INSTITUTO TECNOLÓGICO DE CHIHUAHUA

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

“ALGORITMOS DE SEGUIMIENTO DE OBJETOS EN SECUENCIAS DE VIDEO BASADOS EN FILTROS DE CORRELACIÓN”

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRA EN CIENCIAS EN INGENIERÍA ELECTRÓNICA

PRESENTA:

ING. ANDREA RIVERO OLIVAS

DIRECTOR DE LA TESIS:
DR. MARIO IGNACIO CHACÓN MURGUÍA

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

CHIHUAHUA, CHIH., OCTUBRE 2019

Chihuahua, Chih. 4 de octubre de 2019

C. ANDREA RIVERO OLIVAS
PRESENTE

Por este conducto le comunico que a propuesta del Jurado de Examen, la División de Estudios de Posgrado e Investigación ha concedido autorización para la impresión de su tesis para obtener el grado de Maestra en Ciencias en Ingeniería Electrónica, cuyo título es:

"Algoritmos de seguimiento de objetos en secuencias de video basados en filtros de correlación"

La tesis presenta el siguiente contenido de capítulos:

I Antecedentes
II Instrumento de evaluación de métodos
III Seguimiento de objetos mediante filtros de correlación
IV Resultados y conclusiones

ATENTAMENTE
Excelencia en Educación Tecnológica®
"La técnica por el Engrandecimiento de México"

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CHIHUAHUA
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

MTRO. LUIS CARDONA CHACÓN
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

Chihuahua, Chih. 4 de octubre de 2019

MTRO. LUIS CARDONA CHACÓN
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
PRESENTE

Por medio de la presente notificamos a usted que en cumplimiento de los requerimientos para la obtención del grado de Maestra en Ciencias en Ingeniería Electrónica, el documento de tesis de la **C. ANDREA RIVERO OLIVAS**, ha sido aprobado y aceptado para su impresión. El título de la tesis es:

"Algoritmos de seguimiento de objetos en secuencias de video basados en filtros de correlación"

Por lo que proponemos, le sea concedida la autorización de impresión correspondiente.

Agradeciendo la atención a la presente, quedamos de usted:

ATENTAMENTE
Excelencia en Educación Tecnológica®
"La técnica por el Engrandecimiento de México"

DR. MARIO IGNACIO CHACÓN MURGUÍA
DIRECTOR DE TESIS

DR. JUAN ALBERTO RAMÍREZ QUINTANA
MIEMBRO DEL JURADO DE EXAMEN

EDUCACIÓN SECRETARÍA DE EDUCACIÓN PÚBLICA | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CHIHUAHUA
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

MTRA. ALMA DELIA CORRAL SÁENZ
MIEMBRO DEL JURADO DE EXAMEN

DR. JAVIER VEGA PINEDA
MIEMBRO DEL JURADO DE EXAMEN

CARTA CESIÓN DE DERECHOS

En la ciudad de Chihuahua el día **15 de octubre de 2019, el** que suscribe **C. ANDREA RIVERO OLIVAS** de la Maestría en Ciencias en Ingeniería Electrónica, con número de control **G17061504**, adscrito a la **DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN**, del Instituto Tecnológico de Chihuahua, manifiesta que es autor intelectual del presente trabajo de Tesis bajo la dirección del **Dr. Mario Ignacio Chacón Murguía** y cede los derechos del trabajo titulado ***"Algoritmos de seguimiento de objetos en secuencias de video basados en filtros de correlación"***, al Tecnológico Nacional de México y/o Instituto Tecnológico de Chihuahua para su difusión, divulgación, transmisión, reproducción, así como su digitalización con fines académicos y de investigación.

C. ANDREA RIVERO OLIVAS

Chihuahua, Chihuahua, a 15 de octubre de 2019

Dra. María Elena Álvarez-Buylla Roces
Directora del CONACYT

Por este conducto aprovecho la ocasión para saludarla e informarle que a la fecha he obtenido el grado de **Maestría en Ciencias en Ingeniería Electrónica** en la División de Estudios de Posgrado e Investigación del Instituto Tecnológico de Chihuahua. Agradezco todo el apoyo brindado por CONACYT ya que el otorgamiento de la beca para estudios de Posgrado, permitió dedicarme de tiempo completo al programa y con ello lograr el cumplimiento del objetivo principal del convenio establecido.

Sin otro particular por el momento, me es grato quedar de usted como su seguro servidor, no sin antes reiterar nuevamente todo mi agradecimiento.

A t e n t a m e n t e

Andrea Rivero Olivas
Exbecaria CONACYT

“Tus talentos y habilidades irán mejorando con el tiempo, pero para eso has de empezar”
-Martin Luther King

“He aquí mi secreto, que no puede ser más simple: solo con el corazón se puede ver bien; lo esencial es invisible a los ojos”
-El principito

Para mis padres, Francisca y Martin,
mis hermanos, Claudia y Mauricio,
y para el amor de mi vida, Daniel.

AGRADECIMIENTOS

A Dios

A veces volteo al cielo, sonrío y digo “yo sé que fuiste tú”.
Gracias por permitirme llegar hasta este momento tan importante en mi formación profesional, por acompañarme en cada paso y bendecirme de tantas maneras.

A mis padres

Por brindarme su apoyo incondicional y nunca dejar de creer en mí. Cada logro en mi vida ha sido gracias a su amor, trabajo y sacrificio.

A mis hermanos

Por ser parte de mi vida y guiarla con sus consejos y enseñanzas.

A Daniel

Por su amor que me acompaña en todo momento, bueno o malo, triste o feliz y por alentarme a perseguir y hacer mis sueños realidad.

A mis amigos

Raúl, Adrián, Eduardo, Luis, Abimael, Pepe, David, Xavier, Javier, Alonso, Iván, Alejandro, Oscar, Jorge, Manuel y Gerardo, por brindarme su amistad y hacer de cada momento en el laboratorio uno especial.

A mis profesores

En especial al Dr. Mario Chacón que con su dirección y colaboración fue posible la realización de este trabajo y por todas sus enseñanzas que me acompañarán toda la vida.
También agradezco a la Mtra. Alma Corral, Dr. Javier Vega y Dr. Juan Ramírez que fueron parte de mi formación profesional.

Al Tecnológico Nacional de México / I. T. Chihuahua

Gracias al Tecnológico Nacional de México / I. T. Chihuahua por el apoyo recibido para el proyecto “Algoritmo de detección de movimiento en secuencias de video basado en arquitectura de redes neuronales tradicionales o de aprendizaje profundo y lógica difusa”, clave 5162.19-P.

RESUMEN

ALGORITMOS DE SEGUIMIENTO DE OBJETOS EN SECUENCIAS DE VIDEO BASADOS EN FILTROS DE CORRELACIÓN

Ing. Andrea Rivero Olivas
Maestría en Ciencias en Ingeniería Electrónica
División de Estudios de Posgrado e Investigación del
Instituto Tecnológico de Chihuahua
Chihuahua, Chih., 2019
Director de Tesis: Dr. Mario Ignacio Chacón Murguía

La visión por computadora es una rama de la inteligencia artificial que crea modelos inspirados en la visión biológica. En esta se encuentran las áreas de detección de objetos en movimiento, reconocimiento de objetos y seguimiento de objetos, por mencionar algunas. Esta última consiste en la estimación del estado del objeto de interés (posición, velocidad, aceleración, etc.) dentro de una secuencia de video. Llevar a cabo esta estimación se torna difícil si durante el seguimiento se presentan situaciones como oclusiones o deformaciones que cambian el aspecto del objeto de interés. No obstante, el área de seguimiento de objetos ha recibido gran atención de varios investigadores debido a las diferentes aplicaciones de esta como sistemas de vigilancia inteligentes, conducción autónoma e interacción humano-computadora.

En esta investigación se presenta un algoritmo de seguimiento de objetos denominado como FCCNN obtenido de aplicar una metodología de diseño propuesta que inicia con la determinación del enfoque principal y prosigue con el desarrollo de varios algoritmos de seguimiento de objetos que conducen a la versión final FCCNN.

El enfoque principal es determinado por un instrumento de evaluación de métodos aplicados al seguimiento de objetos cuyo propósito es indicar cuáles presentan las mejores características y por lo tanto cuál enfoque obtiene mejor desempeño en el área. Los resultados de este instrumento de evaluación concluyen que los filtros de correlación son el enfoque buscado, por lo que todos los algoritmos desarrollados en esta tesis comparten el uso de este.

Primero se desarrollaron los algoritmos *fc-hog*, *fc-color* y *fc-cnn* que utilizan las características de histograma de gradientes orientados, histograma de color difuso y características extraídas de una red neuronal convolucional preentrenada, respectivamente. La evaluación de estos algoritmos concluye que *fc-cnn* brinda mejor desempeño por lo que se continúa la investigación sobre este.

Posteriormente, se incorporaron algunos métodos a *fc-cnn* como la detección de oclusiones y deformaciones, lo que dieron lugar a varios algoritmos derivados de este. Luego de la evaluación de estos con 100 secuencias de video se encontró aquel con mejores resultados convirtiéndolo en el algoritmo FCCNN de esta tesis.

FCCNN logra una precisión de 0.70, un éxito de 0.62 y una velocidad de procesamiento de 13.479 FPS lo que demuestra que el algoritmo desarrollado es competente al ser comparado con otros de la literatura reciente.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

NOTACIÓN

Notación	Descripción
$I(x,y)$	Imagen.
$I_R(x,y)$	Componente rojo del espacio de color RGB.
$I_G(x,y)$	Componente verde del espacio de color RGB.
$I_B(x,y)$	Componente azul del espacio de color RGB.
PF	Puntuación final del método evaluado con el IE.
Do	Criterio de documentación.
De	Criterio de desempeño.
DS	Criterio de desafíos superados.
Do_{Pr}	Subcriterio de procesamiento.
Pr_v	Métrica de indica velocidad de procesamiento.
Pr_{cc}	Métrica de cantidad de cuadros en los videos.
V_{cc}	Número de videos en los que se da a conocer la cantidad de cuadros.
V	Cantidad total de videos utilizados.
Pr_r	Métrica de resolución de los videos.
V_r	Cantidad de videos en los que se da a conocer la resolución.
Pr_{ec}	Métrica de espacio de color.
Pr_{tr}	Métrica de procesamiento en tiempo real.
Pr_{ee}	Métrica de uso de un esquema de entrenamiento.
Pr_{cp}	Métrica de cantidad de parámetros que utiliza el método.
Pa	Cantidad de parámetros que se utiliza en el método.
P_{max}	Cantidad máxima de parámetros permitida en el método.
Do_R	Subcriterio de reproducibilidad.
R_{in}	Métrica de infraestructura utilizada.
ei	Elemento de infraestructura (*pro*, *ram*, *cg* o *sl*).
pro	Procesador.
ram	Memoria de acceso aleatorio (RAM).
cg	CPU/GPU.
sl	*Software*/lenguaje de programación.
R_{cfg}	Métrica de configuración de los parámetros.
P_{cfg}	Cantidad de parámetros en los que se indica su valor configurado.
Do_{Ev}	Subcriterio de evaluación.
Ev_{bd}	Métrica de cantidad de bases de datos.
bdc	Cantidad de bases de datos que se usaron de manera completa.
bdi	Cantidad de bases de datos que se usaron de manera incompleta.
bd	Cantidad total de bases de datos utilizadas.

Ev_{cn}	Métrica de evaluación cuantitativa.
Ev_{cl}	Métrica de evaluación cualitativa.
Ev_{cfg}	Métrica de una sola configuración.
Do_C	Subcriterio de comparación.
De_{ve}	Métrica de velocidad de procesamiento.
fps	Velocidad de procesamiento reportada en cuadros por segundo.
fps_{max}	Velocidad de procesamiento máxima en cuadros por segundo.
De_p	Métrica de precisión.
CLE	Error de localización central.
(x_a,y_a)	Centro del rectángulo delimitador producido por el algoritmo.
(x_{gt},y_{gt})	Centro del rectángulo delimitador *ground truth*.
P_P	Precisión mediante promediado.
P_U	Precisión mediante umbral.
P_A	Precisión mediante AUC.
w_P	Ponderación para la forma de dar a conocer la precisión/éxito por promedio.
w_U	Ponderación para la forma de dar a conocer la precisión/éxito por umbral.
w_A	Ponderación para la forma de dar a conocer la precisión/éxito por AUC.
th	Umbral de corte para la precisión mediante promediado.
P_P'	Precisión por promediado normalizada.
De_e	Métrica de éxito.
RS	Razón de superposición.
r_{gt}	Región definida por el rectángulo delimitador *ground truth*.
r_a	Región definida por el rectángulo delimitador producida por el algoritmo.
E_P	Éxito mediante promediado.
E_U	Éxito mediante umbral.
E_A	Éxito mediante AUC.
De_{eex}	Métrica de uso de esquema de evaluación externo.
N_E	Nivel de uso del esquema de evaluación externo.
E	Cantidad de esquemas de evaluación externos utilizados.
M_u	Cantidad de métricas utilizadas que pertenecen al esquema de evaluación externo.
M_E	Cantidad total de métricas en el esquema de evaluación externo.
De_m	Métrica de cantidad de métricas utilizadas.
Me	Cantidad de métricas utilizadas para evaluar al método.
M_{max}	Cantidad máxima métricas.
De_d	Métrica de cantidad de desafíos.
D	Cantidad de desafíos sobre los cuales el algoritmo es puesto a prueba.
D_{max}	Cantidad máxima de desafíos.

De_v	Métrica de cantidad de videos.
V_{max}	Cantidad máxima de videos.
De_a	Métrica de cantidad de algoritmos utilizados en la comparación.
Al	Cantidad de algoritmos utilizados en la comparación.
A_{max}	Cantidad máxima de algoritmos de comparación.
LG	Resultado tipo lingüístico en los desafíos.
CN	Resultado tipo cuantitativo en los desafíos.
CL	Resultado tipo cualitativo en los desafíos.
des	Desafío.
x	Índice renglón.
y	Índice columna.
n	Número de cuadro de video.
$I(x,y,n)$	Cuadro n de una secuencia de video.
$T(x,y,n)$	Porción rectangular que contiene al objeto de interés en $I(x,y,n)$.
$T_g(x,y,n)$	Versión de $T(x,y,n)$ en escala de grises.
G_r	Algoritmo para la conversión del espacio de color RGB a escala de gris.
E_c	Algoritmo de ecualización de histograma.
$T_e(x,y,n)$	Versión con realce de contraste de $T_g(x,y,n)$.
dx	Diferencia en la intensidad de los pixeles vecinos en dirección horizontal.
dy	Diferencia en la intensidad de los pixeles vecinos en dirección vertical.
$g(x,y)$	Magnitud del gradiente.
$\theta(x,y)$	Orientación del gradiente.
hc	Histograma de gradientes orientados nivel celda.
φ	Intervalo de orientación en el histograma de gradientes orientados.
$h\omega_i^x$	Ponderación en la interpolación espacial horizontal para $g(x,y)$.
$h\omega_i^y$	Ponderación en la interpolación espacial vertical para $g(x,y)$.
$h\omega_\varphi$	Ponderación en la interpolación en orientación para $g(x,y)$.
d_i^x	Distancia horizontal entre el elemento (x,y) y el centro de la celda i.
δx	Distancia horizontal entre los centros de las celdas vecinas.
d_i^y	Distancia vertical entre el elemento (x,y) y el centro de la celda i.
δy	Distancia vertical entre los centros de las celdas vecinas.
θ_φ	Valor de orientación centro del intervalo φ.
ϑ	Tamaño del intervalo en el histograma de gradientes orientados.
hb	Histograma de gradientes orientados nivel bloque.
h_{hog}	Histograma de gradientes orientados del objeto de interés.
L_{hog}	Tamaño del histograma de gradientes orientados.
S	Número de intervalos en el histograma de gradientes orientados.
C	Número de celdas en el bloque.

Bl	Número de bloques.
L	Variable lingüística de entrada del sistema difuso.
a	Variable lingüística de entrada del sistema difuso.
b	Variable lingüística de entrada del sistema difuso.
$color$	Variable lingüística de salida del sistema difuso.
MF_L	Valor difuso de la variable lingüística L.
MF_a	Valor difuso de la variable lingüística a.
MF_b	Valor difuso de la variable lingüística b.
MF_c	Valor difuso de la variable lingüística $color$.
df	Valor de defusificación.
υ	Dominio de la función de pertenencia agregada.
$H\upsilon$	Conjunto de valores υ cuyos grados de pertenencia son los más altos.
ζ	Función que realiza las operaciones de fusificación, evaluación de las reglas difusas, agregación y defusificación.
$T_d(x,y,n)$	Salida del sistema difuso al evaluar $T(x,y,n)$.
h_{color}	Histograma de color difuso del objeto de interés.
np	Número de pixeles en $T_d(x,y,n)$.
c	Intervalo en h_{color}.
$\delta(\cdot)$	Función delta Kronecker.
$F(x,y)$	Filtro en la red neuronal convolucional.
$MP(x,y)$	Mapa de características.
ch	Profundidad de los datos.
L_M	Dimensión de $MP(x,y)$.
L_I	Dimensión de $I(x,y)$.
L_F	Dimensión de $F(x,y)$.
L_P	Tamaño del *padding*.
L_S	Tamaño del *stride*.
β	Escalar *bias*.
$T_r(x,y,n)$	$T(x,y,n)$ preprocesada.
$\xi_i(x,y,n)$	Entrada a la red neuronal convolucional en la capa i.
$O_i(x,y,n)$	Salida de la red neuronal convolucional en la capa i.
Na	Nivel de activación del objeto de interés en el mapa de características.
$K_i(x,y)$	Máscara para el cálculo del nivel de activación en la capa i.
(cx,cy)	Pixel central.
mp_i	Número de mapas de características con mayor nivel de activación en la capa i.
$F_{i,j}(x,y)$	Filtros convolucionales seleccionados.
$mc^l(x,y)$	Mapas de características seleccionados.
q	Características extraídas de $T(x,y,n)$.

r	Salida de correlación deseada.
γ	Filtro de correlación.
ψ	Mapa de respuesta.
z	Características extraídas de la ROI.
Q	Características extraídas de $T(x,y,n)$ en el dominio de la frecuencia.
R	Salida de correlación deseada en el dominio de la frecuencia.
Γ	Filtro de correlación en el dominio de la frecuencia.
Z	Características extraídas de la ROI en el dominio de la frecuencia.
A	Numerador del filtro de correlación en el dominio de la frecuencia.
B	Denominador del filtro de correlación en el dominio de la frecuencia.
η	Parámetro de aprendizaje del filtro de correlación.
λ	Parámetro de regularización.
r_{hog}	Salida de correlación deseada del algoritmo *fc-hog*.
A_{hog}	Amplitud de r_{hog}.
μ_{hog}	Media de r_{hog}.
σ_{hog}	Desviación estándar de r_{hog}.
Γ_{hog}	Filtro de correlación en el dominio de la frecuencia del algoritmo *fc-hog*.
H_{hog}	Histograma de gradientes orientados del objeto de interés en el dominio de la frecuencia.
R_{hog}	Salida de correlación deseada en el domino de la frecuencia.
ψ_{hog}	Mapa de respuesta del algoritmo *fc-hog*.
ra	Radio de apertura de cada espiral de la ventana deslizante dada en pixeles.
α	Ángulo en radianes entre cada muestra tomada por la ventana deslizante.
es	Cantidad de espirales en la ventana deslizante.
$\rho(n)$	Posición estimada del objeto de interés en el cuadro n.
z_{hog}	Histograma de gradientes orientados de la muestra.
Z_{hog}	Histograma de gradientes orientados de la muestra en el dominio de la frecuencia.
mt	Muestra que contiene al objeto de interés en los algoritmos *fc-hog* y *fc-color*.
h_{dc}	Descriptor de la distribución espacial del color del objeto de interés.
L_{dc}	Tamaño del descriptor de la distribución espacial del color.
r_{dc}	Salida de correlación deseada del algoritmo *fc-color*.
A_{dc}	Amplitud de r_{dc}.
μ_{dc}	Media de r_{dc}.
σ_{dc}	Desviación estándar de r_{dc}.
Γ_{dc}	Filtro de correlación en el dominio de la frecuencia del algoritmo *fc-color*.
H_{dc}	Descriptor de la distribución espacial del color del objeto de interés en el dominio de la frecuencia.
R_{dc}	Salida de correlación deseada en el domino de la frecuencia.

ψ_{dc}	Mapa de respuesta del algoritmo *fc-color*.
z_{dc}	Descriptor de la distribución espacial del color de la muestra.
Z_{dc}	Descriptor de la distribución espacial del color de la muestra en el dominio de la frecuencia.
r_{cnn}	Salida de correlación deseada del algoritmo *fc-cnn*.
A_{cnn}	Amplitud de r_{cnn}.
σ_{cnn}	Desviación estándar de r_{cnn}.
Γ^{l}_{cnn}	Filtros de correlación en el dominio de la frecuencia del algoritmo *fc-cnn*.
MC^{l}	Mapas de características en el dominio de la frecuencia.
R_{cnn}	Salida de correlación deseada en el domino de la frecuencia.
ψ_{cnn}	Mapa de respuesta del algoritmo *fc-cnn*.
z^{l}_{cnn}	Mapas de características de la ROI en cuadros posteriores.
Z^{l}_{cnn}	Mapas de características de la ROI en el dominio de la frecuencia en cuadros posteriores.
f_{σ}	Factor de la desviación estándar.
σ_x, σ_y	Desviaciones estándar de la máscara super-gaussiana.
N	Ancho del objeto de interés.
M	Alto del objeto de interés.
ω	Ponderación de los mapas de respuesta.
Ψ	Mapa de respuesta final.
PSR	Razón entre el pico y la región del lóbulo lateral del mapa de respuesta final.
ψ_{max}	Magnitud del pico de correlación en el mapa de respuesta final.
μ_{sl}	Media de la región del lóbulo lateral.
σ_{sl}	Desviación estándar del lóbulo lateral.
PSR_{cf}	Valores *PSR* de los mapas de respuesta de cuadros donde no se presentaron oclusiones ni deformaciones.
$\hat{PSR}$	Promedio ponderado de los valores de PSR_{cf}.
C_{cf}	Número de cuadros recientes donde no se presentaron oclusiones ni deformaciones.
ω_{PSR}	Ponderaciones de los valores de PSR_{cf}.
N_{inc}	Nivel de incertidumbre.
$occdef$	Estado lógico de la presencia de oclusiones y deformaciones.
τ_{inc}	Umbral que define cuando se detecta o no una oclusión o deformación.

CAPÍTULO I. ANTECEDENTES

La popularidad de la visión por computadora ha incrementado en los últimos años y es tan basta su utilidad en diversas aplicaciones que ha propiciado la continua investigación de este campo. Una de las áreas de mayor auge dentro de la visión por computadora es el área de seguimiento de objetos, misma que consiste en la determinación del estado, dígase su posición, velocidad y aceleración, entre otros, de uno o varios objetos de interés. En los últimos años se han desarrollado gran cantidad de algoritmos de seguimiento de objetos basados en diferentes técnicas y métodos con el propósito de superar los diversos desafíos que conlleva esta tarea. En las siguientes secciones se explicarán con más detalle algunos de los conceptos esenciales dentro del tema de seguimiento de objetos y se hablará del estado del arte que se tiene en esta área.

1.1 Visión por computadora

El sentido de la vista permite percibir el mundo de una manera gráfica, brindando toda clase de información como color, textura, forma, posición y tamaño, de los diferentes objetos en el espacio. En conjunto, los ojos y cerebro crean un sistema de adquisición de información complejo y tan útil que no es de asombrarse que el interés de recrear este proceso se esté llevando a cabo en el mundo moderno.

