UNIVERSIDAD AUTÓNOMA METROPOLITANA
UNIDAD IZTAPALAPA División de Ciencias Básicas e ingeniería
Casa abierta al tiempo

CONTEO DE PERSONAS MEDIANTE VIDEOCÁMARAS

Tesis para obtener el grado de:

Maestra en Ciencias (Matemáticas Aplicadas e Industriales)

Por:

Jessica Teresa Rojas Cuevas

Asesor:

Dr. Mario Gerardo Medina Valdez

Jurado calificador:

Presidente: Dr. Boris Escalante Ramírez

Secretario: Dr. John Goddard Close

Vocal: Dr. Mario Gerardo Medina Valdez

10 de abril de 2013

UNIVERSIDAD AUTÓNOMA METROPOLITANA
UNIDAD IZTAPALAPA División de Ciencias Básicas e ingeniería
Casa abierta al tiempo

CONTEO DE PERSONAS MEDIANTE VIDEOCÁMARAS

Tesis para obtener el grado de:

Maestra en Ciencias (Matemáticas Aplicadas e Industriales)

Por:

Jessica Teresa Rojas Cuevas

Asesor:

Dr. Mario Gerardo Medina Valdez

Jurado calificador:

Presidente: Dr. Boris Escalante Ramírez

Secretario: Dr. John Goddard Close

Vocal: Dr. Mario Gerardo Medina Valdez

10 de abril de 2013

AGRADECIMIENTOS

Leonardo, por todos los momentos de adversidad a mi lado, te agradezco a ti por tus palabras, comprensión y ánimo, y sobre todo, agradezco tu incondicionalidad...

Mamá, papá, hermano, los tengo en mi mente y corazón todo el tiempo. Mis logros siempre los dedico a ustedes.

Agradezco al Dr. Mario Medina, por haber forjado una bonita amistad, por su apoyo, ánimo y confianza en mí para lograr este camino.

A mis amigos, que siempre me dieron ánimo y confianza para continuar y por los momentos felices que juntos recorrimos en este camino.

A la Dra. María Luisa Sandoval, por sus consejos, por apoyarme en situaciones difíciles y por escucharme siempre que así lo necesité.

Agradezco al Dr. John Goddard Close y al Dr. Boris Escalante Ramírez por hacerme el honor de formar parte de mi jurado, así como las observaciones que me permitieron mejorar el presente proyecto.

Agradezco al Consejo Nacional de Ciencia y Tecnología por el apoyo económico brindado para la realización del presente proyecto de maestría.

Dios, gracias por darme sabiduría, paciencia y vida para conseguir esta meta.

RESUMEN

El conteo automatizado de personas es una tarea utilizada principalmente en plazas comerciales, lugares muy concurridos y cerrados, pues las aglomeraciones pueden ser difíciles de manejar, por lo que es necesario realizar una estimación del número de personas que hay para actuar de acuerdo a las circunstancias.

Aunque en la actualidad existen sistemas especializados para esta tarea, se requiere de un gasto adicional para la adquisición del equipo, por ello se propone la utilización de cámaras de video, pues en la mayoría de los lugares donde se requiere tener un estimado del número de personas, ya cuenta con cámaras de vigilancia.

En este trabajo se propone un sistema de conteo de personas con videocámaras para videosecuencias con flujo peatonal moderado. En el módulo de detección de personas se ha utilizado el método de detección de rostros de Viola-Jones. Por otro lado, para realizar el conteo de una persona a la salida de la videosecuencia, se utiliza el Filtro de Kalman. Además, se propone un método de asociación de datos para casos de múltiples rostros o problemas de oclusión.

Los resultados son favorables ya que se obtuvo hasta un 92 % de aciertos durante el conteo. En el análisis ROC del seguimiento se obtuvo una precisión de hasta 0.92, una razón de verdaderos positivos de 0.9379 y una razón de falsos positivos de 0.1524.

Este proyecto ha sido presentado en el congreso internacional *First North*

American Meeting On Industrial And Applied Mathematics, Oaxaca, Méx., 2010 y en el *XLIV Congreso Nacional De La Sociedad Matemática Mexicana*, San Luis Potosí, Méx., 2011.

ABSTRACT

Automated counting people processes are being of extensive use in malls and other crowded and closed places. As reliable estimation of the number of persons is an important problem in visual surveillance in order to act depending to different scenarios according to the size of the crowd.

Although currently there are specialized systems for this task, it requires the acquisition of expensive equipment. Therefore, we propose to use a video camera to count, since there are surveillance cameras in the majority of places in which we need to know an estimation for the number of people.

In this work we propose a counting people system by using a video camera in a place where the pedestrian flow is moderate. To perform the detection of a person we use a face detection method due to Viola and Jones. On the other hand, to follow and count a person after been detected we use the Kalman filter, the counting of the person is done at the moment when leaving the video. Additionally, we propose a data association method for the case of multiple faces or occlusion problems.

The results are favorable because we obtain a precision of 92 % for the people counting problem. In the ROC analysis tracking we get a 0.92 of accuracy and 0.9379 for true positives and 0.1594 for false positives rate.

Results for this project have been presented on the international conference

First North American Meeting On Industrial And Applied Mathematics, Oaxaca, Méx., 2010 as well as on the *XLIV Congreso Nacional De La Sociedad Matemática Mexicana*, San Luis Potosí, Méx., 2011.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

CAPÍTULO 1

INTRODUCCIÓN

1.1. Contexto e identificación de la problemática

En muchas ocasiones es necesario considerar y determinar el tránsito peatonal de un determinado lugar. Un ejemplo muy claro es el caso de los supermercados, los cuales exigen el conocimiento aproximado del número de clientes que tienen por día para así poder establecer la estrategia de marketing, horarios y contratación de mano de obra para los días con mayor afluencia.

Otro ejemplo es el conocimiento del número de personas que existe en lugares con abundante concurrencia para así determinar los mejores patrones de seguridad en caso de alguna contingencia.

Existe una gran cantidad de aplicaciones del seguimiento de peatones. Los sistemas de seguimiento específicos para aplicaciones de realidad virtual y aquellas en que la medición del rendimiento deportivo requiere que partes específicas del cuerpo sean rastreadas. En contraste, la supervisión de la seguridad, la detección, conteo de personas, control de tráfico peatonal y aplicaciones de identificación del patrón de flujo de tráfico peatonal enfatizan el seguimiento en un nivel más robusto. En este

caso todos los individuos en una escena pueden ser considerados como unidades únicas indivisibles. Por supuesto, un sistema que puede realizar el seguimiento simultáneo en todas las escalas es altamente deseable, pero hasta ahora, tal sistema no existe (Masoud y Papanikolopoulos [5]).

El conteo de personas hoy en día es una herramienta muy socorrida para diversas aplicaciones, pero todos con la meta de conocer el flujo global de las personas, por lo que ahora no solamente es necesario determinar el número de personas que existen dentro de un lugar si no que también es importante conocer su dirección.

1.2. Algunos sistemas existentes

Existen diferentes aplicaciones comerciales para el conteo de personas, los cuales aún tienen desventajas grandes frente al costoso software especializado para el conteo de personas, dado que, por ser este especializado, requiere de dispositivos propiamente costosos y exclusivo solamente para su fin, es decir, requieren de un espacio y sus aplicaciones son limitadas.

Es posible obtener una estimación del tráfico peatonal con los diversos métodos existentes para el conteo de personas, sin utilizar un software costoso, tal es el caso del contador mecánico que consiste en torniquetes y puertas giratorias colocados en las entradas de las instalaciones, sin embargo, este sistema tiene el problema de no permitir el flujo rápido en las entradas cuando existe mucha afluencia. Otro caso es el contador térmico, el cual tiene un sensor que identifica el gradiente entre la temperatura ambiental y la temperatura corporal, este tipo de contador es ineficiente cuando la temperatura ambiental es semejante a la corporal. El contador manual permite que una persona detecte el flujo peatonal de manera manual, con la evidente desventaja que una persona tiene cuando existen grandes acumulaciones de personas, además de que el contador manual usualmente tiene sólo 4 dígitos.

Otra técnica utilizada es el contador mediante rayos infrarrojos, es decir, un dispositivo infrarrojo se comunica con un receptor de extremo a extremo, cuando una persona pasa, la comunicación es intervenida y en este caso el contador se activa

y agrega una persona a la cuenta. El problema que existe con este mecanismo es la denominada *oclusión*[1]. El problema de oclusión es el generado debido al amotinamiento de las personas, es decir, al cruzar un grupo de personas al mismo tiempo, el contador sólo detecta una.

Actualmente existen estudios acerca del conteo de personas mediante videocámaras.

Sin lugar a dudas con el auge que ha tomado la tecnología en cuanto a imágenes y video, ha sido posible la adquisición de cámaras a muy bajo costo, el cual permite tener un almacenamiento confiable de los sucesos acontecidos durante el día con lo que no sólo puede ser utilizado con fines de seguridad, si no también el conteo de personas.

En este trabajo se implementa un algoritmo en el ambiente Matlab que permite el conteo de personas mediante el uso de cámaras de video. En primera instancia se realiza la detección de personas, para posteriormente continuar con su seguimiento y finalmente concluir con el conteo de personas.

1.3. Objetivos

Objetivo general:

Desarrollar un algoritmo para el conteo de personas mediante el uso de videocámaras en lugares con moderada afluencia peatonal y con iluminación constante.

Objetivos particulares.

- Detección de personas.
- Evitar el problema de oclusión en el proceso de seguimiento y conteo de personas mediante el uso de videocámaras.

[1]Oclusión: Se le llama así al hecho de tener 2 o más objetos muy cercanos entre sí de tal forma que alguno de los objetos no es identificado debido a que el sensor no lo detecta total o parcialmente.

- Proponer un algoritmo que permita segmentar, detectar, seguir y contar personas en videos.

1.4. Motivación

Existen diversos sistemas de conteo de personas que, aunque resultan efectivos, son realmente costosos, sin mencionar el espacio que pueden ocupar y los recursos utilizados y restringidos únicamente para ellos.

Actualmente en la mayoría de los lugares de gran concurrencia, se poseen sistemas de seguridad, como el uso de videocámaras de vigilancia. Si además de utilizar las videocámaras con las que ya se cuenta en ese tipo de lugares para vigilancia, pudieran ocuparse para el seguimiento y conteo de personas, sería realmente útil y ahorraría una gran cantidad de dinero en instalaciones como supermercados, cines o el sistema de transporte colectivo *metro*.

Hoy en día la investigación enfocada al procesamiento de imágenes está avanzada y ha llegado a ser muy especializada. Por ejemplo, es posible el seguimiento y detección de partes específicas del cuerpo para poder asegurar que se trata de una persona. Mediante estos avances en la ciencia pueden utilizarse diversos mecanismos de reconocimiento de formas para seguir y contar personas.

Existe una poderosa herramienta matemática que permite la lectura, manejo y procesamiento de imágenes y video, el ambiente Matlab. Este ambiente es uno de los más conocidos y utilizados, por lo que entender comandos, instrucciones y archivos.m no es una tarea difícil para quien desee comprender a fondo el algoritmo realizado en este trabajo.

1.5. Funcionamiento global del sistema

El sistema consiste en tres distintos módulos:

- Detección de personas
- Seguimiento de personas
- Conteo de personas

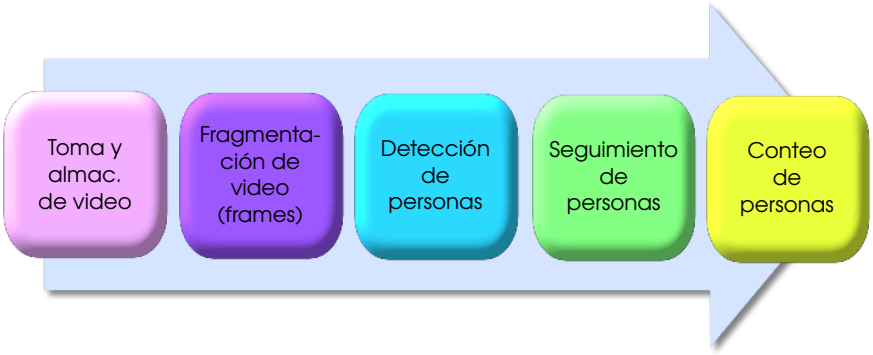

Figura 1.1: Diagrama del funcionamiento del sistema

En la Figura 1.1 podemos observar como se pretende implementar al sistema propuesto. El primer módulo (Detección de personas) es realizado mediante detección frontal de rostros mediante el método de Viola Jones (ver Capítulo 2). En el caso del segundo y tercer módulo, Seguimiento y Conteo de personas, se realiza uno como consecuencia de otro, es decir, el conteo es el resultado del seguimiento de rostros en las fronteras de cada fotograma. Este procedimiento es realizado mediante el filtro discreto de Kalman (Capítulo 3).

1.6. Trabajos Relacionados

Hoy en día el conteo de personas mediante el uso de videocámaras ha tomado un gran impulso en el ámbito de la investigación. El conteo de personas es un tema realmente redituable pues el software y hardware para este fin es realmente costoso, por lo que generar un nuevo mecanismo a bajo costo es un tema de interés.

Existen problemas de seguimiento de personas en lugares donde la multitud genera oclusión total o parcial de algún individuo. En trabajos como Yang et al. [6] y Zhao y Nevatia [7] se toca precisamente este tipo de problema. Por un lado Yang et al. [6] establecen que es posible trabajar con multitudes si se colocan dispositivos que cubran todos los ángulos posibles, es decir, se consideran varias videocámaras colocadas estratégicamente. Obviamente este proceso tardaría gran cantidad de tiempo si se utiliza videocámaras comunes, pues procesar imágenes de 8 cámaras de video en tiempo real es una labor casi imposible computacionalmente hablando.

Es por eso que en lugar de utilizar videocámaras utiliza cámaras con sensores de imágenes (8 sensores de imágenes) colocados en el área de conteo, las cuales omiten el segundo plano y solamente trabajan con las siluetas de las personas. Este trabajo resuelve el problema de las multitudes, pero con la desventaja de tener que adquirir 8 sensores de imágenes para el único y exclusivo fin de conteo de personas. Otra de las cosas es que trabajan bajo el supuesto de un segundo plano totalmente fijo, es decir, este trabajo no funciona en el caso de tener escaleras eléctricas, puertas en movimiento o algún elemento móvil en el segundo plano. El trabajo es realmente rescatable considerando el hecho de tener al menos 2 videocámaras en lugar de sensores de imágenes y trabajar únicamente con la silueta de las personas para así poder contar personas en tiempo real sin tener que pasar por el procesamiento de imágenes.

Por otro lado, Zhao y Nevatia [7] han trabajado con secuencias de video para la segmentación de humanos utilizando una sola cámara fija. La técnica es mediante el procesamiento de imágenes utilizando un modelo estocástico para el reconocimiento de personas. Ellos realizan una caracterización de acuerdo a la densidad de concurrencia, siendo la clasificación de mayor énfasis aquella en la cual hay aglomeraciones muy abundantes de personas. El problema se reduce a una solución mediante cadenas de Markov Monte Carlo. En este caso, la detección de humanos se basa principalmente en detectar la cabeza de una persona, como esto no es suficiente, también puede auxiliarse con el torso y las piernas, realizando una primera aproximación mediante elipsoides. Este algoritmo nos ayuda a tener una detección de humanos bastante precisa, es conveniente para lugares donde la concurrencia es abundante, sin embargo no se ha hablado nada acerca del conteo en tiempo real.

En el caso de Masoud y Papanikolopoulos [5], el sistema que proponen es realmente completo y puede ser utilizado incluso en escenarios con movimiento ya que considera escenarios en donde el viento podría mover las ramas de algún árbol cercano. La propuesta de estos autores es muy interesante y resuelve el problema de oclusión en tiempo real con una sola cámara. A diferencia de Cai y Aggarwal [8] y Yang et al. [6] sólo se necesita de un dispositivo para poder asegurar el conteo de personas evitando la oclusión. La forma en que se plantea en este trabajo es reducir la imagen a una imagen que contiene solamente *blobs* eliminando a priori

el segundo plano, de tal forma que sólo se tengan imágenes de siluetas, tal que la imagen anterior y la imagen actual deben ser semejantes en posición y tamaño, bajo dos suposiciones, la primera es que no se pueden dividir dos blobs y separan otros dos simultáneamente al pasar de una imagen a otra; la segunda es que el tamaño de el rectángulo que encierra a los blobs debe tener al menos la mitad del tamaño del rectángulo más pequeño que encierra al blob más pequeño. El mecanismo utilizado para el reconocimiento de peatones no requiere un procesamiento y reconocimiento estricto de la imagen de alguna parte del cuerpo, por lo que el procesamiento en tiempo real es posible. La desventaja que tiene este sistema es que sin importar que sean peatones, algún animal o un vehículo, si cubre las restricciones de tamaño se cuenta como persona. Este sistema es muy efectivo para lugares donde la concurrencia es abundante pero es lo suficientemente espaciada para que los blobs puedan ser reconocidos como un individuo.

Un algoritmo mucho más sencillo pero basado en la misma idea de Masoud y Papanikolopoulos [5] es el propuesto por H. et al. [9]. Este método utiliza una sola cámara colocada de manera estratégica de tal forma de que esta queda en un pequeño pasillo y captura imágenes desde arriba de las personas. El área de trabajo se divide en 3 partes para que al pasar de una parte a otra se tenga la velocidad de la persona, este pasillo sólo puede tener flujo peatonal bidireccional, además de que por la posición de la cámara no se pueden percibir imágenes de personas lejanas a menos que la persona sea muy baja de estatura, en cuyo caso la variación entre los tamaños de siluetas es mínima, por lo que se puede determinar un área media (A_0) que ocuparía una persona en la imagen. De esta forma, se elimina el background y posteriormente se segmenta la imagen, de modo que sólo se tenga que trabajar con blobs. Si el área del blob está en $[0.6A_0,\ 1.4A_0)$ la imagen es reconocida como una persona, si el área del blob es mayor que $1.4A_0$, se consideran como dos personas. Existen criterios para determinar el número de personas en base al área media del blob de una persona en la imagen (A_0), con lo que se elimina el problema de oclusión. Como puede observarse este método es muy sencillo por lo que el procesamiento en tiempo real con una sola cámara es posible. Las desventajas en este método son dos: posicionamiento a priori de una cámara en un lugar donde el flujo es bidireccional y moderado y la segunda desventaja es que cualquier objeto o animal que transite por ahí será considerado como persona si cumple con las expectativas de área. Pero las

ventajas para este procedimiento son muchas, la principal es la sencillez y rapidez del algoritmo.

De manera muy parecida es el modo de trabajar de Kim et al. [10]. Su trabajo consiste en colocar la cámara de manera estratégica, este sistema trabaja con el background pues está diseñado para condiciones en los exteriores (donde la intensidad de luz, temperatura, viento, entre otras pueden variar), por lo que la sustracción del background no resulta de una simple sustracción con respecto a la imagen anterior. El sistema también trabaja únicamente con blobs pero a diferencia de H. et al. [9], el conteo lo hacen simultáneamente con el seguimiento de personas, es decir, tratan de predecir el comportamiento y dirección de la persona.

El trabajo realizado en Zhao et al. [11] es un trabajo mucho más preciso y que no requiere un nivel de complejidad como en Zhao y Nevatia [7], más sin embargo puede asegurar la detección, seguimiento y conteo de personas pues realiza una detección de rostros y un seguimiento de personas mediante un filtro de Kalman. Este sistema está diseñado para lugares cerrados con moderada concentración de personas con una cámara ubicada de manera estratégica, por lo que problemas como la falta o cambio drástico de condiciones naturales como luz, temperatura, viento no son problema, pues es un lugar estable. Este sistema resuelve el problema de la oclusión mediante el seguimiento de rostros y además resuelve el problema de decidir si las detecciones que aparecen en las imágenes necesariamente son personas. El conteo de personas se realiza mediante el análisis de las trayectorias de los peatones.

A continuación se tiene una tabla comparativa en la cual se pueden observar las ventajas más importantes que ofrece cada uno de los trabajos previos.

De la Tabla 1.1 puede observarse un panorama general de la tecnología existente en cuanto a conteo de personas se refiere. En ella se puede observar que los sistemas estudiados tienen la principal tarea de cubrir el problema de oclusión, por lo que todos lo abordan. Los trabajos de conteo de personas en tiempo real atacan el tema utilizando un método de bajo nivel para detección de personas (ver Capítulo 2), por lo que permiten el rápido procesamiento de los fotogramas, sin embargo, todo aquel elemento fuera del *background* es considerado como una persona por lo que se pierde precisión, además de que son susceptibles a los cambios de iluminación y de ambiente. Por otro lado la detección de alguna parte del cuerpo es una opción que aunque aumenta la complejidad del sistema permite discernir mejor la presencia de

Tabla 1.1: Comparación entre los distintos métodos existentes

Trabajo relacionado	**Resuelve Oclusión**	**Tiempo Real**	**Complejidad**	**Multicámara**	**Detección**
Kim et al. [10]	√	√	Regular	No	Blobs
Masoud y Papanikolopoulos [5]	√	√	Regular	No	Blobs
H. et al. [9]	√	√	Baja	No	Blobs
Yang et al. [6]	√	√	Regular	Si	Blobs
Zhao y Nevatia [7]	√	No	Alta	No	Cabezas
Zhao et al. [11]	√	No	Regular	No	Rostros

peatones en la escena.

Dado que se requiere de un sistema capaz de realizar el conteo de la mejor forma posible, en este trabajo se presenta un sistema basado en Zhao et al. [11] cuyas características, en consecuencia, son similares a las mostradas en la Tabla 1.1.

CAPÍTULO 2

DETECCIÓN DE PERSONAS

En trabajos relacionados puede observarse que la segmentación y detección de personas es un elemento primordial y resulta la primera parte a realizar por el algoritmo que se desea proponer, ya que una vez hecha la detección de personas es posible realzar el seguimiento y finalmente el conteo de esta persona.

Se han comentado algunos de los diversos métodos de detección de personas, y se observó además que todos los métodos son mecanismos buenos, que tienen muchas ventajas y realmente pocas desventajas. Discernir entre que método es el mejor no es una tarea fácil. El criterio para tomar una elección no puede ser simplemente uno pues existen muchas ventajas en, por ejemplo, trabajar con simples siluetas, pero también existen ventajas en trabajar con el cuerpo humano completo o con el rostro de las personas. Estos criterios se elegen de acuerdo a la aplicación específica que se le desea dar al sistema propuesto.

En este caso, se desea trabajar con un sistema que será utilizado en un ambiente estable, esto es, que la temperatura, el viento y la luz sea estable, como ocurre en lugares cerrados (bancos, supermercados, estaciones de autobús, etc.), sin embargo se debe considerar que posiblemente exista un *background* o segundo plano móvil, pues puede existir en este tipo de lugares puertas que se abren y cierran, escaleras eléctricas, *etc*, por lo que es difícil que, al trabajar únicamente con siluetas, no se tenga un "falso positivo"[1], es por ello que se desea tener ya

[1]Falso positivo: declarar que una región de una imagen es una detección cuando no lo es.

sea una buena aproximación del *background* no estático o una buena detección de personas, mediante detección completa o de alguna parte específica del cuerpo humano. Ambas propuestas son viables y anteriormente estudiadas en trabajos relacionados. Considerando que al final el *background* del sistema no es algo que sea de interés en el conteo de personas, hacer una detección completa o de alguna parte del cuerpo es una manera para resolver el problema de detección y conteo específicamente de personas. Con ello, una vez identificado el cuerpo humano, el *background* no estable del sistema ya no es un problema para el conteo de personas.

