UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

RECONSTRUCCIÓN DE FONDO DE ESCENA A PARTIR DE SECUENCIAS DE VÍDEO

Autor: Carolina Fernández-Pedraza Jorde.
Tutor: Diego Ortego Hernández.
Ponente: José María Martínez Sánchez.

Julio 2016

RECONSTRUCCIÓN DE FONDO DE ESCENA A PARTIR DE SECUENCIAS DE VÍDEO

Autor: Carolina Fernández-Pedraza Jorde
Tutor: Diego Ortego Hernández
Ponente: José María Martínez Sánchez

Video Processing and Understanding Lab
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio 2016

Trabajo parcialmente financiado por el Ministerio de Economía y Competitividad del Gobierno de España bajo el proyecto TEC2014-53176-R (HAVideo) (2015-2017)

Resumen

En este TFG se proponen una serie de algoritmos de inicialización de fondo que se basan tanto información temporal como información espacial.

Primero, se estudiará el estado del arte. Después, se procederá a la implementación de dos algoritmos, uno a nivel de bloque y otro a nivel de pixel, donde se han usado medidas de continuidad del bloque/pixel con su vecindario para valorar las posibilidades de los candidatos a fondo: la región espacial del *frame* actual que se está analizando o la región del *background* que había hasta ahora.

Finalmente, se evaluará el rendimiento de estos dos algoritmos más otros dos que se han considerado relevantes en la literatura respecto a un *dataset* dividido en distintos tipos de desafíos para cada algoritmo.

Palabras clave

Fondo, frente, pixel, bloque, estimación de movimiento, continuidad, vecindario, evento.

Abstract

In this work, several algorithms for background initialization are proposed, all based on temporal and spatial information.

First, the state-of-art has been explored. Then, two algorithms have been implemented, one of them uses block-division and the other one pixel-division, where smoothness has been computed to evaluate what spatial region fits best in the new background: the new region of the actual frame or the region that already exists in the background.

Finally, these two algorithms and other two that have been already implemented, will be evaluated based on a dataset that is classified in several challenges to overcome for each algorithm.

Keywords

Background, foreground, pixel, block, motion estimation, smoothness, neighborhood, event.

Agradecimientos

Lo primero de todo me gustaría agradecer a mi tutor Diego Ortego por haberme dado la oportunidad de realizar este TFG, por la paciencia que ha tenido y por haberme ayudado siempre que lo he necesitado.

A mis padres, porque estos 4 años del Grado no han sido fáciles, pero aun así lo han aguantado todo, siempre ahí y al fin lo he conseguido.

A mis compañeros (y amigos) de clase en especial a Sergio, Edu, Pilar, Silvia, Elisa, Ana y Anthony. Todos sabemos lo duro que ha sido este camino: muchas horas compartidas de estudio, de prácticas, hojas y hojas de apuntes, de exámenes, y bastantes agobios... y gracias a vuestra ayuda, cada uno aportando su granito de arena me lo habeis hecho más fácil, gracias chicos.

Tan importantes han sido las personas con las que he contado dentro de la universidad como las que estaban fuera, las que ayudan a desconectar y sin las que no sería la msima: Clau, Javi, Manolo, Ana, Carlos, Silvia y Fede, son innumerables las cosas que hemos vivido y las que nos quedan por vivir.

Y Marce... todo lo que me has visto conseguir no hubiese sido lo mismo sin ti, gracias por estar.

Gracias a todos.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

En la actualidad, el análisis automático de secuencias de vídeo-vigilancia es un importante área de investigación como consecuencia de la necesidad de seguridad en entornos públicos como aeropuertos, estaciones de tren o eventos masivos. Además, la creciente difusión de sistemas de cámaras y de vídeo-seguridad a un precio asequible ha propiciado un incremento en la implantación de sistemas de análisis automático de vídeo. Una de las principales fases de estos sistemas es la detección de objetos o personas en el entorno analizado, que en muchas ocasiones se realiza a través de algoritmos de sustracción de fondo o *Background Subtraction* [6]. Sin embargo, estos algoritmos tienen una gran limitación a la hora de mantener un modelo de fondo correcto a lo largo del tiempo, quedando dicho modelo obsoleto en muchas ocasiones. En este punto, la inicialización o re-inicialización del modelo de fondo es necesaria para poder operar de forma fiable, siendo los algoritmos de inicialización de fondo o *Background Initialization* (BI) [7] una respuesta a este problema.

La tarea de BI consiste en, a partir de un conjunto de *frames*, construir un fondo de la escena libre de los objetos que en algún momento han tenido movimiento (ver Figura 1.1). Estos algoritmos de inicialización deben operar correctamente en distintos escenarios, donde dificultades como la densidad de objetos, los cambios de iluminación, los objetos estáticos o el ruido en la imagen son habituales. Es decir, la tarea de BI plantea diversos retos debido a la reducción de la visibilidad del fondo a reconstruir [8]. Por otro lado, los algoritmos de la literatura no comparten un marco común de evaluación, existiendo diversos algoritmos evaluados en distintas secuencias.

Por tanto, la motivación de este TFG es la de contribuir al campo de BI mediante la implementación y evaluación de varios algoritmos para conocer mejor el estado real

Figura 1.1: Ejemplo de inicialización de fondo. Las tres primeras columnas son 3 frames de la secuencia donde se ven objectos que ocluyen el fondo. En la columna de la derecha está el fondo reconstruido por los algoritmos de inicialización de fondo.

de estos algoritmos.

1.2. Objetivos

El objetivo de este TFG es la implementación de varios algoritmos de incialización de fondo del estado del arte y la generación de un dataset para evaluar los algoritmos desarrollados junto con algoritmos relevantes de la literatura. El objetivo principal se divide en los siguientes sub-objetivos:

1. Estudio del estado del arte actual. En esta etapa se van a estudiar las diversas propuestas existentes en la literatura para la inicialización de fondo. Este estudio tiene como objetivo conocer las soluciones actuales para BI.

2. Implementación de algoritmos: En esta etapa se van a seleccionar e implementar algoritmos de la literatura en MATLAB. El objetivo es incrementar el número de algoritmos disponibles en el VPULab, para comprender mejor el rendimiento de los algoritmos actuales de BI.

3. Elaboración del *dataset*: En esta etapa se va a construir un *dataset* con distintos tipos de retos a superar por los algoritmos. El objetivo es crear un conjunto de secuencias con retos diversos para, posteriormente, evaluar el rendimiento de los algoritmos.

4. Evaluación de los algoritmos: En esta etapa final, se evaluaran los algortimos implementados juntos con técnicas relevantes de la literatura. El objetivo es observar en qué escenarios funcionan mejor cada uno de ellos.

1.3. Organización de la memoria

La memoria consta de los siguientes capítulos:

- Capítulo 1: Introducción, motivación del trabajo y objetivos.
- Capítulo 2: Estudio del estado del arte en la inicialización de fondo de escena en secuencias de vídeo.
- Capítulo 3: Algoritmos seleccionados e implementados del estado del arte.
- Capítulo 4: Evaluación de resultados en el *dataset* generado.
- Capítulo 5: Conclusiones generales y trabajo futuro.

Capítulo 2

Estado del arte

2.1. Introducción

La primera etapa de una gran variedad de aplicaciones de vídeo-vigilancia, como el reconocimiento de actividades o el seguimiento de objetos, es segregar el frente (*foreground*, FG) u objetos de la escena del fondo (*background,* BG) de la misma. Esta tarea se suele abordar mediante algoritmos de sustracción de fondo (*Background Subtraction,* BS) [6] que proporcionan máscaras binarias de las regiones de frente mediante comparaciones entre cada frame analizado y un modelo de fondo de la escena. Los algoritmos de sustracción de fondo se componen principalmente de 4 etapas: Modelado, que representa estadísticamente el fondo; Inicialización, que genera un modelo inicial del fondo; Mantenimiento, que actualiza el modelo de fondo para incorporar las variaciones espacio-temporales del mismo; y Detección, que determinar el FG a través de la comparación de los frames entrantes con el modelo de fondo. La inicialización de fondo (*Background Initialization,* BI) [7] es un paso crucial en distintos entornos de aplicación:

- Segmentación de vídeo [9]: Se utiliza el fondo para proceder a la extracción del FG.
- Compresión de vídeo [10]: El fondo representa información redundante de la escena.
- Restauración de vídeo [11]: El fondo ayuda a reparar partes deterioradas de la imagen.
- Privacidad de personas [12]: El fondo es una información que puede explotarse para preservar la privacidad de las personas.

- Fotografía computacional [13]: El fondo proporciona una imagen libre de objetos, que es lo que busca el usuario a partir de un conjunto de frames que contienen objetos de FG.

A pesar de la importancia de la incialización de fondo, se trata de un campo con relativamente poca investigación en comparación con el resto de fases de BS [6].

BI consiste en construir una imagen de fondo de la escena (B) libre de objetos a partir de un conjunto de frames que contienen FG tanto en movimiento como estático. Esta etapa, es el punto de partida del *Modelado* y *Mantenimiento* de la escena. Generalmente, los algoritmos de BS emplean esquemas de BI sencillos, asumiendo que el fondo de la escena se puede obtener fácilmente a partir de los primeros frames de la secuencia, pues se asume que están libres de objetos. Esta suposición puede no cumplirse en muchos escenarios de vídeo-vigilancia como centros comerciales, aeropuertos o estaciones de tren donde puede haber muchos objetos estáticos o multitudes. En definitiva, construir el fondo correctamente es un tarea compleja, cuyos dos problemas principales se pueden resumir a continuación:

- Visibilidad del B: Cuando los píxeles que forman el fondo son visibles durante un corto periodo de tiempo, por la oclusion continuada del mismo, el fondo deja de ser la información temporalmente dominante complicando la tarea de inicialización.

- Factores fotométricos. Los cambios de iluminación pueden variar el valor del BG a lo largo de los frames analizados.

2.2. Algoritmos de la literatura

La tarea de BI recibe diversos nombres en la literatura [8][9]: *Bootstraping* [2][14], BG *estimation* [11][5], BG *generation* [3][15] o BG *reconstruction* [16]. Las estrategias de inicialización de fondo se pueden clasificar según el método en el que se basen [17]: *Temporal Statistics*, *Sub-intervals of stable intensity*, *Iterative Mode Completion* u *Optimal Labeling*. No obstante, se va a utilizar la clasificación propuesta en [18] donde los algoritmos se organizan en función de la estrategia aplicada: temporal o espacial. La información temporal y espacial se puede usar en modo *batch* o en un análisis *online*, operando a nivel de pixel o de bloque.

Las aproximaciones que emplean estrategias temporales están presentes en muchos algoritmos de BS [17] donde B se construye a partir de una actualización del primer *frame* de la secuencia [14][19]. Sin embargo, estas aproximaciones se basan en la mayoría de los casos, en la suposición de que B está presente la mayoría del tiempo en

la escena. La mediana es una alternativa empleada en algunos trabajos [20][21], pero esta estrategia falla cuando los objetos están estáticos más del 50 % del tiempo, pues son considerados como parte de B. Además, algunos algoritmos emplean estimación de movimiento para evitar que objetos del FG pasen a formar parte de B, concretamente flujo óptico [22][23] y diferencias entre *frames* [2][3]. La estabilidad temporal también se utiliza [22][23] para obtener varios candidatos a representar el B en cada región espacial. No obstante, dicha estabilidad no es capaz de agrupar correctamente los candidatos que representan B, ya que supone que los intervalos no continuos de tiempo modelan representaciones distintas de B. En consecuencia, se prefiere emplear técnicas de *clustering* que tienen en cuenta similaridades entre intervalos no continuos [8][5][24].

Aunque algunas aproximaciones solo emplean un análisis temporal [21][1], es necesario incluir también un análisis espacial para evitar la suposición de que B es temporalmente dominante en la secuencia. Es habitual emplear restricciones basadas en el color para decidir si un pixel pertenece a FG o BG que estaba previamente ocluido[3]. En [5], extendido en [24], la continuidad espacial se modela a nivel de bloque usando la Transformada Discreta del Coseno (DCT, *Discrete Cosine Transform*) bajo un *framework* MRF (*Markov Random Field*), junto con un módulo iterativo que corrige posibles errores. Para reducir la complejidad computacional de la DCT, en [25] se emplea la Transformada de Hadamard junto con una corrección del B seleccionado cuando el gradiente en los bordes de bloques adyacentes es alto. En [8], se utiliza un esquema de bloques solapados donde la continuidad espacial reside en las diferencias de color (áreas no solapadas) y en la distancia *chi-square* (áreas solapadas) entre el bloque candidato a B y el B ya fijado. Además, las relaciones de color y gradiente entre bloques vecinos se utilizan en [26] para construir B. Recientemente, [18] realiza una generación del fondo basada en continuidades espaciales multi-camino, es decir, llegando a los bloques a reconstruir desde distintos caminos espaciales. También existen aproximaciones que encapsulan la información de continuidad espacial junto con información temporal en un *frameworks* de minimización de energía como *Loopy Belief Propagation* [27][28], *Graph Cuts* [11], *Conditional Mixed-State* MRF [16] o *Dynamic* MRF[9]. Finalmente, el uso del flujo óptico en posiciones vecinas realizado por [22] y [23] (el primero incluye una ecualización de la intensidad del flujo óptico para compensar las distintas profundidades a los que se mueven objetos en una escena) se puede también considerar como información espacial.