La visión por computadora es una rama de la inteligencia artificial que crea modelos inspirados en la visión biológica, de tal manera que, mediante la interpretación y descripción de las imágenes, se puedan reconocer y analizar los diversos objetos presentes y su posición en el espacio [1]. Esta es una disciplina compleja que ha requerido muchos años de investigación, sin embargo, de acuerdo con [1] en las últimas décadas ha tenido grandes avances básicamente por 3 factores:

1. El desarrollo tecnológico en las capacidades de procesamiento y almacenamiento en las computadoras.
2. Los avances teóricos en los principios y algoritmos para el procesamiento y análisis de imágenes.

3. La creciente necesidad del procesamiento automático de imágenes que se capturan y almacenan en grandes cantidades en diversas áreas como medicina, seguridad y tránsito de vehículos, por mencionar algunas.

Hay un gran número de áreas en la visión por computadora, entre ellas podemos mencionar el reconocimiento de objetos, detección de objetos en movimiento, reconstrucción de escenas y seguimiento de objetos, siendo esta última el principal tema en este trabajo.

1.2 Detección de objetos

Un algoritmo de seguimiento de objetos requiere de un mecanismo de detección, que consiste en la identificación de objetos en cada cuadro de video y métodos de segmentación que realizan la agrupación de los pixeles de estos [2]. De acuerdo con [2], el primer paso para la detección es distinguir entre los objetos del primer plano y los del fondo estacionario. Para lograr esto, se crea un mapa de píxeles del primer plano en cada cuadro y luego se agrupan las regiones conectadas en este mapa para después extraer las características individuales de los objetos, por ejemplo, cuadros delimitadores, áreas y perímetros.

Un enfoque común para la detección de objetos es usar información de un solo cuadro. Sin embargo, algunos métodos de detección hacen uso de la información temporal calculada a partir de una secuencia de cuadros para reducir el número de detecciones falsas. Esta información temporal suele ser en forma de diferenciación de cuadros, que resaltan las regiones que cambian dinámicamente en cuadros consecutivos [2]. La diferenciación entre cuadros consecutivos es un método con una implementación simple y con un buen desempeño para fondos estáticos, sin embargo, para mantener un buen resultado, el fondo debe estar libre de objetos en movimiento [3]. Otro tipo de método para la detección de objetos es el método de sustracción de fondo. El primer paso para este es realizar un modelado de fondo que es usado como modelo de referencia. Este modelo es comparado con cada cuadro de video para la determinación de posibles variaciones o aparición de objetos en la escena [3].

1.3 Representación del objeto para el seguimiento

En el desarrollo de un algoritmo de seguimiento de objetos es común empezar con la pregunta ¿Qué es lo que se pretende seguir?, la respuesta a esto es sencilla, sin embargo,

comunicarlo a una computadora resulta tener mayor complejidad. La representación de un objeto consiste en la manera en que este será visto por la computadora por lo que se debe elegir una representación que mejor lo caracterice. De acuerdo con Yilmaz *et al.* [4] los objetos pueden ser representados a través de su forma y su apariencia.

1.3.1 Tipos de representaciones del objeto

Como se ha mencionado, la representación de un objeto de interés se puede llevar a cabo a través de su forma y apariencia. La diferencia entre estos dos términos es que la apariencia se refiere al aspecto que luce el objeto mientras que la forma es la figura exterior que posee este. El uso de alguna de estas representaciones depende del tipo de objeto de interés con el que se esté trabajando.

1.3.1.1 Representación basada en apariencia

Existen varias características utilizadas para la representación de la apariencia del objeto, entre las cuales se pueden mencionar el color y la textura.

La característica de color es una de las más usadas dentro de la literatura en el área de seguimiento de objetos [5]–[7]. Las ventajas más importantes de la característica de color son la capacidad de representar el contenido visual de una imagen, la simplicidad en la extracción de la información del color y su independencia ante el tamaño de la imagen y su orientación [8]. Para la representación del objeto utilizando esta característica se debe descubrir cuál es el espacio de color más adecuado. Un espacio de color es una representación numérica de los colores, generalmente se basan en un espacio tridimensional donde cada color queda representado por un único punto de este espacio. Algunos ejemplos de espacio de color son el RGB (*red, green, blue*) y el HSI (*hue, saturation, intensity*).

El espacio de color RGB es un modelo aditivo en donde los colores primarios rojo, verde y azul se combinan para producir una amplia gama de colores [9]. Su espacio tridimensional es un cubo como se muestra en la Figura 1.1. En el espacio de color RGB cada pixel de una imagen se representa como un vector de 3 componentes como lo muestra la Ec. (1.1),

$$I(x,y) = \{I_R(x,y), I_G(x,y), I_B(x,y)\} \tag{1.1}$$

donde x y y son los índices de renglón y columna de la imagen $I(x,y)$ e $I_R(x,y)$, $I_G(x,y)$ e $I_B(x,y)$ son las componentes rojo, verde y azul de $I(x,y)$, respectivamente, representados como un valor entero de 8 bits entre 0 y 255.

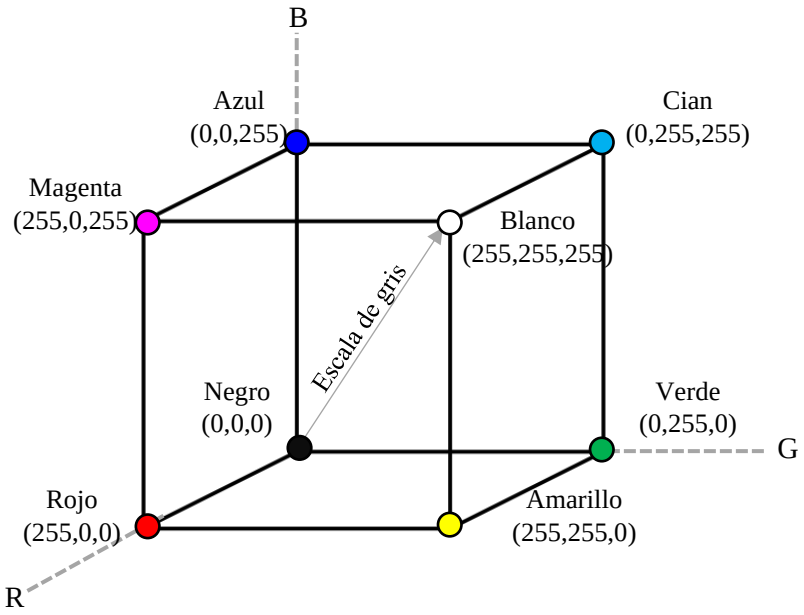


Figura 1.1. Espacio de color RGB.

Mientras que el espacio de color RGB es apropiado para la detección y despliegue de los colores, no es el mejor para otras tareas como la segmentación debido al hecho de que es redundante. La alta correlación entre cada uno de los canales de este espacio de color lo hacen inapropiado para la segmentación por color [9].

El espacio de color HSI ha sido utilizado en el procesamiento de imágenes debido al hecho de que puede separar la crominancia de los valores de intensidad en la imagen. El primer componente *hue* describe el color principal mientras que el componente *saturation* indica el tono o la pureza del color. El componente final, *intensity*, describe el brillo en la imagen [9]. En la Figura 1.2 se muestra la representación tridimensional del espacio de color HSI.

El contenido de color de una imagen puede ser representado mediante un histograma donde se mide la frecuencia de ocurrencia de un determinado valor de color en los pixeles de una imagen. Como es sabido, los histogramas de color no incluyen información espacial,

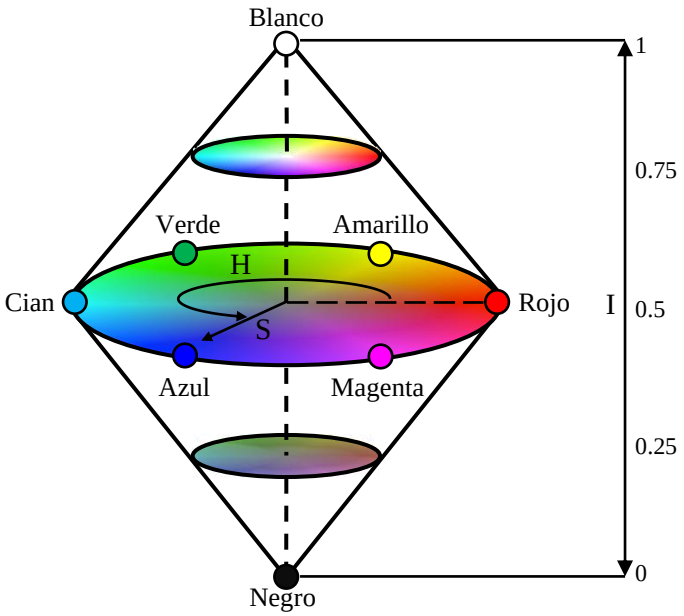


Figura 1.2. Espacio de color HSI.

por lo que tienen la desventaja de no ser robustos ante los cambios de apariencia significantes. Sin embargo, son relativamente invariantes a la translación, rotaciones sobre el eje de la imagen, rotaciones pequeñas fuera del eje, cambios de escala y oclusiones parciales [8].

La textura es otra característica que puede ser utilizada para la representación de la apariencia del objeto de interés. La textura puede definirse como la medición de intensidad de la variación de la superficie que cuantifica las propiedades como la suavidad y la regularidad de esta. En comparación con el color, la textura requiere de un procesamiento adicional para generar los descriptores y además es menos sensitiva a los cambios de iluminación [4].

1.3.1.2 Representación basada en forma

De acuerdo con Yilmaz *et al.* [4] las representaciones basadas en forma más utilizadas en el seguimiento de objetos son las siguientes:

- **Puntos**. El objeto puede ser representado por un solo punto, como por ejemplo el centroide de este, o un conjunto de puntos. En general, la representación por puntos es adecuada para el seguimiento de objetos que ocupan pequeñas regiones en la imagen.

- **Formas geométricas primitivas**. El objeto es representado mediante figuras geométricas como rectángulos o elipses. Esta representación es más adecuada para objetos rígidos, es decir, aquellos que no sufren cambios en su forma. No obstante, también se puede utilizar para objetos no rígidos debido a la sencillez y simplicidad en su construcción.
- **Silueta y contorno del objeto**. El contorno de un objeto define la frontera de este. Por otro lado, la región interna del contorno es a lo que se llama silueta. Este tipo de representaciones son adecuadas para el seguimiento de figuras complejas no rígidas.
- **Modelos de formas articuladas**. Los objetos articulados son aquellos que están compuestos por elementos que están unidos por articulaciones o junturas. Un ejemplo de esto es el cuerpo humano, ya que los distintos elementos que lo conforman como torso, piernas, brazos, cabeza y pies están conectados por articulaciones. Una manera de construir un modelo de este tipo es a través de la representación de cada elemento que lo constituye por cilindros o elipses.
- **Modelos de esqueleto**. Un esqueleto representa la forma de un objeto con un número relativamente pequeño de píxeles, de tal manera que todos los píxeles del esqueleto son estructuralmente necesarios. Este tipo de representación puede ser utilizada para objetos articulados y rígidos.

La elección de alguna de las representaciones descritas anteriormente, depende principalmente del tipo de objeto de interés, es decir, si es rígido o articulado. Otro factor que también puede influir en esta decisión es que tan robusta se requiere esta representación, por ejemplo, si el objeto de interés se encuentra inmerso en un escenario con pocos objetos a su alrededor y de iluminación controlada, probablemente la elección de una representación utilizando puntos sea suficiente para generar un algoritmo cuya estabilidad y resultados se encuentren dentro de rangos aceptables, por otro lado, si se trata de un escenario complejo lo mejor sería elegir una representación más completa que involucre más características. En la Figura 1.3 se ilustran cada una de las representaciones basadas en forma que han sido descritas.

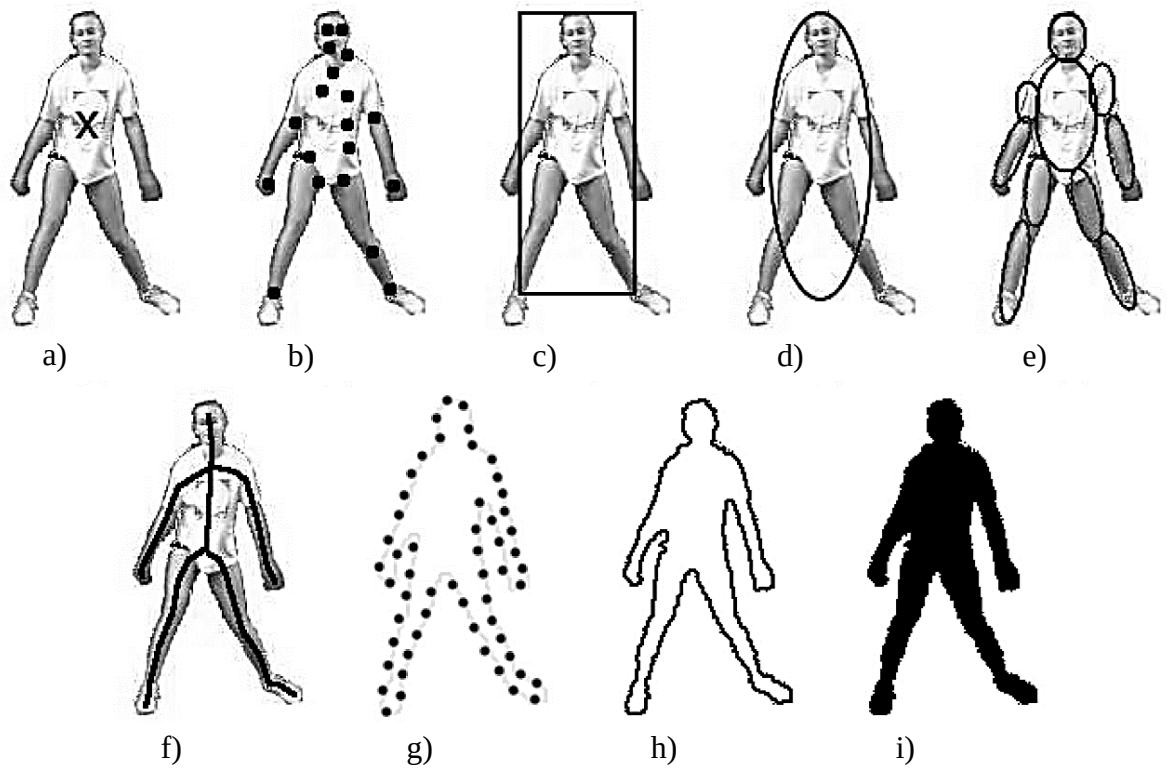


Figura 1.3. Representaciones de objetos basadas en forma: a) centroide, b) conjunto de puntos, c) forma rectangular, d) forma elíptica, e) múltiples figuras, f) esqueleto, g) puntos de control en el contorno, h) contorno completo e i) silueta [4].

1.4 Selección y extracción de características

Una característica es definida como una representación local y significativa del objeto o parte del objeto de interés [10], es una función de una o más mediciones, cada una de las cuales especifica alguna propiedad cuantificable de un objeto [11].

La extracción de características es el proceso para generar características mientras que la selección de características es el proceso que reduce el número de estas [11]. El resultado de estos dos procesos es un conjunto de características que se utilizan para describir a un objeto.

De acuerdo con [10] el conjunto de características seleccionadas para la descripción del objeto debe cumplir con los siguientes aspectos:

- Debe ser exclusivo del objeto de interés y significativo para la aplicación bajo consideración. En el seguimiento de objetos, se desea un conjunto de características que

identifique de forma única al objeto de interés de entre otros en escena. Por ejemplo, una mala selección de característica cuando se desea el seguimiento de una cebra que se mueve entre una manada de cebras sería el color, dado que todas las cebras comparten el mismo patrón blanco y negro.

- Debe ser descriptivo, pero a la vez debe ser flexible. El objeto de interés es susceptible a deformaciones, rotaciones, cambios de iluminación y variaciones en la escala, entre otros factores. Es deseable que un algoritmo de seguimiento de objetos pueda funcionar correctamente aun ante dichos cambios. La selección de características debe generar un conjunto que sea robusto o invariante a estas transformaciones que puede sufrir el objeto de interés.
- El último requisito para el conjunto de características es que la extracción de estas debe ser lo más eficiente posible, ya que la mayoría de los algoritmos de seguimiento de objetos se esfuerzan por funcionar en tiempo real.

Determinar qué tipo de características es conveniente utilizar para describir al objeto de interés es una de las tareas más importantes en el diseño de un algoritmo de seguimiento de objetos e incluso algunos autores concuerdan que de esta selección depende en gran medida el éxito del algoritmo [12], [13].

1.4.1 Tipos de características

Las características pueden ser clasificadas como *hand-crafted* y *non-handcrafted* [14]. A continuación, se describirán cada una de ellas.

Las características *hand-crafted* son aquellas que han sido diseñadas manualmente. Su diseño regularmente involucra encontrar un equilibrio entre exactitud y eficiencia computacional. Un ejemplo es *Scale Invariant Feature Transform* (SIFT) que es conocida por su robustez ante las rotaciones en el objeto y las variaciones de escala, sin embargo, su robustez implica un alto costo computacional.

Por otro lado, las características *non-handcrafted* provienen del uso de un paradigma de aprendizaje profundo (*deep learning*). Este paradigma permite la creación de redes complejas para darle solución a diferentes problemas, por ejemplo, el problema de clasificación de imágenes. Usualmente estas redes están en forma de redes neuronales convolucionales, entre

otras, donde las capas profundas (*deep layers*) de estas redes complejas actúan como un conjunto de extractores de características que a menudo son genéricos y hasta cierto punto independientes de alguna otra tarea de clasificación. Esto significa que el paradigma de aprendizaje profundo obtiene un conjunto de características aprendidas directamente por las observaciones de las imágenes de entrada.

La diferencia entre las características *non-handcrafted* y las *hand-crafted* es que las primeras son obtenidas de forma autónoma y en cambio las segundas son diseñadas previamente por personas expertas para extraer el conjunto de características elegidas.

Debido a que las características *non-handcrafted* extraídas por una red neuronal convolucional (CNN, por sus siglas en inglés) dependen en gran medida del conjunto de datos de entrenamiento, se debe tener especial cuidado en la selección de este. Por lo regular un conjunto de datos de entrenamiento representativos se obtiene al producir o seleccionar una gran cantidad de imágenes, en el orden de millones. Esta solución, sin embargo, está lejos de ser ideal, ya que requiere un esfuerzo humano considerable en la selección de imágenes adecuadas para construir el conjunto de datos de entrenamiento y un costo computacional sustancial durante la fase de entrenamiento. Estos inconvenientes pueden superarse parcialmente mediante la explotación de técnicas semisupervisadas y no supervisadas para reducir el trabajo humano y mediante el uso de computación paralela para disminuir los costos computacionales involucrados en el entrenamiento.

1.5 Seguimiento de objetos

El seguimiento de objetos es una de las áreas de mayor importancia en la visión por computadora con múltiples aplicaciones tales como interacción humano-computadora, video vigilancia inteligente, control de tráfico, reconocimiento de actividad humana, conducción automática, robótica industrial y video juegos. El seguimiento de objetos consiste en determinar o estimar la posición del objeto de interés en movimiento dentro de una secuencia de video [15]. Se trata de un área que ha recibido gran atención y que se ha trabajado por varios años. Muchos investigadores la consideran entre las áreas de mayor desafío dentro de la visión por computadora puesto que en ella se presentan gran cantidad de dificultades que tienen que ser consideradas para garantizar la robustez y eficiencia del algoritmo o método desarrollado.

Es común utilizar la información visual que proviene de las cámaras de video para el desarrollo de los algoritmos de seguimiento de objetos. El objetivo primordial de estos es conocer el estado de un objeto móvil en todo cuadro de video para con ello generar su trayectoria [16]. Cada algoritmo de seguimiento de objetos requiere de un método de detección ya sea para cada cuadro de video o para cuando aparece por primera vez el objeto de interés en la escena [3]. Dentro de la literatura consultada, desafortunadamente no se reporta algún algoritmo de seguimiento de objetos capaz de funcionar bajo cualquier condición, sin embargo, muchos de estos han logrado un gran avance dentro del área de seguimiento de objetos.

1.5.1 Desafíos en el seguimiento de objetos

El área de seguimiento de objetos conlleva el manejo de situaciones críticas consideradas como desafíos que dificultan la realización de la tarea principal del algoritmo de seguimiento de objetos. En [17] se mencionan 8 situaciones críticas que se describen a continuación:

- **Oclusiones.** Ocurren cuando un objeto rastreado o sus atributos clave utilizados para reconocer su identidad no están disponibles para que un sensor de cámara siga rastreando su estado espacial mientras el objeto aún está presente en la escena [18]. De acuerdo con [15] las oclusiones pueden ser clasificadas como autooclusión, oclusión entre objetos y oclusión por la estructura del escenario.
- **Cambio de apariencia.** Ocurre en la mayoría de los objetos, especialmente en los no rígidos. Se refiere al cambio que sufren estos por las nuevas posiciones que adquieren sus componentes a lo largo del movimiento.
- **Fondo abarrotado.** Un fondo abarrotado se refiere a aquel escenario que contiene gran cantidad de elementos. Esta condición puede incentivar otros problemas como la presencia de objetos similares o que sucedan oclusiones con mayor frecuencia.
- **Escalamiento.** Este evento ocurre cuando los objetos se mueven hacia la cámara o se alejan de ella haciendo que su tamaño en el cuadro de video aumente o disminuya respectivamente.

- **Variación de la iluminación.** La variación de la iluminación ocurre cuando los objetos se acercan o alejan de una fuente de iluminación o cuando esta se ve modificada, provocando que muchas de las características de los objetos sean alteradas.
- **Ruido en la imagen.** Se refiere a la presencia de una variación aleatoria no deseada en los valores de los pixeles de las imágenes. Puede deberse por varias razones, por ejemplo, un mal funcionamiento en el sistema de adquisición o un mal procesamiento en las imágenes. Para estos casos se requiere de un preprocesamiento que permita la eliminación del ruido en la imagen.
- **Objetos similares.** Este problema se presenta cuando en la escena existen elementos similares al objeto de interés lo que podría crear confusión acerca de cuál es el objeto que realmente se quiere seguir.
- **Movimientos erráticos o complejos.** Se refiere a cuando el objeto de interés es imprevisible o caprichoso en su movimiento, lo que provoca que se requieran técnicas más avanzadas para la generación de un algoritmo capaz de manejar y lidiar con tales movimientos.

1.5.2 Metodologías en el seguimiento de objetos

Como ya se ha mencionado anteriormente, un seguidor de objetos busca obtener el estado de estos a lo largo de una secuencia de video. Para lograrlo, se han propuesto e implementado distintos enfoques, no obstante, de acuerdo con los autores J. Joshan Athanesious y P. Suresh [19] podemos categorizarlos de acuerdo con las siguientes metodologías: seguimiento puntual, seguimiento por *kernel* y seguimiento por silueta. A continuación, se describirá en que consiste cada una de estas.

- **Seguimiento puntual.** Esta metodología representa a los objetos en movimiento mediante puntos característicos. Este método es apropiado cuando el objeto de interés es de tamaño reducido, sin embargo, su seguimiento se vuelve un problema complejo durante la incidencia de oclusiones [3], [19]. Algunos ejemplos de métodos que utilizan el seguimiento puntual son el filtro de Kalman y el filtro de partículas.