La opción más natural para la detección de personas dados los criterios ya establecidos, es la detección de rostros. Cabe mencionar por lo tanto, que si se hace de esta manera, el conteo de personas asegura que realmente se tengan humanos en el registro de personas, unque el conteo es realizado de manera unidireccional, es decir, el sistema no puede realizar la detección de personas si la posición no es de frente a la cámara.

2.1. Métodos de detección de rostros

Una vez establecida la manera en la cual se realizará la detección de personas, es importante saber acerca de las distintas técnicas existentes para la detección de rostros. En Hjelmȧs y Low [1], Yang et al. [12] pueden observarse distintos métodos existentes de detección de rostros.

Las técnicas para la detección de rostros según Hjelmȧs y Low [1] están dadas en la Figura 2.1.

Por otro lado, las técnicas de detección de rostros de acuerdo con Yang et al. [12] se clasifican en 4 categorías:

1. **Métodos basados en conocimientos:** Métodos basados en reglas que codifican el conocimiento humano de lo que constituye una cara típica. Por lo general, las reglas capturan la relación entre características faciales. Estos métodos están diseñados principalmente para localización de rostros.

2. **Enfoques de características invariantes:** Estos algoritmos tienen el objetivo de encontrar características estructurales que existen incluso cuando

Falso negativo: es no detectar una región de la imagen en la que sí existe una detección.

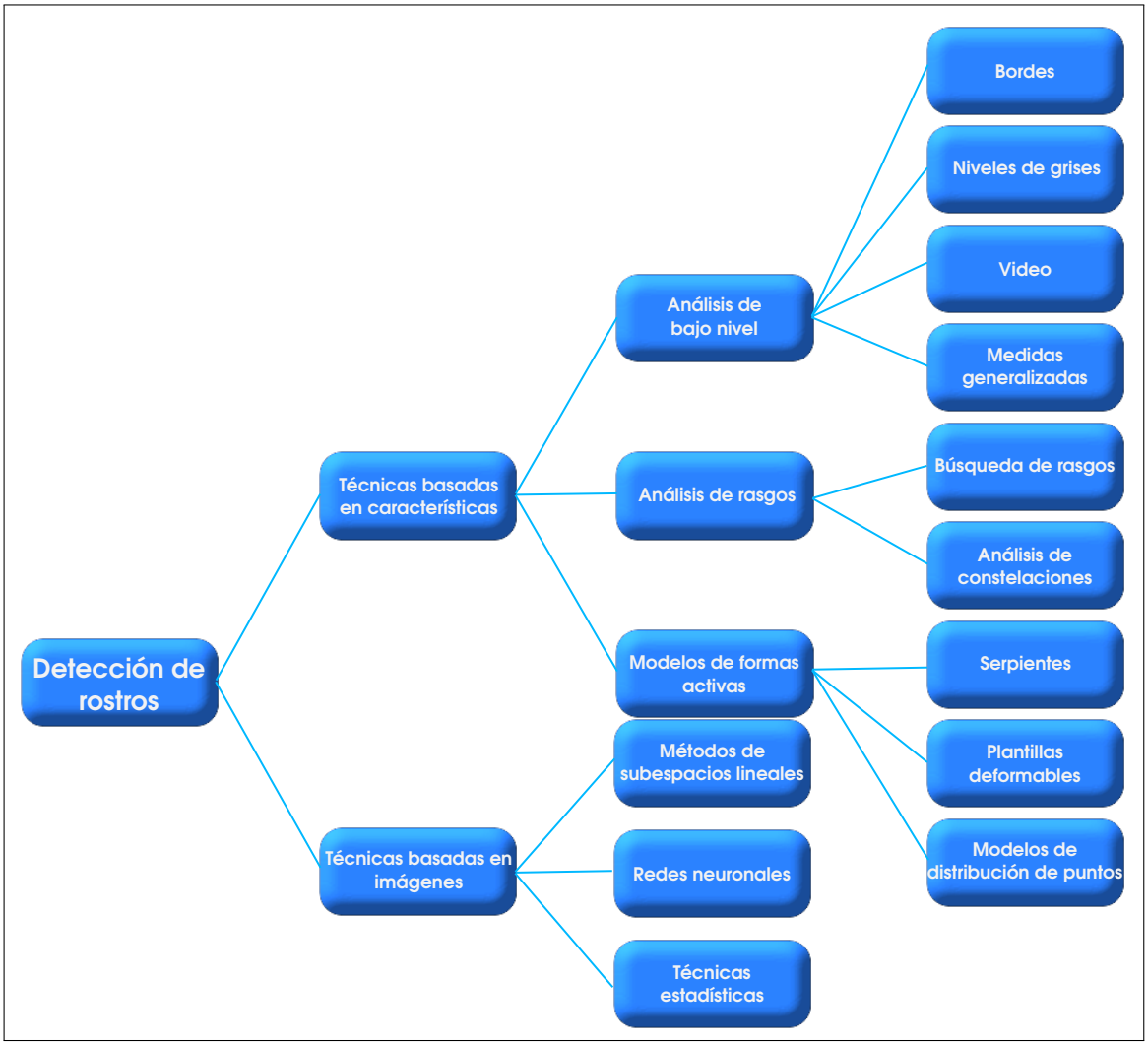

Figura 2.1: Técnicas existentes para la detección de rostros. Hjelmȧs y Low [1]

la pose, el enfoque o las condiciones de iluminación varían (color de piel, rasgos faciales o texturas). Estos métodos están diseñados principalmente para localización de rostros.

3. **Métodos de correspondencia de plantillas:** Varios patrones estándar de la cara se almacenan para describir la cara como un todo o como características faciales separadamente. Para la detección se calculan las correlaciones o algún tipo de distancias entre la imagen de entrada y los patrones almacenados. Estos métodos se utilizan tanto para la detección como localización de rostros.

4. **Métodos basados en apariencia:** Estos métodos son entrenados a partir de un conjunto de imágenes que captura la variabilidad representativa de la apariencia facial. Estas plantillas aprendidas son utilizadas para la detección. Se basan en técnicas de análisis estadístico o algoritmos de aprendizaje. Estos métodos se utilizan tanto para la detección como localización de rostros.

Hjelmàs y Low [1] tiene una visión más amplia y detallada de los métodos de detección de rostros. Las secciones siguientes serán descritas basadas en dicha referencia.

2.1.1. Análisis de bajo nivel

La segmentación en imágenes no triviales es una de las tareas más difíciles en el procesamiento de imágenes (González y Woods [13]).

La idea principal del proceso de segmentación de una imagen es reducir y/o eliminar aquella información que puede ser poco relevante para poder manejar solamente aquella que pueda tener una mayor utilidad en la aplicación deseada. Como es lógico suponer en una imagen en esas condiciones es mucho mas fácil encontrar y distinguir características estructurales. En general, el proceso de segmentación de imágenes consiste en binarizar la imagen de tal forma que se obtengan 2 regiones o bien para cada color en RGB hacer este mismo proceso para cada dimensión. Existen dos formas principales para llevar a cabo este proceso, una de ellas es mediante la umbralización y la segunda es mediante la detección de bordes. En visión computacional el trabajo con imágenes binarias es muy importante ya sea para realizar segmentación por intensidad de la imagen, para generar algoritmos de reconstrucción o reconocer estructuras. La forma mas común de generar imágenes binarias es mediante la utilización del valor umbral de una imagen a escala de grises, para ello se elige un valor limite (o bien un intervalo) a partir del cual todos los valores de intensidades mayores serán codificados como uno mientras que los que estén por debajo de este valor umbral serán codificados como cero.

En visión computacional es de utilidad para hacer reconocimiento de objetos o bien para segmentar regiones, extraer los bordes de objetos (que en teoría delimitan sus tamaños y regiones) Jimenez y Navarro [14].

Bordes

La detección de bordes es una técnica que nos permite delinear el contorno de la cara y así poder reconocer ciertas características del faciales. Por ejemplo, en Craw et al. [15] se plantea la extracción de bordes para poder reconocer y trazar el esquema de la cabeza. Puede ser utilizado para detectar ojos, boca y nariz a partir del delineado del contorno de la cara.

Existen varios métodos de detección de bordes, esquinas u otros puntos de interés. Para detectar un borde en una imagen es necesaria la definición de derivada, es decir, un borde existe cuando hay un salto grande de un nivel en la escala de grises o de color, por lo que el cambio es evaluado mediante la primera o la segunda derivada. Debido a que se está trabajando con pixeles ordenados en forma de una matriz, la derivada debe ser considerada en una aproximación discreta, por lo que existen distintos métodos de aproximación para la primer derivada:

- Gradiente de una imagen: El gradiente de una imagen $f(x, y)$ en un punto (x, y) se define como un vector bidimensional dado por la siguiente ecuación, siendo este un vector perpendicular al borde:

$$\mathbf{G}[f(x,y)] = \nabla f(x,y) = \begin{bmatrix} f_x \\ f_y \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} f(x,y) \\ \frac{\partial}{\partial y} f(x,y) \end{bmatrix} \tag{2.1}$$

  donde el vector G apunta en la dirección de variación máxima de f en el punto (x, y) por unidad de distancia, donde su magnitud y dirección están dadas por:

$$|\mathbf{G}| = \sqrt{f_x^2 + f_y^2} \tag{2.2}$$

$$\phi(x,y) = tan^{-1}\frac{f_x}{f_y} \tag{2.3}$$

  respectivamente.

  Es una práctica habitual tomar la magnitud del gradiente usando la métrica $|(u, v)| = |u| + |v|$. De esta forma:

$$|\mathbf{G}| = |f_x| + |f_y| \tag{2.4}$$

Para el caso de la derivada en (2.1) es posible utilizar las diferencias de primer orden entre 2 pixeles adyacentes, definidas por:

$$|f_x| = \frac{f(x+\triangle x, y) - f(x-\triangle x), y}{2\triangle x} \tag{2.5}$$

$$|f_y| = \frac{f(x, y+\triangle y) - f(x, y-\triangle y)}{2\triangle y} \tag{2.6}$$

Si ahora definimos a $g(x, y)$ como la imagen binarizada en relación a los bordes detectados:

$$g(x,y) = \begin{cases} 1 \; si \; \mathbf{G}[f(x,y)] > T \\ 0 \; si \; \mathbf{G}[f(x,y)] \leq T \end{cases} \tag{2.7}$$

donde T es un valor umbral no negativo elegido convenientemente.

- Operadores de Sobel

  Tanto el operador gradiente como el operador de Sobel son operadores que pueden expresarse mediante convolución discreta en la cual es necesario la utilización de máscaras, que en este caso son matrices que otorgan un cierto peso a la función $f(x, y)$.

  En el caso del operador de Sobel, f_x, f_y pueden expresarse mediante las siguientes ecuaciones:

$$f_x = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \tag{2.8}$$

$$f_y = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \tag{2.9}$$

  donde los distintos valores están distribuidos de acuerdo a la región de la imagen que se esté tratando, de dimensión 3×3:

$$\begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix} \tag{2.10}$$

y las máscaras para el cálculo de f_x y f_y están dadas por:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.11}$$

y

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.12}$$

respectivamente.

Cada uno de los siguientes operadores pueden calcularse mediante el uso de máscaras.

- Operador de Prewitt

  Este operador es muy similar al de Sobel, donde las máscaras a utilizar están dadas por:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.13}$$

  para f_x, y

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.14}$$

  para f_y.

- Operador de Roberts

  Este operador marca solamente los puntos de borde, sin informar nada acerca de la orientación de estos. Es un operador simple que trabaja muy bien en imágenes binarias. La máscara es de la forma:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{2.15}$$

- Máscaras de Kirsch

  Se denominan también de brújula por que se definen considerando una máscara simple y rotándola en las 8 direcciones principales de la brújula. El valor del módulo del gradiente resulta ser el máximo de los 8 valores mientras que la dirección queda determinada por el ángulo asociado a la máscara que ha generado dicho valor máximo. La máscara de Krisch es de la forma:

$$\begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \tag{2.16}$$

- Máscaras de Robinson

  Las máscaras de Robinson se usan de manera muy similar a las máscaras de Kirsch. La máscara que deberá ser rotada es como sigue:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.17}$$

  Nótese que las máscaras de Robinson son simétricas con respecto a una línea central de ceros en la horizontal, vertical y diagonales. Se pueden calcular cuatro filtros y los restantes son el negativo de los anteriores, por lo que son menos costosos computacionalmente que las máscaras de Kirsch.

- Máscaras de Frei-Chen

  Las máscaras de Frei-Chen están dados por un conjunto de 9 máscaras, estas máscaras forman un conjunto completo de vectores base. Esto significa que se puede representar cualquier sub-imagen de tamaño 3×3 en términos de cada una de esas imágenes.

- Algoritmo de Canny

  El algoritmo de Canny se fundamenta en la teoría de operadores de la primera derivada y resulta particularmente interesante debido a que extrae bordes y cierra contornos evitando posibles rupturas de los mismos durante su extracción. Consiste en obtener el gradiente del Gaussiano aplicado a una

vecindad. Se puede obtener aplicando el filtro Gaussiano (ver parte anexa) y a la imagen resultante aplicarle el gradiente mediante cualquiera de las máscaras mencionadas anteriormente.

Para más detalle sobre cada uno de estos operadores, se puede consultar en Martínez y de la Cruz García [16].

Matlab tiene precargados los métodos de Canny, Sovel, Prewitt y Roberts, mediante la función *edges*:

```
BW = edge(I,'sobel')
```

Donde BW es la imagen binarizada (blanco y negro), I es la matriz donde está contenida la imagen y *Sobel* es el método.

Es posible que en lugar de cualquiera de estos métodos se requiera de otro. Para estos casos pude programarse un archivo.m de tal forma que este contenga la función que genere la máscara adecuada.

Niveles de grises

En este caso se desea encontrar un punto óptimo en la escala de grises, de tal forma que este punto sea el valor umbral para separar la imagen en una imagen binaria, de esta forma se pueden extraer las formas más oscuras de la cara tales como labios, ojos, cabello, cejas e incluso vello facial.

Este proceso consiste en binarizar una imagen separándola en dos regiones, esto se obtiene mediante el histograma de la imagen. Se produce el histograma y se obtienen los dos picos más grandes, el valor que se encuentra entre estos picos es un valor *valle*, que será elegido como el valor del umbral T. Usualmente el valle es elegido por el usuario mediante observación, o bien, realizando una aproximación del histograma a funciones conocidas tomando los puntos mínimos como valores valles T. Así, los valores menores a T se les asigna el valor 0 y los valores mayores al umbral se les asignará el valor 1, con lo que se obtiene una imagen binaria.

Cuando los histogramas obtenidos son demasiado complejos y existe más de una opción de umbral, es difícil tener un criterio de elección para poder umbralizar la imagen.

Selección de umbral óptimo

Para realizar la selección del umbral óptimo dadas L intensidades de gris y una imagen de $M \times N$, se pueden clasificar los MN pixeles en n clases C_i de tal forma que la varianza sea mínima. Esto se resuelve mediante Teoría de desiciones para la selección de clases.

Existen dos métodos que actúan bajo el criterio de elección de la Teoría de Decisiones, es decir, se elige la clase que tenga la mínima varianza.

- **Método de Otsu**. Elige el umbral T correspondiente al nivel de intensidad que proporcione la mínima varianza ponderada. Supone que el histograma se comporta como dos Gaussianas cuya intersección será tomada como el umbral T. Para mayor detalle puede verse la referencia Otsu [17]

- **Método de Ridler-Calvard**. Se trata de un método iterativo que obtiene el umbral mediante el método del punto medio. Es posible saber más acerca de este algoritmo en Martínez y de la Cruz García [16]

En el ambiente MatLab es posible obtener de manera directa el umbral T de manera directa con un comando precargado:

```
level = graythresh(I)
```

donde I es la matriz asociada a la imagen en escala de grises y *level* es el valor de T, este calor es calculado mediante el método de Otsu.

Es posible binarizar por regiones de manera manual, conociendo de antemano el valor de T mediante:

```
BW = im2bw(I,T)
```

Color

Si bien la información en escala de grises nos proporciona la representación de la imagen basada en características, el color es un medio más poderoso de la apariencia del objeto. Debido a las dimensiones extra que tiene el color, dos formas similares entre sí representados en escala de grises podrían ser muy diferentes en un espacio de color. Los diferentes colores de piel humana dan lugar a un grupo compacto en el espacio de colores, incluso cuando se consideran diferentes razas.

Uno de los modelos más utilizados para la representación de color es el modelo RGB (red, green, blue), de los cuales los colores de una imagen son el resultado de la combinación de los colores rojo, verde y azul. Dado que la variación entre el color de la piel está dado en gran medida por la luminosidad, es preferible en muchas ocasiones utilizar los colores RGB normalizados y así filtrar el efecto de luminosidad. Los colores normalizados se obtienen de la siguiente manera:

$$\begin{aligned} r &= \frac{R}{R+G+B} \\ g &= \frac{G}{R+G+B} \\ b &= \frac{B}{R+G+B} \end{aligned}$$

donde *r+g+b=1*. Por lo tanto, los colores normalizados pueden ser obtenidos por la combinación de *rgb* normalizada y también pueden ser relacionados entre sí, por ejemplo, para el color azul *b=1-r-g*. En el análisis de color de piel, un histograma de r y g muestra la información del color de rostro. Al comparar la información de color de un pixel en el segmento de la cara con los valores de r y g, la probabilidad de que el pixel pertenece a la región de la carne de la cara puede deducirse.

Existen otros espacios de color donde se utiliza la lumnancia (brillo) y crominancia (color). Si se suprime la parte de luminancia, pueden evitarse también problemas de iluminación no constante. El modelo YIQ (In-fase Quadrature) ha sido utilizado en la detección de rostros. La componente Y proporciona la luminancia y las componentes IQ proporcionan la crominancia, I proporciona matices cian y naranja , mientras que Q proporciona matices verde y magenta. Mediante la conversión del modelo *rgb* al YIQ puede averiguarse el valor de I, el cual es muy útil para detectar el color de piel, incluso las asiáticas. La conversión también suprime de manera efectiva el *background* de la imagen, lo que facilita la detección de rostros en ambientes más naturales. Otros modelos de color que también pueden ser utilizados son HSV , YES, YCrCb, YUV, CIE-xyz, $L*a*b*$, $L*u*v*$, CSN y UCS/Farnsworth.

La segmentación en imágenes a color actúa bajo el mismo principio que la segmentación en escala de grises, pero teniendo en cuenta las 3 dimensiones de RGB.

Movimiento (video)

El movimiento se utiliza principalmente para videosecuancias, es decir, si se tiene disponible la videosecuencia, puede hacerse un análisis de regiones mediante la diferencia entre *frames* o fotogramas. Este enfoque sencillo permite identificar un primer plano en movimiento de manera eficiente, independientemente del *background*. Se detectan regiones importantes tales como ojos o partes de la cara y se le da seguimiento con la diferencia de *frames*. Otra manera de detectar el movimiento es detectando el movimiento de bordes utilizando un filtro espacio-temporal. El proceso involucra la convolución de la imagen en escala de grises $I(x, y)$ con el operador temporal de segundo orden de bordes $m(x, y, t)$ el cual está relacionado con el filtro Gaussiano de la siguiente forma:
Sea el filtro Gaussiano

$$G(x, y, t) = \left(\frac{a}{\pi}\right)^{3/2} e^{-a(x^2+y^2+u^2t^2)} \tag{2.18}$$

y el operador temporal de segundo orden, (2.19)

$$m(x, y, t) = -\left(\nabla^2 + \frac{1}{u^2}\frac{\partial^2}{\partial t^2}\right) G(x, y, t) \tag{2.20}$$

donde u es un factor del tiempo y a es el ancho del filtro. Luego, el operador temporal de bordes convoluciona con la imagen.

$$S(x, y, t) = I(x, y, t) \star m(x, y, t) \tag{2.21}$$

donde la convolución está definida como:

$$I(x, y, t) \star m(x, y, t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(a, b, c)m(a - x, b - y, c - t)\partial a\partial b\partial c$$

que nos proporciona el movimiento de la imagen $I(x, y, t)$.

Medidas generalizadas

Las características visuales tales como bordes, segmentación, movimiento y color son utilizadas en las primeras etapas de la vision humana. Este pre-procesamiento permite que posteriormente se realicen análisis de alto nivel en el cerebro. Basados en esta observación, es razonable proponer que a nivel computacional se realice primero el procesamiento de bajo nivel de las propiedades generales de la imagen.

La simetría de los rasgos humanos es una virtud innegable que puede ser utilizada para un lenguaje que, aunque no es de alto nivel, nos da una aproximación muy efectiva de la localización de los rasgos faciales. La medida de la simetría asigna una magnitud en la localización de todos los pixeles de una imagen, basada en la contribución de los pixeles circundantes. La magnitud de la simetría $M_\sigma(p)$ para el pixel p está dada por:

$$M_\sigma(p) = \Sigma_{i,j\in\Gamma(p)} C(i,j) \tag{2.22}$$

donde $C(i,j)$ es la contribución del pixel circundante de coordenadas (i,j) para el conjunto de pixeles definido por $\Gamma(p)$; los cuales se definen de la siguiente forma:

$$\Gamma(p) = \left[(i,j)|\frac{p_i + p_j}{2} = p\right] \tag{2.23}$$

$$C(i,j) = D_\sigma(i,j)P(i,j)r_i r_j \tag{2.24}$$

donde $D_\sigma(i,j)$ es una función de distancia, $P(i,j)$ es una función de fase y r_k se definen como:

$$D_\sigma(i,j) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{\|p_i - p_j\|}{2\sigma}} \tag{2.25}$$

$$P(i,j) = (1 - cos(\theta_i + \theta_j - 2\alpha_{i,j}))(1 - cos(\theta_i - \theta_j)) \tag{2.26}$$

$$r_k = log(1 - \|\nabla p_k\|) \tag{2.27}$$

$$\theta_k = arctan\left(\frac{\frac{\partial}{\partial y}p_k}{\frac{\partial}{\partial x}p_k}\right) \tag{2.28}$$

donde p_k es algún punto (x,y), ∇p_k es el gradiente de la intensidad en el punto p_k, y $\alpha_{i,j}$ es el ángulo en sentido contrario de las manecillas del reloj formado por la línea que pasa por los puntos p_i y p_j, y la horizontal.

Usando el mapa de magnitudes, se obtiene un 95 % de detecciones de ojos y bocas en bases de datos similares. Algunos experimentos revelan que esta medida es insensible a bordes y a figuras no convexas en el *background*.

En la Figura 2.2 puede observarse el mapeo de M_σ y la imagen original en donde puede apreciarse la detección de ojos y boca.

