGORITMOS DE SEGUIMIENTO UTILIZANDO MODELOS DE SALIENCIA

**CRISTÓBAL ALEJANDRO UNDURRAGA RIUS**

Miembros del Comité:
DOMINGO MERY
MIGUEL TORRES
PABLO ZEGERS
JORGE RAMOS

Tesis para optar al grado de
Magister en Ciencias de la Ingeniería

Santiago de Chile, Octubre 2011

*A mi familia y amigos, quienes me han acompañado durante mi vida.*

## AGRADECIMIENTOS

Me gustaría agradecer a todos los que me han apoyado en estos años de investigación y aprendizaje. Primero quisiera agradecer el apoyo de Pedro Cortez con quien compartí gran parte de mi carrera universitaria y durante el desarrollo de nuestras respectivas tesis. Además me gustaría dar las gracias a Hans-Albert Löbel y a Christian Pieringer, por su ayuda en la captura de los vídeos presentados en esta tesis, además de todos sus consejos y conocimientos que sirvieron para definir mi proyecto. Por último me gustaría agradecer a mi profesor asesor, Domingo Mery por su constante guía, por enseñarme que para defender una idea hay que poder respaldarla con resultados y por confiar en mis ideas.

*Ninguna investigación humana puede llamarse científica a menos que siga su curso a través de la exposición y demostración matemática.*

—LEONARDO DA VINCI, Tratado sobre Pintura

*Lo maravilloso de aprender algo es que nadie puede arrebatárnoslo.*

—B.B.KING

*Para crear lo fantástico, primero debemos entender lo real.*

—WALT DISNEY

## INDICE GENERAL

## INDICE DE FIGURAS

## INDICE DE TABLAS

## RESUMEN

Uno de los grandes desafíos de la visión por computador es mejorar los sistemas automáticos para la detección y seguimiento de objetos o regiones en un conjunto de imágenes. Un enfoque que ha cobrado importancia recientemente se basa en la extracción de descriptores, tales como el descriptor de covarianza, ya que logran permanecer invariantes en las regiones de estas imágenes a pesar de los cambios de posición, traslación, rotación y escala. Utilizando el mismo descriptor de covarianza proponemos, en este trabajo, un novedoso algoritmo de saliencia, el cual detecta las zonas más importantes de una imagen y es capaz de determinar en una imagen aquella(s) región(es) más relevantes que pueden ser utilizadas tanto en el reconocimiento como en el seguimiento de objetos. Nuestro método se basa en la cantidad de información (la magnitud de variación de distintas características) de cada pixel en una imagen y nos permite adaptar las regiones para maximizar la diferencia de información con su entorno. Esto nos permite incrementar la precisión de los algoritmos de seguimiento hasta en un 27%, sin comprometer demasiado el recall de éste, y aumentar hasta un 92% de precisión si solo nos enfocamos en aumentar ésta. Con estas mejoras a la precisión de los algoritmos de seguimiento evitamos que éstos se confundan con el fondo al momento de seleccionar una región que incluya una persona.



**Palabras Claves:** Visión por Computador, Sistemas de Seguimiento, Mapas de Saliencia, Descriptor de Covarianza.

## ABSTRACT

One of the challenges of computer vision is to improve the automatic systems for the recognition and tracking of objects in a set of images. One approach that has recently gained importance is based on extracting descriptors, such as the covariance descriptor, because they manage to remain invariant in the regions of these images despite changes of position, translation, rotation and scale. In this work we propose, using the Covariance Descriptor, a novel saliency system, which detect the regions that are more important in an image, and is able to find the most relevant regions in an image, which can be used for recognition and tracking objects. Our method is based on the amount of information (magnitude of the variance of diferent characteristics) from each pixel in the image, and allows us to adapt the regions to maximize the difference of information between the region and its environment. This allow us to improve the tracker's precision up to a 27%, with out compromising to much the recall, and increasing to 92% the precision if we focus only in improving it. With these improvements to the precision of tracking algorithms, we can prevent that they get confused with the background at the moment of selecting an initial region that includes a person.



**Keywords:** Computer Vision, Tracking Systems, Saliency Maps, Covariance Descriptor.

# 1. INTRODUCCION

Existen muchas discusiones sobre si “una imagen vale mil palabras”. Confiamos en lo que vemos, de hecho, anteponemos la vista a cualquier otro sentido. Si vemos algo y nos da cierta impresión, es muy difícil que los otros sentidos la puedan contradecir. Cámaras con mejor resolución, pantallas de diferentes tamaños, nuevos métodos de reproducción de vídeos, son cosas que vemos a diario porque en cierto sentido nos fascina lo visual. Pero no nos quedamos ahí, incluso los navegadores de internet son cada vez más interactivos, incorporando mejor calidad de fotos, vídeos, utilizando elementos 3D, etc., todo esto con el fin de llamar la atención visual del usuario.

Es en esta fascinación por lo visual que muchas aplicaciones que se efectúan manualmente por un operador se han visto de a poco reemplazadas por sistemas computacionales. Estos aprovechan la objetividad de un computador para analizar vídeos y ejecutar acciones predeterminadas. Es esta actividad que se denomina visión artificial o visión por computador, la cual tiene como propósito programar sistemas computacionales para que entiendan y analicen imágenes o vídeos. Aunque estamos lejos de igualar la capacidad de reconocimiento de objetos y de rostros del ser humano, seguimos construyendo sistemas que tratan de imitarlo.

Hoy en día, los algoritmos de análisis de vídeo (detección, reconocimiento y seguimiento) han sido centro de atención por sus grandes beneficios en sistemas de seguridad, de análisis de tráfico, control de calidad de alimentos, entre muchas otras aplicaciones (ver Figura 1.1). Algunos ejemplos son: en sistemas de seguridad, se han hecho aplicaciones que detectan fuego para extinguirlo de manera automática (Yuan, 2010) o algoritmos que detectan objetos olvidados para detectar posibles bombas en lugares públicos (Bhargava, Chen, Ryoo, & Aggarwal, 2009); en sistemas de análisis de tráfico, se utiliza la visión artificial para asistir a los conductores y así prevenir accidentes, por ejemplo

detectando las señaleticas y advirtiendo al conductor (Ruta, Porikli, Watanabe, & Li, 2009) o detectando vehículos en la calle (Fossati, Schönmann, & Fua, 2010); en control de calidad, vemos aplicaciones para el conteo rápido de productos, tales como medicamentos en pastillas (Možina, Tomaževič, Pernuš, & Likar, 2009), o la detección de fallas al soldar elementos o estructuras metálicas, como las llantas (Carrasco & Mery, 2010). Inclusive, hoy en día las consolas para videojuegos constan de un sistema de cámaras, con las cuales analizan nuestros movimientos como forma de controlar el juego (Shotton et al., 2011). Pero estos algoritmos siguen siendo muy precarios, o muy específicos a problemas dados.

(a) Asistencia de distancia para conductores.


(b) Detección de objetos olvidados en un metro.

(c) Segmentación de una pastilla para el conteo rápido.

FIGURA 1.1. Ejemplos de sistemas basados en vision por computadora e inteligencia artificial.

## 1.1. Motivación

El ser humano para reconocer un objeto trata de utilizar la característica que da mayor información sobre el mismo y que abarca mayor área. Por ejemplo, para detectar a una persona en una multitud, tenderíamos a utilizar como característica el color de la ropa, ya que es aquella la que más área abarca. Pero si ésta no da la información necesaria para diferenciar, pasaríamos a la siguiente que más información nos entregara, y así sucesivamente. Para reconocer objetos en una imagen existen diferentes enfoques para

definirlos. Por ejemplo, a través de: puntos de interés (descriptores con información relevante) (Harris & Stephens, 1988) y (Lowe, 1999); bolsa de palabras (zonas que definen al objeto) (Nowak, Jurie, & Triggs, 2006); características de una región (variación de las características de la imagen) (Tuzel, Porikli, & Meer, 2006); aspecto local (zonas de saliencia)(Jugessur & Dudek, 2000); entre otros.

Por otra parte, la visión por computadora ha buscado aprovechar las cámaras de seguridad en diferentes aplicaciones, como por ejemplo: en la lucha contra los robos, haciendo un análisis de comportamiento de los clientes; en la distribución de productos, a través de un análisis del tráfico de clientes; en marketing personalizado, haciendo un reconocimiento de los clientes y analizando sus compras; en conteo de personas, a través del reconocimiento de personas; entre muchas otras. Es por esto que uno de los algoritmos que más interesa es el de seguimiento de personas, el cual detecta personas que entren en el cuadro de la cámara y luego seguirlas a través de la secuencia de vídeo. Este interés que surge por este algoritmo es debido a la variedad de aplicaciones en las cuales se puede aplicar, como las previamente descritas. En todos los años de investigación sobre algoritmos de seguimiento, se han producido una gran diversidad de enfoques para resolver este problema, pero aun no se encuentra completamente resuelto (Yilmaz, Javed, & Shah, 2006).

Además, al ver una imagen o escena, el ser humano se concentra en ciertas zonas más que otras. La ley general de la percepción dice que las formas simples y sin excesos de información son más fáciles de percibir, pero en la realidad esto no siempre es cierto y puede llegar incluso a ser que estas formas sean las más fáciles de confundir (Scholl, 2001). Además, una característica de la percepción humana es que procesamos las imágenes selectivamente, concentrándonos en zonas que contienen información más relevante. De aquí nacen otro tipo de algoritmos conocidos como algoritmos de saliencia, los cuales tratan de definir los lugares donde centraríamos la vista en un primer lugar. Estos algoritmos

generan un imagen en blanco y negro, del mismo tamaño que la imagen original, donde un pixel o una zona tiende al negro si no tiene mucha importancia y al blanco si es más importante. Por ejemplo en un pasillo de supermercado lo más saliente sería las góndolas ya que están hechas para llamar nuestra atención (ver Figura 1.2).

(a) Imagen Original (b) Mapa de Saliencia

FIGURA 1.2. Ejemplo de un mapa de Saliencia en un supermercado.

## 1.2. Defínción del Problema

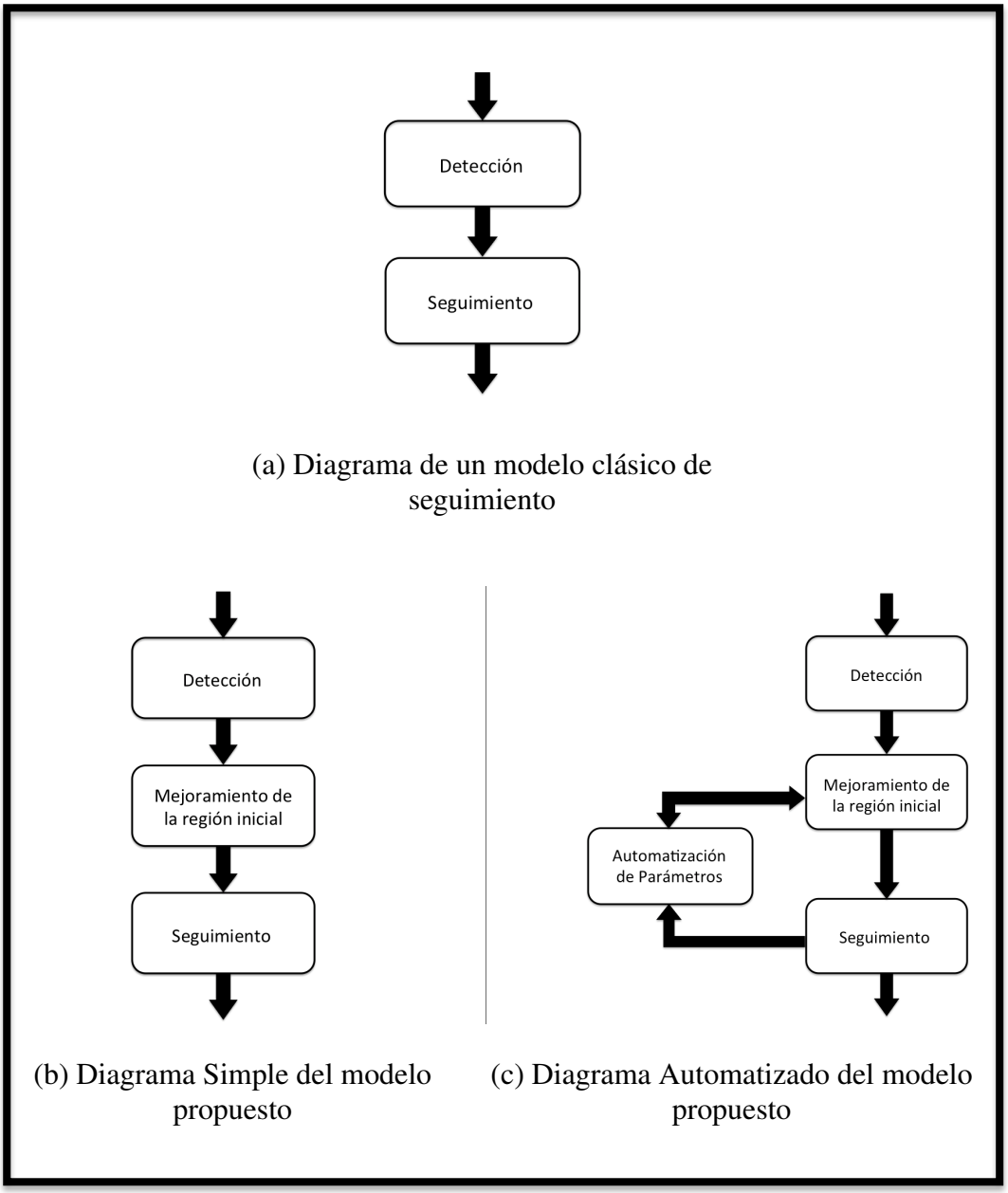
### 1.2.1. Hipótesis

La hipótesis de este trabajo es que a través de algoritmos de saliencia se puede mejorar los algoritmos de seguimiento de personas en lugares con fondo llamativo, como lo son por ejemplo, los pasillos de supermercados.

### 1.2.2. Objetivos

Nuestro objetivo principal es proponer un nuevo modelo de detección y seguimiento con el cual mejorar el desempeño de los algoritmos de seguimiento. En contraste con el modelo clásico (ver 1.3-a), nuestro modelo se basa en utilizar la saliencia para modificar la ventana donde se define el objeto a seguir y de esta forma eliminar el fondo de ésta

(ver Figura 1.3-b) y el cual a través del entrenamiento puede predecir que tan necesario es disminuir la ventana y en cuanto es necesario hacerlo (ver Figura 1.3-c).




(a) Diagrama de un modelo clásico de seguimiento

(b) Diagrama Simple del modelo propuesto

(c) Diagrama Automatizado del modelo propuesto

FIGURA 1.3. Diagrama del modelo estandar para seguimiento de objetos y del modelo propuesto para seguimiento de objetos.