- **Seguimiento por *kernel*.** En este tipo de metodología la palabra *kernel* se refiere a una máscara en forma de figura geométrica utilizada para la representación del objeto, generalmente rectangular o elíptica. El seguimiento del objeto se realiza mediante el cálculo del movimiento de la máscara en cada cuadro de video [20]. Una de las desventajas de esta metodología es que la figura que representa al objeto no coincide en su totalidad con la forma de este, dejando partes importantes fuera de la máscara y partes del fondo dentro de esta [3], [19]. Algunos ejemplos de métodos de seguimiento de objetos basados en esta metodología son comparación de plantillas (*template matching*) [21], *mean shift* y *support vector machine* (SVM) [22].
- **Seguimiento por silueta.** Muchos objetos poseen formas complejas que difícilmente pueden ser representadas por alguna figura geométrica [3], [19]. Los métodos basados en silueta proporcionan una descripción más precisa sobre la forma de los objetos. El propósito de un método de seguimiento basado en esta metodología es encontrar la región del objeto en cada cuadro de video por medio de un modelo generado por los cuadros anteriores [20]. Comparación de forma (*shape matching*) [23] y comparación de contorno (*contour matching*) [24] son algunos ejemplos de métodos que siguen esta metodología.

1.6 Revisión de la literatura sobre los algoritmos de seguimiento de objetos

Para dar una idea general sobre el avance actual dentro de la investigación en el área de seguimiento de objetos se llevó a cabo una revisión de literatura y se seleccionaron aquellos trabajos más recientes, de preferencia no anteriores al año 2015, cuyo enfoque es el desarrollo de métodos, técnicas y algoritmos orientados al seguimiento de objetos, en cualquiera de sus modalidades, ya sea seguimiento único o múltiple de objetos. Para ello se utilizaron las bases de datos *IEEEXplore* de IEEE, *ScienceDirect* de Elsevier y *SpringerLink* de Springer. Estos artículos seleccionados se clasificaron de acuerdo con la técnica principal empleada, quedando las siguientes categorías: algoritmos basados en filtros, algoritmos basados en *tracking-by-detection*, algoritmos basados en aprendizaje profundo y algoritmos basados en la combinación de varios enfoques. En las siguientes secciones se describirán brevemente estos trabajos encontrados para cada categoría.

1.6.1 Algoritmos basados en filtros

Esta categoría es la más amplia en cuanto a trabajos encontrados, en ella entran todos aquellos basados en filtros de Kalman, filtros de partículas, filtros de correlación y filtros armónicos.

1.6.1.1 Filtros de partículas

Gran parte de los trabajos encontrados en esta subcategoría se centran en solucionar o mejorar la metodología en la que consiste el filtro de partículas. A continuación, se describen estos trabajos.

Naushad Ali *et al.* [25] presentan un método para el seguimiento de múltiples objetos. El objetivo principal de este trabajo es realizar el seguimiento en secuencias de video cuando las partes del objeto de interés se ocluyen debido a elementos del fondo u otros objetos que de igual manera se siguen. Para ello hacen uso de las características *salient feature points* (SFP) y de una aproximación del filtro de partículas.

Huaping Liu y Fuchun Sun [6] proponen un método para el seguimiento de objetos donde se utiliza el filtro de partículas basado en *Markov Chain Monte Carlo* (MCMC) que determina el peso de cada partícula usando la técnica de cálculo de probabilidad incremental. El objetivo de este trabajo es presentar un método para calcular el peso de las partículas de tal manera que el costo computacional sea menor en comparación con el enfoque tradicional.

Por otra parte, Daneshyar y Nahvi [5] proponen un algoritmo para el seguimiento de objetos que utiliza un filtro de partículas modificado basado en la eliminación de las partículas sin importancia con el objetivo de mejorar la precisión y el tiempo de cómputo del algoritmo. La modificación propuesta sobre el filtro de partículas consiste en la creación de una máscara binaria con base en dos métodos: diferenciación de cuadros (*frame differencing*) y mezcla gaussiana (*gaussian mixture*). Esta máscara determina que pixeles son del fondo y cuáles del primer plano, siendo este último el que contiene al objeto de interés. De esta manera, las partículas ubicadas en el fondo de la máscara binaria son eliminadas.

Gwangmin Choe *et al*. [26] introducen los conceptos de *spline resampling* y *global transition model*, que son métodos que ayudan a resolver los problemas ocasionados por el filtro de partículas tradicional como el empobrecimiento y degeneración de partículas, y el

fondo dinámico ocasionado por el movimiento de la cámara, respectivamente, en una aplicación de seguimiento de objetos.

Mark David Jenkins *et al*. [27] utilizan un marco de filtro de partículas de muestreo de importancia selectiva (SSIR) en combinación con el filtro autoregresivo lineal (AR) de estimación de movimiento. Este último predice la posición del objeto basado en las trayectorias previas y con ello maximiza la probabilidad de muestreo sobre la región de la imagen que posiblemente contiene al objeto, ya que, en situaciones complejas, el modelo de apariencia puede no ser robusto como para proporcionar los pesos a las partículas o para que estos sean lo suficientemente precisos como para generar una distribución densa de partículas alrededor de la ubicación real del objeto.

Issam Elafi *et al*. [28] presentan un método para el seguimiento de múltiples objetos. El propósito de este trabajo es crear un sistema capaz de detectar y seguir automáticamente todos los objetos en movimiento sin información previa sobre estos. El método consiste en dos partes: el proceso de detección y el proceso de seguimiento. Para el primero se utiliza el método de sustracción de fondo y para el segundo se hace uso del filtro de partículas. Para este último se integra la característica de color en forma de histograma que servirá para darle la capacidad al sistema de manejar situaciones de escalamiento y rotación.

Por otra parte, Zhou *et al*. [29] presentan un algoritmo de seguimiento de objetos que combina el filtro de partículas y la representación esparcida para utilizar tanto la información global como local y crear un modelo dinámico. El objetivo principal es encontrar un método que maneje las variaciones de iluminación, cambios de apariencia y oclusiones parciales en el objeto de interés.

1.6.1.2 Filtros de correlación

Los filtros de correlación han sido muy utilizados dentro del área de seguimiento de objetos especialmente por su alta eficiencia computacional. Kai Chen *et al.* [30] presentan un método basado en el uso de los filtros de correlación donde se abordan principalmente los inconvenientes que poseen estos causados por el desplazamiento circular que se lleva a cabo durante su implementación. El método presentado en [30] consiste en un filtro de correlación estructurado constituido por varios de estos: uno para la región holística y el resto para las

regiones locales generadas. Estos filtros se ponderan en función de la confianza que tienen de poder estimar la localización del objeto de interés sobre todo en situaciones de oclusión y deformaciones. Además, se implementa un modelo para suprimir el fondo al mismo tiempo que se enriquece la zona del objeto de interés mediante un histograma de color basado en el clasificador de Bayes. Esto hace que los filtros de correlación sean menos sensibles al contexto del fondo.

1.6.1.3 Filtros armónicos

Los algoritmos de búsqueda de armonía han sido aplicados en otras áreas como ingeniería mecánica, civil o eléctrica como métodos eficientes de optimización. De igual manera, han sido implementados en áreas de visión por computadora y segmentación de imagen. Jaco Fourie *et al.* [31] presentan el método de filtro armónico basado en el algoritmo de búsqueda de armonía para realizar la tarea de seguimiento de objetos. La arquitectura del filtro armónico consiste en dos componentes: el seguidor y el optimizador. El seguidor crea una plantilla con el histograma de color bidimensional del objeto de interés utilizando las componentes *hue* y *saturation* del espacio de color HSV. Esta plantilla se envía al optimizador de búsqueda de armonía que inicia un proceso iterativo para determinar la ubicación del objeto de interés.

1.6.1.4 Filtros de Kalman

Los filtros de Kalman y sus variantes han sido ampliamente utilizados para aplicaciones que involucran el seguimiento de objetos.

Mirunalini *et al*. [15] utilizan el filtro de Kalman junto con las características SIFT para verificar la presencia de un objeto en secuencias de video y seguirlo en entornos ocluidos y no ocluidos. Por otro lado, Wu-Chih Hu *et al*. [32] proponen un esquema de seguimiento de objetos basado en el filtro de Kalman que utiliza el centro de gravedad de la región del objeto en movimiento delimitado por un rectángulo.

1.6.2 Algoritmos basados en *tracking-by-detection*

Los algoritmos basados en *tracking-by-detection* emplean un clasificador entrenado para distinguir el objeto de interés del escenario que lo rodea [22]. Junwei Li *et al.* [22] hacen uso

de SVM con salida estructurada como clasificador que mitiga la incertidumbre en las muestras cuando se etiquetan como positivas o negativas.

Otro algoritmo basado en *tracking-by-detection* es el presentado por Ruochen Fan *et al.* [7] que se enfoca en resolver el problema de oclusión, sobre todo cuando esta tiene una duración considerable, y en completar trayectorias discontinuas presentadas durante los momentos de oclusión. El método está conformado por un módulo seguidor que hace uso de la CNN llamada fast-RCNN que detecta objetos similares al objeto de interés y SVM como módulo selector que discrimina entre estos objetos y determina cuál se debe utilizar para reiniciar al seguidor.

1.6.3 Algoritmos basados en aprendizaje profundo

El aprendizaje profundo es un subcampo específico del aprendizaje automático (*machine learning*). Una de las técnicas más utilizadas actualmente basada en el aprendizaje profundo son las CNN. Algunos algoritmos de seguimiento de objetos se basan en el uso de una CNN como mecanismo extractor de características en conjunto con algún otro enfoque para llevar a cabo el seguimiento de objetos, tal es el caso del método presentado por Chao Ma *et al.* [33] en el que se utilizan múltiples filtros de correlación y características extraídas de las capas convolucionales de la CNN. El objetivo principal del método es aprovechar la información semántica de la última capa y la información espacial de las capas anteriores de la CNN.

De forma similar, Peng Zhang *et al.* [34], hacen uso de características extraídas por una CNN y en conjunto con los filtros de correlación codifican la información espacial del cuadro. La localización espacial del objeto es aquella donde se encuentre el máximo valor de la respuesta de correlación de apariencia. Por otro lado, mediante el uso de la extracción *motion-saliency* se obtiene el mapa de respuesta de movimiento entre cuadros. Estas dos informaciones, espacial y temporal se utilizan para determinar la localización final del objeto mediante una función que utiliza un proceso de optimización de energía.

Otro trabajo para el seguimiento de objetos basado en el aprendizaje profundo es el propuesto por Daniel Gordon *et al*.[35]. Ellos presentan un algoritmo que utiliza las capas convolucionales para generar la apariencia del objeto, capas recurrentes para recordar información de apariencia y movimiento, y una capa de regresión para generar la ubicación del

objeto. La red utilizada LSTM está entrenada con una combinación de videos reales y sintéticos, lo que le da al método la capacidad de manejar diferentes tipos de objetos.

1.6.4 Enfoques combinados

Dentro de los trabajos consultados existen algunos que combinan varios enfoques para la creación de un método más robusto, tal es el caso de Jing Wang *et al*. [36] quienes presentan un algoritmo para el seguimiento de objetos mediante el filtro de partículas y un modelo de representación del objeto con características personalizadas. La construcción de este modelo se realiza con las características CNN que describen las propiedades generales del objeto, un histograma de gradientes orientados (HOG) y la característica de color que guía la construcción del modelo. La inclusión de esta última tiene doble propósito: el primero de ellos consiste en determinar el grado de similitud en el color entre el fondo y el objeto de interés y de acuerdo con este definir al objeto como fuerte o débil. Así, si se trata de un objeto fuerte la representación de este contendrá los tres elementos mencionados anteriormente de lo contrario solo consistirá en las características CNN y HOG. El segundo propósito es la descripción de distribución de color en el objeto. Además, presentan un método para la reducción del costo computacional que se lleva durante el muestreo de las partículas en el filtro, donde estas son ponderadas con base en la distancia y además son filtradas, eliminando aquellas que superen el rango de distancia máxima establecido entre estas y el resultado de la iteración anterior.

Jun Sun *et al*. [21] presentan un método que combina la comparación de plantillas con el filtro de Kalman para el seguimiento de objetos veloces. El método busca resolver el cambio de apariencia en los objetos cuando estos se mueven a gran velocidad y el alto costo computacional generado por el tradicional mecanismo de comparación de plantillas. El primer problema es abordado mediante la creación de plantillas correlacionadas y el segundo mediante el uso de un filtro de Kalman mejorado que estima la localización del objeto y con la ayuda de una función de evaluación de similitud modificada se encuentra aquel que es compatible, pudiendo así reducir el costo computacional.

Otro trabajo que se basa en la combinación de enfoques es el presentado por Xiao Yun y Gang Xiao [37] quienes proponen un método para el seguimiento de objetos que considera información visual que describe las propiedades visuales del objeto de interés y lo ubica en

cada cuadro con la ayuda de un clasificador bayesiano e información de movimiento estimada por un filtro de Kalman.

Otro trabajo dentro de esta categoría es el propuesto por Xin Zhang *et al*. [38] quienes presentan un método que utiliza filtros de correlación para la estimación del estado del objeto de interés que consiste en la posición y escala de este. Adicionalmente se incluye un clasificador de regresión logística que realiza la tarea de detectar nuevamente al objeto de interés cuando el resultado de la posición de los filtros de correlación no son los esperados. El conjunto de filtros de correlación se crea con base en las características HOG y atributos de color e intensidad de la imagen.

1.7 Conclusiones

En este capítulo se trataron algunos de los conceptos básicos para el entendimiento del seguimiento de objetos. También se mostraron varios de los trabajos realizados en el área y el enfoque principal que adoptan. Se ha explicado la importancia que tiene la selección de una representación y un conjunto de características descriptivo para realizar el modelado de la apariencia del objeto de interés y con ello diferenciarlo del resto de la escena, así como también la importancia de utilizar un modelado de movimiento. La selección de cada uno de los elementos que conforman a un algoritmo de seguimiento de objetos se ve afectada directamente por la escena donde se mueve el objeto, la naturaleza de este mismo y las necesidades de la aplicación.

En los capítulos siguientes se dará a conocer el desarrollo del algoritmo de seguimiento de objetos de esta investigación.

CAPÍTULO II. INSTRUMENTO DE EVALUACIÓN DE MÉTODOS

En esta tesis se presenta un algoritmo de seguimiento de objetos obtenido a partir de una metodología de diseño propuesta. El diseño del algoritmo inicia con la determinación de su enfoque principal que es obtenido mediante un instrumento de evaluación de métodos aplicados al seguimiento de objetos. Este instrumento tiene como propósito establecer cuáles métodos presentan las mejores características y por lo tanto que enfoque reporta mejores resultados. A continuación, se describirá el instrumento de evaluación propuesto.

2.1 Diseño del instrumento de evaluación

El instrumento de evaluación (IE) evalúa un método aplicado al seguimiento de objetos mediante el análisis de su artículo de revista/conferencia o cualquier otra fuente de información escrita donde se describa este. La evaluación involucra tres criterios: documentación (*Do*), desempeño (*De*) y desafíos superados (*DS*). El criterio *Do* evalúa el nivel de detalle acerca del método en la fuente de información escrita, el criterio *De* evalúa el desempeño numérico y la validación de este y, por último, el criterio *DS* evalúa la robustez del método en situaciones desafiantes como oclusiones, deformaciones y fondos abarrotados. Cada uno de estos criterios posee una puntuación que se pondera de acuerdo con el nivel de importancia que tiene en el IE. La ponderación es establecida en uno para *Do*, dos para *DS* y tres para *De*. La Ec. (2.1) describe la puntuación final (*PF*) que tendrá el método que se evalúa con el IE.

$$PF = Do + 3De + 2DS \tag{2.1}$$

A continuación, se dará una descripción detallada sobre los criterios *Do*, *De*, *DS* y las métricas que los componen. Por simplicidad considere a FI como la fuente de información escrita donde el autor describe al método.

2.1.1 Documentación

Algunos autores revelan más detalles sobre sus métodos que otros en las FI, siendo algunos de estos detalles puntos clave para el crecimiento intelectual en el área. Por ello, el criterio *Do* evalúa el nivel de detalle dado por el autor acerca del método. Algunos puntos tratados en este criterio tienen relación al desempeño del método, pero no deben de confundirse con el criterio

De ya que estos puntos son tratados desde una perspectiva de mención y no desde una cuantitativa.

El criterio *Do* está dividido en 4 subcriterios nombrados de la siguiente manera: procesamiento, reproducibilidad, evaluación y comparación. A continuación, se describirán estos y se darán a conocer las métricas utilizadas para cada uno.

1. Procesamiento: el subcriterio de procesamiento denotado por Do_{Pr} evalúa si se da a conocer aquella información sobre cómo funciona el método y sobre los datos utilizados para generar los resultados. Los puntos evaluados son los siguientes:
   a) Indica velocidad de procesamiento, Pr_v: esta métrica evalúa si en la FI se da a conocer la velocidad de procesamiento alcanzada por el método, ya sea en cuadros por segundo (FPS, por sus siglas en inglés) o segundos por cuadro. La puntuación de Pr_v se define por la Ec. (2.2).

$$Pr_v = \begin{cases} 1 & si\ se\ indica\ la\ velocidad\ alcanzada \\ 0 & de\ otra\ forma \end{cases} \tag{2.2}$$

   b) Cantidad de cuadros en los videos, Pr_{cc}: esta métrica evalúa en qué grado la FI da a conocer la cantidad de cuadros que conforman cada uno de los videos utilizados en la evaluación y diseño del método. Este dato se asume conocido para los videos provenientes de una base de datos disponible. La Ec. (2.3) define la puntuación para Pr_{cc} en la que se considera el número de videos en los que se da a conocer la cantidad de cuadros (V_{cc}) *y* la cantidad total de videos utilizados (V).

$$Pr_{cc} = \frac{V_{cc}}{V} \tag{2.3}$$

   c) Resolución de los videos, Pr_r: esta métrica evalúa en qué grado la FI da a conocer la resolución de cada uno de los videos utilizados en la evaluación y diseño del método. Este dato se asume conocido para los videos que provienen de una base datos disponible. Su puntuación considera la cantidad de videos en los que se da a conocer la resolución (V_r) como se muestra en la Ec. (2.4).

$$Pr_r = \frac{V_r}{V} \tag{2.4}$$

d) Espacio de color, Pr_{ec}: esta métrica evalúa si la FI da a conocer el espacio de color sobre el cual trabaja el método. Para evitar ambigüedades se debe indicar explícitamente este dato, de lo contrario se dará una puntuación de cero como lo expresa la Ec. (2.5).

$$Pr_{ec} = \begin{cases} 1 & \textit{si se indica explícitamente el espacio de color} \\ 0 & \textit{de otra forma} \end{cases} \tag{2.5}$$

e) Procesamiento en tiempo real, Pr_{tr}: esta métrica evalúa si el método es capaz de funcionar a una velocidad de tiempo real. Para obtener el puntaje correspondiente se debe indicar de manera explícita esta capacidad en la FI como se muestra en la Ec. (2.6)

$$Pr_{tr} = \begin{cases} 1 & \textit{si se indica explícitamente el procesamiento en tiempo real} \\ 0 & \textit{de otra forma} \end{cases} \tag{2.6}$$

f) Uso de un esquema de entrenamiento, Pr_{ee}: esta métrica evalúa si el método utiliza o no un esquema de entrenamiento para su funcionamiento, siendo la respuesta positiva desfavorable para la puntuación, ya que el uso de un entrenamiento implica mayor cantidad de recursos necesarios para su funcionamiento, por ejemplo, imágenes de entrenamiento y tiempo invertido, lo que desfavorecen al método. La Ec. (2.7) muestra el puntaje para Pr_{ee}.

$$Pr_{ee} = \begin{cases} 1 & \textit{si no utiliza un esquema de entrenamiento} \\ 0 & \textit{de otra forma} \end{cases} \tag{2.7}$$

g) Cantidad de parámetros que utiliza, Pr_{cp}: es común que los algoritmos requieran de una configuración de parámetros previa a la ejecución. Si esta cantidad es grande, entonces el usuario se verá en la difícil tarea de ajustar cada uno de ellos para lograr un funcionamiento óptimo a diferencia de los algoritmos que requieren del ajuste de 1 o 2 parámetros. Es por eso, que esta métrica evalúa la cantidad de parámetros que se utiliza en el método (Pa) reportada en la FI en función de una cantidad máxima de parámetros (P_{max}) establecida por el evaluador. La puntuación dada a Pr_{cp} está definida por la Ec. (2.8).

$$Pr_{cp} = \begin{cases} 1 & \text{Si } Pa \leq P_{max} \\ \dfrac{P_{max}}{Pa} & \text{Si } Pa > P_{max} \end{cases} \tag{2.8}$$

A partir de las métricas descritas anteriormente, el subcriterio de Do_{Pr} tendrá una puntuación que es equivalente a la suma de los resultados obtenidos, como lo muestra la Ec. (2.9).

$$Do_{Pr} = Pr_v + Pr_{cc} + Pr_r + Pr_{ec} + Pr_{tr} + Pr_{ee} + Pr_{cp} \tag{2.9}$$

2. Reproducibilidad: el subcriterio de reproducibilidad denotado por Do_R evalúa si la FI del método reporta la información necesaria para realizar la réplica de este. Los puntos evaluados son los siguientes:
   a) Infraestructura utilizada, R_{in}: esta métrica evalúa la mención de la infraestructura empleada durante el tiempo de diseño y evaluación del método. Se consideran 4 elementos que la conforman: el procesador, la memoria de acceso aleatorio (RAM, por sus siglas en inglés), el CPU o GPU y el *software* o lenguaje de programación empleado, cuyas abreviaturas son *pro*, *ram*, *cg y sl*, respectivamente. La puntuación para cada una de estas es descrita por la Ec. (2.10), donde *ei* hace referencia al elemento de infraestructura *pro*, *ram*, *cg* o *sl*.

$$ei = \begin{cases} 1 & si\ se\ indica\ el\ elemento\ de\ infraestructura \\ 0 & de\ otra\ \ forma \end{cases} \tag{2.10}$$

   De esta manera, el valor de la métrica de R_{in} equivale a la suma de las puntuaciones de los *ei* mencionados, como se muestra en la Ec. (2.11).

$$R_{in} = pro + ram + cg + sl \tag{2.11}$$

   b) Configuración de los parámetros, R_{cfg}: esta métrica evalúa la cantidad de parámetros en los que se indica su valor configurado (P_{cfg}) en la FI sobre el total de estos (Pa). Su puntuación está dada por la Ec. (2.12).

$$R_{cfg} = \frac{P_{cfg}}{Pa} \tag{2.12}$$

De esta manera, el subcriterio Do_R tendrá la puntuación equivalente a la suma de los resultados obtenidos por las métricas R_{in} y R_{cfg}, como se muestra en la Ec. (2.13).

$$Do_R = R_{in} + R_{cfg} \tag{2.13}$$

3. Evaluación: el subcriterio de evaluación denotado por Do_{Ev} involucra los elementos utilizados para llevar a cabo la evaluación del método. Se consideran los siguientes puntos:

a) Cantidad de base de datos, Ev_{bd}: esta métrica evalúa la cantidad de bases de datos utilizadas durante la evaluación del método y las divide en dos grupos: bases de datos que se usaron de manera completa (*bdc*) y bases de datos que se usaron de manera incompleta (*bdi*). La suma de estas da a conocer el número total de bases de datos utilizadas (*bd*). Usar una base de datos de forma completa durante la evaluación del método refleja mayor información acerca de su desempeño en comparación a cuando se utiliza una base de datos de manera incompleta. Por ello la puntuación de Ev_{bd} está dada por la Ec. (2.14) que describe que si todas las bases de datos se han utilizado de forma completa entonces Ev_{bd} tendrá valor de 1, pero que si hay bases de datos utilizadas de forma parcial entonces Ev_{bd} será menor ya que *bdi* es multiplicado por 0.5.

$$Ev_{bd} = \frac{1}{bd}\left(bdc + \frac{1}{2} bdi \right) \tag{2.14}$$

b) Evaluación cuantitativa, Ev_{cn}: esta métrica evalúa si se reportan los resultados de una manera cuantitativa en la FI, esto es, a través de resultados numéricos. Su puntuación está dada por la Ec. (2.15).

