UNIVERSIDAD DE VALLADOLID

ESCUELA TÉCNICA SUPERIOR

DE INGENIEROS DE TELECOMUNICACIÓN

PROYECTO FIN DE CARRERA

INGENIERO TÉCNICO DE TELECOMUNICACIÓN.

SISTEMAS DE TELECOMUNICACIÓN

DESARROLLO DE UNA APLICACIÓN DE DETECCIÓN DE MOVIMIENTO BASADA EN COMPARATIVA ESTRUCTURAL DE IMÁGENES

AUTOR: RAFAEL ORMAECHEA IZQUIERDO

TUTOR: SANTIAGO AJA FERNÁNDEZ

Septiembre de 2012

DESARROLLO DE UNA APLICACIÓN DE DETECCIÓN DE MOVIMIENTO BASADA EN COMPARATIVA ESTRUCTURAL DE IMÁGENES

AUTOR: RAFAEL ORMAECHEA IZQUIERDO

TUTOR: SANTIAGO AJA FERNÁNDEZ

TÍTULO:	Desarrollo de una aplicación de detección de movimiento basada en comparativa estructural de imágenes.
AUTOR:	Rafael Ormaechea Izquierdo
TUTOR:	Santiago Aja Fernández
DEPARTAMENTO:	Teoría de la Señal y Comunicaciones e Ingeniería Telemática

Miembros del Tribunal

PRESIDENTE:

SECRETARIO:

VOCAL:

FECHA DE LECTURA:

CALIFICACIÓN:

RESUMEN DEL PROYECTO

La creciente necesidad de seguridad ha propiciado el desarrollo en los últimos años de muy variados sistemas de vigilancia. En este sentido muchos han sido los avances que se han llevado a cabo en el campo del procesado de señales para hacer más eficiente a los sistemas de seguridad.

En el siguiente proyecto presentamos el desarrollo de una aplicación de detección de movimiento cuyo funcionamiento está basado en la comparativa estructural de imágenes, y que puede aplicarse a nuevos dispositivos móviles de pequeño tamaño.

El método que se ha empleado para la detección consiste en la aplicación de una medida de calidad de imagen, la similitud estructural, que nos permite determinar las diferencias existentes entre dos imágenes. Una vez presentada la nueva propuesta, se han realizado diversas pruebas para estudiar el buen funcionamiento del método desarrollado.

PALABRAS CLAVE

Sistema de vigilancia, Detección de movimiento, Comparación estructural, SSIM.

ABSTRACT

The growing need of security has led to the development of a great number of video surveillance systems. In relation to this idea, many advances have been made in the field of signal processing in order to make security systems more efficient and less dependent of human surveillance.

In the former paper we introduce the development of a motion detection application, which performance is based on image structural comparison, and that can be applied to new small size mobile devices.

The method used for detection is based on the application of an image quality measurement, structural similarity, which allows us to determine the existing differences between two images. Once this new approach is introduced, several tests have been made in order to study the successful performance of the developed method.

KEY WORDS

Video surveillance system, Motion detection, Structural comparison, SSIM

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

INTRODUCCIÓN

1.1. INTRODUCCIÓN

En los últimos años el número de técnicas de procesado de señal han sufrido un crecimiento exponencial, debido principalmente a las grandes posibilidades que ofrecen los nuevos dispositivos electrónicos al alcance de un mayor número de usuarios y a la gran importancia que han alcanzado en el mundo actual el tratamiento de imagen y video.

En concreto, acorde con la creciente necesidad de vigilancia y seguridad tanto en espacios públicos como privados, se ha incrementado la investigación en técnicas y procedimientos para desarrollar nuevos sistemas de videovigilancia más eficientes.

La videovigilancia, que se puede definir como el conjunto de técnicas empleadas para detectar, reconocer o seguir objetos de manera autónoma a partir de secuencias de imágenes obtenidas por una cámara, encuentra sus principales aplicaciones, por ejemplo, en la detección de intrusos en zonas de seguridad, el control de tráfico en carreteras, aduanas, puertos y aeropuertos, vigilancia de procesos industriales, control de multitudes… [6, 7, 8,10].

Aparte de los sistemas “profesionales” o comerciales de videovigilancia basados en circuitos cerrados de televisión y cámaras IP, también es posible disponer de sistemas más sencillos para el uso personal. En conjunción con la aparición de numerosas aplicaciones para teléfonos móviles y dispositivos de última generación, el desarrollo de nuevas aplicaciones basadas en detección de movimiento que puedan implementarse sobre dispositivos móviles abre una amplia línea de trabajo.

En este sentido, ya existen diversas aplicaciones disponibles como por ejemplo tinyCam Monitor o IP Cam Viewer, que permiten gestionar una red de cámaras. Otras, como Motion Detector Pro, son capaces de detectar presencia [11]. En ese sentido,

nos proponemos realizar una aplicación que sea capaz de detectar presencias o movimientos de objetos.

Existen variedad de técnicas para el desarrollo de aplicaciones de detección de movimiento y seguimiento de objetos, cada una con sus ventajas e inconvenientes que los hacen apropiados para situaciones determinadas. La literatura relativa a este tipo de procesado es muy amplia, lo cual nos permite tener diferentes alternativas. Dentro de la detección de objetos disponemos de varias técnicas, como por ejemplo modelado y sustracción de fondo (Gaussian mixture model, dynamic textures) y *video segmentation* (Flujo óptico, mean shift). Para seguimiento de objetos, existen métodos como la correspondencia de puntos, la coincidencia de regiones geométricas o evolución del contorno [9].

En este trabajo, sin embargo, no vamos a emplear ninguno de los métodos anteriormente mencionados para implementar nuestra aplicación, sino que vamos a partir de otro método basado en la similitud estructural para detectar los posibles cambios que se producen entre dos imágenes consecutivas de la secuencia de vídeo [1,2].

1.2. OBJETIVOS

El principal objetivo del proyecto consiste en el desarrollo de una aplicación que permita realizar operaciones de detección de movimiento y seguimiento en grabaciones de video, que sea robusta y ligera para un su posterior implementación en dispositivos móviles con cámara integrada.

A partir del objetivo principal podemos concretar los siguientes subobjetivos:

1. Desarrollo de un método ligero que permita detectar movimiento y sea robusto a cambios de iluminación y pequeños movimientos de cámara.
2. Confección de una base de datos formada por vídeos de diversa índole para comprobar el funcionamiento de la aplicación y para que esté disponible en un uso futuro.
3. Realizar un estudio de la idoneidad de la aplicación para la detección de movimientos en distintos entornos, tanto de interior como de exterior, en diferentes situaciones de iluminación y ante distintos tipos de movimientos.

1.3. FASES Y MÉTODOS

Para el desarrollo del trabajo y la consecución de los anteriores objetivos, se han seguido los siguientes puntos:

1. Planteamiento del problema. Estudio de la literatura relacionada con el problema de la detección y el seguimiento de objetos en movimiento. Análisis de las diversas técnicas y métodos disponibles.
2. Desarrollo de un prototipo de la aplicación. Implementación del algoritmo que permita conseguir los objetivos propuestos. Para ello se empleará como base el software Matlab.
3. Entrenamiento y validación de la aplicación. El entrenamiento se realiza mediante una serie de vídeos prueba que contienen las características básicas que queremos evaluar para determinar el comportamiento inicial del programa.
4. Corrección de problemas y refinamiento de la aplicación. Solución de los problemas detectados durante la fase de entrenamiento.
5. Evaluación del funcionamiento final de la aplicación. Para esta fase se empleará un conjunto de vídeos variados en distintos entornos y condiciones.
6. Recopilación y estudio de los resultados obtenidos. El análisis de los resultados permite extraer las conclusiones finales y determinar si el método empleado es apropiado.

CAPÍTULO 2

ESTADO DEL ARTE: SISTEMAS Y TÉCNICAS DE VIGILANCIA

2.1. SISTEMAS DE VIGILANCIA

En este apartado vamos a comentar la evolución de los sistemas de vigilancia. Según Valera y Velastin [6], podemos clasificar los sistemas de vigilancia en tres generaciones, dependiendo de la tecnología que emplean y las funcionalidades que proporcionan.

La evolución tecnológica de los sistemas de videovigilancia comenzó con los sistemas de CCTV (sistemas de circuito cerrado de televisión). Estos sistemas consisten en un número de cámaras situadas en múltiples localizaciones remotas y conectadas a un conjunto de monitores, normalmente ubicados en una habitación de control.

Actualmente, la mayoría de los sistemas CCTV usan técnicas analógicas para la distribución y almacenamiento de imágenes. Las cámaras convencionales de CCTV normalmente usan dispositivos de carga acoplada (CCD) digitales para capturar las imágenes, pero también cámaras térmicas y de visión nocturna. Luego, la imagen digital se transforma en una señal de vídeo compuesta y analógica que se conecta a la matriz CCTV, a los monitores y a los equipos de grabación mediante cable coaxial.

Sin embargo, el hecho de tratarse de tecnología totalmente analógica implica algunos problemas, ya que la conversión digital-analógica provoca cierta degradación en la imagen y la señal analógica es por si misma susceptible al ruido. Otro problema es que estos sistemas dependen totalmente de un operario que debe encargarse personalmente de la vigilancia de todo el panel de control, tarea que se complica al aumentar el número de cámaras del sistema. Sin embargo, estos sistemas CCTV son muy utilizados en muchos ámbitos, debido a su sencillez de funcionamiento. Una mejora de los sistemas CCTV consiste en incluir tecnología digital. Estos sistemas CCTV más modernos basados en el uso de cámaras IP conectadas a una red permiten

mayores prestaciones, como almacenamiento digital, facilidad de uso y de integración en otros sistemas…

Por tanto, es posible disponer de sistemas CCTV digitales, tomando ventaja del formato digital inicial de las imágenes capturadas, usando ordenadores de altas prestaciones y combinándolos con tecnología de visión artificial. La mejora tecnológica proporcionada por estos sistemas ha llevado al desarrollo de sistemas semiautomáticos, conocidos como sistemas de segunda generación. La mayor parte de la investigación en sistemas de videovigilancia de segunda generación se basa en el desarrollo de algoritmos para la detección automática y en tiempo real de eventos, ayudando al usuario a reconocer dichos sucesos.

Los llamados sistemas de vigilancia de tercera generación hacen referencia a los sistemas capaces de manejar un gran número de cámaras, muchos puntos de vigilancia, recursos dispersos geográficamente y reflejar la naturaleza jerárquica y distribuida del proceso humano de vigilancia. Desde el punto de vista del procesado de imagen, estos sistemas se basan en la distribución de las capacidades de procesado sobre la red, y en caso de dispositivos de procesado de señal integrados, deben proporcionar las ventajas de escalabilidad propias de los sistemas distribuidos. Los principales objetivos que los sistemas de vigilancia de tercera generación deben procurar son: proporcionar un buen entendimiento de la escena y atraer la atención del operador responsable en tiempo real hacia las situaciones de mayor importancia.

En líneas generales, el objetivo último de las generaciones de sistemas de vigilancia consiste en diseñar sistemas inteligentes capaces de sustituir a la videovigilancia tradicional basada en la observación pasiva de imágenes, que es ineficiente cuando el número de cámaras excede la capacidad del operario de controlarlas, y capaces de analizar y comprender los comportamientos de personas en secuencias de vídeo.

A continuación se recogen algunas de las características principales de las generaciones de sistemas de vigilancia

1º generación

Técnicas	Sistemas analógicos CCTV.
Ventajas	Proporcionan buenas prestaciones en algunas situaciones. Emplean una tecnología que ya es conocida y madura.
Problemas	Uso de tecnología analógica para el almacenamiento y distribución de las imágenes.
Avances posibles	Tecnología digital vs. analógica. Grabación de vídeo digital Compresión de vídeo CCTV.

2º generación

Técnicas	Videovigilancia automatizada combinando tecnología de visión por computadora con sistemas CCTV.
Ventajas	Incrementa la eficiencia de la vigilancia de los sistemas CCTV.
Problemas	Se requieren robustos algoritmos de detección y seguimiento para el análisis de comportamiento y conducta.
Avances posibles	Algoritmos de visión artificial robustos en tiempo real. Aprendizaje automático de variabilidad en la escena y de modelos de comportamiento.

3º generación

Técnicas	Sistemas de vigilancia automatizados de área amplia.
Ventajas	Información más precisa como resultado de combinar diferentes tipos de sensores. Distribución.
Problemas	Distribución de la información. Metodología de diseño/diseño de la metodología. Plataformas multi-sensores.
Avances posibles	Inteligencia distribuida frente a centralizada. Fusión de datos. Esquema de razonamiento probabilístico. Técnicas de vigilancia multicámara.

Tabla 2.1. Evolución de los sistemas de vigilancia

2.1.1. APLICACIONES

La creciente demanda de seguridad por parte de la sociedad conduce a la necesidad de actividades de vigilancia en muchos entonos. Actualmente, la demanda de vigilancia remota por motivos de seguridad ha recibido especial atención, sobre todo en los siguientes campos:

- Aplicaciones de transporte, como aeropuertos, entornos marítimos, vías de ferrocarril y metro, autovías, donde se utiliza para medir el volumen de tráfico y reunir en tiempo real información sobre el tráfico.
- Reconocimiento basado en el movimiento, esto es, identificación de personas basado en su modo de andar, detección automática de objetos…
- Navegación de vehículos, es decir, emplear rutas adquiridas por vídeo y capacidades para evitar obstáculos.

- Lugares públicos como bancos, supermercados y aparcamientos, y también hogares privados.
- Vigilancia remota de actividades humanas, como asistencia a partidos de fútbol, manifestaciones…
- Vigilancia para obtener cierto control de calidad en procesos industriales.

2.2. TÉCNICAS EMPLEADAS EN SISTEMAS DE VIGILANCIA

En este apartado vamos a comentar las diferentes etapas que conforman un sistema inteligente de videovigilancia, según se definen en la bibliografía. Según [6] y [12] las etapas más comunes son: detección y reconocimiento de objetos, seguimiento de objetos, análisis de comportamiento y almacenamiento en base de datos.

- Detección de objetos: Es el primer paso en la mayoría de sistemas. Consiste en determinar y separar las regiones que corresponden con objetos en movimiento del resto de la imagen. Este paso es muy importante para etapas posteriores, como el seguimiento y análisis, ya que limita las zonas de la imagen que deben ser procesadas.
- Seguimiento de objetos: Implica el seguimiento y el cálculo de la trayectoria de los objetos detectados entre fotogramas consecutivos de la secuencia de vídeo.
- Análisis de comportamiento: Consiste en el reconocimiento de patrones de movimiento considerados como peligrosos dentro de los vídeos analizados.
- Base de datos: La etapa final es el almacenamiento de forma eficiente y ordenada de los distintos tipos de datos obtenidos durante la vigilancia para luego recuperarlos en caso de necesidad.

Otros posibles elementos dentro de un sistema de vigilancia son:

- Clasificación de objetos: Diferentes regiones en movimiento pueden corresponder con diferentes tipos de objetos. Para un posterior seguimiento o evaluación de comportamiento de los elementos de interés, puede ser importante clasificar correctamente los distintos tipos de objetos en movimiento.
- Identificación de personas: Es posible modelar características físicas y biométricas de ciertas personas, como la forma de la cara o la forma de andar, para identificar a personas que tienen o no permitida la presencia en una determinada zona.
- Fusión de datos: La fusión de datos es necesaria cuando disponemos de un sistema formado por varios dispositivos de entrada. En estos casos tenemos varias cámaras que permiten cubrir un área mayor y eliminar ambigüedades, y

cada una proporciona una información diferente. Por tanto es necesario generar una única medida a partir de múltiples observaciones.

A continuación vamos a ver algunas de las posibles implementaciones o técnicas existentes en la literatura para llevar a cabo las etapas de detección, seguimiento y análisis de conducta. Para realizar este repaso a los algoritmos más comunes empleados en visión artificial, vamos a seguir la clasificación realizada por Yilmaz, Javed y Shah en [9] y Ko en [12].

En la siguiente figura se muestra un esquema de las distintas fases que conforman un sistema genérico de vigilancia inteligente y automático.

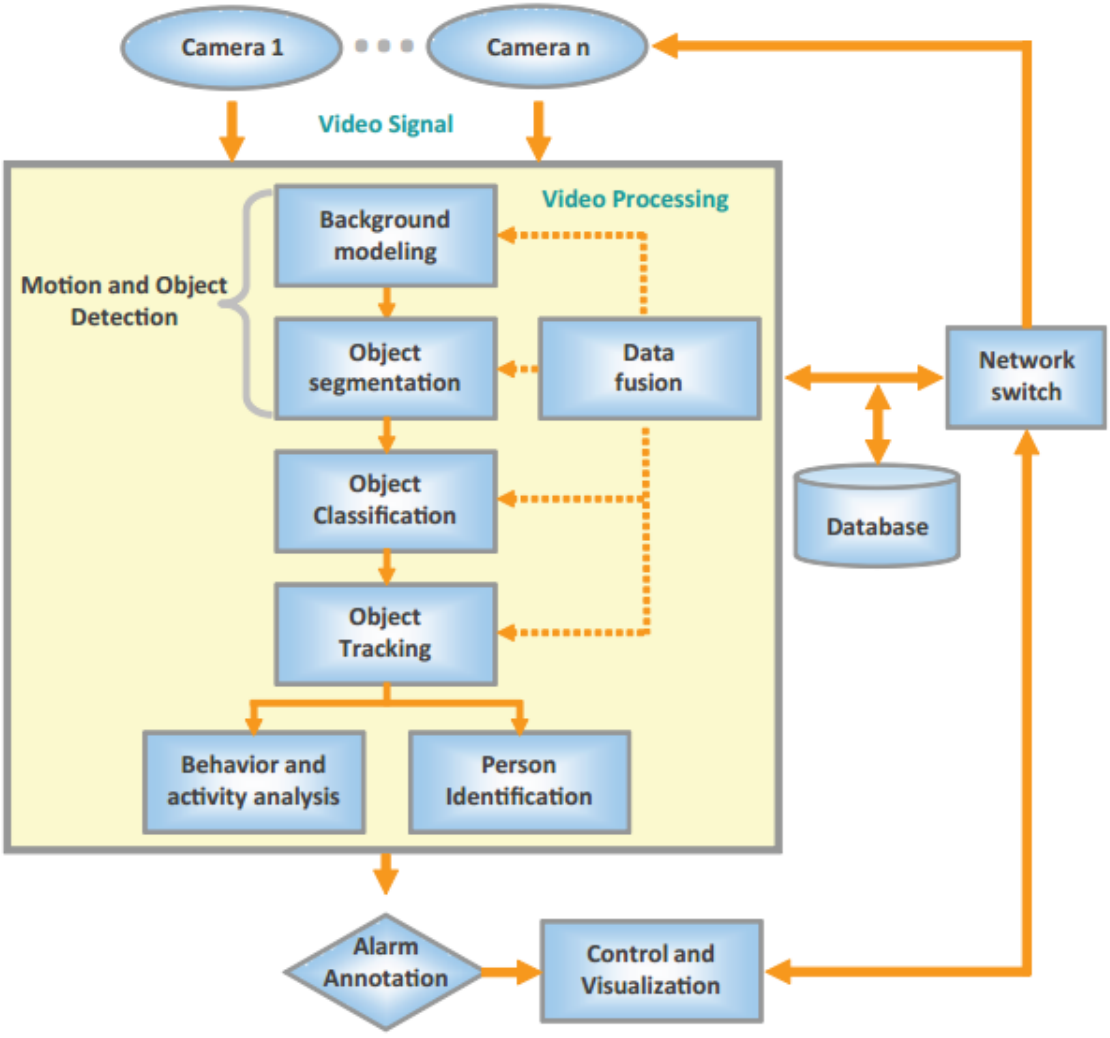


Figura 2.1. Esquema de las etapas de un sistema de vigilancia.
Imagen obtenida de [12]

2.2.1. DETECCIÓN DE OBJETOS

El primer paso de la mayoría de sistemas de videovigilancia comienza con la detección de objetos o personas en movimiento. La detección de objetos principalmente consiste en separar regiones que corresponden con objetos en movimiento del resto de la imagen. El resto de procesos, como el seguimiento, dependen profundamente de esta etapa. A continuación veremos diferentes técnicas de detección:

- **Modelado y substracción del fondo**

El concepto básico consiste en mantener un modelo estático del fondo del entorno en el que se está grabando, y comparar la secuencia actual del vídeo con el fondo. Los objetos en movimiento se detectan encontrando los cambios en la imagen diferencia, obtenida al restar la imagen actual y el fondo. Los píxeles que han sufrido algún tipo de cambio se marcan para un procesado posterior. Normalmente, se aplica un algoritmo de conexión o relleno para obtener regiones conectadas que se corresponden con los objetos.

Para llevar a cabo el modelado y substracción del fondo existen varios métodos:

- Modelo Gaussiano (Gaussian model)

El primer método que dio popularidad al modelado del fondo consistía en modelar el color de cada píxel del fondo como una función Gaussiana 3D (en espacio de color YUV). La media y la covarianza se calculan en los primeros fotogramas consecutivos. Una vez que se dispone del fondo, se calcula para cada píxel entrante la probabilidad de su color, y los píxeles que se desvían del fondo se catalogan como objetos en movimiento.

Sin embargo, a veces un modelo basado en una sola función Gaussiana no es suficiente, especialmente en entornos de exterior, ya que en estos casos pueden percibirse varios colores debido a sombras o reflejos. Una mejora se consigue usando un modelo multimodal, como una mezcla de varias funciones Gaussianas, para caracterizar el color de cada píxel.

- Modelo no paramétrico

En este caso, además de emplearse información del color de cada píxel, se incorpora información espacial. Así, el píxel actual no sólo se compara con su correspondiente píxel del fondo, sino que también se compara con los píxeles próximos. En consecuencia, este método puede soportar pequeños movimientos de la cámara o en el fondo.

- Eigen-background

Esta técnica parte de un punto de vista diferente, ya que modela el fondo mediante el uso de autovectores, que recogen todos los cambios de iluminación posibles en la imagen. El modelo se crea tomando varias muestras de la imagen y eligiendo los autovectores más importantes. Los objetos en movimiento no pueden ser bien descritos por este modelo, por tanto la detección se realiza proyectando la imagen actual sobre el modelo de fondo y encontrando las diferencias entre ambas imágenes.

En la siguiente imagen, tomada de [9], se puede ver cómo funciona este método. En (a) se tiene la imagen de entrada con objetos, en (b) la imagen creada a partir de proyectar la imagen de entrada sobre el autoespacio, y en (c) la imagen de diferencia.

(a) (b) (c)

Figura 2.2. Substracción del fondo por descomposición en autoespacios

En conclusión, los métodos de modelado y substracción del fondo son muy utilizados porque son capaces de caracterizar cambios de iluminación, ruido y movimientos periódicos del fondo y pueden detectar objetos en gran variedad de situaciones, además de ser eficientes computacionalmente.

Sin embargo, en la práctica a veces las regiones que corresponden a objetos se encuentran incompletas. Puede darse que los objetos estén divididos en varias regiones o que tengan huecos en su interior debido a que no hay garantías de que las características del objeto sean diferentes de las características del fondo.

Por otro lado, la principal limitación del modelado del fondo es que necesita que las cámaras se encuentren inmóviles, ya que el movimiento distorsiona la caracterización del fondo. En estos casos habría que utilizar métodos que fueran capaces de regenerar el fondo cada cierto intervalo de tiempo.

- **Segmentación**

La segmentación de video puede definirse como la división de las secuencias de imágenes en regiones perceptualmente similares. Cada algoritmo de segmentación se enfrenta a dos problemas: establecer un criterio para determinar una buena partición y un método para lograr particiones eficientes.

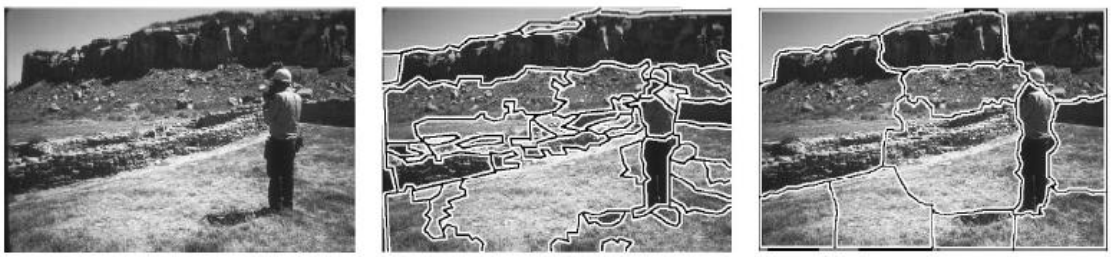
- Mean Shift (Cambio medio)

Desarrollado por Camaniciu [20], el método se basa en detectar grupos (*clusters*) en la unión del espacio formado por el color y las coordenadas espaciales. Dada una imagen, el algoritmo se inicializa eligiendo un gran número de centros de grupo elegidos aleatoriamente entre los datos. Cada centro se desplaza al punto que se corresponde con la media de los datos que se encuentran dentro de un elipsoide centrado en el centro del grupo. El vector que se origina entre el nuevo centro y el original se llama vector de cambio medio (*mean shift vector*) y se calcula de forma iterativa hasta que el centro del grupo no se desplaza.

Su principal limitación es que necesita una elección precisa de diversos parámetros para funcionar de manera más eficiente, además de ser costosa desde el punto de vista computacional.

- Graph cut

Este método se plantea como un problema de partición de conjuntos, donde cada píxel se considera como un vértice de un gráfico. La similitud entre dos píxeles se considera como el peso del borde entre estos dos vértices, y se computa como la semejanza entre el color, brillo o textura entre los píxeles. La segmentación se lleva a cabo formando bordes que se consiguen uniendo puntos con un peso elevado.



(a) *(b)* *(c)*

Figura 2.3. Segmentación de la imagen en (a), mediante la técnica mean-shift (b) y graph-cuts (c). Imagen tomada de [9].

Un inconveniente de este sistema es que para imágenes grandes los requisitos de memoria y procesado pueden ser elevados. Por el contrario, requiere seleccionar menos parámetros de forma manual que el modelo anterior.

- Contornos activos

Esta forma de realizar la segmentación consiste en obtener el contorno del objeto, de forma que éste encierre fielmente el borde del objeto. La formación del contorno se gobierna mediante un funcional de energía que define la adecuación del contorno a la región del objeto. El funcional de energía tiene comúnmente la siguiente forma:

$$E(\Gamma) = \int_0^1 E_{int}(V) + E_{im}(V) + E_{ext}(V)\, ds$$

Donde s es la longitud de la curva del contorno Γ, E_{int} incluye las limitaciones de regularización, E_{im} incluye energía basada en apariencia y E_{ext} especifica limitaciones adicionales. E_{int} normalmente incluye un término de curvatura para encontrar el contorno más pequeño. E_{im} se puede calcular de forma local o global. Localmente, se calcula mediante el gradiente evaluado en el contorno y las características globales (color y textura) se calculan dentro y fuera de los objetos.

Un punto importante en estos métodos es la inicialización del contorno. Una de las maneras más comunes consiste en situar el contorno fuera de la región del objeto y encogerlo sucesivamente hasta encontrar el borde del objeto. Esta aproximación, sin embargo, requiere conocer información previa sobre el objeto o el fondo.

- Flujo óptico

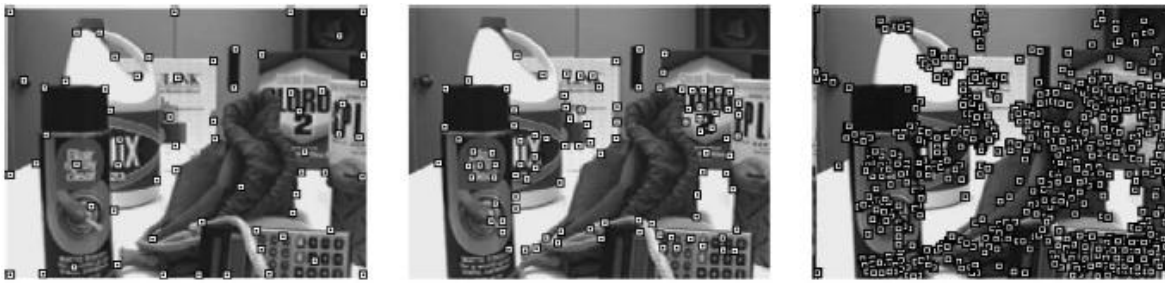
Los métodos basados en flujo óptico solucionan los problemas de la segmentación utilizando las características de los vectores de flujo de objetos en movimiento en el tiempo para detectar las regiones que cambian en una secuencia de imágenes.

El flujo óptico consiste en un campo formado por vectores de desplazamiento que define la traslación de cada píxel en una región. Se calcula mediante la limitación de brillo, que supone que el brillo de los píxeles se mantiene constante en secuencias sucesivas. Las técnicas más conocidas para implementar el flujo óptico son las propuestas por Horn y Schunck o Lucas y Kanade.

- **Detectores de puntos**

Los detectores de puntos se utilizan para encontrar puntos de interés en imágenes que tienen una textura destacada. Una cualidad deseada de los puntos de interés es que sean invariantes a cambios de iluminación y del punto de vista de la cámara.

En la literatura, los métodos de detección de puntos de interés más destacados son el operador de interés de Moravec, el detector de puntos de interés de Harris, el detector de KLT y el detector SIFT (Scale Invariant Feature Transform)



(a) *(b)* *(c)*

Figura 2.4. Detección de puntos de interés, aplicando los operadores Harris (a), KLT (b), y SIFT(c). Imagen tomada de [9]

- **Aprendizaje supervisado**

La detección de objetos puede realizarse aprendiendo automáticamente diferentes vistas de los objetos a partir de una serie de ejemplos. El aprendizaje de diferentes puntos de vista de los objetos exime de la obligación de almacenar un conjunto completo de modelos. Dado un conjunto de ejemplos, los métodos de aprendizaje supervisado generan una función que mapea las entradas a las salidas deseadas.

Categoría	Ejemplos
Modelado de Fondo	Modelo Gaussiano (Gaussian mixture) Modelo no paramétrico Eigen-background
Segmentación	Mean-Shift Contornos activos Graph-cut Flujo óptico
Detectores de Puntos	Operador de Moravec Operador de Harris Detector KLT Detector SIFT (Scale Invariant Feature Transform)
Aprendizaje Supervisado	Clasificador SVM (Support Vector Machines) Adaptative Boosting

Tabla 2.2. Técnicas de detección de objetos

2.2.2. SEGUIMIENTO

La etapa de seguimiento de objetos pretende generar la trayectoria de un objeto durante el tiempo conociendo su posición en cada secuencia del video, además de proporcionar la región del espacio que ocupa en cada momento. Estas dos operaciones pueden hacerse de manera conjunta o separada.

En cualquier caso, los objetos se representan utilizando diversos modelos de apariencia y formas:

- Puntos. El objeto puede representarse mediante un sólo punto, llamado centroide o baricentro, o mediante múltiples puntos. Esta representación es más adecuada para el seguimiento de objetos de pequeño tamaño.
- Formas geométricas. La forma del objeto se representa como un rectángulo o una elipse. El movimiento del objeto en este tipo de representaciones se modela por traslación, afinidad o trasformación proyectiva (homografía). Aunque las formas geométricas son más adecuadas para representar objetos rígidos, también pueden utilizarse para el seguimiento de objetos no rígidos.
- Contorno y silueta del objeto. La representación del contorno define el borde del objeto, mientras que la región dentro del contorno se denomina silueta. Estas representaciones son útiles para el seguimiento de formas complejas y no rígidas.
- Modelos articulados. Estos modelos están formados por diferentes partes unidas entre si, cada una formada por elipses. La relación entre cada una de las partes está controlada por modelos cinemáticos.
- Modelo de esqueleto. El esqueleto del objeto puede obtenerse aplicando la transformada del eje medio (*medial axis*) a la silueta del objeto.

Dependiendo del tipo de representación elegido para el objeto, se pueden usar unas u otras formas de desplazamiento. Por ejemplo, si se ha elegido la representación mediante puntos sólo puede usarse un modelo de traslación, y se ha optado por una representación mediante formas geométricas, son más apropiadas la afinidad y la transformación proyectiva.

- **Seguimiento de puntos**

Los objetos detectados en fotogramas consecutivos se representan mediante puntos, y la asociación de dichos puntos se basa en el estado previo del objeto. Este método requiere de mecanismos externos para detectar objetos en cada fotograma. El principal problema que presenta este tipo de seguimiento es determinar los puntos cuando hay fallos de detección, entradas y salidas de objetos, oclusiones.

El seguimiento de puntos es adecuado cuando tenemos en la imagen objetos de pequeño tamaño, los cuales pueden ser representados por un único punto. Cuando tenemos objetos de mayor tamaño, es necesario el empleo de varios puntos, lo cual se convierte en un problema ante la presencia de varios objetos, ya que es necesario distinguir los objetos entre si.

Las técnicas basadas en correspondencia de puntos pueden dividirse en métodos deterministas y métodos estadísticos.

- Métodos deterministas

Estas técnicas definen el coste de asociar cada objeto del fotograma anterior a un objeto del fotograma actual, mediante una serie de restricciones (proximidad, velocidad máxima, movimientos suaves…). La solución más eficiente se encuentra empleando un algoritmo para minimizar dicho coste.

Algunos de los algoritmos propuestos en la literatura son el rastreador MGE (desarrollado por Salari) y el rastreador GOA (Veenman).

- Métodos estadísticos

Los métodos estadísticos tienen en cuenta el ruido y las perturbaciones aleatorias que pueden alterar la correcta detección de los objetos de la escena.

Dentro de estos métodos, hay que distinguir entre los que son capaces de funcionar cuando tenemos un único elemento móvil en la escena y los que son útiles con múltiples objetos.

El filtro de Kalman se emplea cuando queremos determinar el estado de un solo objeto y el ruido presente en la imagen tiene una distribución Gaussiana. Para ello hace uso de dos etapas: predicción y corrección. El filtro de Kalman es de gran importancia en el campo del procesado de señal, pero también para el control y el guiado de vehículos o en visión artificial.