Por otra parte proponemos un nuevo modelo de saliencia, el cual se basa en el descriptor de covarianza (Tuzel et al., 2006), el cual provee una mejor herramienta que otros algoritmos de saliencia, en especial para el algoritmo de seguimiento On-line Naive Bayes Nereast Neighbor (Cortez, Mery, & Sucar, 2010), ya que ambos se basan en el mismo descriptor.

## 1.3. Métodos Existentes

### 1.3.1. Métodos de Seguimiento

Un subconjunto de los algoritmos de seguimiento son aquellos que se basan en un modelo de apariencia, los cuales aprenden y mantienen un modelo de la apariencia del objetivo, con el fin de poder detectarlo a medida que transcurre el tiempo. Estos algoritmos primero necesitan definir el objetivo a seguir y para ello existen diferentes maneras de representarlo, por ejemplo como: un punto, varios puntos, un rectángulo, una elipse, un esqueleto, el contorno, etc. Pero no existen algoritmos que efectúen una optimización de esta representación inicial para definir un objeto que se desea seguir, al menos no hay ninguno que sea de nuestro conocimiento.

Como los objetos pueden cambiar durante el seguimiento, los modelos de apariencia necesitan una forma de actualizarse, por lo cual se han desarrollado métodos adaptivos de seguimiento. Esto quiere decir que los algoritmos se van adaptando a las variaciones de la apariencia del objeto durante el seguimiento. Estos métodos adaptivos pueden ser separados en dos tipos: *every-frame-update* y *selective-update*.

Los métodos *every-frame-update* son aquellos algoritmos que actualizan su modelo de apariencia en cada cuadro. Por ejemplo, Collins et al. presentan un mecanismo en línea para la selección automática de un subconjunto de características durante el seguimiento. Este método busca seleccionar las características que discriminan mejor entre el objeto y

el fondo de la imagen (Collins, Liu, & Leordeanu, 2005). Siguiendo esta misma línea, Grabner et al. proponen un sistema en línea de selección de características utilizando AdaBoost (Grabner, Grabner, & Bischof, 2006). Una de las mejoras que proponen en su trabajo es la posibilidad de entrenar en línea el clasificador del objeto, lo que permite adaptarse a los cambios de apariencia. Por otro lado, al utilizar características de cómputo rápido, el algoritmo funciona en tiempo real. Más recientemente Babenko et al. utilizando también un sistema de boosting para la clasificación proponen un algoritmo que utiliza Multiple Instances Learning, o MIL, para el aprendizaje del modelo (Babenko, Yang, & Belongie, 2009). El método MIL en vez de utilizar instancias etiquetadas como correctas o incorrectas, utiliza dos bolsas de instancias, una con instancias correctas y la otra con instancias incorrectas, con el fin de inferir el concepto de correcto. Claramente este tipo de algoritmos muestran muy buenos resultados debido a que se adaptan rápidamente a cambios de apariencia en el modelo, pero de esta misma si la clasificación es inicialmente mala entonces tiende más rápidamente a que el seguimiento falle.

Por el otro lado, los métodos *selective-update* son aquellos algoritmos que tienen un criterio para actualizar su modelo, evitando actualizar cuando el seguimiento es poco preciso. Kalal et al. presenta un sistema de seguimiento con un algoritmo de detección del objeto (Kalal, Matas, & Mikolajczyk, 2009). El detector es entrenado en línea utilizando dos procesos para evitar actualizaciones impresisas: el proceso de crecimiento consiste en la selección apropiada de muestras a partir de la trayectoria del objeto siempre y cuando cumplan con algún criterio de similitud, y el proceso de podaje consiste en eliminar las detecciones incorrectas ya que asume que el objeto es único dentro de la imagen. Utilizando un proceso de crecimiento, Cortez et al. proponen un sistema MIL que utiliza la matriz de covarianza como característica del objeto que se actualiza solo si existe en las nuevas detecciones una mayor cercanía al modelo de apariencia, que alguna instancia anterior que se haya guardado (Cortez et al., 2010).

Para ambas categorías es necesario una buena inicialización ya que comenzaran a aprender el modelo de apariencia desde el primer cuadro. Por esto proponemos una metodología para mejorar los sistemas de seguimiento, mejorando la detección inicial y por consecuencia mejorando el modelo inicial de apariencia de éstos.

Por otro lado, en un contexto más biológico, el seguimiento de objetos está fuertemente ligado a tareas atencionales, por ejemplo el movimiento del ojo o la saliencia en una imagen. Esto se debe a que el ser humano no mira una escena de manera estática, los ojos se mueven de manera muy rápida para crear un mapa mental de lo que es importante en la escena para luego poder enfocar y ver más detalladamente la imagen. Esto sería lo que trataría de emular los algoritmos de saliencia con los mapas de saliencia.

### 1.3.2. Métodos de Saliencia

Los algoritmos de saliencia pueden ser separados en dos grupos: bottom-up y top-down. Los bottom-up son aquellos sistemas que no conocen ningún dato a priori de lo que es saliente. Solamente a través de la información que proporciona la imagen, se extraen características para determinar lo que sobresale en ella. Por otro lado, los algoritmos top-down conocen información de antemano sobre lo que quieren que sobresalga, esto quiere decir, que buscan resaltar zonas donde posiblemente se encuentre un objeto dado. Por ejemplo, si tenemos una imagen donde aparece un grupo de personas, un método bottom-up podría resaltar cualquier parte de la imagen, una zona donde exista mucha iluminación o una región con varios colores o con un alto contraste, etc., en cambio un método top-down podría ser entrenado para resaltar rostros, y resaltar las áreas donde podría haber una cara (ver Figura 1.4).

Nosotros proponemos un novedoso método bottom-up de saliencia que utiliza la covarianza de un grupo de características extraídas a nivel de cada pixel de una imagen. Recordemos que la covarianza es la medición estadística de la variación o relación entre

(a) Imagen Original

(b) Mapa de Saliencia Bottom-Up

(c) Mapa de Saliencia Top-Down

FIGURA 1.4. Ejemplo de los Mapas de Saliencias obtenidos con diferentes métodos de saliencia.

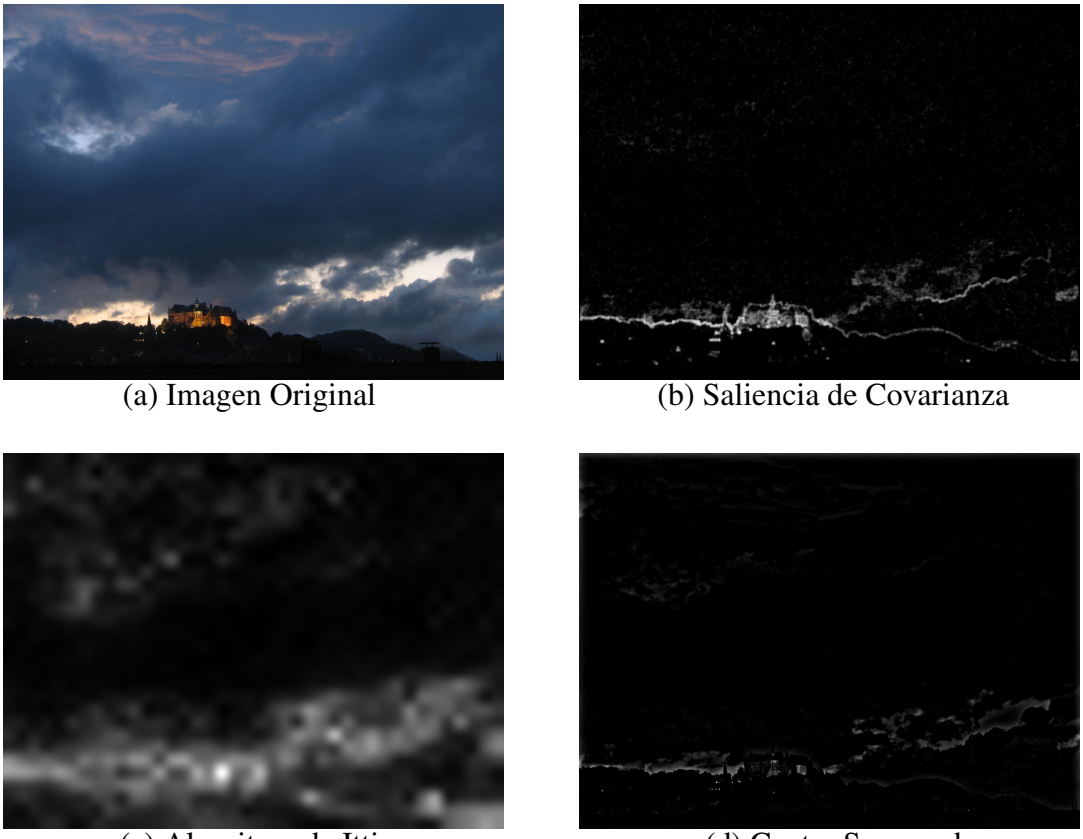
dos variables aleatorias, esta puede ser negativa, cero o positiva, dependiendo de la relación entre ellas. En nuestro caso las variables aleatorias representarán las características. Para evaluar la efectividad de nuestro sistema de saliencia, lo comparamos con el algoritmo de Itti et al. (Itti, Koch, & Niebur, 2002), derivado de una posible arquitectura de la visión humana, y el algoritmo de saliencia Center Surround propuesto por Achanta et al. (Achanta, Estrada, Wils, & Süsstrunk, 2008).

El modelo de saliencia de Itti et al. (2002) es uno de los primeros en su tipo y es uno de los más utilizados para comparar los nuevos algoritmos que han surgido. El modelo está inspirado en la teoría de integración de características, teniendo cuidado en

la construcción para que el modelo sea neurobiológicamente posible. Este se basa en la arquitectura neuronal de los primeros primates, la cual toma una imagen como entrada y la descompone en tres canales: intensidad, color y orientación. Tomando la diferencia de la respuesta de un filtro en diferentes escalas, se crean mapas de características. Estos mapas son normalizados y combinados en escalas y orientaciones obteniendo un mapa por canal. Finalmente los mapas resultantes de cada canal son combinados dando como resultado el mapa de saliencia. Una crítica es que su objetivo es predecir la fijación de la vista humana, lo que puede ser poco útil en alguna otra aplicación, como por ejemplo en segmentar objetos en una imagen.

A diferencia del método anterior que utiliza varios canales para luego combinarlos linealmente, el método de Achanta et al. (2008) utiliza solo un canal. La saliencia es determinada como el contraste local de una región con respecto a su vecindad en diferentes escalas. El contraste se evalúa como la diferencia entre el vector promedio del espacio de color $CIELab$ de un región centrada en un pixel $(x, y)$ y el vector promedio de su vecindad. Otra diferencia es que para efectuar la saliencia en diferentes escalas, en este método no se modifica la imagen sino el tamaño de la vecindad, dando como resultado una mejor resolución en el mapa de saliencia (ver Figura 1.5).

Uno de los últimos algoritmos de saliencia que han aparecido es el algoritmo de saliencia sensible al contexto (Goferman, Zelnik-Manor, & Tal, 2010). Este algoritmo es muy completo, ya que trata de definir la saliencia desde diferentes puntos de vista: saliencia local, saliencia global, contexto inmediato y factores de alto nivel. Es este último punto que lo diferencia de los algoritmos anteriores ya que agrega una regla top-down. La integración de factores de alto nivel quiere decir que se detecta los objetos reconocibles, los rostros y personas en las imágenes. Esto se debe a que para nosotros siempre lo que es reconocible resalta a la vista. Aunque los resultados son excelentes, el tiempo de procesamiento y el



(a) Imagen Original (b) Saliencia de Covarianza

(c) Algoritmo de Itti (d) Center Surround

FIGURA 1.5. Ejemplo de los Mapas de Saliencias obtenidos con los algoritmos utilizados.

consumo de recursos de memoria hacen esta técnica inviable para algoritmos que buscan utilizarlo para mejorar y mantener su funcionamiento en tiempo real.

## 1.4. Desventajas de los enfoques existentes

Hemos visto diferentes enfoques para el seguimiento de objetos los cuales se difieren por la forma en que representan el objeto, por las características que se extraen del objeto o en la forma que la apariencia es modelada. Estas decisiones dependen del contexto en el que se aplica el algoritmo de seguimiento. Es por esto que un gran número de algoritmos han sido desarrollados bajo diferentes escenarios y condiciones de iluminación. Lo más apropiado es experimentar con algunos y determinar cual es el que se acomoda más a las necesidades de nuestra aplicación, o como es nuestra intención tratar de eliminar el

contexto de la forma de representar al objeto y por consecuencia extraer características propias del objeto y no de algo que produce ruido.

Por otro lado, no existen verdaderas desventajas sobre los algoritmos de saliencias. Esto se debe a que para una misma imagen, distintos algoritmos de saliencia pueden destacar diferentes áreas, debido a las características que cada uno utiliza para definir la saliencia. Lo que está ocurriendo últimamente es que los algoritmos se han dejado de comparar con otros por esta misma razón, y están mostrando resultados en otras aplicaciones utilizando algoritmos de saliencia. Por ejemplo, un algoritmo de saliencia puede ser mejor que otro algoritmo de saliencia al ser aplicado en un algoritmo de segmentación, o de recorte de regiones de la imagen, o de cambios de tamaño sin perder información valiosa, etc..

En resumen, no existe un enfoque como el propuesto, o al menos que sea del conocimiento nuestro. Es por esto que no solo nos contentamos con presentar nuevas algoritmos sino que presentamos cualquier ventaja que se produzca gracias al nuevo enfoque.

## 1.5. Contribuciones Originales

En primer lugar, utilizamos nuestro algoritmo de saliencia para mejorar los sistemas de seguimiento en ambientes donde el fondo sea llamativo, como por ejemplo los supermercados. Nuestro algoritmo de saliencia también puede ser utilizado como extractor de puntos de interés aunque su tiempo de ejecución es bastante elevado comparado con algoritmos que ya han sido optimizados como lo ha sido SIFT (Lowe, 1999).

Por otro lado, se efectuó un estudio sobre la importancia de las características en la matriz de covarianza (Cortez, Undurraga, Mery, & Soto, 2009). Debido al aumento en la utilización del descriptor de covarianza en algoritmos de seguimiento, decidimos desarrollar un método de saliencia que utilizará este descriptor como base, ya que de esta

forma estaríamos utilizando elementos previamente calculados y reduciríamos los tiempos de cómputo.

Por último proponemos una novedosa métrica para comparación de algoritmos de seguimientos, la cual responde de una forma más acorde a lo esperado para los casos de borde, donde la precisión o la exactitud es mala.

Un resumen de las contribuciones será publicado en la Conferencia Iberoamericana de Reconocimiento de Patrones 2011, (CIARP 2011).