$$Ev_{cn} = \begin{cases} 1 & \textit{si se realiza una evaluación cuantitativa} \\ 0 & \textit{de otra forma} \end{cases} \tag{2.15}$$

c) Evaluación cualitativa, Ev_{cl}: esta métrica evalúa si se reportan los resultados de una manera cualitativa en la FI, es decir, a través de gráficas o cuadros de video con rectángulos delimitadores que muestren como se va dando el seguimiento del objeto a lo largo de la secuencia de video. La puntuación de esta métrica se muestra en la Ec. (2.16).

$$Ev_{cl} = \begin{cases} 1 & \textit{si se realiza una evaluación cualitativa} \\ 0 & \textit{de otra forma} \end{cases} \tag{2.16}$$

d) Una sola configuración, Ev_{cfg}: esta métrica evalúa si se utilizó la misma configuración de los parámetros durante toda la evaluación del método. Para evitar ambigüedades, este dato debe indicarse de manera explícita en la FI. La puntuación de Ev_{cfg} se muestra en la Ec. (2.17).

$$Ev_{cfg} = \begin{cases} 1 & \textit{si se mantiene una sola configuración} \\ & \textit{durante toda la evaluación} \\ 0 & \textit{de otra forma} \end{cases} \quad (2.17)$$

Así pues, el subcriterio Do_{Ev} tendrá la puntuación equivalente a la suma de los resultados obtenidos por las métricas descritas anteriormente, como se muestra en la Ec. (2.18).

$$Do_{Ev} = Ev_{bd} + Ev_{cn} + Ev_{cl} + Ev_{cfg} \quad (2.18)$$

4. Comparación: el subcriterio de comparación denotado por Do_C evalúa la manera en que esta es llevada a cabo. Se considera que la comparación del algoritmo en el método puede ser contra algoritmos de la misma y/o diferente naturaleza. Es importante que se realicen ambas comparaciones, de manera que de la primera se deduzcan mejoras dentro del mismo enfoque y de la segunda se deduzcan ventajas con respecto a otros. La Ec. (2.19) muestra el puntaje para este subcriterio.

$$Do_C = \begin{cases} 1 & \textit{si se utilizan los dos tipos de comparación} \\ 0.5 & \textit{si se utiliza alguno de los dos tipos de comparación} \\ 0 & \textit{de otra forma} \end{cases} \quad (2.19)$$

De esta manera, el criterio de *Do* recibe el puntaje descrito por la Ec. (2.20).

$$Do = Do_{Pr} + Do_R + Do_{Ev} + Do_C \quad (2.20)$$

2.1.2 Desempeño

El desempeño de un método involucra varios aspectos como el tiempo, precisión y exactitud, y además la validez de estos resultados. En el criterio *De* se evalúan estos aspectos a través de las siguientes métricas:

1. Velocidad de procesamiento, De_{ve}: esta métrica evalúa la velocidad de procesamiento reportada en la FI. Este dato puede estar dado en FPS o en segundos por cuadro, donde para este último es necesario una conversión a FPS. La velocidad se normaliza con base en un valor establecido por el evaluador denotado por fps_{max} que define la cantidad de FPS máxima a partir de la cual el método se categoriza como lo suficientemente rápido. La Ec. (2.21) muestra el puntaje dado para esta métrica, donde *fps* es la velocidad reportada del método.

$$De_{ve} = \begin{cases} \dfrac{fps}{fps_{max}} & \text{Si } fps < fps_{max} \\ 1 & \text{Si } fps \geq fps_{max} \end{cases} \tag{2.21}$$

2. Precisión, De_p: la precisión es una de las métricas utilizadas para medir el desempeño de un algoritmo de seguimiento de objetos. Se basa en el cálculo de la distancia euclidiana entre el centro del rectángulo delimitador producido por el algoritmo y el centro del rectángulo delimitador *ground truth*, del objeto que se sigue. Esta distancia también es conocida como error de localización central (*CLE*) y está dada en pixeles de acuerdo con la Ec. (2.22),

$$CLE = \sqrt{(x_a - x_{gt})^2 + (y_a - y_{gt})^2} \tag{2.22}$$

donde (x_a,y_a) y (x_{gt},y_{gt}) son los centros de los rectángulos delimitadores del algoritmo y *ground truth*, respectivamente.

Algunos autores presentan sus resultados de precisión mediante el promediado de las distancias *CLE* obtenidas en cada cuadro de video (P_P). Otra manera de reflejar la precisión es mediante el porcentaje de cuadros en los que la distancia *CLE* es menor a un umbral establecido (P_U), generalmente de 20 pixeles. Por último, otra forma de mostrar el resultado de precisión consiste en calcular el área bajo la curva (AUC, por sus siglas en inglés) de la gráfica "razón de precisión – umbral", en la que se da a conocer el porcentaje de cuadros en los que la distancia *CLE* se encuentra por debajo del umbral correspondiente (P_A).

Cada una de estas formas para presentar la precisión se utilizan de manera indistinta en la literatura, y cómo es posible imaginar, no pueden ser tratadas por igual, ya que no reflejan el mismo tipo de información. Por ejemplo, el cálculo del promedio es sensible a datos atípicos lo que puede ocasionar que el resultado no represente correctamente la precisión del método. Por otro lado, utilizar un umbral refleja de mejor manera el desempeño ya que da a conocer el porcentaje de triunfo, donde este se define con base en el umbral establecido. De manera similar, el AUC da a conocer la precisión de una manera más completa ya que muestra el nivel de triunfo del método a diferentes umbrales.

Es por ello que en la evaluación de la precisión De_p no solo se considerará su valor reportado en la FI sino también el cálculo que la originó. Se propone la Ec. (2.23) para obtener el puntaje de la métrica De_p dentro de este IE,

$$De_p = \begin{cases} P_P^{'} + w_P & Si\ la\ precisión\ es\ por\ promedio \\ P_U + w_U & Si\ la\ precisión\ es\ por\ umbral \\ P_A + w_A & Si\ la\ precisión\ es\ por\ AUC \end{cases} \quad (2.23)$$

donde P indica el cálculo de la precisión que toma un valor entre cero y uno y w es el peso asignado a la manera de darla a conocer. Los valores de w propuestos son 0.7 para w_P, 0.9 para w_U y 1 para w_A. Estos valores van de acuerdo con la descripción dada acerca de la importancia que tiene cada una de las maneras de presentar los resultados de precisión. Debido a que P_P está dada en pixeles es necesario realizar una normalización sobre esta de manera que tome valores de 0 a 1. Se propone dividir el resultado de P_P entre un umbral de corte (th) y este resultado restarlo a 1. De esta manera, cuando P_P se acerque al umbral, el nuevo valor de la precisión será menor, y cuando P_P se mantenga baja, el nuevo valor será alto. Si P_P supera el umbral, se tendrá un valor de 0. Así, el nuevo valor de P_P normalizado denotado por $P_P^{'}$ queda definido por la Ec. (2.24).

$$P_P^{'} = \begin{cases} 1 - \dfrac{P_P}{th} & \text{Si } P_P \le th \\ 0 & \text{Si } P_P > th \end{cases} \quad (2.24)$$

Si la presentación de los resultados de precisión del método se realiza en más de una manera, solo se contabilizará aquella con mayor peso w.

3. Éxito, De_e: el éxito es una de las métricas utilizadas para medir el desempeño de un algoritmo de seguimiento de objetos. Se basa en el cálculo de la razón de superposición (RS) que mide el grado de superposición entre el conjunto de pixeles de la región *ground truth* (r_{gt}) y el conjunto de pixeles de la región producida por el algoritmo (r_a) a lo largo del video. Este cálculo está definido por la Ec. (2.25),

$$RS = \frac{|\, r_{gt} \bigcap r_a \,|}{|\, r_{gt} \bigcup r_a \,|} \quad (2.25)$$

donde $\cap$ y $\cup$ denotan las operaciones de intersección y unión, respectivamente y $|\cdot|$ denota cardinalidad.

Algunos autores presentan sus resultados de éxito mediante el promediado de las RS obtenidas en cada cuadro de video (E_P). Otra manera de reflejar el éxito es mediante el

porcentaje de cuadros en los que la *RS* es mayor a un umbral establecido (E_U), generalmente de 0.5. Por último, otra forma de mostrar el resultado de éxito consiste en calcular el AUC de la gráfica "razón de éxito – umbral", en la que se da a conocer el porcentaje de cuadros en los que la *RS* se encuentra por encima del umbral correspondiente (E_A).

De la misma manera en cómo sucede con la métrica de precisión, cada una de estas formas de dar a conocer el éxito del método no refleja el mismo tipo de información, por lo que para De_e no solo se considerará el valor del éxito reportado en la FI, sino también el cálculo que lo originó, por lo cual se propone la Ec. (2.26) para obtener el puntaje de esta métrica,

$$De_e = \begin{cases} E_P + w_P & Si\ el\ éxito\ es\ por\ promedio \\ E_U + w_U & Si\ el\ éxito\ es\ por\ umbral \\ E_A + w_A & Si\ el\ éxito\ es\ por\ AUC \end{cases} \tag{2.26}$$

donde *E* es el cálculo del éxito que toma un valor entre cero y uno y *w* es el peso asignado a la manera de dar a conocer el éxito del método. Los valores de *w* son los mismos que los utilizados en la métrica De_p. Si la presentación de los resultados de éxito del método se realiza en más de una manera, solo se contabilizará aquella con mayor peso *w*.

4. Uso de esquemas de evaluación externos, De_{eex}: algunos autores adoptan las métricas o emplean esquemas de evaluación diseñados para determinar el nivel de desempeño de un algoritmo de seguimiento de objetos. El uso de estas métricas y esquemas producen resultados que son más precisos y estándares, ya que son diseñados especialmente para el área de seguimiento de objetos dando paso a que la comparación entre algoritmos sea más simple y directa.

   Por ello, la métrica de De_{eex} evalúa si en la FI del método se reporta el uso de este tipo de esquemas y en qué grado se hace. De acuerdo con la Ec. (2.27) De_{eex} incluye el nivel de uso del esquema (N_E) y la cantidad de esquemas utilizados (*E*).

$$De_{eex} = \begin{cases} \frac{1}{E}\sum_{i=1}^{E} N_{E_i} & si\ utiliza\ esquemas\ de\ este\ tipo \\ 0 & si\ no\ utiliza\ esquemas\ de\ este\ tipo \end{cases} \tag{2.27}$$

donde N_E está definido por la Ec. (2.28) siendo M_u y M_E la cantidad de métricas utilizadas que pertenecen al esquema de evaluación y la cantidad total de métricas en este, respectivamente.

$$N_E = \frac{M_u}{M_E} \tag{2.28}$$

5. Cantidad de métricas utilizadas, De_m: estas métricas son las utilizadas para medir el desempeño del método y que son reportadas en la FI. No se incluyen aquellas que son propuestas por el esquema de evaluación externo, en caso de que se hiciera uso de alguno. La cantidad de métricas utilizadas (Me) es normalizada con respecto a una cantidad de métricas máxima (M_{max}) definida por el evaluador que indica la cantidad de estas suficientes para realizar una evaluación completa. La Ec. (2.29) muestra el puntaje para esta métrica.

$$De_m = \begin{cases} \dfrac{Me}{M_{max}} & \text{Si } Me < M_{max} \\ 1 & \text{Si } Me \geq M_{max} \end{cases} \tag{2.29}$$

6. Cantidad de desafíos, De_d: esta métrica evalúa la cantidad de desafíos mencionados en la FI sobre los cuales el método es puesto a prueba (D). En el área de seguimiento de objetos estos desafíos se muestran en la forma de oclusiones, cambios de apariencia y condiciones de la escena, entre otros. El evaluador define una cantidad máxima de desafíos (D_{max}) suficiente para considerar que el método es evaluado de manera completa, como se muestra en la Ec. (2.30).

$$De_d = \begin{cases} \dfrac{D}{D_{max}} & \text{Si } D < D_{max} \\ 1 & \text{Si } D \geq D_{max} \end{cases} \tag{2.30}$$

7. Cantidad de videos, De_v: aquí se evalúa la cantidad de videos utilizados durante la evaluación del método (V) que se reporta en la FI. V es normalizada con respecto a una cantidad de videos máxima (V_{max}) definida por el evaluador. El uso de pocos videos revela poca información sobre el funcionamiento del método, a diferencia de usar una cantidad considerable de estos que da a conocer la respuesta del método ante un mayor número de situaciones. La puntuación para esta métrica esta descrita mediante la Ec. (2.31).

$$De_v = \begin{cases} \dfrac{V}{V_{max}} & \text{Si } V < V_{max} \\ 1 & \text{Si } V \geq V_{max} \end{cases} \tag{2.31}$$

8. Cantidad de algoritmos utilizados en la comparación, De_a: los algoritmos de comparación son utilizados para demostrar las mejoras y ventajas del algoritmo en el método bajo análisis. A mayor cantidad de algoritmos que formen parte de este grupo de comparación, la evaluación se vuelve más robusta y generalizada. Como se muestra en la Ec. (2.32) la métrica de De_a considera la cantidad de algoritmos utilizados en la comparación (Al) del método y una cantidad máxima de algoritmos de comparación (A_{max}) establecida por el evaluador suficiente para tener una evaluación robusta y generalizada.

$$De_a = \begin{cases} \dfrac{Al}{A_{max}} & \text{Si } Al < A_{max} \\ 1 & \text{Si } Al \geq A_{max} \end{cases} \tag{2.32}$$

Una vez descritas las métricas que conforman el criterio de desempeño De el puntaje de este será dado de acuerdo con la Ec. (2.33).

$$De = De_{ve} + De_p + De_e + De_{eex} + De_m + De_d + De_v + De_a \tag{2.33}$$

2.1.3 Desafíos superados

El criterio de desafíos superados (DS) indica el grado de triunfo que ha tenido el método con base en la cantidad de situaciones críticas que este ha logrado superar. Para ello, es importante definir qué condiciones son necesarias para concluir que el método ha logrado superar el desafío, o al menos para indicar el grado en que este se ha superado.

En la FI la mayoría de los autores reportan los resultados de sus métodos de manera lingüística, cualitativa o cuantitativa con base en las comparaciones realizadas con otros algoritmos, ya sean de la misma naturaleza que el desarrollado o diferente, y mediante el uso de varios videos que contienen diferentes desafíos.

Los resultados lingüísticos consisten en indicar mediante enunciados que el método desarrollado es capaz de manejar una situación crítica en específico de una mejor manera respecto a otros o bien, que se destaca de entre los demás métodos. Por otra parte, los resultados

cualitativos consisten en el uso de gráficas que reflejan los resultados de una métrica o cuadros de video con rectángulos delimitadores del objeto de interés que muestran cómo se va dando el seguimiento a lo largo de la secuencia de video. Finalmente, el cuantitativo consiste en aplicar las métricas de seguimiento de objetos para con ello obtener un valor numérico que refleje el nivel de triunfo que tiene el método respecto a un video con situaciones críticas específicas y frente a los demás métodos.

Así pues, se puede concluir que si la FI incluye resultados favorables lingüísticos, cualitativos y cuantitativos sobre un desafío en específico implica que el método en cuestión ha logrado superarlo, o bien, que si cumple con alguno de esos 3 se tendrá cierta validez sobre haber superado el desafío lo que podría asociarse a un cierto grado de superación de este.

Con esto, la evaluación para el criterio de *DS* propuesta es de la siguiente manera: se consideran tres formas para presentar los resultados de los métodos ante los diferentes desafíos las cuales son lingüística (*LG*), cuantitativa (*CN*) y cualitativa (*CL*), mismas que se describieron anteriormente. Cada una de estas formas conlleva un peso que expresa la validez del resultado, es decir, los resultados lingüísticos por sí solos tienen menor peso o validez respecto a los resultados cualitativos, y estos a su vez tienen menor peso que los resultados cuantitativos. La razón de que los resultados lingüísticos sean los de menor validez se debe a la carencia de formalidad en estos. Si el autor muestra resultados ya sean favorables o no solo mediante enunciados, los lectores no podrán verificar su validez si no se complementan ya sea con resultados visuales o numéricos. Por otra parte, los resultados cualitativos y cuantitativos dan al lector mayor certeza sobre lo que se está reportando. Por lo anterior, se proponen las siguientes ponderaciones para cada una de las formas de presentar los resultados, note que la suma de estos da un valor unitario: 0.15 para *LG*, 0.35 para *CL* y 0.5 para *CN*. Estos valores se escogieron empíricamente donde cada uno representa la validez que tiene cada forma de exponer los resultados.

Así pues, la puntuación para un desafío en específico (*des*) queda definida por la Ec. (2.34) y tiene el siguiente significado: si un *des* tiene un resultado favorable lingüístico se le otorgará una puntuación de 0.15 al desafío. Si además presenta un resultado favorable cualitativo en el mismo desafío tendrá ya un acumulado de 0.5 y si igualmente muestra resultados numéricos que destaquen dará un total de 1 para el desafío en específico.

$$des = LG + CL + CN \tag{2.34}$$

Es importante mencionar que solo se evalúan los desafíos para los cuales el método en particular es diseñado o puesto a prueba, y que todo aquel desafío no mencionado no aportará al puntaje del criterio de *DS*. De esta manera, no solo se asignará un valor de 0 a los desafíos que no fueron superados sino también a aquellos que no entraron en la evaluación del método.

Para establecer que un resultado lingüístico, cualitativo o cuantitativo es favorable se considera lo siguiente: para conseguir la puntuación de 0.15 en *LG* para un *des*, el autor debe enunciar el triunfo del método ya sea expresando que este presenta mejores resultados que los algoritmos utilizados en comparación o que es capaz de manejar el desafío. Para conseguir la puntuación de 0.35 en *CL* para un *des*, el autor debe mostrar ya sea las gráficas que reflejen los resultados de las métricas o los cuadros de video con rectángulos delimitadores del objeto de interés para visualizar como se lleva a cabo el seguimiento de este a lo largo de la secuencia de video. Si las gráficas o los cuadros de video demuestran el mejor resultado de entre los algoritmos utilizados en la comparación se otorga la puntuación correspondiente. Para conseguir la puntuación de 0.5 en *CN* para un *des*, el autor debe mostrar los resultados numéricos ya sea de manera tabular o sobre leyendas en las gráficas y además estos deben ser los mejores de entre los demás resultados dados por los algoritmos utilizados en la comparación.

Los desafíos más comunes encontrados en la literatura y mismos que serán utilizados para la evaluación de este criterio son los enlistados a continuación:

- ✓ Oclusión total
- ✓ Oclusión parcial
- ✓ Cambios de apariencia o deformaciones
- ✓ Fondos abarrotados
- ✓ Escalamiento
- ✓ Variación en la iluminación
- ✓ Presencia de ruido
- ✓ Objetos similares
- ✓ Movimientos erráticos o complejos
- ✓ Videos de baja resolución
- ✓ Entrada y salida de escena
- ✓ Fondo dinámico
- ✓ Movimiento rápido
- ✓ Rotación fuera del plano
- ✓ Rotación en el plano
- ✓ Borrosidad por movimiento

Cada uno de estos desafíos es evaluado para determinar el grado de superación logrado por el método, de tal manera que el criterio *DS* obtendrá una puntuación equivalente a la suma de los grados de superación alcanzados por el método, como se expresa en la Ec. (2.35).

$$DS = \sum_i des_i \tag{2.35}$$

2.2 Aplicación del instrumento de evaluación

El propósito del IE consiste en determinar que métodos presentan las mejores características y por lo tanto que enfoques reportan mejores resultados. Para esto, se elaboró una muestra representativa del universo de los métodos aplicados al seguimiento de objetos. La muestra está compuesta de 29 métodos de fecha reciente, cada uno con su FI e incluye los enfoques de filtros de partículas, filtros de correlación y redes neuronales convolucionales, entre otros. Una descripción general de estos métodos se muestra en la Tabla 2.1 y Tabla 2.2. Además se incluyeron otros 10 métodos cuyos algoritmos de seguimiento de objetos están posicionados dentro de los 10 primeros lugares en VOT2017 *challenge* [39], esto para realizar una comparación entre los resultados obtenidos con el IE y los resultados reportados en esta competencia.

VOT2017 *challenge* realiza un posicionamiento de 51 algoritmos de seguimiento de objetos cuyos enfoques incluyen CNN *matching*, filtros de correlación discriminativos y SVM estructurado, entre otros. En la evaluación de VOT2017 *challenge* se utilizan 3 métricas primarias: *accuracy*, *robustness* y *expected average overlap* (EAO), siendo esta última la que define a los algoritmos ganadores. *Accuracy* es el promedio de la superposición entre los rectángulos delimitadores calculados por el algoritmo y de *ground truth* durante los periodos de seguimiento exitosos. Por otra parte, *robustness* mide cuántas veces el algoritmo pierde al objeto de interés durante el seguimiento. Por último, EAO es un estimador de la superposición promedio que se espera que alcance el algoritmo de seguimiento en una gran colección de secuencias de video a corto plazo con las mismas propiedades visuales que el conjunto de datos dado. La Tabla 2.3 muestra los resultados de *accuracy*, *robustness* y EAO de los 10 primeros lugares en VOT2017 *challenge*.

Tabla 2.1. Información general de los 29 métodos. (Parte 1/2).

Nombre del artículo	Enfoque	Año	Ref.
Harmony filter: A robust visual tracking system using the improved harmony search algorithm	Filtro armónico	2010	[31]
Efficient visual tracking using particle filter with incremental likelihood calculation	Filtro de partículas	2012	[6]
Multiple object tracking with partial occlusion handling using salient feature points	Filtro de partículas	2014	[25]
Hierarchical Convolutional Features for Visual Tracking	Filtros de correlación	2015	[33]
Moving object detection and tracking from video captured by moving camera	Filtro de Kalman	2015	[32]
Visual tracking based on particle filter with spline resampling	Filtro de partículas	2015	[26]
Visual tracking in complex scenes through pixel-wise tri-modeling	*Pixel-wise likelihood tri-modeling*	2015	[40]
A multiple template approach for robust tracking of fast motion target	Comparación de plantillas y filtro de Kalman	2016	[21]
Unsupervised detection and tracking of moving objects for video surveillance applications	Filtro de partículas	2016	[28]
Moving object tracking with feature learning and inheriting	Filtro de partículas	2017	[41]
Moving objects tracking based on improved particle filter algorithm by elimination of unimportant particles	Filtro de partículas	2017	[5]
Multiple Object Tracking Using Motion Vectors from Compressed Video	Flujo óptico	2017	[42]
Object tracking method based on hybrid particle filter and sparse representation	Filtro de partículas y representación esparcida	2017	[29]
Object tracking using a convolutional network and a structured output SVM	Máquina de vectores de soporte de salida estructurada y CNN	2017	[22]
Object tracking using color-feature guided network generalization and tailored feature fusion	Filtro de partículas	2017	[36]
Online object tracking based on CNN with spatial-temporal saliency guided sampling	Filtros de correlación	2017	[34]
Robust tracking-by-detection using a selection and completion mechanism	*Tracking-by-detection*, *compressive tracking*, fast-RCNN, SVM y NAR	2017	[7]
Spiral visual and motional tracking	Filtro de Kalman y clasificador de Bayes	2017	[37]

Tabla 2.2. Información general de los 29 métodos. (Parte 2/2).