Sin embargo, cuando en la escena tenemos más de un objeto hay que asociar a cada uno sus medidas correspondientes, entonces el filtro Kamlan ya no es eficiente. También puede darse el caso de que los objetos estén lo suficientemente cerca como para que no sea posible hacer una asociación y una medida correcta.

Existen varias técnicas para asociación de datos que son capaces de solucionar estos problemas, como por ejemplo Joint Probability Data Association Filter (JPDAF) o Multiple Hypothesis Tracking (MHT).

- **Kernel tracking (Seguimiento de núcleo)**

Esta técnica caracteriza los objetos como plantillas rectangulares o elípticas, y el seguimiento se realiza calculando el movimiento de los objetos entre un fotograma y el siguiente. Los tipos de movimiento detectados son la traslación, rotación y afinidad.

Para realizar este tipo de seguimiento se pueden emplear varios tipos de mecanismos, que se diferencian dependiendo de cómo se representan los objetos, el número de objetos que se siguen, cómo se determina el movimiento de los objetos… En este apartado sólo vamos a nombrar algunos de los más significativos, como son el método de mean-shift, que ya vimos en el apartado 2.2.1 de detección de objetos pero esta vez aplicado al seguimiento, y el método KLT (Kande-Lucas-Tomasi). El algoritmo KLT se basa en el concepto de flujo óptico, ya que mediante la generación del campo de vectores de flujo se calcula la traslación y el desplazamiento del objeto contenido en una forma geométrica.

El principal objetivo de los métodos de esta categoría es estimar el movimiento de los objetos. Con la representación de objetos basada en regiones, el cálculo de su movimiento define implícitamente el propio movimiento, la orientación y la forma del objeto. Una de las limitaciones de representar a los objetos mediante formas geométricas simples es que partes de los objetos pueden quedar fuera de la forma definida.

Figura 2.5. Seguimiento con plantilla elíptica
Imágenes tomadas de [27].

- **Seguimiento de la silueta**

Los métodos basados en el seguimiento de la silueta se emplean cuando los objetos tienen formas complejas y no pueden describirse adecuadamente mediante formas geométricas sencillas. El seguimiento se lleva a cabo estimando para cada fotograma la región de la imagen que corresponde con el objeto, empleando para ello un modelo

del objeto que se crea a partir de los fotogramas anteriores. Este tipo de métodos se utilizan cuando es necesario el seguimiento de la región completa del objeto, pero la ventaja más importante del seguimiento de siluetas es su flexibilidad para manejar gran variedad de formas de objetos. Se pueden clasificar en dos categorías:

- Seguimiento del contorno.

La búsqueda del contorno se realiza expandiendo iterativamente el contorno original a su nueva posición en el fotograma actual. Este método requiere que parte del objeto en el fotograma actual se solape con la región del objeto en el fotograma anterior.

El seguimiento por contorno puede realizarse mediante dos diferentes enfoques. El primero emplea distintos estados para definir los objetos en función de su forma y sus parámetros de movimiento, que se actualizan en cada instante de tiempo. El segundo determina el contorno minimizando la energía del contorno usando técnicas de minimización como el método del gradiente.

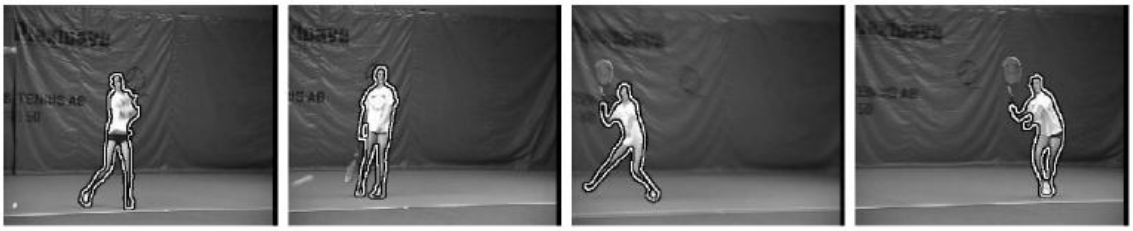


Figura 2.6. Seguimiento de contorno

- Coincidencia de la forma.

Se parte de la silueta del objeto o de su modelo asociado y se buscan coincidencias en el fotograma actual. La búsqueda se realiza calculando la similitud entre el objeto y el modelo generado a partir de la silueta del objeto en el fotograma anterior. En este enfoque, se considera que la silueta sólo se desplaza de un fotograma al siguiente, por lo que no se tienen en cuenta objetos no rígidos.

2.1.1. ANÁLISIS DE COMPORTAMIENTO

Uno de los mayores retos dentro del campo de la visión por computadora y la inteligencia artificial es el entendimiento y el aprendizaje de la conducta a partir de observar actividades en un vídeo. Las investigaciones en este campo se centran principalmente en el desarrollo de métodos para el análisis de datos de vídeo con el

objetivo de extraer y procesar información relacionada con el comportamiento de objetos físicos, como por ejemplo personas, en una escena.

Categoría	Ejemplos
Seguimiento de Puntos	
■ Métodos deterministas ■ Métodos estadísticos	Rastreador MGE Rastreador GOA Filtro de Kalman JPDAF (Joint Probability Data Association Filter) MHT (Multiple Hypothesis Tracking)
Kernel Tracking (Seguimiento de núcleo)	
■ Modelos de apariencia basados en forma	Mean – shift Algoritmo KLT (Kande-Lucas-Tomasi)
Seguimiento de Silueta	
■ Seguimiento de contorno	Métodos variacionales Métodos heurísticos
■ Coincidencia de forma	Hausdorff Transformada de Hough

Tabla 2.3. Técnicas de seguimiento de objetos

En sistemas de videovigilancia automatizados, la detección fiable de comportamiento humano sospechoso o peligroso es un asunto de gran importancia. Un sistema de este tipo normalmente necesita de la combinación eficaz de técnicas de procesado de imágenes y de inteligencia artificial. Las técnicas de procesado de imagen se utilizan para proporcionar características de la imagen de bajo nivel. Las técnicas de inteligencia artificial se emplean para proporcionar decisiones expertas. Se han realizado numerosas investigaciones sobre técnicas de procesado de imagen de bajo nivel como por ejemplo detección, reconocimiento y seguimiento de objetos; sin embargo, han sido pocos los estudios que han dado un entendimiento y una clasificación fiable del comportamiento humano a partir de secuencias de video.

La detección de comportamientos implica el modelado y la clasificación de actividades humanas según ciertas reglas, pero esto no es un proceso sencillo, dada la diversidad y complejidad de los movimientos. Aun así, la idea es dividir los movimientos observados en algunos estados discretos y luego clasificarlos adecuadamente.

Muchos enfoques en este campo del procesado de vídeo incorporan métodos para la detección de eventos específicos. El principal inconveniente de estas técnicas es que son sólo especificas para ciertas aplicaciones pero no pueden aplicarse en otras circunstancias. Otro punto de vista es dividir el procesado en dos etapas:

- Un módulo de procesado de imagen de bajo nivel se emplea para extraer señales visuales y eventos primitivos.
- Esta información se utiliza en un módulo de inteligencia artificial de alto nivel para detectar patrones de comportamiento más complejos.

Dividiendo el problema en dos etapas, se pueden usar técnicas más sencillas e independientes del entorno en cada etapa.

El reto consiste en combinar las técnicas disponibles de reconocimiento de comportamiento para acercarse a la gran diversidad de situaciones que existen en el mundo real. El aprendizaje de patrones de comportamiento puede considerarse como la clasificación de datos variables en el tiempo, como por ejemplo, asociar una secuencia desconocida a un grupo de secuencias de referencia que representan comportamientos comunes o aprendidos. El problema fundamental consiste en aprender las secuencias de referencia a partir de muestras de entrenamiento, y en diseñar los métodos de entrenamiento y de asociación para hacer frente con eficacia a pequeñas variaciones de los datos característicos dentro de cada clase de patrón de movimiento. Los principales métodos para el análisis de comportamiento son los siguientes:

- Hidden Markov Models (HMMs): Un HMM es una herramienta estadística empleada para modelar secuencias caracterizadas por un conjunto de secuencias.
- Dynamic Time Warping (DTM): La DTM (deformación dinámica del tiempo) es una técnica que alinea de manera óptima secuencias de tiempo de longitud variable. Sirve para calcular la semejanza o para encontrar correspondencias entre dos series de tiempo relacionando un patrón de prueba con un patrón de referencia.
- Finite-State Machine (FSM): FSM o autómata de estado finito, es un modelo de comportamiento compuesto por un número finito de estados, transiciones entre estos estados y acciones. Una máquina de estados finitos es un modelo abstracto de una máquina con una memoria primitiva interna.
- Nondeterministic-Finite-State Automaton (NFA): Un NFA o un autómata de estado finito no determinista es una máquina de estado finito en al cual cada par de estados y símbolos de entrada pueden existir varios estados siguientes posibles. Esto lo diferencia de los autómatas finitos deterministas (DFA), en los cuales el siguiente estado posible está unívocamente determinado.
- Time-Delay Neural Network (TDNN): TDNN es una técnica para analizar datos variables en el tiempo. En TDNN, las unidades de retardo se añaden a una red general y estática, y algunos de los valores anteriores de una secuencia variable

en el tiempo se utilizan para predecir el próximo valor. A medida que el conjunto de datos disponible aumenta, se está haciendo más énfasis en las redes neuronales para la representación de información temporal. Los métodos basados en TDNN han sido aplicados con éxito en situaciones como reconocimiento de gestos y lectura de labios.

CAPÍTULO 3

ESTADO DEL ARTE: MEDIDAS DE CALIDAD DE IMAGEN

En este proyecto, el método que se va a utilizar para detectar movimiento en secuencias de vídeo es el algoritmo SSIM. Sin embargo, SSIM originalmente fue desarrollado por Z. Wang [2] como un método para determinar la calidad de una imagen. En el presente capítulo realizaremos un repaso sobre los mecanismos de evaluación de la calidad de imagen, entre los que se encuentra la similitud estructural (SSIM).

En los últimos años, ha habido un interés creciente en el desarrollo de métodos de evaluación objetiva de la calidad de imagen, *image quality assessment* (IQA), que permiten predecir de forma automática los comportamientos humanos en la evaluación de la calidad de imagen. Estas medidas de IQA perceptual tienen amplias aplicaciones en la evaluación, control, diseño y optimización de sistemas de adquisición de imágenes, sistemas de comunicación, de procesado y de visualización.

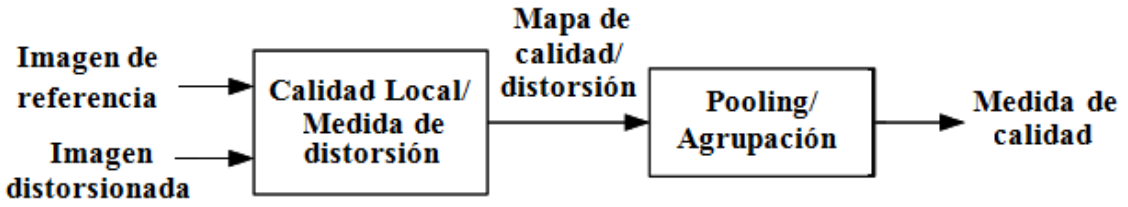


Figura 3.1. Esquema básico de un sistema de evaluación de la calidad de imágenes [19]

Además de SSIM, existen otros mecanismos para determinar la calidad de la imagen, entre ellos uno de los más conocidos es el error cuadrático medio (MSE), pero existen otros muchos que se fundamentan en distintos principios. Muchas de las implementaciones para realizar la IQA adoptan una estructura común de dos etapas: la primera consiste en la medida local de la calidad/distorsión de la imagen, y la segunda

es la agrupación o puesta en común de los datos. Tales aproximaciones se estiman que son consistentes con el funcionamiento del sistema visual humano (HVS).

3.1. INTRODUCCIÓN

Las imágenes digitales están sujetas a gran variedad de distorsiones durante las etapas de procesado, compresión, almacenamiento, transmisión o reproducción; y cualquiera de ellas puede resultar en la degradación de la calidad visual. Para aplicaciones en las que el objetivo último sea que el ser humano visualice las imágenes, el único método adecuado para cuantificar la calidad de la imagen es a través de una evaluación subjetiva.

En la práctica, sin embargo, las evaluaciones subjetivas normalmente son inconvenientes y costosas. El objetivo en la investigación en la evaluación objetiva de la calidad de imagen es desarrollar medidas cuantitativas que puedan predecir automáticamente la calidad de la imagen percibida.

Un método de medición objetiva de la calidad de imagen puede jugar gran variedad de roles en aplicaciones de procesado de imágenes. En primer lugar, se puede usar para monitorizar dinámicamente y ajustar la calidad de imagen. En segundo lugar, puede emplearse para optimizar algoritmos y la configuración de parámetros en sistemas de procesado de imágenes. Tercero, se puede utilizar para referenciar algoritmos y sistemas de procesado de imágenes.

Las métricas objetivas pueden clasificarse dependiendo de si se dispone de una imagen original (sin distorsión), con la cual la imagen que está siendo medida se compara. La mayoría de los enfoques existentes se conocen como *full-reference*, lo que significa que una imagen de referencia completa es conocida con la cual podemos comparar imágenes distorsionadas. En muchas aplicaciones prácticas, sin embargo, la imagen de referencia no está disponible y es necesario un enfoque “ciego” o sin referencia (*no-reference*). En un tercer método, la imagen de referencia sólo está disponible parcialmente, en forma de un conjunto de características extraídas para ayudar a evaluar la calidad de la imagen. Se conoce como evaluación de calidad de referencia reducida (*reduced-reference*). Como hemos dicho, los métodos *full-reference* son los más abundantes y los que vamos a comentar a continuación [2].

El método *full-reference* más simple y más utilizado es el error cuadrático medio (MSE), que se calcula promediando el cuadrado de la diferencia de intensidad entre pixeles distorsionados y de referencia, junto con la cantidad correspondiente de relación señal-ruido, *peak signal-to-noise ratio* (PSNR).

Estos métodos son atractivos porque son fáciles de calcular, tienen un significado físico claro y son convenientes en el contexto de optimización. Pero no están muy bien adaptados a la calidad visual percibida. En las tres últimas décadas, una gran cantidad de esfuerzo ha ido dirigido al desarrollo de métodos de evaluación de calidad que tomen ventaja de las características conocidas del sistema de visión humano (HVS).

3.2. EVALUACIÓN DE LA CALIDAD DE IMAGEN BASADA EN LA SENSIBILIDAD DEL ERROR

En el contexto de la evaluación de la calidad de una imagen, dicha imagen puede verse como la suma de una señal de referencia no distorsionada y una señal de error. Un supuesto ampliamente aceptado es que la pérdida de la calidad de percepción está directamente relacionada con la visibilidad del error.

La implementación más simple de este método es el MSE (*Mean Squared Error*), que cuantifica de manera objetiva la potencia de la señal de error. Pero dos imágenes con el mismo MSE pueden tener diferentes tipos de error, algunos de los cuales son mucho más visibles que otros. La mayoría de enfoques propuestos en la literatura intentan ponderar los diferentes aspectos de la señal de error de acuerdo con su visibilidad, según lo determinado por medidas psicofísicas en los seres humanos o las mediciones fisiológicas en los animales.

3.2.1. ESTRUCTURA

La siguiente figura ilustra un esquema genérico de evaluación de calidad de imagen basado en la sensibilidad de error.

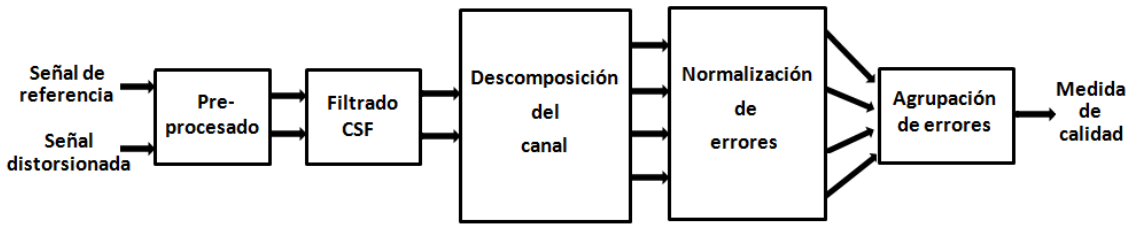


Figura 3.2. Esquema de un sistema de evaluación de calidad basado en la sensibilidad de error

Aunque varían en los detalles, la mayoría de modelos pueden describirse con un diagrama similar [2]. Las etapas principales del procedimiento son las siguientes:

- Pre-procesado: En esta etapa se suele realizar una gran variedad de operaciones para eliminar distorsiones de las imágenes que se comparan. En primer lugar, las señales distorsionadas y de referencia son correctamente alineadas y escaladas. En segundo lugar, la señal debe ser convertida a un espacio de color que sea más apropiado para el HVS. En tercer lugar, las métricas de evaluación de calidad deben convertir los valores de los píxeles almacenados en la memoria en valores de luminancia en el dispositivo de visualización a través de transformaciones punto a punto no lineales. Finalmente, las señales distorsionadas y de referencia se modifican mediante operaciones no lineales para simular adaptación a la luz.
- Filtrado CSF: La función de la sensibilidad al contraste (CSF) describe la sensibilidad del HVS a diferentes frecuencias temporales y espaciales que están presentes en los estímulos visuales. Algunas métricas de calidad de imagen incluyen una etapa que pondera la señal de acuerdo con esta función (normalmente se implementa usando un filtro lineal que se aproxima a la respuesta en frecuencia del CSF). Por otra parte, otras métricas más recientes eligen implementar CSF como un factor de normalización, después de la descomposición del canal.
- Descomposición del canal: Las imágenes son típicamente separadas en sub-bandas (comúnmente llamados canales), que son selectivos en frecuencia y orientación. Existen varias opciones cuando se trata de implementar la descomposición: Mientras algunos métodos de evaluación de calidad realizan complejas descomposiciones de canal que intentan aproximarse a las respuestas neuronales en la corteza visual primaria, muchas métricas usan transformadas más sencillas como la transformada discreta del coseno (DCT) o transformadas wavelet separables.
- Normalización de errores: El error (la diferencia) entre la señal de referencia descompuesta y las señales distorsionadas en cada canal se calcula y normaliza de acuerdo con cierto modelo de enmascaramiento, que tiene en cuenta el hecho de que la presencia de una componente de una imagen disminuirá la visibilidad de otra componente próxima, temporal o espacialmente. El mecanismo de normalización pondera la señal de error en un canal mediante un umbral de visibilidad variable en el espacio. El umbral de visibilidad en cada punto es calculado en base a la energía de la referencia y/o a los coeficientes de distorsión en un entorno.
- Agrupación del error (*Error pooling*): La fase final de todas las métricas de calidad debe combinar las señales de error normalizadas a través de los diferentes canales en un único valor. Para la mayoría de los métodos de

evaluación de la calidad, la agrupación toma la forma de una normalización de Minkowski

$$E(\{e_{l,k}\} = (\sum_{l}\sum_{k} |e_{l,k}|^{\beta})^{1/\beta}$$

donde $e_{l,k}$ es el error normalizado del n-ésimo coeficiente en el canal n-ésimo, y β es una constante elegida típicamente entre 1 y 4. La agrupación de Minkowski puede realizarse sobre el espacio (índice k) y luego sobre frecuencia (índice l) o viceversa. Para proporcionar ponderación espacial, puede usarse un mapa espacial que indique la importancia relativa de las diferentes regiones.

3.2.2. LIMITACIONES

El principio subyacente en el que se basa el enfoque de sensibilidad del error es que la calidad percibida se estima mejor cuantificado la visibilidad de los errores. Esto se consigue, esencialmente, simulando las propiedades funcionales de las etapas tempranas del HSV, caracterizadas mediante experimentos psicológicos y psicofísicos. Aunque este enfoque ha encontrado aceptación casi universal, es importante reconocer sus limitaciones. En particular, el HSV es un sistema complejo y altamente no lineal, pero la mayoría de modelos están basados en operadores lineales o cuasi-lineales que han sido caracterizados usando estímulos simplistas y restringidos. Así, los enfoques basados en la sensibilidad del error se basan en una serie de grandes supuestos y generalizaciones [2]:

- El problema de la definición de calidad: El mayor problema es que no está claro que el error de visibilidad deba ser equiparado con pérdidas de calidad, ya que algunas distorsiones pueden ser percibidas claramente pero no son demasiado molestas.
- El problema del umbral: Los experimentos que subyacen bajo de muchos de los modelos de sensibilidad del error están diseñados para estimar el umbral bajo el cual el estímulo es apenas visible. Sin embargo, muy pocos estudios indican si estos modelos cercanos al umbral pueden ser extendidos para caracterizar distorsiones mucho mayores que el nivel de umbral.
- El problema de la complejidad de la imagen natural: La mayor parte de los experimentos se realizan empleando modelos o patrones sencillos, como puntos, barras o enrejados sinusoidales. Sin embargo, todos estos modelos son mucho más simples que las imágenes del mundo real, que podrían pensarse

como la superposición de un número mucho mayor de patrones simples. Por tanto, no es posible asegurar que los modelos sean capaces de evaluar la calidad de imágenes naturales de estructura compleja.

- El problema de la no-correlación: El uso de la métrica de Minkowski para la agrupación espacial de errores supone asumir que los errores son estadísticamente independientes. Esto sería cierto si el procesado previo a la agrupación eliminara las dependencias en las señales de entrada, sin embargo, esto no ocurre para métodos de descomposición lineal del canal, como por ejemplo la transformada wavelet.
- El problema de la interacción cognitiva: La comprensión cognitiva y acciones del procesamiento visual influyen en la calidad percibida de la imagen. Es por esto que un observador humano calificará de diferente manera una imagen dependiendo de la información de que disponga, las instrucciones que haya recibido o la atención que esté prestando. La mayor parte de las medidas de la calidad de imagen no tienen en consideración estos efectos ya que son difíciles de parametrizar.

3.2.3. MSE

A continuación procedemos a comentar en qué consiste el método del error cuadrático medio, y también veremos qué propiedades son ventajosas y cuales son inconvenientes para considerar el MSE como un buen mecanismo de evaluación de calidad [21].

Se supone que X e Y son dos señales discretas de longitud finita (por ejemplo, dos imágenes), donde N es el número de muestras (píxeles, en el caso de imágenes). El error cuadrático medio (MSE) entre las imágenes es:

$$MSE(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2$$

En el MSE, nos referimos a la señal error como $e_i = x_i - y_i$, que es la diferencia entre la señal original y la señal distorsionada. Si partimos de que una de las señales es original o tiene calidad aceptable, entonces es MSE puede ser visto como la medida de la calidad de la señal. Una forma más general es la norma l_p:

$$d_p(x,y) = (\sum_{i=1}^{N}|e_i|^p)^{1/p}$$

En la literatura relacionada con el procesado de imágenes, normalmente el MSE se convierte en la relación de pico de señal a ruido (peak signal-to-noise ratio - PSNR):

$$PSNR = 10 \log_{10} \frac{L^2}{MSE}$$

Donde L es el rango dinámico de la imagen. Por ejemplo, imágenes que utilizan 8 bits por cada píxel, $L = 2^8 = 255$. La PSNR es útil si se comparan imágenes con rango dinámico, pero de otra forma no proporciona información distinta a la de MSE.

- **Ventajas de MSE**

1) Es simple. Esta libre de parámetros y no es costoso de calcular, ya que para cada muestra sólo se realiza una multiplicación y dos sumas. No tiene memoria, el cálculo del error de una muestra es independiente del resto.
2) Todas las normas l_p son medidas de la distancia válidas en $\boldsymbol{R}^N$, lo que satisface las siguientes condiciones:
   - No negativo: $d_p(x, y) \geq 0$
   - Identidad: $d_p(x, y) = 0$ si y sólo si X=Y
   - Simetría: $d_p(x, y) = d_p(y, x)$
   - Desigualdad triangular: $d_p(x, z) \leq d_p(x, y) + d_p(y, z)$

   El caso $p = 2$, (proporcional a la raíz cuadrada del MSE) es la medida de la distancia en un espacio euclídeo N-dimensional.
3) Tiene un claro significado físico. Tal como está expresado, MSE es la forma natural de definir la energía de la señal de error. Por el teorema de Parseval, la medida de la energía se conserva después de cualquier transformación ortogonal y lineal. Esta propiedad garantiza que la energía de una distorsión de la señal en el dominio transformado es igual que en dominio de la señal.
4) El MSE es una medida excelente en el contexto de la optimización, ya que posee las propiedades de convexidad, simetría y diferenciabilidad. Los problemas de optimización del Mínimo-MSE normalmente tienen solución analítica cerrada.
5) El MSE es apropiado para estimación y estadísticas. El MSE es aditivo para fuentes de distorsión independientes.
6) El MSE es ampliamente utilizado simplemente por el hábito y la costumbre. Históricamente, se ha empleado para optimizar una gran variedad de aplicaciones de procesado de señal como diseño de filtros, compresión de señal, restauración, reconstrucción y clasificación.

- **Inconvenientes de MSE**

MSE posee muchas propiedades que hacen su uso favorable, sin embargo no ofrece buenas prestaciones cuando se emplea para medir la calidad de una imagen ya que no se adapta bien a la percepción humana.

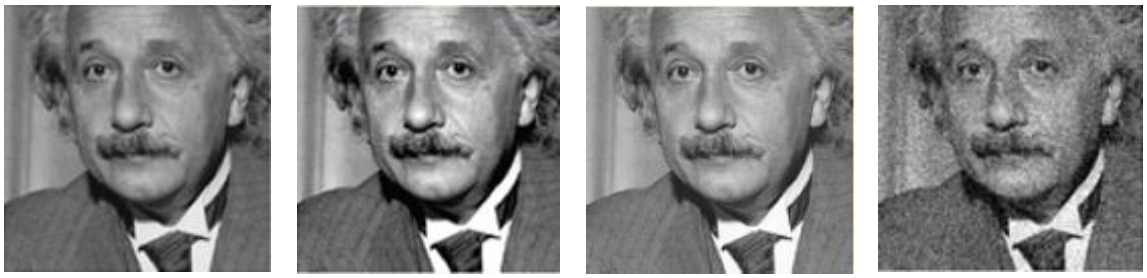
Esto se debe en parte a unos supuestos implícitos que se deben tener en cuenta cuando se trabaja con MSE. Estos supuestos son muy restrictivos, ya que imponen limitaciones a las muestras de señal, cómo interactúan entre ellas y con el error [21].

1) La calidad de la señal es independiente de las relaciones temporales o espaciales entre las muestras de la señal original. En otras palabras, si las señales original y distorsionada se reordenan aleatoriamente, el MSE entre ambas se conserva.
2) La fidelidad de la señal es independiente de cualquier relación entre la señal original y la señal de error. Dada una señal de error, el MSE es el mismo, independientemente de la señal a la que añada.
3) La calidad de la señal es independiente del signo de las muestras de la señal de error.
4) Todas las muestras de la señal tienen la misma importancia en la medida de la calidad.

En las siguientes imágenes de la figura 3.3, obtenidas de [21], vamos a ver que MSE no es un método muy adecuado para establecer un criterio objetivo de calidad en imágenes si lo comparamos con otros mecanismos, como por ejemplo SSIM, el cual se explicará en el próximo apartado.

Ante diferentes tipos de alteraciones de la señal, el valor de calidad proporcionado por MSE es muy parecido. Sin embargo, es fácilmente apreciable a simple vista que la calidad de las imágenes varía ostensiblemente. Por otra parte, el índice SSIM aporta unos valores más cercanos a la realidad, ya que tiene un funcionamiento más cercano al comportamiento de la percepción humana.

La figura (a) se corresponde con la imagen de referencia, y se le asignan los valores de índice máximos, MSE=0 y SSIM=1. La imagen (b) ha sufrido una variación de contraste. Para el índice SSIM la calidad de esta señal es bastante elevada (SSIM=0.928) y para MSE =306. En el caso de las siguientes imágenes (c) y (d) se ha producido un cambio de iluminación y se ha añadido ruido blanco Gaussiano, respectivamente. Mientras que MSE otorga a ambas imágenes el mismo valor (MSE=309), y similar al caso anterior, la calidad de las señales es claramente diferente y consecuentemente SSIM las evalúa de distinta forma: SSIM= 0.987 y SSIM=0.576, respectivamente.



(a) MSE=0 SSIM=1 | (b) MSE=306 SSIM=0.928 | (c) MSE=309 SSIM=0.987 | (d) MSE=309 SSIM=0.576

Figura 3.3. Comparación de índices MSE y SSIM frente a distintos tipos de distorsión

3.3. EVALUACIÓN DE LA CALIDAD DE IMAGEN BASADA EN LA SIMILITUD ESTRUCTURAL

Las señales de imágenes naturales están altamente estructuradas: sus píxeles presentan fuertes dependencias, sobre todo cuando están espacialmente próximos. Estas dependencias tienen información importante sobre la estructura de los objetos de la escena. La métrica de error de Minkowski, que mencionábamos en el apartado anterior, está basada en diferencias punto a punto de la señal, que son independientes de la estructura subyacente de la señal.

Aunque la mayoría de las medidas de calidad basadas en la sensibilidad del error descomponen la señal mediante transformaciones lineales, esto no elimina las dependencias más fuertes. La motivación del nuevo enfoque es encontrar un modo más directo de comparar las estructuras de las señales de referencia y distorsionadas.

Para presentar y comprender las características de este nuevo enfoque se pueden exponer algunos de los puntos que lo diferencian de la filosofía de sensibilidad del error [2]:

- El método de sensibilidad del error estima el error percibido para cuantificar las degradaciones de las imágenes, mientras que la nueva filosofía considera las degradaciones en imágenes como los cambios percibidos en la variación de la información estructural.
- El paradigma de sensibilidad del error es una aproximación que simula la función de las etapas tempranas en el HSV. El nuevo paradigma es una aproximación arriba-abajo, imitando el supuesto funcionamiento de todo el HSV. Esto, por otra parte, supera el problema del supra-umbral mencionado anteriormente porque no se basa en umbrales psicofísicos para cuantificar la distorsión percibida.

- Los problemas de complejidad de la imagen natural y de no-correlación también se evitan en cierta medida debido a que la nueva filosofía no trata de predecir la calidad de imagen mediante la acumulación de errores. En cambio, el nuevo método propone evaluar los cambios estructurales directamente entre dos señales estructuradas de forma compleja.

3.3.1. SSIM

El índice SSIM (*Structural SIMilarity*), que como hemos dicho sirve para realizar la medida de la calidad de una imagen, puede tomar variedad de formas dependiendo de si se implementa en una escala, sobre varias escalas. La aproximación mediante SSIM fue originalmente motivada por la observación de que las señales de imágenes naturales están altamente estructuradas, sus píxeles presentan fuertes dependencias, sobre todo cuando están espacialmente próximos, y estas dependencias contienen información importante de la estructura de los objetos en la escena [2].

La filosofía principal subyacente en el índice SSIM original es que el sistema visual humano está adaptado a extraer la información estructural de las escenas y por tanto, al menos para la medida de la fidelidad de una imagen, la conservación de la estructura de la señal es un punto importante. De forma equivalente, con el objetivo de conseguir medidas de calidad, un algoritmo puede buscar medir la distorsión estructural que sufre una imagen. Las distorsiones que afectan a una imagen pueden clasificarse como estructurales y no estructurales. A continuación vamos a determinar las diferencias existentes entre ambas [21].

- Las distorsiones no estructurales (un cambio de luminancia o luminosidad, un cambio de contraste, distorsión Gamma, y un desplazamiento espacial) son causadas por las condiciones que suceden durante la toma de las imágenes y su visualización. Estas distorsiones no cambian las estructuras de las imágenes de los objetos en la escena.
- Sin embargo, otras distorsiones (ruido aditivo y compresión con pérdidas y emborronamiento) cambian significativamente la estructura de los objetos. Si consideramos que el sistema visual humano busca identificar y reconocer objetos dentro de una escena, entonces debe ser altamente sensible a las distorsiones estructurales y compensar automáticamente las no estructurales. Por tanto, una medida de la calidad de la señal objetiva debe simular este comportamiento.

- **Aplicaciones**

SSIM se ha utilizado para evaluar los resultados del procesado de imágenes en un creciente número de aplicaciones [21]. Como por ejemplo:

- Fusión de imágenes.
- Compresión de imágenes.
- Calidad de imagen cromática.
- Transmisión (streaming) de video inalámbrico.
- Vigilancia.
- Imágenes de radar.
- Imágenes de infrarrojo.
- Imagen de resonancia magnética (MRI).
- Imágenes de cromosomas.
- Teledetección.
- Reconocimiento de objetos.

- **Funcionamiento**

La idea básica de SSIM consiste en partir de dos imágenes X e Y que se van a comparar. Si consideramos que una de las señales tiene calidad perfecta, entonces la medida de la similitud puede servir como una medida cuantitativa de la calidad de la segunda señal. El índice SSIM local mide las similitudes entre tres elementos de las dos imágenes: la luminancia o valores de brillo $\mathrm{l}(\mathrm{x},\mathrm{y})$, el contraste $c(x,y)$ y la estructura $s(x,y)$. Estas semejanzas locales se expresan usando medidas estadísticas y se combinan para formar el índice SSIM local [2].