## 1.6. Organización de la Tesis/Documento

El siguiente trabajo se organiza de la siguiente manera: en el capítulo 2 presentamos las bases matemáticas de nuestro algoritmo y algunos resultados preliminares que dieron paso al algoritmo final; en el capítulo 3 vemos la implementación de los experimentos y los algoritmos utilizados; en el capítulo 4 veremos los resultados obtenidos tanto probando nuestro algoritmo de saliencia, como en la automatización en el cálculo de parámetros; y en el capítulo 5 presentaremos las conclusiones que podemos rescatar y analizaremos los posibles trabajos a futuro.

# 2. MARCO TEORICO

El sistema de seguimiento que proponemos, a diferencia del modelo original que incluye solamente la detección inicial y el seguimiento, se compone de cuatro fases: detección, mejoramiento de la región inicial, seguimiento y automatización.

En esta sección veremos los conceptos matemáticos básicos tanto utilizados como desarrollados o definidos por nosotros. La fase de mejoramiento de la región inicial incluye el cálculo del descriptor de covarianza, el cual es utilizado para obtener el mapa de saliencia del nuevo algoritmo propuesto. Luego reducimos en un porcentaje dado la región inicial de tal manera de eliminar partes que correspondan al fondo de la imagen (ver Figura 2.1).

Por otro lado, la fase de automatización consta del cálculo de una métrica para evaluar si los algoritmos de seguimiento mejoran su desempeño, denominada métrica de efectividad, la cual es utilizada en el entrenamiento del algoritmo para determinar automáticamente el porcentaje óptimo de información.

A continuación repasaremos la definición y el método de cálculo del descriptor de covarianza para luego establecer el nuevo modelo de saliencia propuesto, el algoritmo de reducción de la región inicial, el cálculo de la métrica de efectividad y por último el modelo de red bayesiana para la automatización del parámetro.

## 2.1. Descriptor de Covarianza

El descriptor de covarianza propuesto por Porikli et al. en (Tuzel et al., 2006), se define formalmente como:

$$F(x, y, i) = \phi_i(I, x, y) \tag{2.1}$$

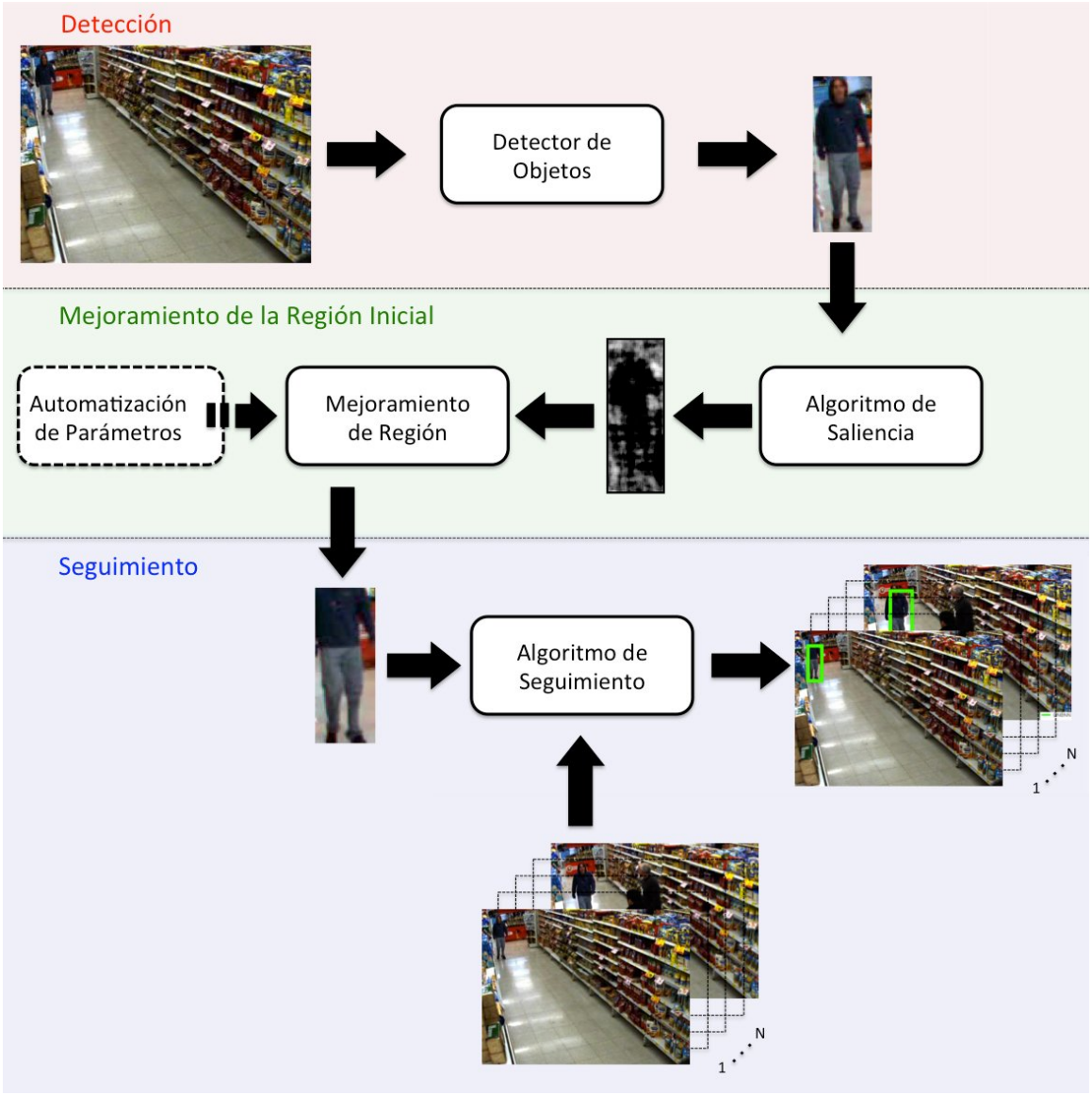



FIGURA 2.1. Diagrama del proceso de mejoramiento de la región definida como persona al inicio de un vídeo (región inicial).

donde $I$ es una imagen (la cual puede estar en RGB, tonos de grises, infrarrojo, etc.), $F$ es una matriz de $W \times H \times d$, donde $W$ es el ancho de la imagen, $H$ el alto de la imagen y $d$ es el número de características utilizadas y $\phi_i$ es la función que relaciona la imagen con la $i$-ésima característica, es decir la función que obtiene la $i$-ésima características a partir de

la imagen $I$, por ejemplo: intensidad de rojo, intensidad de verde, intensidad de azul, etc. Es importante destacar que las características se obtienen a nivel del pixel.

El objetivo es representar el objeto a partir de la matriz de covarianza de la matriz $F$, construida a partir de estas características, en la cual las diagonales representan la varianza de cada característica, mientras que el resto representa la correlación entre las características.

La matriz de covarianza tiene las siguientes ventajas como descriptor: 1) unifica información tanto espacial como estadística del objeto; 2) provee una elegante solución para fusionar distintas características y modalidades; 3) tiene una dimensionalidad muy baja; 4) es capaz de comparar regiones, sin estar restringido a un tamaño de ventana constante o fija, ya que no importa el tamaño de la región, el descriptor es la matriz de covarianza, que es de tamaño constante $d \times d$; 5) la matriz de covarianza puede ser fácilmente calculable, para cualquier región o sub-región.

A pesar de todos los beneficios que trae la representación del descriptor a partir de la matriz de covarianza, el cálculo para cualquier sub-ventana o región dado una imagen, utilizando los métodos convencionales, la hace computacionalmente prohibitiva. Porikli et al. en (Porikli & Tuzel, 2006) proponen un método computacionalmente superior, para calcular la matriz de covarianza de cualquier sub-ventana o región (rectangular) de una imagen a partir de la formulación de la imagen integral. El concepto de la imagen integral fue inicialmente introducida por Viola and Jones et al. en (Viola & Jones, 2001), para el cómputo rápido de características de Haar.

Sea $P$ una matriz de $W \times H \times d$, el tensor de la imagen integral

$$P(x', y', i) = \sum_{x<x', y<y'} F(x, y, i) \quad i = 1 \dots d \tag{2.2}$$

Sea $Q$ una matriz de $W \times H \times d \times d$, el tensor de segundo orden de la imagen integral

$$Q(x', y', i, j) = \sum_{x<x', y<y'} F(x, y, i)F(x, y, j) \tag{2.3}$$

$$i, j = 1 \dots d$$

Ahora, sea

$$P_{x,y} = \left[ \begin{array}{ccc} P(x, y, 1) & \dots & P(x, y, d) \end{array} \right]^T \tag{2.4}$$

$$Q_{x,y} = \left( \begin{array}{ccc} Q(x, y, 1, 1) & \dots & Q(x, y, 1, d) \\ \vdots & \ddots & \vdots \\ Q(x, y, d, 1) & \dots & Q(x, y, d, d) \end{array} \right) \tag{2.5}$$

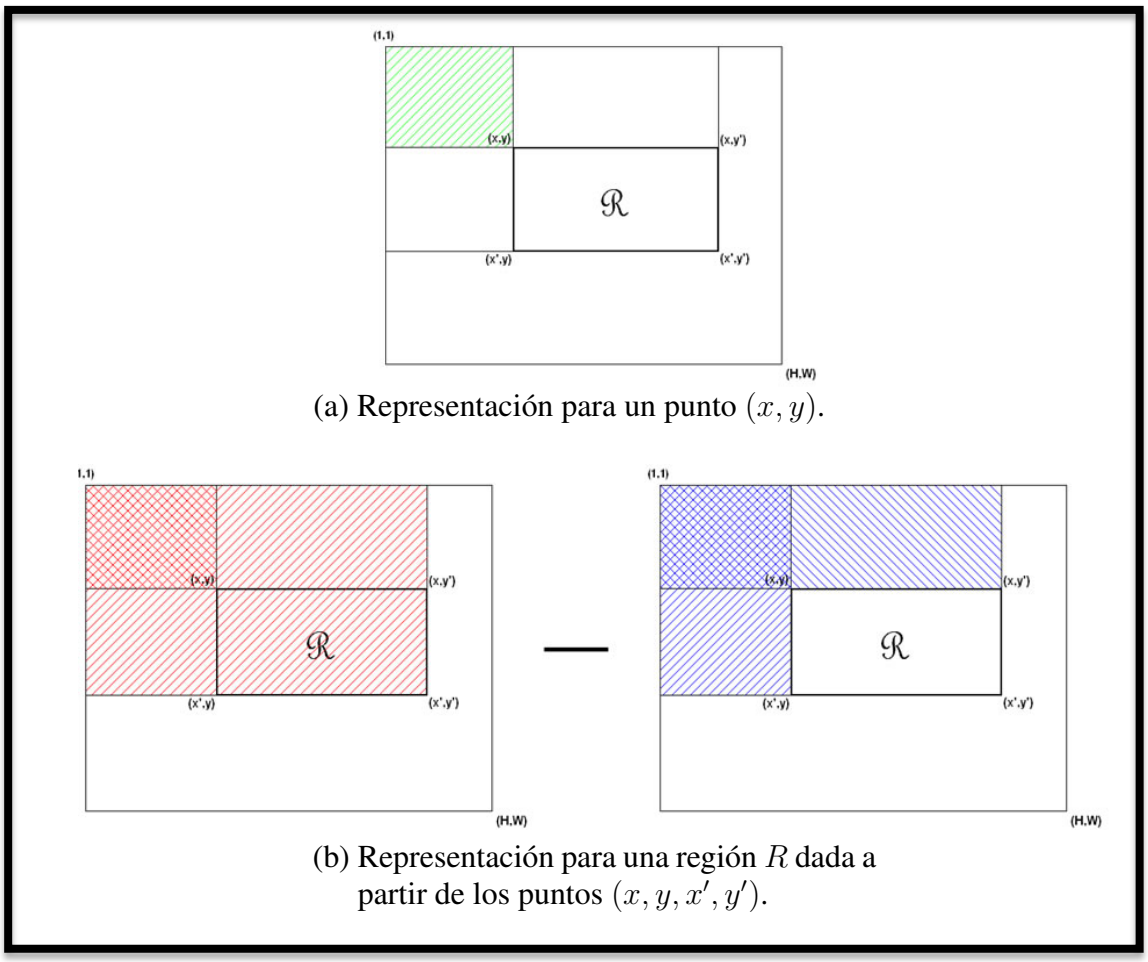
Hay que notar que la matriz $Q_{x,y}$ es simétrica y que para calcular $P$ y $Q$ se necesitan $d + (d^2 + d)/2$ pasos. La complejidad de calcular la imagen integral es de $O(d^2WH)$. Utilizando el método de la imagen integral vemos que la covarianza de cualquier región de la imagen se calcula como:

$$R_Q = Q_{x,y} + Q_{x',y'} - Q_{x',y} - Q_{x,y'} \tag{2.6}$$

$$R_P = P_{x,y} + P_{x',y'} - P_{x',y} - P_{x,y'} \tag{2.7}$$

$$C_{R(x,y;x',y')} = \frac{1}{n-1}[R_Q - \frac{1}{n}R_P R_P^T] \tag{2.8}$$

Donde $n = (x' - x)(y' - y)$. De esta forma, después de construir el tensor de primer orden $P$ y el tensor de segundo orden $Q$, la covarianza de cualquier región se puede computar en $O(d^2)$.




(a) Representación para un punto $(x, y)$.

(b) Representación para una región $R$ dada a partir de los puntos $(x, y, x', y')$.

FIGURA 2.2. Representación gráfica del cálculo de la matriz de covarianza.

## 2.2. Cálculo del Modelo de Saliencia de Covarianza

Una vez calculada la matriz de covarianza para una región dada, buscamos determinar la cantidad de información que contiene un píxel, por eso definimos la región del descriptor

como la vecindad al punto, el cual se obtiene a partir de (2.8). También se necesita una métrica para evaluar la magnitud de la matriz de covarianza. En nuestros experimentos se probó con el mayor valor singular, con la norma infinita, el determinante y el logaritmo del valor absoluto del determinante, siendo este último el que mejor resultados entregó. Por lo tanto, definimos la magnitud de la matriz $C_R$ obtenida como el valor absoluto del determinante de ésta. En teoría de la información es común utilizar el logaritmo para determinar la dispersión de la información. Por ende definimos la cantidad de información $S$ para un punto $(x, y)$ como el logaritmo del determinante de la matriz de covarianza en la vecindad $V$ del pixel:

$$S(x, y) = \log(|\det(C_{R(V)})| + c) \tag{2.9}$$

como deseamos obtener valores positivos definimos la variable $c$ como 1 para obtener solo valores positivos de información.