Nombre del artículo	Enfoque	Año	Ref.
Tracking of object in occluded and non-occluded environment using SIFT and Kalman filter	Filtro de Kalman y SIFT	2017	[15]
Tracking with Extraction of Moving Object under Moving Camera Environment	Filtro de partículas	2017	[43]
Visual object tracking by correlation filters and online learning	Filtros de correlación y clasificador de regresión logística	2017	[38]
Visual object tracking via enhanced structural correlation filter	Filtros de correlación	2017	[30]
Convolutional neural networks based scale-adaptive kernelized correlation filter for robust visual object tracking	Filtros de correlación y CNN	2017	[44]
Visual Object Tracking Based on Mean-shift and Particle-Kalman Filter	*Mean-shift* y filtro de partículas-Kalman	2017	[45]
Deep learning to frame objects for visual target tracking	CNN	2017	[46]
Contour-based object tracking in video scenes through optical flow and gabor features	Flujo óptico y clasificador Adaboost	2018	[24]
Re3: Real-Time Recurrent Regression Networks for Visual Tracking of Generic Objects	Red recurrente LSTM (*Long short-term memory*)	2018	[35]
Selective Sampling Importance Resampling Particle Filter Tracking With Multibag Subspace Restoration	Filtro de partículas	2018	[27]
Online multiple object tracking via exchanging object context	*Tracking-by-detection*	2018	[47]

Para dar comienzo con la evaluación de los métodos, primero se recabó toda la información de la FI de cada método que el IE necesita. Luego se definieron los valores de los parámetros de las métricas que utiliza el IE. Estos valores se muestran en la Tabla 2.4 y fueron definidos de acuerdo con la información recabada o mediante el criterio del evaluador.

Para los valores de M_{max}, D_{max}, V_{max} y A_{max} se optó por utilizar el número mayor de métricas, desafíos, videos y algoritmos de comparación reportados en las FI de los 29 métodos evaluados. Para P_{max}, th y fps_{max} se estableció que 5 parámetros, 30 pixeles y 20 FPS serian suficiente para considerar que el método es fácil de configurar, que su precisión es aceptable y que opera en tiempo real, respectivamente.

Tabla 2.3. Primeros 10 lugares dentro de VOT2017 *Challenge*.

Posición	Algoritmo	Enfoque	EAO	*Accuracy*	*robustness*	Ref.
1	LSART	CNN *matching*	0.323	0.493	0.218	-
2	CFWCR	CNN *matching* y filtros de correlación discriminativos	0.303	0.484	0.267	-
3	CFCF	Filtros de correlación discriminativos	0.286	0.509	0.281	[48]
4	ECO	CNN *matching* y filtros de correlación discriminativos	0.280	0.483	0.276	[49]
5	Gnet	CNN *matching* y filtros de correlación discriminativos	0.274	0.502	0.276	-
6	MCCT	CNN *matching* y filtros de correlación discriminativos	0.270	0.525	0.323	-
7	CCOT	CNN *matching* y filtros de correlación discriminativos	0.267	0.494	0.318	[50]
8	CSRDCF	Filtros de correlación discriminativos	0.256	0.491	0.356	[51]
9	SiamDCF	CNN *matching* y filtros de correlación discriminativos	0.249	0.500	0.473	-
10	MCPF	CNN *matching* y filtros de correlación discriminativos	0.248	0.510	0.427	[52]

Tabla 2.4. Configuración de los parámetros en el IE.

Parámetro	P_{max}	fps_{max}	M_{max}	D_{max}	V_{max}	A_{max}	th
Valor	5	20	10	14	162	35	30

En la Figura 2.1 se muestran los resultados obtenidos al realizar la evaluación de los 29 métodos de la Tabla 2.1 y Tabla 2.2. Sus *PF* están dadas en porcentajes donde la puntuación máxima posible dada por el IE es de 79 puntos. Estos resultados indican que los 10 primeros lugares están ocupados en su mayoría por métodos que utilizan filtros de correlación y/o redes neuronales convolucionales.

Para comprobar la congruencia de estos resultados, se realizó la evaluación de los 10 métodos mostrados en la Tabla 2.3 con el IE. Como se aprecia en esta tabla, el enfoque predominante de acuerdo con el posicionamiento de VOT2017 *challenge* es CNN *matching* y filtros de correlación discriminativos, lo que concuerda con los resultados de la evaluación de los 29 métodos mostrados en la Figura 2.1.

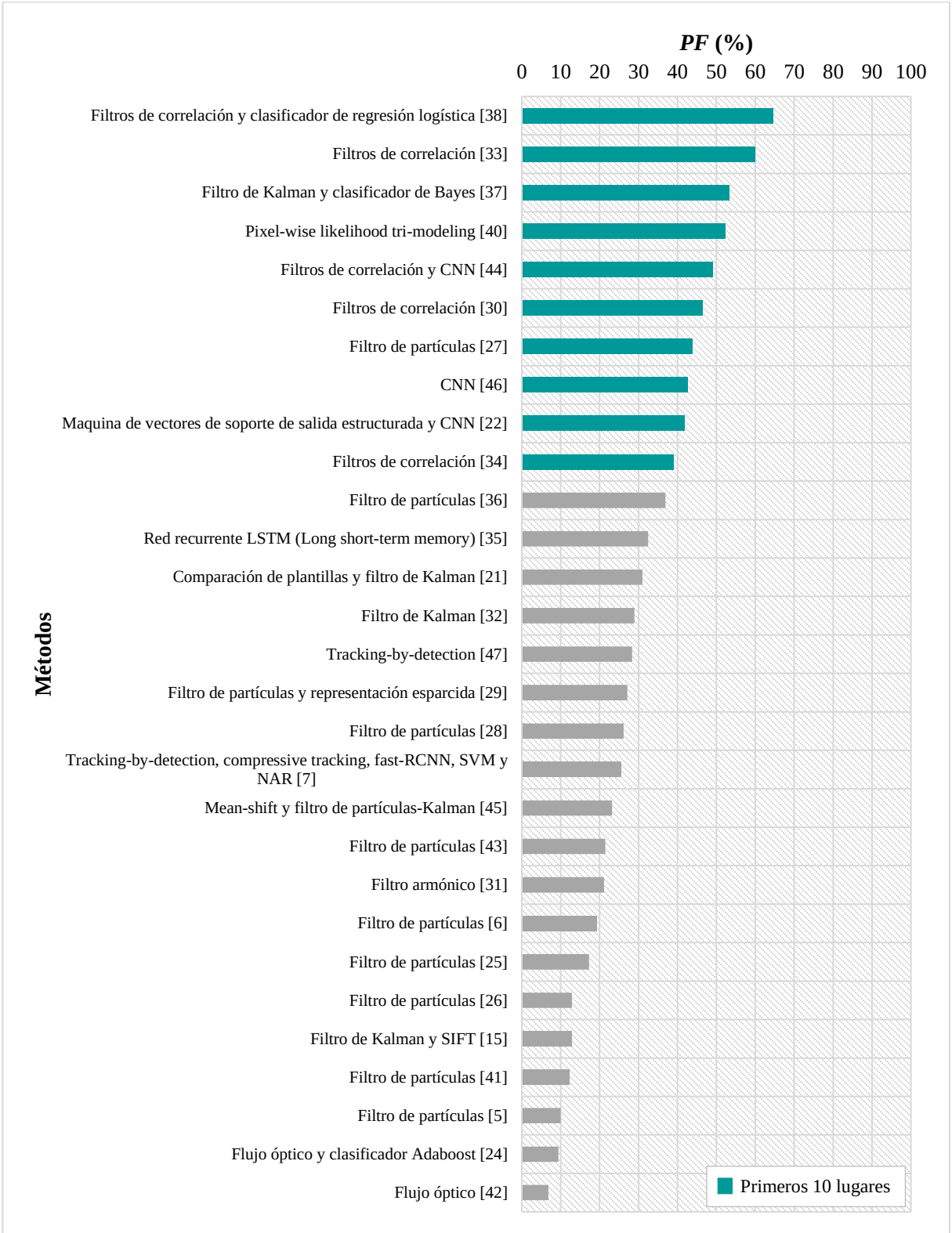


Figura 2.1. Resultados de la evaluación de los 29 métodos aplicados al seguimiento de objetos con el IE.

Al iniciar la evaluación de los 10 métodos de la Tabla 2.3 surgió el inconveniente de que no todos cuentan con una FI lo que significa que no podrán ser evaluados con el IE. De los 10 métodos enlistados, solo 5 tienen FI, por lo que se optó por realizar dos experimentos.

El primer experimento consiste en evaluar los 5 métodos que tienen FI utilizando el IE y comparar los resultados obtenidos con VOT2017 *challenge*. El resultado de este experimento es que 3 de los 5 métodos pertenecientes a VOT2017 *challenge* están dentro de los 10 primeros lugares en el posicionamiento del IE como lo muestra la Tabla 2.5 y Tabla 2.6, donde solo los 10 primeros lugares están escritos en color negro. Una de las razones por la que los 2 métodos restantes no se encuentran dentro de estos 10 primeros es que en su FI no se reportan resultados relacionados al criterio de *DS*, por lo que en ambos casos el puntaje es puesto a 0.

Tabla 2.5. Resultados de la evaluación de los 5 métodos de VOT2017 *challenge* que tienen FI con el IE. (Parte 1/2).

	Método/Enfoque	***PF* (%)**	**Posicionamiento en VOT2017 *Challenge***
1	Filtros de correlación y clasificador de regresión logística [38]	64.61	
2	Filtros de correlación [33]	59.84	
3	Filtro de Kalman y clasificador de Bayes [37]	53.29	
4	Filtros de correlación discriminativos (CSRDCF) [51]	53.08	8
5	CNN *matching* y filtros de correlación discriminativos (MCPF) [52]	52.81	10
6	*Pixel-wise likelihood tri-modeling* [40]	52.24	
7	Filtros de correlación discriminativos (CFCF) [48]	51.41	3
8	Filtros de correlación y CNN [44]	49.01	
9	Filtros de correlación [30]	46.41	
10	Filtro de partículas [27]	43.8	
11	CNN [46]	42.53	
12	Máquina de vectores de soporte de salida estructurada y CNN [22]	41.73	
13	Filtros de correlación [34]	38.93	
14	Filtro de partículas [36]	36.93	
15	Red recurrente LSTM (Long short-term memory) [35]	32.48	
16	CNN matching y filtros de correlación discriminativos (ECO) [49]	31.74	4
17	Comparación de plantillas y filtro de Kalman [21]	31.01	
18	CNN matching y filtros de correlación discriminativos (CCOT) [50]	30.87	7
19	Filtro de Kalman [32]	28.92	
20	Tracking-by-detection [47]	28.36	

Tabla 2.6. Resultados de la evaluación de los 5 métodos de VOT2017 *challenge* que tienen FI con el IE. (Parte 2/2).

	Método/Enfoque	*PF* (%)	Posicionamiento en VOT2017 *Challenge*
21	Filtro de partículas y representación esparcida [29]	27.1	
22	Filtro de partículas [28]	26.14	
23	*Tracking-by-detection*, *compressive tracking*, fast-RCNN, SVM y NAR [7]	25.49	
24	*Mean-shift* y filtro de partículas-Kalman [45]	23.19	
25	Filtro de partículas [43]	21.46	
26	Filtro armónico [31]	21.11	
27	Filtro de partículas [6]	19.19	
28	Filtro de partículas [25]	17.27	
29	Filtro de partículas [26]	12.86	
30	Filtro de Kalman y SIFT [15]	12.82	
31	Filtro de partículas [41]	12.09	
32	Filtro de partículas [5]	10.01	
33	Flujo óptico y clasificador Adaboost [24]	9.407	
34	Flujo óptico [42]	6.844	

El segundo experimento consiste en evaluar los 10 métodos de VOT2017 *challenge* utilizando una de las métricas del criterio de desempeño *De* del IE que mejor se relacione con alguna de las métricas utilizadas por VOT2017 *challenge*. Se eligió la métrica de *accuracy* que es el promedio de la superposición entre los rectángulos delimitadores calculados por el algoritmo y de *ground truth* y que por lo tanto se asemeja a la métrica de éxito De_e en su forma de promedio utilizada en el IE.

En la Tabla 2.7 y Tabla 2.8 se muestran los resultados del segundo experimento, en donde los 10 primeros lugares están escritos en color negro. Los puntajes de cero en estas tablas se deben a que en la FI no se reporta un valor de éxito numérico válido y por lo tanto su puntaje es el mínimo. Los resultados de este experimento muestran que 5 de los 10 métodos se encuentran dentro de los 10 primeros lugares.

Con los resultados obtenidos de los dos experimentos podemos concluir que el IE produce resultados similares a los obtenidos por VOT2017 *challenge*, y que por lo tanto tiene congruencia. Adicionalmente, con base en estos resultados se puede tomar una decisión respecto al enfoque sobre el cual se puede desarrollar el algoritmo de seguimiento de objetos

de esta investigación, que puede ser el de filtros de correlación y/o redes neuronales convolucionales. Para conocer los detalles de la evaluación presentada en este capítulo consulte el Apéndice A.

Tabla 2.7. Resultados de la evaluación de los 10 métodos de VOT2017 *challenge* utilizando el IE en la métrica de De_e. (Parte 1/2).

	Método/Enfoque	De_e	Posicionamiento en VOT2017 *Challenge*
1	*Tracking-by-detection*, *compressive tracking*, fast-RCNN, SVM y NAR [7]	1.873	
2	Filtros de correlación discriminativos (CFCF) [48]	1.685	3
3	Filtros de correlación y clasificador de regresión logística [38]	1.656	
4	CNN [46]	1.632	
5	CNN *matching* y filtros de correlación discriminativos (CCOT) [50]	1.631	7
6	CNN *matching* y filtros de correlación discriminativos (MCPF) [52]	1.616	10
7	CNN *matching* y filtros de correlación discriminativos (ECO) [49]	1.614	4
8	Filtro de Kalman y clasificador de Bayes [37]	1.605	
9	Filtros de correlación y CNN [44]	1.6	
10	Filtros de correlación discriminativos (CSRDCF) [51]	1.587	8
11	Filtros de correlación [30]	1.566	
12	Filtros de correlación [33]	1.562	
13	Máquina de vectores de soporte de salida estructurada y CNN [22]	1.513	
14	*Pixel-wise likelihood tri-modeling* [40]	1.496	
15	Filtro de partículas [36]	1.411	
16	Filtros de correlación [34]	1.36	
17	Filtro de partículas [27]	1.264	
18	CNN matching y filtros de correlación discriminativos (MCCT)	1.225	6
19	CNN matching y filtros de correlación discriminativos (Gnet)	1.202	5
20	CNN matching y filtros de correlación discriminativos (SiamDCF)	1.2	9
21	CNN matching (LSART)	1.193	1
22	CNN matching y filtros de correlación discriminativos (CFWCR)	1.184	2
23	Comparación de plantillas y filtro de Kalman [21]	0	
23	Flujo óptico y clasificador Adaboost [24]	0	
23	Filtro de partículas [6]	0	
23	Filtro armónico [31]	0	
23	Filtro de Kalman [32]	0	
23	Filtro de partículas [41]	0	
23	Filtro de partículas [5]	0	
23	Flujo óptico [42]	0	

Tabla 2.8. Resultados de la evaluación de los 10 métodos de VOT2017 *challenge* utilizando el IE en la métrica de De_e. (Parte 2/2).

	Método/Enfoque	De_e	Posicionamiento en VOT2017 *Challenge*
23	Filtro de partículas [25]	0	
23	Filtro de partículas y representación esparcida [29]	0	
23	Red recurrente LSTM (*Long short-term memory*) [35]	0	
23	Filtro de Kalman y SIFT [15]	0	
23	Filtro de partículas [43]	0	
23	Filtro de partículas [28]	0	
23	Filtro de partículas [26]	0	
23	*Mean-shift* y filtro de partículas-Kalman [45]	0	
23	*Tracking-by-detection* [47]	0	

2.3 Conclusiones

En este capítulo se presentó el diseño de un instrumento de evaluación de métodos aplicados al seguimiento de objetos que tiene como propósito determinar cuáles presentan las mejores características y por lo tanto que enfoque produce mejores resultados ya que sobre este se desarrollará el algoritmo de seguimiento de objetos de esta investigación. Luego de una evaluación de 29 métodos se encontró que aquellos basados en filtros de correlación y/o redes neuronales convolucionales son los que se posicionan en los primeros lugares. Para corroborar que el instrumento de evaluación tiene coherencia se llevó a cabo una comparación con los resultados de la evaluación de algoritmos de seguimiento de objetos dirigida por VOT2017 *challenge*. Esta comparación demuestra que el instrumento de evaluación produce resultados similares con VOT2017 *challenge* y por lo tanto congruentes.

Los resultados dados por el instrumento de evaluación sirven de apoyo en la determinación del enfoque que seguirá el algoritmo de seguimiento de objetos presentado en esta investigación. De acuerdo con los resultados obtenidos el enfoque seleccionado es el de filtros de correlación, ya que tanto en VOT2017 *challenge* como en los resultados del instrumento de evaluación propuesto, los métodos que implementaban este enfoque fueron los que se posicionaron en los primeros lugares.

CAPÍTULO III. SEGUIMIENTO DE OBJETOS MEDIANTE FILTROS DE CORRELACIÓN

El seguimiento de objetos es una de las áreas de visión por computadora que consiste en estimar el estado de un objeto de interés (OI) en los cuadros subsecuentes de una secuencia de video. Este estado tiene características del OI como posición, velocidad, aceleración, entre otras. De acuerdo con [30], el principal desafío en el seguimiento de objetos surge desde dos perspectivas: una es aprender la información acerca del OI a partir de un solo cuadro de video inicial, mientras que la otra es considerar las variaciones que sufre el OI, por ejemplo, oclusiones, deformaciones, rotaciones en el plano y fuera del plano, entre otras.

En este capítulo se presenta el desarrollo de varios algoritmos de seguimiento de objetos basados en filtros de correlación que anteceden a la versión denominada como FCCNN de esta tesis, y se organiza de la siguiente manera: en la sección 3.1 se da a conocer la metodología de evaluación propuesta por Wu *et al.* [53] que es utilizada para medir el desempeño de los algoritmos desarrollados, mientras que en la sección 3.2 se describe la base de datos empleada para la evaluación. La sección 3.3 presenta las características utilizadas para describir al OI. En la sección 3.4 se explican los fundamentos de los filtros de correlación necesarios para el desarrollo de los algoritmos *fc-hog*, *fc-color* y *fc-cnn*, que son descritos en la sección 3.5. Por último, la sección 3.6 muestra los métodos adicionales que se incorporan a *fc-cnn* para crear a los algoritmos *fc-cnn_kf* y *fc-cnn_inc*.

3.1 Metodología de evaluación

Todo algoritmo debe ser sometido a una etapa de evaluación cualitativa y cuantitativa que determinará el nivel de robustez, exactitud y precisión de este. De acuerdo con A. Ali *et al* [17], en la evaluación cualitativa los autores muestran una serie de cuadros de una secuencia de video donde el OI se ve delimitado por un rectángulo centrado en la posición estimada que ilustra el resultado de su algoritmo de seguimiento, como se muestra en la Figura 3.1. La evaluación de este tipo se realiza visualmente, de manera que, entre más cerca se encuentre el rectángulo al OI o al *ground truth*, mejores resultados se tienen en esta evaluación. Sin embargo, una evaluación cualitativa no proporciona una comparación objetiva entre diferentes algoritmos, por lo que es conveniente además realizar una evaluación cuantitativa.

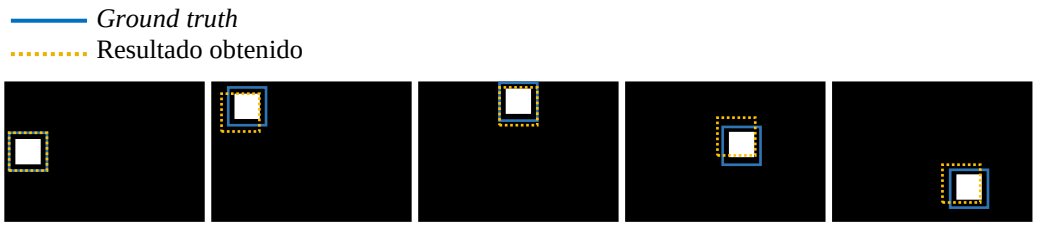


Figura 3.1. Ejemplo de una evaluación cualitativa.

Por ello, Wu *et al.* [53] propone una serie de métricas y procedimientos para medir el desempeño y robustez de un algoritmo de seguimiento de objetos. Esta metodología es común encontrarla aplicada en la literatura lo que representa una ventaja ya que permite llevar a cabo una comparación más directa entre los algoritmos de seguimiento de objetos. Para el análisis cuantitativo se utilizan las mediciones de precisión y éxito, y adicionalmente se incorporan los procedimientos de *one pass evaluation* (OPE), *temporal robustness evaluation* (TRE), *spatial robustness evaluation* (SRE), *one-pass evaluation with restart* (OPER) y *spatial robustness evaluation with restart* (SRER) que evalúan la robustez del algoritmo de seguimiento de objetos. Aunque en esta investigación no se utilizarán cada uno de los cinco procedimientos de evaluación, en las secciones posteriores serán descritos cada uno de los elementos que componen a esta metodología de acuerdo con Wu *et al.* [53], toda la información descrita pertenece a este autor a menos que se indique lo contrario.

3.1.1 Gráfica de precisión y éxito

Una métrica utilizada para medir la precisión del algoritmo de seguimiento de objetos es conocida como error de la localización central (*CLE*) y se define como la distancia euclidiana entre la posición estimada por el algoritmo (x_a,y_a) y la posición *ground truth* (x_{gt},y_{gt}), como se muestra en la Ec. (2.22). La precisión de un algoritmo en una secuencia de video se obtiene al promediar los valores *CLE* calculados en cada cuadro y sus unidades son pixeles por lo que, entre menor sea el valor de esta, mejor desempeño tiene el algoritmo de seguimiento de objetos.

Sin embargo, este cálculo podría no representar el desempeño del algoritmo adecuadamente, ya que un promedio es sensible a datos atípicos. Por ejemplo, suponga el algoritmo A que sigue correctamente al OI pero que en determinado momento llega a perderlo, y el algoritmo B que

no sigue en su mayoría al OI, pero se mantiene cercano a este. En este ejemplo, el algoritmo A es preferible por dos razones:

1) La localización del OI de A es mejor que la de B.
2) A puede incorporar un mecanismo de redetección, lo que permite continuar con el seguimiento correctamente.

No obstante, el cálculo de la precisión mediante el promedio de los valores *CLE* podría indicar que B es mejor que A, ya que, la posición estimada por A puede llegar a ser aleatoria cuando se pierde al OI, produciendo valores *CLE* atípicos que, al promediarse con el resto, genera un valor de precisión alto. Por otra parte, ya que el algoritmo B se mantiene cercano al OI, el valor de esta puede llegar a ser menor que el obtenido para A.

Es por ello que para dar a conocer la precisión de un algoritmo se recomienda utilizar el porcentaje de cuadros en los que el valor *CLE* está por debajo de un umbral. Este cálculo se conoce como razón de precisión (RP) y está dado en porcentaje, por lo que, entre mayor sea el valor, mejor desempeño tiene el algoritmo de seguimiento de objetos. Para una mejor visualización se utiliza una gráfica donde el eje horizontal corresponde al umbral que varía de 0 a 50 pixeles y el eje vertical es la RP, como se ilustra en la Figura 3.2a. El promedio de las RP de cada umbral es mostrado en la leyenda de la gráfica como un valor característico de la precisión del algoritmo.

El éxito es otra métrica utilizada dentro del área de seguimiento de objetos y puede calcularse al obtener la razón de superposición (*RS*) expresada por la Ec. (2.25) en cada cuadro de video y posteriormente promediar estos valores. Sin embargo, utilizar este cálculo presenta las mismas desventajas que se citaron anteriormente para la precisión, por lo que en su lugar se recomienda utilizar el porcentaje de cuadros en los que la *RS* se encuentra por encima de un umbral dado y que se denomina como razón de éxito (RE). Para facilitar la visualización, se utiliza una gráfica donde el eje vertical es la RE y el eje horizontal es el umbral que varía de 0 a 1, donde 1 representa una superposición perfecta y 0 cuando no existe esta, como se muestra en la Figura 3.2b. En la leyenda de la gráfica se muestra el área bajo la curva (AUC) que se utiliza como un valor característico del éxito del algoritmo y se calcula al promediar las RE de cada umbral. De esta manera, entre mayor sea el valor del éxito, mejor desempeño tiene el algoritmo de seguimiento de objetos.