En resumen, se han propuesto varias estrategias para afrontar la tarea de BI. Aunque ninguna de ellas es capaz de operar correctamente en todas las situaciones, se puede observar un cierto consenso en torno a la idea de continuidad espacial en el fondo como mejor alternativa para su reconstrucción.

Capítulo 3

Algoritmos seleccionados

3.1. Introducción

Una vez analizada la literatura, se han seleccionado varios algoritmos para su implementación en MATLAB. Teniendo en cuenta las dimensiones de un TFG, algunos algoritmos no son abordables debido a su complejidad. En consecuencia, se han elegido dos algoritmos[2] [3] de cierta relevancia en el estado del arte a la vez que abordables en un TFG. Estos algoritmos tienen la particularidad de operar de forma online, requisito deseable para un algoritmo de BI, a la vez que uno opera a nivel de pixel [3] y el otro a nivel de bloque [2].

3.2. Algoritmo basado en clasificaciones a nivel de bloque

3.2.1. Estructura general

El algoritmo de BI propuesto en [2] realiza un análisis *online* espacio-temporal a nivel de bloque de la secuencia. Este algoritmo lleva a cabo la inicialización de fondo en 4 etapas (Ver Figura 3.1).

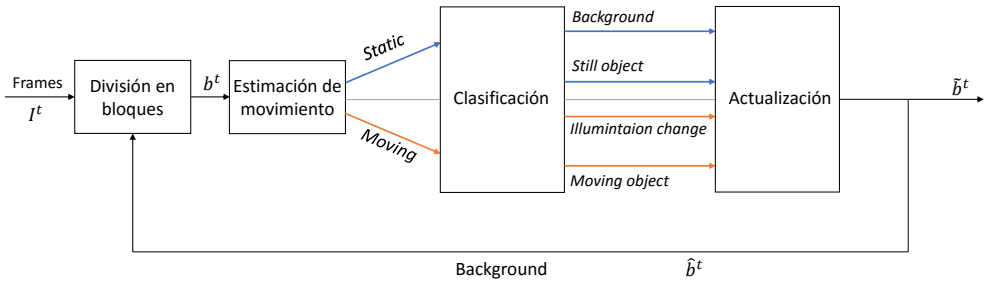


Figura 3.1: Esquema general del algoritmo propuesto en [1] de inicialización de fondo.

1. División en bloques: En primer lugar, se divide el frame actual en bloques cuadrados. En la implementación, en caso de obtener un número no entero de bloques, se ha realizado zero padding para completar los píxeles necesarios de los bordes.

2. Estimación de movimiento: Después, determinamos si ese bloque es estático o no (*static* o *moving*) mediante un mecanismo de estimación de movimiento que hace uso del frame actual, el frame anterior y una medida de comparación entre ambos.

3. Clasificación: A continuación, se hace una clasificación del bloque actual en *background*, *still object*, *illumination change* o *moving object* en función de un umbral y del coeficiente de correlación entre el bloque del frame y el bloque del BG y de si previamente ha sido determinado como *static* o *moving*.

4. Actualización: Finalmente, cada incialización de fondo $\hat{B}^t$se obtiene de una actualización del modelo de fondo $\hat{B}^{t-1}$y la estimación de movimiento. Dicha actualización varía en función del tipo de bloque obtenido en la etapa de Clasificación. Así, una vez analizados todos los frames, se obtiene el modelo de fondo que ha sido actualizado frame a frame valiéndose del tipo de clasificación obtenida.

Para definir las operaciones que realiza el algoritmo es necesario definir la notación del mismo. La estimación del movimiento se lleva a cabo utilizando el frame actual de la secuencia de vídeo I^t(niveles de gris) y el frame anterior I^{t-1}(nivel de gris), mientras que el fondo generado en cada instante temporal t, se designa comoB^t. Por otro lado, $I^t_{(x,y)}$, $I^{t-1}_{(x,y)}$ *y* $B^t_{(x,y)}$ hacen referencia a un píxel genérico *(x,y)* perteneciente, respectivamente, a I^t, I^{t-1} y B^t. Cada frame tiene un tamaño de $W{\times}H$ píxeles, y a su vez, estos se dividen en bloques cuadrados no solapados de $N{\times}N$ píxeles. La notación *(i,j)* indica los índices del bloque donde *i = 0, 1, 2,..., (W/N)-1* y *j = 0, 1, 2,..., (H/N)-1*. De esta manera, $b^t_{(i,j)}$, $b^{t-1}_{(i,j)}$ y $\tilde{b}^t_{(i,j)}$ representan los bloques *(i,j)* de I^t, I^{t-1} y B^t respectivamente. Adicionalmente, $\hat{B}^t$ es el modelo inicial de background y $\hat{b}^t_{(i,j)}$ el bloque *(i,j)* en $\hat{B}^t$.

3.2.2. División en bloques

El algoritmo comienza realizando una división en bloques cuadrados de $N \times N$ píxeles de cada $frame$, tanto para el fondo como para el nuevo *frame* a analizar en cada instante temporal. Además, cuando el algoritmo comienza su operación *(t=1)*

cada bloque $\hat{b}^1_{(i,j)}$ del fondo $\hat{B}^1$se establece como "no definido" y se etiqueta como negro (ver Figura 3.2e). De esta manera, el fondo está incompleto hasta que las sucesivas actualizaciones de fondo $\hat{B}^t$ definan los bloques vacíos.

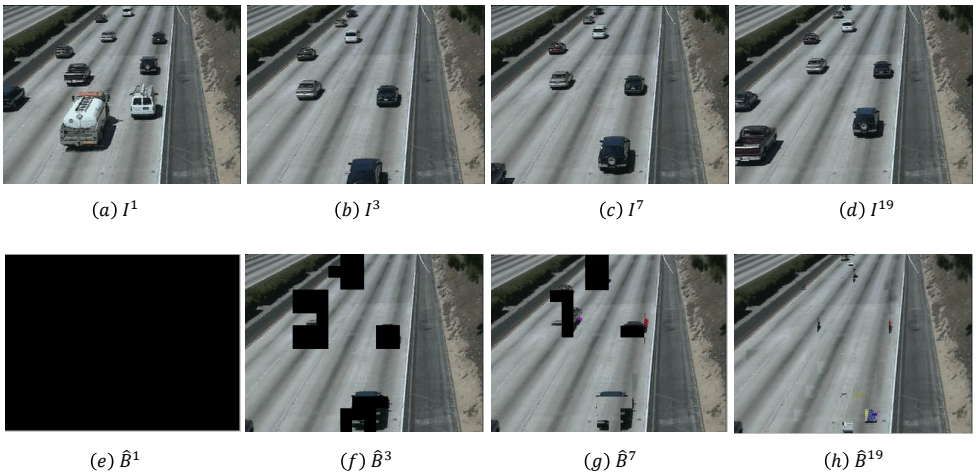
3.2.3. Estimación de movimiento

Una vez hecha la división en bloques, se procede a realizar la estimación de movimiento a nivel de bloque entre dos frames consecutivos I^t e I^{t-1}, usando un algoritmo de *block matching* para determinar si un bloque está (*moving*) o no (*static*) en movimiento. Este algoritmo usa la suma de diferencias absolutas (SAD, *Sum of Absolute Differences*) como medida de *block matching* entre el bloque $b^t_{(i,j)}$ en I^t y su correspondiente bloque en I^{t-1}, para definir el vector de movimiento. Para estimar dicho vector se emplea un área de búsqueda de $\pm N/2$, de forma que si el bloque con mínimo SAD de I^t en el área de búsqueda es menor que el 90 % del SAD para el vector con movimiento nulo, el bloque se establece como *moving* y en caso contrario como *static*. Es importante resaltar que dado que el fondo se presume como una representación estática de la escena, solo bloques de tipo *static* serán candidatos a definir el fondo.

A continuación, para cada bloque "no definido" $\hat{b}^{t-1}_{(i,j)}$ en $\hat{B}^{t-1}$, si su bloque homólogo del frame I^t se establece como *static*, entonces ese bloque se fija como fondo. De lo contrario, el bloque $\hat{b}^{t-1}_{(i,j)}$ en $\hat{B}^{t-1}$ sigue manteniéndose como "no definido" en el instante t, no definiéndose hasta que sea establecido como *static* en instantes temporales fururos. Por ejemplo, en el instante t=*2*, cada bloque de $\hat{B}^2$ se obtiene de $\hat{B}^1$ y el resultado de la estimación de movimiento entre los frames I^1 e I^2. Nótese que en ese instante de tiempo todavía quedan algunos bloques "no definidos" (etiquetados como negro). La Figura 3.2 presenta el funcionamiento anteriormente descrito, mostrando en la segunda fila el proceso de definición del fondo, hasta que en el frame 19 ya no queda ningún bloque "no definido" (ver Figura 3.2h), es decir, en adelante todos los bloques ya están fijados y solo están sujetos a los procesos de actualización (ver Subsección 3.2.5 Actualización de bloques).

3.2.4. Clasificación

Una vez clasificados los bloques entre *static* o *moving*, hay que hacer otra clasificación del bloque del frame actual I^t en una de las 4 categorías posibles llamadas *background* (fondo), *still object* (objeto estático), *illumination change* (cambio de iluminación) y *moving object* (objeto en movimiento). Esta clasificación es necesaria para conocer el tipo de estaticidad o de movimiento que se está tratando. No es lo mismo que un objeto que haya irrumpido en la escena se quede parado (*still object*) o



(a) I^1 (b) I^3 (c) I^7 (d) I^{19}

(e) $\hat{B}^1$ (f) $\hat{B}^3$ (g) $\hat{B}^7$ (h) $\hat{B}^{19}$

Figura 3.2: Ejemplo del proceso de inicialización de fondo: (a)-(f) son los frames originales de la secuencia; (e)-(h) son las sucesivas inicializaciones de fondo.

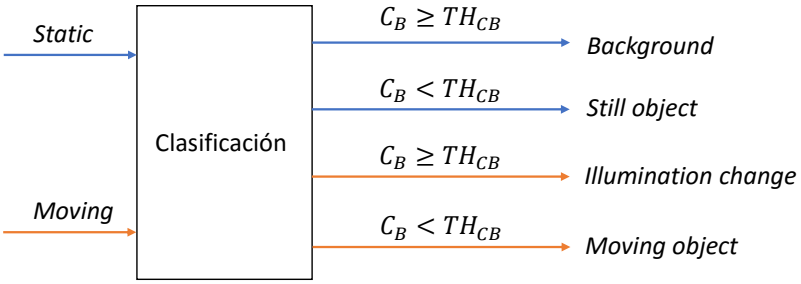


Figura 3.3: Diagrama de flujo de la clasificación de bloques.

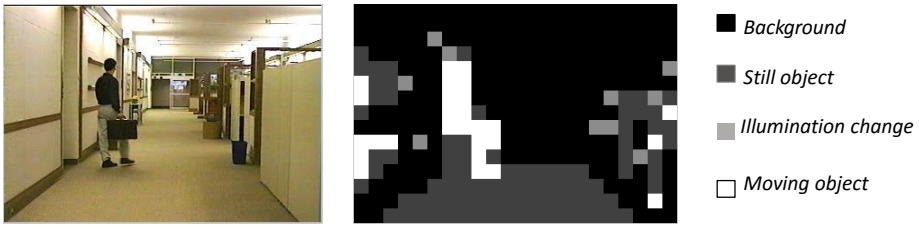
que antes hubiese un objeto, este se vaya y ahora quede visible el fondo (*background*). De la misma manera, cuando se produce un cambio repentino en los valores de una zona del *frame* (bloque *moving*), es necesario saber si ha sido porque ha habido un cambio de iluminación (*illumination change*) o porque ha aparecido un objeto que se mueve (*moving object*). Para realizar la clasificación mencionada se hace uso del coeficiente de correlación $C_B(i,j)$ entre el bloque $b^t_{(i,j)}$ en I^t y el bloque $\hat{b}^t_{(i,j)}$en $\hat{B}^t$:

$$C_B(i,j) = \frac{\sum \left|\left| b^t_{(i,j)} - \mu_{b^t_{(i,j)}} \right| \times \left| \hat{b}^t_{(i,j)} - \mu_{\hat{b}^t(i,j)} \right|\right|}{\sqrt{\sum \left| b^t_{(i,j)} - \mu_{b^t_{(i,j)}} \right|^2} \times \sqrt{\sum \left| \hat{b}^t_{(i,j)} - \mu_{\hat{b}^t(i,j)} \right|^2}}, \tag{3.1}$$

donde μ_b es la media de los píxeles en el bloque (ver Figura 3.3) . El umbral TH_{CB} se ha fijado empíricamente a 0.7 (en [2] no se especifica su valor).