**Algoritmo 1**: Modelo de Saliencia

**Data**: Una imagen $I$ de tamaño $W$x$L$

**Result**: Mapa de Saliencia $S$

Cálculo de los tensores $P$ y $Q$ de la imagen $I$;

Definición de $V$ como el tamaño de la vecindad;

Definición de $v$ = floor($V$/2);

**for** $i \leftarrow v$ **to** $L$ **do**

  **for** $j \leftarrow v$ **to** $W$ **do**

    $S(i, j) = \log(|\det(C_{R(i-v,j-v;i+v,j+v)})| + 1)$;

  **end**

**end**

## 2.3. Mejoramiento de Regiones Iniciales

Con el mapa de saliencia ya obtenido, buscamos determinar la ventana donde se concentra la mayor cantidad de información. Para esto creamos un algoritmo que reduce el tamaño de la ventana para maximizar la información dentro de ella. Para un rápido cálculo, utilizamos el mismo método que para calcular rápidamente la matriz de covarianza: primero, creamos la imagen integral $I_S$ del mapa de saliencia $S$; luego, calculamos la información dentro de una región como:

$$S(R) = I_S(x, y) + I_S(x', y') - I_S(x', y) - I_S(x, y') \tag{2.10}$$

Definimos una línea como un rectángulo con un lado de un pixel de largo. Luego inicializamos la ventana como toda la imagen y luego comenzamos a reducirla. Nos ponemos un punto de parada: definimos que porcentaje de la información de la imagen queremos que quede dentro de la ventana. Entonces, para cada costado calculamos cuanta información proporciona y el que entregue menos información es reducido, así sucesivamente hasta obtener una región que contenga el porcentaje de información total definido de la imagen (ver Figura 2.3).

## 2.4. Métrica de Efectividad

Para evaluar algoritmos de seguimiento existen dos métricas ampliamente utilizadas: la precisión, que determina el porcentaje de pixeles que estaban dentro del objeto a seguir; y el recall, que determina el porcentaje de los pixeles del total de los pixeles del objeto que fueron detectados. Pero tener dos métricas que son igualmente de importantes es un problema. Es por esto que otra métrica es necesaria, la cual combine las dos métricas, como lo hace $F_{score}$ (2.11). Sin embargo, para un análisis a futuro de la trayectoria preferimos

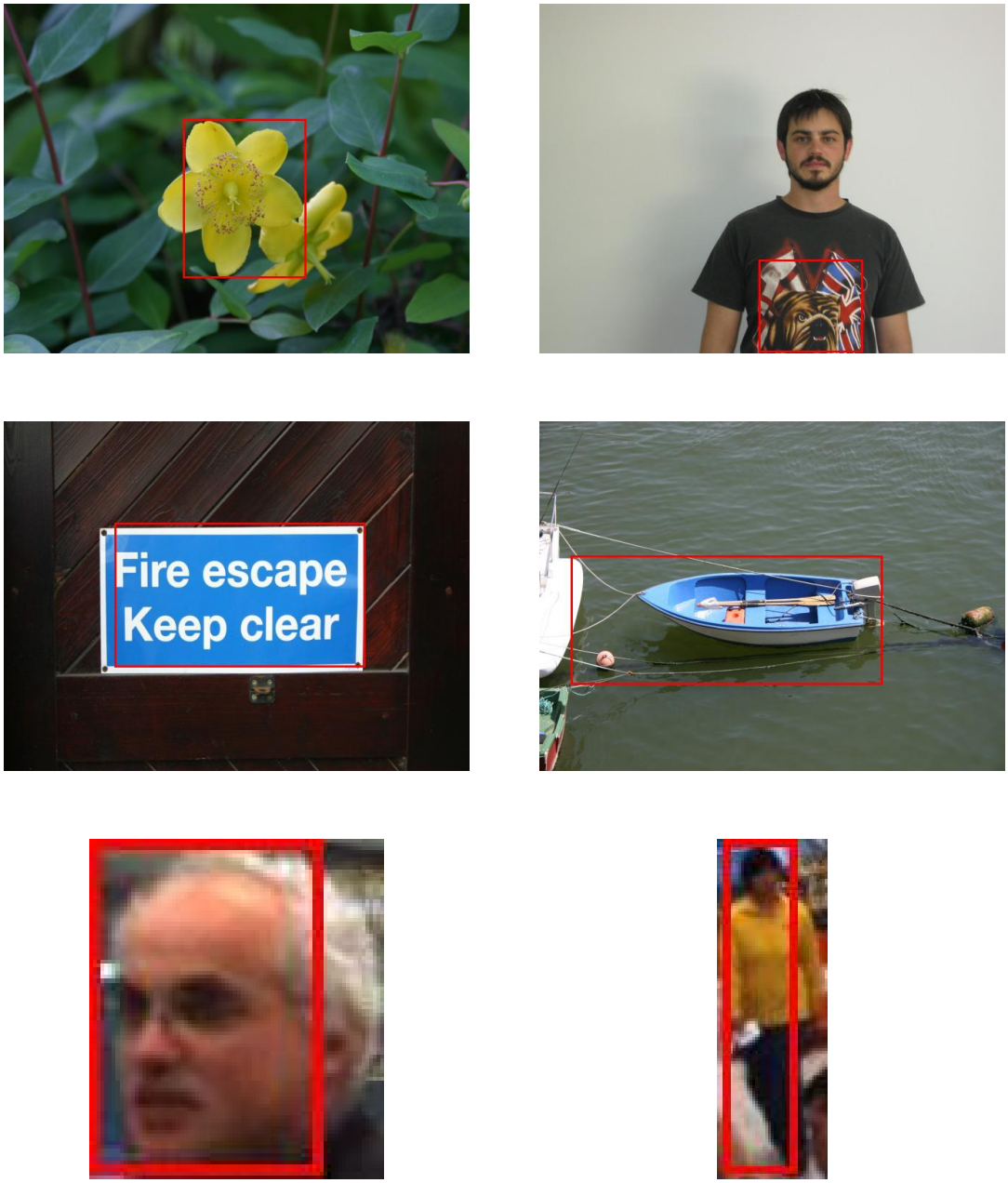



FIGURA 2.3. Ventanas resultantes dejando un 66% de la información total de la imagen dentro de ellas.

darle mayor importancia a la precisión que al recall, es por esto que aconsejamos el uso de $F_{0.5-Score}$ (2.12).

$$F_{Score} = 2\frac{precision * recall}{precision + recall} \tag{2.11}$$

$$F_{0.5-Score} = \left(1 + \left(\frac{1}{2}\right)^2\right) \frac{precision * recall}{\left(\frac{1}{2}\right)^2 * precision + recall} \tag{2.12}$$

Pero en los casos donde una de las dos métricas, la precisión o el recall, es menor a 1 entonces, la métrica $F_{0.5-Score}$ no muestra grandes cambios a pesar de que la otra métrica si lo haga. Puede que estos casos no sean muy recurrentes y por eso el uso de esta métrica es más que aceptada. Pero en nuestro experimentos nos vimos en la necesidad de que en los casos bordes, se notara que había una mejoría si una de las variables cambiaba. Es por esto que basándonos en la media geométrica, proponemos utilizar un nuevo *score*, al cual llamamos métrica de efectividad y se define como:

$$E_{Score} = \sqrt[3]{precision^2 * recall} \tag{2.13}$$

Esta nueva métrica es una poderosa herramienta al momento de elegir el mejor porcentaje de información tomando en cuenta la precisión y el recall. Esta métrica no fue utilizada para compararnos con otros métodos existentes, sino para determinar el mejor porcentaje para un set de imágenes dadas.

## 2.5. Selección automática del porcentaje de información

Hemos descrito un algoritmo que permite elegir una mejor ventana inicial, sin embargo, aun debemos definir el parámetro correspondiente al porcentaje de información

que vamos a dejar dentro de la región inicial. Si queremos que el funcionamiento sea automático, entonces debemos definir un método que establezca el valor del parámetro. Para esta tarea hemos elegido utilizar una red bayesiana debido a que tenemos datos que son porcentuales y pueden ser utilizados como probabilidades. Además para seguir mejorando estos tipos de sistemas es conveniente entender como influyen diferentes aspectos por lo que utilizar algoritmos de caja negra como lo son las redes neuronales no es recomendable (Heckerman, Geiger, & Chickering, 1995). Para nuestra red bayesiana definimos los aspectos que pueden influir en el resultado del seguimiento como: el porcentaje de información ($A_i$); el conjunto de vídeos de entrenamiento (V); el conjunto de vídeos donde un objeto va en el mismo sentido (S); el conjunto de vídeos de entrenamiento dada su semejanza al vídeo de test (C); y el éxito del algoritmo de seguimiento (E). De la distribución conjunta tenemos:

$$\arg\max_i P(E|A_iVSC) \tag{2.14}$$

asumiendo independencia entre las variables, tenemos:

$$\arg\max_i P(E|A_iV)P(E|A_iS)P(E|A_iC) \tag{2.15}$$

utilizando el teorema de probabilidades totales, tenemos:

$$\arg\max_i \alpha \sum_{k=1}^{m} [P(E|A_iV_k)P(V_k)] \sum_{k=1}^{m} [P(E|A_iS_k)P(S_k)] \sum_{k=1}^{m} [P(E|A_iC_k)P(C_k)] \tag{2.16}$$

donde: $V_k$ es un vídeo del set de entrenamiento; $P(V_k)$ es la probabilidad de ocurrencia, uno partido por el numero total de vídeos; $C_k$ es un vídeo del set de entrenamiento; $P(C_k)$ es la probabilidad de que la región de test sea parecida a la región de entrenamiento dada

la similitud entre las matrices de covarianza del objeto de test y el objeto de entrenamiento en el vídeo $C_k$.

Para determinar la similitud entre dos matrices de covarianza utilizamos la métrica, para matrices semi definidas positivas ($SPD^+$), Log-Euclidiana (Ayache, Fillard, Pennec, & Nicholas, 2007) la cual se define como :

$$\rho(X, Y) = \| \log(X) - \log(Y) \| \tag{2.17}$$

donde $\log(X)$ es el mapa logarítmico de la matriz de covarianza, el cual es definido por la descomposición de valores singulares de la matriz $X$. Se define la descomposición de valores singulares para la matriz $X$ como:

$$\log(X) = U \Sigma U^T \tag{2.18}$$

donde $U$ es una matriz ortonormal y $\Sigma = Diag(\lambda_1, \ldots, \lambda_n)$, la matriz diagonal de valores propios de $X$. Por ende, el mapa logarítmico queda definido como:

$$\log(X) = U \left[ Diag(\lambda_1, \ldots, \lambda_n) \right] U^T \tag{2.19}$$

## 2.6. Resultados Preliminares

En esta sección mostramos algunos resultados de experiencias menores, las cuales se fueron efectuando en el transcurso de la tesis. Estas tuvieron como fin entender mejor las capacidades de nuestro algoritmo.

### 2.6.1. Elección de Características

Una duda que surgió fue si lo que obteníamos estaba demasiado influenciado por las características de intensidad de la imagen provocando un simple reconocedor de bordes.

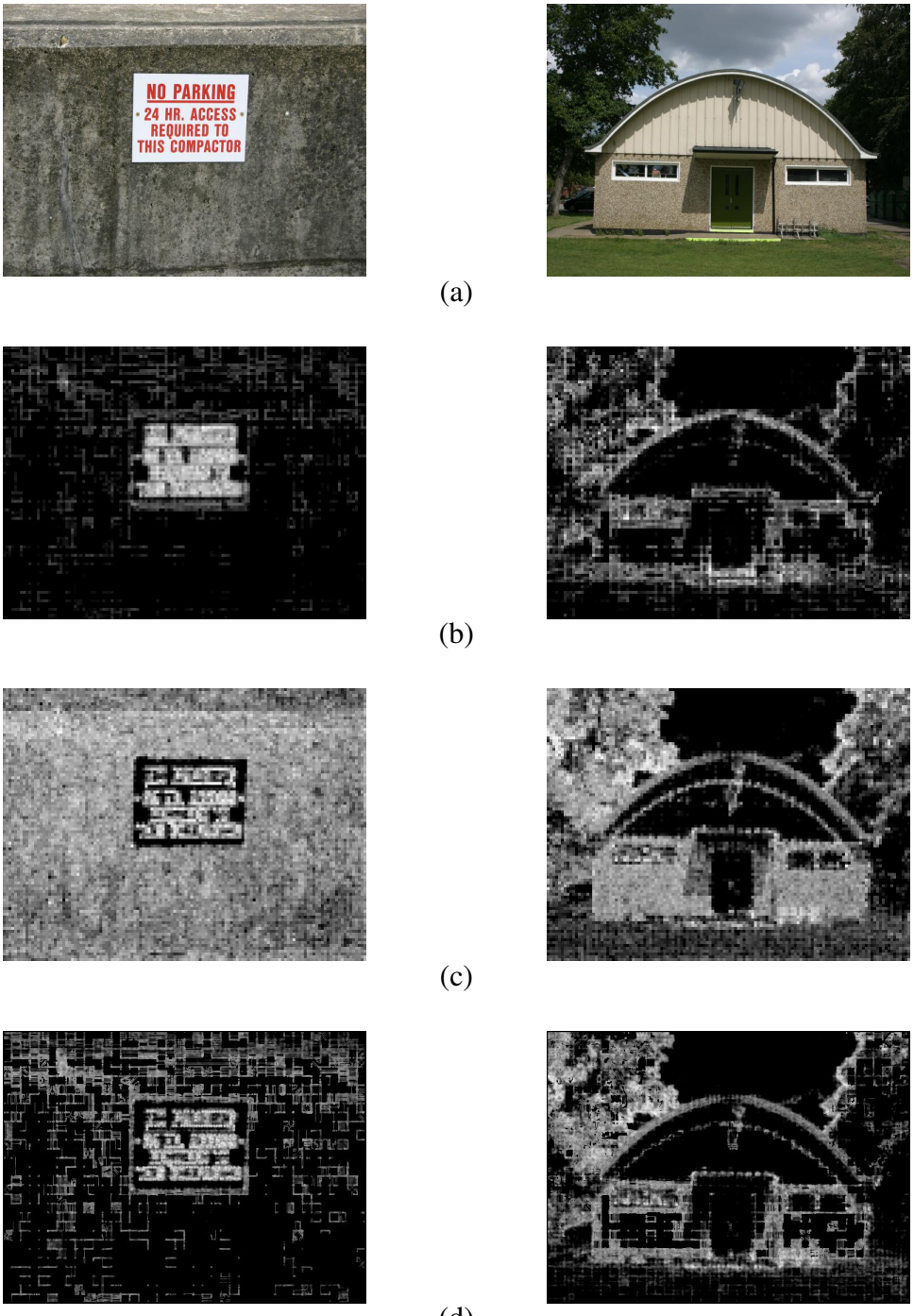



FIGURA 2.4. Ejemplo del mapa de saliencia obtenido a través de distintas matrices de Covarianza: (a) imagen original; (b) imagen del mapa de saliencia con características de colores; (c) imagen del mapa de saliencia con característica de intensidad; (d) imagen del mapa de saliencia con ambas características.