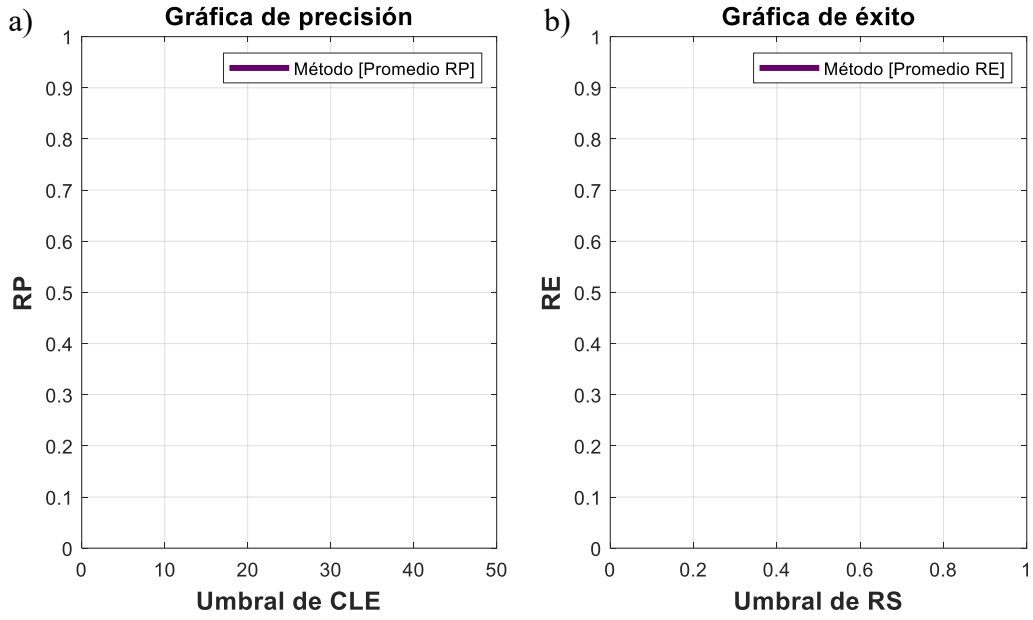


Figura 3.2. Gráficas de a) precisión y b) éxito.

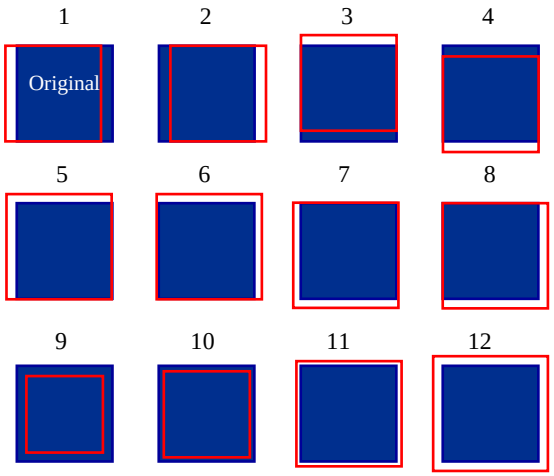
3.1.2 Evaluación de la robustez

La forma convencional de evaluar los algoritmos de seguimiento consiste en inicializarlos con la información del primer cuadro de video designada como estado inicial, y ejecutarlos hasta que finalice la secuencia. Luego, se da a conocer ya sea la precisión o éxito calculado. Este procedimiento es conocido como evaluación de una sola pasada o *one-pass evaluation* (OPE) y tiene los siguientes inconvenientes: primero, un algoritmo de seguimiento puede ser sensible a la inicialización en el primer cuadro, y su desempeño con diferentes estados iniciales o cuadros puede variar significativamente y segundo, la mayoría de los algoritmos no tienen mecanismos de reinicio y los resultados después de las fallas de seguimiento no proporcionan información significativa. Por ello, se incorporan dos procedimientos más para analizar la robustez de un algoritmo ante la inicialización al perturbarlo temporalmente (por ejemplo, comenzar en diferentes cuadros de video) o al perturbarlo espacialmente (por ejemplo, comenzar con diferentes estados iniciales). A continuación, se describirán estos procedimientos:

- *Temporal robustness evaluation* (TRE). La evaluación a la robustez temporal prueba al algoritmo varias veces dentro de la misma secuencia de video. En cada prueba, se

mide el desempeño del algoritmo desde un cuadro en particular de la secuencia, hasta el final de esta con la inicialización correspondiente de ese cuadro. Con esto, se resuelve el problema presentado en el procedimiento OPE, donde la parte más temprana de la secuencia es más importante debido a que, cuando el algoritmo pierde al objeto de interés, la información posterior a este evento resulta en no ser informativa. Los resultados de todas las pruebas se promedian para generar el puntaje de TRE.

- *Spatial robustness evaluation* (SRE). Una inicialización precisa suele ser importante para los algoritmos de seguimiento, pero en la práctica es difícil lograrlo debido a errores causados por los algoritmos detectores o por el etiquetado manual. La evaluación a la robustez espacial pone a prueba al algoritmo ante diferentes estados que se generan al desplazar o escalar ligeramente el rectángulo *ground truth*. Se utilizan ocho desplazamientos espaciales que incluyen 4 centrales y 4 desde las esquinas y 4 variaciones de escala. La cantidad de desplazamiento es del 10% del tamaño del objeto y la relación de escala es de 0.8, 0.9, 1.1 y 1.2 del *ground truth*. En la Figura 3.3 se ilustran los 12 estados generados. El puntaje de SRE es el promedio de estas 12 evaluaciones.



1. Desplazamiento hacia la izquierda
2. Desplazamiento hacia la derecha
3. Desplazamiento hacia arriba
4. Desplazamiento hacia abajo
5. Desplazamiento superior izquierdo
6. Desplazamiento superior derecho
7. Desplazamiento inferior izquierdo
8. Desplazamiento inferior derecho
9. Escala a 0.8
10. Escala a 0.9
11. Escala a 1.1
12. Escala a 1.2

Figura 3.3. Desplazamientos espaciales y variación de escala para SRE propuesto por Wu *et al.* [53].

3.1.3 Evaluación de la robustez utilizando el reinicio

Los algoritmos de seguimiento pierden al OI cuando la apariencia de este cambia drásticamente o cuando se presentan condiciones complejas en el escenario. Una vez que el algoritmo falla, es poco probable que se recupere sin la ayuda de una entrada externa. Aunque TRE está diseñado para mitigar el efecto que tiene este evento en la evaluación, diferentes procedimientos pueden ser útiles para describir el desempeño del algoritmo, tal es el caso de *One-pass evaluation with restart* (OPER) y *spatial robustness evaluation with restart* (SRER).

OPER consiste en medir el desempeño del algoritmo en términos de superposición y de ser necesario, volver a inicializar el algoritmo con el estado del cuadro posterior luego de que se presente una falla. Se establece que ha sucedido una falla cuando el valor de superposición en un cuadro es menor que un umbral establecido. El promedio del puntaje de superposición en todos los cuadros y el número total de fallas muestran la exactitud y la estabilidad del algoritmo, respectivamente. Por otro lado, SRER evalúa si un algoritmo es sensible a la perturbación espacial tal y como se lleva a cabo en SRE, pero con la diferencia de que se realiza el procedimiento de inicialización descrito en OPER para cada estado creado.

En la Figura 3.4 se ilustran los procedimientos OPE (Figura 3.4a), TRE (Figura 3.4b), SRE (Figura 3.4c), OPER (Figura 3.4d) y SRER (Figura 3.4e) descritos previamente. Los rectángulos verdes representan los *ground truth* y el resto de los colores denotan el resultado de un algoritmo. Los rectángulos punteados representan la inicialización del algoritmo.

3.2 Object Tracking Benchmark

Object Tracking Benchmark (OTB) [53] es una colección de 100 secuencias de video para el seguimiento de objetos recopiladas de la literatura reciente. Los autores de esta base de datos han etiquetado cada una de las secuencias de video con 11 atributos. Cada atributo representa un desafío específico presentado en el seguimiento de objetos. Los 11 atributos son: variación en la iluminación (IV), variación de escala (SV), oclusión (OCC), deformación (DEF), movimiento borroso (MB), movimiento rápido (FM), rotación sobre el plano (IPR), rotación fuera del plano (OPR), fuera de vista (OV), fondo abarrotado (BC) y baja resolución (LR).

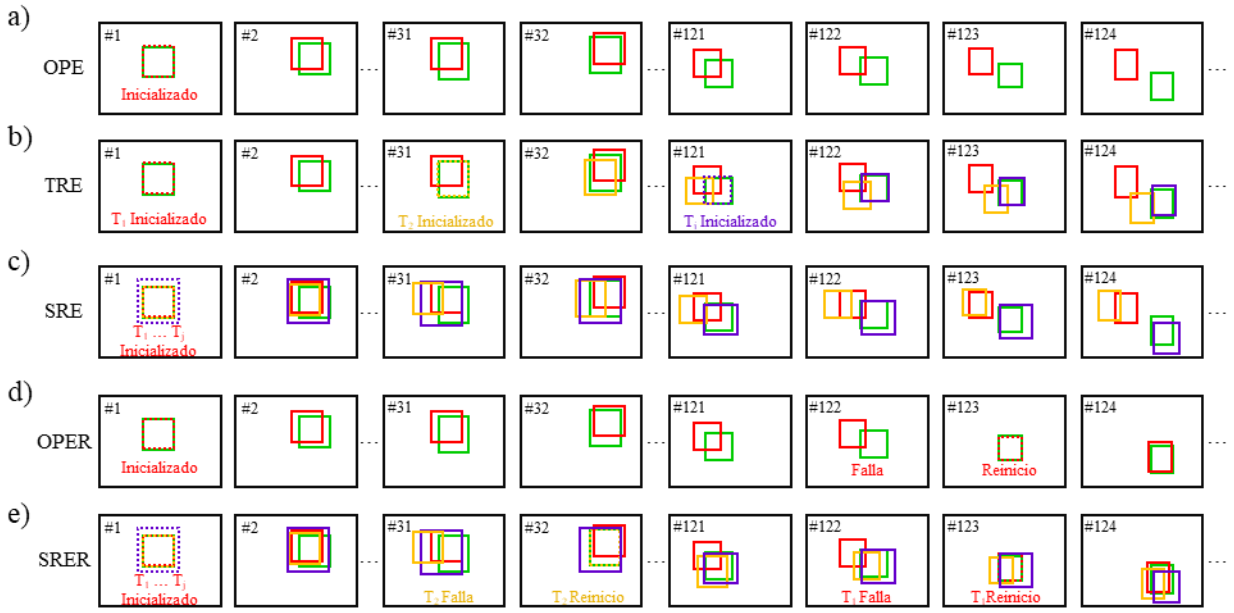


Figura 3.4. Procedimientos en evaluación propuesta por Wu *et al.* [53]: a) OPE, b) TRE, c) SRE, d) OPER y e) SRER.

Cada secuencia de video puede contener más de un atributo y algunos de los atributos ocurren con mayor frecuencia que otros. En la Figura 3.5 se ilustra la distribución de cada atributo en toda la base de datos OTB.

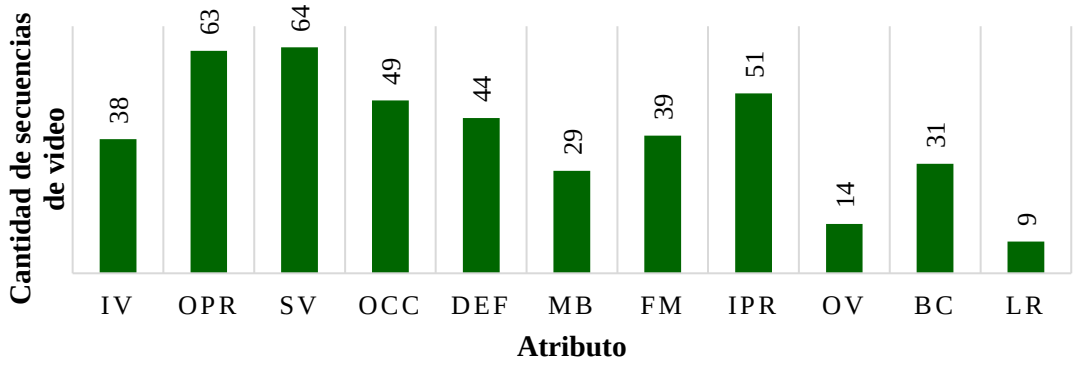


Figura 3.5. Distribución de cada atributo en la base de datos OTB [53].

En la Figura 3.6 se ilustra el primer cuadro de video de cada una de las 100 secuencias de video de OTB con el OI delimitado por un rectángulo.

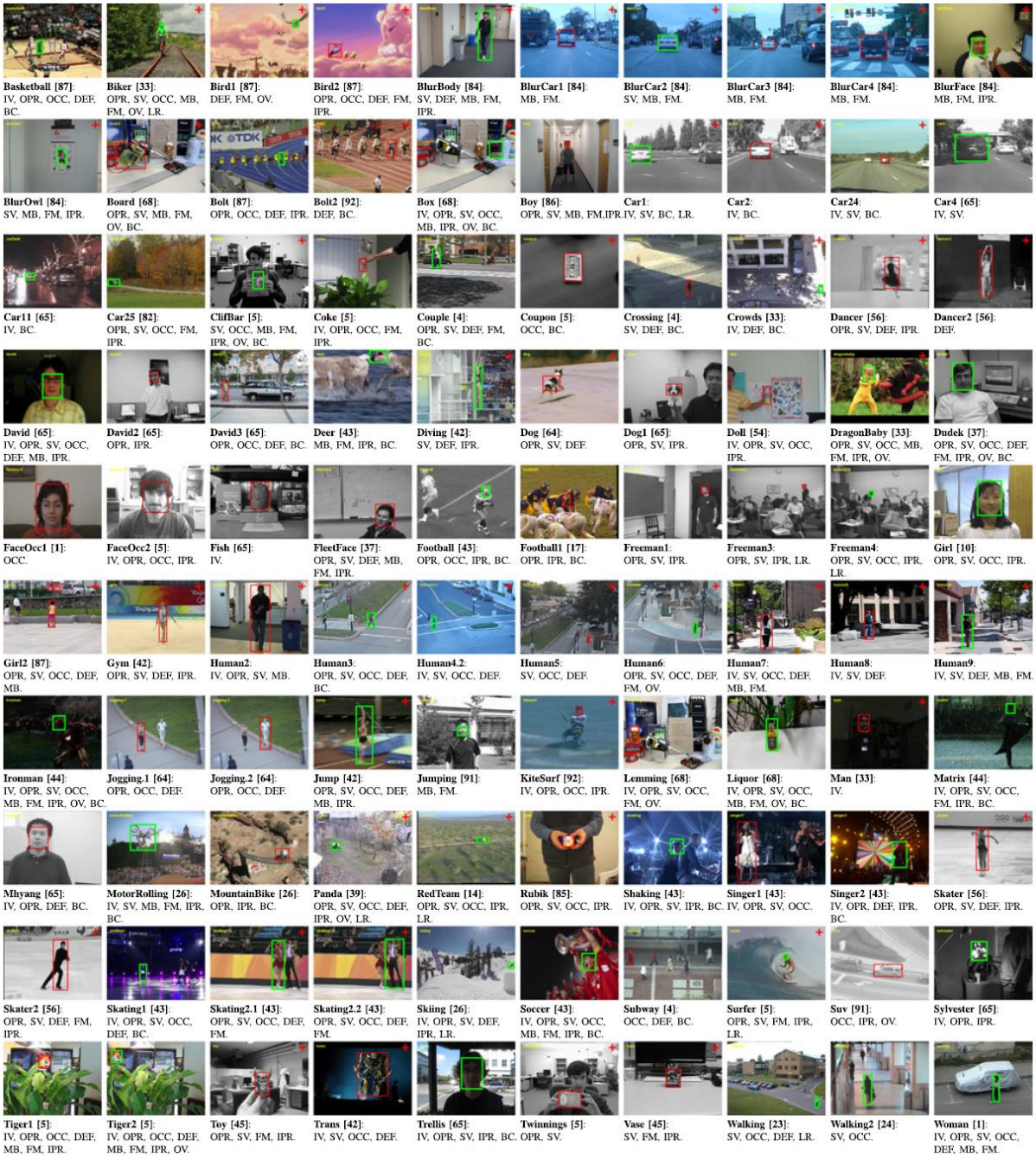


Figura 3.6. Colección de las 100 secuencias de video OTB, cada una etiquetada con sus atributos. Algunos cuadros de video fueron recortados para una mejor ilustración [53].

Adicionalmente, cada secuencia de video es provista con sus anotaciones de *ground truth* para cada cuadro de video. Estas anotaciones corresponden al rectángulo delimitador que

encierra al objeto de interés en cada cuadro de video. Cada anotación es un vector de 4 elementos, donde los primeros dos denota la coordenada horizontal y vertical de la esquina superior izquierda del rectángulo en el cuadro de video y los últimos dos elementos denotan el ancho y alto del rectángulo en pixeles. Puede consultar el Apéndice B para conocer los nombres y la cantidad de cuadros de cada secuencia de la base de datos OTB.

3.3 Características para la descripción del objeto de interés

De manera general, los algoritmos de seguimiento de objetos están basados en dos componentes principales: un modelo de movimiento que describe los estados de un objeto a lo largo del tiempo y predice su estado probable y un modelo de apariencia que representa la información del aspecto del objeto rastreado y verifica las predicciones en cada cuadro [12]. Varios autores concuerdan con que el diseño del modelo de apariencia es un paso crucial en el desarrollo de un algoritmo de seguimiento de objetos y que gran parte del éxito del algoritmo depende del modelo de apariencia [12], [36], [37].

En esta sección se explicarán el conjunto de características elegidas que servirán para la construcción del modelo de apariencia del objeto de interés. Las características elegidas son: histograma de gradientes orientados, color y características convolucionales. La decisión sobre su elección está basada en un análisis de la literatura realizada en este trabajo sobre algoritmos de seguimiento de objetos, siendo estas tres características las más utilizadas dentro de la literatura consultada.

3.3.1 Histograma de gradientes orientados

El histograma de gradientes orientados (HOG, por sus siglas en inglés) es un algoritmo introducido por [54] que consiste en codificar la apariencia y forma de un objeto a partir del cálculo del gradiente dispuesto en forma de histograma.

El gradiente se define como un cambio direccional en la intensidad de los pixeles. Está compuesto por dos valores: la dirección donde el cambio de intensidad es máximo y la magnitud de dicho cambio. En la Figura 3.7 se ilustra el gradiente de una imagen que contiene la forma de un copo de nieve (Figura 3.7a), donde la magnitud se representa proporcionalmente a la intensidad de los pixeles (Figura 3.7b) y la orientación se representa mediante flechas cuyas

direcciones corresponden a la orientación del gradiente (Figura 3.7c). En este ejemplo se ilustra como el gradiente codifica información acerca de la apariencia y la forma del copo de nieve.

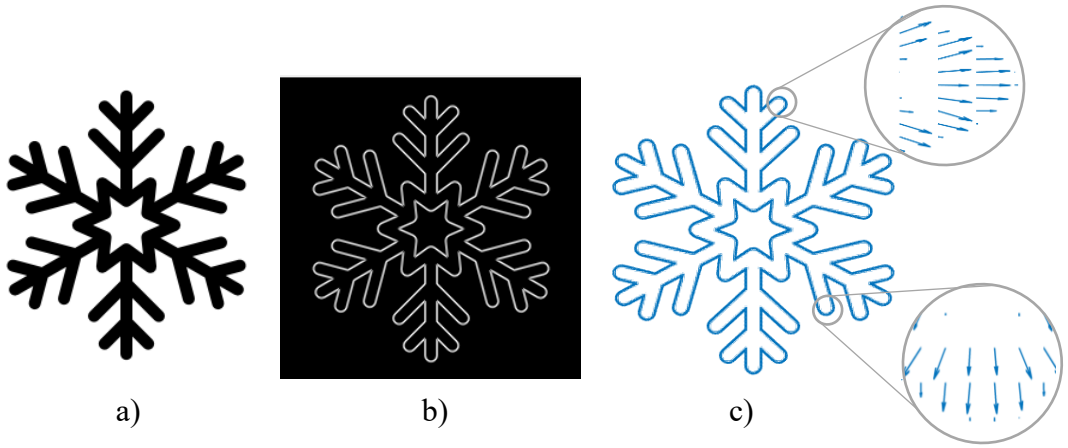


Figura 3.7. Ejemplo del gradiente: a) imagen original, b) magnitud del gradiente y c) orientación del gradiente.

El eje horizontal y vertical del HOG corresponden a la orientación y magnitud de los gradientes, respectivamente. A continuación, se describen los pasos del algoritmo de extracción de características para el cálculo del HOG que son obtenidos principalmente de [54] y [55].

Sea $I(x,y,n)$ el cuadro n de una secuencia de video y $T(x,y,n)$ la porción rectangular que contiene al OI en $I(x,y,n)$, ambos en el espacio de color RGB, los pasos para obtención del HOG son:

1. Se realiza un preprocesamiento que consiste en convertir a $T(x,y,n)$ a escala de grises mediante el algoritmo G_r explicado en [56], obteniendo así a $T_g(x,y,n)$:

$$G_r : T(x,y,n) \rightarrow T_g(x,y,n) \tag{3.1}$$

   Esto permite que en lugar de realizar el cálculo del gradiente en cada canal del espacio de color de $T(x,y,n)$ se realice una sola vez para $T_g(x,y,n)$. Posteriormente, se lleva a cabo un realce de contraste sobre $T_g(x,y,n)$ mediante el algoritmo de ecualización de histograma E_c descrito en [57], obteniendo así a $T_e(x,y,n)$:

$$E_c : T_g(x,y,n) \rightarrow T_e(x,y,n) \tag{3.2}$$

   La ecualización de histograma transforma los valores de intensidad de $T_g(x,y,n)$ de manera que su histograma coincida con uno de distribución uniforme. Esta transformación permitirá una mejor definición en la forma del OI, sobre todo cuando este es poco visible.