Si se representa cada bloque del frame actual con un nivel de gris distinto en

función de la categoría a la que pertenece, se obtiene la imagen de representación de bloques (ver Figura 3.4b).



(a) I^{35} (b) Representación de bloques del *frame* I^{35}

Figura 3.4: Ejemplo de representación de bloques: (a) Frame de la secuencia; (b) Representación de los bloques según la clasificación obtenida.

3.2.5. Actualización de fondo

Al iniciar el algoritmo, se tenía un *background* con todos los bloques sin definir (etiquetados en negro) y hasta que un bloque no se clasificase como *static* en la etapa de estimación de movimiento, no pasaba a formar parte del fondo. En cuanto un bloque del *background* ya se definiese, a partir de ese mismo instante solo había que ocuparse de actualizarlo. Y la tarea de actualización es lo que se va a explicar en esta Subsección, que consta de cuatro apartados dependiendo del tipo de clasificación que haya obtenido en la etapa 3.2.4 Clasificación.

Mediante un mecanismo de actualización de bloques, cada bloque $\hat{b}^t_{(i,j)}$ en $\hat{B}^t$ se puede actualizar para obtener el correspondiente bloque $\tilde{b}^t_{(i,j)}$ en B^t como se explica a continuación. Tanto los bloques *background* como los *illumination change*, se actualizan como un suavizado temporal (*temporal smoothing*). Por otro lado, los bloques *still object* y los *moving object* se actualizan con reemplazamiento de bloques (*block replacement*).

- *Background*: el bloque $\tilde{b}^t_{(i,j)}$ en B^t se actualiza como:

$$\tilde{b}^t_{(i,j)} = \alpha \cdot \hat{b}^t_{(i,j)} + (1-\alpha) \cdot b^t_{(i,j)} \ , \tag{3.2}$$

  donde α (peso de actualización) se ha establecido como 0.9 en este algoritmo.

- *Still object*: el bloque $\tilde{b}^t_{(i,j)}$ en B^t se actualiza como:

$$\begin{cases} \tilde{b}^t_{(i,j)} = b^t_{(i,j)} & si\ Contador_{still} \geq TH_{still} \\ \tilde{b}^t_{(i,j)} = \hat{b}^t_{(i,j)} & resto \end{cases} , \tag{3.3}$$

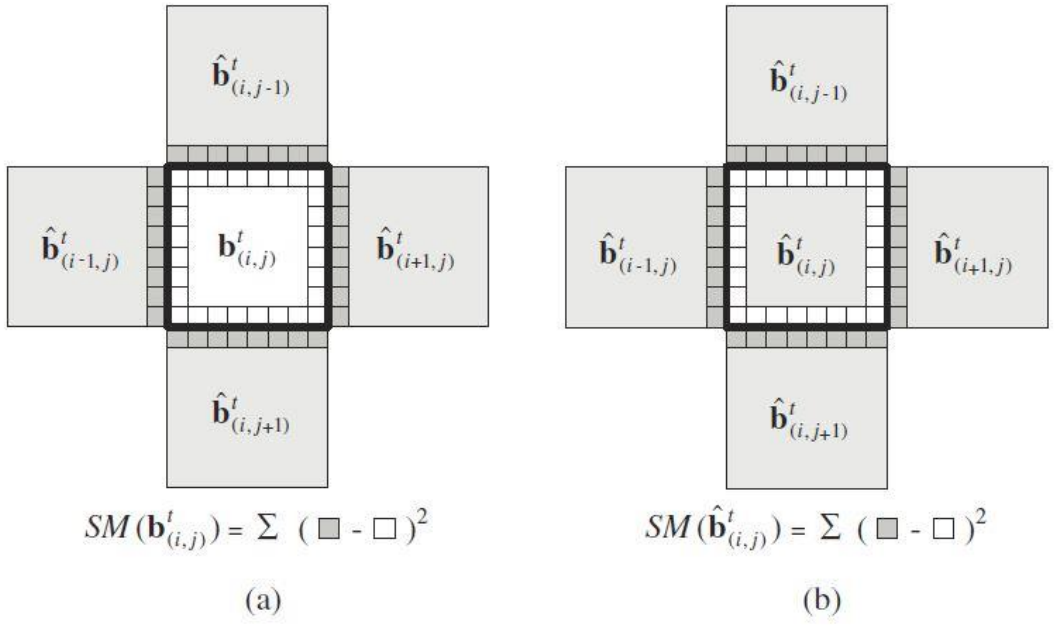


Figura 3.5: Las medidas de bloque $SM\left(b^t_{(i,j)}\right)$ y $SM\left(\hat{b}^t_{(i,j)}\right)$ de un bloque *moving object* en el proceso de actualización. Fuente: [2].

donde $Contador_{still}$ es el número de veces consecutivas que el bloque $b^t_{(i,j)}$ en I^t es clasificado como *still object y* TH_{still} es un umbral temporal que en este algoritmo se ha establecido en 20, es decir, si el bloque $b^t_{(i,j)}$ en I^t ha sido clasificado 20 o más veces como *still object,* éste pasa directamente a formar parte del fondo. Visualmente, esto quiere decir que si un objeto se queda estático durante el suficiente tiempo, éste se integra en el fondo.

- *Illumination change*: el bloque $\tilde{b}^t_{(i,j)}$ en B^t se actualiza de forma similar que en el caso de *background.*

- *Moving object*: el bloque $\tilde{b}^t_{(i,j)}$ en B^t se actualiza como:

$$\begin{cases} \tilde{b}^t_{(i,j)} = b^t_{(i,j)} & si\ SM\left(b^t_{(i,j)}\right) < SM\left(\hat{b}^t_{(i,j)}\right) \\ \tilde{b}^t_{(i,j)} = \hat{b}^t_{(i,j)} & resto \end{cases}, \tag{3.4}$$

donde $SM\left(b^t_{(i,j)}\right)$ y $SM\left(\hat{b}^t_{(i,j)}\right)$ son las medidas de continuidad del bloque $b^t_{(i,j)}$ de I^t embebido en $\hat{B}^t$ y del bloque $\hat{b}^t_{(i,j)}$ también embebido en $\hat{B}^t$ (ver Figura 3.5) . Esta medida es muy usada en múltiples algoritmos gracias a sus buenos resultados. $SM\left(b^t_{(i,j)}\right)$ se define como la suma de las diferencias al cuadrado de las cuatro fronteras del bloque $b^t_{(i,j)}$ de I^t con las fronteras de los cuatro bloques

vecinos $\hat{b}^t_{(i-1,\,j)}$, $\hat{b}^t_{(i+1,\,j)}$, $\hat{b}^t_{(i,\,j-1)}$ y $\hat{b}^t_{(i,\,j+1)}$ en $\hat{B}^t$ (ver Figura 3.5a). Formalmente, se define como:

$$SM\left(b^t_{(i,j)}\right) = \sum_{b=0}^{N-1}\left(\hat{B}^t_{(iN-1,\,jN+b)} - I^t_{(iN,\,jN+b)}\right)^2 + \sum_{b=0}^{N-1}\left(\hat{B}^t_{(iN+N,\,jN+b)} - I^t_{(iN+N-1,\,jN+b)}\right)^2 + \sum_{a=0}^{N-1}\left(\hat{B}^t_{(iN+a,\,jN-1)} - I^t_{(iN+a,\,jN)}\right)^2 + \sum_{b=0}^{N-1}\left(\hat{B}^t_{(iN+a,\,jN+N)} - I^t_{(iN+a,\,jN+N-1)}\right)^2. \tag{3.5}$$

De la misma manera $SM\left(\hat{b}^t_{(i,j)}\right)$ se define formalmente como (ver Figura 3.5b):

$$SM\left(b^t_{(i,j)}\right) = \sum_{b=0}^{N-1}\left(\hat{B}^t_{(iN-1,\,jN+b)} - \hat{B}^t_{(iN,\,jN+b)}\right)^2 + \sum_{b=0}^{N-1}\left(\hat{B}^t_{(iN+N,\,jN+b)} - \hat{B}^t_{(iN+N-1,\,jN+b)}\right)^2 + \sum_{a=0}^{N-1}\left(\hat{B}^t_{(iN+a,\,jN-1)} - \hat{B}^t_{(iN+a,\,jN)}\right)^2 + \sum_{b=0}^{N-1}\left(\hat{B}^t_{(iN+a,\,jN+N)} - \hat{B}^t_{(iN+a,\,jN+N-1)}\right)^2. \tag{3.6}$$

Con esta medida podemos "camuflar" un objeto del *foreground* que no queremos que forme parte del *background,* ya que sabemos que ese objeto va a tener una discontinuidad más alta con el fondo que el propio fondo. Una vez analizados todos los *frames*, se elige la última versión del *background* actualizado que es la inicialización que se estaba buscando.

3.3. Algoritmo basado en clasificaciones a nivel de píxel

3.3.1. Estructura general

El segundo algoritmo de BI implementado es el propuesto en [3] y realiza un análisis *online* espacio-temporal de la secuencia a nivel de pixel. Globalmente, el algoritmo realiza un modelado estadístico de la variación del nivel de luminancia de los píxeles, con el objetivo de caracterizar las variaciones asociadas al fondo de la escena. Además, para realizar la reconstrucción del fondo se definen unas posiciones de referencia dinámicas y estáticas en la escena para distinguir entre dos tipos de movimiento: objeto que ocluye el fondo y objeto que, al irse, permite que se vea el fondo. El algoritmo puede estructurarse en 3 etapas (ver Figura 3.6) :

1. Filtrado de píxeles salientes: En esta estapa se detectan los píxeles que tienen cambios importantes en la secuencia, pues sus variaciones no caracterizan el fondo.

2. MLS (*Maximum Liklihood Set*): En esta etapa se contabilizan las variaciones de niveles de luminancia entre frames consecutivos, denominadas eventos. Estos eventos se contabilizan en aquellos píxeles que se hayan determinado como no

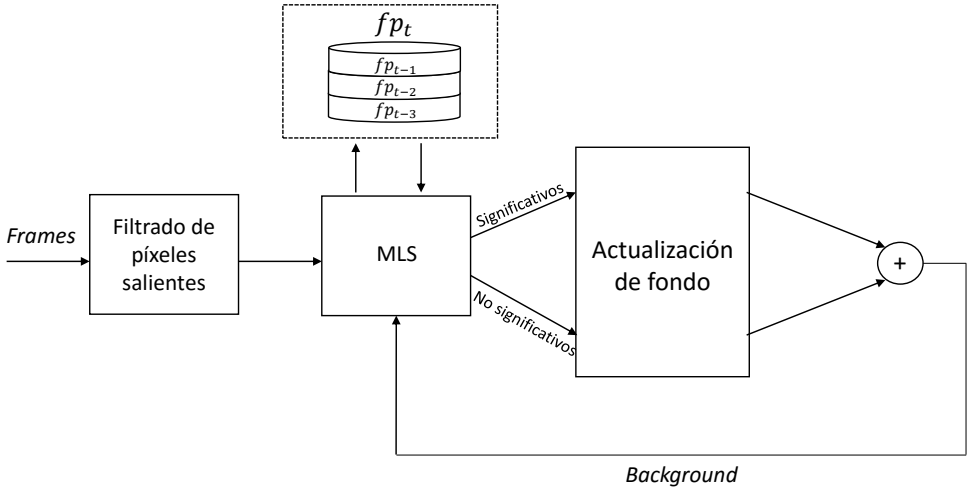


Figura 3.6: Esquema general de funcionamiento del algoritmo [3].

salientes en la etapa anterior y se acumulan *frame* a *frame* para modelar la distribución de variaciones entre niveles de gris del fondo y formar la tabla MLS. La pertenencia o no a la tabla MLS define la tipología de píxeles analizados en "Significativos" o "No Significativos".