Por lo tanto hicimos experimentos para determinar la influencia del color y de la intensidad en el mapa de saliencia (Figura 2.4). De los resultados determinamos que la intensidad resalta tanto bordes como texturas, mientras que los colores resalta mayormente los lugares donde existen un variaciones de color. Por lo tanto la suma de las características nos da un detector de bordes resaltando las zonas con cambios de color y opacando las zonas con textura. Por otra lado, los detectores de bordes son más puntuales mientras que el descriptor utiliza una región dándonos una gran ventaja sobre los métodos de saliencia que utilizan bordes (Rosin, 2009).

Nuestro método utiliza once características para el tensor F, el cual queda definido como:

$$F(x,y,i) = [x \quad y \quad R \quad G \quad B \quad |I_x| \quad |I_y| \quad \sqrt{|I_x|^2 + |I_y|^2} \quad |I_{xx}| \quad |I_{yy}| \quad \sqrt{|I_{xx}|^2 + |I_{yy}|^2}] \tag{2.20}$$

Para un ánalisis más profundo en la selección de características para el descriptor de covarianza y el método de cálculo de la matriz de covarianza, alentamos al lector de leer el trabajo previamente hecho en este tema (Cortez et al., 2009).

### 2.6.2. Algoritmo de Puntos de Interés

Una de las ventajas de nuestro algoritmo frente a otros algoritmos de saliencia es que los puntos contienen la cantidad de información que existe a su alrededor. Si un punto tiene un alto valor, él y su vecindad son de interés para nosotros ya que implica que visualmente es una zona de altos cambios. Por lo tanto los puntos de interés son los puntos con mayor valor. El tamaño de las zonas de interés puede ser regulado a través del tamaño de la región del descriptor de Covarianza. En realidad un punto es de interés si su zona es de interés, la cual es equivalente a la región del descriptor.

Si utilizamos pequeñas regiones para el descriptor de covarianza, obtenemos zonas de interés pequeñas las cuales asemejan más a puntos de interés que a zonas. En cambio si utilizamos regiones grandes, obtenemos posibles regiones iniciales para algoritmos de seguimiento. Decimos posibles ya que como nos basamos en un sistema de saliencia, es muy posible que lo más saliente sea el fondo de la imagen (ver Figura 2.5).

De esta misma forma se puede variar el tamaño de la región y crear un set de puntos de interés con diferentes tamaños pero el costo en tiempo de ejecución se eleva al tener que volver a calcular el mapa de saliencia para cada tamaño de región.

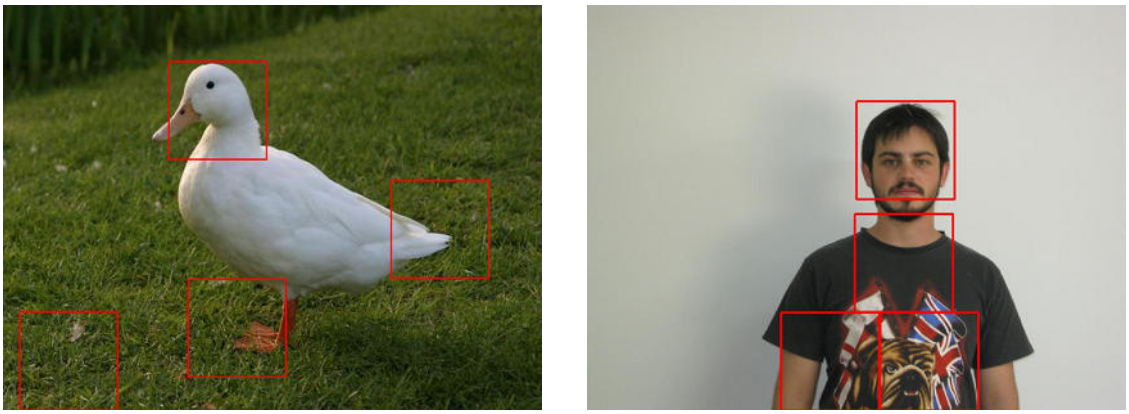


FIGURA 2.5. Ejemplo de las 4 zonas de mayor interés obtenidos con una región cuadrada de tamaño 115 pixel por lado.

# 3. METODOLOGIA

Para poner a prueba el algoritmo de saliencia, tomamos vídeos de personas caminando a través de un pasillo de un supermercado, donde la saliencia ocurre más en el fondo de la imagen que en la persona. De esta forma podemos utilizar los métodos de saliencia para eliminar el fondo de la ventana inicial. Para las pruebas creamos un conjunto de 30 vídeos, a los cuales se les etiquetó en cada uno de ellos una persona que pasó caminando. Como punto de partida tomamos dos regiones de inicialización: una manual, a partir de la etiquetación; y una automática, a partir del algoritmo de detección de personas de Felzenszwalb (Felzenszwalb, Girshick, McAllester, & Ramanan, 2010). Luego como experimento base, se ejecutó un algoritmo de seguimiento a partir de estas regiones, para luego comparar los resultados con los experimentos donde utilizamos un algoritmo de saliencia para modificar la región inicial y luego entregarsela al algoritmo de seguimiento.

## 3.1. Conjuntos de vídeos

Para elaborar un conjunto de evaluación, conseguimos permiso para tomar vídeos de un supermercado de Santiago. De estos vídeos que nos dan más de tres horas de grabación, tomamos algunas secuencias reales y otras que fueron preparadas por nosotros. Todo esto se hizo con el fin de probar lo más ampliamente posible los algoritmos de seguimiento, debido a que íbamos a reducir la ventana de seguimiento entonces debería tolerar oclusiones y semejanzas con otros elementos.

Al final creamos un conjunto de 30 vídeos, en los cuales seguimos a una sola persona desde su aparición en el vídeo hasta su salida, donde nos preocupamos de tener una variedad en el largo de los vídeos. Por ejemplo el vídeo más corto consta de solo 25 cuadros, equivalente a 1 segundo, mientras que el más largo consta de 1400 cuadros, equivalente a 56 segundos. Para cada vídeo se etiqueto a una persona en cada cuadro manualmente para poder tener los datos de posición de la persona y así luego comparar con los datos

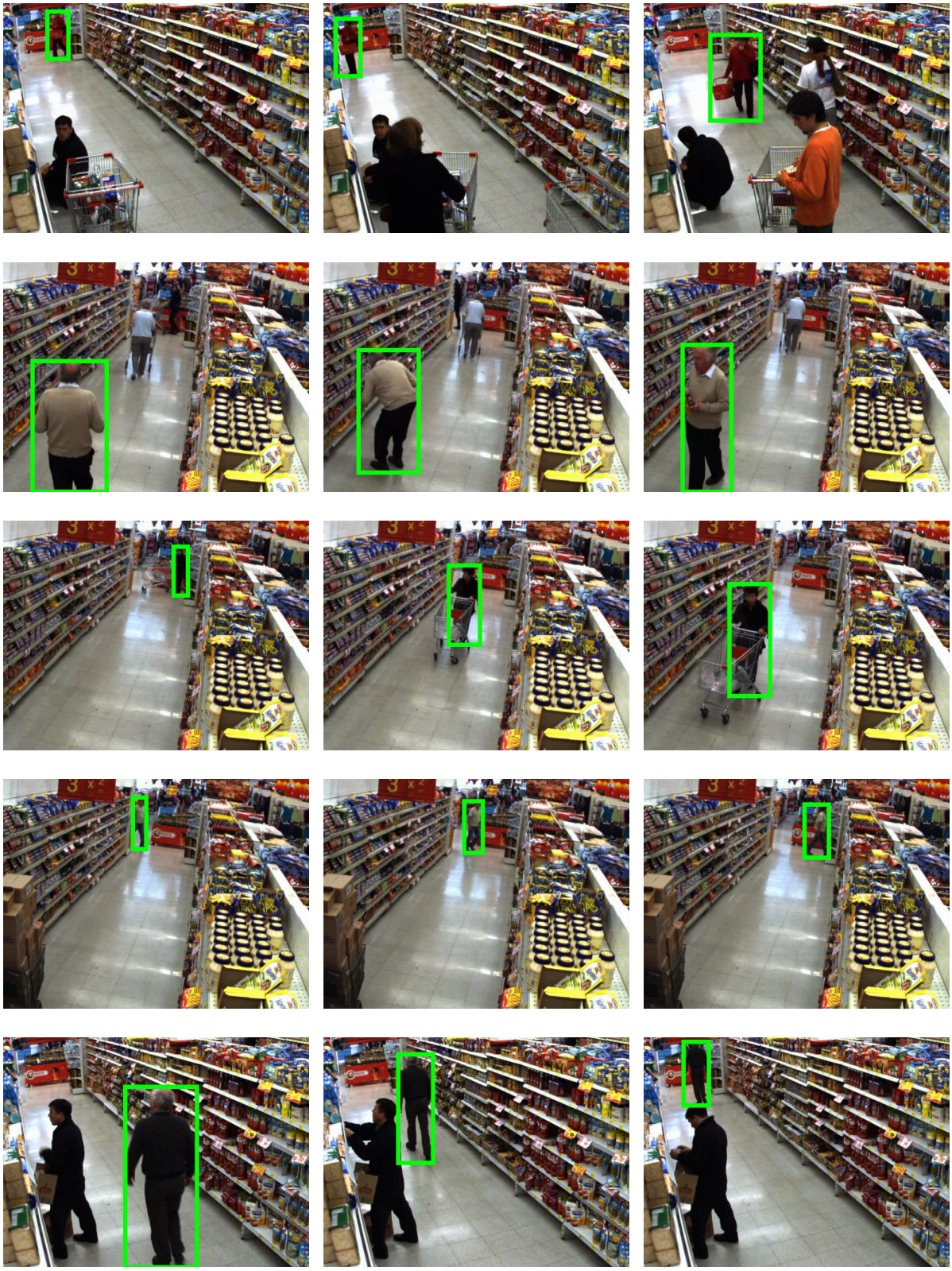
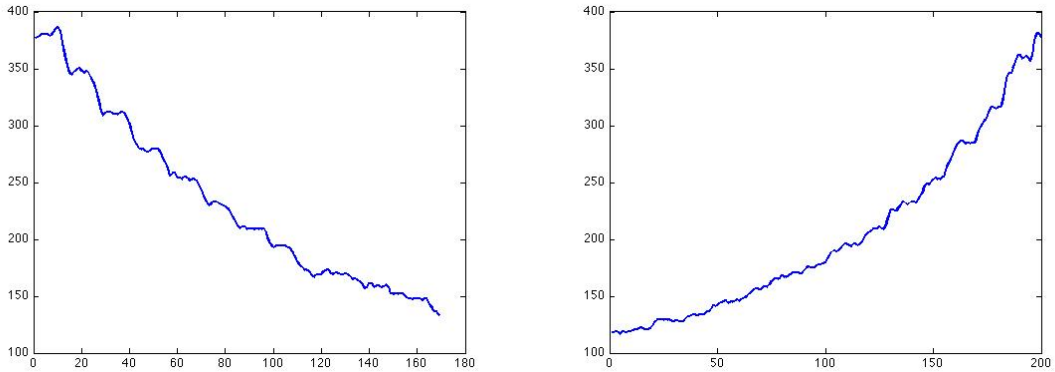


FIGURA 3.1. Ejemplos de los vídeos utilizados en la implementación.

obtenidos por los algoritmos de seguimiento. Con todo esto, formamos una base de datos de 30 vídeos con 8744 cuadros etiquetados y un total de 349 segundos, donde el promedio de duración de los videos es de 291 cuadros, lo que equivale a 11,6 segundos por vídeo.

Con esta etiquetación de personas en diferentes actividades en el pasillo, se pueden elaborar análisis de sus movimientos. Una buena característica es el ancho de la ventana que contiene a la persona ya que es más estable, a diferencia de la altura, debido al movimiento de las piernas. Por ejemplo si vemos la curva del ancho a través del tiempo de una persona que viene por el pasillo, ésta tiene forma parabólica creciente. Por el otro lado, si vemos la de una persona que va yendo por el pasillo, ésta tiene una forma parabólica decendiente (ver Figura 3.2). Cabe destacar que esto es solo uno de los beneficios que entrega la base de datos de vídeos elaborada, y que en ningún momento se efectuó algún tipo de trabajo sobre el tipo de movimiento.




(a) Curva de Ida (b) Curva de Venida

FIGURA 3.2. Curva del ancho de una ventana (en pixeles) que contiene a una persona en función del tiempo transcurrido (en cuadros) del vídeo.

## 3.2. Detección de Personas

Para inicializar los algoritmos de seguimiento debemos otorgarles la región inicial, donde se encuentra el objeto que deseamos seguir, en nuestro caso la región donde se

encuentra la persona a seguir. Por ende, decidimos hacer un experimento base donde el fondo es el menor posible estando la persona completa dentro de la región a seguir. Esta región la obtenemos de la etiquetación previamente hecha. Pero como nuestro deseo es probar nuestro método en un ambiente lo más real posible, experimentamos utilizando un detector de personas creado por Felzenszwalb (Felzenszwalb et al., 2010).

El detector de Felzenszwalb es un sistema de detección de objetos a base de mezclas de modelos de parte deformables a múltiples escalas. Este sistema se basa en representar un objeto como una colección de partes que capturan la apariencia local de éste, y que son dispuestas en una configuración deformable, donde algunas partes están conectadas entre si. Este sistema de partes deformables no es suficiente porque con ello solo describimos un modelo del objeto, tal como el problema de detectar bicicletas donde generalmente se describe solo la mountain bike y no un tandem. Para resolver esto, Felzenszwalb utiliza un sistema de mixturas o mezclas de módelos del objeto para resolver el problema de distintos tipos del objeto y de distintas vistas del modelo (ver Figura 3.3). Además cada modelo es formado con varios filtros sobre los histogramas de orientación de gradientes (HOG). Modificando las resoluciones de los filtros obtenemos que con menor resolución se toman en mayor cuenta las características más generales del objeto y con mayor resolución, características más precisas. Con esto podemos reutilizar modelos de partes que pueden aparecer en diferentes objetos como por ejemplo una rueda puede aparecer en un automóvil o en una motocicleta.