2. Se lleva a cabo el cálculo del gradiente en $T_e(x,y,n)$. Este se realiza mediante el uso de alguna máscara discreta de derivada. En la literatura existen varias máscaras empleadas para la obtención del gradiente, sin embargo, en [54] se llevó a cabo un experimento donde se probaron varias máscaras con el fin de determinar cuál de ellas era la más adecuada para la construcción del HOG. Las máscaras que se evaluaron son: 1-D no centrada [-1, 1], centrada [-1, 0, 1] y cúbico-corregida [1, -8, 0, 8, -1], la máscara Sobel 3x3, y las máscaras diagonales 2x2. El resultado del experimento demostró que 1-D centrada [-1, 0, 1] es la máscara más adecuada para la obtención del gradiente en la construcción del HOG. La máscara 1-D centrada se utiliza para calcular el gradiente horizontal *dx* y vertical *dy* de $T_e(x,y,n)$ como se muestran en las Ecs. (3.3) y (3.4).

$$dx(x,y) = T_e(x+1,y,n) - T_e(x-1,y,n) \tag{3.3}$$

$$dy(x,y) = T_e(x,y+1,n) - T_e(x,y-1,n) \tag{3.4}$$

Luego se calcula la magnitud g y orientación θ del gradiente por medio de las Ecs. (3.5) y (3.6).

$$g(x,y) = \sqrt{dx(x,y)^2 + dy(x,y)^2} \tag{3.5}$$

$$\theta(x,y) = tan^{-1}\frac{dy(x,y)}{dx(x,y)} \tag{3.6}$$

3. Se divide a $g(x,y)$ y $\theta(x,y)$ en celdas de igual tamaño. Una celda es una región rectangular cuyo tamaño está en pixeles.
4. Se construyen agrupaciones de varias celdas conectadas llamadas bloques. Los bloques que se forman pueden estar superpuestos entre sí. El tamaño de esta superposición es definido por el diseñador.
5. Se realiza una normalización en $g(x,y)$ dentro de cada bloque formado, utilizando el método L2-Hys [54]. Esto ayudará a minimizar la variación de la iluminación en $T_e(x,y,n)$.
6. Se divide el rango de la orientación del gradiente en S número de intervalos para el histograma de gradientes orientados.
7. Se obtienen los histogramas de gradientes orientados para cada celda: cada elemento en $g(x,y)$ es puesto en el respectivo intervalo (*bin*) de orientación de acuerdo con su $\theta(x,y)$, esto

es, el histograma de la celda denotado por hc_i, donde i es el número de celda, queda definido por la Ec. (3.7),

$$hc_i(\varphi) = \sum_{(x,y)} h\omega_i^x(x,y) h\omega_i^y(x,y) h\omega_\varphi(x,y) g(x,y) \tag{3.7}$$

donde φ es un intervalo de orientación y $h\omega_i^x$, $h\omega_i^y$ y $h\omega_\varphi$ son ponderaciones para cada valor en $g(x,y)$ definidos por las Ecs. (3.8) a (3.10),

$$h\omega_i^x(x,y) = \max\left(0{,}1 - \frac{d_i^x}{\delta x}\right) \tag{3.8}$$

$$h\omega_i^y(x,y) = \max\left(0{,}1 - \frac{d_i^y}{\delta y}\right) \tag{3.9}$$

$$h\omega_\varphi(x,y) = \max\left(0{,}1 - \frac{\left|\theta(x,y) - \theta_\varphi\right|}{\vartheta}\right) \tag{3.10}$$

donde d_i^x es la distancia horizontal entre el elemento (x,y) y el centro de la celda i, δx es la distancia horizontal entre los centros de las celdas vecinas, d_i^y es la distancia vertical entre el elemento (x,y) y el centro de la celda i, δy es la distancia vertical entre los centros de las celdas vecinas, θ_φ es el valor de orientación centro del intervalo φ y ϑ es el tamaño del intervalo en el histograma de gradientes orientados.

8. Se concatenan los histogramas *hc* contenidos dentro de un mismo bloque, como lo expresa la Ec. (3.11), lo que genera un histograma a nivel bloque denotado por *hb*, donde *C* es el número de celdas en el bloque.

$$hb = \{hc_1\ hc_2\ hc_3 \ldots hc_C\} \tag{3.11}$$

9. Se concatenan los histogramas *hb* para obtener el HOG de $T_e(x,y,n)$, denotado como h_{hog} y definido por la Ec. (3.12), donde *Bl* es el número de bloques formados.

$$h_{hog} = \{hb_1\ hb_2\ hb_3 \ldots hb_{Bl}\} \tag{3.12}$$

El histograma de gradientes h_{hog} es un vector cuyo tamaño L_{hog} depende del número de intervalos *S* definidos en el histograma, el número de celdas en un bloque, *C*, y el número de bloques *Bl* formados,

$$L_{hog} = (S)(C)(Bl) \tag{3.13}$$

En la Figura 3.8 se ilustra el procedimiento para el cálculo de h_{hog}.

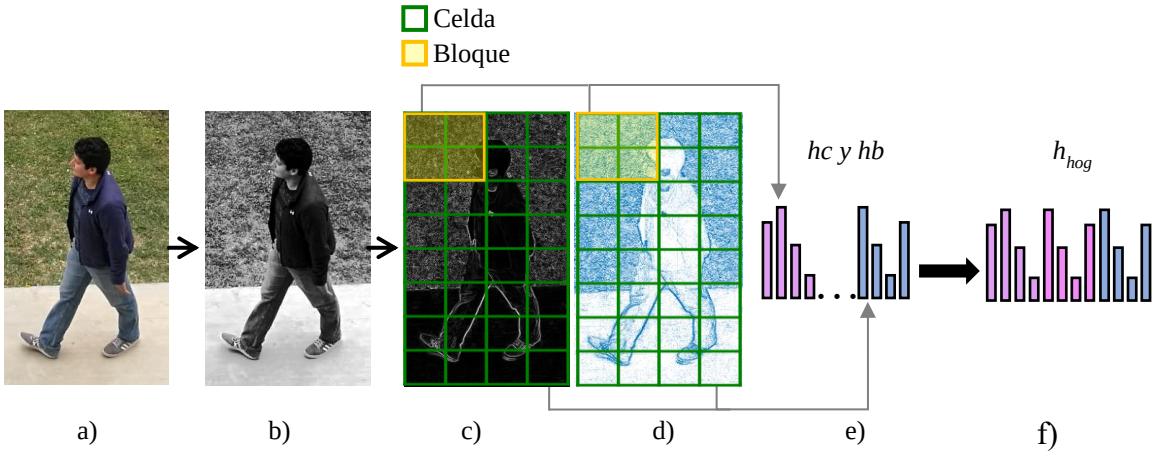


Figura 3.8. Procedimiento para la obtención del histograma de gradientes orientados h_{hog}: a) imagen original, b) imagen preprocesada, c) magnitud del gradiente $g(x,y)$, d) orientación del gradiente $\theta(x,y)$, e) concatenación de los histogramas y f) obtención de h_{hog}.

3.3.2 Histograma de color difuso

El color es una característica que es fácilmente perceptible por una persona y es de las más utilizadas en la literatura para describir los OI en los algoritmos de seguimiento de objetos. El contenido de color puede ser representado por un histograma, cuyos intervalos son predefinidos para acumular un valor de color específico. Sin embargo, estos enfoques convencionales son sensibles a los errores de cuantificación [58], es decir, valores similares pueden quedar contenidos en diferentes intervalos y valores diferentes pueden quedar contenidos en el mismo, tal como se aprecia en la Figura 3.9.

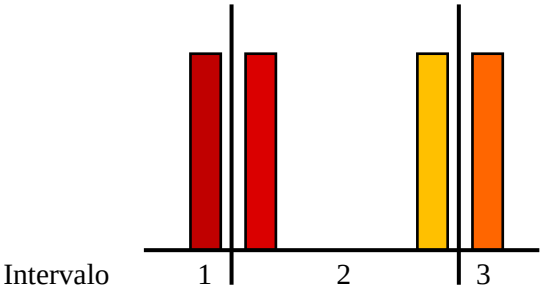


Figura 3.9. Cuantización del espacio de color [58].

Motivado por esto, se propone el uso de un histograma de color difuso para la descripción del color en el OI. La idea principal de este histograma es reflejar la información de color en los intervalos correspondientes que representan un color específico, por ejemplo, amarillo, azul, verde, entre otros.

En [59] se presenta un enfoque de lógica difusa para la construcción de un histograma de color. Este trabajo sirvió de base para el diseño del sistema difuso a partir del cual se obtiene el histograma de color difuso que se propone en esta investigación. La principal diferencia con [59] es que el sistema difuso que se diseña tiene una mayor resolución basada en más funciones de pertenencia y contempla un conjunto de muestras de color para la definición de los parámetros y reglas del sistema difuso.

El histograma de color difuso es obtenido mediante un sistema tipo Mamdani compuesto de 3 entradas nombradas como *L*, *a* y *b* y 1 salida nombrada como *color* como se ilustra en la Figura 3.10. Cada una de las entradas corresponde a un canal del espacio de color CIELab que es elegido porque correlaciona los valores numéricos de color consistentemente con la percepción visual humana. Posteriormente, estos valores son fusificados y evaluados por un conjunto de reglas cuyas salidas son agregadas y defusificadas para la obtención de la salida del sistema.

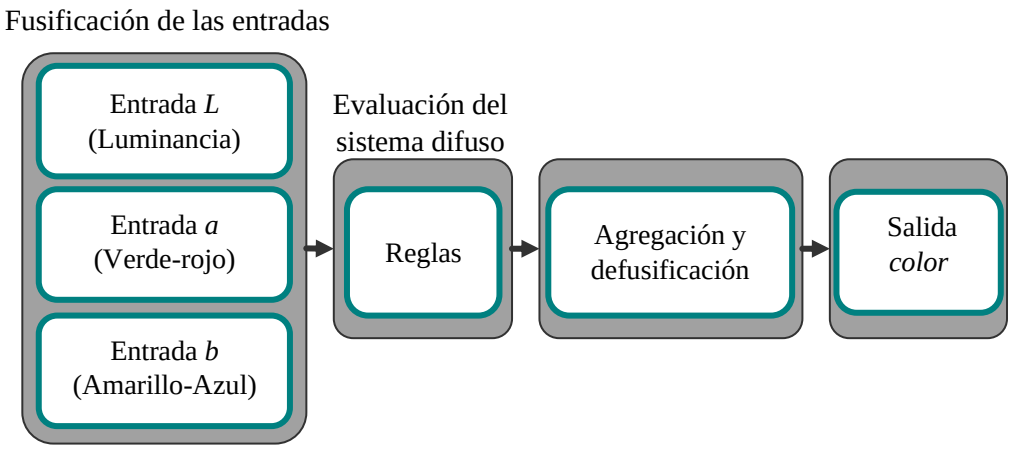


Figura 3.10. Diagrama general del sistema difuso para el histograma de color difuso.

Las funciones de pertenencia para cada variable lingüística de entrada y salida tienen forma ya sea trapezoidal o triangular, de forma similar a [59]. En la Tabla 3.1 se muestra la configuración general del sistema difuso.

Tabla 3.1. Configuración del sistema difuso.

Variable lingüística	Universo de discurso	Valor difuso	Forma de la función de pertenencia	Parámetros
L	[-10 110]	*black*		[-10,-10,0,40]
		gray		[15,50,85]
		white		[71,100,110,110]
a	[-90 100]	*green*		[-90,-90,-86.18,-60]
		sgreenish		[-75,-55,-30]
		lgreenish		[-45,-25,0]
		amiddle		[-10,0,10]
		lreddish		[0,25,45]
		sreddish		[30,55,75]
		red		[60,98.24,100,100]
b	[-110 100]	*blue*		[-110,-110,-107.86,-60]
		sblueish		[-75,-55,-30]
		lblueish		[-45,-25,0]
		bmiddle		[-10,0,10]
		lyellowish		[0,25,45]
		syellowish		[30,55,75]
		yellow		[60,94.48,100,100]
color	[0 12]	*White*		[0,0.2,0.8,1]
		Gray		[1,1.2,1.8,2]
		Black		[2,2.2,2.8,3]
		Brown		[3,3.2,3.8,4]
		Red		[4,4.2,4.8,5]
		Magenta		[5,5.2,5.8,6]
		Orange		[6,6.2,6.8,7]
		Yellow		[7,7.2,7.8,8]
		Green		[8,8.2,8.8,9]
		Cyan		[9,9.2,9.8,10]
		Blue		[10,10.2,10.8,11]
		Violet		[11,11.2,11.8,12]

Para la variable lingüística *L* que representa la luminancia, se definieron los valores difusos de *black*, *gray* y *white*, donde *black* y *white* tienen forma trapezoidal y *gray* tiene forma triangular. Para la variable lingüística *a* que corresponde a la distancia entre verde y rojo, se definieron los valores difusos de *green*, *sgreenish*, *lgreenish*, *amiddle*, *lreddish*, *sreddish* y *red*, donde para *green* y *red* se utilizan funciones de pertenencia trapezoidales y para el resto de los valores difusos triangulares. De manera similar, para la variable lingüística *b* que representa la

distancia entre azul y amarillo, se definieron los valores difusos de *blue*, *sblueish*, *lblueish*, *bmiddle*, *lyellowish*, *syellowish* y *yellow*, donde *blue* y *yellow* tienen forma trapezoidal y el resto triangular.

Para definir los valores difusos de la variable lingüística de salida nombrada como *color*, se consideró el diagrama de cromaticidad de CIE 1976 de la Figura 3.11b. En él vemos que los colores que podemos observar se agrupan en regiones específicas. Por ello se definieron los valores difusos de *Blue*, *Red*, *Yellow*, *Magenta*, *Brown*, *Green*, *Violet*, *Orange*, *Cyan*, *Black*, *Gray* y *White* cuyas funciones de pertenencia tienen forma trapezoidal.

Los parámetros de las funciones de pertenencia de las variables lingüísticas de entrada (*L*, *a* y *b*) fueron ajustados experimentalmente mediante el uso del conjunto de muestras de color que se ilustra en la Figura 3.11a.

a) b)

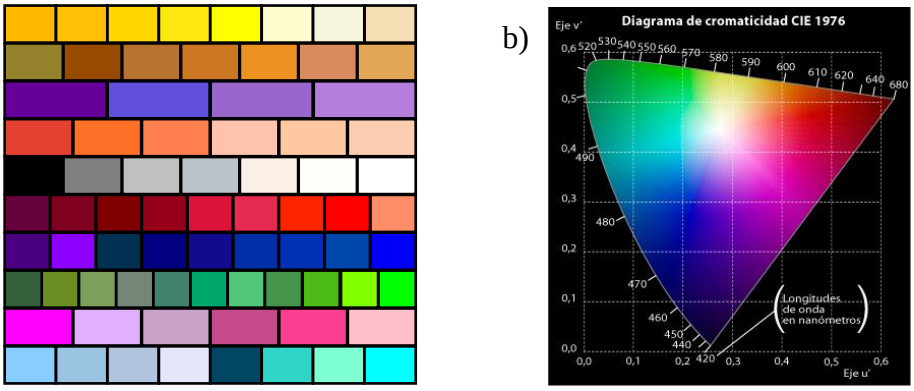


Figura 3.11. a) Conjunto de muestras de color para el diseño del sistema difuso y b) diagrama de cromaticidad CIE 1976.

Estas muestras de color fueron elegidas porque pueden ser fácilmente clasificadas por una persona como uno de los siguientes 12 colores: blanco, gris, negro, café, rojo, magenta, naranja, amarillo, verde, cian, azul y violeta. El ajuste de parámetros se realizó de la siguiente forma: se toma una muestra cuyo color ya está definido y se introduce al sistema difuso. Los parámetros de las funciones de pertenencia son modificados manualmente con el propósito de obtener en la salida el color deseado. El universo de discurso de cada variable lingüística de entrada corresponde a todos los valores que pueden tomar las componentes L, a y b del espacio de color CIELab. Para obtener estos valores se variaron los niveles del espacio de color RGB y se determinaron los rangos de cada componente.

Los parámetros de las funciones de pertenencia de la variable lingüística de salida *color* se definieron de tal manera que los trapezoides fueran iguales, como en [59]. El universo de discurso de *color* fue definido de 0 a 12 ya que se consideran 12 colores.

El conjunto de funciones de pertenencia de las variables de entrada y salida se ilustran en la Figura 3.12 y Figura 3.13, respectivamente.

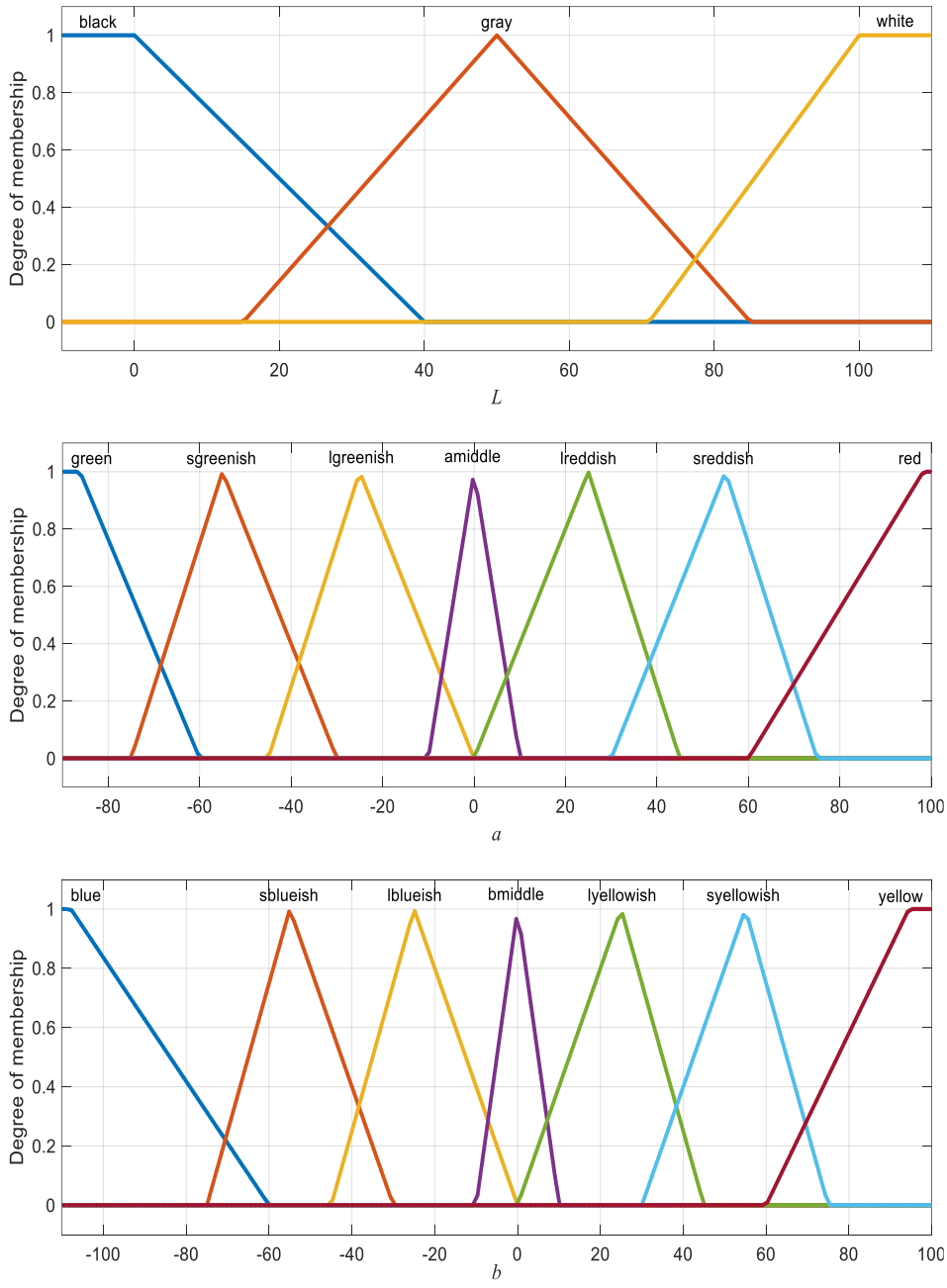


Figura 3.12. Funciones de pertenencia para las variables lingüísticas de entrada *L*, *a*, *b*.

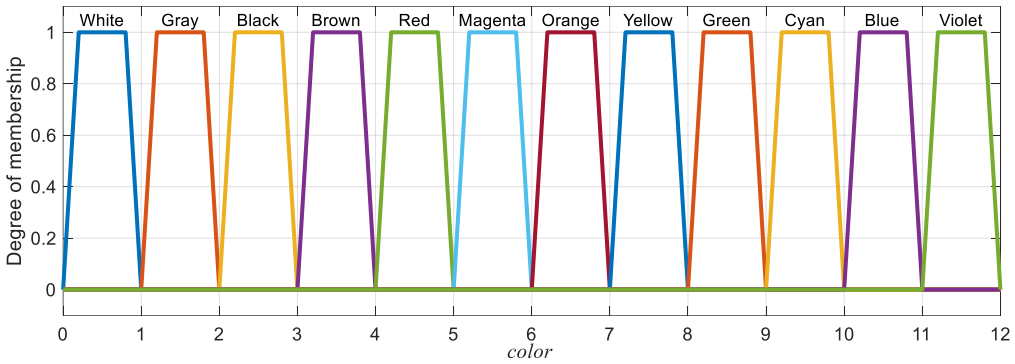


Figura 3.13. Funciones de pertenencia para la variable lingüística de salida *color*.

Debido a que se tienen 3 valores difusos para *L*, 7 para *a* y 7 para *b*, el sistema difuso tendrá un total de 147 reglas cuya estructura prototipo se muestra por la Ec. (3.14),

$$\begin{aligned} &\textbf{Si } L \textbf{ es } MF_L \textbf{ y } a \textbf{ es } MF_a \textbf{ y } b \textbf{ es } MF_b \\ &\textbf{entonces } color \textbf{ es } MF_c \end{aligned} \tag{3.14}$$

donde MF_L, MF_a, MF_b, MF_c son los valores difusos de las variables lingüísticas *L*, *a*, *b* y *color*, respectivamente. Estas reglas se definieron con la ayuda del conjunto de muestras de color de la Figura 3.11a.

La defusificación es el paso final para la generación de la salida de un sistema de inferencia difusa tipo Mamdani. Para defusificar se utiliza el método de mitad del máximo (*middle of máximum*, MOM por sus siglas en inglés) que se expresa en la Ec. (3.15),

$$df = \frac{\sum_{\upsilon_i \in H\upsilon} \upsilon_i}{|H\upsilon|} \tag{3.15}$$

donde *df* es el valor defusificado, υ es el dominio de la función de pertenencia agregada, $H\upsilon$ es el conjunto de valores υ cuyos grados de pertenencia son los más altos y $|H\upsilon|$ es el cardinal del conjunto $H\upsilon$.

En la Figura 3.14 se muestra un ejemplo de defusificación utilizando el método MOM. En este ejemplo, el dominio de la función de pertenencia agregada υ son los valores del 0 al 12 y el conjunto $H\upsilon$ lo componen los valores 4, 5 y 6 ya que alcanzan grados de pertenencia máximos. Al aplicar la Ec. (3.15) el valor *df* obtenido es 5.

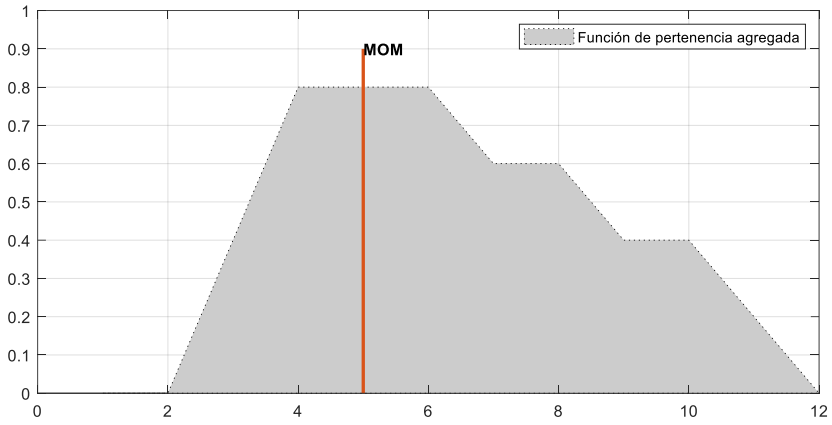


Figura 3.14. Ejemplo de una función de pertenencia agregada y la defusificación MOM.

Aunque es común utilizar el método de centroide en lugar del método MOM, para los fines de esta aplicación resultó que MOM brinda mejores resultados. Esto se debe a que la idea principal del sistema difuso es categorizar el conjunto de valores *L*, *a* y *b* en aquel color que mejor se asocie a estos de acuerdo con el cumplimiento de las reglas. De esta forma, el método MOM define cuales valores en el dominio de la función de pertenencia agregada son mayores y escoge aquel que se encuentre a la mitad de estos.