3. Actualización del fondo: El fondo se fija inicialmente con el primer *frame* y en esta etapa se llevan a cabo actualizaciones del mismo a partir de la pertenencia o no MLS. El petenecer a la MLS determina que el pixel es de fondo (variación no significativa), mientras que la no pertenencia define una variación característica (variación significativa).

3.3.2. Filtrado de píxeles salientes

El algoritmo empieza detectando los píxeles salientes *frame* a *frame*. Estos píxeles son los que representan un cambio significativo en su valor de un *frame* a otro. Para ello, se diseña un filtro de saliencia inter-frame (*ISF, Inter-frame Saliency Filter*) que formalmente se define como sigue:

$$D_n = \sum_{c=1}^{3} (I_{n,c} - I_{n-1,c})^2 \quad , \tag{3.7}$$

$$t_n = \frac{1}{N\,M} \sum_{i=1}^{N} \sum_{j=1}^{M} D_n(i,\, j) \quad , \tag{3.8}$$

$$F_n(i,\, j) = \begin{cases} 0 & si\ D_n(i,\, j) > t_n \\ 1 & resto \end{cases}, \tag{3.9}$$

donde D_n es la diferencia píxel a píxel entre el *frame* I^t y el frame anterior I^{t-1} (el tamaño del frame es $N \times M$), c es el canal de color (RGB), t_n es el umbral de saliencia y finalmente, F_n es el filtro de saliencia que se buscaba, donde 0 significa que el pixel es saliente y 1 lo contrario.

3.3.3. Construcción de tabla de eventos y MLS

A continuación, usamos el resultado del filtrado de la etapa anterior para registrar las variaciones de intensidad de valor de pixel p (en el *frame* anterior) a un valor de pixel x (en el *frame* actual) en aquellos píxeles detectados como no salientes. Esta variación se denomina evento $E(x|p)$. Registrando todos los eventos *frame* a *frame*, podemos construir un histograma de eventos para cada nivel p y cada nivel x a través de una matriz $L \times L$, siendo $L = 0..,255$ el número de niveles de gris. Formalmente, este histograma de eventos se define como:

$$H_p = \sum E(x|p),\ x,\ p = 0, 1, ..., L-1, \tag{3.10}$$

donde $E(x|p)$=1 define que un pixel que ha pasado de valer p a valer x. Normalizando este histograma obtenemos una función densidad de probabilidad que se define como:

$$f_p = H_p(x) \times \left(\sum_{x=0}^{L-1} H_p(x) \right)^{-1}, \tag{3.11}$$

donde f_p es la probabilidad de $E(x|p)$. En la Figura 3.7 se puede ver un ejemplo de f_p para un valor de p=120. Se puede observar que la gráfica está centrada en el valor de intensidad 120 con un máximo en ese mismo valor. Esto quiere decir que lo más probable es que un pixel de valor p=120, siga valiendo 120 en el *frame* siguiente. Según x se aleje de 120, la probabilidad de que se produzca ese evento ($E(x|120)$) es cada vez menor.

A continuación, a partir de las distribuciones de eventos, f_p, se construye la tabla MLS [29] (*Maximum Likelihood Set*) que va a recoger de forma binaria las variaciones del *background*. La tabla *MLS* tiene el mismo tamaño que la tabla f_p ($L \times L$) y para cada valor de p se define de la siguiente manera:

$$MLS_p = \{x \mid f_p(x) \neq 0\}. \tag{3.12}$$

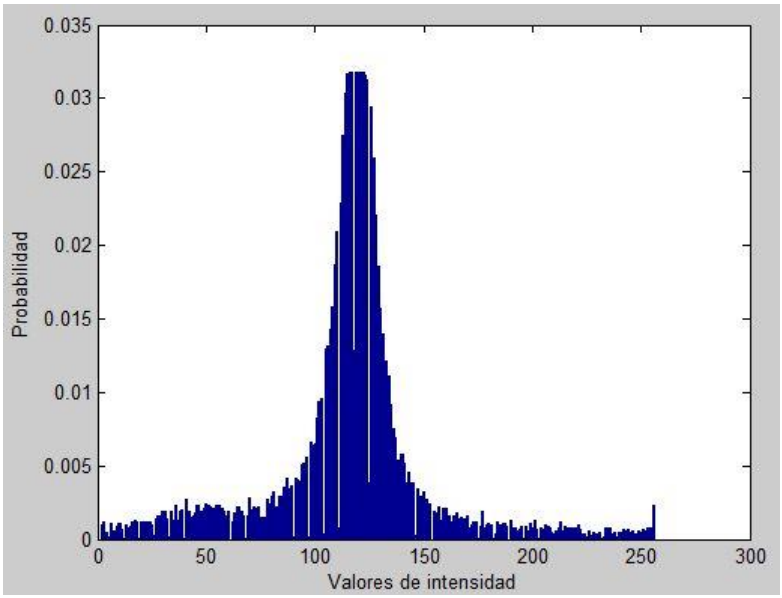


Figura 3.7: Distribución f_p de las variaciones del nivel p=120.

Lógicamente, hay 256 (L=256) MLS_p y cada una de ellas es una *look-up table* para las variaciones de intensidad del *background*. Sea $B(i_0, j_0) = p$ un pixel del *background* y $C(i_0, j_0) = x$ el mismo pixel del *frame* actual. Esta tabla MLS, la vamos a utilizar para determinar en instantes futuros de tiempo, si un pixel ha sufrido un cambio "Significativo" o "No Significativo". Si x pertenece al MLS_p correspondiente, esa variación de pixel se clasifica como "No Significativa" o por el contrario como "Significativa".

3.3.4. Actualización de fondo en los píxeles "No Significativos"

Cuando ya se sabe qué píxeles han sufrido un cambio "Significativo" y cuales uno "No Significativo", aplicamos la estrategia que se sigue para las variaciones "No-Significativas" de los píxeles, es una actualización gradual del fondo que se rige por la siguiente ecuación:

$$B_t(i, j) = B_{t-1}(i, j) + K \times [C_t(i, j) - B_{t-1}(i, j)]\,, \tag{3.13}$$

donde B_{t-1} es el fondo en el instante anterior, C_t es el frame actual de la secuencia y B_t es el fondo actual. K es el factor de ganancia que se establece como $f_p(x)$ (Ec. 3.11) donde $p = B_{t-1}(i, j)$ y $x = C_t(i, j)$.

3.3.5. Actualización de fondo en los píxeles "Significativos"

Las variaciones "Significativas" son causadas por un nuevo valor de pixel que es distinto del modelo de fondo actual. Hay dos posibilidades para este nuevo valor: (1) es por un objeto en movimiento o (2) es porque el frente se ha ido y ha dejado visible el fondo. El reto ahora consiste en saber cuál de los dos casos se ha producido. La clave para este reto es la continuidad espacial de este nuevo valor del pixel más que de la cantidad de tiempo que lleve expuesto. Un criterio muy popular para calcular esta continuidad incluye la similaridad del color y la integración de la textura.

La estrategia que se va a seguir en este algoritmo [3] necesita localizar algunos píxeles de referencia estáticos (*SRPs*, *Static Reference Pixels*) para que se mantenga un estado estable en la escena. Sin embargo, esta verificación a través de los SRPs fallará si un objeto en movimiento tiene una similaridad con el fondo mayor que el propio fondo que estaba ocluido. Por otro lado, un objeto en movimiento también tiene continuidad consigo mismo y de este concepto surge la necesidad de verificación de píxeles de referencia dinámicos (*DRPs, Dynamic Reference Pixels*) que chequea la continuidad espacial de un objeto en movimiento. Los pasos a seguir para implementar esta estrategia son:

1. Inicializar los conjuntos SRP y DRP de la siguiente manera:

$$SRPs = \left\{ \theta(i,\, j) \mid \frac{NS(i,\, j)}{n} \geq \gamma,\ 0 \leq \gamma \leq 1 \right\}, \tag{3.14}$$

$$DRPs = \emptyset, \tag{3.15}$$

   donde n es el número de frames analizados hasta el momento, *NS(i, j)* es el número de veces que el píxel $\theta(i,\, j)$ ha sido considerado como "No Significativo", γ es la relación de estabilidad que para este algoritmo se ha fijado a 0.7.

2. Medir la continuidad espacial del pixel "Significativo". Para ello se emplea un criterio de similaridad de color.

$$SM(\theta) = min \left\{ \sqrt{\sum_{c=1}^{3} [\theta_c(i,\, j) - \theta_c^{NB}(i,\, j)]^2} \right\}, \quad \theta^{NB}(i,\, j) \in SRPs, \tag{3.16}$$

   donde $\theta(i_0,\, j_0)$ es el pixel clasificado como "Significativo" cuyo vecindario es $\theta^{NB}(i,\, j)$ (en [3] no se especificaba cómo era el vecinadario y para la implementación se ha tomado un vecindario de 3×3) y c es el canal de color. Siendo X el valor de pixel del *background* actual e Y el nuevo valor de pixel del frame, sus respectivas dis-similaridades de color en el vecindario $\theta^{NB}(i,\, j)$ son *SM(X)*

y SM(Y).

3. Siendo $\hat{B}$ el *background* actualizado y U el objeto en movimiento:

$$si\, Y \in \left\{\theta^{NB}(i,\, j) \cap DRPs\right\},\ \hat{B} = X\, y\, U = Y. \tag{3.17}$$

Esto quiere decir que si el pixel del *frame* actual pertenece al conjunto de *DRPs* del vecindario, este pixel que se está analizando es de un objeto en movimiento y no queremos que pase a formar parte del fondo, por eso nos quedamos con el pixel que había hasta ahora en el fondo, sin sustituirlo tal y como describe la Ec. 3.17. Es el caso en el que un objeto en movimiento ha entrado a la escena ocluyendo el fondo.

$$si\, X \in \left\{\theta^{NB}(i,\, j) \cap DRPs\right\},\ \hat{B} = Y\, y\, U = X. \tag{3.18}$$

Si por el contrario, el pixel X (pixel del fondo) pertenece al conjunto de *DRPs* del vecindario, quiere decir que el pixel que se está analizando es del fondo y lo que ha pasado es que antes había un objeto que se ha ido y ahora deja ver el fondo. En este caso sí queremos actualizar ese pixel del fondo tal y como describe la Ec. 3.18.

$$\begin{aligned} &si\, X,\, Y \notin \left\{\theta^{NB}(i,\, j) \cap DRPs\right\}, \\ &\hat{B} = argmin_{\theta}\{SM(X),\, SM(Y)\}, \\ &U = argmax_{\theta}\{SM(X),\, SM(Y)\}. \end{aligned} \tag{3.19}$$

Como último caso, si ni el pixel Y ni el pixel X pertenecen al *DRP* del vecindario, se integrará en el *background* el pixel que tenga menos dis-similaridad con el vecindario. De la msima manera, se considerará como objeto en movimiento el pixel que tenga más dis-similaridad con su vecindario (ver Ec. 3.19).

4. Actualizar los *SRPs* y los *DRPs*. Para preparar el análisis del siguiente pixel, hay que actualizar los puntos de referencia estáticos (*SRPs*) y dinámicos (*DRPs*) con los píxeles que en este análisis se han considerado *background* u objeto en movimiento respectivamente como muestra la Ec. 3.20.

$$SRPs = \hat{B} \cup SRPs,\ DRPs = U \cup DRPs. \tag{3.20}$$

5. Repetir los pasos del 2 al 4 hasta que se hayan analizado todos los píxeles "Significativos" del frame actual. Depués, hay que volver a empezar desde el paso 1.

Finalmente, se elige la última versión del *background* actualizado que es la inicialización que se estaba buscando.

Capítulo 4

Trabajo experimental

En este capítulo se va a evaluar el rendimiento de los algoritmos implementados frente a algoritmos relevantes del estado del arte.

Para tal fin, el capítulo se encuentra dividido en las siguientes secciones: marco de evaluación (sección 4.1) donde se detalla la generación del *dataset* (subsección 4.1.1) y las métricas utilizadas para la evaluación (subsección 4.1.2), y evaluación comparativa (sección 4.2) donde se comparan los resultados obtenidos para cada algoritmo.

4.1. Marco de evaluación

4.1.1. Generación de *dataset*

El *dataset* utilizado se ha construido en base a 4 categorías o 4 retos que se han considerado interesantes a la hora de realizar una inicialización de fondo a partir de una secuencia de vídeo. Es importante resaltar que un *dataset* completo es algo que no existe en la literatura. Se han generado 10 secuencias para cada categoría, todas ellas extraídas de *datasets* públicos de vídeo-vigilancia en el estado del arte. Estas 4 categorías son:

- ***Baseline***: Esta categoría la forman secuencias sencillas, con poca densidad de objetos en la escena y sin objetos estáticos.
- ***Clutter***: En esta categoría se han incluido secuencias con gran movimiento en la escena que provoca la oclusión continuada de ciertas zonas del fondo y en consecuencia, la inicialización del mismo se ve comprometida.
- ***Low framerate***: Esta categoría consta de secuencias similares a las de baseline, pero con un framerate bajo. Tener un *framerate* bajo, puede afectar a los

algoritmos a la hora de detectar movimiento en la escena, pues los objetos de las secuencias sufren grandes deslazamientos bruscos entre *frames*.