## 3.3. Algoritmos de Seguimiento

Como algoritmo de seguimiento utilizamos dos que son del estado del arte: el On-line Naive-Bayes Nearest Neighbor de (Cortez et al., 2010) y el TMD real-time algorithm (Kalal et al., 2009). Estos dos métodos son algoritmos de seguimiento en línea y con mínima información previa. Esto quiere decir que no utilizan información futura del vídeo,

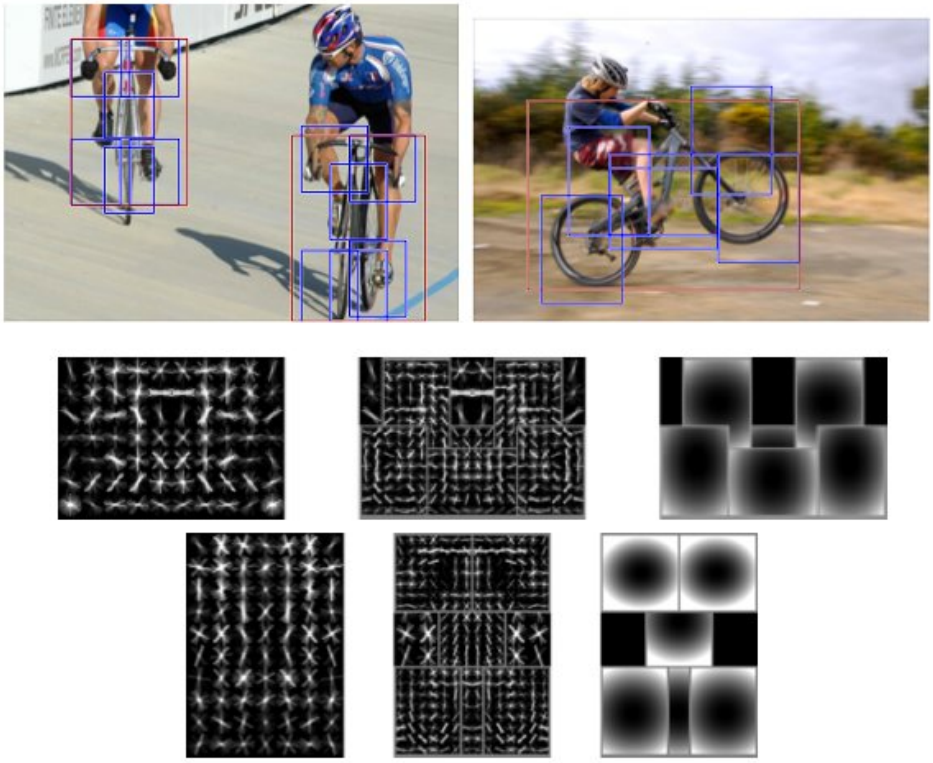


FIGURA 3.3. Ejemplo del sistema de Felzenszwalb para detección de objetos. Ejemplo de la importancia de utilizar mezcla de módelos para detectar bicicletas vistas desde el frente y desde un costado, y de la importancia de utilizar módelos de partes deformables que permite detectar una bicicleta mientras hacen un ”wheelie” (Felzenszwalb et al., 2010).

procesando de un cuadro a la vez, y que no conocen previamente el objeto a seguir, sino que la única información es la que se puede extraer del primer cuadro donde es seleccionado.

El On-line Naive Bayes Nearest Neighbor es un simple modelo de apariencia con sólo un parámetro, el cual es robusto a prolongadas oclusiones parciales y a drásticos cambios en apariencia. Su estrategia se basa en representar el objeto con el descriptor de covarianza y utilizar un clasificador en línea de vecinos cercanos para seguir el objeto en una secuencia de vídeo. Con este método logran un alto rendimiento reduciendo en promedio un 33% el error en pixeles de la ubicación del centro del objeto, en comparación con otros métodos de aprendizaje en línea.

Este algoritmo recibe un conjunto de parámetros que describen un conjunto de ventanas de búsquedas, donde $(x_i, y_i)$ representan al centro de la ventana, $(w_i, h_i)$ representan el ancho y el largo de esta y $\theta_i$ representa su rotación. Además este algoritmo almacena el modelo de apariencia $M$, el cual es una lista de ventanas que se han ido agregando debido a su parecido con la ventana inicial, y las distancias $R$, la cual es una lista de distancias promedios al modelo, de las ventanas al momento de agregarlas al modelo.

**Algoritmo 2**: On-line NBNN

**Data**: Dataset $\{x_i, y_i, w_i, h_i, \theta_i\}_{i=1}^{N}$, el modelo $M$ y las distancias promedio almacenadas $R$

**Result**: $x_{k^*}, y_{k^*}, w_{k^*}, h_{k^*}, \theta_{k^*}$

**for** $k \leftarrow 1$ **to** $N$ **do**

$I = \text{getImage}(x_k, y_k, w_k, h_k, \theta_k)$;

$P_k = \text{getCovarianceDescriptors}(I)$;

$d_k = d(M, P_k)$;

**end**

$k^* = \arg\min_k d_k$;

$P^B = P_{k^*}$;

$f = \arg\min_{i \in f} \omega(M, P^B, i)$;

**if** $s_f < N$ **then**

$j_f = |s_f + 1|$;

**else**

$j_f = \arg\max_{i \in s_f} R(f, i)$;

**end**

$M(f, j_f) = P^B(f)$;

$R(f, j_f) = \omega(M, P^B, f)$;

Por su parte el algoritmo TMD, por sus siglas en inglés Tracking-Modeling-Detecting (Siguiendo-Modelando-Detectando), integra seguimiento adaptivo con aprendizaje en línea del detector especifico del objeto. En comparación con el algoritmo anterior en vez de crear un modelo de apariencia, el TMD crea un detector especifico para el objeto seleccionado. Aunque en el estado del arte utilizar un detector basado en boosting es bastante común, Kalal et al. proponen un nuevo detector que utiliza como características el 2bit Binary Patterns (2bitBP) basado en Local Binary Patterns.

Por otra parte la trayectoria es mejorada por dos procesos: uno de crecimiento y otro de podaje. El proceso de crecimiento consiste en la selección apropiada de muestras a partir de la trayectoria del objeto. Las nuevas muestras son sometidas a un criterio de similitud, como por ejemplo tener una distancia a la primera muestra menor a un umbral, y si cumplen con éste son utilizadas para actualizar el detector. El proceso de podaje consiste en eliminar las detecciones incorrectas. El detector puede encontrar más de un objeto en la imagen, y asumiendo que existe solo uno, se utiliza aquel que este sobre la trayectoria del objeto. Estos procesos tienen como fin mejorar el modelo del detector del objeto, disminuyendo falsos positivos y logrando adaptarse mejor a los cambios de aspecto.

**Algoritmo 3**: TMD framework

**Data**: Select $x_0$, $L_0 = x_0$

**for** $t \leftarrow 1$ **to do**

Track last patch $x_{t-1}$;

Detect patches contained in online model $L_{t-1}$;

$L_t \leftarrow L_{t-1} \cup$ Positive samples from growing;

$L_t \leftarrow L_{t-1} \setminus$ Negative samples from pruning;

$x_t \leftarrow$ Most confident patch (detected or tracked);

**end**

# 4. RESULTADOS

En esta sección veremos los resultados obtenidos: en una primera parte, por el algoritmo de saliencia propuesto por nosotros en comparación con otros algoritmos de saliencia del estado del arte; y en una segunda parte, los resultados de la automatización de la selección del parámetro de porcentaje de información dependiendo de como definimos el éxito del algoritmo de seguimiento. Ambos experimentos fueron realizados sobre un conjunto de 30 videos con una media de 10 segundos de duración.

## 4.1. Algoritmo de Saliencias

Para comparar nuestro algoritmo de saliencia efectuamos cuatro experimentos: inicializamos con *ground thruth* los dos algoritmos de seguimiento y luego inicializamos con el algoritmo de detección de personas de Felzenszwalb. De esta forma obtenemos una buena comparación del algoritmo tanto en un ambiente controlado como en uno real donde la inicialización es más imprecisa. En ambos casos, para reducir el tamaño de la ventana inicial se utilizaron tres algoritmos de saliencia: el propuesto por nosotros (Cov), el Center Surround (CS) y el de Itti et al. (Itti).

Los resultados muestran que los algoritmos de saliencia incrementan fuertemente la precisión, alrededor de un 30% en promedio para el algoritmo ONBNN (ver Tabla 4.1). Sin embargo esta alza produce, en ciertos casos, la disminución del recall. Nuestro método de saliencia obtiene resultados altos en la precisión, y sin comprometer tanto el recall, y en algunos casos incluso logra aumentar el nivel de recall. Esto se ve reflejado en que obtenemos mejores resultados de $F_{0.5-Score}$ y $E_{Score}$. El algoritmo Center Surround da resultados muy parecidos a nuestro algoritmo pero se encuentra levemente por debajo en los resultados de *scores*. Esto se debe a que sus buenos resultados en precisión influyen en el recall, el cual desciende. Por último el algoritmo de Itti et al. muestra los peores resultados debido a que reduce en demasía la región inicial. Es por esto que obtiene buenos

TABLA 4.1. Porcentaje promedio para el algoritmo ONBNN.

| | (%) | Precision | | | Recall | | | $F_{0.5-Score}$ | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Saliency | | Cov | C.S. | Itti's | Cov | C.S. | Itti's | Cov | C.S. | Itti's |
| Ground Truth | 10 | **90.20** | 86.13 | 4.67 | **23.14** | 15.86 | 0.04 | **50.15** | 40.81 | 0.18 |
| | 20 | **87.71** | 87.57 | 10.94 | **29.86** | 23.53 | 0.52 | **57.71** | 52.60 | 1.90 |
| | 30 | **90.57** | 90.29 | 25.32 | **37.56** | 32.07 | 0.90 | **65.15** | 62.04 | 3.20 |
| | 40 | **85.90** | 84.89 | 39.64 | **41.25** | 38.90 | 1.66 | **66.08** | 65.84 | 5.94 |
| | 50 | 82.90 | **83.24** | 51.96 | **45.96** | 41.37 | 3.11 | **67.84** | 65.35 | 10.48 |
| | 60 | **83.69** | 81.77 | 60.88 | 47.56 | **47.71** | 5.32 | **69.62** | 68.16 | 15.93 |
| | 70 | **85.74** | 83.85 | 73.14 | **51.30** | 48.35 | 7.04 | **72.52** | 66.46 | 20.77 |
| | 80 | **78.13** | 76.68 | 77.28 | 50.35 | **51.17** | 10.06 | **67.89** | 66.46 | 26.95 |
| | 90 | 76.38 | 78.28 | **79.59** | **52.78** | 52.76 | 13.23 | 67.93 | **68.42** | 33.37 |
| | 100 | | 63.82 | | | 52.79 | | | 59.61 | |
| Felzenszwalb | 10 | **80.66** | 79.04 | 5.98 | **20.58** | 17.31 | 0.06 | **44.62** | 42.84 | 0.18 |
| | 20 | 85.03 | **85.21** | 13.51 | **29.46** | 26.73 | 0.16 | **56.79** | 55.91 | 0.69 |
| | 30 | **81.08** | 76.81 | 23.05 | **34.64** | 32.36 | 0.42 | **59.80** | 57.93 | 1.71 |
| | 40 | **81.32** | 78.71 | 34.40 | **40.66** | 39.04 | 0.92 | **65.07** | 63.04 | 3.65 |
| | 50 | 74.24 | **77.56** | 50.14 | 42.17 | **43.35** | 2.14 | 62.07 | **64.54** | 8.43 |
| | 60 | **71.24** | 69.94 | 55.44 | **45.86** | 44.15 | 3.16 | **60.62** | 52.23 | 11.69 |
| | 70 | **73.53** | 70.81 | 66.09 | **48.41** | 46.92 | 4.75 | **65.01** | 62.62 | 16.76 |
| | 80 | 66.31 | 69.27 | **75.82** | 47.00 | **48.74** | 6.74 | 59.60 | **62.48** | 22.74 |
| | 90 | 67.81 | 63.55 | **83.75** | **51.22** | 46.85 | 9.56 | **62.18** | 58.12 | 29.57 |
| | 100 | | 60.43 | | | 51.35 | | | 56.89 | |

resultados en los porcentajes altos, 80% y 90%, pero mantiene un recall extremadamente bajo, no alcanzando ni a la mitad de los otros algoritmos.

Cuando se utiliza el algoritmo de Felzenszwalb para detectar personas como regiones iniciales apreciamos que los resultados no son tan buenos como los con ground truth pero aun así se produce una mejora del 20% en la precisión. El comportamiento de los algoritmos de saliencia se mantiene igual. Nuestro algoritmo es el que en general responde mejor, seguido con resultados muy parecidos por el algoritmo Center Surround, y por último el algoritmo de Itti que solo es considerable para los porcentajes altos.

Por último se puede apreciar que los mejores resultados para el ground truth se obtienen utilizando un 70% de la información total, y para Felzenszwalb se obtienen utilizando un 40% de la información. Mejorando las zonas iniciales hemos podido prevenir que los

TABLA 4.2. Porcentaje promedio para el algoritmo TMD.

| | (%) | Precision | | | Recall | | | $F_{0.5-Score}$ | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Saliency | | Cov | C.S. | Itti's | Cov | C.S. | Itti's | Cov | C.S. | Itti's |
| Ground Truth | 10 | **49.48** | 40.01 | 0.00 | **16.41** | 10.23 | 0.00 | **32.17** | 24.56 | 0.00 |
| | 20 | 52.34 | **56.06** | 0.00 | **20.68** | 17.72 | 0.00 | 36.12 | **38.22** | 0.00 |
| | 30 | **66.04** | 60.74 | 3.33 | **27.00** | 20.04 | 0.01 | **47.13** | 41.41 | 0.06 |
| | 40 | **70.06** | 61.77 | 4.54 | **31.36** | 27.72 | 0.49 | **51.55** | 46.96 | 0.35 |
| | 50 | 67.53 | **70.62** | 16.75 | **34.02** | 31.40 | 1.97 | **52.05** | 51.66 | 4.96 |
| | 60 | **67.40** | 67.40 | 19.24 | 36.77 | **37.76** | 2.72 | 52.70 | **54.65** | 7.18 |
| | 70 | 61.65 | **62.85** | 27.07 | 34.80 | **37.86** | 3.24 | 50.53 | **51.79** | 9.25 |
| | 80 | 58.14 | **62.86** | 26.96 | **37.35** | 36.24 | 3.30 | 49.06 | **50.43** | 10.79 |
| | 90 | 59.86 | **60.64** | 41.67 | **37.25** | 35.72 | 5.76 | **50.10** | 49.94 | 17.67 |
| | 100 | | 52.51 | | | 38.89 | | | 47.30 | |
| Felzenszwalb | 10 | 15.20 | **18.11** | 0.00 | 3.75 | **4.36** | 0.00 | 8.90 | **10.89** | 0.00 |
| | 20 | 23.95 | **27.20** | 0.00 | 6.87 | **7.63** | 0.00 | **15.72** | 15.71 | 0.00 |
| | 30 | 30.82 | **39.94** | 0.00 | 8.76 | **12.58** | 0.00 | 19.86 | **26.40** | 0.00 |
| | 40 | 35.74 | **61.83** | 0.00 | 13.03 | **21.23** | 0.00 | 25.98 | **42.97** | 0.00 |
| | 50 | 35.25 | **64.49** | 0.00 | 13.47 | **28.67** | 0.00 | 25.88 | **50.00** | 0.00 |
| | 60 | 37.43 | **63.68** | 0.00 | 19.50 | **32.23** | 0.00 | 30.32 | **50.91** | 0.00 |
| | 70 | 46.58 | **67.09** | 0.00 | 25.64 | **33.67** | 0.00 | 40.03 | **54.15** | 0.00 |
| | 80 | 61.95 | **65.02** | 0.00 | 34.50 | **35.20** | 0.00 | 50.89 | **52.99** | 0.00 |
| | 90 | **64.80** | 59.53 | 0.00 | 37.42 | **38.10** | 0.00 | **54.24** | 50.88 | 0.00 |
| | 100 | | 50.20 | | | 36.29 | | | 42.08 | |

algoritmos de seguimiento se queden pegados en la posición inicial, ya que reduciendo la región los algoritmos se confunden menos con el fondo.