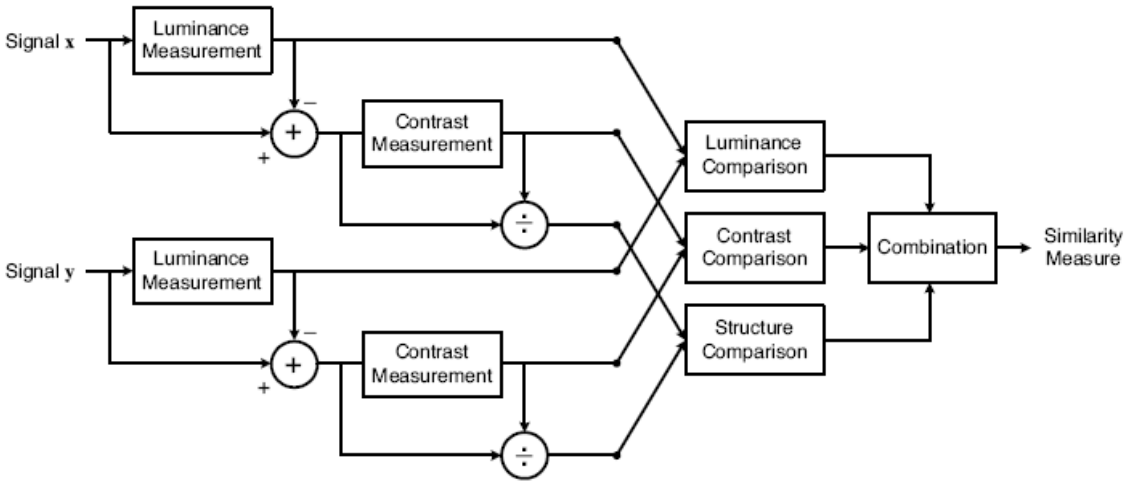


Figura 3.4. Esquema de funcionamiento de SSIM [2]

En primer lugar, se compara la luminancia de cada señal. Asumiendo señales discretas, ésta se estima como la intensidad media:

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i$$

La función de comparación de la luminancia $l(x, y)$, es función de μ_x y μ_y, y se define:

$$l(x, y) = \frac{2\mu_x\mu_y + C1}{{\mu_x}^2 + {\mu_y}^2 + C1}$$

Las constantes C1 se han incluido para evitar inestabilidad cuando ${\mu_x}^2$ y ${\mu_y}^2$ es muy próximo a cero. Con el mismo fin se han incluido las constantes C2 y C3 que se aplican en la comparación de contraste y estructura, descritas más adelante.

En segundo lugar, se elimina la intensidad media de la señal. En forma discreta, la señal resultante $x - \mu_x$ se corresponde con la proyección del vector x sobre el hiperplano definido por:

$$\sum_{i=1}^{N} x_i = 0$$

Se utiliza la desviación estándar (la raíz cuadrada de la varianza) como una estimación del contraste de la señal:

$$\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{\frac{1}{2}}$$

Por tanto, la comparación del contraste $c(x, y)$, es la comparación entre σ_x y σ_y.

$$c(x, y) = \frac{2\sigma_x\sigma_y + C2}{{\sigma_x}^2 + {\sigma_y}^2 + C2}$$

La comparación de la estructura $s(x, y)$ se lleva a cabo después de la resta de la luminancia y de la normalización de la varianza. La señal es normalizada (dividida) por su propia desviación estándar, de modo que las dos señales comparadas tienen desviación estándar unitaria. Específicamente, se asocian dos vectores unitarios $(x - \mu_x)/\sigma_x$ y $(y - \mu_y)/\sigma_y$. La correlación (producto interno) entre ellos es una forma simple y efectiva para cuantificar la semejanza estructural. Entonces, la función de comparación de estructura se define:

$$s(x,y) = \frac{\sigma_{xy} + C3}{\sigma_x \sigma_y + C3}$$

Como en la medida de la luminancia y el contraste, se introduce una constante tanto en el denominador como en el numerador. En la forma discreta σ_{xy} puede estimarse como:

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)$$

Finalmente, las tres componentes se combinan para producir una medida de similitud general y se obtiene el índice SSIM entre las dos señales x e y:

$$SSIM(x,y) = l(x,y) \cdot c(x,y) \cdot s(x,y) =$$
$$= \left(\frac{2\mu_x\mu_y + C1}{\mu_x{}^2 + \mu_y{}^2 + C1}\right) \cdot \left(\frac{2\sigma_x\sigma_y + C2}{\sigma_x{}^2 + \sigma_y{}^2 + C2}\right) \cdot \left(\frac{\sigma_{xy} + C3}{\sigma_x\sigma_y + C3}\right)$$

El índice SSIM se calcula localmente dentro de una ventana cuadrada deslizante que se mueve píxel a píxel a través de la imagen, generando un mapa SSIM. El índice SSIM de toda la imagen se calcula entonces por la agrupación del mapa SSIM, por ejemplo, simplemente promediando los valores SSIM a través de la imagen.

Un punto importante es que las tres componentes son relativamente independientes. Por ejemplo, cambios de luminancia y/o contraste no afectarán a la estructura de la imagen. También deseamos que la medida de similitud cumpla las siguientes condiciones:

1) Simetría: S(x,y) = S(y,x)
2) Acotación : S(x,y)<=1
3) Máximo único: S(x,y)=1, si y solo si X=Y

En la siguiente figura, tomada de [21], se muestra el funcionamiento del método SSIM. La primera imagen (a) se corresponde con la imagen original y que se considera que tiene calidad perfecta. La segunda imagen (b) es la imagen original a la que se ha aplicado compresión JPEG. Para comprobar la calidad de la señal comprimida respecto a la de referencia se genera el mapa SSIM (c), que puede verse en la última imagen. SSIM captura correctamente los efectos de falsos contornos en la zona del cielo y los bloques que producen en los bordes de los objetos.

(a) (b) (c)

Figura 3.5. Evaluación de calidad mediante SSIM

3.4. EVALUACIÓN DE LA CALIDAD DE LA IMAGEN BASADA EN VARIANZA LOCAL

Además de SSIM existen otros métodos que utilizan en su análisis la información estructural de la imagen, como el método *Quality Index based on Local Variance* (QILV) [18].

Este método se basa en la comparación de la varianza local de dos imágenes, con el objetivo de manejar de forma más apropiada la no estacionariedad de las imágenes que se están comparando. Los procesos no estacionarios surgen de forma natural en imágenes en las cuales estructuras están presentes, y cambios en el comportamiento estructural implican un cambio en la no estacionariedad. Este método puede verse como un nuevo procedimiento independiente o como un complemento de otros, como por ejemplo SSIM.

El principio en el que se basa este nuevo mecanismo es que una gran cantidad de la información estructural de una imagen se encuentra codificada en su distribución de varianza local. En el índice SSIM, se calcula la varianza local de las dos imágenes, pero el índice global conjunto únicamente considera la media de ambos valores y, por lo tanto, la no estacionariedad de las imágenes no se tiene en cuenta.

La varianza local de una imagen I se define como:

$$Var(I_{i,j}) = E\{(I_{i,j} - \overline{I_{i,J}})^2\}$$

Siendo $\overline{I_{i,J}} = E\{I_{i,j}\}$ la media local de la imagen. La varianza también puede ser estimada utilizando una estructura ponderada de vecindad $\eta_{i,j}$ centrada en el píxel que está siendo analizado.

El tamaño de la vecindad $\eta_{i,j}$ debe estar relacionado con la escala de las estructuras previstas en una situación determinada. La varianza local estimada se utiliza como medida de la calidad de la semejanza estructural entre dos imágenes. Para determinarla, se emplearán algunos de sus estadísticos. En primer lugar, la media de la varianza local μV_I se estima como:

$$\hat{\mu}V_I = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} Var(I_{i,j})$$

La desviación estándar de la varianza local se define:

$$\sigma_{V_I} = \left(E\{(Var(I_{i,j}) - \mu V_I)^2\}\right)^{1/2}$$

Y puede estimarse como:

$$\hat{\sigma}_{V_I} = (\frac{1}{MN-1}\sum_{i=1}^{M}\sum_{j=1}^{N}(Var(I_{i,j}) - \mu V_I)^2)^{1/2}$$

Finalmente, la covarianza entre las varianzas de dos imágenes I y J se define como:

$$\sigma_{V_I V_J} = E\{(Var(I_{i,j}) - \mu V_I)(Var(J_{i,j}) - \mu V_J)\}$$

Y se estima como:

$$\hat{\sigma}_{V_I V_J} = \frac{1}{MN-1}\sum_{i=1}^{M}\sum_{j=1}^{N}(Var(I_{i,j}) - \mu V_I)(Var(J_{i,j}) - \mu V_J)$$

Finalmente, se define el índice de calidad basado en varianza local (QILV) entre dos imágenes I y J de la siguiente manera:

$$QILV(I,J) = \frac{2\mu_{V_I}\mu_{V_J}}{{\mu_{V_I}}^2 + {\mu_{V_J}}^2} \cdot \frac{2\sigma_{V_I}\sigma_{V_J}}{{\sigma_{V_I}}^2 + {\sigma_{V_J}}^2} \cdot \frac{\sigma_{V_I V_J}}{\sigma_{V_I}\sigma_{V_J}}$$

En la expresión, el primer término realiza una comparación entre la media de las distribuciones de la varianza local de ambas imágenes. El segundo compara la desviación estándar de las varianzas locales. El último introduce coherencia espacial.

Aunque la expresión de QILV se ha definido de forma semejante a la expresión del índice SSIM, este último es la media de los estadísticos locales de la imagen, mientras que QILV opera con los estadísticos globales de las varianzas locales de las imágenes.

3.5. EVALUACIÓN DE LA CALIDAD DE IMAGEN BASADA EN LA FIDELIDAD DE LA INFORMACIÓN VISUAL

Además de métodos que capturan la pérdida de estructura de la señal, existen otras posibilidades para implementar técnicas *full-reference.* Para lograr mejores predicciones de calidad, se estima que el modelado del sistema visual humano puede ser el planteamiento más adecuado. La premisa subyacente es que la sensibilidad del sistema visual es diferente para distintos aspectos de la señal recibida, como brillo, contraste, contenido frecuencial, interacción entre distintos componentes de la señal, y tiene sentido calcular el peso del error entre las señales de test y de referencia una vez que se han tenido en cuenta las distintas sensibilidades del HVS.

Los métodos de fidelidad de la información visual (*visual information fidelity* - VIF) incorporan modelos estadísticos de todos los componentes del sistema de comunicación: el transmisor, el canal y el receptor, [17] y [21]. Esta aproximación intenta relacionar la fidelidad o calidad de la señal con la cantidad de información que es compartida entre dos señales.

Las imágenes naturales de calidad perfecta pueden modelarse como la salida de una fuente estocástica. Si no existen distorsiones, esta señal pasa a través del canal HVS de un observador humano antes de entrar en el cerebro, que extrae información cognitiva. En el caso de imágenes distorsionadas, se supone que la señal de referencia pasa a través de un canal de distorsión antes de llegar al HVS.

La información compartida se cuantifica usando el concepto de información mutua, una medida ampliamente utilizada en teoría de la información. La medida propuesta en [17] se basa en la cuantificación de dos tipos de información mutua: la información mutua entre la entrada y la salida del canal HVS cuando el canal de distorsión no está presente (información de la imagen de referencia) y la información mutua entre la entrada del canal de distorsión y la salida del canal HVS para la imagen test. Debido a que la información mutua es una medida estadística que únicamente puede ser relacionada vagamente con la percepción humana de la información, esto establece límites en el cantidad de información relevante que puede extraerse de una señal, siempre que los modelos hipotéticos de la fuente, de la distorsión del canal y de la distorsión del receptor sean precisos.

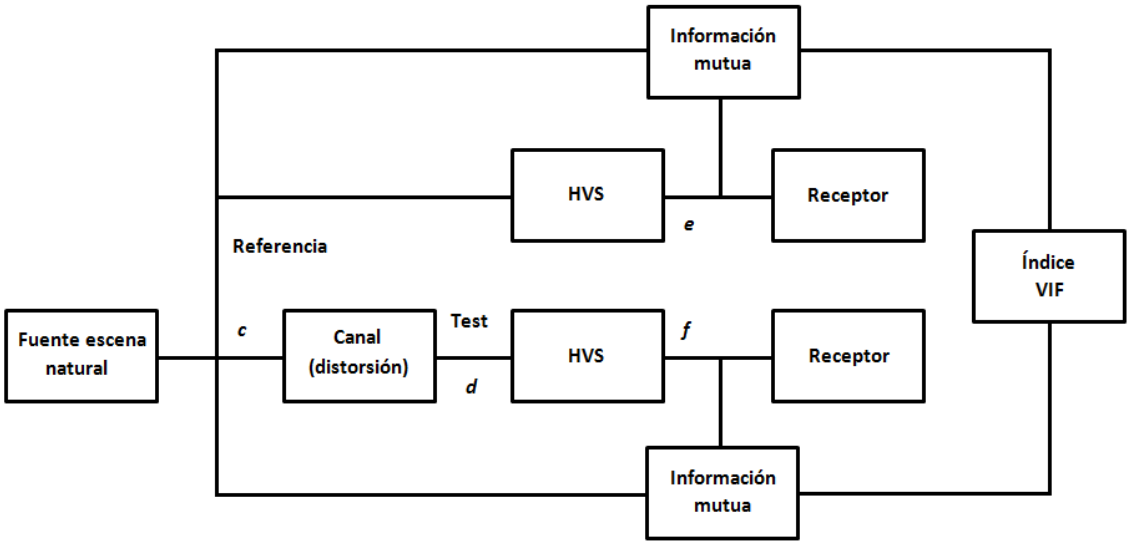


Figura 3.6. Esquema de un sistema VIF

La imagen de referencia se modela como una mezcla Gaussiana (GSM) en el dominio wavelet, que ha demostrado ser capaz de modelar eficientemente las distribuciones marginales no Gaussianas de los coeficientes wavelet de imágenes naturales, además de capturar las dependencias entre las magnitudes de coeficientes wavelet adyacentes. A continuación se presentan los modelos que caracterizan los elementos de un sistema de comunicaciones.

1) Modelo de fuente o transmisor: Sea $\boldsymbol{c}$ una colección de M coeficientes wavelet adyacentes extraídos de una parte de la sub-banda wavelet. Entonces el modelo GSM es sencillo: se caracteriza $\boldsymbol{c}$ como $\boldsymbol{c} = \sqrt{z}\boldsymbol{u}$, donde $\boldsymbol{u}$ es un vector Gaussiano de media nula y $\sqrt{z}$ es una variable aleatoria, escalar e independiente.
2) Modelo de distorsión: El propósito de un modelo de distorsión es describir cómo los estadísticos de una imagen son alterados por el operador de distorsión genérico. Se emplea para caracterizar todas las distorsiones que pueden ocurrir entre las señales de referencia y de prueba, incluyendo distorsiones artificiales como la compresión.

   El modelo asume que la distorsión de una imagen puede ser descrita como la combinación de la atenuación de energía uniforme en dominio wavelet con ruido aditivo: $\boldsymbol{d} = g\boldsymbol{c} + v$, donde $\boldsymbol{c}$ y $\boldsymbol{d}$ son vectores aleatorios extraídos de la misma localización en la misma sub-banda wavelet en las imágenes de referencia y distorsionada, respectivamente. Aquí, g es un factor de ganancia determinístico y escalar que representa una distorsión, mientras que v es ruido blanco Gaussiano de media nula, aditivo y con covarianza $C_v = \sigma_v^2 \boldsymbol{I}$. Aunque

este modelo puede ser considerado demasiado general por no considerar ningún tipo especifico de distorsión, proporciona una buena aproximación.

3) Modelo de receptor: El modelo de receptor, o modelo HVS, se considera como un "canal de distorsión" que limita la cantidad de información que puede atravesarlo, y su propósito es cuantificar la incertidumbre que el HVS añade a la señal. Por simplicidad, se agrupa todas las fuentes de incertidumbre del HSV en una componente de ruido aditivo, llamado *virtual noise* y se modela como ruido blanco Gaussiano de media nula estacionario en dominio wavelet:

$$\boldsymbol{e} = c + n\,;\ \boldsymbol{f} = d + n$$

Aquí, $\boldsymbol{e}$ y $\boldsymbol{f}$ son vectores de coeficientes aleatorios en la misma sub-banda wavelet en las imágenes de referencia y distorsionada, respectivamente, que denotan las señales a la salida del modelo HVS y de donde el cerebro extrae la información cognitiva. Por último, n es ruido blanco Gaussiano independiente con matriz de covarianza $C_n = \sigma_n^2 \boldsymbol{I}$.

3.5.1. ÍNDICE VIF

La información mutua $I(\boldsymbol{c}|\boldsymbol{e})$ cuantifica la cantidad de información que puede ser extraída de la salida del HVS por el cerebro cuando la imagen test está siendo visualizada. Sin embargo, estamos interesados en la calidad de un par de imágenes test-referencia particular, y no en la calidad media del conjunto de imágenes cuando pasan a través del canal de distorsión. Es, por tanto, razonable sintonizar el modelo de escena natural a una imagen de referencia específica tomando $I(\boldsymbol{c};\boldsymbol{e}|z)$ en lugar de $I(\boldsymbol{c}|\boldsymbol{e})$. De la misma forma para $I(\boldsymbol{c};\boldsymbol{f}|z)$.

Una vez conocidos los modelos estadísticos del transmisor, el canal y el receptor, la información mutua entre $\boldsymbol{c}$ y $\boldsymbol{e}$, y entre $\boldsymbol{c}$ y $\boldsymbol{f}$, es dado por:

$$I(\boldsymbol{c};\boldsymbol{e}|z) = \frac{1}{2}\log\frac{|zC_u + \sigma_n^2\boldsymbol{I}|}{|\sigma_n^2\boldsymbol{I}|} = \frac{1}{2}\sum_{j=1}^{M}\log\left(1 + \frac{z\lambda_j}{\sigma_n^2}\right)$$

$$I(\boldsymbol{c};\boldsymbol{f}|z) = \frac{1}{2}\log\frac{|g^2 zC_u + (\sigma_v^2 + \sigma_n^2)\boldsymbol{I}|}{|(\sigma_v^2 + \sigma_n^2)\boldsymbol{I}|} = \frac{1}{2}\sum_{j=1}^{M}\log\left(1 + \frac{g^2 z\lambda_j}{\sigma_v^2 + \sigma_n^2}\right)$$

En estas expresiones, la matriz de covarianza $\boldsymbol{C}_u = \boldsymbol{Q}\Lambda\boldsymbol{Q}^T$ se ha factorizado, donde Λ es una matriz diagonal cuyas entradas de la diagonal son los autovalores $\lambda_1, \lambda_2, \ldots, \lambda_M$.

La información mutua se calcula en cada localización espacial en cada sub-banda de la imagen, utilizando estimaciones de probabilidad máxima de z, g y σ_v. Si asumimos que existe independencia entre los parámetros de distorsión, la información mutua total puede calcularse como una simple suma.

Intuitivamente, la medida de calidad debe relacionar la cantidad de información de la imagen que el cerebro puede extraer de la imagen test en relación con la cantidad de información que puede extraer de la información de referencia. Esto se puede conseguir como la razón entre ambas medidas. Por tanto, finalmente, el índice VIF se define como la relación entre la información mutua sumada.

$$VIF = \frac{I(\boldsymbol{C};\boldsymbol{F}|z)}{I(\boldsymbol{C};\boldsymbol{E}|z)} = \frac{\sum_{i=1}^{N} I(c_i;f_i|z_i)}{\sum_{i=1}^{N} I(c_i;e_i|z_i)}$$

3.5.2. PROPIEDADES DE VIF

1) El índice VIF está limitado por debajo de cero, que sucede cuando $I(\boldsymbol{C};\boldsymbol{F}|z) = 0$ y $I(\boldsymbol{C};\boldsymbol{E}|z) \neq 0$, lo que significa que toda la información sobre la imagen se ha perdido.
2) Cuando la señal no está distorsionada de ninguna forma, entonces $g = 1$ y $\sigma_v^2 = 0$. Esto implica que $I(\boldsymbol{C};\boldsymbol{F}|z) = I(\boldsymbol{C};\boldsymbol{E}|z)$, y VIF es la unidad.
3) Para todos los tipos de distorsión en la práctica, el índice VIF está acotado en [0,1].
4) Un aumento del contraste de la imagen de referencia que no añade ruido, provoca que el índice VIF tome un valor mayor que la unidad, lo que significa que la imagen mejorada tiene mejor calidad que la imagen de referencia. Teóricamente, el aumento del contraste implica mayor SNR a la salida de las neuronas, permitiendo al cerebro mayor capacidad para distinguir objetos es la señal visual. El índice VIF es capaz de incluir esta mejora de la calidad visual. Esta característica lo diferencia de otros métodos tradicionales de medida de calidad.

En las siguientes imágenes, obtenidas de [17], se pueden comprobar las propiedades mencionadas. La figura (a) representa la imagen original que sirve de referencia para comparar con las demás, y consecuentemente se le asigna un índice VIF=1. En la imagen (b) se ha producido un realce del contraste, lo que mejora la calidad de la imagen. En este caso, la imagen recibe un índice VIF=1.10, lo cual es coherente con la propiedad 4 porque la imagen obtenida es de mejor calidad que la imagen original. En las figuras (c) y (d) la imagen de referencia ha sufrido operaciones de emborronamiento y de compresión mediante el algoritmo JPEG, respectivamente.

En ambos casos la calidad de la imagen se ha visto afectada y el valor de VIF será menor que la unidad. Concretamente, el emborronamiento provoca un VIF=0.07 y la compresión un VIF=0.10.

(a) Imagen de referencia VIF=1

(b) Mejora de contraste VIF=1.10

(c) Emborronamiento VIF=0.07

(d) Compresión JPEG VIF=0.10

Figura 3.7. Ejemplo de funcionamiento del índice VIF

3.6. EVALUACIÓN DE LA CALIDAD DE IMAGEN BASADA EN COMPARACIÓN DE HISTOGRAMAS

Una aproximación diferente a la medida de la calidad de imágenes consiste en utilizar medidas de similitud de lógica difusa (*fuzzy*) para comparar los histogramas de imágenes digitales. Comenzaremos presentando algunos conceptos sobre lógica difusa antes de ver cómo se aplican estos conceptos al análisis de imágenes, que se encuentran detallados en [16] y [25].

3.6.1. INTRODUCCIÓN

- **Conjuntos difusos (*fuzzy sets*)**

Un conjunto difuso A en un universo X se caracteriza por X- [0,1], que asocia con cada elemento x de X un grado de pertenencia $\mu_A(x)$ de x en el conjunto A. Vamos a denotar el grado de pertenencia como $A(x)$, y la clase de conjuntos difusos en un universo X como $F(X)$.

- **Medidas de semejanza**

Existen numerosas medidas propuestas para expresar la semejanza entre conjuntos difusos. La definición más común se puede ver como la relación binaria difusa en $F(X)$, tal que $F(X)$ x $F(X) \rightarrow [0,1]$, que cumple las propiedades de:

- Reflexión.
- Simetría.
- Mínimo transitiva.

- **Operadores de lógica difusa**

Las operaciones de la teoría de conjuntos *co* (complemento), ∩ (intersección), ∪ (unión) pueden extenderse a operaciones de conjuntos difusos. Basados en los operadores de la lógica tradicional, negación, conjunción, disyunción, aparecen operadores de *fuzzy logic* como *negator*, *conjunctor*, *disjunctor* o *implicator*.

De estos operadores de lógica difusa se derivan un gran número de clases de medidas de semejanza.

1) Un *negator* es un mapeo $[0,1] - [0,1]$ decreciente N, que satisface las condiciones de borde $N(0) = 1$ y $N(1) = 0$. El operador más popular es el *standard negator* $\mathrm{N_s}$, que se define como $\mathrm{N_s}(x) = 1 - x$ para todo x en $[0,1]$.
   El complemento A^c de un conjunto difuso A en X se define: $A^c(x) = 1 - A(x)$ para todo x en X.

2) Un *conjunctor* es un mapeo $[0,1]^2 - [0,1]$ creciente T que satisface las condiciones de borde $T(0,0) = T(0,1) = T(1,0) = T(1,1) = 1$. Los más comunes son:
   - $T_M(x,y) = \min(x,y)$.
   - $T_P(x,y) = x \cdot y$.
   - $T_W(x,y) = \max(0, x + y - 1)$ para todo (x,y) en $[0,1]^2$.

   La intersección $A \cap_T B$ de dos conjuntos A y B en un universo X se define como: $(A \cap_T B)(x) = T\big(A(x), B(x)\big) = \mathrm{Min}[A(x), B(x)]$.

3) Un *disjunctor* es un mapeo $[0,1]^2 - [0,1]$ creciente S que satisface las condiciones de borde $S(1,0) = S(0,1) = S(1,1) = 1$ y $S(0,0) = 0$. Los más populares son:
   - $S_M(x,y) = \max(x,y)$.
   - $S_P(x,y) = x + y - x \cdot y$.
   - $S_W(x,y) = \min(1, x+y)$ para todo (x,y) en $[0,1]^2$.

   La unión $A \cup_S B$ de dos conjuntos difusos A y B en un universo X se define como: $(A \cup_S B)(x) = S(A(x), B(x)) = \text{Max}[A(x), B(x)]$.

4) Un *implicator* es un mapeo $[0,1]^2 - [0,1]$, I, con mapas parciales decreciendo primero y aumentando después, que satisface las condiciones de borde $I(0,0) = I(0,1) = I(1,1) = 1$ y $I(1,0) = 0$. Los implicadores más importantes son:
   - $I_{KD}(x,y) = \max(1-x, y)$.
   - $I_W(x,y) = \min(1,1-x-y)$.
   - $I_R(x,y) = 1 - x - x \cdot y$, para todo (x,y) en $[0,1]^2$.

- **Propiedades relevantes para el procesado de imagen**

A continuación se exponen algunas características que deben cumplir las medidas de semejanza para poder aplicarse a la comparación de imágenes [25]:

- Reflexividad: Si se tienen dos imágenes idénticas, la medida de la semejanza proporciona una salida igual a la unidad.
- Simetría: La salida obtenida por la medida de similitud es independiente del orden en que consideren las dos imágenes de entrada.
- Reacción al ruido: Una medida no debe verse demasiado afectada por el ruido, ya que la imagen distorsionada debe ser similar a la original, y debe disminuir con respecto a un aumento del porcentaje de ruido.
- Reacción al oscurecimiento y a la iluminación: Si se aumenta o disminuye la luminosidad de una imagen con un valor constante, la medida debe generar un valor elevado.

3.6.2. COMPARACIÓN DE HISTOGRAMAS

- **Medidas clásicas de comparación de histogramas**

El histograma de una imagen en escala de grises es un gráfico que muestra la distribución de los diferentes niveles de gris. El valor del histograma de una imagen A en un nivel de gris g es el número total de píxeles en la imagen que tienen el nivel g, y se denota como $h_A(g)$. Así, la expresión del histograma de la imagen A se representa como:

$$h_A(g) = \sum_{(i,j)\in X} \delta(g - A(i,j))$$

Normalmente se emplean mediciones de distancia para determinar cuánto difieren dos histogramas. Como los histogramas pueden considerarse vectores, las métricas más comunes empleadas para histogramas son las basadas en la métrica de Minkowski (L_p), que se utiliza en espacios de vectores. Dos ejemplos son la *distancia Manhattan* y la *distancia Euclídea* , que tienen como expresiones:

$$d_{L1}(h_A, h_B) = \sum_{g=0}^{|G|-1} |h_A(g) - h_B(g)|$$

$$d_{L2}(h_A, h_B) = \sqrt{\sum_{g=0}^{|G|-1} |h_A(g) - h_B(g)|^2}$$

Otra técnica para la comparación de histogramas es la llamada intersección de histogramas. Para dos histogramas $h_A(g)$ y $h_B(g)$ la intersección de histogramas se define como:

$$d_I(h_A, h_B) = \sum_{g\in G} \min(h_A(g), h_B(g))$$

Sin embargo, si A y B tienen el mismo tamaño puede probarse que la suma de $d_{L1}(h_A, h_B)$ y $2d_I(h_A, h_B)$ es igual a una constante. Por consiguiente, no hay una diferencia significativa entre la intersección de histograma y la distancia Manhattan.

- **Aplicación directa de medidas de semejanza a histogramas**

Comparar dos histogramas en el contexto de la teoría de conjuntos difusos es razonable porque el histograma de una imagen puede transformarse en un conjunto difuso en el universo de los niveles de gris [25], dividiendo los valores del histograma para cada nivel de gris por el máximo número de píxeles con el mismo nivel de gris. De esta forma el valor de gris más común consigue el grado de pertenencia 1 en el conjunto difuso asociado al histograma, y cualquier otro valor de gris menos frecuente obtiene un grado de pertenencia más pequeño.

Por consiguiente, un histograma normalizado está en concordancia con la idea intuitiva detrás de un conjunto difuso: el elemento más típico de un universo tiene grado de pertenencia 1, mientras que los elementos menos comunes pertenecen al conjunto

difuso en menor medida, que se expresan mediante grados de pertenencia menores que la unidad. De esta manera, se obtiene la siguiente expresión del grado de pertenencia del valor de gris g en el conjunto difuso Fh_A asociado con el histograma h_A de la imagen A:

$$Fh_A(g) = \frac{h_A(g)}{h_A(gM)}$$

Con $h_A(gM) = \max_{g \in G} h_A(g)$. Como los histogramas de imágenes digitales pueden ser vistos como conjuntos difusos en el universo del rango de valores de grises, es interesante investigar si las medidas de semejanza introducidas originalmente para comparar dos conjuntos difusos pueden aplicarse a histogramas normalizados.

- **Aplicación de medidas de semejanza a histogramas ordenados**

Las medidas de similitud pueden aplicarse también de otra manera en histogramas asociados a imágenes digitales [25]. Los valores de un histograma pueden ordenarse de manera que el valor de gris menos frecuente se sitúa en la primera posición del histograma y el resto se ordenan de forma creciente. A continuación el histograma es normalizado de igual forma que en el caso anterior; y, por consiguiente, el nivel de gris más frecuente adopta el grado 1 del conjunto asociado al histograma y el resto de grados de pertenencia son menores que la unidad y están dispuestos en orden creciente.

Al contrario que ocurría en el caso de histogramas normalizados, donde las frecuencias de los distintos niveles de gris se comparaban nivel a nivel, en este caso la frecuencia del nivel de gris más recurrente en la imagen A se compara con el nivel de gris más recurrente en la imagen B, la frecuencia del segundo nivel de la imagen A se compara con el de la imagen B, y así sucesivamente se comparan todas las frecuencias de los diferentes niveles de gris en orden creciente de frecuencia. Puede ocurrir que dos frecuencias de dos valores de gris distintos se comparen, dependiendo de la posición que ocupen dentro del histograma ordenado. Si expresamos el histograma ordenado de una imagen A como o_A, obtenemos la siguiente expresión para el conjunto difuso asociado al histograma ordenado. Desde $i = 1, \dots |G|$, con G el universo de niveles de gris y con $o_A(g) = \max_{g \in G} h_A(g)$.

$$Oh_A(i) = \frac{o_A(i)}{o_A(|G|)}$$

3.7. MEDIDAS DE SEMEJANZA BASADAS EN VECINDAD

En este apartado veremos cómo se realizan medidas de semejanza basadas en vecindad utilizando lógica difusa, continuando con el punto de vista del apartado anterior.

Para ello, se comparan dos imágenes A y B, dividiendo ambas imágenes en partes de 8x8 y se calcula la semejanza entre las dos partes. Para calcular la semejanza se emplean las medidas basadas en píxeles. Algunas de estas medidas, desarrolladas en [22] son las siguientes:

$$S_1(A,B) = 1 - (\frac{1}{MN}\sum_{x\in X}|A(x) - B(x)|^r)^{\frac{1}{r}}$$

$$S_6(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Si la imagen es dividida en P partes y la semejanza entre la parte A_i de la imagen A y la parte B_i de la imagen B se denota por $S(A_i, B_i)$, entonces la semejanza entre las dos imágenes A y B se obtiene mediante la media ponderada de las similitudes en todas las partes:

$$S^n(A,B) = \frac{1}{P}\sum_{i=1}^{P} w_i \cdot S(A_i, B_i)$$

El peso w_i se define como la semejanza entre la homogeneidad h_{A_i} de la parte i en la imagen A y la homogeneidad h_{B_i} de la parte i en la imagen B. La homogeneidad se calcula como la semejanza entre el nivel de gris del píxel con mayor intensidad y el nivel de gris del píxel con menor intensidad, usando una relación de parecido s. Estos parámetros se definen de la siguiente forma:

$$s(x,y) = min(1, \max(0, \frac{6}{5} - 2|x-y|))$$

$$h_{A_i} = s(\max_{(x,y)\in A_i} A(x,y), \min_{(x,y)\in A_i} A(x,y))$$

$$w_i = s(h_{A_i}, h_{B_i})$$

Por último se considera la unión de las medidas de similitud basadas en vecindad y en histogramas para incorporar las características de la imagen de manera óptima. Esta nueva medida de la calidad se consigue simplemente multiplicando ambas componentes:

$$Q_{i,j}(A,B) = S_i^n(A,B) \cdot H_j(A,B)$$

De esta forma se obtiene un índice de calidad de imagen definido matemáticamente y universal, es decir, que no depende de las imágenes que están siendo comparadas, de las condiciones de visualización o de los observadores individuales.

3.8. PASO DE IMÁGENES A VÍDEOS

Después de haber visto distintos tipos de medidas de la calidad de imágenes, la pregunta que se plantea es si dichos métodos pueden aplicarse a imágenes en movimiento o vídeo [21]. A este respecto, existe una gran demanda por parte de la industria de comunicación multimedia de medidas de la calidad de vídeo (VQA – *video quality assessment*).

La forma más sencilla y obvia de implementación de VQA sería aplicar medidas de calidad de imagen a cada frame y luego calcular la media de todos los fotogramas. De hecho, SSIM se utiliza de esta forma con buenos resultados.

Sin embargo, dos aspectos importantes de las señales de vídeo no se tienen en cuenta en las implementaciones anteriores:

1) Existen fuertes correlaciones entre dos fotogramas adyacentes que definen estructuras temporales y espacio-temporales.
2) Las señales de vídeo contienen un importante movimiento estructurado.

La forma más común de tener en cuenta las correlaciones temporales es el filtrado temporal o espacio-temporal. Se aplican filtros lineales o bancos de filtros a lo largo de las direcciones espaciales y temporales, y las señales filtradas se normalizan para reflejar efectos perceptuales adicionales, como la función de sensibilidad al contraste CSF. Un ejemplo de VIF adaptado a señales de vídeo se tiene en [14].