- ***Static objects***: Esta categoría se centra en secuencias que contienen objetos estáticos ocluyendo el fondo durante más del 50 % de su duración. Los objetos estáticos dificultan la inicialización del fondo debido a que invalidan la asumción de fondo como aquella representación de la escena visualmente dominante.

Las secuencias seleccionadas para cada categoría han sido seleccionadas de *datasets* públicos para vídeo-vigilancia: MALL [1], MVI[2], CAVIAR[3], CUHK[4], ATON[5], CVRL[6], IITK[7], CD14[8], APIDIS[9], AVSS07[10], PBI[11], Wallflower[12], PETS09[13], SAIVT[14], LOST[15], LIMU[16] y I2R[17].

En las Tablas 4.1, 4.2, 4.3 y 4.4 se muestran los detalles de todas las secuencias seleccionadas para el *dataset* junto con los respectivos *ground-truths* de cada una. Un *ground-truth* es el fondo ideal de la secuencia que se usa como referencia para la evaluación, por eso es esencial construirlos. Cada tabla cuenta con el nombre de la secuencia, el *dataset* de donde ha sido extraída, el número de frames de la secuencia original, el número de frames seleccionados para aislar el reto que se quería analizar y por último, la resolución de los frames. Además, la tabla de la categoría *Low framerate* cuenta con un campo de fps que especifica el *framerate* de cada secuencia, ya que el del resto de categorías es de 25 fps. En algunos casos, se ha escogido una misma secuencia original para dos retos distintos, pero se han recortado fragmentos de *frames* distintos donde cada uno alberga el reto que se quiere analizar por separado.

[1] http://www.eecs.qmul.ac.uk/~ccloy/downloads_mall_dataset.html
[2] http://homepages.ed.ac.uk/cblair2/csvt/
[3] http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/
[4] http://www.ee.cuhk.edu.hk/~jshao/CUHKcrowd_files/cuhk_crowd_dataset.htm
[5] http://cvrr.ucsd.edu/aton/shadow/index.html
[6] https://www.ecse.rpi.edu/~cvrl/humanbody/
[7] http://www.cse.iitk.ac.in/users/vision/traffic-datasets/dataset1/dataset1.html
[8] http://changedetection.net/
[9] http://sites.uclouvain.be/ispgroup/index.php/Softwares/APIDIS
[10] www.eecs.qmul.ac.uk/~andrea/avss2007.html
[11] www.diegm.uniud.it/fusiello/demo/bkg/
[12] research.microsoft.com/en-us/um/people/jckrumm/wallflower
[13] www.cvg.rdg.ac.uk/PETS2009/index.html
[14] https://wiki.qut.edu.au/display/saivt/SAIVT-SoftBio+Database
[15] http://lost.cse.wustl.edu/
[16] limu.ait.kyushu-u.ac.jp/dataset/en/
[17] http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html

Tabla 4.1: Detalles de las secuencias de la categoría *Baseline*

Nombre	*Dataset*	Fr. originales	Fr. utilizados	Resolución
1. bankst	MVI	1-31873	1100-1610	720×464
2. CAVIAR1	CAVIAR	1-726	325-724	384×256
3. CAVIAR2	CAVIAR	1-1500	900-1360	344×288
4. Square	CUHK	1-90425	40000-40400	720×576
5. Highway	ATON	1-500	1-470	320×240
6. HighwayI	ATON	1-440	1-440	320×240
7. HighwayII	ATON	1-500	1-500	320×240
8. HumanBody2	CVRL	1-899	70-470	320×240
9. Gate Traffic	IITK	1-6539	1100-1465	360×288
10. pedestrians	CD14	1-1100	300-700	360×240

Tabla 4.2: Detalles de las secuencias de la categoría *Clutter*

Nombre	*Dataset*	Fr. originales	Fr. utilizados	Resolución
1. camera1-ps	APIDIS	1-190000	500-900	400×300
2. camera5-ps	APIDIS	1-190000	500-900	400×300
3. AVSS_AB_EVAL	AVSS07	1-32626	6000-6460	720×576
4. Board	PBI	0-227	0-227	200×164
5. bootstrap	Wallflower	1-3456	772-1172	160×120
6. Crowd	PETS09	1-200	1-200	768×576
7. Foliage	PBI	1-394	1-394	200×144
8. PeopleAndFoliage	PBI	1-349	1-350	320×240
9. Campus_test	SAIVT	1-90020	1-400	704×576
10. Snellen	PBI	1-322	1-322	144×144

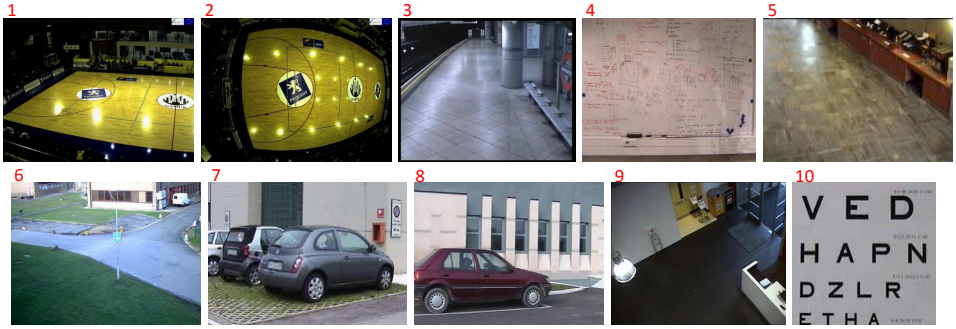
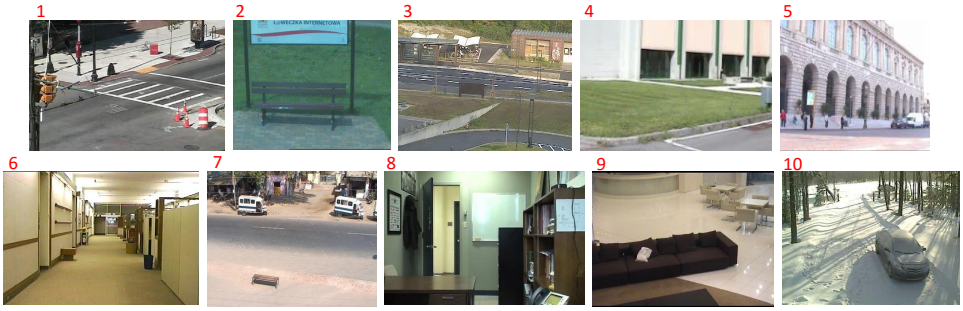


Tabla 4.3: Detalles de las secuencias de la categoría *Low framerate*

Nombre	*Dataset*	Fr. originales	Fr. utilizados	fps	Resolución
1. JapMall	LOST	1-60	1-60	10	640×480
2. CruceNevado	LOST	1-687	1-400	5	640×480
3. Payot	LOST	1-3458	600-1000	1	320×240
4. Intersection	LIMU	1-5000	100-500	3	320×240
5. Plaza Mayor	LOST	1-800	230-630	5	640×480
6. mall	Mall	1-623	87-450	5	640×480
7. ShoppingMall	I2R	1-258	1-258	5	320×256
8. tramCrossroad	LOST	1-420	1-420	1	640×350
9. tunnelExit	LOST	1-500	1-450	3	700×440
10. turnpike	LOST	1-400	1-400	5	320×240

Tabla 4.4: Detalles de las secuencias de la categoría *Static objects*

Nombre	*Dataset*	Fr. originales	Fr. utilizados	Resolución
1. BusPolaco	LOST	1-2500	1550-1923	352×288
2. abandonedBox	CD14	1-4500	769-1169	432×288
3. BusStopMorning	LIMU	1-5000	1885-2200	320×240
4. CaVignal	PBI	1-257	1-257	200×136
5. granguardia	PBI	1-500	1-380	200×174
6. HallAndMonitor	COST 211	1-296	1-296	352×240
7. Gate Traffic	IITK	1-6539	1400-1700	288×360
8. office	CD14	1-2100	1600-2049	360×240
9. sofa	CD14	1-2750	1672-2102	320×240
10. winterDriveway	CD14	1-2500	1550-2000	320×240



4.1.2. Métricas de evaluación

Esta Subsección describe los parámetros que se han usado para la evaluación de los algoritmos. Estos parámetros se han propuesto inicialmente en [30], que se basan en las diferencias en niveles de grises entre la incialización y *ground-truth*, las diferencias de color o el número de píxeles erróneos con un margen de error en cada pixel de valor 20.

- **Average Gray-level Error (AGE)**: es la media de la diferencia absoluta de los niveles de gris entre los *ground-truths* y los fondos generados por los algoritmos. El valor puede estar en el rango [0,L-1] donde L es valor máximo de niveles de gris, típicamente L=256. Cuanto más bajo sea el AGE, mejor es el fondo generado.

- **Percentage of Error Pixeles (pEPs)**: un pixel erróneo es un pixel del fondo generado cuyo valor difiere del valor del píxel homólogo en el *ground-truth* más de un umbral (en el script de evaluación se ha usado th=20). El pEPs es la relación entre el número de píxeles erróneos y el número total de píxeles de la

imagen. Su valor varía entre [0,1]. Cuanto más bajo sea el pEPs, mejor es el fondo generado.

- **Percentage of Clusteres Error Pixeles (pCEPs)**: un grupo (*cluster*) de píxeles erróneos (CEP) se define como cualquier pixel erróneo en el que sus vecinos de conectividad 4 también son píxeles erróneos. pCEPs es la relación entre el número de CEPs y el número total de píxeles de la imagen. Su valor varía entre [0,1]. Cuanto más bajo sea el pCEPs, mejor es el fondo generado.

- **Peak-Signal-to-Noise-Ratio (PSNR)**: se define como:

$$PSNR = 10log_{10}\left(\frac{(L-1)^2}{MSE}\right), \tag{4.1}$$

  donde L es el valor máximo de niveles de gris, típicamente L=256 y MSE es el error cuadrático medio entre el fondo generado y el *ground-truth* correspondiente. El PSNR se expresa en decibelios y cuanto mayor sea, mejor es el fondo generado.

- **Multi-Scale Structural Similaruty Index (MS-SSIM)**: es un parámetro de medida propuesto en [31] que compara características de estructura para determinar la calidad de la imagen. Su valor varía entre [0,1]. Cuanto mayor sea el MSSSIM, mejor es el fondo generado.

- **Color image Quality Measure (CQM)**: este es un parámetro basado en una transformación reversible entre el espacio de colores YUV y con el PSNR calculado en cada banda YUV individual. Como con el PSNR, el valor del CQM se expresa en decibelios y cuanto mayor sea el CQM, mejor es el fondo generado.

4.2. Evaluación comparativa

En esta sección se presenta el análisis de 4 algoritmos: [2] y [3] implementados para este TFG y [4] y [5] que ya se han proporcionado implementados para finalmente evaluar los cuatro sobre el *dataset* generado en este TFG. Este análisis se va a realizar para cada categoría en las subsecciones 4.2.1, 4.2.2, 4.2.3 y 4.2.4 respectivamente.

4.2.1. Resultados de la categoría *Baseline*

Los resultados medios en esta categoría se presentan en la Tabla 4.5. Se puede observar que los algoritmos BIB[2], DCT [5] y RMR [4] tienen un rendimiento similar

Tabla 4.5: Resultados promedio de secuencias *Baseline*.

Algoritmo	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BIB [2]	4.24	3.05	0.77	0.95	31.07	44.53
BIP [3]	5.11	4.14	1.33	0.92	27.72	43.65
RMR [4]	3.65	1.98	0.77	0.96	32.83	45.71
DCT [5]	3.55	2.36	0.38	0.97	33.95	47.68

Tabla 4.6: Resultados promedio de secuencias *Clutter*.