Figura 2.2: Imagen original y M_σ. Tomada de Hjelmȧs y Low [1]

2.1.2. Análisis de rasgos

En análisis de bajo nivel muchas veces genera resultados poco confiables, por ejemplo, con un modelo de color de piel puede detectarse un rostro y además un objeto que se encuentre en el *background* y que sea de un color parecido, por lo que el porcentaje de detecciones de rostros es muy buena, pero es muy probable obtener un falso positivo. Este problema puede ser resuelto mediante un modelo de análisis de rasgos de alto nivel. Existen dos técnicas para el análisis de rasgos; por una parte la búsqueda de rasgos y por otra el llamado *análisis de constelaciones*.

Búsqueda de rasgos

La búsqueda de rasgos comienza con la determinación de las características faciales más prominentes, partiendo de esto se puede estimar la ubicación del rostro mediante mediciones antropológicas. Por ejemplo, si se detecta un area pequeña en la parte superior y un área mayor en la inferior significa que la cabeza está arriba y lo de abajo son los hombros, y un par de regiones oscuras en la cabeza nos confirma una exitosa detección del rostro.

Análisis de constelaciones

Algunos de los algoritmos utilizados en esta técnica están fuertemente basados en reglas heurísticas tomada de modelos de imágenes faciales sujetas a condiciones

fijas. En este tipo de análisis es posible la aparición de errores al intentar hacer una detección en una imagen con un *background* complejo o con poses no frecuentes del rostro, por lo que los avances en esta área orillaron a desarrollar modelos robustos de agrupación de rasgos faciales, como el análisis estadístico.

En la Figura 2.3 puede observarse un panorama general de el análisis de constelaciones.

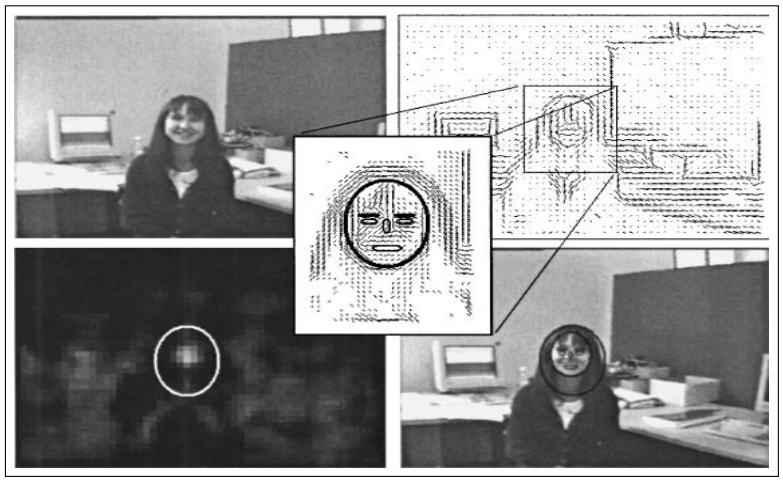

Figura 2.3: Detección de rostros mediante análisis de constelaciones. Tomada de Hjelmȧs y Low [1]

2.1.3. Análisis de formas activas

Este tipo de modelos representan el aspecto físico real del rostro, por lo tanto, se utiliza un modelo de alto nivel. Este tipo de modelos interactúan con las características de la imagen local (bordes, brillo) y poco a poco se van deformando hasta tomar la forma de la entidad.

Serpientes

Los contornos activos son usualmente utilizados para la detección de las fronteras de la cabeza. Para lograrlo, los contornos primero inicializan en las vecindades de las fronteras. La evolución de una serpiente se consigue reduciendo al mínimo una función de energía, la E_{snake} (analogía con los sistemas físicos), que se denota como:

$$E_{snake} = E_{interna} + E_{externa} \tag{2.29}$$

donde $E_{interna}$, $E_{externa}$ son las funciones de energía interna y externa respectivamente.La energía interna es la parte que depende de las propiedades intrínsecas de las serpientes y define su evolución natural (reducción o ampliación). La energía externa contrarresta los efectos de la energía interna.

Plantillas deformables

La localización de los límites de rasgos faciales no es una tarea fácil ya que es difícil organizar todos los elementos en una única entidad global. El bajo contraste de la luminosidad también hace problemática la detección de bordes. Existe un concepto más allá de la idea de serpientes, en la cual se incorpora la información global del ojo para mejorar la fiabilidad del proceso de extracción. Se toma una plantilla de los ojos que trabaja bajo el mismo principio que las serpientes en la cual se consideran los rasgos más sobresalientes usando 11 parámetros.

El mecanismo de deformación consiste en la minimización del descenso más pronunciado de una combinación de energía externa debido al valle, punta, bordes y el brillo de la imagen.

Modelos de distribución de puntos (PDM)

Los PDM (Points Distribution Models) son descripciones compactas parametrizadas de las formas basados en estadísticas. El contorno de los PDM se discretiza en un conjunto de puntos etiquetados. Las variaciones de estos puntos primeros son parametrizados en un conjunto de entrenamiento que incluye objetos de diferentes tamaños y posturas. Utilizando el análisis de componentes principales (PCA), las variaciones de los rasgos en un conjunto de entrenamiento se construyen como un modelo flexible lineal. El modelo cuenta con la media de todos los rasgos en los escenarios y los principales modos de variación de cada punto

$$x = \bar{x} + P\mathbf{v} \tag{2.30}$$

donde x representa un punto en el PDM, $\bar{x}$ es la función media del conjunto de entrenamiento para ese punto, $P = [p_1 p_2 ... p_t]$ es la matriz de los t vectores más

significativos de la covarianza.

El modelo engloba todas las características faciales tales como ojos, cejas, boca y nariz teniendo un 95 % de detecciones.

2.1.4. Técnicas basadas en imágenes

Las técnicas mencionadas anteriormente son utilizadas bajo ciertas restricciones sobre el segundo plano (*background*), luz, etc. Existe una necesidad de técnicas que pueden actuar en escenarios más hostiles, tales como la detección de múltiples rostros con un alto nivel de desorden en el segundo plano. Esta restricción ha inspirado una nueva área de investigación que trata la detección de rostros como un problema de reconocimiento de patrones. Al formular el problema como un aprendizaje que reconoce un patrón de la cara a partir de ejemplos. Este tipo de técnicas están inmersas en el área de inteligencia artificial

Modelos de subespacios lineales

Estos modelos incluyen las técnicas de análisis del componente principal (PCA), análisis del discriminante lineal (LDA) y análisis de factor (FA). En este tipo de técnicas es usual trabajar con los eigenvectores y eigenvalores de la matriz de covarianza de la distribución, en donde cada una de las caras puede ser expresada como combinación lineal de los eigenvectores más grandes, comúnmente llamados *eigenfaces*.

La idea general de esta técnica consiste en que a partir de un conjunto de fotogramas que representan rostros, generar una base canónica de un subespacio lineal que represente los componentes principales de un rostro. Esta base canónica deberá ser obtenida de los *eigenvectores*. Por lo tanto, los rostros del conjunto inicial deberá generarse de la combinación lineal de la base canónica (*eigenfaces*).

La detección de rostros se realiza proponiendo una distancia mínima entre el espacio de eigenfaces y el rostro representado. Si la distancia entre el espacio lineal y el rostro es menor a la distancia mínima entonces se realiza la detección. Esta técnica, finalmente, es una técnica de optimización.

Redes neuronales

Las redes neuronales se han aplicado exitosamente en muchos de los problemas de reconocimiento de patrones, tales como el reconocimiento óptico de caracteres, reconocimiento de objetos y la conducción autónoma de un robot. Dado que la detección de rostros puede ser tratada como un problema de reconocimiento de clases de patrones, han sido propuestas diferentes arquitecturas de redes neuronales. La ventaja de utilizar redes neuronales para la detección de rostros es la viabilidad de la formación de un sistema para captar la densidad condicional de patrones complejos de rostros.

Sin embargo, una desventaja es que la arquitectura de red tiene que ser ampliamente sintonizada (número de capas, el número de nodos, la tasa de aprendizaje, etc) para conseguir un rendimiento eficiente.

Esta técnica usualmente es utilizada para completar imágenes borrosas o incompletas. Es uno de los métodos más utilizados para la detección de rostros debido a su alto porcentaje de detecciones correctas y el bajo porcentaje de falsos negativos. Requiere una fase de entrenamiento en el cual se requiere un conjunto de *c*aras y otro conjunto de *no caras* de tal forma que el algoritmo permita establecer un buen criterio de decisión entre dos clases (caras y no caras).

De acuerdo con Viennet y Soulié [18] una neurona es un proceso elemental que se compone de (ver Figura 2.4):

- un estado interno $p_i \in \mathcal{P}$ donde $\mathcal{P}$ es el conjunto de procesos internos.
- un conjunto finito de señales de entrada $s_1,\ s2,\ ...,\ s_n$
- una una función de transición de estado g tal que $p_i = g(s_1, s_2, ..., s_n)$, es decir, el estado iterno p_i depende de las señales de entrada.

Una red neuronal es el conjunto de neuronas interconectadas que se caracteriza por:

- su arquitectura: el número de neuronas y un esquema de interconecciones
- las funciones de transición de las neuronas: los pesos W_{ik} y las funciones g

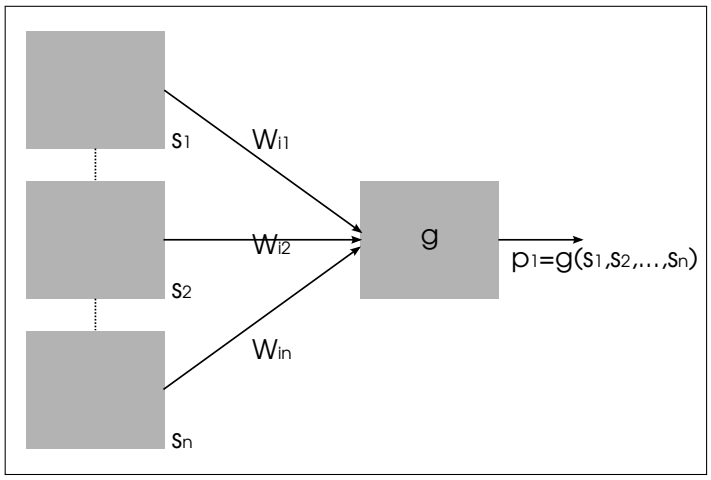

Figura 2.4: Neurona.

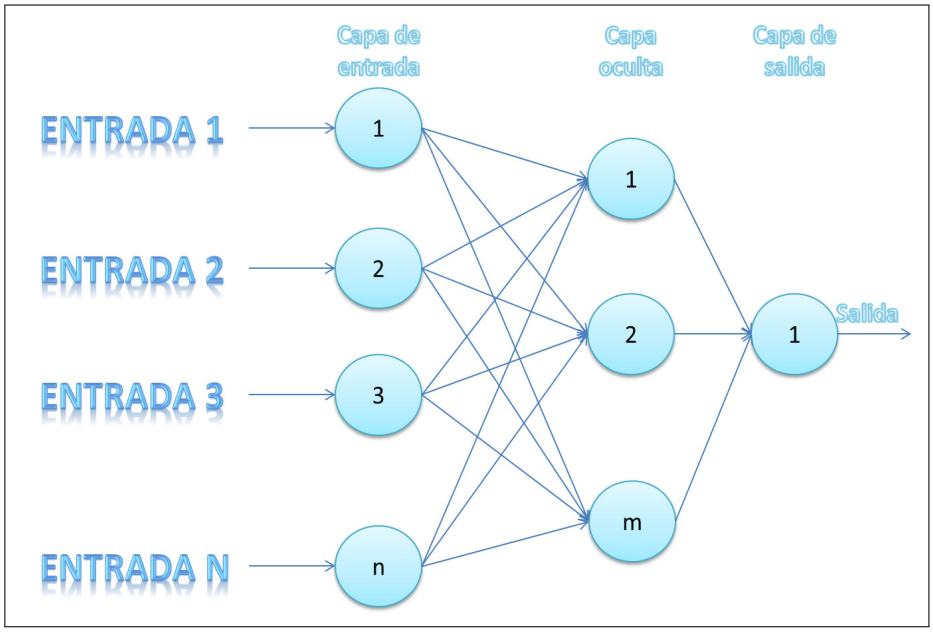

Figura 2.5: Red Neuronal con arquitectura de Perceptrón simple.

En la Figura 2.5 se muestra un ejemplo de una arquitectura de redes neuronales (Perceptrón simple).

Hjelmȧs y Low [1], Yang et al. [12], Li y Jain [19], Zhang y Zhang [20] mencionan que uno de los trabajos más completos y eficientes en cuanto a detección de rostros se refiere es el elaborado por Rowley et al. [2] realizado mediante redes neuronales

interconectadas retinalmente. En la Figura 2.6 se puede observar el funcionamiento de esta propuesta que consiste en sub-muestrear las imágenes en ventanas de 20×20 pixeles y realizar un pre-porcesamiento para corregir factores de iluminación en la imagen original, posteriormente entran a la red neuronal que consiste de 26 unidades en la capa oculta, de las cuales 4 unidades trabajan con subregiones de 10×10 pixeles, 16 con subregiones de 5×5 pixeles y 6 unidades buscan en franjas horizontales traslapadas de 20×5. La salida de esta red será de -1 o de 1 si la ventana no contiene o contiene un rostro respectivamente. El proceso es escalado sobre la ventana inicial por un factor de 1.2. Cuando existen detecciones traslapadas, se propone un umbral que nos indica el número mínimo de traslapes para reconocer una detección como verdadera, en caso de que una detección sea reconocida como verdadera entonces los traslapes son considerados falsos positivos y rechazados.

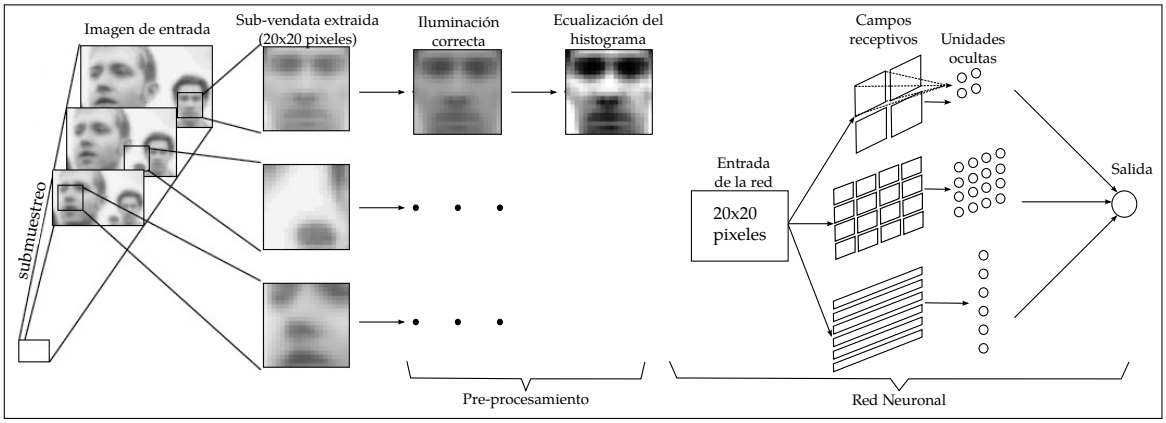

Figura 2.6: Proceso de detección de rostros de Rowley et al. [2].

En ese mismo año, los autores propusieron una mejora para imágenes rotadas, ya que la primer propuesta sólo hace detecciones para imágenes frontales con un ángulo de inclinación de $\pm 10^o$ teniendo hasta un 90,5 % detecciones acertadas para imagenes frontales y hasta un 70,6 % de aciertos para imágenes frontales rotadas.

Técnicas estadísticas

Este tipo de algoritmos no suponen ningún tipo de información previa de la tipología de una cara; sino, que a partir de un conjunto de muestras (imágenes de caras y de no caras) de entrenamiento extraen la información relevante que diferencia

un objeto cara de un objeto no cara. Son sistemas basados en teoría de la información, máquinas de soporte vectorial y reglas de decisión de Bayes. Son algoritmos de aprendizaje al igual que los subespacios lineales y redes neuronales, con la diferencia que no toma decisiones ponderadas, si no de acuerdo a la probabilidad de pertenecer a un grupo. Este grupo incluye uno de los métodos más referenciados y utilizados actualmente: el detector de caras AdaBoost (Rama y Tarrés [21]).

Las técnicas anteriormente mencionadas son utilizadas para la detección de rostros, pero si se desea tener un panorama más amplio en Hjelm*à*s y Low [1], en Yang et al. [12] se puede encontrar más información de cada una de las técnicas de detección de rostros.

2.2. Detección de rostros. Método de Viola-Jones

El método propuesto por Paul Viola y Michael Jones [22], es uno de los métodos más usados hoy en día para la detección de rostros ya que ha permitido segmentar múltiples rostros en una imagen con tiempos de procesamiento bajos (Rama y Tarrés [21]). De acuerdo con Caruana y Niculescu-Mizil [23] los métodos basados en *boosting*, tal como el método de Viola-Jones, son los que tienen un simple y extremadamente efectivo comportamiento, mejor aún que redes neuronales, árboles de decisiones y algoritmos de máquinas de soporte vectorial. De acuerdo con Viola y Jones [22] el algoritmo es hasta 15 veces más efectivo que Rowley et al. [2] con hasta el 95 % de verdaderos positivos. Actualmente los avances relacionados a la detección de rostros están enfocados en *Boosting* y basados en el método de Viola Jones como plantean Zhang y Zhang [24].

A partir de la publicación Viola y Jones [22] en 2001, la detección de rostros, ojos, labios y rasgos esenciales del rostro dio un giro, convirtiéndose en el caballito de Troya en cuanto a detección de rostros. En 2004 se publicó un segundo artículo Viola y Jones [25], el cual generaliza la detección de cualquier otro objeto, aunque en principio se tenía como objetivo resolver el problema de detección de rostros. De acuerdo con Viola y Jones [26], el método es un método estadístico de detección de rostros (ver Hjelm*à*s y Low [1]), sin embargo autores como Li y Jain [19], Zhang y Zhang [20], Tabatabaie et al. [27] consideran que el método está mejor agrupado

bajo la clasificación de Yang et al. [12] en los *Métodos basados en apariencia* o bien, consideran el *boosting* como un meta-algoritmo adicional a los métodos propuestos en Hjelmȧs y Low [1].

El auge que ha tomado este algoritmo es debido a que la ejecución de este es en tiempo real, además de otras tres importantes aportaciones. La primera de ellas es la introducción de la imagen integral, la segunda es la extracción de características y la tercera es la clasificación en cascada. El método fue elaborado y probado para imágenes de 384×288 pixeles. En la Figura 2.7 podemos observar las etapas de este método propuesto en Viola y Jones [22].

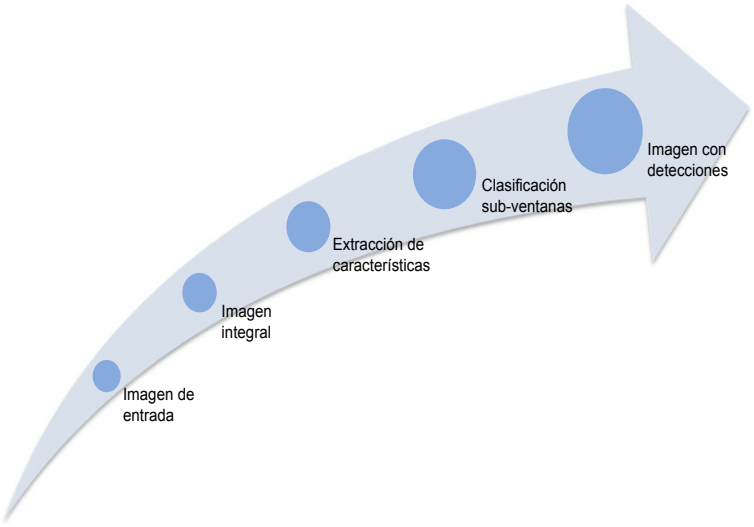

Figura 2.7: Diagrama del método de Viola-Jones

2.2.1. Imagen integral

La imagen integral es una representación compacta de la imagen original. Esta representación es uno de los aportes más importantes del método del Viola-Jones y permite trabajar de manera rápida con la imagen, pues no trabaja con las intensidades de cada pixel, sino que trabaja con una imagen acumulativa que resulta de operaciones básicas. Sea una imagen de $N \times M$ pixeles, la transformación a la imagen integral resulta ser un arreglo de tamaño $N \times M$.

Consideremos una imagen y en ella un pixel $(x,\ y)$, la imagen integral localizada

en $(x,\ y)$ contiene la suma de los valores de las intensidades en los pixeles que se encuentran arriba y a la izquierda de este punto de la imagen, incluyendo al punto $(x,\ y)$, de esta forma,

$$ii(x,y) = \sum_{x' \leq x,\ y' \leq y} i(x',y') \tag{2.31}$$

donde $ii(x,y)$ es la *imagen integral* e $i(x,y)$ es la *imagen original* (matriz de intensidades).

Sea $D = [\triangle x_1 \times \triangle y_1] \bigcup [\triangle x_2 \times \triangle y_2] \bigcup ... [\triangle x_m \times \triangle y_m] = [a_1, b_1) \times [a_2, b_2)$ el dominio de una función $f : \mathbb{R}^2 \to \mathbb{R}$ y $(x_j,\ y_j)$ un punto en una subregión $\triangle x_j \triangle y_j$ donde $\triangle x_j$ y $\triangle y_j$ son gradientes de pixeles en el eje x y y, respectivamente. Entonces, el volumen del i-ésimo paralelepípedo está dado por:

$$V_i = f(x_i,\ y_i) \triangle x_i \triangle y_i$$

Si se toma la suma de Riemann y se hace tender el número de particiones a infinito se obtiene la doble integral.

$$\begin{aligned} \lim_{m \to \infty} \sum_{i=1}^{m} f(x_i,\ y_i) \triangle x_i \triangle y_i &= \lim_{\|\triangle \to 0} \sum_{i=1}^{m} f(x_i,\ y_i) \triangle x_i \triangle y_i \\ &= \int \int_D f(x,y)\, dx\, dy \end{aligned}$$

Si $\triangle x_i = \triangle y_i = 1\ \forall i$, entonces

$$\sum_{i=1}^{m} f(x_i,\ y_i) \approx \int \int_D f(x,y)\, dx\, dy$$

entonces,

$$ii(x,y) = \sum_{x' \leq x,\ y' \leq y} i(x',y') \approx \int \int_{x' \leq x,\ y' \leq y} i(x',y')\, dx'\, dy' \tag{2.32}$$

Es decir, la imagen integral es la integral doble de las filas y las columnas hasta el punto (x,y) de la imagen original.

Usando el siguiente par de funciones de recurrencia la imagen integral puede ser calculada en una sola ejecución:

$$s(x,y) = s(x,y-1) + i(x,y) \tag{2.33}$$
$$ii(x,y) = ii(x-1,y) + s(x,y) \tag{2.34}$$

donde $s(x,y)$ es la suma acumulativa de las filas donde $x \in 1,2,...,M,\ y \in 1,2,...,N$, $s(x,0) = 0$ y $ii(0,y) = 0$. En la Figura 2.8 podemos observar la representación gráfica de la imagen integral.