El movimiento es uno de los aspectos más importantes de la información de los vídeos. A pesar de esto, pocos algoritmos VQA detectan movimiento explícitamente y emplean la información de movimiento directamente. El uso directo del movimiento es deseable ya que el filtrado temporal no puede capturar totalmente los movimientos. De hecho, el movimiento no crea todas las variaciones de intensidad temporal en vídeos, y la respuesta de los filtros temporales no puede proporcionar la velocidad de movimiento, ya que los atributos de intensidad afectan a la respuesta de los filtros.

Otros métodos más recientes buscan detectar movimiento y convertir la información de movimiento en factores de ponderación en la etapa de agrupación (*pooling*) de la medida de calidad.

Un enfoque diferente implica el uso de estimación del flujo óptico, para guiar de forma adaptativa el filtrado espacio-temporal utilizando bancos de filtros de Gabor tridimensionales. La diferencia de este método es que se selecciona un subconjunto de filtros en cada posición basándose en la dirección y la velocidad de movimiento, de forma que el eje principal del conjunto de filtros se orienta en la dirección de movimiento en el dominio de la frecuencia. La evaluación de la calidad se realiza sólo con los coeficientes calculados por los filtros seleccionados. Las distorsiones puramente espaciales, que se producen dentro de cada fotograma, provocan cambios en los componentes frecuenciales a lo largo del plano y son capturados en las salidas de los filtros. Las distorsiones puramente temporales, distorsiones entre fotogramas, resultan en cambios en el eje a lo largo del cual el plano intersecciona con los filtros de Gabor.

3.9. RESUMEN

	Ventajas	Inconvenientes
MSE/PSNR	- Fácil de calcular. Está libre de parámetros. Eficiente desde el punto de vista computacional. - No necesita memoria, cada muestra se calcula de forma independiente. - Tiene significado físico claro, define la energía de la señal de ruido. - Su uso está muy extendido en distintos campos.	- La forma en que MSE se representa no se corresponde con la percepción humana. - No es demasiado útil como media de calidad. - Dos imágenes con misma energía de ruido pueden tener diferente calidad, pero igual MSE. - Todas las muestras de señal tiene igual importancia y la medida no depende de las relaciones entre muestras.
SSIM	- Evalúa los cambios en la estructura de la imagen. - Los resultados generados son coherentes con la percepción humana. - Proporciona una buena medida para distintos tipos de imágenes.	- Calcula el índice global como la media de la varianza local de las dos imágenes. - No tiene en consideración la no-estacionariedad.
QILV	- Puede emplearse por separado o junto a otros métodos. - Supera las limitaciones de SSIM de no estacionariedad.	
VIF	- Se aproxima a la percepción humana mediante el modelado del sistema visual. - Tiene en cuenta posibles mejoras en la calidad de la imagen, por incremento del contraste	- El modelado de fuente, canal de dispersión y receptor puede ser simple, al no generalizar en ningún tipo de dispersión.
Fuzzy logic	- Combinar medidas de comparación de vecindad y de histogramas permite tener un índice de calidad universal y definido de forma matemática.	- Se basa en complejos desarrollos matemáticos.

Tabla 3.1. Resumen de métodos de evaluación de calidad

CAPÍTULO 4

DISEÑO DE LA APLICACIÓN

En este capítulo nos proponemos exponer los distintos pasos que se han seguido para desarrollar la aplicación de detección de movimiento. Dicha aplicación debe ser un método ligero y robusto a pequeñas alteraciones de iluminación y de movimiento de la cámara.

Como venimos diciendo, para la implementación de nuestra aplicación no vamos a emplear ninguno de los métodos descritos en el capítulo 2, sino que se utilizará el método SSIM enunciado en el apartado anterior, ya que nos permite realizar la tarea de detección de una manera sencilla y proporcionando buenos resultados.

La herramienta en la que nos vamos a apoyar para el desarrollo de este proyecto es el software Matlab; ya que sus características propias, como la presencia de *toolboxes* de funciones como el Image Processing Toolbox (IPT), le convierten en una de las mejores alternativas para el tratamiento de imágenes.

4.1. ETAPAS DE DISEÑO DEL ALGORITMO

Antes de centrarnos en la descripción de cada uno de los apartados del sistema propuesto, realizaremos una presentación del funcionamiento general, lo cual nos permitirá tener una visión global del sistema. Podemos distinguir en el sistema cuatro fases.

La primera etapa se corresponde con la función de calibrado, que comprende los primeros frames del video. Su cometido es determinar las regiones de la escena en las cuales no vamos a considerar su movimiento.

La segunda fase es la detección, que se realiza durante el resto del video. Se emplea el algoritmo SSIM para comprobar dónde se han producido movimientos.

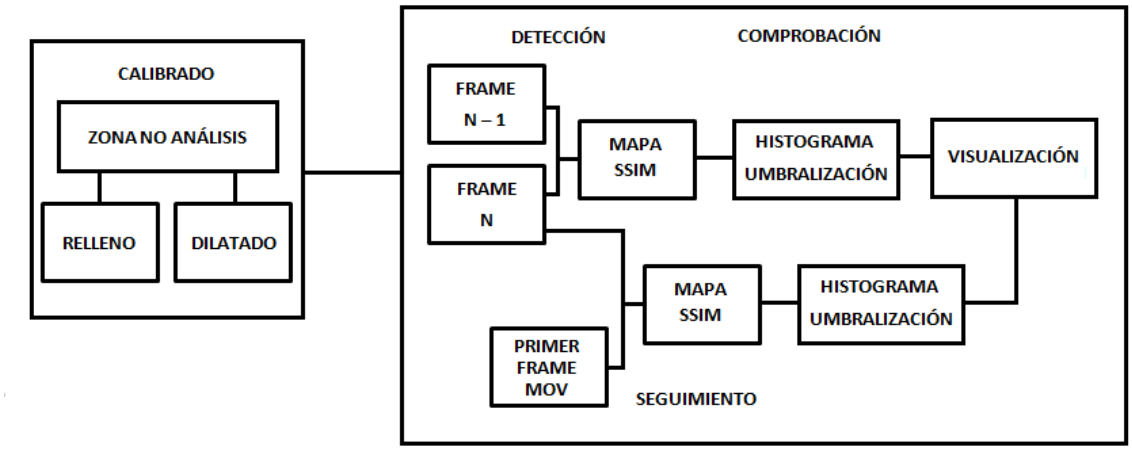


Figura 4.1. Esquema de la aplicación

A continuación, se comprueba que las zonas detectadas anteriormente corresponden con objetos en movimiento. Entre todos los puntos obtenidos, se decide cuáles se consideran como un desplazamiento.

Finalmente, la última fase corresponde con un análisis en paralelo al proceso anterior mientras un objeto se encuentra en movimiento. De esta manera es posible comprobar el punto de partida del movimiento y trayectoria por la escena, y, adicionalmente, podemos detectar movimientos más lentos que no ha sido posible percibir en las fases anteriores.

4.1.1. CALIBRADO

Dentro de una escena es frecuente que haya elementos con movimiento continuo o periódico, como por ejemplo el movimiento de las imágenes en la pantalla de una televisión o un ordenador, el movimiento de un animal en su jaula, etc. Estos movimientos no son relevantes si lo que queremos es detectar la presencia de personas u otro tipo de objetos, pero provocarían que el algoritmo de detección se activase y se produjeran falsas detecciones.

Por tanto, el primer paso del algoritmo consiste en determinar las zonas de la escena donde se producen este tipo de movimientos para descartarlos. Consideramos que previamente al inicio del proceso de detección en sí mismo existe una fase en la que la cámara captura los puntos que no deben considerarse para el tratamiento posterior.

Para ello dedicamos los primeros fotogramas del video a hacer este procesado. El número de fotogramas es variable y puede elegirse en función del grado de precisión

que se desee a la hora de eliminar estos movimientos. Como estos movimientos son aleatorios, no es posible determinar con exactitud el cuántos fotogramas deben dedicarse al proceso de calibrado ni tener la certeza de que todos los movimientos de este tipo que puedan producirse hayan sido detectados.

La implementación del calibrado comprende los siguientes pasos:

- Se van leyendo sucesivamente los fotogramas que comprenden la secuencia de video y por parejas se aplica la función SSIM. Como resultado obtenemos el mapa SSIM correspondiente a las diferencias existentes entre ambas imágenes.
- El mapa SSIM es una imagen en escala de grises, cuyos píxeles toman valores entre 0 (negro) y 1 (blanco). El algoritmo SSIM funciona de tal manera que los tonos más claros de la imagen se corresponden con aquellos puntos en los que las dos imágenes comparadas tienen mayor similitud, mientras que las zonas más oscuras se corresponden con aquellas en las que la correspondencia es menor, es decir, se ha producido un movimiento.
- Para poder interpretar mejor los resultados es necesario convertir la imagen en escala de grises en una imagen binaria, para lo cual hay que realizar la elección de establecer un umbral. Para establecer el umbral debemos conocer que valores de los píxeles del mapa SSIM se corresponden con movimientos y cuales con alteraciones de la imagen provocadas por impurezas, fallos debidos a compresión…, para lo cual hacemos uso del histograma de la imagen.
- **Umbralización:** Debido a sus propiedades intuitivas y a la sencillez de su implementación, establecer un umbral en una imagen es un punto importante en aplicaciones de segmentación de imágenes.

  Podemos ver cómo establecer un correcto umbral partiendo del histograma de una imagen en escala de grises $f(x,y)$ compuesta por objetos claros en un fondo oscuro, de modo que los píxeles correspondientes a los objetos y al fondo tienen valores que se agrupan en torno a dos niveles. Una forma evidente de extraer los objetos del fondo es elegir un umbral T que se encuentra entre ambos niveles. Entonces cualquier punto (x,y) para el cual $f(x,y) > T$ pertenece a los objetos, y en caso contrario corresponde con el fondo.

  Finalmente, una imagen después de aplicar el umbral se convierte en una señal binaria definida de la siguiente forma:

$$g(x,y) = \begin{cases} 1, & si\ f(x,y) > T \\ 0, & si\ f(x,y) \leq T \end{cases}$$

Cuando T depende sólo de $f(x,y)$, es decir, de los valores de gris, T es un umbral global. Si depende tanto de $f(x,y)$ como de alguna propiedad local del punto (x,y) entonces es un umbral local. Además, si T depende de las coordenadas espaciales x e y, entonces el umbral es dinámico o adaptativo.

Existen métodos automáticos para determinar el umbral más apropiado para cada imagen. Uno de los más frecuentes y que está implementado en Matlab en la función ***graythresh***, es el método de Otsu.

- Como el movimiento que estamos tratando es aleatorio, es posible que zonas adyacentes a las que estamos detectando también sufran movimiento. Para tener esto en cuenta vamos a realizar unas operaciones de dilatación y de relleno de elementos en la imagen binaria. Ambos procedimientos pertenecen a lo que se conoce como operaciones morfológicas, que se encargan de extraer las componentes de la imagen que son útiles en la representación y descripción de la forma de los objetos.
- **Dilatación:** La operación de dilatación es fundamental en el procesado de imagen morfológico. Es una operación que agranda o aumenta el tamaño de objetos en una imagen binaria. La manera específica y el alcance con el que se desarrolla el engrosamiento depende del llamado elemento estructurador. Computacionalmente, los elementos estructuradores normalmente se representan mediante una matriz formada por 1s y 0s. Gráficamente, el proceso de dilatación se realiza trasladando el origen del elemento estructurador por toda la imagen y comprobando dónde se solapa con píxeles de valor 1. La imagen de salida toma valor 1 en cada punto en que está situado el centro tal que el elemento estructurador se solapa al menos con un píxel con valor 1 de la imagen de entrada.

  Matemáticamente, la dilatación se expresa como operaciones de conjuntos. La dilatación de A por B, se define:

$$A \oplus B = \{z |\ (\tilde{B})_z \cap A \neq \emptyset\}$$

  Donde $\emptyset$ es el conjunto vacío, B es el elemento estructurador y $\tilde{B}$ es el reflejo de B. Esto es, la dilatación de A por B es el conjunto formado por todas las localizaciones del centro de los elementos estructuradores donde B trasladado se superpone al menos alguna parte de A.

  La dilatación es un proceso conmutativo, es decir $A \oplus B = B \oplus A$. Se considera como convenio en el procesado de imagen que el primer término de $A \oplus B$ es la imagen y el segundo se corresponde con el elemento estructurador, que normalmente es de menor tamaño que la imagen.

  La función que lleva a cabo la dilatación de una imagen binaria o en escala de grises es ***imdilate***. Dicha función necesita como argumentos de entrada la

imagen a dilatar y un elemento estructurador. El elemento estructurador puede tener varias formas, ser simétrico o no; en nuestra caso hemos empleado una matriz 7x7.

0	0	0	1	0	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
1	1	1	1	1	1	1
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	0	0	1	0	0	0

- **Relleno:** La reconstrucción es una transformación morfológica que incluye dos imágenes y un elemento estructurador (en lugar de sólo una imagen y un elemento estructurador). Una imagen, el marcador, es el punto de partida para la transformación. La otra imagen, la máscara, limita la transformación. El elemento estructurador usado define la conectividad. Por ejemplo, un elemento con conectividad 8, es una matriz 3x3 de 1s cuyo centro se encuentra en (2,2).

  Si g es la máscara y f es el marcador, la reconstrucción de g a partir de f, denotado como $R_g(f)$, se define por el siguiente proceso iterativo:

  1. Inicializar h_1 como la imagen marcador f.
  2. Crear el elemento estructurador.
  3. Repetir: $h_{k+1} = (h_k \oplus B) \cap g$ hasta que $h_{k+1} = h_k$.

  La reconstrucción morfológica tiene un gran número de aplicaciones, cada una determinada por la selección de las imágenes marcador y máscara. Si elegimos la imagen marcador, f_m, para que sea 0 en toda la imagen excepto en el borde de la imagen, donde $1 - f$:

$$f_m(x,y) = \begin{cases} 1 - f(x,y), si\ (x,y) está\ en\ el\ borde\ de\ f \\ 0, \qquad en\ cualquier\ otro\ punto \end{cases}$$

  Entonces $g = [R_{f^c}(f_m)]^c$ tiene el efecto de llenar los huecos en f.

  En Matlab podemos realizar el relleno de una imagen binaria empleando la función ***imfill***, que utiliza el algoritmo de *flood-fill*.
- Realizar las operaciones anteriores genera una imagen formada por 1s (zonas sin movimiento) y 0s (zonas de movimiento). Durante todo el proceso de calibrado vamos sumando entre sí todas estas imágenes que se generan en cada iteración. El resultado final es una matriz formada de números enteros, 0 para los píxeles sin movimiento y mayor que 0 para los píxeles en los que ha

habido movimiento al menos una vez. Cuanto mayor sea el valor del número, más veces el píxel de la imagen al que corresponde ha sufrido un movimiento.

- Tenemos que convertir de nuevo la imagen en binaria, para ello podemos establecer como umbral el valor 0 o elegir un número mayor. De esta manera podemos seleccionar un grado de sensibilidad a la hora de determinar las zonas de la imagen que no deben tenerse en cuenta para el procesado posterior. Podemos considerar que los valores más bajos (las zonas de movimiento que menos se ha repetido) no son significativas, porque podrían tratarse de falsas detecciones.
- Finalmente obtenemos la zona de la imagen que no va ha ser evaluada en el resto de la aplicación.

4.1.2. DETECCIÓN

La segunda etapa es la parte central de la aplicación. Consiste en detectar movimiento a partir de dos fotogramas de la secuencia de vídeo utilizando SSIM.

Comenzamos después del último fotograma empleado en la fase de calibrado. Este proceso funciona como un bucle y en cada pasada del bucle vamos obteniendo unos fotogramas. El sistema necesita para su funcionamiento capturar en cada ciclo el fotograma del momento actual y un fotograma anterior.

Partimos de un archivo de video en formato *avi*, el cual hemos cambiado del formato original producido por la cámara con un software conversor de formato. En nuestro caso hemos utilizado el programa Kigo Video Converter.

Cargamos dicho archivo .avi mediante la función de Matlab ***VideoReader***, que permite leer el vídeo y crea una estructura en la que se recoge toda la información. A continuación se muestra una captura que indica los parámetros importantes de un vídeo que contiene la estructura: duración, tasa y número de frames, tipo de imagen, altura y anchura de fotograma, número de bits por píxel…

```
General Settings:
  Duration = 28.0800
  Name = calle7.avi
  Path = C:\Users\Rafael\Desktop\MATLAB R2011b full\Luz
  Tag =
  Type = VideoReader
  UserData = []
```

```
Video Settings:
  BitsPerPixel = 24
  FrameRate = 25
  Height = 240
  NumberOfFrames = 702
  VideoFormat = RGB24
  Width = 320
```

Utilizando el objeto creado mediante ***VideoReader*** podemos leer cualquier fotograma del vídeo mediante la función ***read***, indicando únicamente el objeto de origen y el número del fotograma que queremos obtener.

Dado que la manipulación de imágenes va a ser una constante durante todo el procedimiento, vamos a describir los tres tipos de imágenes que emplearemos en nuestro programa y que están definidos por el IPT. Estos tipos, determinan cómo Matlab interpreta los elementos de la matriz de datos como valores de intensidad de píxel.

- **Binarias**
  Una imagen binaria es un array lógico formado únicamente por 0s y 1s, que se interpretan en la imagen como negro y blanco respectivamente.

- **Escala de grises**
  Una imagen de escala de grises es una matriz de datos cuyos valores representa niveles de intensidad dentro de un rango. Se definen como arrays de clase uint8, uint16, int16, single o double. En las clases single y double el rango de valores transcurre de 0 a 1, para uint8 de [0, 255], para la clase uint16 de [0, 65535] y para int16 [-32768, 32767].

- **RGB**
  Las imágenes RGB o imágenes de color verdadero (*true color*) son imágenes en las cuales cada píxel está especificado por tres valores, cada uno para las componentes roja, verde y azul del color del píxel. Matalb almacena las imágenes RGB como un array M x N x 3. Una imagen RGB puede verse como tres imágenes en escala de grises apiladas, que cuando se aplican a las entradas roja, verde y azul de un monitor muestran una imagen a color en la pantalla.

  A continuación se muestra un ejemplo del resultado de descomponer una imagen de color verdadero en las tres componentes roja, verde y azul, y también pueden verse las diferencias existentes entre ellas. Cuanto más blanca sea una región de una componente, más intenso será el color al que corresponda dicha componente en la imagen RGB.

Figura 4.2. Descomposición de una imagen RGB en sus tres componentes

Igualmente las imágenes RGB pueden ser de diferentes clases: double, uint8 o uint16. El número de bits empleados para representar los valores de los píxeles en las tres imágenes se conoce como profundidad de color o *bit-depth* de una imagen RGB. Normalmente las tres componentes utilizan el mismo número de bits, por tanto, la profundidad de color de una imagen RGB se puede calcular como $(2^n)^3$, donde n es el número de bits en cada imagen. En el caso de que cada imagen se represente por 8 bits, la imagen RGB tiene una profundidad de color de 24 bits, llamada en este caso imagen de Color Verdadero, y permite disponer de $(2^8)^3$= 16.777.216 colores.

Fuera del ámbito de Matlab, además de imágenes RGB existen otros tipos de representaciones de imágenes a color. Los diferentes tipos de imágenes se diferencian por el llamado *espacio de color*, que indica el sistema de coordenadas y el subespacio empleado para determinar cada color que corresponde a un punto.

Existen gran variedad de espacios de color disponibles en la actualidad, que se usan según determinadas aplicaciones. Mientras que el espacio RGB se utiliza

para representar el color en monitores y cámaras de vídeo, los espacios CMY (Cyan, Magenta, Yellow) y CMYK (Cyan, Magenta, Yellow, Black) se utilizan en impresoras. El modelo HSI (Hue, Saturation, Intensity), que se aproxima a la forma que el ser humano describe e interpreta los colores, permite separar la información de color y de gris en una imagen, haciéndolo apropiado para el manejo de técnicas de escalas de grises.

En concreto, el modelo RGB se basa en el espacio de coordenadas cartesianas. El subespacio de interés es un cubo que tiene en los vértices que corresponden con los ejes los colores rojo, verde y azul; en los otros vértices los colores secundarios cian, magenta y amarillo; el color negro se encuentra en el origen de coordenadas y el blanco en el vértice opuesto. Los colores grises (puntos que tienen el mismo valor RGB, o mismo valor en las tres capas) se sitúan en el eje que une el vértice blanco y negro. Los diferentes colores son puntos que se localizan dentro o en la superficie del cubo, y se definen como el vector que los une con el origen.

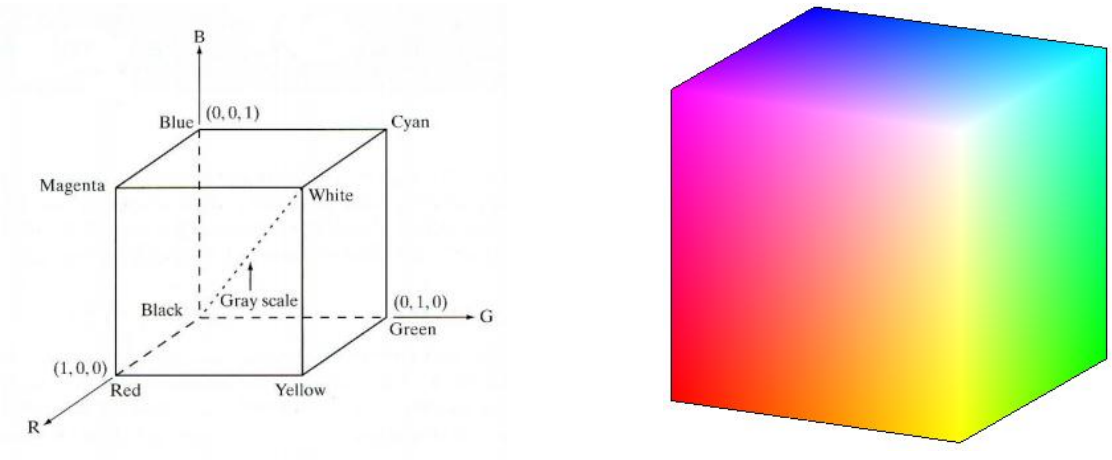


Figura 4.3. Espacio de color RGB

Por otra parte, para visualizar la imagen tenemos varias alternativas. La función ***imshow*** es la función principal del toolbox para mostrar imágenes binarias, en escala de grises o de color verdadero. Por otra parte, ***imtool*** permite el acceso a herramientas para estudiar en profundidad las imágenes, como medir distancias, explorar el valor de cada píxel, etc. Otro ejemplo de visualización es la función ***imagesc***, que escala los valores de los datos de la imagen al rango completo del mapa de color actual.

En la siguiente figura pueden verse las diferencias existentes entre utilizar la función *imagesc* y la función *imshow* al visualizar dos imágenes.

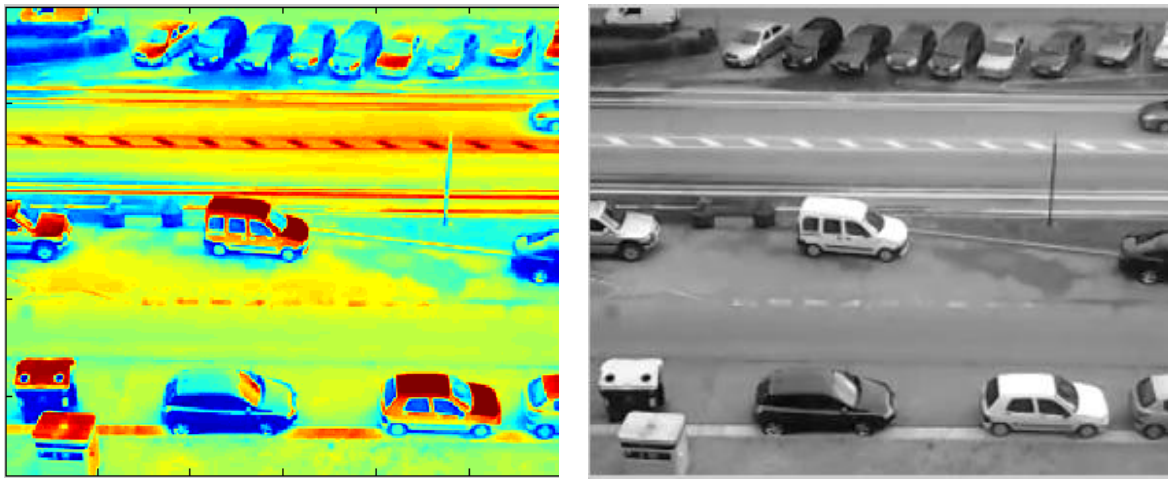


Figura 4.4. Representación de una imagen en escala de grises mediante las funciones imagesc y imshow respectivamente

Para poder utilizar el algoritmo SSIM necesitamos a la entrada imágenes en escala de grises, por tanto debemos convertir las imágenes originales en color a imágenes en escala de grises. La conversión se realiza eliminando el color y la saturación de la imagen y manteniendo la luminancia. La función ***rgb2gray*** realiza la transformación mediante la suma ponderada de los componentes R, G y B de la siguiente manera:

$$0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B$$

En las imágenes siguientes se muestra el resultado de aplicar la función ***rgb2gray*** a una imagen real en color.

Figura 4.5. Transformación de la imagen RGB a escala de grises utilizando la función rgb2gray

Como la imagen original capturada por la cámara tiene utiliza 24 bits por píxel, es decir tiene profundidad de color de 24, la imagen en escala de grises contará con 8 bits, lo que permite representar una totalidad de 256 tonalidades de gris.

- **Algoritmo SSIM**

El siguiente paso consiste en aplicar el algoritmo SSIM para determinar la similitud entre las dos imágenes. En el capítulo 3 se explicaron los conceptos básicos de SSIM y las características del funcionamiento de la función en Matlab, disponible en [26].

La función ***ssim*** devuelve el mapa SSIM que supone la comparación píxel a píxel entre las dos imágenes. Según está definida la función, los objetos que están en movimiento aparecen representados con los colores más oscuros mientras que el fondo inmóvil se representa con los colores más claros. Es importante tener esto en cuenta ya que está denominación es contraria a la mayoría de las funciones de Matlab de procesado morfológico donde los objetos se suelen representar de color blanco (bit 1) y el fondo de color negro (bit 0).

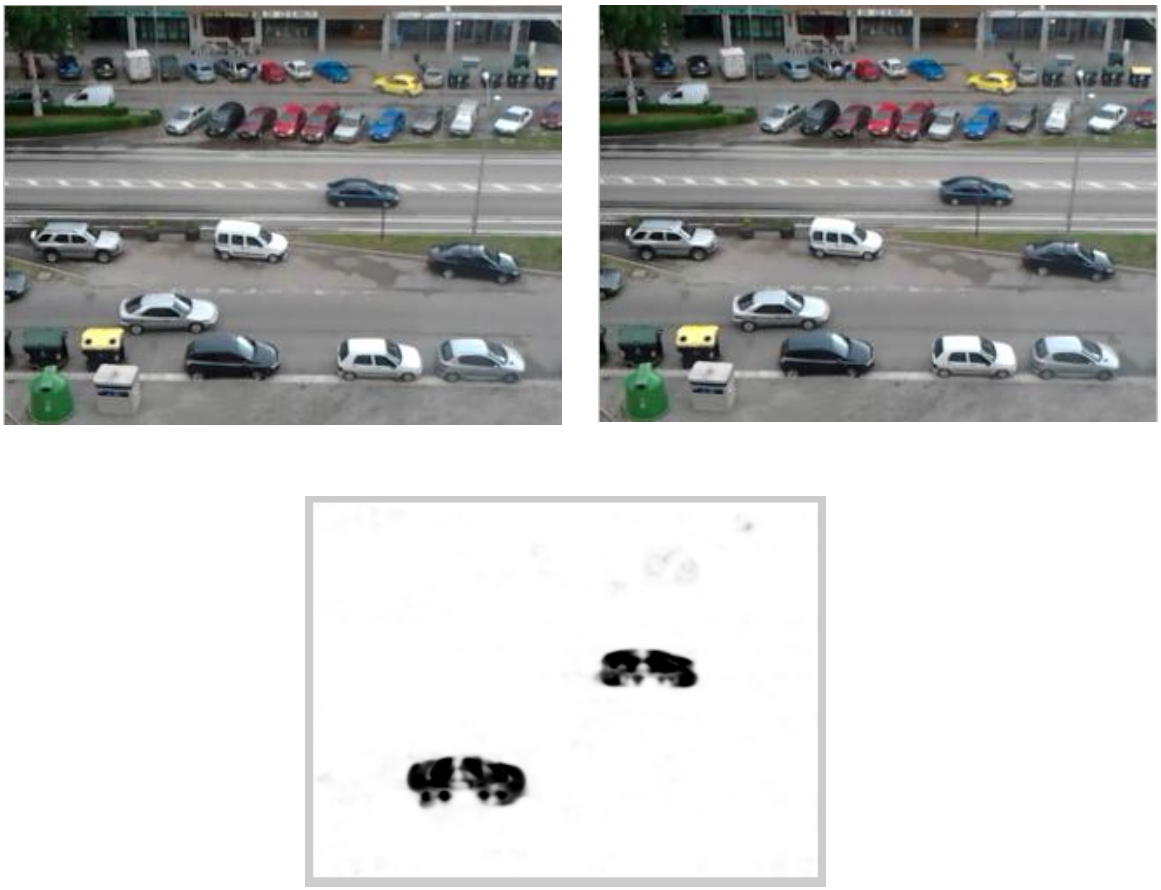


Figura 4.6. Mapa SSIM correspondiente a la comparación de las dos imágenes superiores

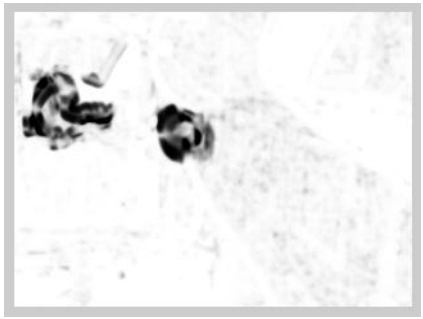
En las imágenes de la figura 4.6 se puede comprobar que el mapa SSIM resultante de aplicar el algoritmo a dos fotogramas detecta el movimiento de dos coches, representando las zonas de la imagen que corresponden a los vehículos en movimiento con píxeles oscuros. También puede observarse que el mapa SSIM obtenido es de menor tamaño que las imágenes originales, debido principalmente al tamaño de la ventana utilizada. En el caso concreto del ejemplo anterior, las imágenes tomadas de la cámara tienen un tamaño de 240x320 píxeles mientras que el mapa SSIM es de tamaño 230x310 píxeles.

(a) Fotograma durante el calibrado

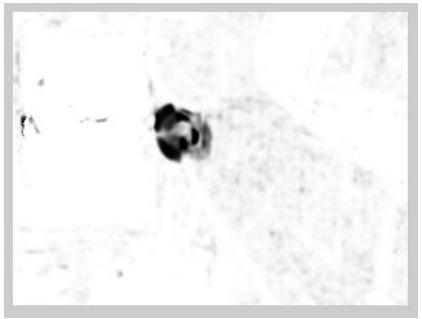
(b) Fondo generado

(c) Fotograma N-1

(d) Fotograma N



(e) Mapa SSIM



(f) Mapa SSIM+fondo

Figura 4.7. Empleo de la máscara del calibrado al análisis final

- **Suma del mapa SSIM con la máscara.**

Este paso consiste en sumar a la imagen *fondo* producida durante el calibrado el mapa SSIM generado anteriormente para discriminar las zonas que no deben ser analizadas. Dado que en la máscara *fondo* los bits que no deben tenerse en cuenta toman valor 1 (color blanco) y el resto son bits de valor 0 (color negro), las zonas negras del *fondo* no implican ningún cambio mientras que las blancas (valor 1) se suman y provocan que el valor del píxel final sea: 1+x donde x es el valor del píxel en el mapa SSIM. En la representación gráfica de la imagen final sigue el criterio que hemos tomado respecto a las imágenes SSIM, ya que las zonas que no se evalúan aparecen siempre en blanco (como si ni hubiera movimiento).

En la figura 4.7 podemos ver cómo funciona lo comentado anteriormente. Las imágenes (a) y (b) corresponden con la fase de calibrado donde el movimiento de las imágenes en la pantalla de un televisor genera una zona, que hemos llamado *fondo,* dentro de la cual no vamos a comprobar el movimiento. En (c) y (d) tenemos dos fotogramas en la fase de detección en los cuales hay un objeto, una pelota en este caso, en movimiento. En (e) se muestra el mapa SSIM generado por los dos fotogramas anteriores y en el que se puede apreciar las zonas de movimiento correspondientes al objeto y a la pantalla de la televisión. Finalmente, (f) presenta el resultado de añadir el *fondo* o máscara generado en el calibrado al mapa SSIM para eliminar el movimiento provocado por la televisión para el análisis posterior.

4.1.3. COMPROBACIÓN

Después de generar el mapa SSIM es necesario determinar los píxeles del mismo que se corresponden con puntos de movimiento y cuáles se deben desestimar por ser provocados por ruido, cambios de iluminación… El procedimiento es similar al que hemos indicado en la fase de calibrado, salvo que esta vez no vamos a realizar el binarizado de la imagen.

Comprobamos la evolución del histograma del mapa SSIM para establecer un umbral adecuado.

- **Histograma [3]**
  El histograma de una imagen digital de escala de grises en el rango [0, L-1] es una función discreta $h(r_k) = n_k$, donde r_k es el nivel de gris k-ésimo y n_k es el número de píxeles en la imagen que tienen en nivel de gris r_k. En Matlab el valor de L-1 es 255 para imágenes de clase uint8, que son del tipo que vamos a utilizar en nuestro programa, y 65535 para uint16.

Una práctica frecuente consiste en normalizar el histograma dividiendo cada uno de sus valores por el número total de píxeles en la imagen, denotado como n. Por tanto, un histograma normalizado se puede expresar como $p(r_k) = n_k/n$, con $k = 0, 1, \dots, L-1$. En términos generales, $p(r_k)$ proporciona una estimación de la probabilidad de que el nivel de gris r_k aparezca. Hay que tener en cuenta que la suma de todas las componentes de un histograma normalizado es igual a 1.