Por otra parte, el algoritmo de seguimiento TMD nos entrega resultados muy diferentes entre el algoritmo iniciado con ground truth y el iniciado con el detector de personas de Felzenszwalb (ver Tabla 4.2). Para el primero, los resultados muestran que nuestro algoritmo da mejores resultados para la mayoría de los porcentajes. Sin embargo los mejores resultados son obtenidos por el algoritmo Center Surround utilizando el 60% de la información total de la región inicial. Aun que los resultados de precisión aumentan hasta en un 18%, como es el caso del 40% para nuestro algoritmo y 50% para el algoritmo Center Surround, el recall disminuye en todos los casos, siendo un 1% lo menos que disminuye.

Para el caso del algoritmo iniciado con el detector de personas, los resultados favorecen al algoritmo Center Surround, teniendo los mejores resultados en siete de los nueve casos. El algoritmo de Itti, no logra funcionar en ninguno de los casos, y nuestro algoritmo funciona para porcentajes altos. Esto se debe a que el algoritmo de seguimiento TMD no funciona con áreas pequeñas a diferencia del algoritmo ONBNN. Pero incluso con esta limitante, obtenemos una mejora del 14% en la precisión y un aumento en el recall de 2% para el mejor caso, el cual es utilizando nuestro algoritmo en un 90%.

## 4.2. Automatización de parámetros

Para evaluar la automatización del parámetro de porcentaje de información, efectuamos cuatro experimentos en la red bayesiana, en las cuales se modificó la forma de determinar el éxito del algoritmo de seguimiento. Primero se definió el éxito como el nivel de precisión, luego como el nivel de recall, como el $F_{0.5-Score}$ y finalmente como el $E-Score$. Esto se aplicó a nuestro algoritmo en los dos métodos de seguimiento, ya que la similitud de los vídeos se hizo a través de la matriz de covarianza de las regiones iniciales (ver Tablas 4.3 y 4.6). Sin embargo como el algoritmo ONBNN también utiliza la matriz de covarianza para efectuar el seguimiento, entonces es posible utilizar nuestro método de automatización en los otros algoritmos de saliencia (ver Tablas 4.4 y 4.5).

Para el algoritmo de seguimiento ONBNN y una inicialización de Ground Truth, los resultados muestran que es posible aumentar la precisión desde un 63.8% hasta un 91.1% con nuestro algoritmo (ver Tabla 4.3), un 90.3% con el algoritmo Center Surround (ver Tabla 4.4) y un 80.3% con el algoritmo de Itti (ver Tabla 4.5). Estos maximos suceden cuando seleccionamos la precisión como definición de éxito para el algoritmo de seguimiento. Sin embargo, esto produce que el recall se desestime y caiga alrededor de 20%. De la misma manera si elegimos el recall como éxito del algoritmo de seguimiento, obtenemos un pequeño aumento del recall en un par de puntos porcentuales, pero la

TABLA 4.3. Porcentaje promedio utilizando la red bayesiana en el algoritmo ONBNN con Saliencia de Covarianza.

| | Var. | Precision | Recall | $F_{0.5-Score}$ | $E_{Score}$ |
|---|---|---|---|---|---|
| Ground Truth | ONBNN | 63.8 | 52.8 | 59.6 | 59.6 |
| | Precision | 91.1 | 32.9 | 62.3 | 62.3 |
| | Recall | 71.4 | 52.6 | 64.0 | 64.0 |
| | $F_{0.5-Score}$ | 83.8 | 50.2 | 70.9 | 70.9 |
| | $E_{Score}$ | 83.8 | 50.2 | 70.9 | 70.9 |
| Felzenszwalb | ONBNN | 60.4 | 51.4 | 56.9 | 56.3 |
| | Precision | 83.4 | 31.2 | 57.3 | 57.7 |
| | Recall | 59.9 | 49.8 | 55.9 | 55.3 |
| | $F_{0.5-Score}$ | 74.3 | 44.2 | 61.9 | 60.7 |
| | $E_{Score}$ | 74.3 | 44.9 | 62.2 | 61.1 |

TABLA 4.4. Porcentaje promedio utilizando la red bayesiana en el algoritmo ONBNN con Saliencia de Center Surround.

| | Var. | Precision | Recall | $F_{0.5-Score}$ | $E_{Score}$ |
|---|---|---|---|---|---|
| Ground Truth | ONBNN | 63.8 | 52.8 | 59.6 | 59.6 |
| | Precision | 90.3 | 33.3 | 61.1 | 61.8 |
| | Recall | 73.7 | 54.8 | 66.4 | 65.4 |
| | $F_{0.5-Score}$ | 80.5 | 51.3 | 66.8 | 65.2 |
| | $E_{Score}$ | 78.9 | 50.1 | 65.5 | 64.3 |
| Felzenszwalb | ONBNN | 60.4 | 51.4 | 56.9 | 56.3 |
| | Precision | 85.2 | 26.8 | 55.8 | 55.7 |
| | Recall | 60.5 | 51.1 | 56.9 | 56.3 |
| | $F_{0.5-Score}$ | 77.1 | 44.0 | 64.2 | 62.5 |
| | $E_{Score}$ | 77.9 | 44.8 | 65.0 | 63.4 |

TABLA 4.5. Porcentaje promedio utilizando la red bayesiana en el algoritmo ONBNN con Saliencia de Itti et al.

| | Var. | Precision | Recall | $F_{0.5-Score}$ | $E_{Score}$ |
|---|---|---|---|---|---|
| Ground Truth | ONBNN | 63.8 | 52.8 | 59.6 | 59.6 |
| | Precision | 80.3 | 15.2 | 32.2 | 39.9 |
| | Recall | 63.8 | 52.8 | 59.6 | 59.6 |
| | $F_{0.5-Score}$ | 63.8 | 52.8 | 59.6 | 59.6 |
| | $E_{Score}$ | 62.5 | 50.7 | 57.2 | 56.9 |
| Felzenszwalb | ONBNN | 60.4 | 51.4 | 56.9 | 56.3 |
| | Precision | 83.7 | 10.8 | 28.4 | 36.4 |
| | Recall | 60.4 | 51.4 | 56.9 | 56.3 |
| | $F_{0.5-Score}$ | 60.4 | 51.4 | 56.9 | 56.3 |
| | $E_{Score}$ | 60.4 | 51.4 | 56.9 | 56.3 |

precisión no aumenta tanto como en el caso anterior, llegando a un 71% para nuestro algoritmo y a un 73% el de Center Surround. El algoritmo de Itti es caso aparte ya que como disminuye demasiado el tamaño de la ventana, los mejores resultados si no consideramos la precisión se dan cuando no utilizamos el algoritmo al inicio, o sea se da con un 100% de la información de la ventana. Por último el $F_{0.5-Score}$ y $E_{Score}$ dan resultados muy parecidos sino idénticos, en todos todos los casos. Los resultados muestran que utilizando estas métricas como definición del éxito, nuestro algoritmo obtiene mejores resultados en precisión y con menor perdida en recall.

Para este mismo algoritmo de seguimiento y una inicialización basada en los resultados del algoritmo de detección de personas de Felzenszwalb, los resultados mantienen el mismo patron. La diferencia es que los resultados son un poco menores debido a que la región inicial es menos precisa, excepto para el algoritmo de Itti, el cual mejora, dándole la oportunidad de ser una buena opción frente a los otros dos algoritmos.

Por último el algoritmo de seguimiento TMD entrega los peores resultados, aunque nuestro algoritmo de saliencia logra mejorarlos en un 16% aproximadamente (ver Tabla 4.6), éstos siguen siendo bajos (alrededor de un 60%). El recall no fue posible aumentarlo en las experiencias inicializadas con ground truth, siendo el caso base (100% de la información) el mejor caso de todos. Pero en las experiencias inicializadas con el algoritmo de Felzenszwalb obtenemos mejoras en precisión y en recall para todos los casos. Por otra parte los malos resultados generales se deben a que ciertos vídeos no pudieron ser evaluados ya que el algoritmo consideraba la región inicial muy pequeña y no funcionaba correctamente. Esta es la misma razón porque los porcentajes más altos dan mejores resultados.

Por otro lado y visto desde una perspectiva con mayor precisión, podemos ver que no tan solo existe un aumento en la precisión de los algoritmos de seguimiento, sino que también existe un aumento del tiempo en que se sigue a las personas en los vídeos (ver

TABLA 4.6. Porcentaje promedio utilizando la red bayesiana en el algoritmo TMD con Saliencia de Covarianza

| | Var. | Precision | Recall | $F_{0.5-Score}$ | $E_{Score}$ |
|---|---|---|---|---|---|
| Ground Truth | ONBNN | 52.5 | 38.9 | 47.3 | 46.5 |
| | Precision | 70.8 | 31.7 | 52.7 | 51.9 |
| | Recall | 49.9 | 38.5 | 44.8 | 44.5 |
| | $F_{0.5-Score}$ | 64.3 | 33.6 | 50.0 | 49.6 |
| | $E_{Score}$ | 66.8 | 33.7 | 51.0 | 50.9 |
| Felzenszwalb | ONBNN | 50.2 | 36.3 | 42.1 | 42.0 |
| | Precision | 63.9 | 36.1 | 51.7 | 50.3 |
| | Recall | 63.5 | 37.1 | 52.4 | 51.5 |
| | $F_{0.5-Score}$ | 66.5 | 37.4 | 55.4 | 53.8 |
| | $E_{Score}$ | 66.5 | 37.4 | 55.4 | 53.8 |

Tablas 4.7 y 4.8). Dado que mejoramos la región inicial evitamos que los algoritmos de seguimiento pierdan a la persona que sigan, debido a un exceso de información sobre el fondo de la imagen. Para el algoritmo On-line Naive Bayes Nearest Neighbor utilizando la saliencia de covarianza obtenemos una mejoría en los vídeos donde el seguimiento se efectua completamente, pasando de 19 vídeos a 27 vídeos (ver Tabla 4.7). Los tres vídeos en los cuales falla nuestro método propuesto es debido a que son vídeos de mayor dificultad, varias oclusiones parciales o completas y movimientos rápidos. Con esto evitamos que el algoritmo de seguimiento no siga a la persona y se quede con el fondo como objetivo principal del seguimiento (ver Figura 4.1). Como hemos visto previamente los resultados con el algoritmo de seguimiento TMD no son tan prometedores, pero existe una mejora tanto en tiempo como en cantidades de vídeos, pasando de 11 vídeos completamente seguidos a 14 vídeos (ver Tabla 4.8).

TABLA 4.7. Resultados obtenidos utilizando la red bayesiana en el algoritmo ONBNN con Saliencia de Covarianza e inicialización Ground Truth. La primera columna es el número del vídeo, la segunda es el tiempo en segundos que dura el vídeo, la tercera y cuarta columna corresponden a los tiempos que el algoritmo ONBNN y el propuesto por nosotros, respectivamente, siguen correctamente a una persona, y la quinta y sexta columna son la precisión del algoritmo ONBNN y el propuesto por nosotros utilizando la saliencia de covarianza y la red bayesiana con la precisión como éxito.

| | | Tiempo | | Precisión | |
|---|---|---|---|---|---|
| Vídeo | Segundos(s) | ONBNN(s) | ONBNN+Cov(s) | ONBNN(%) | ONBNN+Cov(%) |
| 1 | 30.0 | 30.0 | 30.0 | 69.9 | 99.9 |
| 2 | 7.2 | 7.2 | 7.2 | 79.0 | 98.7 |
| 3 | 56.0 | 29.6 | 29.6 | 30.1 | 52.2 |
| 4 | 6.8 | 6.8 | 6.8 | 73.4 | 95.2 |
| 5 | 8.2 | 1.1 | 8.2 | 6.6 | 97.9 |
| 6 | 6.4 | 6.4 | 6.4 | 90.0 | 100.0 |
| 7 | 8.7 | 1.0 | 8.7 | 8.4 | 99.8 |
| 8 | 16.0 | 16.0 | 16.0 | 94.2 | 96.7 |
| 9 | 6.8 | 6.8 | 6.8 | 57.5 | 100.0 |
| 10 | 4.8 | 4.8 | 4.8 | 78.9 | 95.1 |
| 11 | 7.9 | 7.9 | 7.9 | 57.1 | 85.7 |
| 12 | 2.0 | 1.4 | 2.0 | 58.1 | 98.8 |
| 13 | 1.6 | 1.4 | 1.6 | 60.0 | 100.0 |
| 14 | 22.0 | 3.0 | 22.0 | 7.6 | 94.4 |
| 15 | 13.4 | 13.4 | 13.4 | 91.6 | 99.4 |
| 16 | 34.0 | 34.0 | 32.0 | 64.0 | 55.4 |
| 17 | 1.8 | 1.4 | 1.8 | 48.2 | 99.7 |
| 18 | 29.4 | 29.4 | 29.4 | 86.1 | 90.9 |
| 19 | 9.4 | 7.6 | 9.4 | 26.6 | 98.7 |
| 20 | 3.0 | 1.6 | 1.2 | 45.1 | 30.1 |
| 21 | 10.0 | 10.0 | 10.0 | 87.1 | 99.3 |
| 22 | 5.0 | 5.0 | 5.0 | 88.9 | 89.8 |
| 23 | 6.0 | 6.0 | 6.0 | 85.8 | 96.6 |
| 24 | 8.6 | 8.6 | 8.6 | 90.0 | 96.3 |
| 25 | 10.0 | 10.0 | 10.0 | 40.5 | 96.7 |
| 26 | 1.0 | 1.0 | 1.0 | 81.2 | 95.2 |
| 27 | 11.6 | 10.8 | 11.6 | 57.1 | 87.3 |
| 28 | 11.0 | 11.0 | 11.0 | 78.73 | 98.4 |
| 29 | 2.4 | 2.4 | 2.4 | 92.6 | 87.8 |
| 30 | 8.8 | 8.8 | 8.8 | 80.4 | 98.5 |
| Total | 349.8 | 284.4 | 330.7 | - | - |
| Promedio Simple | - | 85.3 | 96.9 | 63.8 | 91.2 |
| Promedio Ponderado | - | 81.3 | 94.6 | 64.7 | 81.7 |