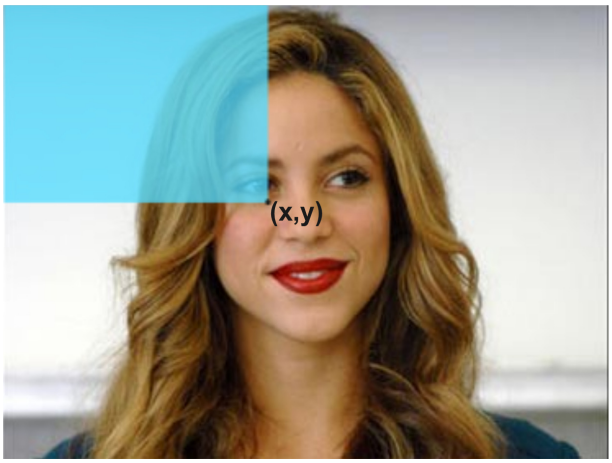

Figura 2.8: Valor de la imagen integral: suma de todos los pixeles en el extremo superior izquierdo

Mediante el cálculo de la imagen integral $ii(x,y)$ puede realizarse el cálculo de cualquier suma dentro de un área de la imagen (ver Figura 2.9). El área D puede ser calculada mediante cuatro referencias. El punto 1 (x_1,y_1) es la imagen integral $A = ii(x_1,y_1)$; el punto 2 (x_2,y_2) es $A+B = ii(x_2,y_2)$; el punto 3 (x_3,y_3) es $A+C = ii(x_3,y_3)$ y el punto 4 (x_4,y_4) es $A+B+C+D = ii(x_4,y_4)$. Por lo tanto, el área D puede ser obtenida de $ii(x_4,y_4) + ii(x_1,y_1) - \{ii(x_2,y_2) + ii(x_3,y_3)\}$.

2.2.2. Extracción de características

La extracción de características es la parte de la propuesta Viola y Jones [22] en la que se entrena a la máquina para que aprenda a reconocer rostros partiendo

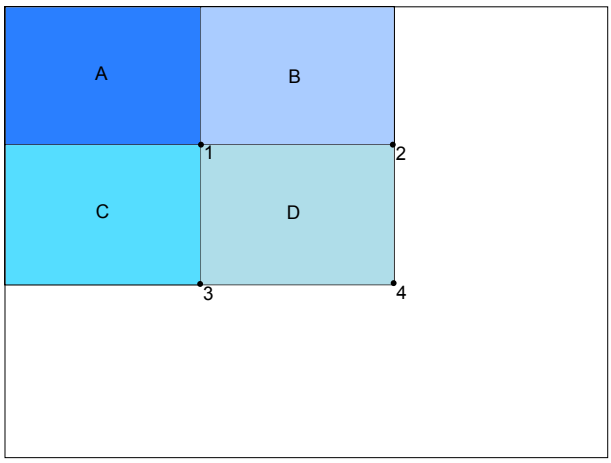

Figura 2.9: Cálculo de un área rectangular mediante imágenes integrales

de características de bases Haar[2].

La extracción de características aprovecha el manejo de la imagen integral y facilita el rápido procesamiento de la imagen en comparación con el manejo directo de intensidades de los pixeles. Las características propuestas son de tres tipos: *Características dos-rectangulares (two-rectangle feature)* que es la diferencia entre la suma de las intensidades de los pixeles de dos regiones rectangulares. Ambas regiones tienen el mismo tamaño y forma y pueden utilizarse tanto vertical como horizontalmente, estas características extrae bordes vertical y horizontal respectivamente. Las *características tres-rectangulares (three-rectangle feature)* se calcula mediante la suma de las intensidades de los pixeles de dos rectángulos exteriores y sustrayendo el rectángulo central, extrayendo líneas. *Características cuatro-rectangular (four-rectangle feature)* se obtienen de la diferencia entre las intensidades de los pixeles de dos rectángulos de una diagonal y los otros dos rectángulos de la otra diagonal, extrayendo líneas diagonales R. y J. [3]. En la Figura 2.10 pueden observarse los tipos de características considerados.

Las características se construyen mediante la adición y sustracción de la imagen integral clara y oscura tomando los vértices inferiores derechos como punto de referencia.

[2]Las bases Haar son filtros que realizan diferencias de intensidades entre regiones de una imagen. Con ello se genera una base de líneas, contornos y puntos que definen así características de objetos.

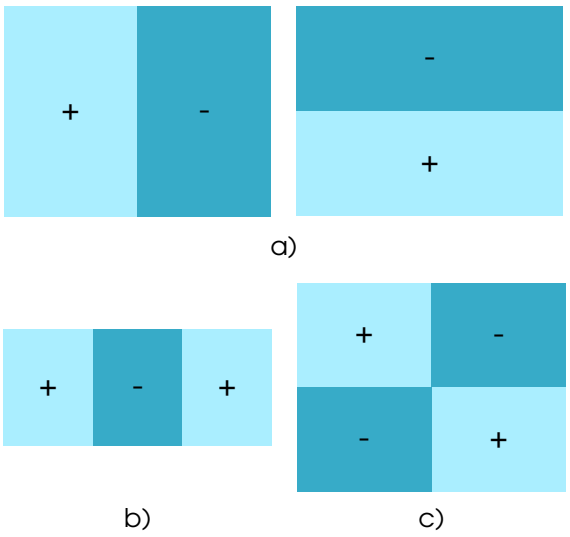

Figura 2.10: Características utilizadas en el método de Viola-Jones. a) two-rectangle. b) three-rectangle. c) four-rectangle

Por lo tanto notemos que a partir de la imagen integral es muy fácil realizar la extracción de características ya que el cálculo de ellas sólo se requiere tomar valores directamente de la imagen integral tal como en la Figura 2.9. En adición a esto, una motivación más para pensar en un mecanismo acelerado puede ser explicado como en Simard et al. [28]. Los autores encontraron que para el caso de operaciones lineales (por ejemplo $f \cdot g$), alguna operación invertible puede ser aplicada a f o a g y su operación inversa al resultado. Por ejemplo, para el caso de convolución puede aplicarse a g la doble derivada y aplicarle una doble integral al resultado:

$$f \star g = \int \int (f \star g'') \tag{2.35}$$

Por la linealidad de la convolución y la integral,

$$\int \int (f \star g'') = \left(\int \int f \right) \star g'' \tag{2.36}$$

$$\Rightarrow f \star g = \left(\int \int f \right) \star g'' \tag{2.37}$$

Se ha demostrado que el cálculo de la convolución es más acelerada si se calcula como en 2.37, sobre todo cuando las matices son dispersas o tienden a ser dispersas.

Ahora bien, la suma de pixeles puede ser expresada como un producto punto $i \cdot r$ donde i es la imagen y r es el rectángulo que recorre a la imagen, tomando valores de 1 en el área de interés y de 0 fuera de dicha área. Actuando de manera análoga a la convoluvión, el producto punto puede ser reescrito como:

$$i \cdot r = \left(\int \int i \right) \cdot r'' \tag{2.38}$$

donde la doble integral de i es la imagen integral como se demuestra en la ecuación (2.32), y la segunda derivada de r (derivada por filas y después por columnas) corresponde a la extracción de características.

Una vez definidas las características a utilizar, el aprendizaje es el paso a seguir para que el método funcione. El mecanismo de aprendizaje que se propone es el llamado AdaBoost. Lo que se hace en este algoritmo es seleccionar un conjunto de *características débiles* que permitan descartar todas aquellas imágenes que no contienen estas características, posteriormente en otro ciclo se someten las últimas imágenes seleccionadas a otro criterio de selección (otro conjunto de características débiles) para descartar nuevamente aquellas imágenes que no se satisfacen. Si se sigue este mecanismo consecutivamente, se logra obtener al final una combinación de características débiles que forman una *característica fuerte*, la cual permite la detección de rostros.

Para ello se necesita un conjunto de imágenes que contengan un rostro, las cuales denominaremos imágenes *positivas*, y otro conjunto de imágenes que no lo contengan (imágenes *negativas*), ver Figura 2.11.

Definimos un clasificador débil $h_j(x, f, p, \theta)$ como una salida binaria (0 ó 1) que consiste en una característica f_j, un umbral θj y una polaridad p_j (indicando la dirección de la desigualdad, es decir, -1 ó 1), de esta forma,

$$h_j = h(x, f_j, p_j, \theta_j) = \begin{cases} 1, & \text{si } p_j f_j(x) < p_j \theta_j \\ 0, & \text{en otro caso} \end{cases} \tag{2.39}$$

En este caso, x es una sub-ventana de 24×24 pixeles. Es decir, si seleccionamos una característica f_j y calculamos su valor en cada una de las imágenes de entrenamiento positivas y negativas, puede establecerse un valor umbral θ_j que pueda

incluir las imágenes positivas, con ello se define un clasificador débil h_j. En la Figura 2.12 se observa la idea del clasificador débil, si fijamos la característica f_j, podemos seleccionar un umbral θ que mejor clasifique las muestras positivas y negativas (caras y no caras).

Para cada característica existe un clasificador potencial. Observemos que para una sub-ventana de 24×24 pixeles tenemos 180 mil características (variando tamaño y orientación de las características Haar) Viola y Jones [26], por lo que tenemos un número muy grande de clasificadores débiles. La tarea de seleccionar un conjunto más pequeño de características tiene un enorme costo computacional, además de que para cada característica f_j se deberá realizar la selección del umbral θ_j y su polaridad p_j para la construcción del clasificador débil h_j que mejor clasifique las caras.

Existe un algoritmo denominado AdaBoost, el cual construye un clasificador fuerte C a partir de la combinación lineal de T clasificadores débiles, cuyos pesos son inversamente proporcionales al error de clasificación. En la Figura 2.13 podemos observar el algoritmo implementado para el aprendizaje mediante AdaBoost.

Aunque el algoritmo disminuye la enorme labor de elegir T características de un gran número de características existentes, la tarea sigue requiriendo de un gran poder computacional siendo un algoritmo de orden $O(nTK)$ donde n es el

(a) Muestra de imagen positiva (b) Muestra de imagen negativa

Figura 2.11: Imágenes para entrenamiento de detección de rostros

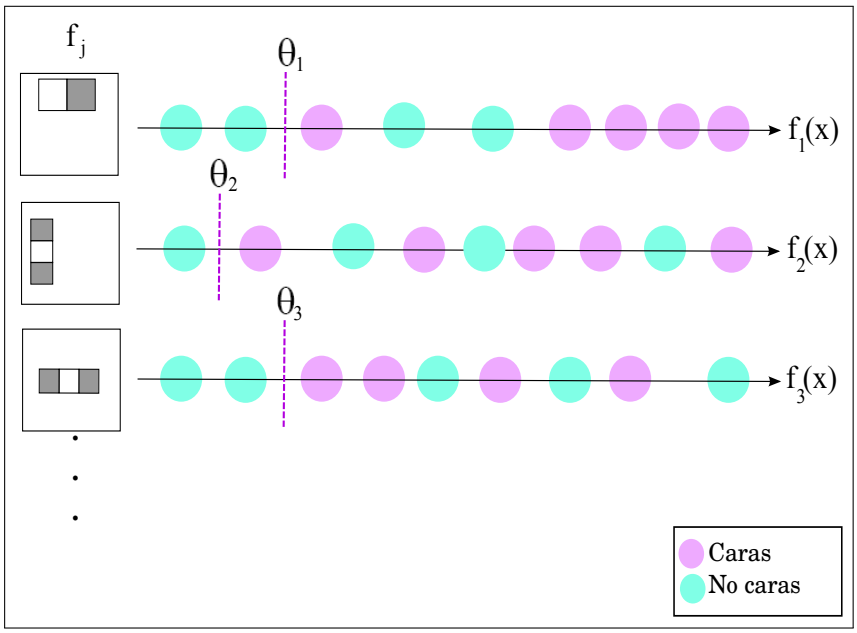

Figura 2.12: Creación de características débiles h_j

número de muestras, T es el número de clasificadores débiles participantes para el clasificador fuerte y K el número total de características. Notemos también que debe calcularse para cada una de las características el valor umbral θ_j y su polaridad p_j, sin embargo este cálculo sólo se realizará durante la primera iteración ya que los clasificadores débiles no dependen del peso ponderado. No obstante, aún requiriendo de un gran poder computacional durante el entrenamiento, este se ve compensado con los satisfactorios resultados que se obtienen a la hora de la detección, ya que el número de falsos positivos y negativos es menor que con otros métodos de detección y de acuerdo con Viola y Jones [22], se pueden hacer detecciones hasta en 15 fotogramas por segundo en imágenes de 384×288 pixeles.

- Dados los ejemplos de imágenes $(x_1, y_1), ...(x_n, y_n)$ donde $y_i = 0,\ 1$ para ejemplos positivos y negativos respectivamente.
- Inicializar los pesos $\omega_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ para $y_i = 0, 1$ respectivamente donde m y l son el número de imágenes negativas y positivas respectivamente.
- Para $t = 1, ..., T$
  1. Normalizar los pesos $\omega_{t,i} \leftarrow \frac{\omega_{t,i}}{\sum_{j=1}^{n} \omega_{t,j}}$
  2. Para cada característica j construimos un clasificador débil h_j y calculamos el error ponderado como:
  $$\epsilon_j = \sum_i \omega_{t,i} \| h_j(x, f, p, \theta) - y_i \|$$
  3. Definir $h_t(x) = h(x, f_t, p_t, \theta_t)$ como el clasificador con el menor error ϵ_t.
  4. Actualizar los pesos:
  $$\omega_{t+1,i} = \omega_{t,i} \beta_t^{1-e_i}$$
  donde $e_i = 0$ si el ejemplo x_i es clasificado correctamente, $e_i = 1$ en otro caso y $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
- El clasificador fuerte final es:
$$C(x) = \begin{cases} 1, & \text{si } \sum_{t=i}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0, & \text{otro caso} \end{cases}$$

donde $\alpha_t = log \frac{1}{\beta_t}$

Figura 2.13: Algoritmo AdaBoost. Se construye un clasificador fuerte C mediante la combinación lineal de T clasificadores débiles h_j

2.2.3. Clasificador en cascada

El clasificador es un diagrama de decisiones de árbol modificado, al cual se le denomina *cascada*. El principio básico de este clasificador es descartar las no-caras en lugar de encontrar las caras y así en lugar de tener un sólo clasificador fuerte (que sería ineficiente) se obtiene una serie de clasificadores fuertes que serán colocados en

cada etapa de la cascada. Cada etapa deberá contener un clasificador cada vez más fuerte, lo que implica un aumento en el número de características en cada clasificador fuerte a medida que la cascada avance.

La imagen es dividida en sub-ventanas tomando como la sub-ventana más pequeña de 24×24 pixeles y escalando 1.25 veces sub-ventanas más grandes que la anterior, tomando un enfoque piramidal, por lo que para una imagen de 384×288 pixeles requerirá una búsqueda en 12 escalas (Figura 2.14).

Figura 2.14: División de la imagen de entrada en sub-ventanas más pequeñas

Todas las sub-ventanas son sometidas al clasificador en cascada. En la primera etapa se encuentra un clasificador fuerte, en caso de que las sub-ventanas sean aceptables para el primer clasificador pasan a la siguiente etapa, y en caso contrario, las sub-ventanas son inmediatamente descartadas. Cuando las sub-ventanas pasan por todos los clasificadores, se obtiene la detección de rostros. El concepto se ilustra en la Figura 2.15.

Usualmente en un clasificador simple se aceptan falsos-negativos para reducir el número de falsos-positivos. En el caso del clasificador en cascada los falsos-positivos no genera ningún problema ya que en las siguientes etapas estos serán descartados.

El tiempo invertido en la elaboración del detector es realmente alto, aunque en la actualidad, como bien se ha mencionado antes, el algoritmo de Viola-Jones es muy socorrido para la detección de objetos, por lo que existe una gran diversidad de librerías y funciones de entrenamiento para la detección de objetos. La de mayor uso es la librería de OpenCV (ver Apéndice C), que permite hacer detecciones de ojos, rostros y boca.

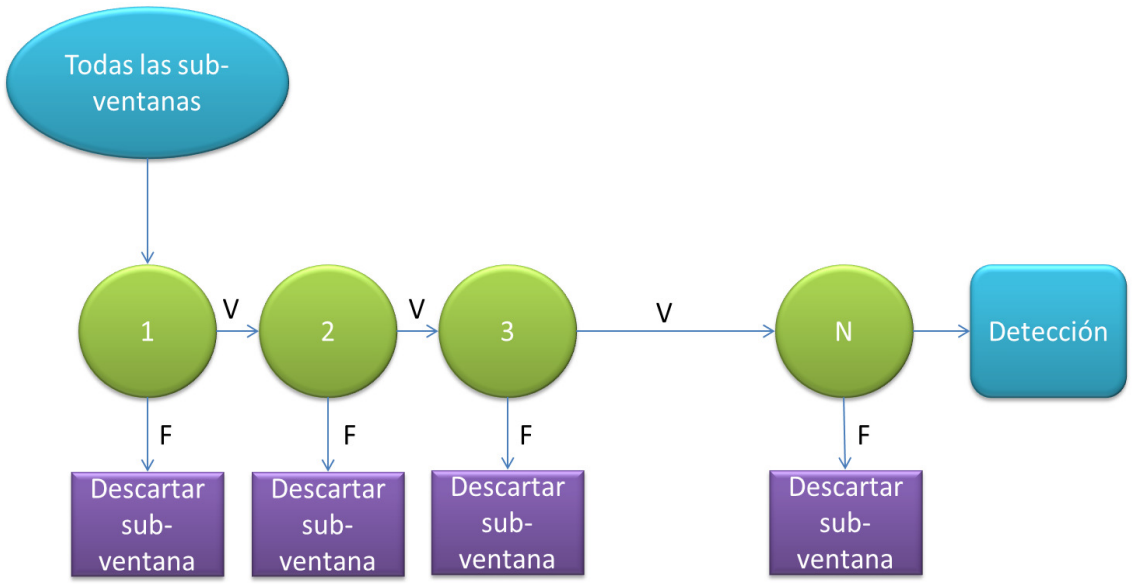

Figura 2.15: Clasificador en cascada

Integración de múltiples detecciones

Dado que las detecciones son insensibles a pequeños cambios de escala y traslación, es inevitable la presencia de múltiples detecciones traslapadas que frecuentemente son falsos positivos y a menudo representan una sola detección. Es necesario realizar un post-procesamiento de las sub-ventanas detectadas para lograr combinar los traslapes y así obtener una sola detección. En la práctica dos regiones son la misma si las fronteras están traslapadas; la detección final está delimitada mediante el promedio de las esquinas de todas las regiones traslapadas.

2.2.4. Avances y mejoras del método de Viola Jones

El auge que ha tomado el método de Viola Jones en la actualidad ha hecho de este un tópico de investigación, de hecho trabajos como el de Zhang y Zhang [20, 24] se han dedicado a recopilar trabajos actuales de detección de rostros y se ha visto que el punto de partida de los métodos actuales está en el método de Viola Jones.

La primera y fundamental mejora que se ha realizado es la de R. y J. [3]. En este trabajo se plantea la incorporación de nuevas características rotadas que detectan

la presencia de líneas y bordes diagonales, omitiendo así las características *cuatro-rectangular* de Viola Jones (Ver Figura 2.16).

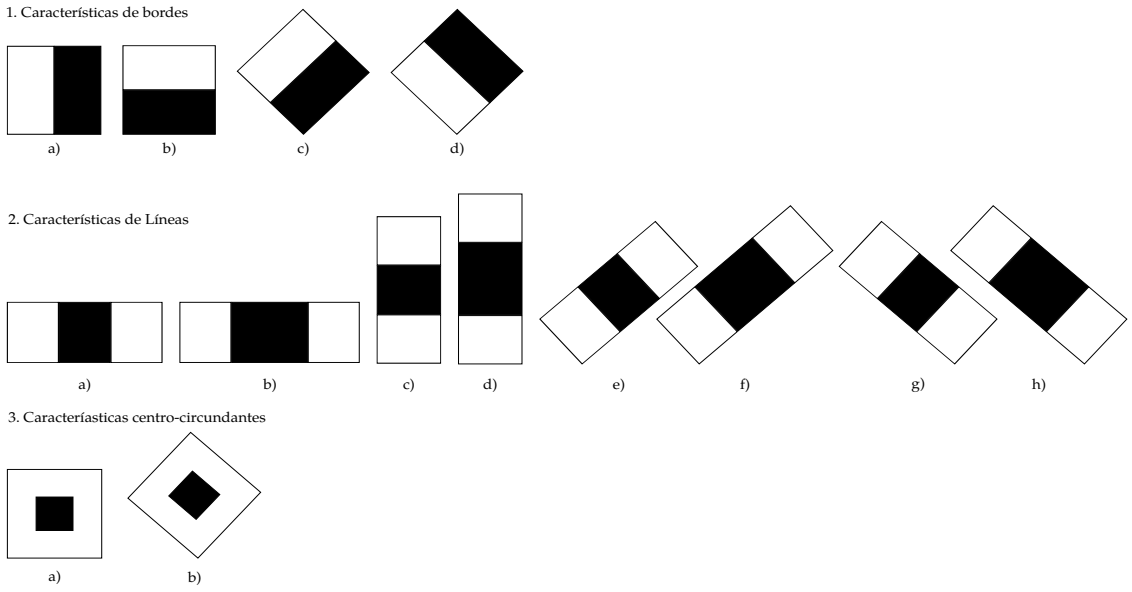

Figura 2.16: Prototipos de características. Áreas negras tienen pesos negativos y áreas blancas pesos positivos. R. y J. [3].

Asímismo se propone una ecuación de recurrencia para el rápido cálculo de las características basados en la ecuación 2.34 propuesta por Viola Jones.

Sea I la imagen original, $i(x, y)$ el pixel de la imagen I en el pinto (x, y) e $ii(x, y)$ la imagen integral con esquina superior izquierda en $(1, 1)$ y esquina inferior derecha en (x, x), entonces la imagen integral $ii(x, y)$ de la imagen está dada por la ecuación 2.31:

$$ii(x, y) = \sum_{x' \leq x,\ y' \leq y} i(x', y')$$

Por otro lado, el cálculo recurrente de la imagen integral partiendo de los pixeles anteriores es:

$$ii(x, y) = ii(x, y - 1) + ii(x - 1, y) + i(x, y) - ii(x - 1, y - 1) \tag{2.40}$$

donde $ii(x,0) = 0,\ ii(0,y) = 0$.

Así el cálculo de un área rectangular mediante imágenes integrales es análogo al cálculo propuesto en el trabajo de Viola Jones. Observando la Figura 2.9, para realizar el cálculo del área del rectángulo D, el cual denominaremos $r(x, y, w, h, \theta)$ cuya esquina superior izquierda es (x, y), ancho w, altura h y ángulo de rotación θ. Entonces la imagen integral del rectángulo r es:

$$RecSum(r) = ii(x,y) + ii(x+w, y+h) - ii(x, y+h) - ii(x+w, y) \tag{2.41}$$

Para características rotadas en un ángulo de 135^o (ver segunda fila de la Figura 2.16 g) y h)) la imagen integral estará acotada por la curva $x - |y - y'|$, por lo que la imagen integral $rii(x, y)$ está dada por:

$$rii(x,y) = \sum_{x' \leq x,\ x' \leq x - |y - y'|} i(x', y') \tag{2.42}$$

y la ecuación de recurrencia:

$$rii(x,y) = rii(x-1, y-1) + rii(x-1, y) + i(x,y) - rii(x-2, y-1) \tag{2.43}$$

con $rii(0,x) = rii(x,0) = rii(-1,y) = 0$. Entonces,

$$RecSum(r) = rii(x+w, y+w) + rii(x-h, y+h) - rii(x,y) - ii(x+w-h, y+w+h) \tag{2.44}$$

Posteriormente Lienhart et al. [29] probó las nuevas características contra el método inicial de Viola Jones y se encontró una mejora de hasta el 10 % con respecto a las características básicas propuestas por Viola-Jones, en la Figura 2.18 podemos observar una curva ROC comparando el entrenamiento usando características básicas contra las características extendidas. Así mismo se encontró de manera experimental que el tamaño de ventana inicial óptimo para la detección de rsotros es de 20×20 pixeles. El algoritmo y entrenamiento de Lienhart et al. [29] está disponible en *Open Computer Vision Library* Bradsky [30].