Los histogramas son la base de numerosas técnicas de procesado en el dominio espacial. La manipulación del histograma puede utilizarse para la mejora de la imagen. Además de proporcionar estadísticos importantes de la imagen, la información inherente de un histograma es útil para otras aplicaciones de procesado como compresión y segmentación de imágenes. Los histogramas son sencillos de calcular en software y se prestan a implementaciones en hardware económicas, lo que los convierte en herramientas populares para el procesado en tiempo real.

En nuestra aplicación, los histogramas nos van a servir para determinar qué valores toman y cómo se distribuyen los píxeles que se corresponden con objetos en movimiento para distinguirlos del fondo inmóvil.

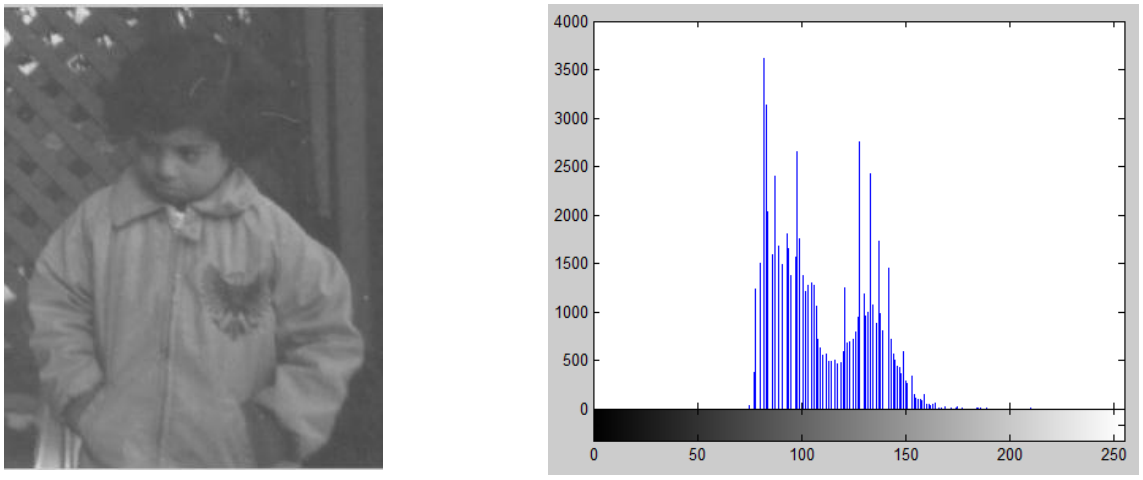


Figura 4.8. Ejemplo de histograma de una imagen

La representación grafica de un histograma consiste simplemente en la visualización de n_k frente a r_k. El eje horizontal corresponde los valores r_k de niveles de gris, y el eje vertical corresponde con el número de píxeles n_k para cada nivel.

Después de determinar el umbral, vamos a marcar sobre la imagen original aquellos píxeles que sufren un movimiento que están por debajo del umbral. Representaremos

dichos píxeles de color rojo, y para ello tenemos que recordar que una imagen RGB está formada a su vez por tres imágenes en escala de grises. El color rojo en el espacio RGB normalizado se representa como [1 0 0] y en imágenes de 8 bits como las que estamos utilizando por [255 0 0]. Entones tenemos que sustituir los valores de los píxeles que queremos modificar por 255 en la primera imagen del conjunto, y por 0 en la segunda y la tercera.

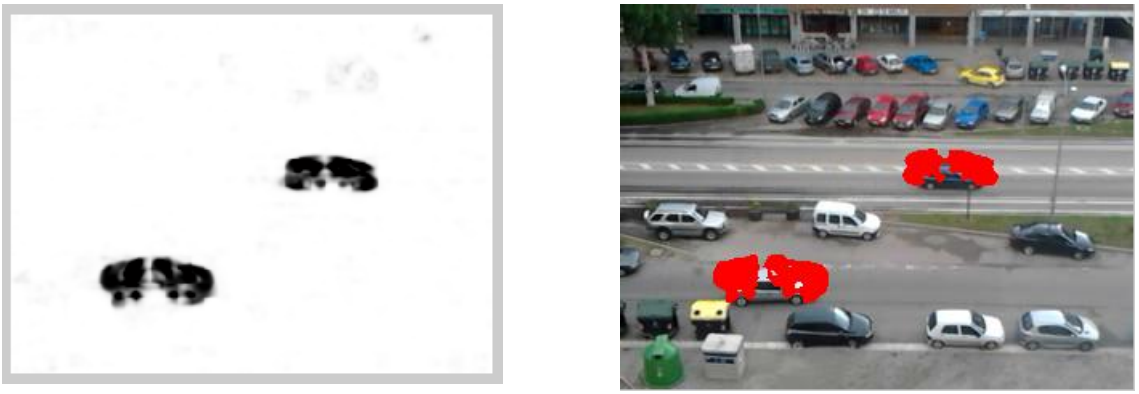


Figura 4.9. Zonas de movimiento sobre la imagen original

Además de representar los objetos en movimiento como los puntos del mapa SSIM que están por debajo del umbral, podemos elegir otros tipos de representaciones, como utilizar rectángulos o puntos.

- Para dibujar un rectángulo alrededor de un objetivo en movimiento, en primer lugar necesitamos conocer los objetos presentes en la escena. Para ello creamos una imagen binaria del mapa SSIM utilizando como umbral el mismo que en el paso anterior. Es posible que los objetos detectados estén divididos en varias partes, por lo que vamos a realizar operaciones de dilatado y expansión sobre la imagen binaria, la cual hemos invertido previamente como ya se ha indicado con anterioridad, para que los objetos tengan forma compacta.

  A continuación utilizamos la función ***bwconncomp***, que encuentra los elementos conectados de una imagen binaria. Esta función devuelve una estructura con los componentes conectados y que tiene cuatro apartados:

```
Connectivity: 4
   ImageSize: [230 310]
  NumObjects: 3
PixelIdxList: {[914x1 double]  [2691x1 double]  [3410x1 double]}
```

Una vez que tenemos los elementos de la imagen binaria, podemos extraer características de ellos empleando la función ***regionprops***. Con esta función es posible conocer propiedades como el área, el perímetro, el diámetro equivalente, el centroide, etc. En concreto nosotros vamos a elegir la opción ***'BoundingBox'***, que crea un array de estructuras con tantas estructuras como el número de elementos hay en la imagen y cada una de ellos contiene un vector con cuatro valores que permiten construir el rectángulo más pequeño: coordenadas X e Y del píxel situado en la esquina superior izquierda y los valores de la anchura la y altura del rectángulo.

Conocidos estos datos, podemos dibujar los rectángulos sobre la imagen RGB de color verde, dando a los puntos que pertenecen el rectángulo los valores [0 255 0]. Si tenemos en cuenta que al inicio de todo este procedimiento expandimos la imagen binaria original, el tamaño de rectángulo que estamos considerando es mayor que el objeto real al que queremos aproximarnos. Para solucionarlo, podemos reducir el tamaño del rectángulo disminuyendo sus medidas por un valor estimado a partir del tamaño del elemento estructurador empleado para realizar la expansión.

- Podemos dibujar un punto que represente un objeto fácilmente mediante dos opciones:
    - Calcular el punto central del rectángulo calculado anteriormente.
    - Utilizando la función ***regionprops*** junto a la opción ***Centroid*** sobre la imagen binaria, de manera similar a lo que se hizo para el rectángulo, para obtener las coordenadas X e Y del centroide del objeto.

  Este punto lo dibujamos sobre la imagen de color amarillo, que en el espacio RGB resulta de la combinación de rojo y verde, por lo que damos a los píxeles los valores [255, 255, 0].

4.1.4. SEGUIMIENTO

La última funcionalidad implementada consiste en determinar el movimiento relativo que ha seguido un objeto o una persona por la escena. El sistema se basa en emplear el algoritmo SSIM sobre el fotograma actual que se está procesando y el primer fotograma en el que el objeto en cuestión aparece en movimiento en la escena. Dicho procedimiento se desarrolla en paralelo al comentado en el apartado anterior.

Con esto se pretende conseguir ser capaces de detectar la presencia de una persona u objeto cuando se ha detenido después de haber estado en movimiento y de haber sido

detectado por el mecanismo de la etapa anterior. Adicionalmente, también es posible detectar elementos que se desplazan a una velocidad menor o tienen un movimiento más reducido. Aunque la primera fase no sea capaz de detectarlos porque la diferencia entre dos fotogramas consecutivos no sea suficientemente grande, al estar comparando fotogramas que están más separados la diferencia se hace patente.

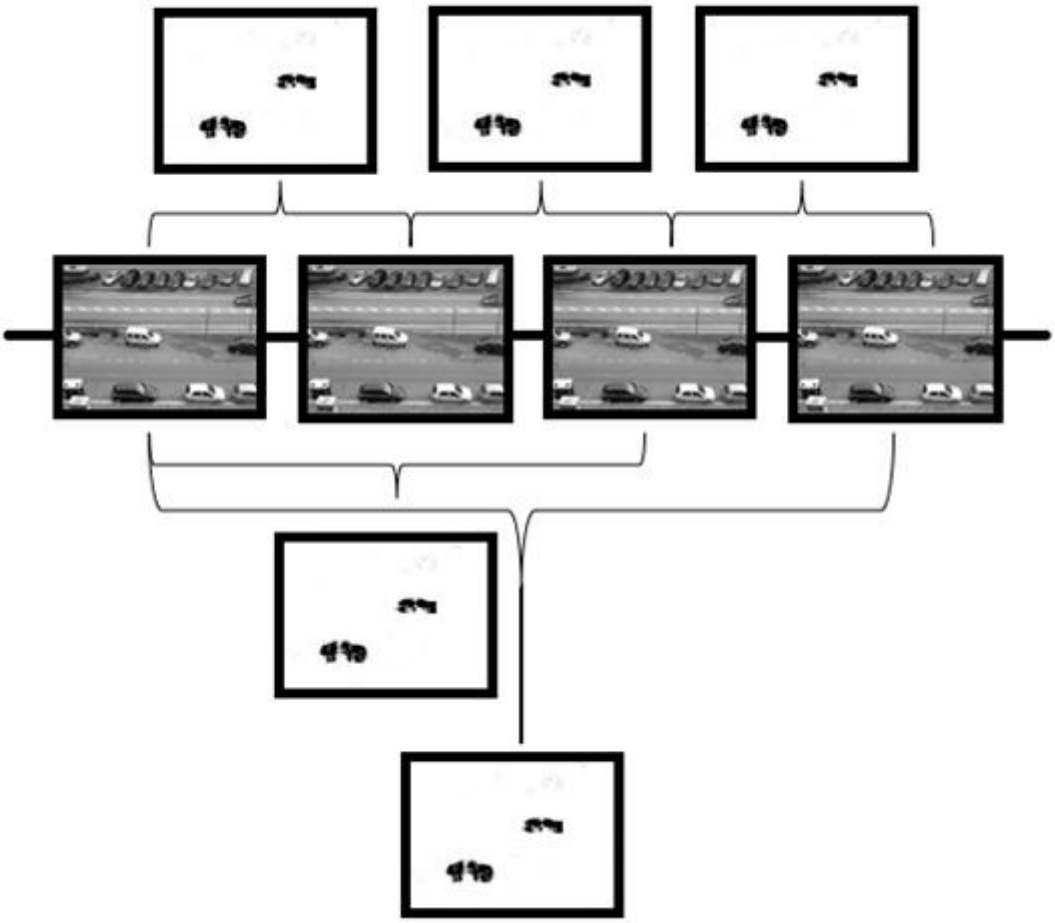


Figura 4.10. Esquema del funcionamiento de la doble detección

4.2. ALGORITMO SSIM EN MATLAB

Para concluir con este capítulo dedicado al diseño y la implementación de la aplicación, vamos a explicar cómo se maneja el algoritmo SSIM que hemos utilizado en nuestro trabajo. El código de Matlab que empleamos pertenece al desarrollador del método, el Dr. Zhou Wang de la universidad de Waterloo, que dispone de una versión del algoritmo de SSIM implementado en Matlab en [26].

A modo de recordatorio, el índice de semejanza estructural (SSIM) es un método para medir la semejanza y entre dos imágenes. También pude ser visto como una medida de la calidad de una de las imágenes que están siendo comparadas, considerando que la segunda imagen es de calidad perfecta.

El algoritmo permite disponer de varios argumentos de entrada. Los dos primeros se corresponden con las dos imágenes que van a ser comparadas. Opcionalmente, podemos elegir los otros parámetros: *L*, el rango dinámico de las imágenes de entrada; *window*, la ventana local dentro de la cual se realiza el cálculo de los parámetros estadísticos (media, desviación estándar y correlación); y *K*, que determina el peso de las constantes *C1*, *C2* y *C3* en la fórmula del índice SSIM. Dichas constantes se definen como:

$$C1 = (K1 \cdot L)^2 \qquad C2 = (K2 \cdot L)^2 \qquad C3 = C2/2$$

Los valores que están predeterminados por el programa para estos parámetros son: El rango dinámico de las imágenes (*L*) es 255; la ventana que se emplea es un filtro simétrico paso bajo Gaussiano de tamaño 11x11 y desviación estándar 1.5; y las constantes *K* toman como valor K1=0.01 y K2=0.03.

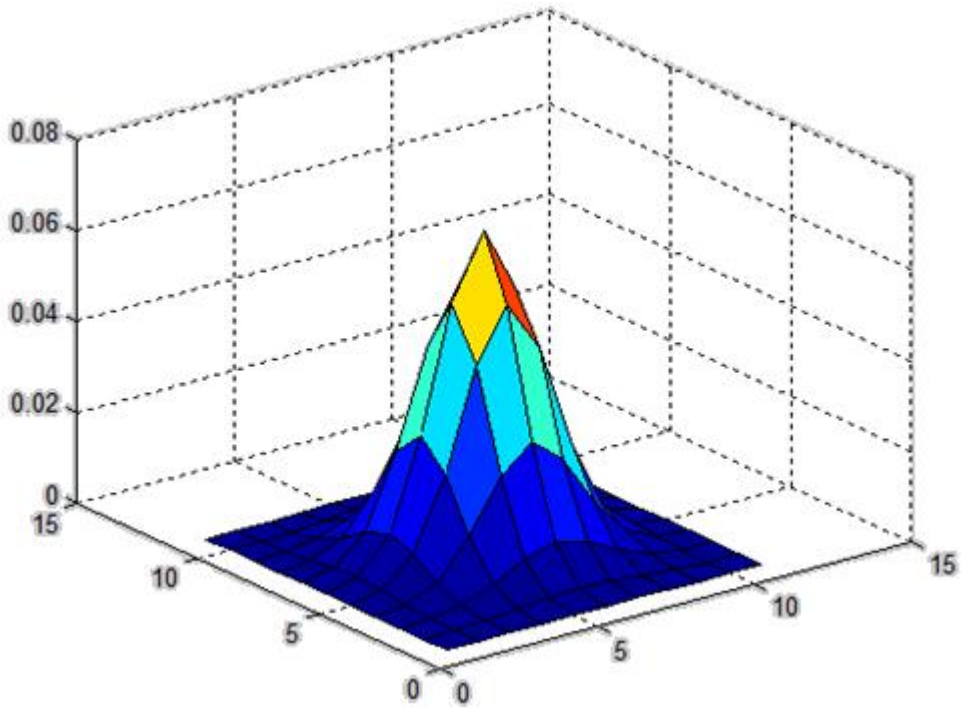


Figura 4.11. Representación de la ventana deslizante Gaussiana

La función devuelve dos valores, el índice SSIM (*mssim*) y el mapa SSIM (*ssim_map*). *mssim* es el valor medio del índice SSIM obtenido al comparar las dos imágenes. Como ya hemos comentado anteriormente, si una de las dos imágenes se considera que es de calidad perfecta, entonces *mssim* es la medida de la calidad de la otra imagen. Por otro lado, *ssim_map* es la imagen que corresponde al índice SSIM en cada píxel de la imagen. El mapa generado tiene un tamaño menor al de las imágenes de entrada, y depende del tamaño de la ventana empleada y del factor de submuestreo.

Antes de aplicar el algoritmo SSIM tal y como se expusimos en el capítulo anterior, hay que adaptar previamente la escala de la imagen; es necesario llevar a cabo un submuestreo, para el cual hay que seguir los siguientes pasos:

1) Tomar $f = \max(1, round(N/256))$, donde N es el número de píxeles de la altura o anchura de la imagen.
2) Calcular la media local en una ventana de fxf píxeles, y realizar un proceso de submuestreo en la imagen por un factor de 3.
3) Aplicar el algoritmo SSIM.

El código de Matlab para realizar este procedimiento consiste en:

```
f = max(1,round(min(M,N)/256));
if(f>1)
    lpf = ones(f,f);
    lpf = lpf/sum(lpf(:));

    img=imfilter(img,lpf,'symmetric','same');
    img = img(1:f:end,1:f:end);
end
```

Aplicado a nuestro caso tenemos imágenes de 240x320 píxeles, entonces mediante la fórmula $f = \max\left(1, round(240/256)\right) = 1$. Vemos que el resto de pasos (media y submuestreo) no van a tener ningún efecto. Pero si por ejemplo, las imágenes fueran de 512x512, tendríamos: $f = \max\left(1, round(512/256)\right) = 2$. Entonces a la imagen se aplica una ventana 2x2:

```
lpf =

    0.2500    0.2500
    0.2500    0.2500
```

Finalmente, de la imagen filtrada se toma una muestra de cada dos, es decir, se submuestrea por un factor de 2.

CAPÍTULO 5

PRUEBAS Y RESULTADOS EXPERIMENTALES

El objetivo de este capítulo consiste en presentar los diferentes resultados obtenidos al llevar a cabo los procedimientos descritos en el capítulo anterior. Las pruebas estarán orientadas a determinar si el algoritmo detecta movimiento real y si es capaz de discriminar falsas detecciones en diferentes entornos. Inicialmente se expondrán algunos casos y ejemplos específicos para ilustrar las situaciones a las que nos enfrentamos y que se producen durante las pruebas. Por último, se recogerán todos los experimentos realizados para presentarlos de manera global y evaluar el rendimiento del sistema.

Los datos que se deriven de estas pruebas se emplearán para extraer diferentes conclusiones y para determinar si el sistema elegido es válido y su comportamiento satisfactorio.

5.1. PRUEBAS

5.1.1. BASE DE DATOS

En primer lugar describimos el banco de datos que se ha empleado para validar el correcto funcionamiento de la aplicación. Para realizar las distintas pruebas se ha confeccionado una base de datos formada por vídeos grabados desde una cámara de un teléfono móvil. Los vídeos son de duración variable, llegando hasta el minuto de duración, tienen una tasa de 25 frames por segundo y una resolución de 320x240 píxeles. Los vídeos se han dividido en distintas categorías, dependiendo de la situación en la que se produzcan y las características presentes en ellos, como por ejemplo:

- Interior
- Exterior
- Poco movimiento

- Mucho movimiento
- Iluminación

5.1.2. RUIDO EN EL MAPA SSIM

El primer comentario que vamos a realizar consiste en analizar el comportamiento del sistema sin ningún tipo de movimiento. Cuando estamos grabando un escenario de interior como una habitación sin que ocurra ningún acontecimiento y la cámara está inmóvil, lo esperado sería que el algoritmo SSIM no detectara ningún tipo de movimiento, que el índice para todos los pixeles fuera 1, o lo que es lo mismo, que el mapa SSIM fuera totalmente blanco. Sin embargo, esto no ocurre así, ya que el comportamiento real se aleja levemente de la predicción anterior.

Durante la grabación de la secuencia de vídeo aunque no haya ningún movimiento de objetos o de la cámara, se introduce cierta cantidad de ruido provocado por la compresión, por cambios de formato, etc. que van a distorsionar fotogramas de la secuencia que en principio deberían ser idénticos. Estos fenómenos son captados por el algoritmo SSIM y en consecuencia pueden apreciarse en el mapa SSIM.

Figura 5.1. Ruido en el mapa SSIM sin movimiento

Si comprobamos algunos valores de los píxeles sobre las imágenes obtenidas podemos observar que los píxeles más oscuros tienen un valor alrededor de 0.9. Por otro lado, al representar el histograma de uno de los mapas SSIM anteriores vemos que los valores

de los píxeles se sitúan cerca de la unidad, por lo que su influencia sobre el rango en el que trabaja SSIM (0 – 1) es limitado. Por este motivo, podemos diferenciar claramente el ruido en una escena inmóvil de los objetos en movimiento, que, como veremos más adelante, tienen unos valores de SSIM más cercanos a 0.

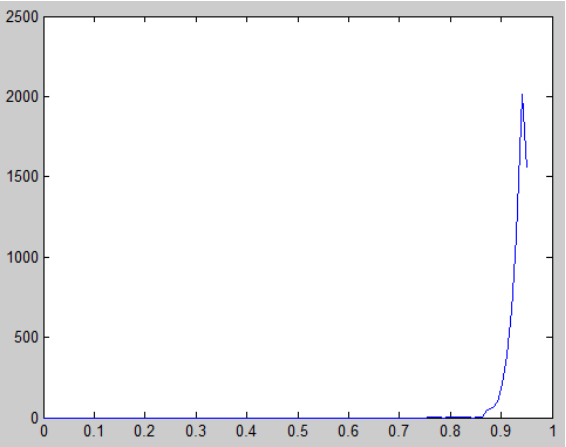


Figura 5.2. Histograma de mapa SSIM sin movimiento

5.1.3. ELECCIÓN DEL UMBRAL

Uno de los puntos más importantes de la aplicación consiste en determinar el umbral adecuado con el cual decidir si hay o no movimiento en una imagen. De la elección del umbral depende la correcta detección y el correcto funcionamiento del sistema. Si elegimos un umbral demasiado bajo estaremos descartando zonas en las que se produce movimiento, aunque sea menos intenso, lo que nos provoca falsos negativos. En cambio, si el umbral es demasiado alto estamos clasificando como movimiento de un objeto algún otro tipo de variación que provoca una distorsión en el mapa SSIM, y que genera falsos positivos o falsas detecciones.

A continuación veremos algunos ejemplos en los que nos plantearemos elegir el umbral más apropiado. De partida, vamos a asumir que el umbral va a ser inferior a 0.8 y en los histogramas siguientes no vamos a representar todo el intervalo [0 – 1] para no tener en cuenta los píxeles que corresponden al fondo inmóvil, con valor cercano a 1 y que son los más numerosos, y a alteraciones puntuales.

El ejemplo que se muestra en la figura 5.3 es un caso muy favorable, en el cual tenemos un movimiento claro y definido. En el mapa SSIM asociado, los píxeles de movimiento destacan mucho sobre el resto porque tienen un color negro muy intenso. En el histograma se puede ver que la mayoría de los valores de estos píxeles están cerca de cero. Por tanto, podemos bajar mucho el umbral sin que se pierda prácticamente información de las zonas de movimiento.

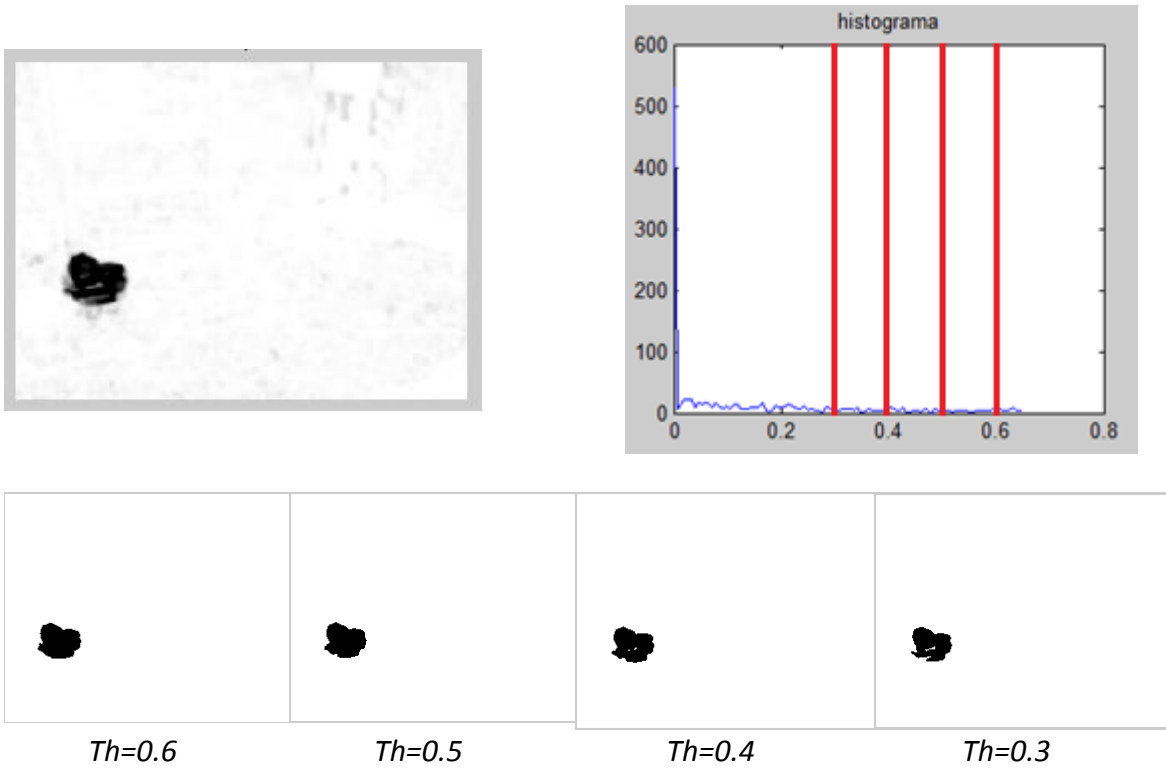


Figura 5.3. Ejemplo de elección de umbral 1

En el caso de la figura 5.4, vemos que los píxeles de la zona de movimiento se encuentran más repartidos por los distintos niveles de gris del histograma. Si variamos el umbral disminuyéndolo progresivamente, entonces estamos eliminando grupos de píxeles que son relevantes para la representación del objeto. A diferencia del ejemplo anterior, ya no podemos elegir un umbral con tanta libertad sino que estamos obligados a afinar más nuestra decisión.

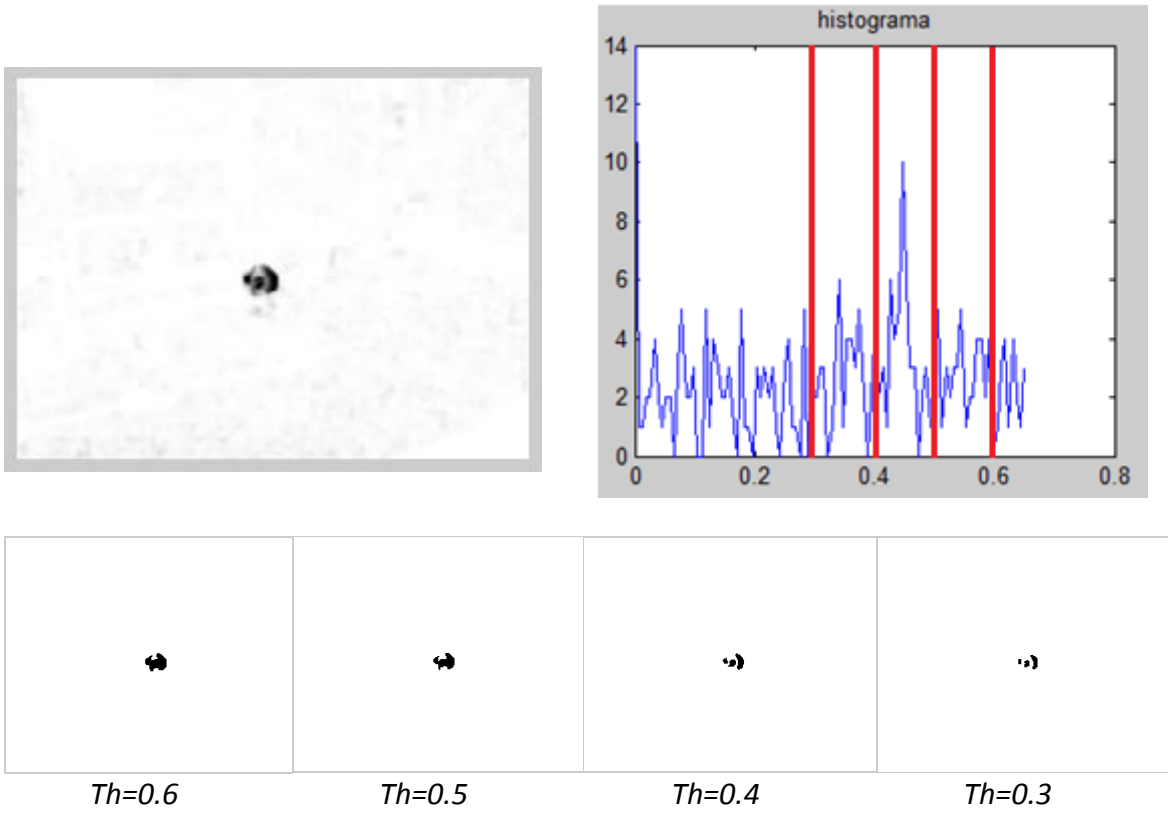


Figura 5.4. Ejemplo de elección de umbral 2

En situaciones como la que se muestra en la figura 5.5, tenemos que el desplazamiento del objeto representado es muy reducido. Como se puede ver, el mapa SSIM lo identifica de forma muy difusa y no existen píxeles que pertenezcan a los niveles de gris del histograma más próximos a cero, como ocurría en casos anteriores. Para que nuestro sistema funcione de forma eficaz, debería ser capaz de detectar este tipo de movimientos más sutiles. En el ejemplo se muestra que únicamente a partir de un umbral de 0.5 comenzamos a percibir los píxeles del objeto. En una situación como esta, el comportamiento más natural consiste en subir el umbral para permitir aumentar el número de píxeles que se analicen, con el riesgo de que cualquier comportamiento anómalo nos produzca una falsa detección.

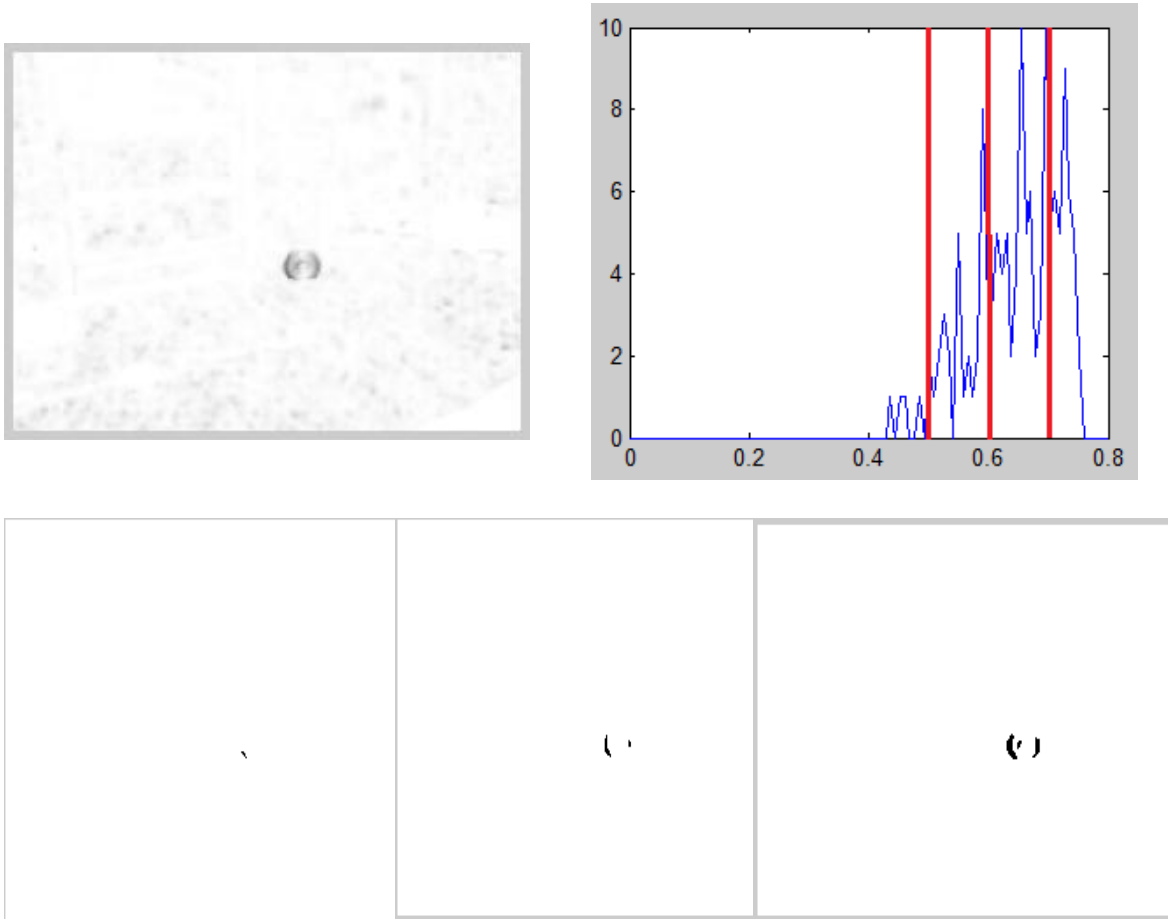


Figura 5.5. Ejemplo de elección de umbral 3

Finalmente hemos considerado que el mejor umbral para hacer nuestros experimentos es tomar un valor entre 0.5 y 0.6, ya que para estos umbrales obtenemos un número suficiente de píxeles para detectar la presencia de un objeto. En algunos casos, este valor incluso se puede superar si tenemos certeza de que no hay ninguna alteración ni movimiento de la cámara.

5.1.4. ESTUDIOS SOBRE EL CALIBRADO

Como hemos comentado en el capítulo anterior, la finalidad del proceso de calibrado consiste en descartar algunos movimientos que son constantes en la escena, pero no son relevantes de cara a su análisis. Por lo tanto es conveniente confeccionar una máscara que recoja todos los movimientos de este tipo que nos provocarían falsas detecciones.