Algoritmo	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BIB [2]	13.74	13.45	10.05	0.84	25.27	34.51
BIP [3]	22.71	33.33	26.68	0.78	20.79	33.80
RMR [4]	9.41	6.24	4.21	0.91	26.27	35.26
DCT [5]	8.17	4.97	3.31	0.94	27.84	36.49

y superior al 95 %, lo cual indica que los fondos se han generado con bastante calidad (ver Figura 4.1). Sin embargo, BIP [3] tiene un rendimiento algo más bajo debido a que las técnicas utilizadas para detectar píxeles salientes no han funcionado bien, ya que algunos píxeles de objetos en movimiento han sido integrados en el fondo, es decir, se han detectado como no significativos píxeles que sí lo eran y eso es lo que ha producido los errores. En la Figura 4.1 se puede observar esta apreciación.

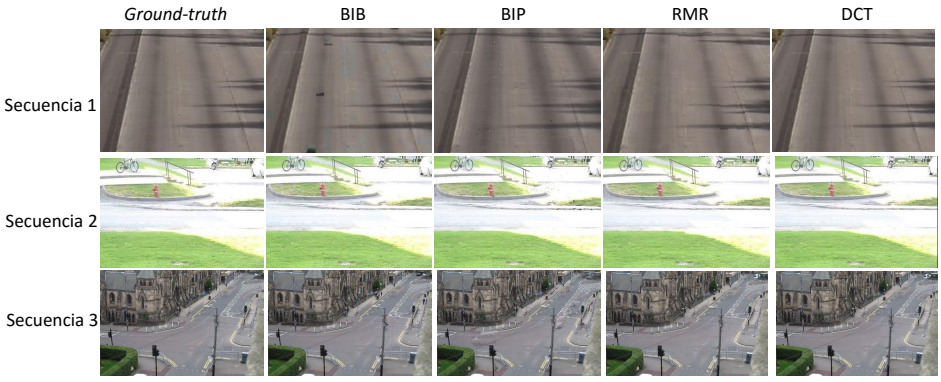


Figura 4.1: Algunos resultados de los algoritmos BIB [2], BIP [3], RMR [4] y DCT [5] para la categoría *Baseline*.

4.2.2. Resultados de la categoría *Clutter*

Los resultados medios en esta categoría se presentan en la Tabla 4.6. El algoritmo con mejores resultados es DCT [5] con un rendimiento en MSSSIM cercano a 95 %. Este buen comportamiento se debe a que sus técnicas de continuidad espacial son

Tabla 4.7: Resultados promedio de secuencias *Low framerate*.

Algoritmo	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BIB [2]	5.37	3.94	1.76	0.93	29.30	41.35
BIP [3]	4.64	2.66	1.07	0.95	30.47	43.00
RMR [4]	4.15	1.69	0.66	0.96	31.34	43.07
DCT [5]	3.39	1.51	0.39	0.97	32.77	45.34

robustas a las oclusiones continuadas del fondo de escena. Por otro lado, de nuevo el peor algoritmo es BIP [3] que realiza un análisis a nivel de pixel que no resulta eficaz (al igual que en la categoría *Baseline*), llegando a obtener un 30 % de píxeles erróneos (pEPs).La Figura 4.2 muestra los resultados obtenidos para la categoría de *Clutter* por cada algoritmo.

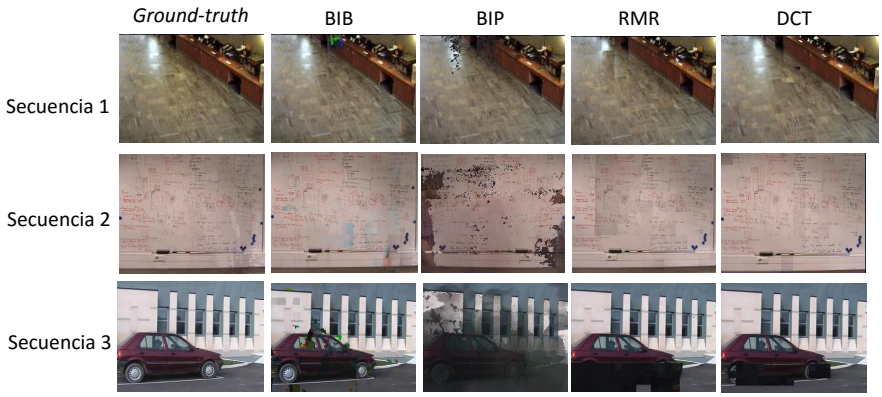


Figura 4.2: Algunos resultados de los algoritmos BIB [2], BIP [3], RMR [4] y DCT [5] para la categoría *Clutter*.

4.2.3. Resultados de la categoría *Low framerate*

Para la categoría de *Low framerate* (ver Tabla 4.7), el algoritmo que mejor funciona es RMR [4], obteniendo un rendimiento cercano al resto de algoritmos. Cabe destacar que el algoritmo a nivel de pixel opera correctamente en esta categoría, lo cual lleva a pensar que la calidad de la estimación de movimiento de BIP [3] depende del *framerate* de las secuencias. Nótese que BIP [3] lleva a cabo una diferencia entre frames adyacentes para calcular el movimiento y que al haber mayores cambios entre frames, se estima mejor. Esta vez el algoritmo que peor ha funcionado ha sido el BIB [2]. Aunque se hayan obtenido buenas inicializaciones, este algoritmo falla ligeramente en la medida de correlación entre bloques propuesta, ya que no ha sido capaz de rechazar completamente los objetos del frente y ha admitido que el bloque que abarca

Tabla 4.8: Resultados promedio de secuencias *Static objects*.

Algoritmo	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BIB [2]	5.42	3.80	1.75	0.91	26.25	40.91
BIP [3]	8.00	7.07	3.66	0.85	22.19	38.03
RMR [4]	3.43	1.73	0.70	0.97	31.99	43.49
DCT [5]	5.13	4.31	2.59	0.93	28.33	42.35

el objeto del frente "encaja" mejor en el fondo (ver Figura 4.3).

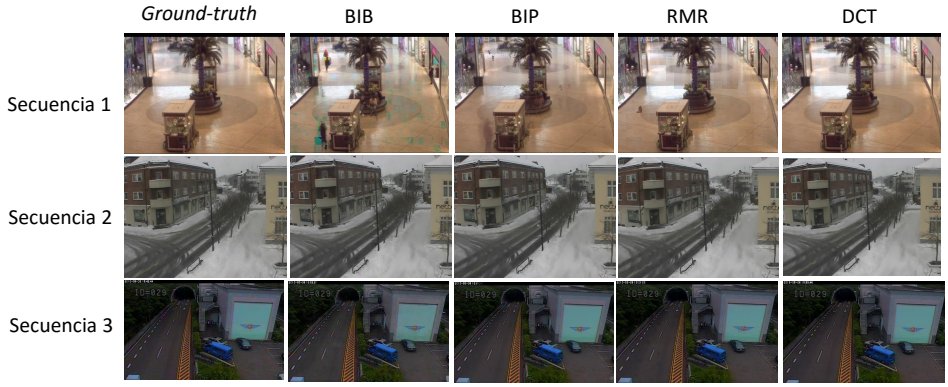


Figura 4.3: Algunos resultados de los algoritmos BIB [2], BIP [3], RMR [4] y DCT [5] para la categoría *Low framerate*.

4.2.4. Resultados de la categoría *Static objects*

Por último, para la categoría de *Static objects* se muestran los resultados de los 4 algoritmos en la Tabla 4.8. RMR [4] es el algoritmo que mejor funciona frente a objetos estáticos, pues está específicamente diseñado para lidiar con este problema. En cuanto al algoritmo con peores resultados es de nuevo BIP [3]. Al igual que en la categoría *Clutter*, en este tipo de escenarios la visibilidad del fondo se ve reducida porque hay objetos que lo ocluyen y ni el filtrado de píxeles salientes, ni la detección de cambios "Significativos", ni las medidas de continuidad de espacial consiguen solventar este reto, como se puede observar en la Figura 4.4, donde se integran píxeles de los objetos estáticos en el fondo.

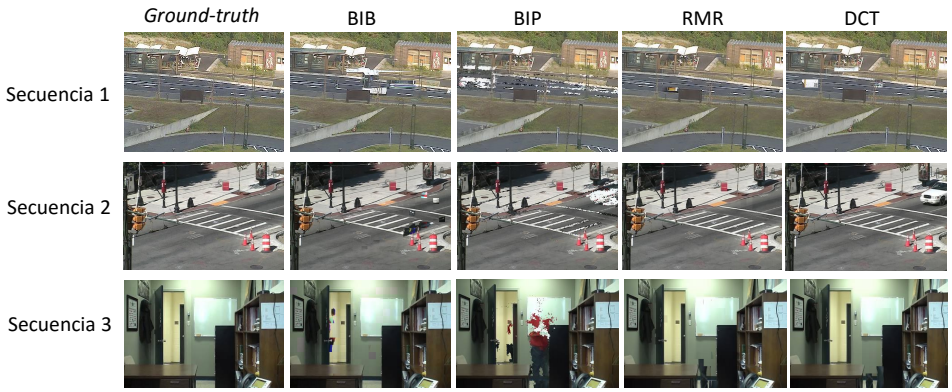


Figura 4.4: Algunos resultados de los algoritmos BIB [2], BIP [3], RMR [4] y DCT [5] para la categoría *Static objects*.

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

En este TFG se han implementado dos algoritmos de inicialización de fondo de cierta relevancia en el estado del arte y se ha generado un *dataset* para evaluar este tipo de algoritmos frente a diversas problemáticas.

En primer lugar, se realizó un estudio del estado del arte para conocer el problema de la inicialización de fondo y las aproximaciones presentes en la literatura. De este estudio, se obtuvo la información acerca de la necesidad de que un buen algoritmo de inicialización de fondo contenga no solo un análisis temporal, sino también un análisis espacial que cuente con medidas de continuidad con el vecindario. Esta necesidad, se debe a que en un entorno cualquiera no se puede asegurar que el fondo de la escena es aquello temporalmente dominante. En consecuencia y dado que se querían implementar varios algoritmos de la literatura, se seleccionaron dos algoritmos basados en un análisis espacio-temporal de la escena.

Tras implementar los algoritmos, se decidió evaluar sus capacidades. Debido a la falta de un *dataset* completo en la literatura, se generó uno propio con 4 categorías y 10 secuencias en cada una. Al observar los resultados obtenidos, se observó que las categorías que mejor rendían eran siempre *Baseline* y *Low framerate*. La única diferencia entre estas dos categorías era el *framerate*, el de *Baseline* tenía 25 fps y el de *Low framerate* podía variar entre 1, 3 o 5 fps. Sin embargo, las categorías de *Static objects* y *Clutter* por regla general se obtenían siempre peores resultados de ellas. En ambas categorías se aislaron retos distintos, pero hay una cosa que tienen en común y es que ambas mantenían ciertas partes del *background* ocluido durante muchos frames, unas con objetos en movimiento (*Clutter*) y otras con objetos estáticos (*Static objects*). Esto solo confirma que la tarea de BI se complica cuando la visibilidad

del fondo que se quiere reconstruir se ve reducida por objectos que lo ocluyen, es decir, no hay ningún algoritmo que funcione bien en todo tipo de escenarios, pero a la vista de los resultados obtenidos, de la gran mayoría de las secuencias se obtenía una muy buena reconstrucción. Por último, el parámetro MSSSIM había veces que superaba el 0.9 (que en teoría era bueno) pero no era muy fiable ya que a la vez se podía dar un porcentaje de píxeles erróneos alto.

5.2. Trabajo futuro

La tarea de inicialización de fondo todavía tiene un amplio margen de mejora, ya que como se ha mencionado, no hay ningún algoritmo que funcione bien en todos los escenarios.

En cuanto a los algoritmos de este TFG, el problema al que habría que enfrentarse es el de la división fija en bloques. Tener este tipo de división puede llevar a una situación en la que la elección del criterio de continuidad de color puede no ser la más correcta. Un esquema multi-resolución donde se computen diferentes reconstrucciones y se combinen, podría ayudar a solventar este problema. Además, un buen esquema multi-resolución puede evitar problemas de no visualización del *background* debido a tamaños de bloque muy grandes.

Atendiendo a las conclusiones de la sección anterior, lo mejor para los algoritmos de BI es que se diseñaran optimizaciones enfocadas a los objetos estáticos o a la alta densidad de movimiento en la escena. Ya que BI es un paso previo a BS y estas tareas están enfocadas principalmente a vídeo-seguridad, es muy común encontrarse altas densidades de personas y objetos abandonados (estáticos), así que es muy importante que los algoritmos de BI funcionen bien en este tipo de escenarios. Una técnica robusta de estimación de movimiento que se combine con el análisis de parámetros estadísticos que registren las variaciones del background a nivel de bloque, y una sólida medida de continuidad serían unas buenas optimizaciones para los algoritmos de BI.