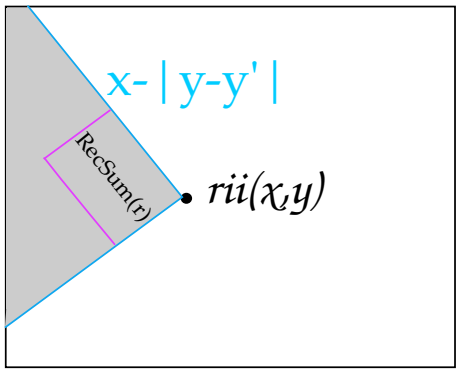

Figura 2.17: Cálculo de imagen integral y rectángulo rotado 135^o. R. y J. [3].

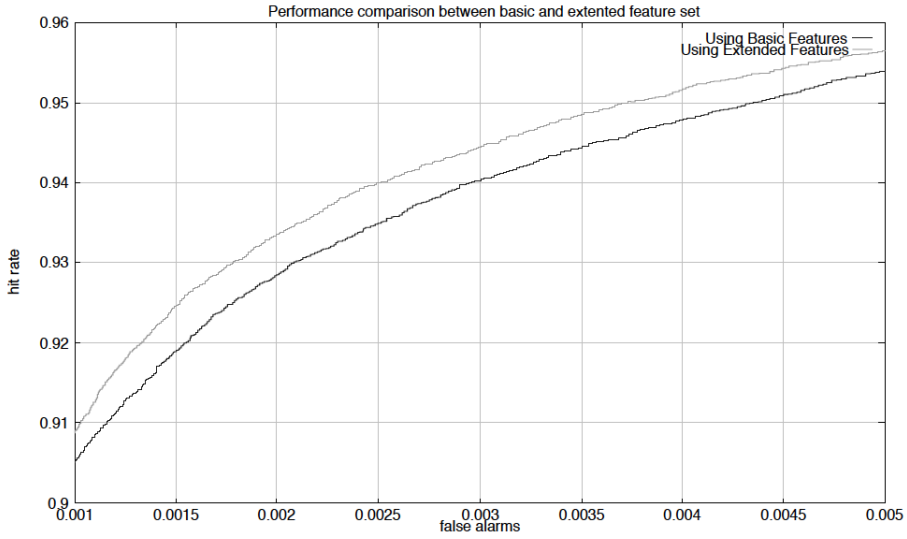

Figura 2.18: Curvas ROC usando características básicas vs características extendidas. R. y J. [3].

Existen otros trabajos basados en el método de Viola Jones que tienen como objetivo la introducción de nuevas características o con el fin de mejorar el mecanismo de aprendizaje, ambas para aumentar la precisión en detección de rostros así como para disminuir el tiempo de entrenamiento. En Zhang y Zhang [20] puede encontrarse una amplia reseña de los avances del método de Viola Jones y sus variaciones.

CAPÍTULO 3

SEGUIMIENTO DE PERSONAS

Indudablemente, el método de localización de personas es una parte clave para poder llevar un buen seguimiento de estas, no obstante, la técnica de seguimiento de personas debe ser veloz y poco robusto, de tal manera que la parte de detección de personas sea explotada al máximo.

Existen técnicas robustas para el seguimiento de objetos, tales como flujo óptico o la técnica del vector de corrimiento de medias. Estas técnicas son las más utilizadas para el seguimiento de objetos en movimiento. La ventaja que ofrecen este tipo de técnicas es la detección simultánea con el seguimiento, con lo cual se obtiene un ahorro en el tiempo de procesamiento. La gran desventaja, la cual es de vital importancia en cuanto a la aplicación que deseamos darle al sistema, es que son muy susceptibles al ruido, a condiciones de oclusión y a fallan en un *background* complejo. Esto se puede solucionar mediante una serie de heurísticas y reforzando el mecanismo con alguna otra técnica de detección, haciendo el algoritmo más robusto.

También existen técnicas para sistemas de control, como lo son los filtros adaptativos, entre los cuales destacan los filtros de Wiener, de Kalman y los filtros $g-h-k$ Brookner [31]. De acuerdo con el tipo de sistema, es el filtro a utilizar. En el caso de un comportamiento determinístico se utiliza el filtro de Wiener y los filtros $g-h-k$. En el caso de sistemas dinámicos aleatorios, es común hacer uso de el filtro de Kalman. Para el caso de seguimiento de objetos lo más común es utilizar el filtro de Kalman, pues el comportamiento del objeto es más bien aleatorio.

En la Figura 3.1 se puede observar una taxonomía de las diversas técnicas existentes para el seguimiento de objetos.

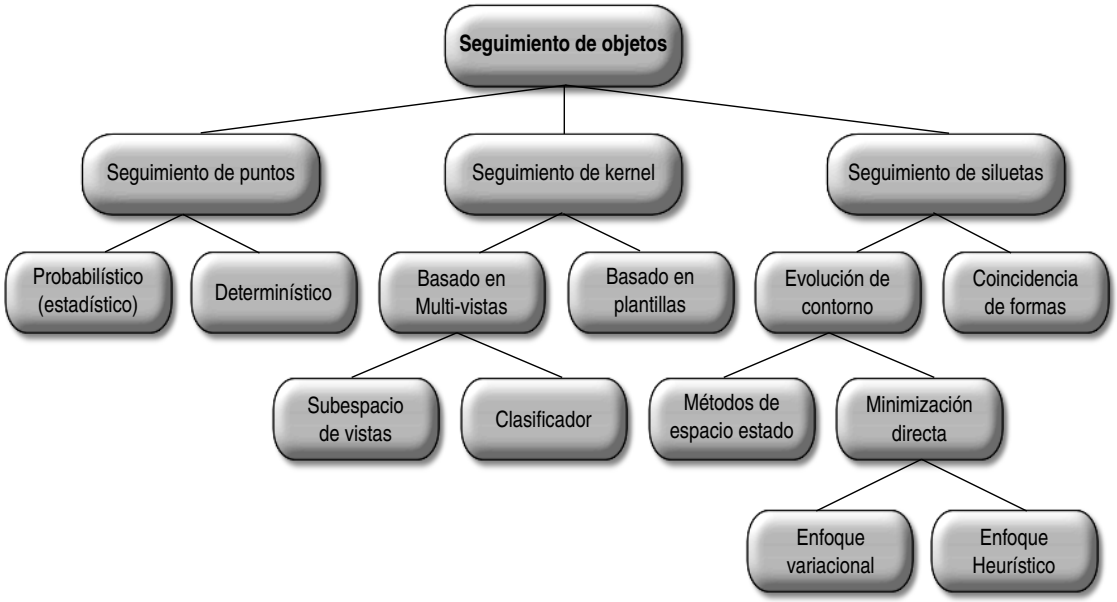

Figura 3.1: Taxonomía de los métodos de seguimiento de objetos. Yilmaz et al. [4].

Seguimiento de puntos: Es cuando el objeto es representado por puntos durante *frames* consecutivos y la asociación de los puntos está basada en el estado del objeto previo, incluyendo en ocasiones posición y movimiento. Este tipo de seguimiento contiene métodos como el Filtro de Kalman y Filtro de partículas.

Seguimiento del kernel: Kernel se refiere a la forma y la apariencia del objeto, por ejemplo una plantilla rectangular o elíptica con su correspondiente histograma. El seguimiento se realiza mediante el cálculo del movimiento del kernel en *frames* consecutivos. Este movimiento es generalmente una transformación paramétrica tal como traslación, rotación, y afín. En él se encuentran métodos como Corrimiento de medias y *Eigentracking*.

Seguimiento de siluetas: El seguimiento se realiza mediante la estimación de la región del objeto en cada *frame*. Los métodos de seguimiento de siluetas usan la información codificada dentro del objeto, dicha información generalmente

es un mapa de bordes. Dados los modelos de los objetos, el seguimiento se hace mediante Coincidencias de Formas o por Evolución de Contornos. Ambos métodos pueden ser considerados como métodos de segmentación en un dominio temporal. Métodos representativos en este enfoque son la transformada de Hough o espacios de Hausdorff.

Dentro de los métodos de **Seguimiento de Puntos** se encuentran tanto los *Métodos Determinísticos por correspondencia* como los *Métodos Probabilísticos por correspondencia* o estadísticos.

Los *Métodos determinísticos* consisten en resolver problemas de optimización los cuales minimizan el costo de correspondencia entre un *frame* t y un *frame* $t-1$ bajo restricciones de movimiento.

Los *Métodos estadísticos por correspondencia* consisten en considerar la medición del sensor del video como una variable que contiene ruido. Usan enfoques de estado-espacio para modelar las propiedades del objeto, tales como posición, velocidad y aceleración y la medición asociada al objeto es la posición en la imagen que usualmente es obtenida mediante algún mecanismo de detección. En este tipo de métodos se contiene al Filtro de Kalman, el Filtro de partículas, el Filtro de asociación de datos conjunta con probabilidad y Seguimiento de múltiples hipótesis.

En Yilmaz et al. [4] puede verse un panorama más amplio de los métodos de seguimiento de objetos existentes.

Los filtros probabilísticos nos permiten una aproximación más certera de la posición de una detección ya que considera que las mediciones son ruidosas para realizar las estimaciones y no se involucra en el mecanismo previo que es la detección de objetos como puede ocurrir en Seguimiento de Kernel o Seguimiento de Siluetas. Así, se puede hacer uso de un eficaz método de detección y aprovechar la alta eficacia del mismo. Aunque el Filtro de Kalman es un mecanismo que surgió ya hace varios años, es el padre de los filtros probabilísticos ya que es un mecanismo que optimiza las predicciones considerando un comportamiento aleatorio Gaussiano, que es una de las distribuciones más sencillas y estudiadas que existen. Esto, en consecuencia hace del filtro de Kalman un mecanismo sencillo y veloz. Este filtro es un filtro muy utilizado y estudiado dentro de la visión por computadora por lo que existen herramientas computacionales que contienen ya librerías con este filtro.

En adelante se utilizará el filtro de Kalman como el método de seguimiento de

objetos en nuestra propuesta.

3.1. Filtro de Kalman

Antes de hablar de filtros, es necesario definir el tipo de sistema con el que se está trabajando.

Un *sistema* es una colección de entidades interrelacionadas que pueden ser consideradas como uno solo. Si los atributos de interés del sistema cambian con respecto al tiempo, este es llamado *sistema dinámico*. Existen 2 tipos de sistemas dinámicos, los discretos y los continuos, los cuales dependen de la variación del sistema con respecto al tiempo. Para el caso continuo, la ecuación de estado de un sistema dinámico lineal se puede reducir a la siguiente expresión:

$$\dot{g}(t) = F(t)g(t) + C(t)u(t)$$

y

$$g_{k+1} = \phi g_k + \Gamma u_k$$

para el caso discreto. En nuestro caso haremos uso del sistema discreto pues la posición k es el k-ésimo fotograma en la videosecuencia.

Si el sistema fuera determinista, el sistema dinámico discreto se comportaría de la siguiente forma:

$$\begin{aligned} g_{k+1} &= g_k + T\dot{g}_k \\ \dot{g}_{k+1} &= \dot{g}_k \end{aligned}$$

En el caso de las personas, el sistema no sólo no tendrá velocidad constante, sino que se verá afectado por una parte aleatoria. Supongamos que esta parte aleatoria tiene una distribución normal con media cero y varianza σ_n^2. De esta manera el sistema tendrá la forma:

$$\begin{aligned} g_{k+1} &= g_k + T\dot{g}_k \\ \dot{g}_{k+1} &= \dot{g}_k + u_k \end{aligned}$$

donde u_k es el cambio aleatorio en la velocidad del tiempo k al tiempo $k+1$.

Observemos que en este caso, la ecuación de estado es una diferencial de primer orden, por lo que la precisión de las mediciones es un aspecto de vital importancia, pues es bien sabida la no continuidad de dicho operador y una mala medición de alguna variable puede darnos una mala aproximación de la medición real. Es así como el sistema tiene la siguiente forma:

$$g_{k+1} = \Phi_k g_k + w_k \tag{3.1}$$

$$z_k = H_k g_k + v_k \tag{3.2}$$

donde u_k y v_k son variables aleatorias con distribución normal con media cero y varianza Q_k y R_k respectivamente. De acuerdo con el sistema mostrado en (A.9) el filtro de Kalman puede ser utilizado mediante las ecuaciones mostradas en la Tabla 3.1 siguiendo la siguiente metodología:

1. Calcular P_{k+1}^- usando P_k, Φ_k y Q_k

2. Calcular K_{k+1} usando P_{k+1}^-, H_{k+1} y R_{k+1}

3. Calcular P_{k+1} usando K_{k+1} y P_{k+1}^-

4. Calcular los valores sucesivos de $\hat{g}_{k+1}$ recursivamente usando los valores calculados anteriormente dados el valor inicial g_0 y la medición z_{k+1}

Tabla 3.1: Ecuaciones del Filtro discreto de Kalman

Modelo del sistema dinámico	$g_{k+1} = \Phi_{k+1} g_k + w_k$
	$w_k \sim \mathfrak{N}(0, Q_k)$
Modelo de las mediciones	$z_k = H_k g_k + v_k$
	$v_k \sim \mathfrak{N}(0, R_k)$
Condiciones iniciales	$E\langle g_0 \rangle = \hat{g}_0$
	$E\langle \tilde{g}_0 \tilde{g}_0^T \rangle = P_0$
Independencia	$E\langle w_k v_j^T \rangle = 0 \ \forall k, j$
Extrapolación estimada de estado	$\hat{g}_{k+1}^- = \Phi_k \hat{g}_k + 1$
Covarianza del error de estimación	$P_{k+1}^- = \Phi_k P_k^- \Phi_k^T + Q_k$
Actualización de estimación de estado	$\hat{g}_k = \hat{g}_k^- + K_k[z_k - H_k \hat{g}_k^-]$
Actualización de la covarianza del error	$P_k = [I - K_k H_k] P_k^-$
Matriz de ganancia de Kalman	$K_k = P_k^- H_k^T [H_k P_k^- H_k^T + R_k]^{-1}$

Mediante el Filtro de Kalman puede saberse si alguna detección nueva en escena pertenece o no a las detecciones anteriores. Notemos además que la nueva predicción únicamente depende de la predicción anterior y que no es necesario conocer todas las anteriores para poder hacer una nueva predicción. Asímismo es de vital importancia tener una muy buena medición de la desviación estándar del ruido blanco, ya que el cálculo de la predicción depende de ella.

En la parte Anexa se presenta la deducción del filtro de Kalman.

3.1.1. Sistema dinámico del movimiento de rostros

En cuanto al movimiento de los rostros se refiere, es adecuado proponer que de un estado k a un estado $k+1$ el movimiento de los rostros se comporte como un **movimiento uniformemente acelerado discreto** pues al ser la variación dt muy pequeño, la aceleración se comporta de manera constante y en caso de no serlo, se compensa con la parte ruidosa inherente al proceso.

La ecuación del movimiento uniformemente acelerado discreto unidimensional es de la forma:

$$x_{k+1} = x_k + v_{x_k}\triangle t + \frac{1}{2}a_x\triangle t^2 \tag{3.3}$$

La ecuación (3.3) se puede escribir matricialmente como:

$$\begin{bmatrix} x_{k+1} \\ v_{k+1} \\ a \end{bmatrix} = \begin{bmatrix} 1 & \triangle t & \frac{1}{1}\triangle t^2 \\ 0 & 1 & \triangle t \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ v_k \\ a \end{bmatrix} \tag{3.4}$$

es decir, tiene la forma

$$X_{k+1} = \Theta_k X_k \tag{3.5}$$

donde

$$X_{k+1} = \begin{bmatrix} x_{k+1} \\ v_{k+1} \\ a \end{bmatrix}, \qquad \Theta_k = \begin{bmatrix} 1 & \triangle t & \frac{1}{1}\triangle t^2 \\ 0 & 1 & \triangle t \\ 0 & 0 & 0 \end{bmatrix}, \qquad X_k = \begin{bmatrix} x_{k+1} \\ v_{k+1} \\ a \end{bmatrix}.$$

Si ahora se propone para cada rostro un movimiento rectilíneo uniforme bidimensional sobre 2 puntos (x_1, y_1) y (x_2, y_2), donde (x_1, y_1) es la esquina inferior izquierda del rectángulo que contiene al rostro y (x_2, y_2) es la esquina superior derecha de dicho rectángulo (ver Fig. 3.2). Entonces se obtiene un sistema dinámico análogo pero con el seguimiento de más de un punto. Por lo tanto,

$$g_k = [x_1, v_{x,1}, a_{x,1}, y_1, v_{y,1}, a_{y,1}, x_2, v_{x,2}, a_{x,2}, y_2, v_{y,2}, a_{y,2}]_k^T \tag{3.6}$$

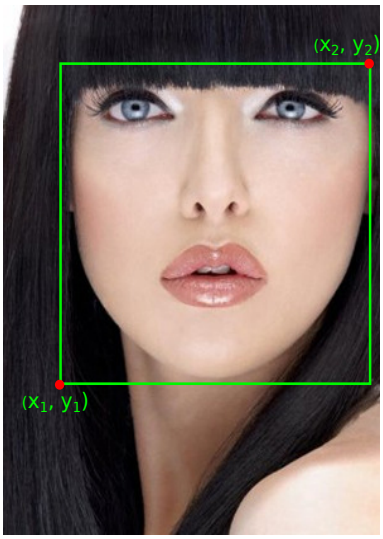

Figura 3.2: Referencia de los puntos a seguir con el filtro de Kalman

3.2. Asociación de datos

El filtro de Kalman es un buen método de tipo predictor-corrector, además de que nos proporciona una medición en casos de oclusión o de falsos negativos. Pero ¿qué sucede cuando se sigue a más de un objeto mediante el filtro de Kalman? Cuando esto sucede existe el problema de asociación de datos, ya que las trayectorias no han sido clasificadas, si no que las trayectorias son mezclas y no se tiene más información que la posición del objeto. Para ilustrar mejor esta idea observemos la Figura 3.3, en la cual vemos que no se ha hecho ninguna clasificación de trayectorias con respecto a cada objeto.

Para resolver este problema, la propuesta en este trabajo es utilizar la predicción del filtro de Kalman y asociarla con la medición más cercana bajo la $norma_2$ ($\|\cdot\|_2$) o norma euclidiana.

Sean:

$M_k = \{z_{1,k}, ...z_{m,k}\}$ el conjunto de mediciones de los m rostros en el tiempo k,

$N_k = \{\bar{g}_{1,k}, ...\bar{g}_{n,k}\}$ el conjunto de predicciones del filtro de Kalman de los n rostros en el tiempo k,

$z_{j,k}$ la medición del j-ésimo rostro en el tiempo k,

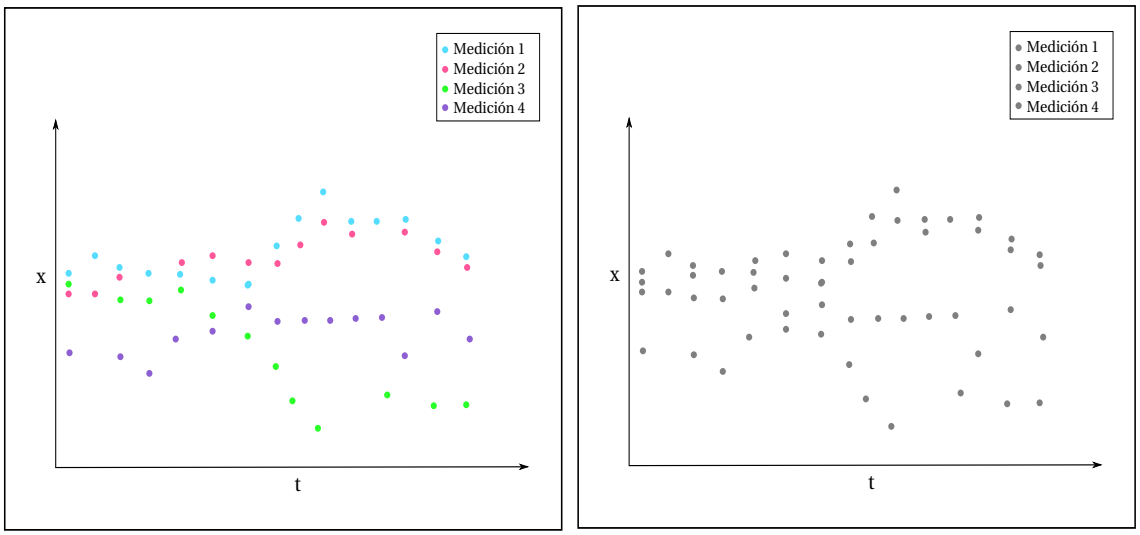

Figura 3.3: Multi-tracking con filtro de Kalman. *Izquierda*. Trayectorias clasificadas por objeto. *Derecha*. Trayectorias sin clasificar.

$\bar{g}_{i,k}$ la predicción del filtro de Kalman de la i-ésima medición en el tiempo k.
H la matriz de observación del Filtro de Kalman
Entonces:

$$z_{i,k} \quad = \quad arg \min_{z_{j,k} \in M_k} \{\|H\bar{g}_{i,k} - z_{j,k}\|_2\} \tag{3.7}$$

Para dicha asociación se construyó un vector de índices de tal forma que la i-ésima posición asocia a la predicción i del filtro de Kalman y el valor en dicha posición corresponde a la medición j-ésima.

$$index = [a_1 \ a_2 \ ... \ a_n], \qquad a_j \in 1, 2, ..., m \tag{3.8}$$

3.2.1. Aparición de un nuevo rostro en la escena

La aparición de un nuevo rostro en una escena no implica ningún problema, pues un pre-procesamiento del k-ésimo frame ayuda a evitar falsos positivos. Además aprovechando el dominio temporal puede condicionarse la aparición de un nuevo

objeto si en fotogramas consecutivos el rostro nuevo es detectado. Si el rostro es detectado en tres fotogramas consecutivos el rostro nuevo es seguido mediante un nuevo filtro de Kalman.

3.2.2. Desaparición de rostros en fotogramas contiguos

Si $n > m$ en el k-ésimo *frame*, es decir, si en el k-ésimo *frame* se tienen más predicciones que mediciones disponibles existen sólo 3 posibilidades:

- Falso negativo
- Oclusión
- Salida del objeto

Para cualquiera de los casos, el arreglo *index* tendrá al menos dos elementos repetidos, lo que significa existen 2 mediciones $z_{j,k}$ asociados a distintas predicciones $\bar{x}_{i,k}$ en el k-ésimo frame. En la Figura 3.4 se puede observar mejor la asociación de datos y el vector *index* en estos casos.