TABLA 4.8. Resultados obtenidos utilizando la red bayesiana en el algoritmo TMD con Saliencia de Covarianza e inicialización Ground Truth. La primera columna es el número del vídeo, la segunda es el tiempo en segundos que dura el vídeo, la tercera y cuarta columna corresponden a los tiempos que el algoritmo TMD y el propuesto por nosotros, respectivamente, siguen correctamente a una persona, y la quinta y sexta columna son la precisión del algoritmo TMD y el propuesto por nosotros utilizando la saliencia de covarianza y la red bayesiana con la precisión como éxito.

| | | Tiempo | | Precisión | |
|---|---|---|---|---|---|
| Vídeo | Segundos(s) | TMD(s) | TMD+Cov(s) | TMD(%) | TMD+Cov(%) |
| 1 | 30.0 | 29.9 | 23.7 | 87.4 | 83.0 |
| 2 | 7.2 | 7.2 | 6.4 | 95.0 | 96.0 |
| 3 | 56.0 | 32.6 | 27.6 | 26.3 | 67.1 |
| 4 | 6.8 | 6.7 | 6.0 | 74.0 | 54.0 |
| 5 | 8.2 | 2.5 | 5.5 | 30.4 | 75.5 |
| 6 | 6.4 | 6.4 | 6.4 | 98.4 | 99.8 |
| 7 | 8.7 | 1.1 | 0 | 5.3 | 0 |
| 8 | 16.0 | 2.0 | 15.9 | 33.9 | 45.9 |
| 9 | 6.8 | 6.8 | 6.8 | 60.6 | 87.3 |
| 10 | 4.8 | 4.8 | 4.8 | 85.1 | 87.6 |
| 11 | 7.9 | 7.9 | 7.0 | 38.1 | 90.7 |
| 12 | 2.0 | 0 | 0 | 0 | 0 |
| 13 | 1.6 | 1.6 | 1.6 | 88.1 | 98.2 |
| 14 | 22.0 | 1.8 | 1.3 | 4.2 | 3.2 |
| 15 | 13.4 | 13.1 | 13.4 | 69.6 | 100.0 |
| 16 | 34.0 | 19.2 | 25.9 | 33.2 | 72.6 |
| 17 | 1.8 | 1.2 | 1.8 | 36.7 | 96.8 |
| 18 | 29.4 | 19.72 | 14.8 | 42.8 | 31.8 |
| 19 | 9.4 | 7.7 | 9.4 | 32.5 | 97.1 |
| 20 | 3.0 | 0.9 | 0.9 | 15.6 | 20.0 |
| 21 | 10.0 | 10.0 | 10.0 | 90.6 | 98.2 |
| 22 | 5.0 | 5.0 | 5.0 | 82.2 | 88.8 |
| 23 | 6.0 | 6.0 | 6.0 | 74.3 | 79.1 |
| 24 | 8.6 | 8.6 | 8.6 | 87.8 | 95.3 |
| 25 | 10.0 | 10.0 | 10.0 | 72.6 | 90.4 |
| 26 | 1 | 0.8 | 0.9 | 79.9 | 64.9 |
| 27 | 11.6 | 2.5 | 5.0 | 12.8 | 25.2 |
| 28 | 11.0 | 9.2 | 10.9 | 39.4 | 89.3 |
| 29 | 2.4 | 0.88 | 2.4 | 18.2 | 93.4 |
| 30 | 8.8 | 7.48 | 92.87 | 60.6 | 92.9 |
| Total | 349.8 | 233.5 | 246.8 | - | - |
| Promedio Simple | - | 70.7 | 78.6 | 52.5 | 70.8 |
| Promedio Ponderado | - | 66.8 | 70.7 | 46.4 | 70.0 |


(a) (b)

FIGURA 4.1. Resultados obtenidos en imágenes con harta información en el fondo. La primera columna corresponde a los resultados base utilizando ONBNN. La segunda columna es el resultado de aplicar nuestro algoritmo a la región inicial.

# 5. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo hemos demostrado que es posible mejorar los algoritmos de seguimiento a través de un pequeño ajuste a la ventana inicial en la cual se encuentra el objeto descrito. Para ello utilizamos algoritmos de saliencia para reducir la ventana inicial y eliminar lo más posible el fondo que se encuentra en el borde de la ventana. Por otro lado desarrollamos un novedoso sistema de saliencia basado en la covarianza de las características en la vecindad de un punto.

## 5.1. Revisión de los Resultados y Comentarios Generales

Los resultados obtenidos previamente confirman que el uso de sistemas de saliencia mejoran la precisión y el desempeño de los algoritmos de seguimiento. Por ejemplo podemos evitar que la ventana se quede con el fondo en vez de seguir a la persona debido a la mayor información que contiene el fondo en comparación a la que tiene la persona a seguir (ver Figura 4.1). Aunque esto no siempre es posible ya que algunos algoritmos de seguimientos no dan buenos resultados con ventanas muy pequeñas, por lo tanto la reducción de la ventana queda descartada para ellos.

Por otro lado vemos que la utilización de los *scores* es aconsejable debido a que aumentan tanto la precisión como el recall, manteniendose en porcentajes medios. Para mejorar la precisión uno debe elegir porcentajes pequeños (entre 10 y 30 porciento) y para el mejorar el recall los porcentajes más altos (entre 80 y 100 porciento). Pero al reducir la ventana estamos perdiendo información que es valiosa en caso de oclusión o de mucha similitud entre dos personas. Dado esto, los mejores resultados se generan con valores medios (entre 40 y 70 porciento).

### 5.1.1. Algoritmo de Saliencias

De los resultados obtenidos podemos concluir que los algoritmos de saliencia incrementan fuertemente la precisión, alrededor de un 26% en promedio para el algoritmo ONBNN. Sin embargo esta alza se produce en conjunto a una disminución del recall. Sin embargo como ya quedó demostrado que en ciertos casos evitamos que el algoritmo se pierda por confundirse con el fondo, existen casos donde lo ganado por estos casos sirve para aumentar el promedio final del recall. Por otra parte demostramos que nuestro método de saliencia obtiene resultados más altos en la precisión y sin comprometer tanto el recall como los otros algoritmos. El algoritmo Center Surround da resultados muy parecidos a nuestro algoritmo pero se encuentra levemente por debajo en los resultados de los *scores*, igual es aconsejable debido a su rapidez para ser calculado. Por último el algoritmo de Itti et al. obtiene en muchos casos los valores de precisión más altos pero es poco estable y en promedio muestra los peores resultados debido a que reduce demasiado la región inicial. Cabe destacar que el algoritmo de Felzenszwalb para detectar personas otorga regiones iniciales menos precisas que ground truth, por lo que los resultados no son tan buenos como los de ground truth pero aun así producen una mejora de sobre el 20% en la precisión.

Adicionalmente comparamos el funcionamiento de los algoritmos de saliencia en el algoritmo TMD, en el cual el comportamiento no se mantiene entre el inicializado con ground truth y el inicializado con el detector de personas. Nuevamente los resultados en precisión aumentan, pero en un menor porcentaje, en el mejor de los casos un 18%. Los resultados se mantienen parejos entre nuestro algoritmo y el de Center Surround. Pero a diferencia del algoritmo anterior, el método de Itti, no logra mejorar en ningún caso el algoritmo de seguimiento, e incluso utilizando el detector de personas, no logra inicializar nunca el algoritmo de seguimiento debido a un problema de éste con regiones muy pequeñas. Por ultimo, a nuestro algoritmo le sucede algo parecido en el caso de

Felzenszwalb, y funciona correctamente solo en valores altos de porcentaje, provocando que el algoritmo Center Surround sea el mejor en la mayoría de los casos.

### 5.1.2. Automatización de parámetros

Otro método que desarrollamos fue el de automatización para la elección del parámetro de porcentaje de información de la ventana. Debido a que uno puede estar interesado en diferentes resultados (mejorar la precisión, mejorar el recall o mejorar ambos) creamos una red bayesiana la cual pudiese ser fácilmente adaptable para alguna de estas metas. De ello obtenemos que los resultados aumentan incluso más de lo que habíamos obtenidos para un valor fijo, por lo tanto la adaptabilidad del parámetro quedó demostrada. Si reenforzamos la precisión obtuvimos valores del 90%, lo que significa un aumento del 40%. Otro caso es el de recall que solo aumenta en un 2% si se prioriza, y un 11% para los *scores* si es que son ellos los que se priorizan. Pero en todo caso, siempre existe un aumento de el parámetro priorizado.

Otro punto importante de destacar, es que se otorgó a la red bayesiana la opción de no utilizar el algoritmo de saliencia para modificar la ventana inicial, lo que nos da aun mayor control sobre el tamaño de la ventana. El recall es la métrica que se ve más beneficiada por esta particularidad de la red bayesiana.

Por último cabe de destacar que esta red bayesiana dio mejores resultados en el algoritmo ONBNN que en el de TMD ya que en el primer algoritmo todas las métricas pudieron ser mejoradas, mientras que en el segundo el recall no pudo ser mejorado lo que provoco también una disminución en los *scores*.

## 5.2. Temas de Investigación Futura

Actualmente gracias a la masificación de hardwares para cálculo de imágenes de disparidad (Shotton et al., 2011), se ve una gran oportunidad para poder integrarlo de

manera eficaz a nuestro sistema de saliencia. Considerando la profundidad como una nueva característica para el descriptor de covarianza podemos alterar nuestro algoritmo de saliencia, y así succesivamente mientras que vayan apareciendo nuevas características que ayuden más a contrastar un objeto del fondo de la imagen. Adicionalmente, sería interesante explorar la posibilidad de crear un sistema de saliencia estéreo, que utilice la información de dos cámaras, para detectar saliencias utilizando solo un descriptor.

Por otro lado, algo rescatable que tiene el algoritmo de saliencia sensible al contexto (Goferman et al., 2010) es la utilización de detectores de objetos para modificar la saliencia. Por ejemplo en nuestro caso se podría utilizar un detector de caras (Viola & Jones, 2001), el mismo detector de personas que utilizamos (Felzenszwalb et al., 2010) o un detector de carros de supermercados.

Pero finalmente lo primordial es mejorar la velocidad de cálculo de nuestro algoritmo de saliencia y del algoritmo de seguimiento ONBNN, dado los buenos resultados obtenidos en este trabajo. La utilización de GPU resulta idónea dado la alta paralelización de estos algoritmos. Otro punto importante es el aumento de la base de datos para que la red bayesiana pueda responder más apropiadamente, o encontrar una manera de actualizarla automáticamente dado los resultados obtenidos en vídeos de testing.

## References


Achanta, R., Estrada, F., Wils, P., & Süsstrunk, S. (2008). Salient region detection and segmentation. *Computer Vision Systems*, 66–75.

Ayache, A., Fillard, P., Pennec, X., & Nicholas, A. (2007). Geometric means in a novel vector space structure on symmetric positive-definite matrices. *Society*, *29*(1), 328–347.

Babenko, B., Yang, M., & Belongie, S. (2009). Visual tracking with online multiple instance learning. In *Ieee computer society conference on computer vision and pattern recognition workshops, 2009. cvpr workshops 2009* (pp. 983–990).

Bhargava, M., Chen, C.-C., Ryoo, M. S., & Aggarwal, J. K. (2009, January). Detection of object abandonment using temporal logic. *Machine Vision and Applications*, *20*(5), 271–281.

Carrasco, M., & Mery, D. (2010, March). Automatic multiple view inspection using geometrical tracking and feature analysis in aluminum wheels. *Machine Vision and Applications*, *22*(1), 157–170.

Collins, R. T., Liu, Y., & Leordeanu, M. (2005, October). Online selection of discriminative tracking features. *IEEE transactions on pattern analysis and machine intelligence*, *27*(10), 1631–43.

Cortez, P., Mery, D., & Sucar, L. (2010). Object Tracking Based on Covariance Descriptors and On-Line Naive Bayes Nearest Neighbor Classifier. *2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, 139–144.

Cortez, P., Undurraga, C., Mery, D., & Soto, A. (2009). Performance evaluation of the Covariance descriptor for target detection. 133–141.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010, September). Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, *32*(9), 1627–45.

Fossati, A., Schönmann, P., & Fua, P. (2010, January). Real-time vehicle tracking for driving assistance. *Machine Vision and Applications*, *22*(2), 439–448.

Goferman, S., Zelnik-Manor, L., & Tal, A. (2010). Context-aware saliency detection. *Proc. CVPR*.

Grabner, H., Grabner, M., & Bischof, H. (2006). Real-time tracking via on-line boosting. *Proc. BMVC*.

Harris, C., & Stephens, M. (1988). A combined edge and corner detector. *4th Alvey Vision Conference*.

Heckerman, D., Geiger, D., & Chickering, D. (1995). Learning Bayesian Networks: The Combination of Knowledge ans Statistical Data. *Machine learning*, *20*(3), 197–243.

Itti, L., Koch, C., & Niebur, E. (2002, March). A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *Colorectal disease : the official journal of the Association of Coloproctology of Great Britain and Ireland*, *4*(2), 147–149.

Jugessur, D., & Dudek, G. (2000). Local appearance for robust object recognition. *cvpr*.

Kalal, Z., Matas, J., & Mikolajczyk, K. (2009, September). Online learning of robust object detectors during unstable tracking. *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 1417–1424.

Lowe, D. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1150–1157 vol.2.

Možina, M., Tomaževič, D., Pernuš, F., & Likar, B. (2009, August). Real-time image segmentation for visual inspection of pharmaceutical tablets. *Machine Vision and Applications*, *22*(1), 145–156.

Nowak, E., Jurie, F., & Triggs, B. (2006). Sampling strategies for bag-of-features image classification. *ECCV*, 490–503.

Porikli, F., & Tuzel, O. (2006, October). Fast Construction of Covariance Matrices for Arbitrary Size Image Windows. *2006 International Conference on Image Processing*, 1581–1584.

Rosin, P. L. (2009, november). A simple method for detecting salient regions. *Pattern Recognition*, *42*(11), 2363–2371.

Ruta, A., Porikli, F., Watanabe, S., & Li, Y. (2009, December). In-vehicle camera traffic sign detection and recognition. *Machine Vision and Applications*, *22*(2), 359–375.

Scholl, B. J. (2001, June). Objects and attention: the state of the art. *Cognition*, *80*(1-2), 1–46.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., et al. (2011). Real-time human pose recognition in parts from single depth images. In *In cvpr.*

Tuzel, O., Porikli, F., & Meer, P. (2006). Region covariance: A fast descriptor for detection and classification. *ECCV*, 589–600.

Viola, P., & Jones, M. (2001). Rapid Object Detection Using a Boosted Cascade of Simple Features. *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, *01*, vol1pp511–518.

Yilmaz, A., Javed, O., & Shah, M. (2006, December). Object tracking. *ACM Computing Surveys*, *38*(4), 13–es.

Yuan, F. (2010, June). An integrated fire detection and suppression system based on widely available video surveillance. *Machine Vision and Applications*, *21*(6), 941–948.

# ANEXO A. RECURSOS ADICIONALES