La función de calibrado se realiza acumulando el mapa SSIM durante los primeros fotogramas de la secuencia de vídeo. Como es lógico, cuanto más tiempo dure el

proceso de calibrado del vídeo, con mayor precisión podremos determinar la zona de movimiento. Sin embargo, como no podemos adivinar de antemano la complejidad del movimiento para definir de manera acorde el número de fotogramas, únicamente podemos hacer estimaciones.

Dicho esto, podemos distinguir entre dos tipos de situaciones que podemos encontrar y en las que el calibrado puede aplicarse:

- En primer lugar tenemos movimiento confinado en una zona concreta y definida. Por ejemplo, el movimiento de las escenas de un televisor está confinado al tamaño de la pantalla y el movimiento de las aspas de un ventilador se limita al círculo que produce con su giro.
- La otra situación implica intentar caracterizar un movimiento más aleatorio del cual no podemos predecir su comportamiento o su zona de actuación, como por ejemplo el caso de una cortina o una tela mecida por el viento.

En las siguientes figuras se muestran casos particulares que reflejan las situaciones mencionadas anteriormente. Cada imagen binaria corresponde con la máscara que se genera al final del proceso de calibrado para distintos intervalos de duración. Los siguientes ejemplos se han realizado dedicando los primeros 100, 200, 300, 400 y 500 fotogramas del vídeo al calibrado. (Hay que tener en cuenta que del número de fotogramas indicados, los frames se tomaban en intervalos de 5 en 5 para agilizar el procesado).

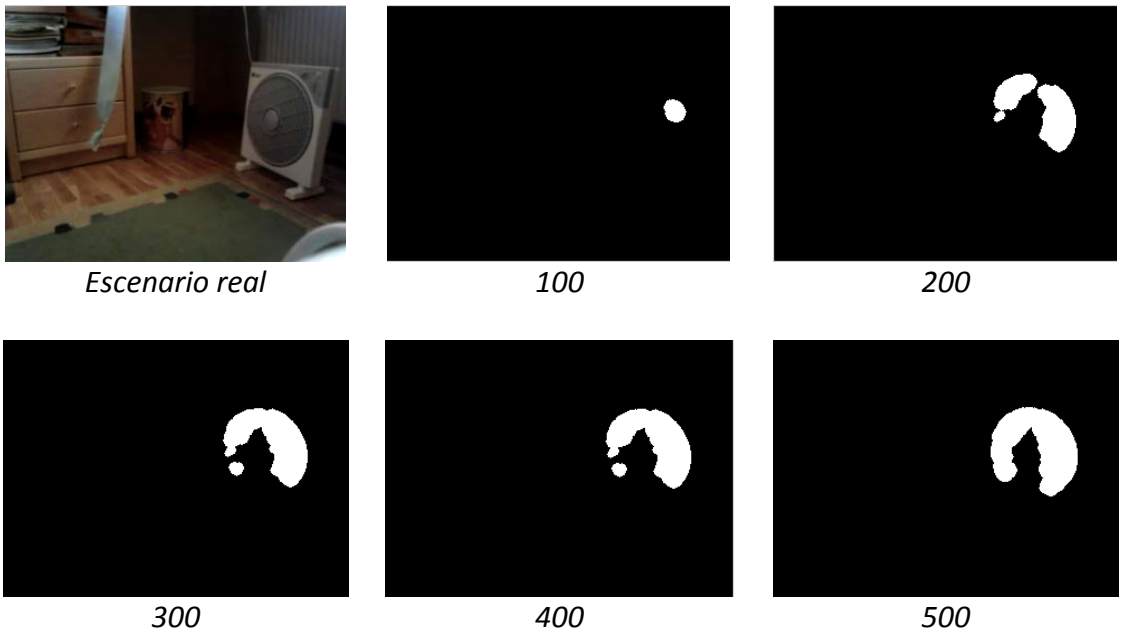


Figura 5.6. Ejemplo 1 del proceso de calibrado (clase 1)

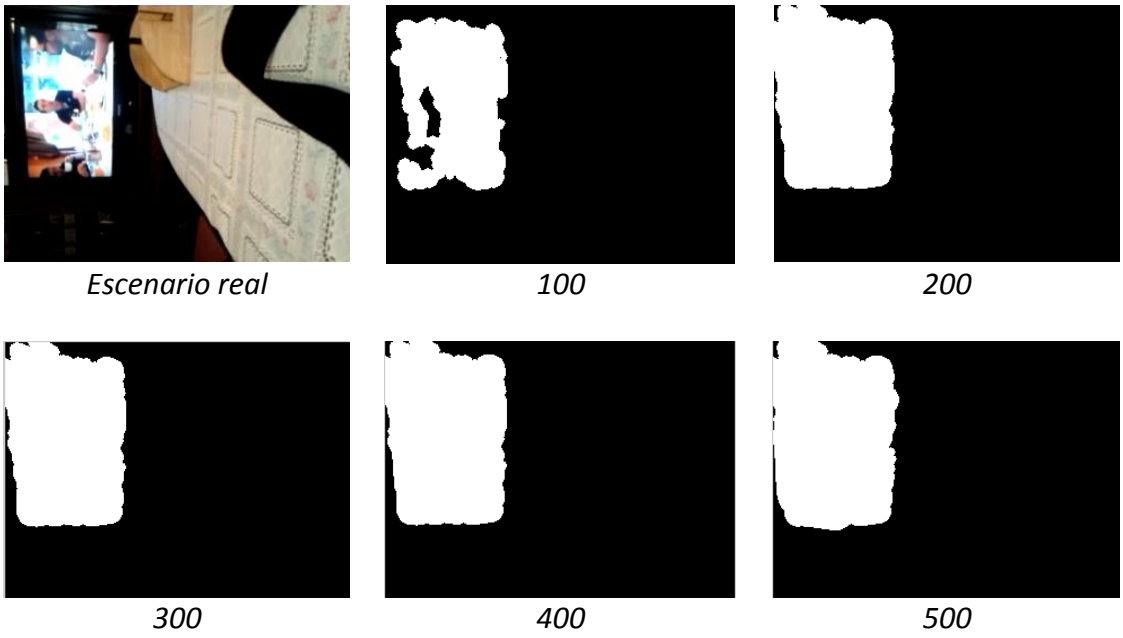


Figura 5.7. Ejemplo 2 del proceso de calibrado (clase 1)

En el ejemplo 2 se puede ver que la consecución de una máscara estable que caracterice el fondo en movimiento se consigue en menos tiempo que en el caso del primer ejemplo; lo cual ilustra el hecho de que no podemos conocer con exactitud la duración óptima de esta etapa.

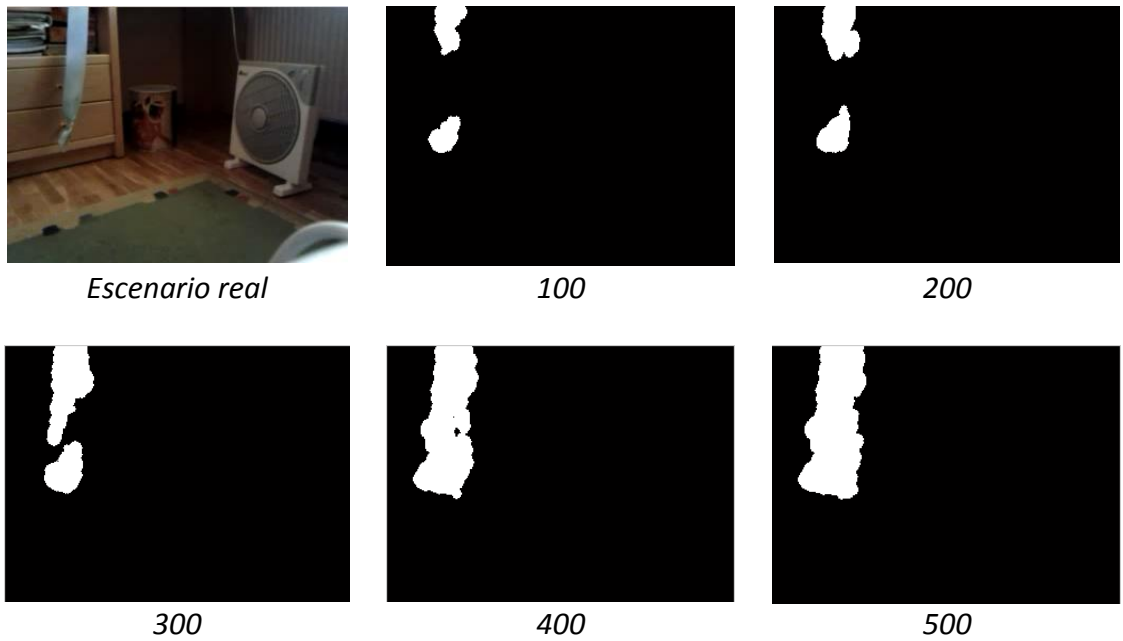


Figura 5.8. Ejemplo 1 del proceso de calibrado (clase 2)

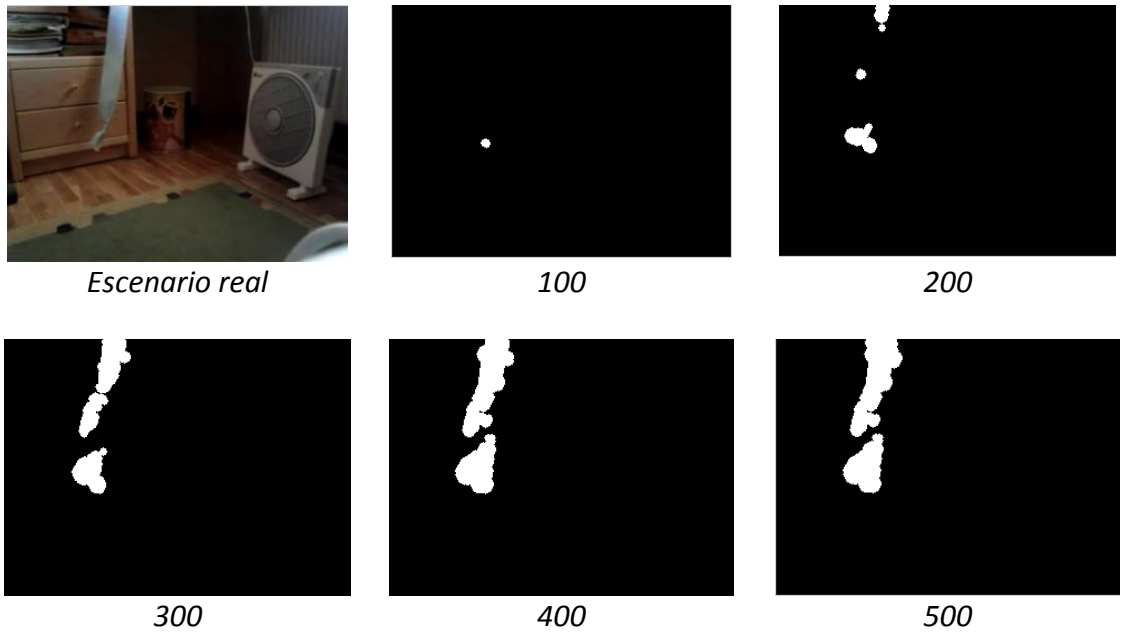


Figura 5.9. Ejemplo 2 del proceso de calibrado (clase 2)

Es necesario tener en cuenta que para realizar la máscara, tal y como se comentaba en el capítulo 4, es necesaria la suma de varias imágenes binarias que representan los mapas SSIM generados durante todo el proceso de calibrado. Una vez se tiene esta suma total, hay que determinar cuántas repeticiones de píxeles (valores de los píxeles de la imagen) se consideran necesarias para que el píxel en cuestión pase a formar parte de la máscara. En los ejemplos anteriores, se estableció que fuera necesario que los píxeles aparecieran al menos tres veces durante todo el proceso. En las siguientes imágenes puede compararse el efecto sobre la máscara de considerar una, dos o tres repeticiones, respectivamente, para la misma duración de la fase de calibrado.

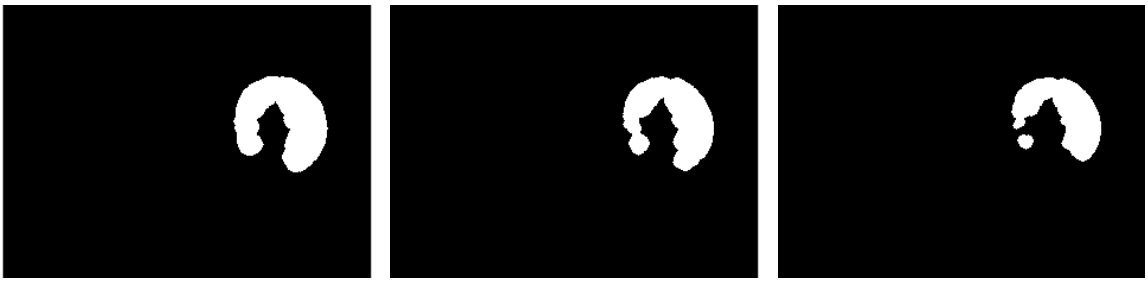
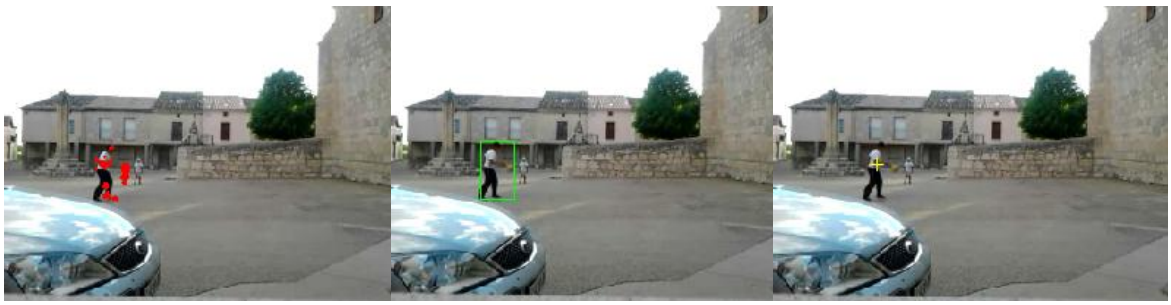


Figura 5.10. Tamaño de máscara

5.1.5. REPRESENTACIÓN DE OBJETOS

Como indicamos en el capítulo sobre las técnicas de detección de movimiento y seguimiento de objetos, los elementos presentes en una escena pueden representarse de diferente forma, mediante por ejemplo: siluetas, puntos, formas geométricas… En este trabajo vamos a optar por la representación utilizando las zonas generadas por el mapa SSIM, puntos y rectángulos.

Los píxeles que representan un movimiento, conseguidos a partir del mapa SSIM, los hemos marcado en rojo sobre el fotograma en cuestión; y aunque esta representación se puede emplear en todo tipo de situaciones, es especialmente más útil cuando el objeto o persona ocupa un elevado espacio del total del fotograma. El uso de un rectángulo que envuelve los puntos detectados por el mapa SSIM y que se adapta a la forma del objeto, y de un punto que marca su centro es más apropiado cuando el tamaño es menor.



(a) Mapa SSIM *(b) Rectángulo* *(c) Punto*

Figura 5.11. Distintas formas de representación

5.1.6. TAMAÑO DE LOS ELEMENTOS

Los elementos en movimiento que nos encontremos dependerán en buena media del tipo de vídeo que estemos tratando. En vídeos de interior consideramos que el espacio que se está analizando es menor, y los elementos más frecuentes serán personas y objetos. Dentro de una habitación o un entorno cerrado, una persona puede ocupar un porcentaje considerable de la escena. En vídeos de exterior, si por ejemplo realizamos la vigilancia de una calle, tendremos la cámara colocada en una posición desde donde tengamos una perspectiva amplia de la calle y los objetivos serán principalmente vehículos y personas. Es razonable pensar que el tamaño relativo de los objetos difiere dependiendo del escenario en el que nos encontremos. Así el tamaño de una persona dentro de una habitación será distinto su tamaño en la calle.

En las siguientes imágenes se pueden observar algunos ejemplos del porcentaje del fotograma que ocupan diferentes elementos en movimiento.

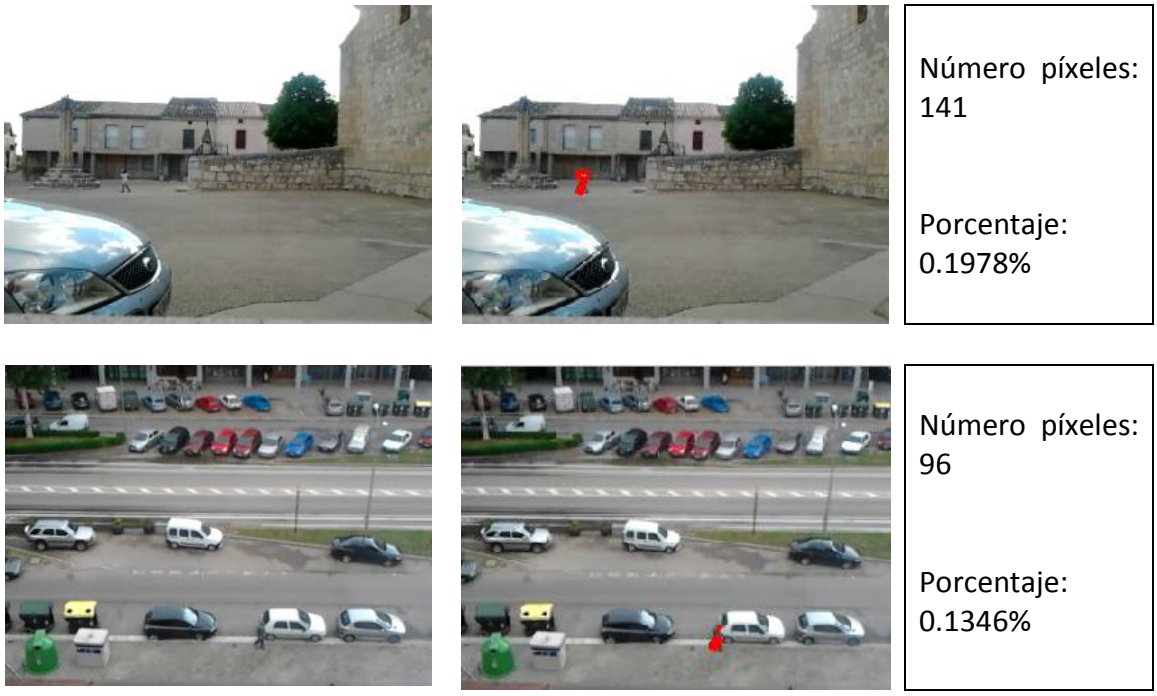


Figura 5.12. Tamaño relativo de personas en exterior

En situaciones como las de la figura anterior, en las que el tamaño de los objetos es muy pequeño con respecto a la totalidad de la imagen, cobra mayor importancia el color de la persona respecto al del fondo. Si el objeto resalta sobre el fondo será más fácil para el algoritmo detectar los cambios, pero si ambos tienen colores parecidos pueden confundirse y la detección no se realiza de forma precisa.

En la figura 5.13 tenemos la presencia de una persona en una habitación interior, en el que vemos principalmente el movimiento del brazo y parte del cuerpo. En este tipo de situaciones en las que aparece un cuerpo de mayor tamaño, en algunas ocasiones todo ello tendrá movimiento y en otras solo algunas partes se moverán, por lo que el tamaño detectado en diferentes situaciones variará.

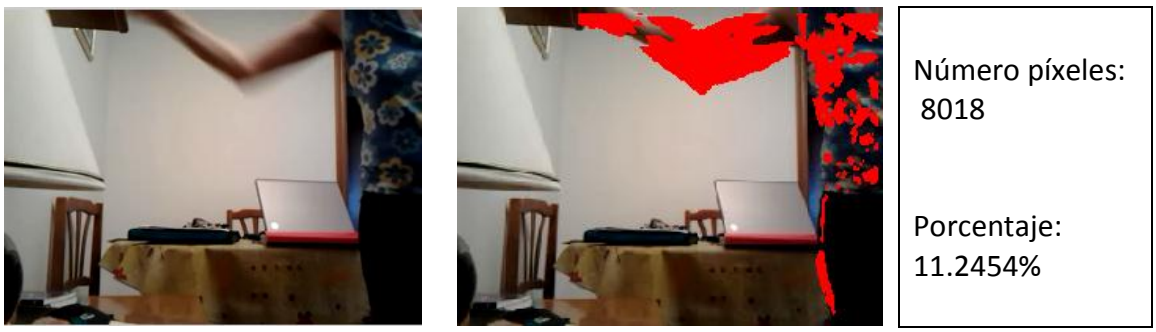


Figura 5.13. Tamaño relativo de personas en interior

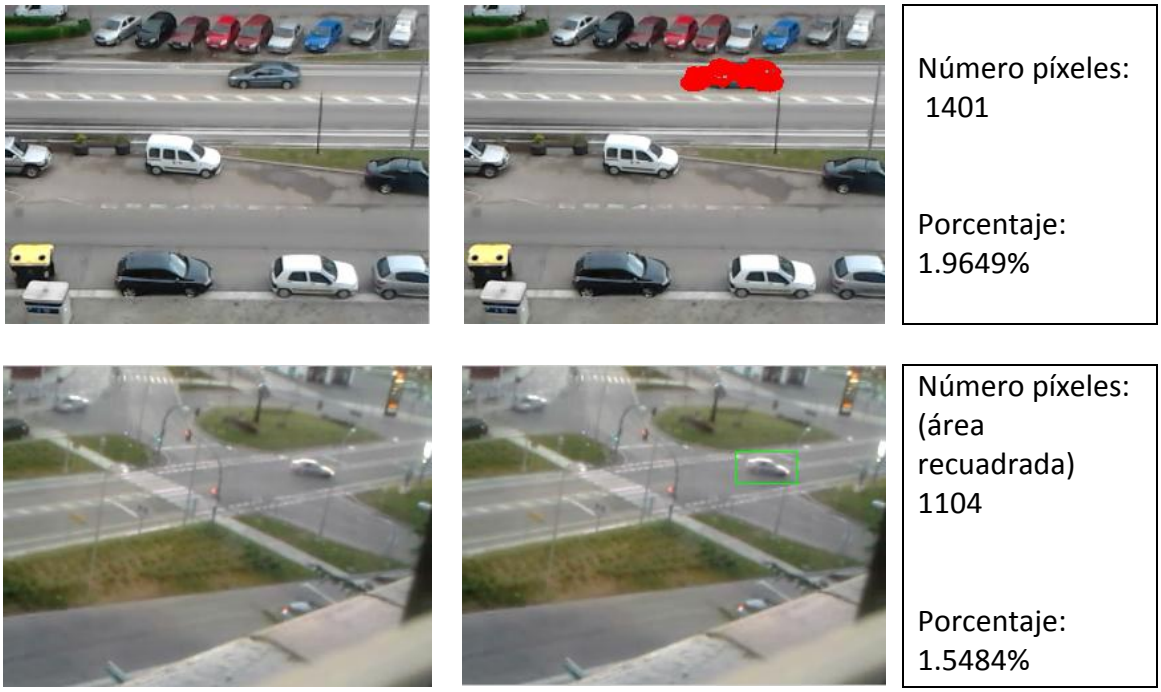


Figura 5.14. Tamaño relativo de coches

En situaciones más difíciles donde el algoritmo no detecta la totalidad de los puntos del objeto que se está moviendo, es más recomendable encerrar el objeto dentro de un rectángulo. Aplicando esta variación calcular el tamaño mediante una aproximación por exceso.

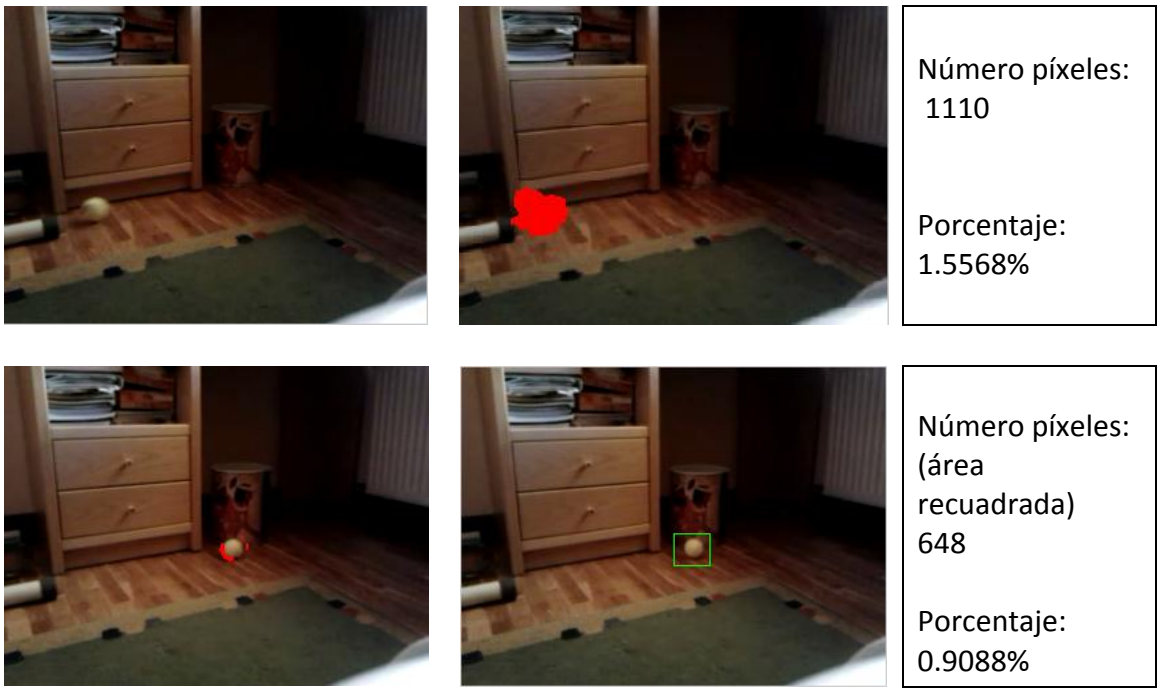


Figura 5.15. Tamaño relativo de objetos en interior (pelota de tenis)

Como conclusión a este apartado, podemos decir que conocer el tamaño de objetos puede aplicarse para distinguir entre falsas detecciones o elementos que no interesa evaluar. Si caracterizamos el tamaño de los elementos recurrentes que son importantes en un escenario concreto, la detección de un número de píxeles sustancialmente menor al objeto más pequeño puede considerarse como poco relevante y despreciarse.

5.1.7. SOLAPAMIENTO DE OBJETOS

Cuando estamos representando los distintos elementos en movimiento de una escena utilizando rectángulos, el área encuadrada se estima mediante los puntos obtenidos en el mapa SSIM que tienen cierta proximidad. En el caso de tener dos objetos cercanos entre si, llegará un punto en el cual el algoritmo no pueda distinguir entre ambos, los considerará como un único elemento y se representará utilizando un solo rectángulo que englobe a los dos.

(a) *(b)*

Figura 5.16. Superposición de objetos

En la figura 5.16.a, el algoritmo distingue con claridad entre el niño y la pelota porque están suficientemente distanciados. Sin embargo, en 5.16.b ya no es posible determinar qué puntos del mapa SSIM corresponden con el niño y cuáles con la pelota y el sistema nos los presenta como si hubiera solamente un elemento de mayor tamaño.

5.1.8. DIVISIÓN DE OBJETOS

El proceso de creación del mapa SSIM se realiza comparando dos fotogramas consecutivos de la secuencia de vídeo (para mayor agilidad, la separación entre dos fotogramas que se van a comparar es de tres). Al observar el desplazamiento de un objeto uniforme entre dos fotogramas, podemos ver en el mapa SSIM que las diferencias entre las dos imágenes y los lugares donde detecta movimiento se corresponden con los extremos del objeto, mientras que el centro del mismo se mantiene similar. Esto puede provocar que el mapa SSIM determine dos zonas de movimiento y se detecte la presencia de dos objetos pequeños y próximos en lugar de uno solo de mayor tamaño.

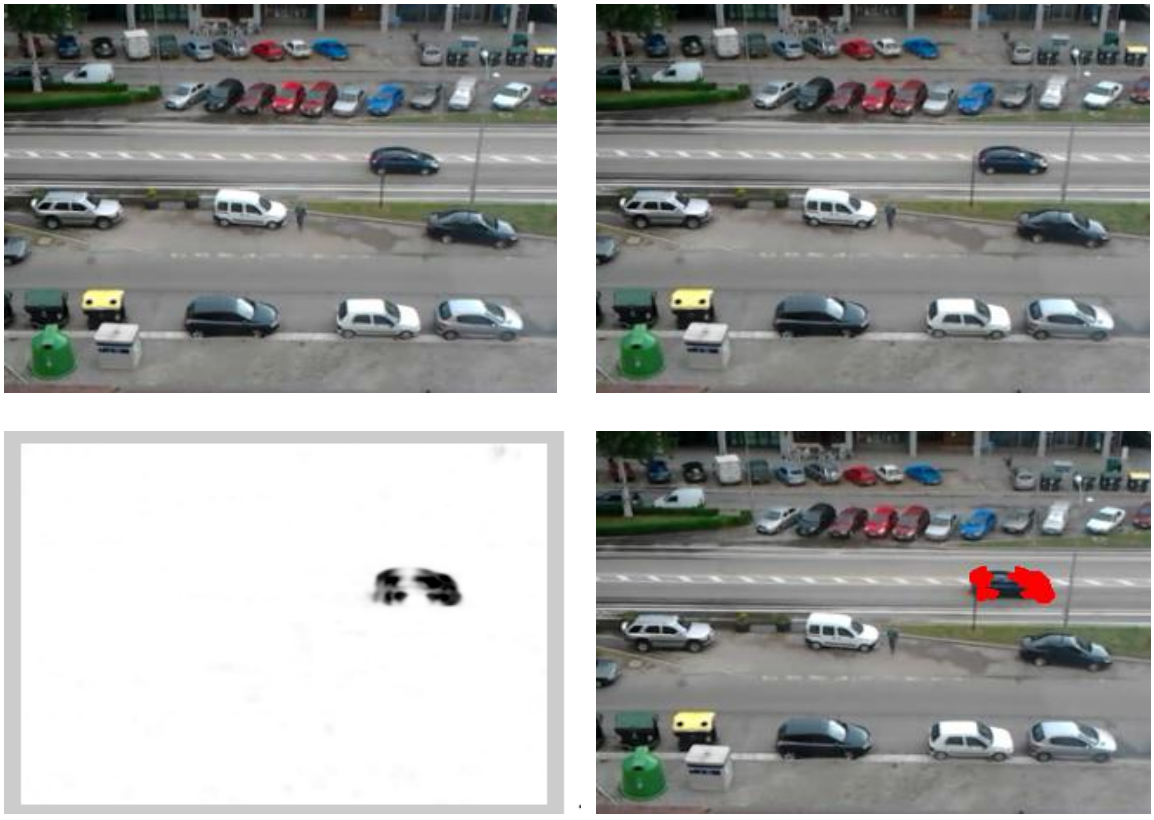


Figura 5.17. División de objetos

En las imágenes anteriores, los píxeles oscuros del mapa SSIM corresponden con los extremos delantero y trasero del coche, considerando la parte central como si no se estuviera moviendo, ya que en las dos imágenes que se comparan esta zona es similar. Este problema puede solucionarse expandiendo y/o rellenando el espacio correspondiente a los coches posteriormente a la obtención del mapa SSIM para que las dos zonas separadas se unan y se puedan considerar como una sola, como se ilustra en la siguiente imagen.

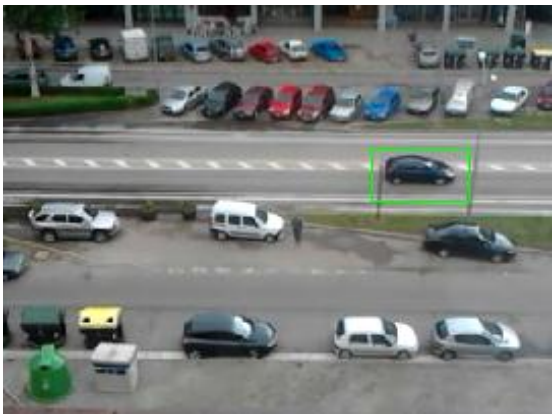


Figura 5.18. Solución a la división de objetos

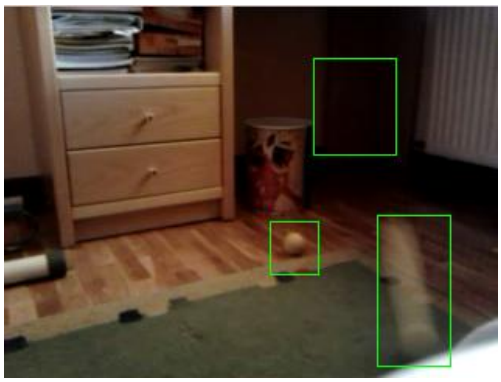
5.1.9. VELOCIDAD DE LOS OBJETOS

La velocidad a la que se desplazan los objetos tiene influencia en la percepción que tenemos del movimiento entre los sucesivos fotogramas. Hay que tener en cuenta que para confeccionar el mapa SSIM se están comparando dos fotogramas separados tres posiciones. En algunas ocasiones, si la velocidad del objeto es demasiado elevada, la diferencia entre las dos imágenes es mayor de lo esperada. En estos casos el resultado que obtenemos al realizar la detección de objetos puede alejarse de la realidad. A continuación se muestra un caso extremo.

En la figura siguiente (5.19) tenemos una situación con dos objetos móviles con distintas velocidades de desplazamiento. La pelota situada en el centro de la escena tiene una velocidad que podemos considerar normal y en (c) se produce una detección correcta. Sin embargo, la pelota a la derecha tiene una velocidad mucho mayor y en los fotogramas que se comparan, (a) y (b), se comprueba que recorre mucha distancia. Esto se traduce en (c) como la detección de dos zonas de movimiento en lugar de detectar un único objeto. También aparece la "estela" del movimiento de la pelota, que se produce al capturar la imagen por la cámara, y que provoca que el tamaño del objeto detectado sea mayor que sus dimensiones reales.