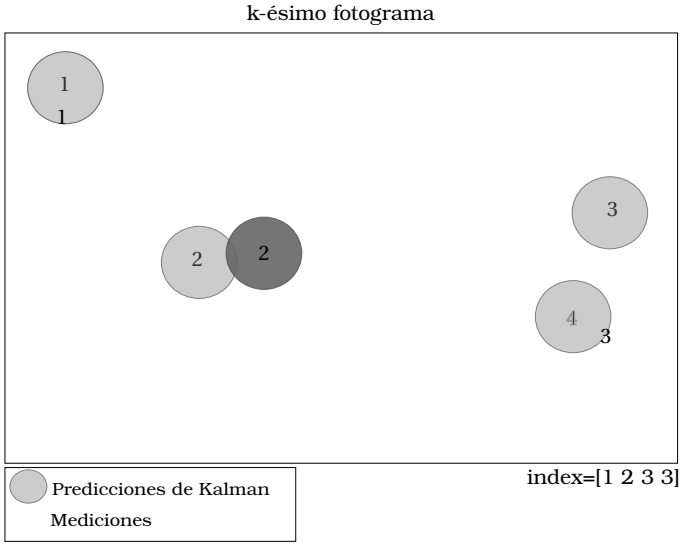

Figura 3.4: Asociación predicción-medición para la desaparición de objetos en la escena.

Los problemas de oclusión y falsos negativos se resuelven de la misma forma.

Tomando del vector *index* sólo aquellas mediciones y predicciones que están asociadas a los elementos repetidos en este vector, se calcula las distancias que hay entre dichas asociaciones, la menor distancia será considerada como una asociación correcta de datos, mientras que el resto de ellas serán consideradas como casos de oclusión y/o falsos negativos. Para estos, se agregará al vector de mediciones las predicciones asociando con ello las mediciones con las predicciones, siendo las mediciones las predicciones. Por ejemplo, tomando como referencia la Figura 3.4, se tienen 4 predicciones y sólo 3 mediciones, por lo que una de las predicciones repetirá a una de las mediciones para la asociación, por lo tanto el vector *index* repite el elemento 3 dos veces en la posición 3 y 4, lo que quiere decir que la medición 3 está asociada a la predicción 3 y a la predicción 4. La predicción 4 está más cercana a la medición 3, lo cual indica que esta es una asociación correcta, por lo tanto la predicción 3 no tiene medición, entonces tomaremos la predicción del filtro de Kalman para solucionar este problema.

Para la salida de un objeto basta con que la predicción de Kalman esté fuera de los umbrales en la imagen. Si esto ocurre, el seguimiento es *aniquilado* y es añadido a un contador que indica el número de objetos que salen del sistema. En la Figura 3.5 puede observarse el mecanismo de salida de una persona que da como resultado el conteo.

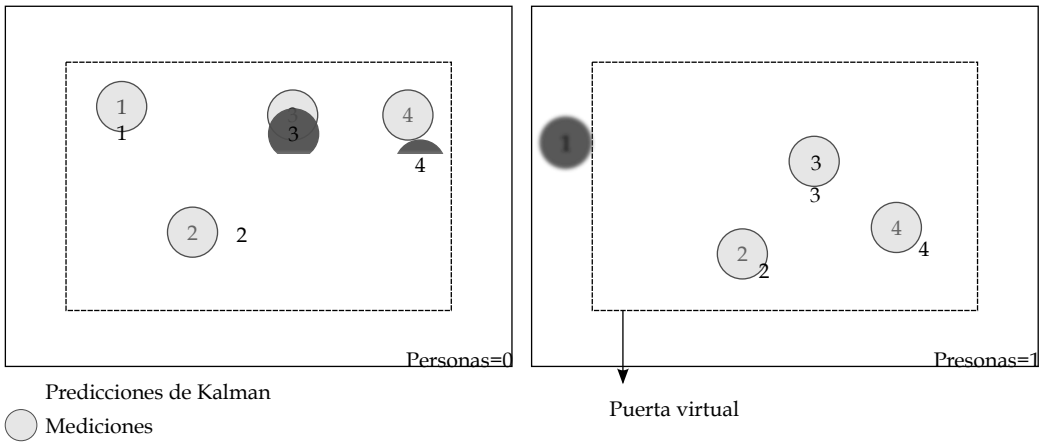

Figura 3.5: Salida y conteo de personas.

CAPÍTULO 4

DESARROLLO E IMPLEMENTACIÓN DEL PROYECTO

En este capítulo se muestra la manera en que el sistema fue implementado así como el enfoque que de acuerdo con los módulos mostrados en la Figura 1.1.

4.1. Enfoque y aportaciones

Para cumplir los objetivos indicados en la sección 1.3 se ha abordado este trabajo de acuerdo con los módulos mostrados en la Figura 1.1. Cada uno de los módulos tiene un enfoque propio, mientras que el enfoque general del proyecto está dirigido hacia la creación de un sistema cuya culminación es la obtención de una interfaz gráfica capaz de realizar el conteo automatizado de personas en una videosecuencia. Así mismo, en este trabajo se ha realizado una propuesta para la resolución de asociación de datos para el seguimiento de múltiples personas, además, se ha realizado una implementación del Filtro de Kalman de acuerdo a la calibración que se ha elaborado a lo largo de este proyecto. El entrenamiento y reconocimiento de rostros fueron tomado de la librería OpenCv 2.2, por lo que por nuestra parte nos dedicamos a elegir un filtro pasa-bajas que reduzca la aparición de falsos positivos en la detección de rostros.

En resumen, las aportaciones de este trabajo son:

- Elaboración de una interfaz gráfica que automatiza el conteo de personas en

una videosecuencia.

- Elección de un filtro y el tamaño de la ventana del mismo para la reducción de falsos positivos en detección de rostros.
- Implementación y calibración del filtro de Kalman para realizar el seguimiento y conteo de personas.
- Realización de una propuesta para la asociación de datos en un ambiente de múltiples personas.

La propuesta de asociación de datos y conteo de personas fueron implementados como se explica en el Capítulo 3.

La interfaz gráfica fue elaborada en la herramienta Matlab, por lo que hay algunas herramientas inherentes a este programa. La versión de Matlab utilizada es 7.12.0.635 (R2011a) de 64 bits para Mac. Así mismo, se hicieron uso de algunos archivos de la librería OpenCv versión 2.2 para Mac .

4.2. Fragmentación de video

Una vez tomado el video, el paso siguiente es obtener cada uno de los fotogramas estáticos que comprenden al video para que en los siguientes módulos pueda trabajarse con fotogramas fijos.

Dado que el sistema está elaborado en Matlab, el formato de video deberá ser aquel que Matlab pueda soportar, tales son .avi, .mpg, .wmv, .asf, y .asx. Existe una orden en Matlab que permite tomar el video y leer uno a uno los fotogramas de la videosecuencia, esta orden es `mmreader`. El video es leído a manera de objeto en Matlab, de tal forma que se tiene un objeto multimedia del cual se puede obtener información básica (como se muestra en la Tabla 4.1), así como el acceso a cada uno de los fotogramas del video.

4.3. Detección de rostros

La detección de rostros se trabaja en cada uno de los fotogramas del video que se desea analizar.

Tabla 4.1: Información del objeto multimedia creado en Matlab cuando se lee el video

Propiedades
Duración
Velocidad de fotogramas por segundo
Tamaño de los fotogramas
Número total de fotogramas
Formato de video
Bits por pixel

La detección de rostros se realiza mediante el Algoritmo de Viola-Jones (V-J) haciendo uso del archivo *FaceDetect.mex* (Ver código fuente en Apéndice D). Se accede a cada uno de los fotogramas y cada uno de ellos es convertido a imagen en formato doble y esta imagen es transformada a una imagen en escala de grises (comando en Matlab `rgb2gray`) y posteriormente sometida a un filtro no lineal (filtro de la mediana) con una máscara de 7×7 pixeles. Posteriormente se realiza una búsqueda por todo el fotograma tratando de obtener alguna detección mediante la función *FaceDetect*, así el valor regresado para cada fotograma con detección es una matriz de $N \times 4$ donde N es el número de rostros detectados en el fotograma. La matriz se conforma por la posición x del rostro, posición y, ancho del rostro y altura de la detección.

La función *FaceDetect* requiere de un archivo que contiene el entrenamiento del algoritmo V-J en formato *xml*, el cual es obtenido de la librería pública OpenCv. En esta librería se cuenta con cuatro archivos de entrenamiento frontal de rostros, los cuales fueron probados evaluando así el desempeño en algunos videos obtenidos en [32] y [33].

4.4. Seguimiento de rostros

El *tracking* o seguimiento de los rostros se realiza de manera simultánea con la detección de rostros, esto es, en el mismo fotograma en el que ya se realizó la detección se hace el seguimiento mediante el filtro de Kalman.

Para poder utilizar el filtro de Kalman asumimos que un rostro se mueve con **movimiento rectilíneo uniformemente acelerado** en ambas direcciones $(x,\ y)$.

Por otro lado, se asume una varianza en el detector de $10I_{4\times4}$ y una varianza en el proceso de $0{,}01I_{12\times12}$, donde $I_{m\times n}$ es la matriz identidad de tamaño $m \times n$, ambas varianzas son constantes con respecto al tiempo. Un intervalo de tiempo $dt = (1\ frame)/(frames/seg)$ donde $frames/seg$ es la tasa de fotogramas por segundo del video. Por lo tanto los valores del filtro del Kalman están dados como sigue.

Sea

$$g_k = \Phi_k g_{k-1} + w_k \quad w \sim N(0, Q_k) \tag{4.1}$$

$$z_k = H_k g_k + v_k \quad v \sim N(0, R_k) \tag{4.2}$$

donde

$$\begin{aligned}
g_k &= [x_1, v_{x,1}, a_{x,1}, y_1, v_{y,1}, a_{y,1}, x_2, v_{x,2}, a_{x,2}, y_2, v_{y,2}, a_{y,2}]_k^T \\
\Phi &= \begin{pmatrix} m & 0 & 0 & 0 \\ 0 & m & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad m = \begin{pmatrix} 1 & \triangle t & \frac{1}{2}\triangle t^2 \\ 0 & 1 & \triangle t \\ 0 & 0 & 0 \end{pmatrix} \\
H &= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \\
R &= 10I_{4\times4} \\
Q &= 0{,}01I_{12\times12} \\
P_0 &= 10^4 I_{12\times12}
\end{aligned}$$

Para el g_0 inicial, se pasa por el detector todos los fotogramas, el primer fotograma que tenga al menos una detección es utilizado para arrancar el filtro de Kalman utilizando las posiciones (x_1, y_1) y (x_2, y_2) como g_0 y unas velocidades y aceleraciones iniciales $v_{x,1} = v_{x,2} = v_{y,1} = v_{y,2} = 0$ y $a_{x,1} = a_{x,2} = a_{y,1} = a_{y,2} = 0$ respectivamente.

CAPÍTULO 5

RESULTADOS

5.1. Descripción y detalles del análisis

El sistema fue probado en videos con formato **mp4, mov** y **avi** a color.

Tres de los videos contienen un solo rostro con movimientos moderados donde no existe salida de personas y el *background* es de un color uniforme. Estos videos tienen un tamaño de *frame* de 720×576 pixeles en formato *avi*, tomados de Chan [32]. Dichos videos se utilizaron para probar la efectividad del filtro, del detector y del seguimiento de rostros, así como para comprobar que el conteo es nulo cuando no existe salida de personas. Estos videos se denominarán videos de Categoría A (Figura 5.1). En la Tabla 5.1 podemos ver los detalles de la cámara.

Figura 5.1: Fotogramas de videos utilizados en la Categoría A

Asímismo se ha utilizado un video en formato *mp4* con 4 personas en movimiento

Tabla 5.1: Detalles de video de la categoría A

Cámara	Sony VX1000E digital cam-corder
Posición de la cámara	Frontal (de frente al rostro)
Personas en el video	1
Formato de codificación de video	AVI
Razón de compresión	5:1
Resolución de muestreo de color	4:2:0
Tamaño de imagen	720 x 576 pixeles
Velocidad de reproducción	25 fps
Frames totales	281, 103 y 418
Formato de color	PPM

donde existe oclusión entre ellas, así como problemas de iluminación y *background* no uniforme, con un tamaño de *frame* de 720×576 pixeles, además de un video de 320×240 en formato *mp4* con una única persona que se mueve rápidamente y el escenario contiene problemas de iluminación que genera reflejos en el *backgraund*, ambos videos están hechos para probar el tracking ya que no contienen salidas de personas (disponibles en Maggio [33], "motinas_emilio_webcam_turning" Video 1 y "motinas_multi_face_fast" Video 2). Estos videos fueron utilizados para comprobar la efectividad del filtro, el *multitracking* y el *multitracking* con oclusión. A estos videos se les asignará el nombre de Categoría B (Figura 5.2). Ver detalles en Tabla 5.2.

Tabla 5.2: Detalles de video de la categoría B

	Video 1	**Video 2**
Cámara	Logitech Quickcam	JVC GR-20EK
Posición de la cámara	Frontal (cintura y cabeza)	Frontal (cintura y cabeza)
Personas en el video	1	4
Formato de imágenes	DivX 6 compression	DivX 6 compression
Tamaño de imagen	320 x 240 pixeles	640 x 480 pixeles
Velocidad de muestreo	10 Hz	25 Hz
Frames totales	447	1275

Figura 5.2: Fotogramas de videos utilizados en la Categoría B

Por otro lado, se ha utilizado la secuencia de imágenes de la base de datos disponible en Wong [34] que consisten en fotogramas que conjuntos forman un video. Se ha utilizado el programa *QuickTime Player 7 Pro* versión 7.6.6 para unir los fotogramas y crear el video en formato *mov* a una frecuencia de 30 fps, un formato de compresión *H.264* y una resolución de la imagen de 384x288 pixeles, la calidad de compresión es alta de acuerdo con dicho programa, lo que significa un bajo porcentaje de pérdidas por compresión. La resolución de la imagen original es de 800x600 pixeles, una resolución mucho mayor a las que se han utilizado para este trabajo, por lo que se decidió disminuir la resolución y trabajar con la resolución utilizada por Viola y Jones [22] para la detección de rostros. La secuencia de imágenes contiene 100

imágenes iniciales cuyo contenido es el *background* y el resto contiene la secuencia de personas que pasan a través de una puerta de entrada para el primer video y de salida para el segundo video. Los videos se tomaron desde una vista superior como se muestra en la Figura 5.3 mediante una cámara *rig* que consiste de 3 cámaras que capturan 3 vistas distintas de una secuencia, cada vista es tomada con una cámara y en este caso se ha tomado en cuenta únicamente la vista de la segunda cámara. Se tomaron 2 de los videos de la base de datos (P2E_S5_C2 y P2L_S5_C2) para realizar las pruebas finales de conteo de personas. En la Tabla 5.3 pueden observarse los datos de origen de la secuencia de imágenes.

Figura 5.3: Cámara Rig

En la Figua 5.4 podemos observar un fotograma a la izquierda del Video P2E_S5_C2 y a la derecha uno del video P2L_S5_C2.

Finalmente como video de Prueba Final (Video PF) se ha tomado un fragmento de video descargado de youtube VideoMuseumOfCityPeople [35] que consiste en una escena de una avenida de Istanbul, Turquía con un movimiento natural, sin control de iluminación ni del movimiento de los transeúntes. La iluminación es de luz natural en un espacio abierto en una posición de la cámara a la altura de la cabeza y ligeramente de perfil. En la Tabla 5.4 pueden observarse algunos detalles del video aunque no

Tabla 5.3: Detalles de video de la categoría C

	Video P2E_S5_C2	**Video P2L_S5_C2**
Cámara	AXIS P1343 Cámara 2	AXIS P1343 Cámara 2
Posición de la cámara	Superior (cabeza y cintura)	Superior (cabeza y cintura)
Personas en el video	24	25
Formato de imágenes	jpg	jpg
Tamaño de imagen	800 x 600 pixeles	800 x 600 pixeles
Velocidad de reproducción	30 fps	30 fps
Frames totales	806	755

Figura 5.4: Fotogramas de los videos utilizados en la categoría C

se conocen los datos de la cámara.

5.2. Análisis de resultados

Se elaboró una interfaz gráfica (GUI) en el ambiente MatLab para un manejo rápido y simplificado del sistema. En dicha interfaz se observa el número de personas contadas a la salida así como la secuencia de video con los rostros detectados. En ella se puede realizar la búsqueda de la ubicación del video, puede accederse a las vistas de cada uno de los fotogramas del video y pueden observarse algunos resultados de procesamiento tales como el número de fotogramas totales del video, el número de

Tabla 5.4: Detalles de Video PF

Posición de la cámara	Frontal-perfil (cabeza y cintura)
Personas en el video	15
Formato de video	mp4
Tamaño de imagen	320 x 240 pixeles
Velocidad de reproducción	29.922 fps
Frames totales	500

fotogramas que contienen rostros, el tiempo de procesamiento en segundo y la tasa de transmisión original del video (frames/seg). En la Figura 5.5 podemos observar una pantalla de como luce la interfaz gráfica obtenida.

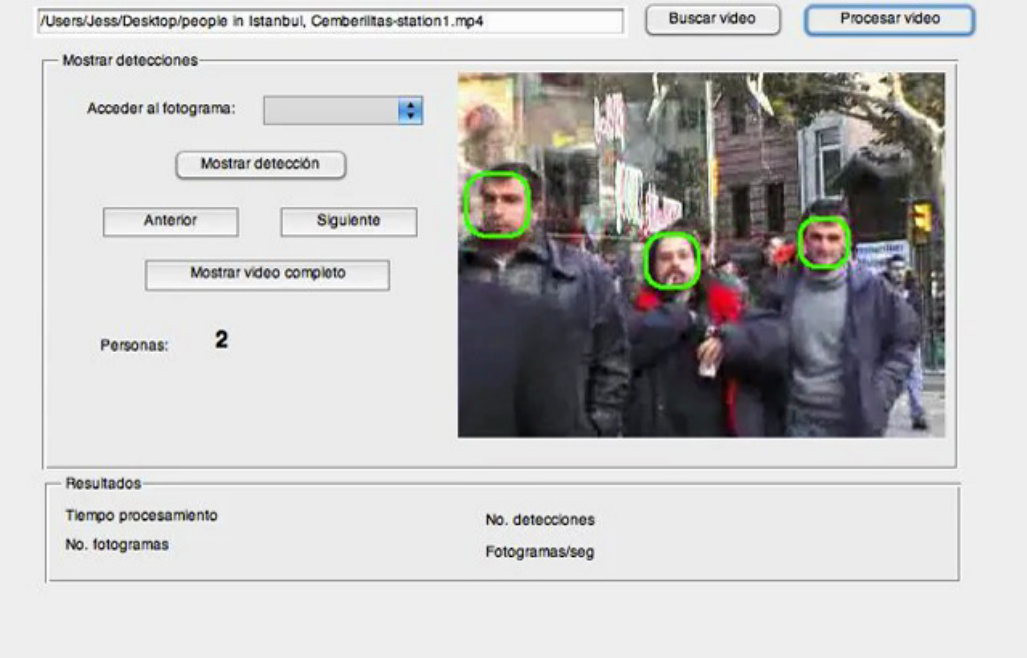

Figura 5.5: Interfaz gráfica desarrollada para el conteo de personas

Detección

En cuanto a detección de rostros se refiere, el tamaño inicial de búsqueda de rostros tiene un tamaño de 24×24 pixeles, por lo que rostros más pequeños que esta sub-ventana no son detectados; dado que el video más pequeño tiene una resolución de 320×240 se espera que haya detecciones, pues se trabaja con videos enfocados y cercanos al rostro, por lo que abarcan al menos un tamaño de rostro mayor al propuesto inicialmente. Para videos comn resoluciones más grandes a las que se proponen en este trabajo es necesario aumentar este tamaño de sub-ventana inicial, pues de lo contrario se detectarán falsos positivos muy pequeños en relación al tamaño del fotograma completo. Esto puede evitarse estandarizando el tamaño de video en una sola resolución, pues trabajar con múltiples resoluciones impide un óptimo desempeño de filtros. Asímismo, dado que el detector de rostros trabaja mediante las integrales de las intensidades de la imagen, es posible tener falsos positivos en sub-ventanas que cumplan con los clasificadores, lo cual puede evitarse con un suavizado de imagen, ya que el detector funciona mediante detección de líneas y bordes (Características bases Haar).

Filtrado

El filtro que se utilizó se obtuvo realizando pruebas con el filtro promediador, el filtro Gaussiano y el filtro de la mediana, dado que el ruido es ruido inherente a la resolución, es decir, pixeles que tienen un color muy distinto a los pixeles vecinos (comúnmente denominadas "imágenes pixeleadas"). Se realizaron pruebas con máscaras cuadradas que van desde un tamaño de 3×3 hasta 9×9 y en el caso del filtro Gaussiano hasta de 11×11 pixeles.

El filtro Gaussiano arrojó buenos resultados para videos de Categoría B y categoría C con un tamaño de máscara de 7 y buenos resultados en videos de categoría A y categoría C con una máscara de tamaño 11×11 con respecto a la existencia de falsos positivos, pero el proceso se vuelve lento y genera falsos negativos en videos de Categoría B y C. El filtro promediador arrojó buenos resultados para videos de todas las categorías con un tamaño de máscara de 7×7 pero generó falsos negativos en videos de categorías B y C. El filtro de la mediana arrojó buenos resultados en videos de todas las categorías con un tamaño de máscara de 7×7 a excepción del

video de la categoría A, el cual contiene una mujer con anteojos y presenta arrugas en el área del cuello. Dado que el filtro de la mediana genera menos artefactos y soluciona gran parte de falsos positivos, este es el filtro a utilizar para el resto del análisis, pues no se presentan grandes efectos de emborronamiento y la máscara a utilizar es de tamaño aceptable para que el proceso no tome demasiado tiempo de procesamiento por *frame*.

El filtro arroja resultados buenos, aunque no en todos los casos, pues se ha trabajado con resoluciones distintas y es bien sabido que el tamaño de la máscara del filtro está en función de la resolución del *frame*. Por otra parte, la influencia del tamaño del rostro se relaciona con la efectividad del filtro, es decir, la resolución espacial del rostro en cada *frame* es crucial para la efectividad del filtro, por lo que generalizar el tamaño de la máscara del filtro resulta difícil para todos los videos. Además de que una máscara muy grande disminuye en gran medida la velocidad de procesamiento pues se requiere de un mayor número de operaciones en cada pixel.

Seguimiento y conteo

Para estas pruebas únicamente se han utilizado los videos de la categoría C y el *Video PF*.

Dado que algunas imágenes todavía tienen falsos positivos, se propone una heurística en la parte de seguimiento la cual consiste en tener un contador que indique el número de veces que se tiene un nuevo rostro en la escena. Si en 3 *frames* consecutivos se tiene un rostro nuevo, el rostro nuevo es aceptado como medido y se comienza el seguimiento en este nuevo rostro partiendo del último frame. Esto ayuda a evitar algunos falsos positivos que se tengan en el video.