En resumen, el análisis realizado en este TFG se puede considerar como punto de partida para que, teniendo en cuenta lo comentado en esta sección sobre el trabajo futuro en la tarea de BI, se pueda continuar esta línea de investigación enfocada a la vídeo-vigilancia.

Bibliografía

[1] H. Wang and D. Suter, *A Novel Robust Statistical Method for Background Initialization and Visual Surveillance*, pp. 328–337. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[2] H.-H. Hsiao and J.-J. Leou, "Background initialization and foreground segmentation for bootstrapping video sequences," *EURASIP Journal on Image and Video Processing*, vol. 2013, no. 1, pp. 1–19, 2013.

[3] R. Zhang, W. Gong, A. Yaworski, and M. Greenspan, "Nonparametric on-line background generation for surveillance video," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, pp. 1177–1180, Nov 2012.

[4] D. Ortego, J. C. SanMiguel, and J. M. Martínez, "Rejection based multipath reconstruction for background estimation in video sequences with stationary objects," *Computer Vision and Image Understanding*, vol. 147, pp. 23–37, 2016.

[5] V. Reddy, C. Sanderson, and B. C. Lovell, "A low-complexity algorithm for static background estimation from cluttered image sequences in surveillance contexts," *CoRR*, vol. abs/1303.2465, 2013.

[6] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 1112, pp. 31–66, 2014.

[7] L. Maddalena and A. Petrosino, "Background model initialization for static cameras," in *Background Modeling and Foreground Detection for Video Surveillance*, pp. 3–1, Chapman and Hall/CRC, 2014.

[8] A. Colombari and A. Fusiello, "Patch-based background initialization in heavily cluttered video," *IEEE Transactions on Image Processing*, vol. 19, pp. 926–933, April 2010.

[9] D. Park and H. Byun, "A unified approach to background adaptation and initialization in public scenes," *Pattern Recogn.*, vol. 46, pp. 1985–1997, July 2013.

[10] M. Paul, "Efficient video coding using optimal compression plane and background modelling," *IET Image Processing*, vol. 6, pp. 1311–1318, December 2012.

[11] X. Chen, Y. Shen, and Y. H. Yang, "Background estimation using graph cuts and inpainting," in *Proceedings of Graphics Interface 2010*, GI '10, (Toronto, Ont., Canada, Canada), pp. 97–103, Canadian Information Processing Society, 2010.

[12] Y. Nakashima, N. Babaguchi, and J. Fan, "Automatic generation of privacy-protected videos using background estimation," in *2011 IEEE International Conference on Multimedia and Expo*, pp. 1–6, July 2011.

[13] M. Granados, H.-P. Seidel, and H. P. A. Lensch, "Background estimation from non-time sequence images," in *Proceedings of Graphics Interface 2008*, GI '08, (Toronto, Ont., Canada, Canada), pp. 33–40, Canadian Information Processing Society, 2008.

[14] L. Maddalena and A. Petrosino, "The sobs algorithm: What are the limits?," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 21–26, June 2012.

[15] R. M. Colque and G. CŽmara-ChŽvez, "Progressive background image generation of surveillance traffic videos based on a temporal histogram ruled by a reward/penalty function," in *2011 24th SIBGRAPI Conference on Graphics, Patterns and Images*, pp. 297–304, Aug 2011.

[16] T. Crivelli, P. Bouthemy, B. Cernuschi-Frías, and J.-f. Yao, "Simultaneous motion detection and background reconstruction with a conditional mixed-state markov random field," *International Journal of Computer Vision*, vol. 94, no. 3, pp. 295–316, 2011.

[17] L. Maddalena and A. Petrosino, "Background model initialization for static cameras," in *Background Modeling and Foreground Detection for Video Surveillance*, pp. 3–1, Chapman and Hall/CRC, 2014.

[18] D. Ortego, J. C. SanMiguel, and J. M. Martínez, "Rejection based multipath reconstruction for background estimation in video sequences with stationary objects," *Computer Vision and Image Understanding*, vol. 147, pp. 23 – 37, 2016. Spontaneous Facial Behaviour Analysis.

[19] P. L. St-Charles, G. A. Bilodeau, and R. Bergevin, "Subsense: A universal change detection method with local adaptive sensitivity," *IEEE Transactions on Image Processing*, vol. 24, pp. 359–373, Jan 2015.

[20] H. L. Eng, K. A. Toh, A. H. Kam, J. Wang, and W. Y. Yau, "An automatic drowning detection surveillance system for challenging outdoor pool environments," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 532–539 vol.1, Oct 2003.

[21] L. Maddalena and A. Petrosino, "The 3dsobs+ algorithm for moving object detection," *Computer Vision and Image Understanding*, vol. 122, pp. 65–73, 2014.

[22] C. C. Chen and J. K. Aggarwal, "An adaptive background model initialization algorithm with objects moving at different depths," in *2008 15th IEEE International Conference on Image Processing*, pp. 2664–2667, Oct 2008.

[23] D. Gutchess, M. Trajkovics, E. Cohen-Solal, D. Lyons, and A. K. Jain, "A background model initialization algorithm for video surveillance," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1, pp. 733–740 vol.1, 2001.

[24] V. Reddy, C. Sanderson, and B. C. Lovell, "An efficient and robust sequential algorithm for background estimation in video surveillance," in *2009 16th IEEE International Conference on Image Processing (ICIP)*, pp. 1109–1112, Nov 2009.

[25] D. Baltieri, R. Vezzani, and R. Cucchiara, "Fast background initialization with recursive hadamard transform," in *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*, pp. 165–171, Aug 2010.

[26] A. Shrotre and L. J. Karam, "Background recovery from multiple images," in *Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE), 2013 IEEE*, pp. 135–140, Aug 2013.

[27] X. Xu and T. S. Huang, "A loopy belief propagation approach for robust background estimation," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–7, June 2008.

[28] C. Guo, S. Gao, and D. Zhang, "Belief propagation algorithm for background estimation based on local maximum weight matching," in *Image and Signal Processing (CISP), 2012 5th International Congress on*, pp. 82–85, Oct 2012.

[29] B. M. Jedynak and S. M. Khudanpur, "Maximum likelihood set for estimating a probability mass function," *Neural Comput.*, vol. 17, pp. 1508–1530, July 2005.

[30] L. Maddalena and A. Petrosino, *Towards Benchmarking Scene Background Initialization*, pp. 469–476. Cham: Springer International Publishing, 2015.

[31] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, vol. 2, pp. 1398–1402 Vol.2, Nov 2003.

Apéndice A

Tablas de resultados

A.1. Tablas completas de resultados para la categoría *Baseline*

Las tablas A.1, A.2, A.3 y A.4 muestran los resultados completos para cada secuencia de la categoría *Baseline* y para cada algoritmo.

Tabla A.1: Resultados completos de secuencias *Baseline* para el algoritmo BIB [2]

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
bankst	3.1383	0.7265	0.2721	0.9808	31.8835	41.9366
CAVIAR1	4.8636	2.3031	1.7822	0.9239	26.4913	40.1199
CAVIAR2	1.1458	0.1862	0.0885	0.9949	40.4943	55.5060
Square	12.8917	17.5485	3.1122	0.8399	21.5697	35.2438
Highway	3.0183	1.3281	0.0586	0.9844	32.6288	35.7696
HighwayI	2.7504	0.6901	0.2474	0.9685	34.5601	49.1194
HighwayII	3.1364	0.8242	0.1068	0.9856	33.4647	41.9792
HumanBody2	4.8636	2.4310	1.4258	0.9546	25.6646	39.6230
Gate Traffic	4.5538	4.3210	0.6173	0.9698	27.0037	37.1306
pedestrians	2.0527	0.2025	0.0012	0.9903	37.0064	68.9440
Promedios	**4.2415**	**3.0561**	**0.7712**	**0.9593**	**31.0767**	**44.5372**

Tabla A.2: Resultados completos de secuencias *Baseline* para el algoritmo BIP [3]

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
bankst	3.4272	1.9328	0.5903	0.9502	28.9002	42.3177
CAVIAR1	5.6263	3.1118	2.1047	0.9280	24.4053	38.1234
CAVIAR2	1.6929	1.1414	0.5168	0.9737	30.1264	55.9911
Square	12.4855	16.2688	2.8366	0.8423	21.7261	35.9528
Highway	6.9155	6.3164	2.8867	0.7938	21.6983	31.1458
HighwayI	6.5087	1.7148	0.5469	0.9524	29.5271	51.4935
HighwayII	3.1954	3.1849	1.6276	0.9639	30.2731	46.5934
HumanBody2	2.5843	0.4362	0.1556	0.9826	33.5783	38.3392
Gate Traffic	6.6663	7.0795	2.1209	0.9033	23.5135	37.0568
pedestrians	2.0732	0.2847	0.0046	0.9870	33.5085	59.5130
Promedios	**5.1175**	**4.1471**	**1.3391**	**0.9277**	**27.7257**	**43.6527**

Tabla A.3: Resultados completos de secuencias *Baseline* para el algoritmo RMR [4]

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
bankst	2.9087	0.4029	0.1847	0.9882	32.6459	43.9394
CAVIAR1	3.5357	0.7955	0.3632	0.9861	33.0538	47.3377
CAVIAR2	0.9439	0.14444	0.0346	0.9950	41.4189	59.4437
Square	3.1300	0.5554	0.2025	0.9784	34.2406	36.7683
Highway	4.0025	2.5768	0.0833	0.9752	30.6181	34.8396
HighwayI	2.8930	1.7240	0.2122	0.9664	33.2857	53.2156
HighwayII	2.0979	0.3815	0.0508	0.9922	37.0787	43.9891
HumanBody2	11.0337	9.6406	6.1823	0.7844	18.9248	33.5409
Gate Traffic	4.3089	3.5436	0.4466	0.9748	28.1400	37.8418
pedestrians	1.6700	0.1192	0.0023	0.9943	38.9228	66.2108
Promedios	**3.6524**	**1.9884**	**0.7762**	**0.9635**	**32.8329**	**45.7127**

Tabla A.4: Resultados completos de secuencias *Baseline* para el algoritmo DCT [5]

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
bankst	2.7263	0.3227	0.1368	0.9904	33.2919	44.1523
CAVIAR1	3.1636	0.3866	0.1923	0.9893	33.3013	50.4270
CAVIAR2	0.8105	0.0661	0.0275	0.9992	44.6735	58.4241
Square	12.1288	16.1933	2.7086	0.8632	22.2993	35.8781
Highway	2.8888	1.2161	0.0182	0.9879	33.3625	36.0232
HighwayI	1.5547	0.2161	0.0130	0.9914	39.2110	59.8101
HighwayII	2.2969	0.5742	0.1654	0.9898	33.6525	42.3635
HumanBody2	3.7493	0.5091	0.0039	0.9960	32.3641	44.5844
Gate Traffic	4.6168	4.1291	0.6337	0.9715	27.1270	36.9942
pedestrians	1.5768	0.0266	0.0000	0.9955	40.2840	68.1747
Promedios	**3.5512**	**2.3640**	**0.3899**	**0.9774**	**33.9567**	**47.6832**

A.2. Tablas completas de resultados para la categoría *Clutter*

Las Tablas A.5, A.6, A.7 y A.8 muestran los resultados completos para cada secuencia de la categoría *Clutter* y para cada algoritmo.