(a) Instante K

(b) Instante K+1



(c) Detección

Figura 5.19. Detección de un objeto con velocidad demasiado alta

5.1.10. MOVIMIENTO DE LA CÁMARA

Hasta ahora hemos considerado que la grabación de los vídeos se realizaba de manera estática, con la cámara totalmente inmóvil. Sin embargo, pueden darse situaciones en las que la cámara sufra pequeñas oscilaciones, ya sea por el efecto del viento si está situada en exteriores, por vibraciones del soporte al que se sujeta, etc. A continuación vamos a comprobar qué efectos tienen pequeños movimientos de la cámara sobre la detección de movimiento.

Al comparar dos imágenes ligeramente desplazadas una respecto a otra, el resultado obtenido es que el mapa SSIM lo interpreta como un movimiento de los contornos de la escena. Un ejemplo de esta situación puede verse en la siguiente figura, donde la imagen de la izquierda ha sufrido una oscilación más suave y la imagen de la derecha un movimiento más brusco.

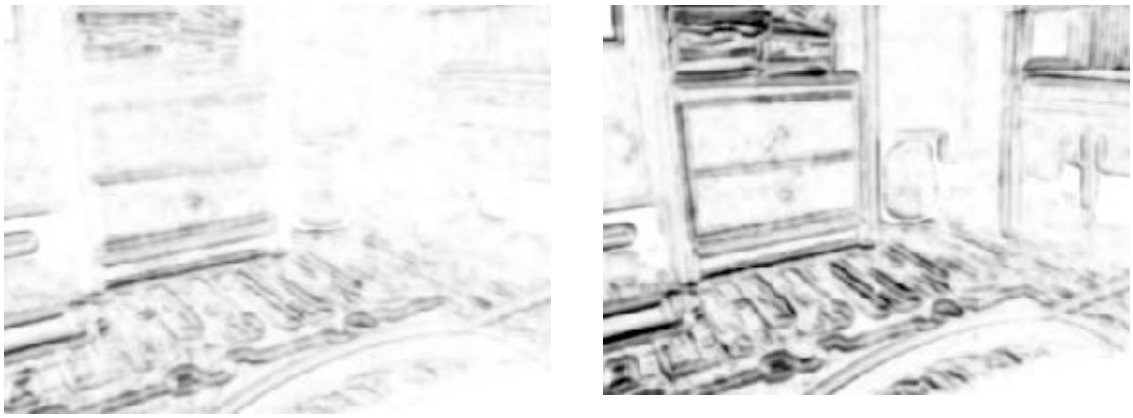


Figura 5.20. Mapa SSIM con movimiento de cámara

El problema que esto nos genera en la detección de movimiento es la posibilidad de que estas alteraciones enmascaren sobre el mapa SSIM el movimiento real de objetos. Dependiendo de la intensidad de la oscilación de la cámara podremos realizar una correcta detección o no.

Figura 5.21. Detección satisfactoria con movimiento de cámara

En este caso, puede separarse sin problemas los píxeles del objeto en movimiento de los provocados por una ligera oscilación, utilizando un umbral de 0.5. En otras ocasiones no va a ser posible elegir un umbral en el intervalo 0.5 – 0.6, el cual se eligió como el más adecuado en apartados anteriores, para evitar las distorsiones, lo que nos provocará falsas detecciones.

En la figura 5.22 la cámara sufre un movimiento de mayor envergadura. Con el umbral situado en 0.5 vemos que el movimiento del objeto apenas puede distinguirse de las perturbaciones producidas por las oscilaciones. En este caso no es posible realizar una buena detección y el sistema funcionaría de manera errónea.

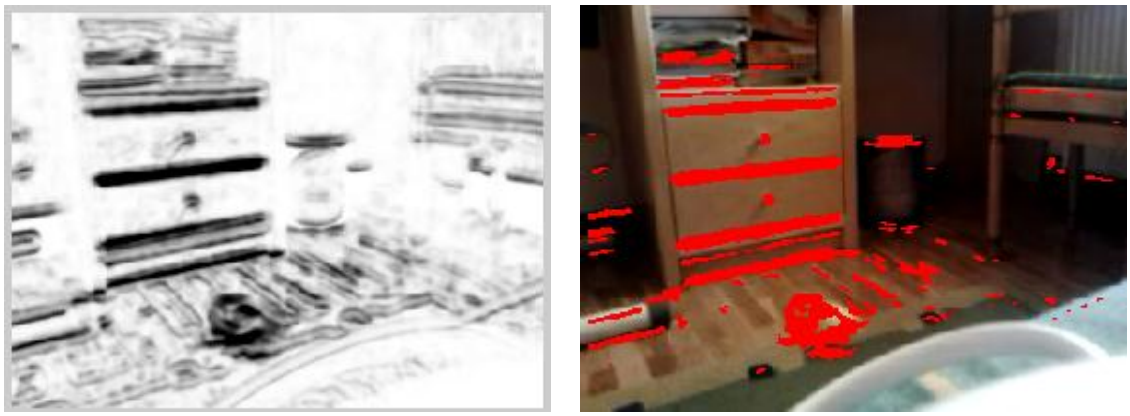


Figura 5.22. Detección fallida con movimiento de cámara

5.1.11. ILUMINACIÓN

Un aspecto importante en detección de movimiento es el efecto de cambios de iluminación en la escena, ya que modificaciones sustanciales en este aspecto pueden provocar falsas detecciones. Existen técnicas en el procesado de imagen que permiten eliminar su efecto, como el filtrado homomórfico, que describimos a continuación:

- **Filtrado homomórfico [3]**
  Para explicar el filtro homomórfico hay que partir de una determinada caracterización de las imágenes. Podemos considerar una imagen como una función $f(x,y)$ que se caracteriza por dos componentes: (1) la cantidad de iluminación de fuente incidente en la escena y (2) la cantidad de iluminación reflejada por los objetos. Entonces, éstas se llaman las componentes de iluminación y reflectancia de la imagen y se representan como $i(x,y)$ y $r(x,y)$, respectivamente. Las dos componentes se combinan como un producto para formar la imagen completa:

$$f(x,y) = i(x,y) \cdot r(x,y)$$

  El modelo de iluminación-reflectancia puede utilizarse para desarrollar un método en el dominio de la frecuencia para mejorar la apariencia de una imagen mediante la compresión del rango de grises y la mejora del contraste.
  La ecuación anterior no puede usarse directamente para operar de forma independiente sobre las componentes frecuenciales de la iluminación y reflectancia porque la Transformada de Fourier de un producto no puede separarse. Esto es:

$$\Im\{f(x,y)\} \neq \Im\{i(x,y)\} \cdot \Im\{r(x,y)\}$$

Sin embargo, si tomamos logaritmos:

$$z(x,y) = \ln f(x,y) = \ln i(x,y) + \ln r(x,y)$$

Entonces:

$$\Im\{z(x,y)\} = \Im\{\ln f(x,y)\} = \Im\{\ln i(x,y)\} + \Im\{\ln r(x,y)\}$$

O:

$$Z(u,v) = F_i(u,v) + F_r(u,v)$$

Donde $F_i(u,v)$ y $F_r(u,v)$ son las Transformadas de Fourier de $\ln i(x,y)$ y $\ln r(x,y)$, respectivamente.
Si procesamos $Z(u,v)$ por medio de un filtro $H(u,v)$, entonces:

$$S(u,v) = H(u,v)Z(u,v) = H(u,v)F_i(u,v) + H(u,v)F_r(u,v)$$

Donde $S(u,v)$ es la Transformada de Fourier del resultado. En el dominio espacial:

$$s(x,y) = \Im^{-1}\{S(u,v)\} = \Im^{-1}\{H(u,v)F_i(u,v)\} + \Im^{-1}\{H(u,v)F_r(u,v)\}$$

Tomando:

$$i'(x,y) = \Im^{-1}\{H(u,v)F_i(u,v)\}$$
$$r'(x,y) = \Im^{-1}\{H(u,v)F_r(u,v)\}$$

La ecuación anterior puede expresarse de la siguiente forma:

$$s(x,y) = i'(x,y) + r'(x,y)$$

Finalmente, como $z(x,y)$ fue formado tomando el logaritmo de la imagen original, la operación inversa (función exponencial) proporciona la imagen mejorada, llamada $g(x,y)$, esto es:

$$g(x,y) = e^{s(x,y)} = e^{i'(x,y)} \cdot e^{r'(x,y)} = i_0(x,y)r_0(x,y)$$

Donde i_0 y r_0 son las componentes de iluminación y reflectancia de la imagen de salida.

El método se basa en un caso especial de un tipo de sistemas conocidos como *sistemas homomorfos*. En esta aplicación en concreto, la clave consiste en la separación de las componentes de iluminación y reflectancia alcanzados en:

$$Z(u,v) = F_i(u,v) + F_r(u,v)$$

Entonces el filtro homomórfico $H(u,v)$, puede operar sobre ambas componentes de forma independientemente:

$$S(u,v) = H(u,v)Z(u,v) = H(u,v)F_i(u,v) + H(u,v)F_r(u,v)$$

La componente de iluminación de una imagen generalmente se caracteriza por pequeñas variaciones espaciales, mientras que la componente de reflectancia tiende a variar bruscamente, especialmente en uniones entre objetos distintos. Estas características llevan a asociar las bajas frecuencias de la Transformada de Fourier del logaritmo de una imagen con la iluminación y las altas frecuencias con la reflectancia. Aunque éstas sean unas aproximaciones algo toscas, se pueden utilizar en la mejora de imagen.

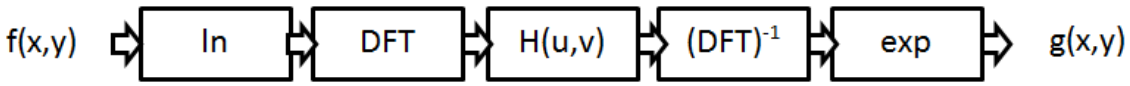


Figura 5.23. Esquema del filtrado homomórfico

Sin embargo, antes de decidir si es necesario emplear algún tipo de corrección, vamos a evaluar el comportamiento del algoritmo SSIM ante cambios de iluminación.

Para estudiar el efecto de la iluminación, se consideran escenarios de interior donde se hace variar la intensidad de la luz, oscureciendo e iluminando la estancia. Según las pruebas que hemos realizado parece que los cambios progresivos en la iluminación de la escena no generan grandes alteraciones en el índice SSIM, salvo en puntuales circunstancias; por lo que consideramos que de forma inicial no es necesario incluir ningún tipo de corrección sobre la iluminación. Sin embargo, en el caso de algún tratamiento futuro podría considerarse añadir alguna mejora al respecto.

También se ha probado la detección de movimiento en diferentes condiciones de iluminación dentro una habitación. Para evaluar la respuesta del programa se han utilizado seis vídeos en los que se lanzaban once pelotas y el resultado obtenido ha sido:

- En los vídeos que tenían dos regiones de iluminación, una con más luz y otra de más sombra, todas las pelotas se han detectado en la zona de luz pero no ha sido posible en la de oscuridad.
- Los otros vídeos tenían una distribución de luz más uniforme, aunque también oscura y se ha detectado movimiento en cinco de seis casos de forma más o menos regular durante el movimiento de la pelota, especialmente al principio cuando se desplazaban a mayor velocidad y el movimiento entre fotogramas era más apreciable.

5.1.12. CONDICIONES NOCTURNAS

Como una prueba extraordinaria, se ha estudiado el comportamiento de la aplicación cuando se aplica a la detección de vehículos en exteriores por la noche. En este tipo de entornos las características más frecuentes son la presencia de zonas oscuras y otras iluminadas por los focos de los coches o por farolas de la calle. Percibimos que todos los coches pueden ser detectados, en parte gracias a los focos cuando no hay otro tipo de iluminación. Sin embargo, estas luces móviles también provocan alteraciones en la escena, ya sea por el propio haz de luz o porque aparezcan reflejos y brillos puntuales en elementos estacionados en la calle, lo que aumenta el número de falsas detecciones. Esto es especialmente notorio si nos guiamos por la comparación del fotograma actual con el primer fotograma donde se detectó movimiento. También es posible que no se produzca la detección del volumen total del vehículo, sino que sólo aparezca cierta zona o que esté dividido en varios trozos.

En las siguientes imágenes puede comprobarse que el algoritmo detecta la presencia correctamente de dos vehículos, un coche y un autobús, pero comete algunos fallos: Aparecen reflejos de las luces en los coches estacionados, los vehículos se muestran divididos, etc.

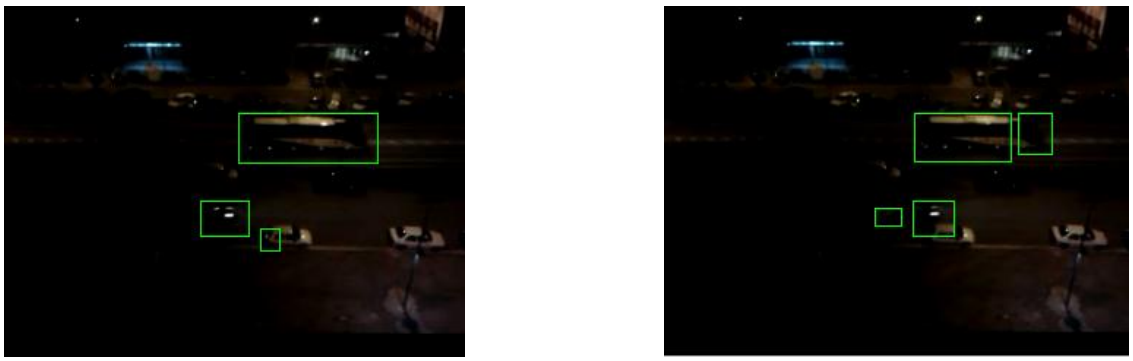


Figura 5.24. Errores de detección en condiciones nocturnas

Una situación parecida la podemos encontrar en habitaciones con poca iluminación. Si en una habitación con poca luz tenemos, por ejemplo, un televisor; los cambios de escena o de secuencia en la pantalla pueden provocar variaciones fuertes e instantáneas de la iluminación. Estas situaciones generan serias alteraciones en el mapa SSIM.

5.2. RESULTADOS

En el siguiente apartado vamos a presentar de manera global los resultados obtenidos al realizar los experimentos que se han comentado anteriormente. Los resultados se agruparán según las características que presenten los vídeos y los comportamientos que se pretendan evaluar.

Previamente, a modo de introducción, vamos a presentar distintos modelos habituales en la literatura que permiten analizar la eficacia de un sistema de clasificación partiendo de un conjunto de datos y basándose en su rendimiento. En concreto, vamos a ver los distintos estados que resultan al realizar la comparación utilizando una tabla y cómo, partiendo de estos datos, se generan gráficas que permiten dar una visión global del comportamiento del sistema.

5.2.1. ANÁLISIS POR CLASIFICADOR BINARIO

Con el fin de establecer una comparación entre un conjunto de datos de prueba podemos crear un *modelo de comparación*, el cual consiste en una función que divida y clasifique las distintas situaciones que se obtienen después de llevar a cabo unos experimentos entre dos clases bien diferenciadas. Ejemplos de funcionamiento se encuentran en [23] y [24].

En nuestro caso podemos definir dos clases diferentes: vídeos con movimiento y vídeos sin movimiento. De esta manera el clasificador se convierte en un sistema con dos entradas y dos salidas, el cual se representa mediante la llamada matriz de confusión o tabla de contingencia.

Según la tabla tenemos dos posibles valores en la realidad: Hay movimiento (*p*) o no hay movimiento (*n*). El sistema hace una predicción de la cual se obtienen dos circunstancias: se detecta movimiento (*p´*) o no se detecta movimiento (*n´*). En la figura 5.25 puede verse la representación gráfica de una tabla de contingencia.

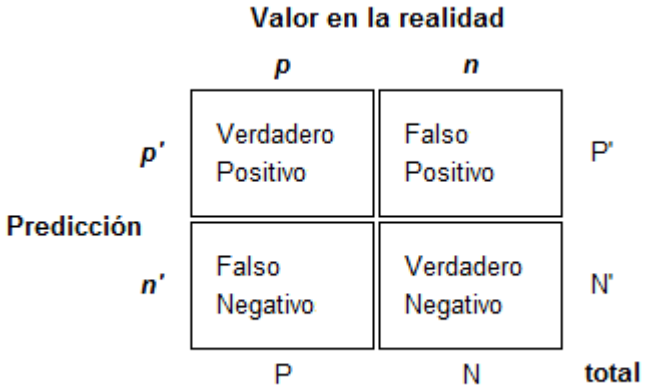


Figura 5.25. Tabla de contingencia

De esta tabla se derivan cuatro situaciones diferentes:

- Verdadero Positivo (VP): Si existe movimiento y éste es detectado.
- Falso Positivo (FP): Si no hay ningún movimiento pero el algoritmo indica presencia.
- Falso Negativo (FN): Si hay un objeto en movimiento pero no se detecta.
- Verdadero Negativo (VN): Si no hay movimiento y no se detecta nada.

A partir de estos cuatro términos se pueden definir otros como:

- Razón de Verdaderos Positivos (VPR) o Sensibilidad: $VPR = \frac{VP}{VP+FN}$
- Razón de Falsos Negativos (FPR) o 1-Especificidad: $FPR = \frac{FP}{VN+FP}$

5.2.2. ANÁLISIS POR MÉTODO GRÁFICO – GRÁFICAS ROC

Un gráfico ROC (Receiver Operating Characteristics) es una técnica para visualizar, organizar y seleccionar clasificadores basándose en su rendimiento, [23] y [24]. Este tipo de análisis se ha usado durante mucho tiempo en teoría de detección de señales para describir la relación entre las tasas de acierto y de falsa alarma de un clasificador; de hecho, fue originalmente empleado durante la Segunda Guerra Mundial para el estudio de señales de radar. El uso de los gráficos ROC se ha extendido a la visualización y análisis del comportamiento de sistemas de diagnóstico, y desde finales de los años 60 comenzó su utilización en el campo de la medicina y de la radiología. En los últimos años el análisis ROC se incrementado para aplicaciones de evaluación y comparación de algoritmos, en el campo del aprendizaje automático (*Machine learning*) y la minería de datos (*Data minering*).

- **Espacio ROC**

Los gráficos ROC son representaciones bidimensionales en las cuales la tasa de verdaderos positivos se representa en el eje Y (sensibilidad) y la tasa de falsos positivos en eje X (1-especificidad). Cada punto de la curva se corresponde con un par específico de sensibilidad y especificidad, y la curva completa proporciona la visión general del comportamiento.

Algunos puntos del espacio ROC tienen especial importancia. El punto inferior izquierdo (0,0) representa el hecho de no emitir nunca una clasificación positiva. La situación contraria, emitir incondicionalmente clasificaciones positivas, se representa por el punto superior derecho (1,1). El punto (0,1) es la clasificación perfecta.

Un punto en el espacio ROC es mejor que otro si se encuentra más hacia el "noroeste" que el primero. Los puntos que se encuentran hacia la izquierda del gráfico ROC, cercanos al eje X, realizan clasificaciones únicamente con grandes pruebas, entonces generan pocas falsas detecciones pero también un baja tasa de verdaderos positivos. Por otro lado, los puntos en el lado superior derecho de una curva ROC hacen clasificaciones positivas con pocas evidencias así que clasifican casi todos los positivos de manera correcta, pero con un elevado número de falsos positivos.

La recta diagonal x=y representa una clasificación aleatoria. Cualquier punto situado por debajo de la diagonal tiene una actuación peor que la adivinación aleatoria.

- **Curva ROC**

Para crear una curva ROC, se considera un sistema de decisión formado como dos distribuciones normales, una para escenas con movimiento y otro sin movimiento. El umbral se sitúa en posiciones diferentes para dividir las distribuciones; si se obtiene un valor por encima del umbral se produce una clasificación positiva y si es por debajo negativa. La sensibilidad y la especificidad se calculan para cada posición y los puntos resultantes se representan en una curva ROC.

En las imágenes de la figura siguiente 5.26, tomadas de [23], se ilustran tres ejemplos de distribuciones con distintos grados de solapamiento. En cada una de ellas se toman diez valores de umbral diferentes. Los puntos calculados se representan simultáneamente en la gráfica ROC. Se puede comprobar que el caso donde las distribuciones tienen menor solapamiento, o están más separadas, es más sencillo realizar correctas clasificaciones ya que la curva ROC asociada a esta situación se aproxima más al punto (0,1), que como hemos dicho anteriormente se corresponde con la clasificación perfecta.

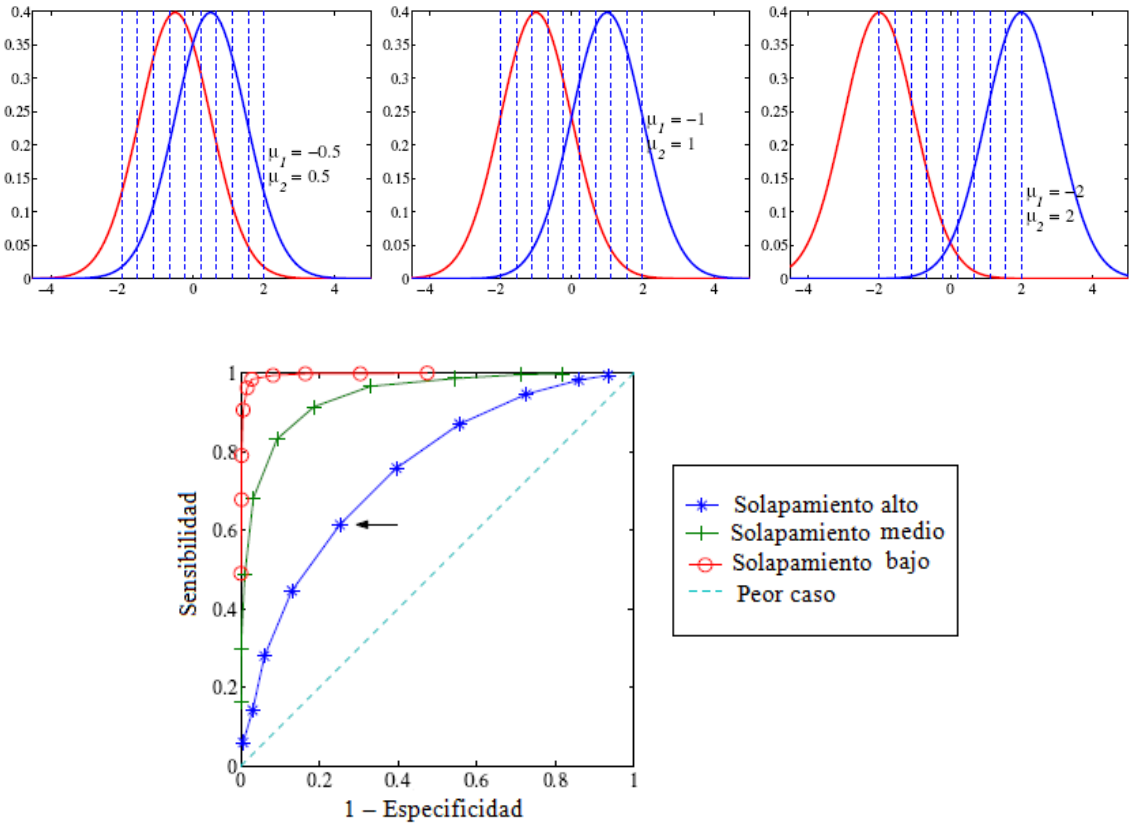


Figura 5.26. Ejemplo de construcción de curvas ROC

5.2.3. RESULTADOS

Previamente se han relazado pruebas del comportamiento del sistema sin que se produzcan movimientos, únicamente evaluando si con un umbral de 0.5 el sistema genera positivos o no.

Se ha realizado una prueba en la cual la cámara se mueve de manera circular y cuando sufre pequeñas vibraciones. Se pretende comprobar cómo afecta este movimiento a la respuesta del sistema. Se estudia el número de fotogramas que presentan alteraciones y los que se presentan inmóviles:

Vídeo	Nº de fotogramas	Bien	Mal
1	83	26 (31.32%)	57 (68.67%)
2	78	20 (25.64%)	58 (74.35%)
Total	161	46 (28.57%)	115 (71.43%)

Tabla 5.1. Efecto de movimientos de cámara circulares

Vídeo	Nº de fotogramas	Bien	Mal
1	38	35 (92.10%)	3 (7.89%)
2	54	35 (64.81%)	19 (35.18%)
Total	92	70 (76.08%)	22 (23.91%)

Tabla 5.2. Efecto de vibraciones de cámara

Para comprobar el efecto de los cambios de iluminación se han realizado varios experimentos en una habitación cambiando progresiva y suavemente la iluminación de la sala mediante un interruptor regulable.

Vídeo	Nº de fotogramas	Bien	Mal
1	82	81 (98.78%)	1 (1.39%)
2	70	69 (98.57%)	1 (1.43%)
3	64	61 (95.31%)	3 (4.68%)
Total	216	211 (97.68%)	5 (2.21%)

Tabla 5.3. Efecto de cambios de intensidad en la iluminación de una habitación

En las siguientes tablas vamos a caracterizar cada vídeo de movimiento en función de los parámetros que hemos mencionado: verdadero positivo (VP), falso positivo (FP), verdadero negativo (VN) y falso negativo (FN). Después, a partir de estos valores vamos a calcular el porcentaje de tiempo del vídeo en el que el sistema se comporta de manera correcta, es decir, detecta un movimiento cuando lo hay y no lo detecta cuando no lo hay, y el porcentaje de detecciones en los fotogramas que presentan un movimiento frente al porcentaje de falsos negativos. También se calcula la relación entre los positivos verdaderos y el número total de fotogramas en los que ocurre algún movimiento, que se obtiene como:

Precisión/Sensibilidad: $\frac{VP}{VP+FN}$

Por último se incluyen comentarios y los errores más frecuentes y característicos que se producen en cada vídeo.

Tipo de vídeo	Nº frames	VP	FP	VN	FN	Acierto	Error	Sensibilidad
1. Interior – bola Poca luz	58	0.3275	0.0344	0.6034	0.0344	93.09%	6.88%	**90.47%**
2. Interior – bola Poca luz	74	0.3978	0	0.4054	0.2027	79.72%	20.27%	**65.90%**
Total	132	0.3636	0.0151	0.4924	0.128	85.6%	14.4%	**73.39%**
	Errores	- La oscuridad aumenta del primero al segundo. - Umbral 0.5 se considera que se ha detectado un movimiento cuando aparece reflejado en la comparación actual, en la relativa o en ambas.						
1-a. Interior bola	124	0.4596	0	0.4838	0.056	94.34%	5.65%	**89.13%**
1-b. Interior bola	124	0.4838	0	0.4838	0.0323	96.76%	3.23%	**93.74%**
2. Interior bola	97	0.5361	0	0.4123	0.0515	94.83%	5.15%	**91.23%**
Total [1-b ; 2]	221	0.5067	0	0.4524	0.0408	95.91%	4.08%	**92.25%**
	Errores	- El primer caso se refiere a la detección actual y el segundo a la doble detección. - Cuando la pelota está cerca de detenerse, el algoritmo con el umbral 0.5 no es capaz de distinguir desplazamientos tan débiles. - En ocasiones, si el color o la textura del objeto es muy parecida a la del fondo en el que se encuentra en cierto momento, el algoritmo tiene dificultad.						
1. Interior bola vibración	82	0.5000	0.0244	0.2683	0.2073	76.83%	23.17%	**70.69%**
2. Interior bola vibración	61	0.2786	0.3114	0.3278	0.0819	60.64%	39.33%	**77.27%**
Total	143	0.4056	0.1468	0.2937	0.1538	69.93%	30.06%	**72.50%**
	Errores	- En el segundo caso la vibración es más fuerte que la primera, por lo que aumentan las falsas detecciones. - Sin embargo, en el primero los objetos se mueven más despacio, en ocasiones no se detectan y aparecen más falsos negativos, por lo que la sensibilidad disminuye.						

Tabla 5.4. Vídeos en interiores I

Tipo de vídeo	Nº frames	VP	FP	VN	FN	Acierto	Error	Sensibilidad
1. Interior-personas	26	0.5	0.2307	0.2307	0.0384	73.07%	26.92%	**92.85%**
2. Interior-personas	39	0.1795	0.1025	0.7179	0	89.74%	10.25%	**100%**
Total	65	0.3076	0.1538	0.5231	0.0153	83.08%	16.91%	**95.23%**
	Errores	- El porcentaje de detección (sensibilidad) es muy alto, y sólo falla en las entradas o salidas de personas de la escena cuando una pequeña parte de ellas puede apreciarse. - Los falsos positivos que se producen por las sombras son los más frecuentes dentro del porcentaje de fallos.						
1. Interior calibrado &pelota	195	0.0205	0.1743	0.8000	0.0051	82.05%	17.94%	**80%**
2 Interior calibrado	230	0	0.2174	0.7826	0	78.26%	21.74%	**-**
Total	425	0.0094	0.1976	0.7905	0.0023	80.00%	20.00%	**80%**
	Errores	- El escenario se corresponde con una habitación con baja iluminación y un televisor donde se aplica el calibrado - Los errores se deben a falsos positivos provocados por cambios bruscos de iluminación. - La segunda comparación no es útil porque se añaden continuamente errores.						

Tabla 5.5. Vídeos en interiores II

Tipo de vídeo	Nº frames	VP	FP	VN	FN	Acierto	Error	Sensibilidad
1(a). Calle– Coche- Día	137	0.7737	0.073	0	0.2189	77.37%	22.62%	**77.94%**
1(b).Calle– Coche- Día	137	0.8102	0.0073	0	0.1825	81.02%	18.98%	**81.61%**
2. Calle – Coche- Día	141	0.7943	0.071	0.1134	0.085	90.77%	9.23%	**90.33%**
Total	278	0.8021	0.0072	0.0575	0.1331	85.95%	14.04%	**85.77%**
	Errores	- El primer caso se refiere a la detección actual y el segundo a la doble detección. Si se emplea la doble detección, considerando el mapa SSIM actual y el relativo, el porcentaje de acierto aumenta, ya que es capaz de seguir objetos que dejan de detectarse un fotograma.						
1(a). Calle – Coche – Tarde	120	0.6926	0	0	0.3083	69.16%	30.83%	**69.16%**
1(b). Calle – Coche – Tarde	120	0.7608	0.1014	0	0.1376	76.08%	23.90%	**84.67%**
2 Calle – Coche – Tarde	113	0.7079	0.1151	0	0.1769	70.79%	29.20%	**80%**
	Errores	- Este escenario tiene peor visibilidad y está tomado desde más lejos, lo que puede explicar las menores prestaciones que en el caso anterior. - De nuevo se comprueba que la segunda comprobación, 1(b), aumenta la efectividad del sistema, aunque también se incremente el número de falsos positivos. - Los errores se deben a la división de los objetivos en varias partes y a no detecciones momentáneas.						
Calle – Coche Noche	142	0.5986	0.2887	0.1126	0	71.12%	28.87%	**100%**
	Errores	- Al estar parcialmente iluminado los objetos vehículos más grandes están divididos en varias partes, por lo que parece que hay mayor número de objetos pequeños, tenemos falsos positivos. - La iluminación de faros causa brillos y reflejos puntuales que generan falsas detecciones (FP). - Por el contrario, siempre detecta la presencia de un vehículo.						

Tabla 5.6. Vídeos en exteriores I

Tipo de vídeo	Nº frames	VP	FP	VN	FN	Acierto	Error	Sensibilidad
1(a). Calle-Persona	118	0.6949	0	0.1186	0.1864	81.35%	18.64%	**78.85%**
1(b). Calle-Persona	118	0.3474	0	0.1335	0.5169	48.09%	51.69%	**40.19%**
	Errores	- Cámara a gran altura y objetivos pequeños. - El primer caso se refiere a la doble detección y el segundo a la detección actual. - Fallos al confundir la textura de la persona con la calle						
1. Exterior-personas	56	0.6607	0	0.2678	0.071	92.85%	7.14%	**90.24%**
2. Exterior – personas	83	0.7952	0.0963	0.0361	0.0723	83.13%	16.86%	**91.66%**
Total	139	0.6376	0.0412	0.2614	0.0596	88.9%	10.1%	**91.45%**
	Errores	- Cuando las personas o los objetos están muy juntos el sistema los considera como uno solo mediante un mismo rectángulo. Se consideran que son falsos negativos - Aparecen reflejos y sombras que se caracterizan como falsos positivos.						
1. Exterior-personas	110	0.7454	0.1091	0.1363	0.0091	88.17%	11.82%	**98.79%**
2. Exterior – personas	83	0.7952	0.0963	0.0361	0.0723	83.13%	16.86%	**91.66%**
3. Exterior – personas	81	0.8642	0.0741	0.0123	0.0493	87.65%	12.34%	**94.59%**
Total	274	0.7956	0.0948	0.0693	0.0401	86.49%	13.49%	**95.19%**
	Errores	- El porcentaje de detección (sensibilidad) es muy alto, y sólo falla en las entradas o salidas de personas de la escena cuando una pequeña parte de ellas puede apreciarse. - Los falsos positivos que se producen por las sombras son los más frecuentes dentro del porcentaje de fallos.						

Tabla 5.7. Vídeos en exteriores II

CAPÍTULO 6

CONCLUSIONES Y LÍNEAS FUTURAS

En este último capítulo se expondrán las conclusiones del trabajo realizado en este proyecto, y se propondrán diferentes pasos a seguir para mejorar el trabajo en un futuro. A semejanza de un resumen final, se hará una recopilación del trabajo y de la investigación llevada a cabo durante la realización del proyecto, y se realizará una revisión de los resultados globales obtenidos tras las pruebas del capítulo anterior.

6.1. CONCLUSIONES

Al comienzo de este proyecto nos planteamos el objetivo de diseñar un prototipo de una aplicación de detección de movimiento, que fuera ligera y que se pudiera implementar a posteriori en dispositivos móviles tales como teléfonos móviles. El entorno que se eligió para el desarrollo del prototipo fue Matlab, por su elevada versatilidad en el tratamiento de imágenes y vídeos.