En este caso, cuando el filtro de repeticiones acepte un nuevo rostro, todo rostro es considerado como nuevo. Este hecho genera consecuencias, ya que si un rostro es seguido y existen falsos positivos u oclusión, el sistema aún lo sigue, es decir, se usan las predicciones de Kalman aún cuando no se tengan detecciones por lo cual tenemos falsos positivos en la escena. Dado que los únicos parámetros que se consideran son las posiciones, en algunos casos en los que el rostro ocluido vuelve a aparecer en la escena, este es considerado como un nuevo rostro y se comienza un nuevo filtro de Kalman, por lo que se tienen 2 mediciones para un mismo rostro. Este problema es

solucionado mediante la asociación de datos descrito en la Sección 3.2, es decir, el seguimiento falso se corrige asociando este seguimiento a la medición más cercana, por lo que aún cuando se tenga un seguimiento falso de un rostro, cuando un nuevo rostro aparezca, el falso seguimiento se le asocia y desaparece el falso seguimiento.

Para casos de oclusión total de una persona a lo largo de todo el video, es poco posible que el detector de rostros y el filtro de Kalman puedan solucionarlo, sin embargo, se han podido resolver casos de oclusión total mediante el filtro de Kalman en casos donde el individuo aparece en la escena al menos una vez en el k-ésimo *frame* y desaparece en un j-ésimo *frame* (con $j > k$).

Para el conteo de rostros, se han puesto puertas virtuales en los cuatro bordes de los *frames* con un grosor de 15 pixeles. En casos donde la salida de personas está fuera de estas puertas virtuales, el conteo no se puede realizar, por lo que es necesario adaptar estas puertas virtuales al video. Otro problema surge cuando el seguimiento se realiza en estas puertas virtuales, por ejemplo, cuando un rostro se queda parado en estas puertas en múltiples *frames*; en este caso, se cuenta a una misma persona el número de *frames* en las que se realiza el seguimiento. Este efecto es inevitable, por lo cual es necesario restringir el sistema para casos donde el flujo peatonal es constante.

En la Tabla 5.5 podemos observar los resultados arrojados en el espacio ROC de un análisis binario (cara, no-cara) después del filtro de Kalman y en la Tabla 5.6 los resultados del conteo.

También se ha probado el sistema con seis videos tomados en el transporte colectivo METRO de la Ciudad de México. Estos son tomados en pasillos y andenes, por lo cual tienen problemas fuertes de iluminación, enfoque, resolución y flujo abundante y multi-direccional de personas. Las tomas están hechas desde arriba hacia abajo, por lo que los *frames* tienen un ángulo de inclinación. Los videos están en formato *avi* y son convertidos a formato *mp4* para tener compatibilidad con MatLab. Tienen una resolución de 352×240 pixeles, son videos en 3 canales (RGB). Ver Figura 5.6.

Para estos videos no se ha realizado un análisis ROC debido a que las condiciones no cumplen con los requerimientos para que el sistema funcione, empezando por la existencia de un flujo multi-direccional de personas. No obstante, los resultados obtenidos son alentadores, pues a pesar de tener problemas fuertes de iluminación,

Tabla 5.5: Resultados obtenidos en el espacio ROC

Parámetros	**Video P2E_S5_C2**	**Video P2L_S5_C2**	**Video PF**
Fotogramas con rostros	618	518	499
Detecciones totales	751	1082	873
Falsos positivos	145	52	62
Falsos negativos	111	73	20
Verdaderos positivos	717	1103	908
Verdaderos negativos	316	289	88
Porcentaje VP (sensibilidad)	86.59 %	93.79 %	94.58 %
Porcentaje FP (1-especificidad)	31.45 %	15.24 %	49.71 %
Precisión	0.80	0.92	0.88

Tabla 5.6: Resultados obtenidos en el conteo de personas

Parámetros	**Video P2E_S5_C2**	**Video P2L_S5_C2**	**Video PF**
Personas reales (salida)	24	25	15
Personas contadas	22	23	13
Error relativo conteo	8.33 %	8 %	13.33 %
Porcentaje de aciertos conteo	91.67 %	92 %	86.67 %

de no tener tomas frontales de rostros, de ser videos "pixeleados" y además un *background* no constante (el metro en movimiento) se logran realizar algunas detecciones y conteo de algunas personas, sin embargo, a pesar de ser el flujo muy abundante, el contador sólo detecta un número pequeño de rostros.

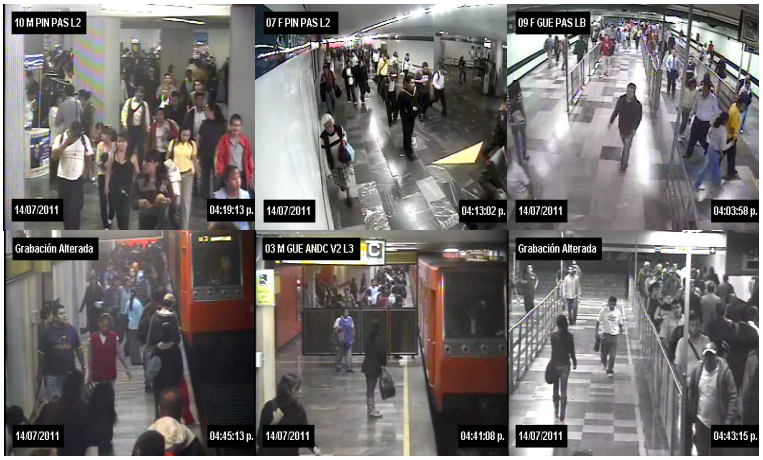

Figura 5.6: Fotogramas de los videos utilizados en el Metro de la Ciudad de México

CAPÍTULO 6

CONCLUSIONES Y TRABAJO A FUTURO

El conteo automatizado de personas es una tarea muy útil pero difícil de realizar. Si bien existen en la actualidad mecanismos automatizados de conteo, todavía se tienen problemas para el eficaz funcionamiento en situaciones de oclusión y bajo la presencia de múltiples personas en un mismo instante t. La tarea se vuelve aún más compleja cuando se realiza la localización de las personas. No obstante, se ha podido realizar dicha tarea aún bajo estas dificultades.

En este trabajo se ha propuesto como objetivo general la realización de un sistema capaz de realizar el conteo de personas en una videosecuencia con flujo peatonal moderado, el cual ha sido logrado bajo las condiciones de iluminación adecuada y resoluciones de video como las utilizadas por Viola y Jones [22] (384x288 pixeles) y R. y J. [3] (320x240 pixeles). Además, se han elaborado tres módulos abordando así cada uno de los objetivos particulares (detectar, seguir y contar personas), la parte de detección se ha logrado mediante la detección de rostros, el seguimiento y conteo se ha logrado mediante le Filtro de Kalman y la parte de asociación de datos propuesta en la Sección 3.2. Asímismo, se ha trabajado y resuelto problemas de oclusión como se propuso en uno de los objetivos particulares.

La detección de rostros funciona muy bien para videos bien iluminados así como en videos en los que se ha realizado un buen filtrado, cuyos parámetros del filtro van de acuerdo a los requerimientos y propiedades del video. Para videos en condiciones adversas, donde existe baja iluminación, donde los rostros no son precisamente

frontales, y la distancia entre el rostro y la cámara es grande, es muy difícil realizar una buena detección aún realizando un pre-procesamiento de la imagen, es por ello que una posible solución consistiría en realizar un entrenamiento para la detección de rostros en este tipo de ambientes, utilizando como imágenes de referencia los *frames* contenidos en los videos recabados en estos escenarios. Como un trabajo a futuro se propone dicho entrenamiento.

El filtro de Kalman en este trabajo es utilizado para evitar la presencia de falsos negativos, incluyendo los generados por problemas de oclusión, ya que de no ser así, en algún fotograma podría perderse la detección justo a la salida de la persona, lo que impide su conteo. Es por ello que se realiza el seguimiento y predicción de un rostro no detectado hasta que el rostro vuelva a aparecer en la escena, lo que provoca la presencia de falsos positivos en múltiples fotogramas (hasta un 49.71 % de falsos positivos), sin embargo, el conteo tiene muy buenos resultados (ver Tablas 5.5 y 5.6) por lo cual podemos asegurar que aún bajo la presencia de falsos positivos, nuestro clasificador está funcionando bien ya que la precisión está por encima de 0.8. Podemos observar también un alto porcentaje de verdaderos positivos de hasta 94.58 %, lo que nos indica que los parámetros utilizados en el suavizado y en la detección de Viola-Jones son los adecuados para una buena clasificación de rostros y no-rostros. El porcentaje de falsos positivos puede ser enormemente disminuido aprovechando el espacio temporal y proponiendo un umbral en el número de fotogramas consecutivos en los que un rostro no re-aparece en la escena, eso depende del balance que se quiera lograr entre falsos positivos y falsos negativos, en este caso se desea un mínimo de falsos negativos pues de no ser así, el conteo puede verse afectado enormemente. Sin embargo, en un trabajo cuyo único objetivo es el seguimiento, el umbral en el espacio temporal resuelve el problema de falsos positivos.

Los resultados obtenidos en el conteo son alentadores ya que pudimos obtener hasta un 92 % de aciertos, lo cual nos dice que este sistema automatizado nos da un buen estimado de el número de personas que salen en un pasillo unidireccional. Uno de los problemas en el conteo de personas es detectar personas aún cuando parpadean, bostezan, hablan, viran la cabeza o que inclusive fuman o portan accesorios en el rostro (lentes, vello facial, *piercings*, etc.), lo que dificulta la detección y por ende el conteo. En este caso se ha resuelto el problema aún en estas circunstancias donde el movimiento de las personas es natural. Además notemos que no se han obtenido

videos sujetos a las necesidades del proyecto, si no que se han tomado bases de datos disponibles a toda comunidad científica y se ha realizado el exitoso conteo en dichos videos. Adicionalmente, el trabajo no está exclusivamente calibrado a un único escenario, a diferencia de los trabajos revisados en el estado del arte, ya que ha sido probado en escenarios distintos, inclusive con resoluciones de video distintas y se han logrado excelentes resultados en la tarea de conteo de personas.

Es importante resolver primero situaciones adversas como problemas de enfoque, iluminación y ruido, para que en un futuro el sistema pueda trabajar con aglomeraciones grandes de personas. Este trabajo se propuso para realizar el conteo de personas en un video con flujo peatonal moderado (en el sentido de que las personas no causen oclusión total en los rostros), pero se ha trabajado en casos donde si existe oclusión y se muestra un buen funcionamiento.

APÉNDICE A

DEDUCCIÓN DEL FILTRO DISCRETO DE KALMAN

Supongamos que tenemos dos diferentes mediciones de la misma cantidad de interés x. La medición del primer y segundo instrumento serán llamadas x_1 y x_2 respectivamente. Además, se sabe que ambos instrumentos tienen ruido blanco el cual está modelado por Gaussianas con desviación estándar σ_1 y σ_2 en cada instrumento.

Se desea combinar ambas mediciones para obtener una sola estimación. Si $\sigma_1 = \sigma_2$, entonces entonces se estimará un promedio de x_1 y x_2. Si $\sigma_1 << \sigma_2$, se considerará a la mejor medición que en este caso es x_1 y de manera análoga, para $\sigma_1 >> \sigma_2$, se considerará a x_2. En otro caso distinto a los anteriores, se realizará un promedio pesado de ambas mediciones, dándole mayor peso a aquella medición que tenga la menor desviación estándar. Propongamos un promedio de la lectura entre su varianza:

$$\hat{x} = \frac{\frac{x_1}{\sigma_1^2} + \frac{x_2}{\sigma_2^2}}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}} = \frac{x_1\sigma_2^2 + x_2\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{A.1}$$

O bien,

$$\begin{aligned} \hat{x} &= \frac{x_1\sigma_2^2}{\sigma_1^2 + \sigma_2^2} + \frac{x_2\sigma_1^2}{\sigma_1^2 + \sigma_2^2} + x_1 - x_1 \\ &= \frac{x_1\sigma_2^2 - x_1\sigma_1^2 - x_1\sigma_2^2}{\sigma_1^2 + \sigma_2^2} + x_1 + \frac{x_2\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \end{aligned}$$

$$= \quad x_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}(x_2 - x_1)$$

$$\hat{x} = x_1 + K(x_2 - x_1) \tag{A.2}$$

donde $K = \frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}$.

De acuerdo con la suposición de la existencia de ruido blanco (comportamiento Gaussiano), podemos escribir la probabilidad de que x sea la mejor medición como:

$$p_1(x) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2}\frac{(x-x_1)^2}{\sigma_1^2}} \tag{A.3}$$

para el primer instrumento, y para el segundo instrumento la probabilidad es:

$$p_2(x) = \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{1}{2}\frac{(x-x_2)^2}{\sigma_2^2}} \tag{A.4}$$

Dado que las dos mediciones son independientes entre sí, podemos combinarlas multiplicando sus distribuciones de probabilidad y normalizando:

$$\begin{aligned}
p(x) &= p_1(x)p_2(x) = Ce^{-\frac{1}{2}\frac{(x-x_1)^2}{\sigma_1^2} - \frac{1}{2}\frac{(x-x_2)^2}{\sigma_2^2}} \\
&= Ce^{-\frac{1}{2}\left[\frac{1}{\sigma_1^2}(x^2-2xx_1+x_1^2)+\frac{1}{\sigma_2^2}(x^2-2xx_2+x_2^2)\right]} \\
&= Ce^{-\frac{1}{2}\left[\left(x^2\left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)-2x\left(\frac{x_1}{\sigma_1^2}+\frac{x_2}{\sigma_2^2}\right)\right)+\left(\frac{x_1}{\sigma_1^2}+\frac{x_2}{\sigma_2^2}\right)\right]} \\
&= Ce^{-\frac{1}{2}\left[x^2\left(\frac{\sigma_1^2+\sigma_2^2}{\sigma_1^2\sigma_2^2}\right)-2x\left(\frac{\sigma_1^2x_2+\sigma_2^2x_1}{\sigma_1^2\sigma_2^2}\right)\right]+D} \\
&= Ce^{-\frac{1}{2}\left[\frac{\sigma_1^2+\sigma_2^2}{\sigma_1^2\sigma_2^2}\left(x^2-2x\frac{\sigma_1^2x_2+\sigma_2^2x_1}{\sigma_1^2+\sigma_2^2}\right)\right]+D} \\
&= Ce^{-\frac{1}{2}\left[\frac{\sigma_1^2+\sigma_2^2}{\sigma_1^2\sigma_2^2}\left(x^2-2x\frac{\sigma_1^2x_2+\sigma_2^2x_1}{\sigma_1^2+\sigma_2^2}+\left(\frac{\sigma_1^2x_2+\sigma_2^2x_1}{\sigma_1^2+\sigma_2^2}\right)^2\right)\right]+\frac{1}{2}\left(\frac{\sigma_1^2x_2+\sigma_2^2x_1}{\sigma_1^2+\sigma_2^2}\right)^2+D} \\
&= Ce^{-\frac{1}{2}\frac{\sigma_1^2+\sigma_2^2}{\sigma_1^2\sigma_2^2}\left(x-\frac{\sigma_1^2x_2+\sigma_2^2x_1}{\sigma_1^2+\sigma_2^2}\right)^2+E}
\end{aligned}$$

$$p(x) = Fe^{-\frac{1}{2}\frac{\sigma_1^2+\sigma_2^2}{\sigma_1^2\sigma_2^2}\left(x-\frac{\sigma_1^2x_2+\sigma_2^2x_1}{\sigma_1^2+\sigma_2^2}\right)^2} \tag{A.5}$$

De la expresión (A.5) podemos observar que el valor de x más probable obtenido de la combinación de las 2 mediciones es el $\hat{x}$ obtenido en (A.1) y que la varianza de los resultados combinados es:

$$\hat{\sigma} = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2} \tag{A.6}$$

Por lo que la ganancia si está dada por:

$$K = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{A.7}$$

Por lo tanto la ecuación de $\hat{x}$ es la obtenida en (A.2) y la varianza está dada por:

$$\hat{\sigma} = (1 - K)\sigma_1^2 \tag{A.8}$$

Esta es la forma del clásico filtro de Kalman unidimensional. Actuando de manera análoga y generalizando las expresiones predecir mediciones, dado el sistema:

$$x_{k+1} = \Phi_k x_k + w_k \tag{A.9}$$

$$z_{k+1} = H_{k+1} x_{k+1} + v_{k+1} \tag{A.10}$$

donde x_k representa la medición real en el tiempo $k+1$, w_k es ruido blanco con varianza Q_k, z_k es la medición ruidosa con ruido blanco con varianza R_k. Las ecuaciones del Filtro de Kalman asociado al sistema son:

$$\hat{x}_{k+1}^- = \Phi \hat{x}_k \tag{A.11}$$

$$P_{k+1}^- = \Phi_k P_k^- \Phi_k^T + Q_k \tag{A.12}$$

donde Φ_k es la matriz de propagación del sistema, $\hat{x}_{k+1}^-$ es la primera estimación de x_{k+1}, $\Phi_k P_k^- \Phi_k^T$ es la matriz de covarianza estimado del sistema. Por otro lado es necesario asociar el valor estimado con la medición, por lo que esta asociación está dada por:

$$\hat{x}_{k+1} = \hat{x}_{k+1}^- + K_{k+1}[z_{k+1} - H_{k+1}\hat{x}_{k+1}^-] \tag{A.13}$$

$$P_{k+1} = [I - K_{k+1} H_{k+1}] P_{k+1}^- \tag{A.14}$$

Estos son los valores actualizados de las estimaciones de x_{k+1} y P_{k+1}, donde

$$K_{k+1} = P_{k+1}^- H_{k+1}^T [H_{k+1} P_{k+1}^- H_{k+1}^T + R_{k+1}]^{-1} \tag{A.15}$$

APÉNDICE B

SUAVIZADO ESPACIAL

Considere una función $f : \mathrm{R}^2 \to \mathrm{R}$ continua en el intervalo $[a, b] \times [c, d]$ la función que describe las intensidades de una imagen. Sea

$$g(x, y) = f(x, y) + \eta(x, y) \tag{B.1}$$

la versión ruidosa de la imagen, con η una variable que no depende de la intensidad en el punto (x, y) (ruido aditivo). Si $f(x, y)$ es suave y monótona en el punto (x, y) y $g(x, y)$ está en las altas frecuencias, entonces se puede obtener una aproximación $\hat{f}(x, y)$ mediante un operador que permita el paso a las bajas frecuencias y elimine las altas frecuencias, es decir, un *filtro pasa-bajas*. Un *suavizado espacial* en una imagen es aquel que no permite cambios abruptos en las intensidades de una imagen (filtro pasa-bajas).

El mecanismo de *suavizado espacial* se realiza mediante una transformación T tal que:

$$T(g) \approx g \ si \ \|g(x, y) - f(x, y)\| < \epsilon$$

En otro caso, $T(g)$ en (x, y) es estimado respecto a la combinación de las intensidades de los vecinos de (x, y).

La linealidad de un filtro depende de las propiedades de la transformación T. Un *filtro lineal* es aquel tal que T es lineal y un *filtro no lineal* es aquel cuya transformación T no es lineal.

B.1. Filtrado lineal

Si definimos la transformación T como:

$$T(g(x,y)) = (h \star g)(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(s,t)f(x-s,y-t)\partial s\partial t$$

entonces T es una transformación lineal que resulta de la *convolución* de h con g, donde h es la función que suaviza la imagen, denominada *máscara de convolución*. La convolución discreta de una imagen está dada por:

$$(h \star g)(x,y) = \sum_s \sum_t h(s,t)g(x-s,y-t) \tag{B.2}$$

Usualmente h está definida en una vecindad de tamaño $m \times n$ y en el resto de la imagen toma un valor de cero, por lo que la convolución se reduce a:

$$(h \star g)(x,y) = \sum_{-u}^{u}\sum_{-v}^{v} h(s,t)g(x-s,y-t), \qquad m = 2u+1, \quad n = 2v+1 \tag{B.3}$$

Algunos ejemplos de este tipo de Filtros son:
Filtro promediador

$$h(s,t) = \frac{1}{mn} \tag{B.4}$$

Filtro gaussiano

$$h(s,t) = \frac{e^{\frac{s^2+t^2}{\sigma^2}}}{2\pi\sigma^2} \tag{B.5}$$

Filtro promedio ponderado

$$(h \star g)(x,y) = \frac{\sum_{-u}^{u}\sum_{-v}^{v} h(s,t)g(x-s,y-t)}{\sum_{-u}^{u}\sum_{-v}^{v} h(s,t)} \tag{B.6}$$

B.1.1. Máscaras de convolución

Para evaluar el filtro en el punto (x,y) de la imagen g mediante la ecuación (B.2) se sigue el siguiente procedimiento:

1. Localizar el punto (x,y) en la imagen g.

2. Rotar la máscara de convolución h 180^o.

3. Colocar el centro de la máscara rotada h en el pixel (x, y).

4. Multiplicar el valor de la máscara rotada en cada pixel por la intensidad de la imagen y sumar estos resultados.

Ejemplo:

Considere la imagen g de tamaño 5×5.

$$g(x,y) = \begin{matrix} 17 & 24 & 1 & 8 & 15 \\ 23 & 5 & 7 & 14 & 16 \\ 4 & 6 & 13 & 20 & 22 \\ 10 & 12 & 19 & 21 & 3 \\ 11 & 18 & 25 & 2 & 9 \end{matrix} \tag{B.7}$$

y la máscara de convolución h de 3×3

$$h(s,t) = \begin{matrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{matrix} \tag{B.8}$$

Se desea evaluar el filtro lineal mediante la ecuación (B.2) en el punto $(2, 4)$. Siguiendo el procedimiento mencionado anteriormente se obtiene:

17	24	1^2	8^9	15^4
23	5	7^7	14^5	16^3
4	6	13^6	20^1	22^8
10	12	19	21	3
11	18	25	2	9

$$\hat{f}(2,4) = (1)(2) + (8)(9)(15)(4) + (7)(7) + (14)(5) + (16)(3) + (13)(6) + (20)(1) + (22)(8) = 575$$

En caso de que se desee filtrar los pixeles de los bordes, se toman varios criterios, por ejemplo, rellenar la máscara de convolución con ceros; otro caso es llenar los espacios faltantes con los valores en el lado opuesto (efecto reflejo); no filtrar estos pixeles y tomar los valores de la imagen original; o desechar los bordes. El criterio depende de la imagen, de las características que se desean obtener y de la importancia que se le desea dar a los valores de los bordes de una imagen.

B.2. Filtros no lineales

Los filtros no lineales son aquellos cuya transformación no es lineal. Existen varios criterios y el principio básico del filtrado no lineal es semejante al suavizado lineal, es decir, se desea mantener los valores en bajas frecuencias y suavizar aquellas altas frecuencias tomando como referencia las intensidades de sus vecinos en una vecindad $m \times n$. Algunos filtros son:

Sea $S_{x,y}$ la vecindad de (x, y) de tamaño $m \times n$ y $\hat{f}$ l imagen filtrada que aproxima a f.