Tabla A.5: Resultados completos de secuencias *Clutter* para el algoritmo BIB [2]

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
camera1-ps	4.0982	3.8417	1.5683	0.9326	26.7280	33.7262
camera5-ps	3.4971	2.5633	0.9758	0.9469	29.6417	35.2234
AVSS_AB_EVAL	8.5125	6.5213	5.1177	0.8373	22.1269	31.6413
Board	3.9727	0.9665	0.1006	0.9300	32.1870	51.6811
bootstrap	4.6873	2.1146	0.4115	0.9453	30.3523	37.6348
Crowd	7.0066	1.4244	0.4116	0.9595	29.5045	38.8001
Foliage	16.8171	24.1736	12.1111	0.7589	20.2388	28.3494
PeopleAndFoliage	13.7952	8.5872	3.2552	0.9303	22.8663	30.5484
Campus_test	3.7165	2.5479	1.7534	0.9495	29.1363	35.0972
Snellen	71.3497	81.8480	74.8505	0.3044	9.9990	22.4067
Promedios	**13.7453**	**13.4589**	**10.0556**	**0.8495**	**25.2781**	**34.5109**

Tabla A.6: Resultados completos de secuencias *Clutter* para el algoritmo BIP [3]

Secuencia	AGE	pEPs %	pCEPs %	MSSSIM	PSNR	CQM
camera1-ps	3.1149	1.2725	0.1458	0.9767	29.3892	35.8458
camera5-ps	3.4049	1.0325	0.1058	0.9560	26.3538	35.2507
AVSS_AB_EVAL	25.7231	42.8224	30.2745	0.6739	17.2473	30.6755
Board	24.6600	41.0640	29.1037	0.6584	16.7704	31.7145
bootstrap	7.7657	6.0573	2.1094	0.9021	23.3209	35.5668
Crowd	6.8729	3.4134	1.3003	0.9556	27.7050	41.6019
Foliage	40.9984	72.4097	57.5868	0.5438	14.1728	28.7889
PeopleAndFoliage	47.8364	74.9479	65.7904	0.6013	12.6093	25.5212
Campus_test	5.8643	3.0986	1.0245	0.9603	28.8643	42.8647
Snellen	60.8615	87.1865	79.4030	0.5956	11.5180	30.1805
Promedios	**22.7102**	**33.3305**	**26.6844**	**0.7824**	**20.7951**	**33.8011**

Tabla A.7: Resultados completos de secuencias *Clutter* para el algoritmo RMR [4]

Secuencia	AGE	pEPs %	pCEPs %	MSSSIM	PSNR	CQM
camera1-ps	2.4202	0.7650	0.1450	0.9839	33.9490	36.0327
camera5-ps	2.4872	0.6242	0.1167	0.9858	34.7973	37.8173
AVSS_AB_EVAL	7.2563	4.0111	3.2391	0.9065	22.9396	32.0538
Board	6.9618	3.4024	0.7012	0.8560	28.5183	50.7903
bootstrap	7.2868	6.4115	3.6042	0.8566	25.5874	37.4192
Crowd	3.9624	0.3072	0.0106	0.9799	33.5176	42.9994
Foliage	15.2388	8.3854	4.6597	0.8669	21.5935	29.1635
PeopleAndFoliage	18.7855	15.2005	12.2018	0.8725	19.0745	24.3120
Campus_test	4.7938	2.8064	2.1548	0.9471	26.5748	34.5305
Snellen	24.9716	20.4909	15.2874	0.8739	16.1783	27.5262
Promedios	**9.4164**	**6.2405**	**4.2120**	**0.9129**	**26.2730**	**35.2645**

Tabla A.8: Resultados completos de secuencias *Clutter* para el algoritmo DCT [5].

Secuencia	AGE	pEPs %	pCEPs %	MSSSIM	PSNR	CQM
camera1-ps	2.5232	0.9267	0.2692	0.9856	32.0900	34.3396
camera5-ps	2.4379	0.4158	0.1433	0.9912	33.5113	36.3081
AVSS_AB_EVAL	5.5325	2.2806	1.6983	0.9276	26.5084	35.3428
Board	5.9120	2.7927	0.7256	0.9459	29.3440	50.7419
bootstrap	4.6431	2.1146	0.6042	0.9627	31.1534	38.8726
Crowd	2.4818	0.1341	0.0045	0.9860	36.6168	46.4231
Foliage	19.3872	14.6181	10.7118	0.8441	18.7843	27.6492
PeopleAndFoliage	15.5640	6.8164	4.2813	0.9115	21.1897	27.7497
Campus_test	3.6910	1.6838	1.3287	0.9656	30.2966	37.1718
Snellen	19.6159	17.9832	13.4115	0.9426	18.9212	30.4005
Promedios	**8.1789**	**4.9766**	**3.3178**	**0.9463**	**27.8416**	**36.4999**

A.3. Tablas completas de resultados para la categoría *Low framerate*

Las Tablas A.9, A.10, A.11 y A.12 muestran los resultados completos para cada secuencia de la categoría *Low framerate* y para cada algoritmo.

Tabla A.9: Resultados completos de secuencias *Low framerate* para el algoritmo BIB [2].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
JapMall	7.7759	6.9095	1.1605	0.9020	26.1122	37.2737
CruceNevado	2.8441	0.5228	0.0482	0.9777	35.0582	42.5553
Payot	4.5575	2.4622	0.6107	0.9418	29.6298	42.6995
Intersection	2.6523	0.1758	0.0091	0.9893	36.1264	52.7562
Plaza Mayor	3.9520	2.2962	0.9355	0.9485	27.7203	44.4990
mall	7.8864	6.8685	3.5326	0.8955	24.4329	38.0268
ShoppingMall	9.2145	12.0764	7.6062	0.8922	26.0751	38.2861
tramCrossroad	4.9214	2.1071	1.0942	0.9423	28.9078	45.6020
tunnelExit	3.2511	0.3357	0.0497	0.9706	33.2390	34.7156
turnpike	6.6672	5.7174	2.5599	0.8620	25.7620	37.1024
Promedios	**5.3722**	**3.9472**	**1.7697**	**0.9322**	**29.3064**	**41.3517**

Tabla A.10: Resultados completos de secuencias *Low framerate* para el algoritmo BIP [3].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
JapMall	7.0504	5.3929	1.4066	0.9034	26.2230	38.4828
CruceNevado	2.4498	0.3672	0.0247	0.9839	36.2354	43.8956
Payot	3.7795	1.5898	0.2292	0.9657	31.0509	44.2658
Intersection	2.6875	0.2734	0.0091	0.9891	35.7902	51.8862
Plaza Mayor	3.1568	1.8034	0.5632	0.9709	30.8485	45.3958
mall	9.0301	8.3216	5.5026	0.9409	25.9963	42.1668
ShoppingMall	7.4456	4.1150	1.5637	0.9394	27.3949	40.2686
tramCrossroad	4.5398	2.8286	1.1821	0.9433	28.9685	48.4135
tunnelExit	2.6404	0.3299	0.1448	0.9844	34.1967	36.5050
turnpike	3.7014	1.6536	0.1445	0.9571	28.0460	38.7405
Promedios	**4.6481**	**2.6675**	**1.0771**	**0.9578**	**30.4750**	**43.0021**

Tabla A.11: Resultados completos de secuencias *Low framerate* para el algoritmo RMR [4].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
JapMall	3.9325	0.3170	0.0116	0.9793	33.5876	42.9074
CruceNevado	2.5709	0.4242	0.0443	0.9770	35.6741	43.1256
Payot	3.6237	1.6693	0.3776	0.9594	31.1161	44.5685
Intersection	2.6528	0.1810	0.0091	0.9895	36.0898	52.5821
Plaza Mayor	2.9659	1.5140	0.6253	0.9669	29.3220	45.7587
mall	5.8300	4.1354	1.8724	0.9500	28.7995	42.9061
ShoppingMall	4.5129	2.5354	0.9570	0.9305	30.6419	40.3919
tramCrossroad	3.7944	0.5558	0.0871	0.9720	32.8268	46.7168
tunnelExit	2.2601	0.3481	0.1617	0.9832	34.4582	37.3855
turnpike	9.3919	5.2782	2.5165	0.9347	20.9781	34.4435
Promedios	**4.1535**	**1.6958**	**0.6663**	**0.9643**	**31.3494**	**43.0786**

Tabla A.12: Resultados completos de secuencias *Low framerate* para el algoritmo DCT [5].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
JapMall	6.3473	4.9600	0.6960	0.9331	27.5153	38.7858
CruceNevado	2.1336	0.3190	0.0225	0.9873	37.0853	44.1579
Payot	3.1424	1.4206	0.4063	0.9756	32.1512	45.7638
Intersection	2.7444	0.1758	0.0091	0.9899	36.0191	52.7074
Plaza Mayor	3.0238	1.5918	0.6917	0.9683	29.3202	45.8102
mall	4.4561	2.9245	1.5000	0.9769	30.2552	43.0389
ShoppingMall	4.5158	1.6016	0.2844	0.9650	30.2480	41.1175
tramCrossroad	2.7121	0.2201	0.0504	0.9869	35.6681	50.7678
tunnelExit	2.0083	0.1799	0.0380	0.9855	36.1741	38.1020
turnpike	2.8940	1.7540	0.2129	0.9604	33.2857	53.2116
Promedios	**3.3978**	**1.5147**	**0.3911**	**0.9729**	**32.7722**	**45.3463**

A.4. Tablas completas de resultados para la categoría *Static objects*

Las Tablas A.13, A.14, A.15 y A.16 muestran los resultados completos para cada secuencia de la categoría *Static objects* y para cada algoritmo.

Tabla A.14: Resultados completos de secuencias *Static objects* para el algoritmo BIP [3].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BusPolaco	4.7042	5.3681	2.8823	0.8035	29.1780	54.8406
abandonedBox	9.3959	5.2485	2.5069	0.9357	20.9714	34.3433
BusStopMorning	6.9891	4.1940	1.1198	0.8648	20.7680	35.0390
CaVignal	13.8113	11.8897	6.2169	0.6372	16.6000	31.3154
granguardia	11.3426	15.6092	8.2586	0.8561	19.0671	44.1631
HallAndMonitor	3.4264	1.6205	0.3172	0.9541	27.0289	37.7678
Gate Traffic	7.6243	7.2849	3.2745	0.9109	21.7720	37.0740
office	9.6400	7.0775	5.4155	0.8616	20.0504	32.1391
sofa	3.4554	2.4388	1.3971	0.9047	25.4241	34.3062
winterDriveway	9.6113	10.0247	5.2604	0.8463	21.1132	39.3567
Promedios	**8.0000**	**7.0756**	**3.6649**	**0.8575**	**22.1973**	**38.0345**

Tabla A.13: Resultados completos de secuencias *Static objects* para el algoritmo BIB [2].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BusPolaco	6.5219	7.6497	4.7230	0.6935	25.2242	52.4734
abandonedBox	9.3227	3.8227	1.3519	0.9376	24.5600	36.2278
BusStopMorning	5.5900	3.5326	1.7201	0.8729	23.6888	38.1887
CaVignal	3.3292	1.3051	0.5331	0.9727	29.3805	45.7747
granguardia	5.5888	7.7931	4.1006	0.9143	23.3944	49.7050
HallAndMonitor	4.0270	2.3686	1.3613	0.9142	26.8446	36.5976
Gate Traffic	4.8079	4.1281	1.0195	0.9617	25.8967	37.4558
office	6.0644	2.3947	0.6806	0.9683	26.0753	37.8483
sofa	2.6316	1.4596	0.8320	0.9717	33.1379	35.2446
winterDriveway	6.3249	3.5508	1.2344	0.9342	24.3363	39.6022
Promedios	**5.4208**	**3.8005**	**1.7556**	**0.9141**	**26.2539**	**40.9118**

Tabla A.15: Resultados completos de secuencias *Static objects* para el algoritmo RMR [4].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BusPolaco	2.3757	1.2676	0.6195	0.9494	34.5825	54.2882
abandonedBox	4.3549	1.0706	0.0619	0.9924	32.0578	40.8075
BusStopMorning	3.5213	0.7839	0.1823	0.9868	31.8338	41.6363
CaVignal	0.7635	0.0147	0.0000	0.9971	43.6597	52.6372
granguardia	4.1603	3.9023	2.4397	0.9720	27.2380	49.0127
HallAndMonitor	2.0718	0.0604	0.0024	0.9932	38.2659	46.6682
Gate Traffic	3.8755	2.6524	0.4032	0.9798	28.1569	38.8896
office	4.5207	1.6840	1.1563	0.9722	27.6215	37.4011
sofa	2.9018	2.4740	1.8490	0.9525	28.8117	33.6053
winterDriveway	5.8309	3.4753	0.3516	0.9724	27.7269	40.0462
Promedios	**3.4376**	**1.7385**	**0.7066**	**0.9768**	**31.9955**	**43.4992**

Tabla A.16: Resultados completos de secuencias *Static objects* para el algoritmo DCT [5].

Secuencia	AGE	pEPs	pCEPs	MSSSIM	PSNR	CQM
BusPolaco	3.1785	3.2384	1.9738	0.9053	30.9566	55.3092
abandonedBox	6.3111	3.7150	2.2015	0.9710	24.4105	36.5809
BusStopMorning	4.5271	1.6445	0.5977	0.9271	25.5381	40.4527
CaVignal	0.9179	0.0221	0.0000	0.9968	43.6768	53.4936
granguardia	8.7922	11.8908	7.2443	0.8441	21.1083	46.8512
HallAndMonitor	2.8137	1.1387	0.5078	0.9674	29.5846	39.4012
Gate Traffic	6.6342	5.7272	2.7932	0.9371	22.2051	36.7092
office	4.0567	1.1817	0.7847	0.9829	28.5342	38.0294
sofa	1.6714	0.8112	0.6367	0.9843	37.5194	38.3686
winterDriveway	12.4751	13.7487	9.1875	0.8244	19.8038	38.3271
Promedios	**5.1378**	**4.3118**	**2.5927**	**0.9340**	**28.3337**	**42.3523**