Previamente al inicio del desarrollo de la aplicación, se hizo un repaso de las etapas que conforman un sistema típico de vigilancia, y a las principales técnicas existentes en la literatura especializada para llevar a cabo una aplicación de detección de movimiento. A pesar de la gran diversidad de posibilidades, para realizar este proyecto nos decantamos por una opción que, al parecer, no ha sido demasiado explorada, la similitud estructural. Por tanto, hemos utilizado un método de evaluación de la calidad de imagen para establecer las diferencias entre fotogramas consecutivos y determinar los elementos que se están moviendo entre ambos. Adicionalmente, se realizó un repaso de distintas técnicas de medida de evaluación de la calidad de imagen.

Por el estudio de los diferentes mecanismos para establecer la calidad de una imagen, conocíamos las características del método SSIM y sabíamos que sus prestaciones eran adecuadas para una buena evaluación de calidad. Sin embargo, necesitábamos

comprobar cuál era su comportamiento aplicado a la detección de movimiento, y si su rendimiento era lo suficientemente satisfactorio como para basar un sistema de detección de movimiento en este algoritmo.

Una vez desarrollado el prototipo completo que realizaba todas las funciones previstas de detección, se procedió a evaluar el comportamiento mediante una serie de vídeos prueba, que recogían situaciones concretas y sencillas. Los resultados que se derivaron de estas pruebas eran muy satisfactorios, ya que el algoritmo SSIM determinaba con precisión todos los posibles movimientos. Después se utilizó una serie de vídeos que reflejan situaciones más complejas para comprobar la respuesta ante distintos escenarios reales que se alejan de los modelos más sencillos y para encontrar las limitaciones del sistema. Estos casos son los que hemos reflejado en las tablas del capítulo 5.

Las pruebas que se han realizado comprendían la detección de personas y vehículos en espacios abiertos y de personas y objetos (en nuestro caso hemos elegido pelotas de tenis) en escenarios de interior. En las mejores condiciones de detección tanto en interior como en exterior, la sensibilidad se encuentra alrededor del 90%, y no se detectan grandes errores. Sin embargo, a continuación vamos a hacer referencia a algunas situaciones concretas en las que la efectividad es inferior.

Como hemos señalado en algunos casos de la tabla, en ocasiones cuando los elementos a detectar son de pequeño tamaño, no se distinguen bien del fondo o su desplazamiento es poco apreciable, el sistema puede tener problemas para detectarlos y el rendimiento disminuye. Sin embargo, también se ha visto reflejado que realizar la segunda comparación aumenta la capacidad de detección; ya que comparar el fotograma actual con el que corresponde al movimiento inicial permite detectar movimientos mucho más sutiles, que no son apreciables entre fotogramas consecutivos.

Los problemas más frecuentes que se han encontrado en la detección se deben a la aparición de falsos positivos por la presencia de sombras, reflejos…, que el sistema considera como objetos reales en movimiento y que alteran la detección. A la hora de hacer las medidas, hemos considerado que la aparición de estos fenómenos es suficiente para disminuir el porcentaje de acierto, aunque los objetos de interés se detecten en todo momento. En la misma línea, también hemos comentado el efecto negativo que suponen los cambios bruscos de iluminación provocados por algunos elementos inmóviles del escenario.

Como SSIM reacciona de una manera similar a como lo hace el sistema de visión humano, al disminuir la iluminación de una escena, si los elementos son menos

perceptibles y se distinguen con mayor dificultad, entonces el rendimiento del sistema disminuye.

En vista a las pruebas realizadas podemos concluir que SSIM es un método eficaz para la detección de movimiento y que puede utilizarse de igual forma que otros métodos más típicos en aplicaciones de vigilancia. De acuerdo con los resultados que hemos visto en las tablas, SSIM tiene un buen comportamiento tanto en entornos de interior como de exterior cuando las condiciones de captura de las secuencias de vídeo son estándar.

6.2. LÍNEAS FUTURAS

A partir del funcionamiento básico del prototipo presentado, existen puntos que pueden mejorarse y desarrollarse con mayor profundidad:

- Establecimiento del umbral: Hemos comentado a lo largo de todo el trabajo la importancia de elegir el umbral más adecuado. Una buena elección hace que sea posible realizar la detección de objetos con precisión en cada situación. En este trabajo hemos seleccionado para todos los casos un valor de umbral de antemano. Sin embargo, sería conveniente encontrar un mecanismo que permitiera ajustar el valor del umbral a la evolución de las condiciones del entorno.
- Eliminación de sombras: A pesar de que el funcionamiento del sistema sea correcto en una situación concreta y que todos los blancos se detecten, las sombras que estos proyectan sobre el suelo también se perciben. El hecho de que se detecten las sombras es una prueba de que el sistema funciona, pero de cara a la vigilancia de personas estaríamos considerando que tenemos más blancos de los que realmente hay. Sería conveniente buscar algún tipo de solución que permita delimitar las sombras de los objetos en movimiento y eliminarlas.
- Eliminación del movimiento de la cámara: Aunque el sistema desarrollado puede funcionar con pequeñas vibraciones de la cámara, grandes alteraciones pueden hacerlo ineficiente. Por tanto, es conveniente asegurar que no se producen ningún tipo de movimientos de la cámara. Para solucionarlo podemos elegir desde incorporar a la cámara mecanismos estabilizadores de imagen o añadir algún tipo de procesado para alinear imágenes.
- Alternativa a la etapa de calibrado: Durante el calibrado localizamos las zonas de la escena con movimiento que nos van a alterar el análisis posterior. Una alternativa o complemento que se puede incorporar a la aplicación final en un

dispositivo móvil, es permitir definir al usuario las coordenadas de una región del espacio donde realizar la detección.

- Solución del efecto de oclusión: La oclusión se produce cuando algún elemento se interpone delante del objeto en movimiento. Este efecto puede clasificarse en tres grupos: Oclusión propia, que ocurre cuando una parte del objeto tapa a otra y es más frecuente en objetos articulados; oclusión entre objetos, dos objetos en movimiento que están siguiéndose se solapan; y oclusión con elementos de inmóviles del escenario.
- Desarrollo de una aplicación para plataformas móviles: Una vez se ha completado el desarrollo del prototipo, el objetivo final sería implementar una aplicación basada en el mismo que sea capaz de funcionar en dispositivos móviles. El mecanismo empleado en este trabajo es lo suficientemente ligero y robusto como para que pueda ser adaptado para su uso en teléfonos móviles, tabletas, etc.

REFERENCIAS

[1] Z. Wang, E. P. Simoncelli, A. C. Bovik, "Multi-scale structural similarity for image quality assessment", *Proceedings of 37th IEEE conference of Signals, Systems and Computers 2003*.

[2] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity", *IEEE Transactions on Image Processing, vol 13, no 4, April 2004*.

[3] R. C. Gonzalez, R. E. Brooks, "Digital Image Processing", Addison-Wesley, Reading, Massachusetts, 2008.

[4] R. C. Gonzalez, R. E. Brooks "Digital Image Processing using Matlab", Upper Saddle River, NJ, Pearson, 2004.

[5] "Image processing toolbox. User´s guide", MathWorks, 2011.

[6] M. Valera, S.A. Velastin, "Intelligent distributed surveillance systems: a review", *IEE Proc.-Vis. Image Signal Process., Vol. 152, No 152, April 2005*.

[7] M. D. Rodríguez, M. Shah, "Visual surveillance in maritime port facilities".

[8] D. Beymer, P. McLauchlan, B. Coifman, J. Malik, "A real-time computer vision system for measuring traffic parameters".

[9] A. Yilmaz, O.Javed, M. Shah, "Object tracking. A review", ACM Computing Surveys, Vol. 38, No. 4, Article 13, December 2006.

[10] S. Srivastava, K. K. Ng, E. J. Delp, "Crowd flow estimation using multiple visual features for scenes with changing crowd densities", *8th IEEE Conference on Advanced Video and Signal-Based Surveillance 2011*.

[11] https://play.google.com/store fecha de última visita: 4 de julio de 2012, Google Play.

[12] T. Ko, “A survey on behaviour analysis in video surveillance applications*”, Raytheon Company, USA.*

[13] Z. Wang, A. C. Bovik, “A universal image quality index”, *IEEE Signal Processing Letters, Vol. 9, No. 3, March 2002*.

[14] H. R. Sheikh, A. C. Bovik, “A Visual Information Fidelity Approach to Video Quality Assessment”, the First International Workshop on Video Processing and Quality Metrics for Consumer Electronics, January 23-25, 2005, Scottsdale, AZ.

[15] J. Rymel, J. Renno, D. Greenhill and J. Orwell, G.A. Jones, “Adaptive eigen-backgrounds for object detection”, *Digital Imaging Research Centre, School of Computing and Information Systems, Kingston University*.

[16] L. A. Zadeh, “Fuzzy Sets”, *Information and Control 8, p.338-353, 1965.*

[17] H. R. Sheikh, A. C. Bovik, “Image Information and Visual Quality”, *IEEE Transactions on Image Processing, Vol. 15, No. 2, February 2006*.

[18] S. Aja Fernández, R. San José Estepar, C. Alberola López, C. F. Westin, “Image Quality Assessment Based on Local Variance*”, Proceedings of the 28th IEEE EMBS Annual International Conference, New York City, USA, Aug 30-Sept 3, 2006*.

[19] Z. Wang, Q. Li, “Information Content Weighting for Perceptual Image Quality Assessment”, *IEEE Transactions on Image Processing, Vol. 20, No. 5, MAY 2011*.

[20] D. Comaniciu, P. Meer, “Mean Shift: A Robust Approach toward Feature Space Analysis”.

[21] Z. Wang, A.C. Bovik**, "**Mean squared error: Love it or leave it? - A new look at signal fidelity measures"**,** *IEEE Signal Processing Magazine*, Vol: 26 No: 1, Page(s): 98-117, January 2009.

[22] D. V. Weken, M. Nachtegael, V. De Witte, S. Schulte, E. Kerre*,* “Constructing similarity measures for color images”, *Fuzzy logic, soft computing and computational intelligence: eleventh international fuzzy systems association world congress, 2005*.

[23] L. Kallin Westin, “Receiver Operating Characteristic (ROC) Analysis – Evaluating discriminance effects among decision support systems”, Department of Computing Science, Umea University.

[24] T. Fawcett, “ROC Graphs: Notes and Practical Considerations for Researchers”, *HP Laboratories, Palo Alto, CA, March 16, 2004*.

[25] D. V. Weken, M. Nachtegael, E. Kerre, “Some New Similarity Measures for Histograms”, *Proceedings of ICVGIP, 2004, 441-446*.

[26] https://ece.uwaterloo.ca/~z70wang/research/ssim/ fecha de última visita: 31 de julio de 2012, University of Waterloo, Electrical and Computer Engineering.

[27] D. Comaniciu, V. Ramesh, P. Meer, “Kernel-Based Object Tracking”.

[28] www.mathworks.com fecha de última visita: 31 de julio de 2012, MathWorks.

ANEXO A: CÓDIGOS

En el presente anexo se incluyen los códigos de Matlab que se han empleado en el desarrollo de la aplicación de detección de movimiento:

- **Principal**

```
%Este script se corresponde con la parte principal de la aplicación.
%Comprende la lectura del archivo de vídeo deseado por el usuario, la
%ejecución de la fase de calibrado completa, la fase de detección y la
%llamada a las funciones necesarias. La última parte del código, se
%corresponde con la visualización de los resultados.


vid=VideoReader('ext8.avi');

%%%%%%%%%%%%%%%%%%%%%%%%%%%% CALIBRADO %%%%%%%%%%%%%%%%%%%%%%%%%%%%

fondo1=0;

for k=2:5:10
    img1=read(vid,k);
    img2=read(vid,k-1);

    i1=rgb2gray(img1);
    i2=rgb2gray(img2);

    [mssim,ssim_map]=ssim(i1,i2);

    BW=im2bw(ssim_map,0.45);
    BW=1-BW;

    BW_fill=imfill(BW);
    H=[0 0 0 1 0 0 0;0 1 1 1 1 1 0;0 1 1 1 1 1 0;1 1 1 1 1 1 1; ...
        0 1 1 1 1 1 0;0 1 1 1 1 1 0;0 0 0 1 0 0 0];
    BW=imdilate(BW_fill,H);
    fondo1=fondo1+BW;

end
```

```
%Los puntos del fondo que se repiten con más frecuencia tienen unos
%valores más altos dentro de la imagen fondo1. Se puede elegir la
%precisión del fondo cambiando el parámetro para quedarnos con mayor o
%menor número de repeticiones

[f,c]=size(fondo1);
fondo2=(fondo1>2);

fondo=imfill(fondo2,'holes');
fondo=imdilate(fondo,H);
imshow(fondo);

%%%%%%%%%%%%%%%%%%%%%%%%%% FIN CALIBRADO %%%%%%%%%%%%%%%%%%%%%%%%%%


comprobar_act=0;
comprobar_ant=0;

%La variable exceso indica si el fotograma actual no tiene movimiento
%pero los anteriores sí lo tienen.
exceso=0;

%Las variables siguientes se utilizan para calcular el centro de los
%objetos que se detectan
centrox=0;
centroy=0;

%En la fase de detección se aplica el algoritmo SSIM con valor de
%umbral 0.5. Se calculan los puntos marcados, por el área de los
objetos o sumando los pixeles obtenidos. Si este número supera un
porcentaje se inicia la doble comparación.

%Cuando se detecta movimiento en una imagen comprobar_act se hace 1 y
%comprobar_ant sigue siendo 0. Entonces se marca la imagen actual como
%referencia (patron) para comparar con ella.
%En la siguiente pasada del bucle comprobar_ant se hace 1 (toma el
%valor que tiene comprobar_act) y entonces se genera una nueva imagen
%ssim con el frame actual y la imagen de referencia. Esta segunda
%comparación se mantiene un frame después de que haya desaparecido el
%movimiento, exceso=1.


%%%%%%%%%%%%%%%%%%%%%%%%%%%% DETECCIÓN %%%%%%%%%%%%%%%%%%%%%%%%%%%%


for k=10:5:vid.NumberOfFrames

    comprobar_ant=comprobar_act;

    img1=read(vid,k);
    img2=read(vid,k-3);

    i1=double(rgb2gray(img1));
    i2=double(rgb2gray(img2));
```

```
%Se calcula el mapa ssim y se suma el fondo del calibrado para
%calcular el histograma correctamente
    [mssim,ssim_map]=ssim(i1,i2);
    fin=ssim_map+fondo;

    [h,x]=hist(fin(fin<0.75),linspace(0,0.8,100));

    rgb=im_rgb(uint8(img2),ssim_map,fondo);

    %dibujar;
%Utilizando la función im_rgb generamos la imagen que se corresponde
%con la imagen que tiene los píxeles de movimiento marcados en rojo.
%Con el script dibujar podemos representar sobre la imagen los
%rectángulos o puntos que se corresponden con los objetos presentes.


    %pp=(ssim_map<0.5);
    %n_pix=sum(sum(pp));
    porcen=(area*100)/(c*f);
%Podemos elegir calcular el porcentaje mediante el área que se calcula
%en el script dibujar o sumando los píxeles con valor de índice ssim
%menor que 0.5.

    if(0.08>porcen)
        comprobar_act=0;
    else
        comprobar_act=1;
    end

    if(comprobar_act==0 && comprobar_ant==1)
        exceso=1;
    end

%Si comprobar_ant==0 el mecanismo se reinicia y se colocan todas las
%variables a cero
    if(comprobar_ant==0)
        exceso=0;
        primero=zeros(f,c);
        centrox=[];
        centroy=[];
    end

%Cuando se detecta por primera vez que existe movimiento, se inician
%la inician la imagen de referencia patron con la que se va a realizar
%la segunda comparación y la imagen primero que se utiliza para
%eliminar en la segunda comparación la primera aparición del objeto en
%la escena por riesgo de que su primera situación pueda permanecer
%visible durante todo el análisis.
    if(comprobar_ant==0 && comprobar_act==1)
        patron=i2;
        primero=(ssim_map<0.5);

        %Las siguientes líneas permiten determinar el centro de cada
        %objeto; únicamente son relevantes si estamos representando
        %los objetos mediante puntos, de otra manera no son
        %importantes
```

```
        %objetos=estr.NumObjects;
        %centrox=[];
        %centroy=[];
        %for u=1:estr.NumObjects
        %    centrox=[centrox round((aa+bb)/2)];
        %    centroy=[centroy round((cc+dd)/2)];
        %end

    end

%La segunda comparación se produce cuando se dan las condiciones
%necesarias comprobar_act==1, comprobar_ant==1 y exceso~=0, entonces
%sec=1.
    if((comprobar_act==1 && comprobar_ant==1)|| exceso~=0)
        [mssim2,ssim_map2]=ssim(i1,patron);
        rgb2=im_rgb_seguimiento(uint8(i2),ssim_map2,fondo,primero);
        dibujar_seguimiento;
        sec=1;
    else
        sec=0;
    end


%Sólo se inicia la doble comparación cuando hay al menos dos imágenes
%ssim consecutivas en las cuales se ha detectado un número de píxeles
%de movimientos superiores al porcentaje. Esta segunda comparación se
%mantiene un frame después de que se haya dejado de detectar un
%movimiento.



    %%%%%%%%%%%%%%%%%%%%%%%% VISUALIZACIÓN %%%%%%%%%%%%%%%%%%%%%%%%%%%

%Para la visualización podemos elegir entre las distintas imágenes que
%hemos generado durante el programa: rgb y rgb2 que contienen los
%píxeles en movimiento marcados en rojo, AA y AA2 que representan los
%objetos mediante rectángulos, y las imágenes BB y BB2 que utilizan la
%representación por puntos.



    subplot(2,3,1)
    imshow(i1,[]),title(sprintf('frame actual %d',k))
    subplot(2,3,2)
    imshow(AA,[]),title('movimiento actual')
    subplot(2,3,4)
    imshow(ssim_map),title('ssim map actual'),xlabel(sprintf('num...
pixels %d area %d',n_pix,area))

    subplot(2,3,5)
    plot(x,h),title('histograma')
    subplot(2,3,6)
    if(sec)
        imshow(ssim_map2),title('ssim map... relativo')
    else
        imshow(ones(size(ssim_map))),title('ssim map... relativo')
    end
    subplot(2,3,3)
```

```
    if(sec)
        imshow(AA2,[]),title('movimiento relativo')
    else
        imshow(AA,[]),title('movimiento relativo')
    end

    pause

end
```

- **SSIM**

```
function [mssim, ssim_map] = ssim(img1, img2, K, window, L)

%
========================================================================
==
% SSIM Index with automatic downsampling, Version 1.0
% Copyright(c) 2009 Zhou Wang
% All Rights Reserved.
%
% ----------------------------------------------------------------------
--
% Permission to use, copy, or modify this software and its
%documentation for educational and research purposes only and without
%fee is hereby granted, provided that this copyright notice and the
%original authors' names appear on all copies and supporting
%documentation.
%This program shall not be used, rewritten, or adapted as the basis of
%a commercial  software or hardware product without first obtaining
%permission of the authors. The authors make no representations about
%the suitability of this software for any purpose. It is provided "as
%is" without %express or implied warranty.
%----------------------------------------------------------------------
-
%
% This is an implementation of the algorithm for calculating the
% Structural SIMilarity (SSIM) index between two images
%
% Please refer to the following paper and the website with suggested
usage
%
% Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image
% quality assessment: From error visibility to structural similarity,"
% IEEE Transactios on Image Processing, vol. 13, no. 4, pp. 600-612,
% Apr. 2004.
%
% http://www.ece.uwaterloo.ca/~z70wang/research/ssim/
%
% Note: This program is different from ssim_index.m, where no
automatic
% downsampling is performed. (downsampling was done in the above paper
% and was described as suggested usage in the above website.)
%
% Kindly report any suggestions or corrections to zhouwang@ieee.org
%
%----------------------------------------------------------------------
-
%
%Input : (1) img1: the first image being compared
%        (2) img2: the second image being compared
%        (3) K: constants in the SSIM index formula (see the above
%            reference). default value: K = [0.01 0.03]
%        (4) window: local window for statistics (see the above
%            reference). default window is Gaussian given by
%            window = fspecial('gaussian', 11, 1.5);
```

```
%        (5) L: dynamic range of the images. default: L = 255
%
%Output: (1) mssim: the mean SSIM index value between 2 images.
%            If one of the images being compared is regarded as
%            perfect quality, then mssim can be considered as the
%            quality measure of the other image.
%            If img1 = img2, then mssim = 1.
%        (2) ssim_map: the SSIM index map of the test image. The map
%            has a smaller size than the input images. The actual size
%            depends on the window size and the downsampling factor.
%
%Basic Usage:
%   Given 2 test images img1 and img2, whose dynamic range is 0-255
%
%   [mssim, ssim_map] = ssim(img1, img2);
%
%Advanced Usage:
%   User defined parameters. For example
%
%   K = [0.05 0.05];
%   window = ones(8);
%   L = 100;
%   [mssim, ssim_map] = ssim(img1, img2, K, window, L);
%
%Visualize the results:
%
%   mssim                         %Gives the mssim value
%   imshow(max(0, ssim_map).^4)  %Shows the SSIM index map
%========================================================================
===


if (nargin < 2 || nargin > 5)
   mssim = -Inf;
   ssim_map = -Inf;
   return;
end

if (size(img1) ~= size(img2))
   mssim = -Inf;
   ssim_map = -Inf;
   return;
end

[M N] = size(img1);

if (nargin == 2)
   if ((M < 11) || (N < 11))
       mssim = -Inf;
       ssim_map = -Inf;
      return
   end
   window = fspecial('gaussian', 11, 1.5);  %
   K(1) = 0.01;                 % default settings
   K(2) = 0.03;                 %
   L = 255;                                  %
end
```

```
if (nargin == 3)
   if ((M < 11) || (N < 11))
       mssim = -Inf;
       ssim_map = -Inf;
      return
   end
   window = fspecial('gaussian', 11, 1.5);
   L = 255;
   if (length(K) == 2)
      if (K(1) < 0 || K(2) < 0)
           mssim = -Inf;
        ssim_map = -Inf;
        return;
      end
   else
       mssim = -Inf;
    ssim_map = -Inf;
       return;
   end
end

if (nargin == 4)
   [H W] = size(window);
   if ((H*W) < 4 || (H > M) || (W > N))
       mssim = -Inf;
       ssim_map = -Inf;
      return
   end
   L = 255;
   if (length(K) == 2)
      if (K(1) < 0 || K(2) < 0)
           mssim = -Inf;
        ssim_map = -Inf;
        return;
      end
   else
       mssim = -Inf;
    ssim_map = -Inf;
       return;
   end
end

if (nargin == 5)
   [H W] = size(window);
   if ((H*W) < 4 || (H > M) || (W > N))
       mssim = -Inf;
       ssim_map = -Inf;
      return
   end
   if (length(K) == 2)
      if (K(1) < 0 || K(2) < 0)
           mssim = -Inf;
        ssim_map = -Inf;
        return;
      end
   else
       mssim = -Inf;
```

```
    ssim_map = -Inf;
       return;
   end
end


img1 = double(img1);
img2 = double(img2);

% automatic downsampling
f = max(1,round(min(M,N)/256));
%downsampling by f
%use a simple low-pass filter
if(f>1)
    lpf = ones(f,f);
    lpf = lpf/sum(lpf(:));
    img1 = imfilter(img1,lpf,'symmetric','same');
    img2 = imfilter(img2,lpf,'symmetric','same');

    img1 = img1(1:f:end,1:f:end);
    img2 = img2(1:f:end,1:f:end);
end

C1 = (K(1)*L)^2;
C2 = (K(2)*L)^2;
window = window/sum(sum(window));

mu1   = filter2(window, img1, 'valid');
mu2   = filter2(window, img2, 'valid');
mu1_sq = mu1.*mu1;
mu2_sq = mu2.*mu2;
mu1_mu2 = mu1.*mu2;
sigma1_sq = filter2(window, img1.*img1, 'valid') - mu1_sq;
sigma2_sq = filter2(window, img2.*img2, 'valid') - mu2_sq;
sigma12 = filter2(window, img1.*img2, 'valid') - mu1_mu2;

if (C1 > 0 && C2 > 0)
   ssim_map = ((2*mu1_mu2 + C1).*(2*sigma12 + C2))./((mu1_sq + mu2_sq
+ C1).*(sigma1_sq + sigma2_sq + C2));
else
   numerator1 = 2*mu1_mu2 + C1;
   numerator2 = 2*sigma12 + C2;
   denominator1 = mu1_sq + mu2_sq + C1;
   denominator2 = sigma1_sq + sigma2_sq + C2;
   ssim_map = ones(size(mu1));
   index = (denominator1.*denominator2 > 0);
   ssim_map(index) =
(numerator1(index).*numerator2(index))./(denominator1(index).*denomina
tor2(index));
   index = (denominator1 ~= 0) & (denominator2 == 0);
   ssim_map(index) = numerator1(index)./denominator1(index);
end

mssim = mean2(ssim_map);

return
```

- **Dibujar**

```
%Este script tiene como objetivo dibujar tanto los rectángulos como
%los puntos que representan los objetos. Calcula los parámetros
%necesarios para pasarlos a las funciones correspondientes.

%En la imagen AA se representarán los rectángulos que envuelven los
%objetos presentes en la escena y sobre BB se dibujarán los puntos.
AA=img1;
BB=img1;

%Se realizan operaciones de relleno y dilatación sobre la imagen
%binaria del mapa ssim.
bin=im2bw(ssim_map,0.5);
bin=1-bin;
bin=imfill(bin);
bin=imdilate(bin,H);
bin=imfill(bin);
bin=imdilate(bin,H);

%Los siguientes comandos obtienen las características de área y de la
%mínima región que envuelve a cada elemento de la imagen actual
estr = bwconncomp(bin, 4);
caja=regionprops(estr,'BoundingBox','Area');
area=0;

%Cada pasada del bucle trabaja sobre uno de los elementos que se han
%detectado.
for u=1:estr.NumObjects
    %Se calculan todas las coordenadas de los vértices del rectángulo.
    aa=round(caja(u).BoundingBox(1))+5;
    bb=aa+caja(u).BoundingBox(3)-5;
    cc=round(caja(u).BoundingBox(2))+5;
    dd=cc+caja(u).BoundingBox(4)-5;

    %Se calculan las coordenadas del punto central del rectángulo.
    pmx=round((aa+bb)/2); %borrar en caso de que
    pmy=round((cc+dd)/2);

    %Se obtiene el área total de todos los objetos de la imagen
    area=area+caja(u).Area;

    %Se llama a las funciones de dibujo
    AA=rectangulo(AA,f,c,aa,bb,cc,dd);
    %BB=punto(BB,aa,bb,cc,dd);

end
```

- **Rectángulo**

```
%Esta función dibuja un rectángulo de color verde sobre el fotograma a
%visualizar que enmarca los objetos detectados anteriormente. La
%función recibe como parámetros la imagen RGB sobre la que se va a
%dibujar el punto, el número de filas y columnas, y las coordenadas
%cartesianas de la situación de los cuatro vértices del rectángulo.
%Devuelve la imagen en color con el rectángulo dibujado.
%Para tener en cuenta que el tamaño del mapa SSIM es de menor tamaño
%que la imagen real, se desplaza el dibujo hacia abajo y hacia la
%derecha.

function AA=rectangulo(AA,f,c,aa,bb,cc,dd)

for ii=1:f
    for jj=1:c

        if((ii==cc || ii==dd)&&(jj>=aa && jj<=bb))
            AA(ii+5,jj+5,2)=255;
            AA(ii+5,jj+5,3)=0;
            AA(ii+5,jj+5,1)=0;
        end
        if((jj==aa || jj==bb)&&(ii>=cc && ii<=dd))
            AA(ii+5,jj+5,2)=255;
            AA(ii+5,jj+5,3)=0;
            AA(ii+5,jj+5,1)=0;
        end
    end
end
```

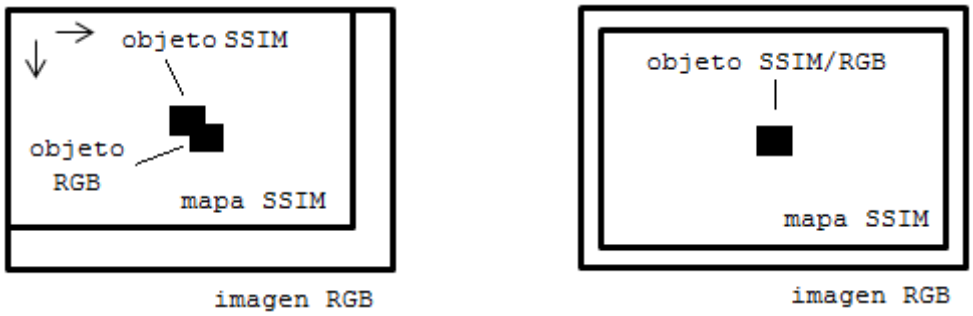


Figura A.1. Desplazamiento del mapa SSIM.
Debido al distinto tamaño de las dos imágenes, las coordenadas de un objeto en el mapa SSIM no se corresponden con el objeto en la imagen RGB. Intentamos centrar el mapa SSIM para que las representaciones de los píxeles en movimiento y del rectángulo sea más próxima a la realidad cuando se visualizan los resultados.

- **Punto**

```
%Esta función dibuja una cruz de color amarillo de 10x10 píxeles sobre
%el fotograma a visualizar que indica el centro de los objetos
%detectados anteriormente. La función recibe como parámetros la imagen
%RGB sobre la que se va a dibujar el punto y las coordenadas
%cartesianas de los cuatro vértices del rectángulo que envuelve el
%objeto, calculado en dibujo, para luego obtener el centro. Devuelve
%la imagen BB con la cruz dibujada.

function BB=punto(BB,aa,bb,cc,dd)

pmx=round((aa+bb)/2);
pmy=round((cc+dd)/2);


for ii=(pmx-5):(pmx+5)
    if(pmx>5 && pmx<316)
        BB(pmy,ii,1)=255;
        BB(pmy,ii,2)=255;
        BB(pmy,ii,3)=0;
    end
end
for ii=(pmy-5):(pmy+5)
    if(pmy>5 && pmy<226)
        BB(ii,pmx,1)=255;
        BB(ii,pmx,2)=255;
        BB(ii,pmx,3)=0;
    end
end
```

- **Im RGB**

```
%Esta función se encarga de marcar en rojo sobre la imagen a color los
%píxeles del mapa SSIM que tienen un valor menor que 0.5, siempre que
%estos no pertenezcan a la zona excluida del análisis por el proceso
%de calibrado. La función recibe la imagen a color del instante
%actual, el mapa SSIM y la imagen fondo que se corresponde con la
%máscara generada en el calibrado. La función devuelve la imagen con
%los píxeles marcados.

function rgb=im_rgb(img2,ssim_map,fondo)

[f,c]=size(fondo);

for ii=1:f
    for jj=1:c
        if(ssim_map(ii,jj)<0.50 && fondo(ii,jj)==0)
            img2(ii+5,jj+5,1)=255;
            img2(ii+5,jj+5,2)=0;
            img2(ii+5,jj+5,3)=0;
        end
    end
end

rgb=img2;
```

- **Im RGB seguimiento**

```
%La siguiente función tiene la misma utilidad que la función anterior,
%pero se utiliza durante la fase de la segunda comparación. Se
%diferencia de la anterior en que el proceso para conseguir la imagen
%final no es el mismo, para que puedan verse dos maneras de lograr el
%mismo objetivo.

function rgb=im_rgb_seguimiento(i2,ssim_map,fondo,primero)

[f,c]=size(fondo);
r=i2;
g=i2;
b=i2;
for ii=1:f
    for jj=1:c
        if(ssim_map(ii,jj)<0.50 && fondo(ii,jj)==0 &&
primero(ii,jj)==0)
            r(ii,jj)=255;
            g(ii,jj)=0;
            b(ii,jj)=0;
        end
    end
end
rgb=cat(3,r,g,b);
```

- **Dibujar seguimiento**

```
AA2=img1;

bin=im2bw(ssim_map2,0.5);
bin=1-bin;
bin=imfill(bin);
bin=imdilate(bin,H);
bin=imfill(bin);
bin=imdilate(bin,H);

estr = bwconncomp(bin, 4);
caja=regionprops(estr,'BoundingBox');


for u=1:estr.NumObjects
    aa=round(caja(u).BoundingBox(1))+5;
    bb=aa+caja(u).BoundingBox(3)-5;
    cc=round(caja(u).BoundingBox(2))+5;
    dd=cc+caja(u).BoundingBox(4)-5;

    AA2=rectangulo(AA2,f,c,aa,bb,cc,dd);

    %a=[pmx,pmy];
    %b=[centrox(1),centroy(1)];

    %d=b-a;
    %p=d(2)/d(1);

    %for ii=1:abs(a(1)-b(1))
    %    x= a(1)+(d(1)/abs(d(1)))*ii;
    %    y= round(a(2)+p*ii);

    %    AA2(y,x,1)=0;
    %    AA2(y,x,2)=0;
    %    AA2(y,x,3)=255;
    %end
end
%for u=1:objetos
%    AA2=punto_seguimiento(AA2,centrox(u),centroy(u));
%end
```

- **Punto seguimiento**

```
%Esta función dibuja una cruz de color amarillo de 10x10 píxeles sobre
%el fotograma a visualizar que indica el centro de los objetos
%detectados anteriormente. La función recibe como parámetros la imagen
%RGB sobre la que se va a dibujar el punto y las coordenadas
%cartesianas de la situación del centro del objeto. Devuelve la
%imagen con la cruz dibujada.

function BB=punto_seguimiento(BB,pmx,pmy)


for ii=(pmx-5):(pmx+5)
    if(pmx>5 && pmx<316)
        BB(pmy,ii,1)=255;
        BB(pmy,ii,2)=255;
        BB(pmy,ii,3)=0;
    end
end
for ii=(pmy-5):(pmy+5)
    if(pmy>5 && pmy<226)
        BB(ii,pmx,1)=255;
        BB(ii,pmx,2)=255;
        BB(ii,pmx,3)=0;
    end
end
```