Media geométrica

$$\hat{f}(x,y) = \left[\prod_{s,t \in S_{x,y}} g(s,t) \right]^{\frac{1}{mn}} \tag{B.9}$$

Media armónica

$$\hat{f}(x,y) = \sum_{s,t \in S_{x,y}} \frac{1}{g(s,t)} \tag{B.10}$$

Media contrarmónica

$$\hat{f}(x,y) = \frac{\sum_{s,t \in S_{x,y}} g(s,t)^{Q+1}}{\sum_{s,t \in S_{x,y}} g(s,t)^{Q}} \qquad Q \text{ es el orden del filtro} \tag{B.11}$$

Mínimo

$$\hat{f}(x,y) = \min_{s,t \in S_{x,y}} g(s,t) \tag{B.12}$$

Punto medio

$$\hat{f}(x,y) = \frac{1}{2} \left[\min_{s,t \in S_{x,y}} [g(s,t)] + \max_{s,t \in S_{x,y}} [g(s,t)] \right] \tag{B.13}$$

Mediana

$$\hat{f}(x,y) = M_{es,t \in S_{x,y}} g(s,t) \tag{B.14}$$

El filtro de la *mediana* es sumamente socorrido debido a su capacidad de reducción de ruido sin suavizar demasiado la imagen.

APÉNDICE C

ACERCA DE OPENCV

Visión por computadora es la transformación de un fotograma o video a una nueva representación, como pasar una imagen de color a escala de grises o remover el movimiento en un fotograma. Dicha transformación es para lograr alguna meta, por ejemplo, la reducción de información para un fácil y rápido acceso.

Dado que los seres humanos somos criaturas visuales, suele creerse que la visión por computadora es una tarea fácil, sin embargo es una tarea que requiere de gran procesamiento y conocimiento para poder realizarse.

OpenCv (Open Source Computer Vision) es una librería de funciones de programación en código abierto para la visión por computadora en tiempo real. OpenCv tiene licencia libre para uso académico y comercial. Está desarrollado para C++, C y Phyton sobre Linux, Adroid, Window y Mac OS y existen desarrollos activos para funcionar en Ruby, MatLab y otros lenguajes, con más de 2500 algoritmos optimizados (disponible en Bradsky [30]). Actualmente MatLab contiene algunas de las librerías para la visión por computadora.

Una de las principales metas de OpenCv es otorgar una infraestructura de simple utilización para que las personas puedan construir sofisticadas aplicaciones de visión por computadora de forma rápida y sencilla. Debido a que el aprendizaje de la máquina y la visión por computadora a menudo van de la mano, OpenCv contiene librerías de entrenamiento y aprendizaje.

Inicialmente fue desarrollado por Intel para avances de aplicaciones intensivas de

CPU. OpenCv fue construido como una manera para hacer visión por computadora universalmente viable. Tardaron siete años (1999-2006) para lanzar y desarrollar la primera versión oficial de la librería (Bradski y Kaehler [36]).

Entre las múltiples funciones que contiene, OpenCv dispone de archivos en fomato *html* cuyo contenido es el entrenamiento para la clasificación en cascada usando el algoritmo de Viola-Jones (Viola y Jones [25]) para la detección de ojos, boca, cuerpo superior frontal, rostros frontal y rostros de perfil.

APÉNDICE D

CÓDIGO FUENTE FUNCIÓN *FACEDETECT.MEX*

La función de detección de rostros está basado en la colaboración de Sreekar Krishna obtenida de la página de MatLab Central[1]. La función es un archivo tipo *mex* cuyas funciones son llamadas de OpenCv.

```
        Entradas: 1. Location of the XML file that contains the Haar cascade
2. Image in which face needs to be located (Gray image only)
Salidas: Matriz NxM
        N es el número de rostros detectado
M = 4; 1: X posición de la cara
2: Y posición de la cara
3: ancho del rostro
4: altura del rostro
        El programa hace la comprobación de errores en
1. número de entradas
2. la imagen está en escala de grises y en formato doble
        Author: Sreekar Krishna
sreekar.krishna@asu.edu
        Version 1.0
Created: May 13 2008

#include "mex.h" // Para archivos MEX
// Para OpenCV
#include "cv.h"
static CvMemStorage* storage = 0;
static CvHaarClassifierCascade* cascade = 0;
void mexFunction(int output_size, mxArray *output[], int input_size, const mxArray *input[])
{
    char *input_buf;
    int buflen;
    mxArray *xData;
    double *xValues;
    int i,j;
    int NoOfCols, NoOfRows;
    /* Verificar el número de argumentos */
```

[1] ttp://www.matworks.com/matlabcentral/fileexchange/19912-open-cv-viola-jones-face-detection-in-matlab

```
if(input_size!=2)
  mexErrMsgTxt("Usage: FaceDetect (<HaarCascade File>, <GrayImage>)");
/* la entrada debe ser una cadena */
if ( mxIsChar(input[0]) != 1)
  mexErrMsgTxt("Input 1 must be a string.");
/* obtener la longitud de la cadena */
buflen = (mxGetM(input[0]) * mxGetN(input[0])) + 1;
/* copiar de los datos de cadena de input[0] en una cadena C input_ buf.     */
input_buf = mxArrayToString(input[0]);
if(input_buf == NULL)
  mexErrMsgTxt("Could not read HarrCascade Filename to string.");
// leer la Haar Cascada del archivo XML
cascade = (CvHaarClassifierCascade*) cvLoad( input_buf, 0, 0, 0);
if( !cascade )
{
   mexErrMsgTxt("ERROR: Could not load classifier cascade" );
}
/* verificar si la imagen de entrada está en formato doble*/
if (!(mxIsDouble(input[1]))) {
  mexErrMsgTxt("Input image must be gray scale and of type double");
}
//Copiar apuntador de entrada
// Esto lleva la imagen de entrada en escala de grises que fue enviada desde MatLab
xData = (mxArray *)input[1];
//Obtiene la matriz de los datos de entrada
// La matriz se rasteriza en una columna
xValues =  mxGetPr(xData);
NoOfCols = mxGetN(xData); // Obtiene el número de columnas en la imagen
NoOfRows = mxGetM(xData); // Obtiene el número de filas en la imagen
/* Obtiene el número de dimensiones en la imagen de entrada */
int number_of_dims = mxGetNumberOfDimensions(input[1]);
if (number_of_dims > 2)
    mexErrMsgTxt("Input image should be gray scale and of type double");
// Crea una IplImage de los datos así la detección de rostros puede ser corrida
IplImage* gray = cvCreateImage( cvSize(NoOfCols, NoOfRows), IPL_DEPTH_8U, 1 );
// Carga la columna de vectores en IplImage
// IplImage es leida en una fila
// Appropriate conversion is carried out here
for(i=0;i<NoOfCols;i++)
   for(j=0;j<NoOfRows;j++)
    {
        int value = xValues[(i*NoOfRows)+j];
        gray->imageData[j*NoOfCols+i] = value;
    }
/*********************************************************************
  * Existe un error en OpenCV, si uno llama a la función cvLoad antes de llamar cualquier otra función de la
  * librería cxCore, se produce un error por la función cvRead que forma parte de cvLoad. Para resolver
  * este error cualquier función de la librería cxcore debe ser llamada. Aquí creamos una imagen ficticia
  * de 11x11 píxeles y erosionar la imagen con un pequeño núcleo.
 *********************************************************************/
IplImage* dummy = cvCreateImage( cvSize(11, 11), IPL_DEPTH_8U, 1 ); // Make a dummy image
IplConvKernel* se = cvCreateStructuringElementEx(3,3,1,1,CV_SHAPE_ELLIPSE); // Create a filter
cvErode(dummy,dummy,se); // Erode
cvReleaseImage( &dummy );
/*****************************************************************
 *                          Detector de rostros                  *
 *****************************************************************/
// Requiredo en el proceso de detección de rostros
storage = cvCreateMemStorage(0);
```

```
    cvClearMemStorage( storage );
    // Detección de rostros con factor de escalamiento 1.1, 2 traslapes y 24x24 tamaño de ventana inicial
    CvSeq* faces = cvHaarDetectObjects( gray, cascade, storage,
                                        1.1, 2, 0/*CV_HAAR_DO_CANNY_PRUNING*/,
                                        cvSize(24, 24) );
    // Asignar variavle de salida
    // Número de filas = número de detecciones
    // Número de columnas = 4
    //  1: Localización de la cara X
    //  2: Localización de la cara Y
    //  3: Ancho de la cara
    //  4: Altura de la cara
    double *Data;
    if (faces->total)
    {
        output[0] = mxCreateDoubleMatrix(faces->total, 4, mxREAL);
        Data = mxGetPr(output[0]); // Obtener el apuntador de la variable de salida
        // Iterar por cada uno de los rostros
        for( i = 0; i < faces->total; i++ )
        {
            CvRect* r = (CvRect*)cvGetSeqElem( faces, i );
           /* La primer columna contendrá la localización x de todas las caras
            * La segunda columna contendrá la localización y de todas las caras */
           Data [i] = r->x;
           Data [i+faces->total] = r->y;
           Data [i+faces->total*2] = r->width;
           Data [i+faces->total*3] = r->height;
           // Depurar
           // printf ("%d %d %d %d\n", r->x, r->y, r->width, r->height);
        }
    }
    else
    {
        output[0] = mxCreateDoubleMatrix(1, 1, mxREAL);
        Data = mxGetPr(output[0]); // Obtener el apuntador de la variable de salida
        Data[0] = -1;
    }
    // Liberar toda la memoria
    cvReleaseImage( &gray );
    return;
}
```

APÉNDICE E

CÓDIGO DEL SISTEMA PROPUESTO

```
mov=mmreader(dir); %leer pelicula donde dir=path
%propiedades de la película
numFrames = mov.NumberOfFrames;
height=mov.Height;
width=mov.Width;
%iniciar conteo de tiempo de procesamiento
tic

%Definir variables del filtro de Kalman
deltat=1/mov.FrameRate;
Q=0.01*eye(12,12);
R=10*eye(4);
Phi=[1, deltat, 0.5*deltat^2, 0, 0, 0,0,0,0,0,0,0
     0, 1, deltat, 0, 0, 0,0,0,0,0,0,0
     0, 0, 1, 0, 0, 0,0,0,0,0,0,0
     0, 0, 0, 1, deltat, 0.5*deltat^2,0,0,0,0,0,0
     0, 0, 0, 0, 1, deltat,0,0,0,0,0,0
     0, 0, 0, 0, 0, 1,0,0,0,0,0,0
     0,0,0,0,0,0,1,deltat, 0.5*deltat^2,0,0,0
     0,0,0,0,0,0,0,1,deltat,0,0,0
     0,0,0,0,0,0,0,0,1,0,0,0
     0,0,0,0,0,0,0,0,0,1,deltat,0.5*deltat^2
     0,0,0,0,0,0,0,0,0,0,1,deltat
     0,0,0,0,0,0,0,0,0,0,0,1];
H=[1, 0, 0, 0 ,0, 0, 0, 0, 0, 0 ,0, 0
   0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0
   0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
```

```
    0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0];
 P=eye(12)*1e4;

P=Phi*P*Phi'+Q;
K=(P*H')/((H*P*H')+R);
mult=K*H;
P=(eye(size(mult))-mult)*P;

%inicializar variables
ka=0;
l=0;
people=0;
flag=0;
movi(1:numFrames) = struct('cdata', [],'colormap', []);

%=========================================================================
%%Deteccion y segimiento
%=========================================================================
%Procesamiento en el i-esimo frame
for i=1:numFrames
    images = read(mov,i); %Leer imagen
    Img = rgb2gray(images); %convertir a escala de grises
    Img=medfilt2(Img,[7 7]); %filtrar con filtro de la maediana
    Face=FaceDetect('haarcascade_frontalface_alt2.xml',double(Img)); %Realizar detecciones
    [m n]=size(Face); %medir número de detecciones
    if Face==-1%Si Face=-1 significa que no hay detecciones
        m=0;
        Face=zeros(1,4);
    else
        flag=flag+1; %Cuando haya al menos una detección inicia la bandera de seguimiento
        if flag==1
            m=0;
            Face=zeros(1,4);
        elseif flag==2%Iniciar g_0
            gold=[Face(:,1), zeros(m,1), zeros(m,1) Face(:,2) zeros(m,1) zeros(m,1) Face(:,3)+Face(:,1),
        end
    end
    if flag>=2%Iniciar seguimiento a partir de 2 frames consecutivos con detecciones


        mold=size(gold,1);
        g=zeros(mold,12); %cargar g en memoria
```

```
xj=zeros(mold,4);
for j=1:mold
    g(j,:)=(Phi*gold(j,:)')'; %Calcular g partiendo del frame anterior con Kalman
    xj(j,:)=(H*g(j,:)')';
end
z=[Face(:,1:2), Face(:,1:2)+Face(:,3:4)]; %Calcular z

%Establecer asociación de datos
dist=zeros(m,mold);
for j=1:m
    for k=1:mold
        dist(j,k)=norm(xj(k,:)-z(j,:)); %%norm(z(j,:));
    end
end
if m>1
      [c,index]=min(dist); %%ndex: posición:xj indice=z
elseif m==1;
      c=dist;
      index=ones(1,mold);
end

%Nuevos rostros en la escena
if mold<m
   l=l+1;
   if l==3 %si existen nuevos rostros en 3 frames consecutivos inicia nuevo seguimiento
       l=0;%%nicia conteo de frames consecutivos
       mnew=mold;
       for j=1:m
           if index~=j
               mnew=mnew+1;
               g(mnew,:)=[z(j,1),zeros(1,2),z(j,2),zeros(1,2),z(j,3),zeros(1,2),z(j,4),zeros(
               xj(mnew,:)=(H*g(mnew,:)')';
               index(mnew)=j;
               c(mnew)=0;
           end
       end
       m=mnew;
   else
       m=mold;
   end;
end

%Mismo número de rostros que el frame anterior
```

```
if m==0
    z=xj;
    elim=zeros(1,mold);
    for j=1:mold
        if (any(xj(j,:)<15))||(xj(j,3)>width-15||xj(j,4)>height-15) %Revisa si no hay entrada y s
            elim(j)=1;
        end
    end
    z(elim==1,:)=[];
    m=size(z,1);
    index=1:m;

% Nuevo rostro o salida de personas
else
    mnew=m;
    for j=1:mold
        if sum(index==j)>1 %Elementos asociados a la misma medición
            var=c;
            var(index~=j)=5000000;
            [r,quedado]=min(var); %Elegir el más cercano a la medición
            indexnew=index;
            xjnew=xj;
            xjnew=xjnew(indexnew==j,:);
            indice=zeros(size(indexnew));
            indice(quedado)=j;
            indice=indice(indexnew==j);
            for k=1:sum(index==j);
                if indice(k)==0 %veridfica si hay salidas
                    if (all (xjnew(k,:)>15))&&(xjnew(k,3)<width-15 && xjnew(k,4)<height-15) %pue
                        mnew=mnew+1;
                        z(mnew,:)=xjnew(k,:); %añadir mediciones de Kalman
                        indice(k)=mnew;
                    end
                end
            end
            indexnew(indexnew==j)=indice;
            xj(indexnew==0,:)=[];
            g(indexnew==0,:)=[];
            indexnew(indexnew==0)=[];
            index=indexnew;

        end
```

```
            end
            m=numel(index);
        end

        if m<mold
            people=people+(mold-m); %Conteo de personas que salen
            set(handles.counter,'String',num2str(people));
        end


        clear gold mnew dist var
        %Convertir la nueva medición en vieja para el nuevo frame
        if m>0
            for j=1:m
                gold(j,:)=(g(j,:)'+K*(z(index(j),:)'-H*g(j,:)'))';
                Fac(j,:)=(H*gold(j,:)')';
            end
        Fac=[Fac(:,1:2), Fac(:,3:4)-Fac(:,1:2)];

              ka=ka+1;
%Desplegar detecciones en pantalla
                DisplayDetections(images,Fac);

                nFrames = mov.NumberOfFrames;
                vidHeight = mov.Height;
                vidWidth = mov.Width;
        else
            flag=0;
        end

    end

                if m~=0
                    movi(i)=(getframe);
                else
                    imshow(images)
                    movi(i)=(getframe);
                end
 %Limpiar variables

            clear g Fac Face c index m mold  xj z cont erase
```

```
end
toc
```

BIBLIOGRAFÍA

[1] E. Hjelmȧs y B. K. Low, "Face detection: A survey," Computer Vision and Image Understanding, pp. 236–274, Apr. 2001.

[2] H. A. Rowley, S. Baluja, y T. Kanade, "Neural network-based face detection," en Transactions On Pattern Analysis and Machine intelligence, vol. 20. IEEE, 1998, pp. 23–38.

[3] L. R. y M. J., "An extended set of haar-like features for rapid object detection," en International Conference on Image Processing 2002, vol. 1. IEEE, Sep 2002, pp. 900–903.

[4] A. Yilmaz, O. Javed, y M. Shah, "Object tracking: A survey," ACM Comput. Surv., vol. 38, no. 4, page 13, Dec 2006.

[5] O. Masoud y N. P. Papanikolopoulos, "A novel method for tracking and counting pedestrians in real-time using a single camera," en Trans. on Vehicular Tech., vol. 50 no. 5. IEEE, 2001, pp. 1267–1278.

[6] D. B. Yang, H. H. González-Baños, y L. J. Guibas, "Counting people in crowds with a real-time network of simple image sensors," en International Conference on Computer Vision (ICCV'03), vol. 1. IEEE, Oct 2003, pp. 122–129.

[7] T. Zhao y R. Nevatia, "Stochastic human segmentation from a static camera," en Workshop on Motion and Video Computing (MOTION'02). IEEE, Dec 2002, pp. 9–14.

[8] Q. Cai y J. K. Aggarwal, "Tracking human motion using multiple cameras," en Proc. 13th Int. Conf. Pattern Recognition, vol. 3. IEEE, Aug. 1996, pp. 68–72.

[9] S. H., J. Tao, y Y.-P. Tan, "People counting by video segmentation and tracking," en International Conference on Control, Automation, Robotics and Vision. ICARCV '06. IEEE, Dec 2006, pp. 1–4.

[10] J.-W. Kim, K.-S. Choi, B.-D. Choi, y S.-J. Ko, "Real-time vision-based people counting system for the security door," en Proc. of 2002 International Technical Conference On Circuits Systems Computers and Communications, Phuket, Jul. 2002, pp. 1418–1421.

[11] X. Zhao, E. Delleandrea, y L. Chen, "A people counting system based on face detection and tracking in a video," en Proceedings of the 2009 Sixth IEEE international Conference on Advanced Video and Signal Based Surveillance AVSS. Washington, DC: IEEE Computer Society, 2009, pp. 67–72.

[12] M.-H. Yang, D. Kriegman, y N. Ahuja, "Detecting faces in images: a survey," Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 1, pp. 34–58, Jan 2002.

[13] R. C. González y R. E. Woods, Digital Images Processing, 2nd ed. Edit Prentice Hall, 2002.

[14] E. V. C. Jimenez y D. Z. Navarro, "Visión por computador utilizando matlab y el toolbox de procesamiento digital de imágenes," –, tutorial. [Online]. Available: http://proton.ucting.udg.mx/tutorial/vision/cursovision.pdf

[15] I. Craw, H. Ellis, y J. R. Lishman, "Automatic extraction of face-feature," en Pattern Recognition Letters, vol. 5 no. 2. Elsevier, Feb. 1987, pp. 183–187.

[16] G. P. Martínez y J. M. de la Cruz García, Visión por computador. Imágenes digitales y aplicaciones, 2nd ed. Alfaomega, 2008.

[17] N. Otsu, "A threshold selection method from gray-level histograms," en Transactions on Systems, Man, and Cybernetics, vol. 9 no. 1. IEEE, 1979, pp. 62–66.

[18] E. Viennet y F. F. Soulié, "Connectionist methods for human face processing," en Face Recognition: From Theory to Application. Berlin/New York: Springer-Verlag, 1998, pp. 1–33.

[19] S. Z. Li y A. K. Jain, Handbook of Face Recognition, 2nd ed. Springer, 2011.

[20] C. Zhang y Z. Zhang, "Boosting-based face detection and adaptation," 2010, synthesis Lectures on Computer Vision, Morgan and Claypool.

[21] A. Rama y F. Tarrés, "Un nuevo método para la detección de caras basado en integrales difusas," en XXII Simposium Nacional de la Unión Científica Internacional de Radio, L. de Resúmenes, Ed., 2007, pp. 1–4.

[22] P. Viola y M. Jones, "Robust real-time object detection," en Second International Workshop On Statistical And Computational Theories Of Vision - Modeling, Learning, Computing, And Sampling, 2001, pp. 1–25.

[23] R. Caruana y A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," en Proceeding ICML '06 Proceedings of the 23rd international conference on Machine learning. ACM New York, NY, USA ©2006, Aug. 2006, pp. 161–168.

[24] C. Zhang y Z. Zhang, "A survey of recent advances in face detection," Microsoft Research, Technical Report MSR-TR-2010-66, Jun 2010.

[25] P. Viola y M. Jones, "Robust real-time face detection," en International Journal of Computer Vision, vol. 57 no. 2. Kluwer Academic Publishers Hingham, MA, USA, May 2004, pp. 137–154.

[26] ——, "Rapid object detection using a boosted cascade of simple features," en Conference On Computer Vision And Pattern Recognition, vol. 1. IEEE, 2001, pp. 511–518.

[27] Z. S. Tabatabaie, R. W. Rahmat, N. I. B. Udzir, y E. Kheirkhah, "A hybrid face detection system using combination of appearance-based and feature-based methods," en International Journal of Computer Science and Network Security, vol. 9 no. 5, 2009, pp. 181–185.

[28] P. Y. Simard, L. Bottou, P. Haffner, y Y. L. Cun, "Boxlets: a fast convolution algorithm for signal processing and neural networks," en Proceedings of the 1998 conference on Advances in neural information processing systems II, I. M. Kearns, S. Solla, y D. Cohn, Eds., vol. 11. MIT Press Cambridge, MA, USA, 1999, pp. 571–577.

[29] R. Lienhart, A. Kuranov, y V. Pisarevsky, "Empirical analysis of detection cascades of boosted classifiers for rapid object detection," Intel Labs, MRL Technical Report, May 2002.

[30] G. Bradsky, "Opencvwiki," May 2012. [Online]. Available: http://opencv.willowgarage.com/wiki/Welcome

[31] E. Brookner, Tracking And Kalman Filtering Made Easy, 1st ed. John Wiley & Sons, Inc, 1998.

[32] C. H. Chan, "The extended m2vrs database," M2VTSD, Oct 2012. [Online]. Available: http://www.ee.surrey.ac.uk/CVSSP/xm2vtsdb/

[33] E. Maggio, "Spevi," Feb. 2007. [Online]. Available: http://www.spevi.org

[34] Y. Wong, "Chokepoint dataset," NICTA, 2011. [Online]. Available: http://itee.uq.edu.au/~uqywong6/chokepoint.html#overview

[35] C. VideoMuseumOfCityPeople, "Youtube istanbul, turkey," Nov 2006. [Online]. Available: http://www.youtube.com/watch?v=lFYcZ3ppvAw&feature=related

[36] G. Bradski y A. Kaehler, Learning OpenCv, 1st ed. O'Reilly, 2008.