Una vez definido el sistema difuso, este puede aplicarse para la obtención del histograma de color difuso denotado como h_{color}. Recordando que $T(x,y,n)$ es la porción rectangular que contiene al OI en el cuadro $I(x,y,n)$, el objetivo del sistema difuso es realizar el mapeo descrito por la Ec. (3.16):

$$\zeta : T(x,y,n) \rightarrow T_d(x,y,n) \tag{3.16}$$

donde ζ es la función que realiza las operaciones de fusificación, evaluación de las reglas difusas, agregación y defusificación, y $T_d(x,y,n)$ denota la salida del sistema difuso cuando $T(x,y,n)$ es evaluada. $T_d(x,y,n)$ es entonces una matriz que codifica el color *c* de cada pixel en $T(x,y,n)$. Para obtener $T_d(x,y,n)$, cada pixel en $T(x,y,n)$ es convertido al espacio de color CIELab y posteriormente, se evalúa cada pixel por el sistema difuso que define el color del mismo. Los valores en $T_d(x,y,n)$ son agrupados en h_{color} como se muestra en la Ec. (3.17),

$$h_{color}(c) = \frac{1}{np} \sum_{(x,y)} \delta\left(T_d(x,y) - c\right) \tag{3.17}$$

donde *np* es el número de pixeles en $T_d(x,y,n)$, *c* es un intervalo en h_{color} que corresponde a uno de los 12 colores que codifica el sistema difuso y $\delta(\cdot)$ es la función delta Kronecker.

El histograma h_{color} describe el contenido de color en $T(x,y,n)$. Si $T(x,y,n)$ se encuentra en escala de gris, se realiza una conversión a RGB repitiendo esta información tres veces para formar cada componente RGB. En la Figura 3.15 se ilustran los histogramas de color difuso de tres imágenes con contenido de color amarillo, azul y verde. Los histogramas d) y e) reflejan que las imágenes a) y b) son en su totalidad amarilla y verde, respectivamente, mientras que el histograma f) muestra que el 80% de la imagen c) es categorizada como azul y el resto como cian.

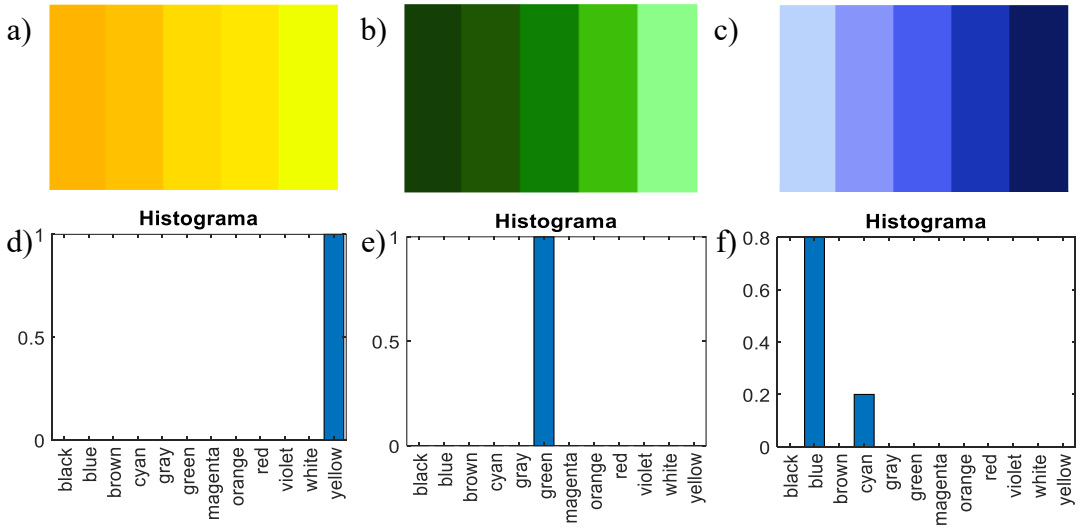


Figura 3.15. Imágenes con contenido a) amarillo, b) verde y c) azul, d) histograma de color difuso de a), e) histograma de color difuso de b) y f) histograma de color difuso de c).

3.3.3 Características basadas en redes neuronales convolucionales

Hasta el momento se han descrito las características de histogramas de gradientes orientados e histograma de color difuso. Este par de características pertenecen al tipo *hand-crafted* [14] que se refieren a aquellas características que se diseñan manualmente. Por otra parte, las características *non-handcrafted* [14] son aquellas que utilizan un paradigma de aprendizaje profundo, como por ejemplo una red neuronal convolucional. En esta sección se dará una introducción acerca del funcionamiento de las redes neuronales convolucionales y como estas pueden ser utilizadas como mecanismos extractores de características.

3.3.3.1 Introducción sobre redes neuronales convolucionales

Una red neuronal convolucional (CNN) es una arquitectura de red para el aprendizaje profundo. De forma general, una CNN es una red neuronal que contiene al menos una capa convolucional. Así como sucede en las redes neuronales tradicionales (RN), las CNN son entrenadas con el propósito de producir la salida deseada en respuesta al estímulo presentado sobre la entrada. Este tipo de redes aprenden directamente de las imágenes de entrenamiento mediante algún algoritmo como el de *backpropagation*.

Las CNN pueden ser descritas en función de las dos etapas que las constituyen: una etapa de extracción de características y una de clasificación, tal como se ilustra en la Figura 3.16.

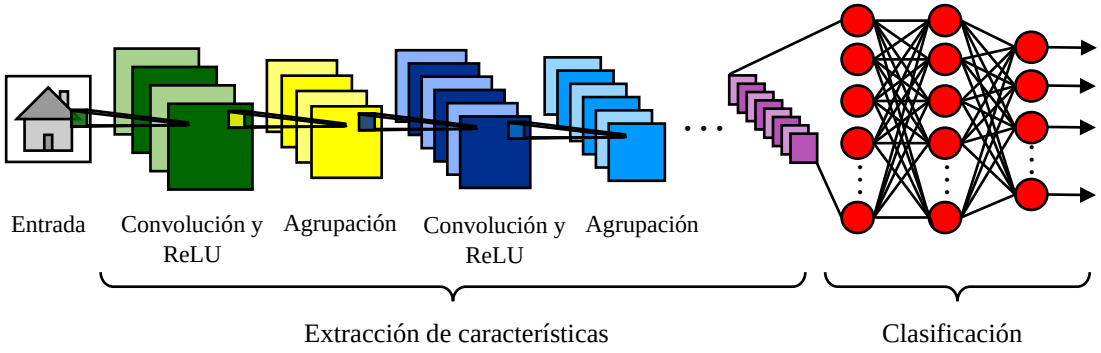


Figura 3.16. Ejemplo de una red neuronal convolucional.

En la etapa de extracción de características se realizan operaciones que alteran los datos con el objetivo de aprender las características específicas de estos [60], que van desde características de bajo nivel como bordes y brillo, hasta características de alto nivel como formas y siluetas. La etapa de extracción de características está compuesta básicamente por tres tipos de capas: capa de convolución, capa de activación y capa de agrupamiento. A continuación, se describirán cada una de estas:

- **Capa de convolución.** En esta capa intervienen dos elementos: un conjunto de filtros y la entrada a la capa de convolución. Cada filtro se convoluciona con la entrada produciendo un mapa de características. Durante el entrenamiento de la CNN los parámetros del filtro son ajustados de tal manera que sea capaz de activar ciertas características de una imagen en particular. Por convención la profundidad en cada filtro es la misma que la profundidad de

la entrada con la que se convolucionan. Considere al término de profundidad como la cantidad de canales.

Sea $I(x,y)$ la entrada a la capa convolucional y $F(x,y)$ el filtro, ambos con profundidad *ch*, la operación de convolución ($*$) queda definida por la Ec. (3.18),

$$MP(x,y) = I(x,y) * F(x,y) = \sum_{ch}\sum_{i}\sum_{j} I_{ch}(x-i, y-j)F_{ch}(i,j) \tag{3.18}$$

donde $MP(x,y)$ es el mapa de características. En la Figura 3.17 se muestra un ejemplo visual de la operación de convolución, donde la entrada $I(x,y)$ y el filtro $F(x,y)$ tienen una profundidad *ch* igual a 3.

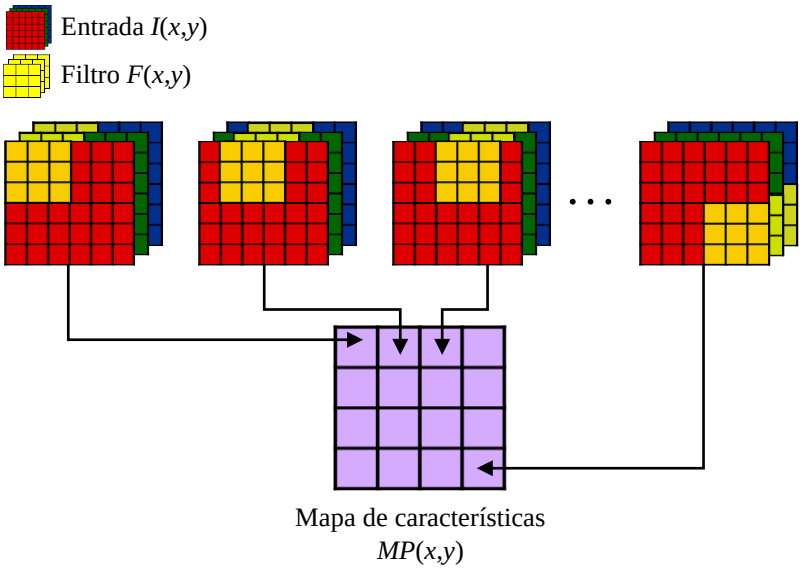


Figura 3.17. Operación de convolución entre $I(x,y)$ y $F(x,y)$ llevada a cabo en la misma capa.

Los problemas con la convolución mostrada de la Figura 3.17 es que la información que contiene $MP(x,y)$ dice muy poco acerca de aquella que está en las orillas de $I(x,y)$ y que además $MP(x,y)$ será una matriz de tamaño reducido, lo que en ocasiones no es deseable. Para manejar estos inconvenientes se introduce el concepto de *padding*, que consiste en agregar elementos alrededor de $I(x,y)$ incrementando así su tamaño. Generalmente, los valores de estos elementos son de cero, por lo que este tipo de *padding* se le conoce como *zero-padding*. En la Figura 3.18 se muestra un ejemplo de *zero-padding* de tamaño [1 1].

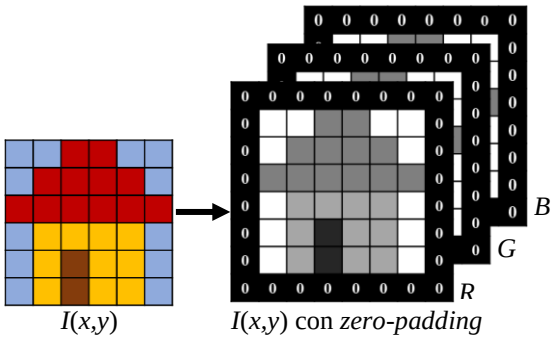


Figura 3.18. Ejemplo de un *zero-padding*.

Otro concepto adicional es el de *stride*, que consiste en el tamaño del desplazamiento horizontal y vertical que realiza $F(x,y)$ sobre $I(x,y)$, que es por omisión de [1 1] como se aprecia en la Figura 3.17 ya que $F(x,y)$ se va moviendo un elemento horizontal y un elemento vertical.

Es así que el tamaño de $MP(x,y)$ queda definido por la expresión (3.19),

$$L_M = \frac{L_I + 2L_P - L_F}{L_S} + \begin{bmatrix} 1 & 1 \end{bmatrix} \tag{3.19}$$

donde L_M es la dimensión de $MP(x,y)$, L_I es la dimensión de $I(x,y)$, L_F es la dimensión de $F(x,y)$, L_P es el tamaño del *padding* y L_S es el tamaño del *stride*. Sustituyendo para el ejemplo de la Figura 3.17 se tiene que L_M es,

$$L_M = \frac{L_I + L_P - L_F}{L_S} = \frac{\begin{bmatrix} 6 & 6 \end{bmatrix} + 2\begin{bmatrix} 0 & 0 \end{bmatrix} - \begin{bmatrix} 3 & 3 \end{bmatrix}}{\begin{bmatrix} 1 & 1 \end{bmatrix}} + \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 4 \end{bmatrix} \tag{3.20}$$

Cuando hay más de un filtro en la capa de convolución se genera más de un mapa de características, como se aprecia en la Figura 3.19. Luego de la convolución se lleva a cabo una última operación que consiste en sumar un elemento escalar llamado *bias* denotado por β a cada elemento de $MP(x,y)$, tal y como se realiza en las RN.

- **Capa de activación.** Esta capa evalúa la salida de la capa convolucional mediante una función de activación (Figura 3.19). Esta función de activación puede ser una función sigmoidal o tangente hiperbólica como las aplicadas en las RN, sin embargo, es más común encontrar que en las CNN se utilice una función de unidad lineal rectificada (ReLU), como se expresa en la Ec. (3.21).

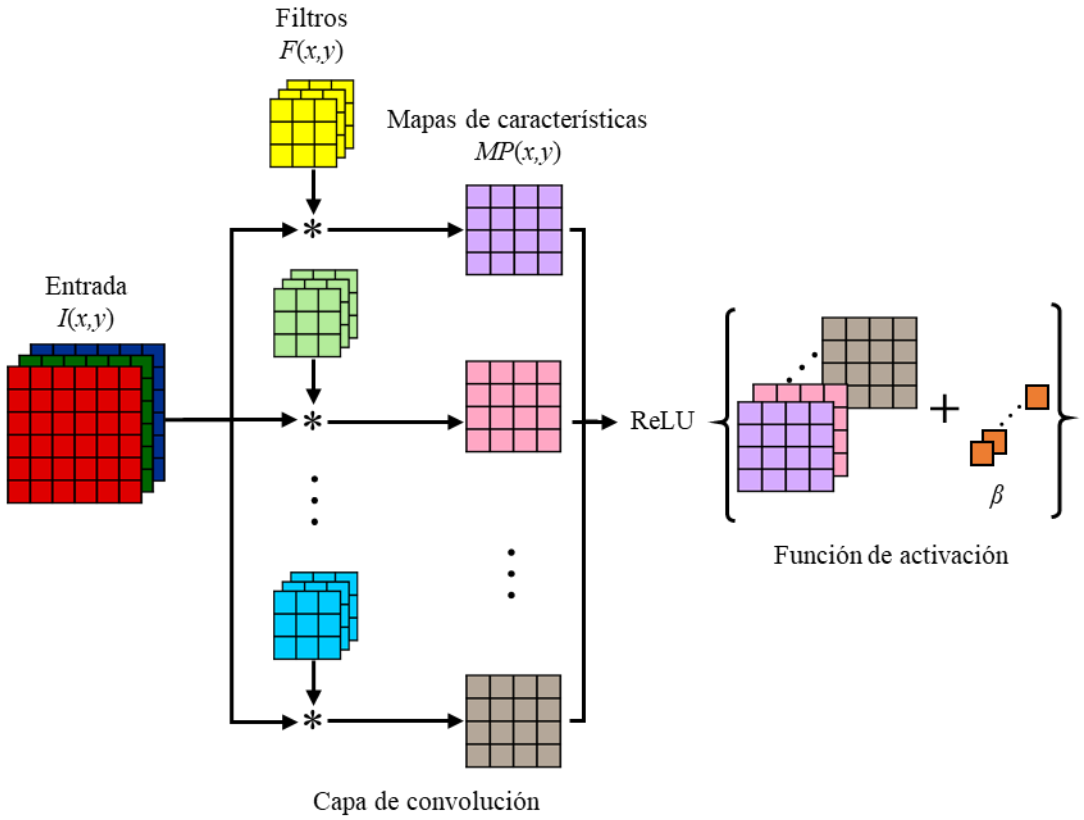


Figura 3.19. Esquema general de una capa convolucional y una capa de activación.

$$\mathrm{ReLU}\left(MP(x,y)+\beta\right)=\max\left(0,MP(x,y)+\beta\right) \tag{3.21}$$

Este tipo de función de activación consiste en asignar los valores negativos a cero y mantener los valores positivos, de tal manera que solo las características activadas prosigan su camino a la siguiente capa.

- **Capa de agrupación.** La capa de agrupación realiza un submuestreo de la salida de la capa de activación, por lo que también son llamadas capas de disminución de resolución. Esto permite que se reduzca el número de parámetros que requieren ser ajustados para realizar un aprendizaje en la CNN. Existen dos tipos de métodos de agrupación: agrupación por promedio (*average pooling*) y agrupación por valor máximo (*max pooling*), siendo este último el método más utilizado. En la Figura 3.20 se muestran los ejemplos numéricos de cada uno de estos métodos. Así como en las capas de convolución, los conceptos de *padding* y *stride* también son aplicables a este tipo de capas.

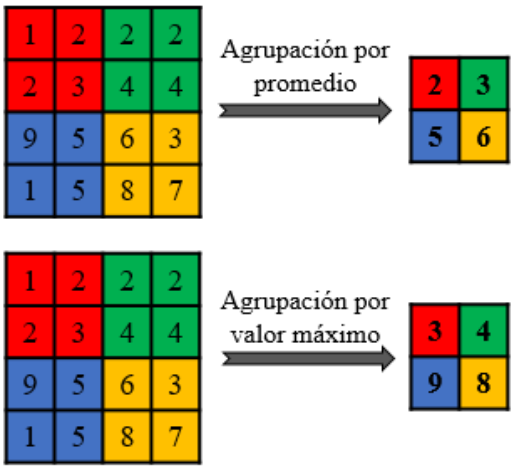


Figura 3.20. Ejemplos numéricos de las agrupaciones por promedio y por valor máximo.

Las operaciones descritas previamente se repiten a lo largo de decenas o cientos de capas en un proceso en el que cada capa aprende a identificar características diferentes. Posterior a la etapa de extracción de características se encuentra la etapa de clasificación. Esta etapa está compuesta de capas totalmente conectadas que generan un vector final cuyo tamaño corresponde al número de clases que la red será capaz de clasificar.

3.3.3.2 Selección de la red neuronal convolucional

A diferencia de las características *hand-crafted* como las vistas previamente (histograma de gradientes orientados e histograma de color difuso), las CNN eliminan la necesidad de una extracción manual, ya que aprenden a detectar diferentes características mediante decenas o cientos de capas convolucionales. Bajo esta perspectiva las características de las CNN parecen ser una mejor opción respecto a las *hand-crafted*, sin embargo, una posible desventaja es que requieren de un conjunto de imágenes extenso para su entrenamiento.

Esto es un problema para esta investigación puesto que la naturaleza de los algoritmos de seguimiento de objetos impide contar con la información suficiente para realizar el entrenamiento de la CNN correctamente ya que la única información que se tiene sobre el OI está contenida en el primer cuadro de la secuencia de video. No obstante, a pesar de esta desventaja si es posible utilizar redes CNN gracias a la transferencia de aprendizaje.

En el área de visión por computadora, la transferencia de aprendizaje es un método que se aplica mediante el uso de modelos preentrenados, es decir, modelos cuya etapa de entrenamiento está finalizada y fue llevada a cabo con una base de datos extensa para dar solución a un problema similar al que se quiere resolver [61]. De acuerdo con [61], el proceso de transferencia de aprendizaje comienza con la selección del modelo preentrenado cuyas características sean tales que se puedan adaptar al problema que se pretende resolver. Estas características pueden ser, por ejemplo, los pesos de la red, su arquitectura y la base de datos con la que fue entrenada.

Para propósitos de esta investigación se seleccionó la transferencia de aprendizaje de la red VGG-16 de K. Simonyan y A. Zisserman [62] cuya arquitectura se muestra en la Tabla 3.2.

En el entrenamiento de la red VGG-16, se utilizó la base de datos *ImageNet* [63] que tiene más de un millón de imágenes agrupadas en mil clases. Esta característica es una de las razones por las que se eligió este modelo preentrenado, ya que los datos de entrenamiento son similares a los objetos con los que se espera trabaje el algoritmo de seguimiento de objetos. Otra de las razones por las que se seleccionó esta red es que su hermana, la red VGG-19 [62], también ha sido utilizada dentro del contexto de seguimiento de objetos como módulo extractor de características en el trabajo de [33]. Sin embargo, el uso de VGG-16 en lugar de VGG-19 es preferible porque la primera es de menor tamaño que la segunda, lo que se traduce en un menor tiempo utilizado durante la propagación de los datos.

Luego de elegir el modelo preentrenado, se define el aprendizaje que se quiere transferir, ya sea, por ejemplo, transferir la arquitectura de la red y entrenarla utilizando datos distintos. Otra manera de transferir el aprendizaje es utilizar la etapa de extracción de características y sustituir la etapa de clasificación por un modelo que mejor se ajuste a nuestro problema a resolver. La definición del tipo de transferencia de aprendizaje puede ser auxiliada mediante la tabla de similitud y tamaño y el mapa de decisión, ambos presentados por [61] y mostrados en la Figura 3.21. La tabla de similitud y tamaño evalúa el tamaño de nuestro conjunto de datos (eje vertical) y su similitud con el conjunto de datos utilizado en el entrenamiento del modelo elegido (eje horizontal). Dependiendo de estas características se define un cuadrante que será utilizado en el mapa de decisión para determinar la manera en que puede ser utilizado el modelo preentrenado.

Tabla 3.2. Arquitectura de la red VGG-16.

No.	Nombre de la capa	Tipo de capa	Características de la capa
1	input	Entrada	224x224x3 con normalización *zerocenter*
2	conv1_1	Convolucional	64 filtros de 3x3x3, L_S: [1 1] y L_P: [1 1]
3	relu1_1	Activación	Función de activación ReLU
4	conv1_2	Convolucional	64 filtros de 3x3x64, L_S: [1 1] y L_P: [1 1]
5	relu1_2	Activación	Función de activación ReLU
6	pool1	Agrupamiento	*Max Pooling* de 2x2, L_S: [2 2] y L_P: [0 0]
7	conv2_1	Convolucional	128 filtros de 3x3x64, L_S: [1 1] y L_P: [1 1]
8	relu2_1	Activación	Función de activación ReLU
9	conv2_2	Convolucional	128 filtros de 3x3x128, L_S: [1 1] y L_P: [1 1]
10	relu2_2	Activación	Función de activación ReLU
11	pool2	Agrupamiento	*Max Pooling* de 2x2, L_S: [2 2] y L_P: [0 0]
12	conv3_1	Convolucional	256 filtros de 3x3x128, L_S: [1 1] y L_P: [1 1]
13	relu3_1	Activación	Función de activación ReLU
14	conv3_2	Convolucional	256 filtros de 3x3x256, L_S: [1 1] y L_P: [1 1]
15	relu3_2	Activación	Función de activación ReLU
16	conv3_3	Convolucional	256 filtros de 3x3x256, L_S: [1 1] y L_P: [1 1]
17	relu3_3	Activación	Función de activación ReLU
18	pool3	Agrupamiento	*Max Pooling* de 2x2, L_S: [2 2] y L_P: [0 0]
19	conv4_1	Convolucional	512 filtros de 3x3x256, L_S: [1 1] y L_P: [1 1]
20	relu4_1	Activación	Función de activación ReLU
21	conv4_2	Convolucional	512 filtros de 3x3x512, L_S: [1 1] y L_P: [1 1]
22	relu4_2	Activación	Función de activación ReLU
23	conv4_3	Convolucional	512 filtros de 3x3x512, L_S: [1 1] y L_P: [1 1]
24	relu4_3	Activación	Función de activación ReLU
25	pool4	Agrupamiento	*Max Pooling* de 2x2, L_S: [2 2] y L_P: [0 0]
26	conv5_1	Convolucional	512 filtros de 3x3x512, L_S: [1 1] y L_P: [1 1]
27	relu5_1	Activación	Función de activación ReLU
28	conv5_2	Convolucional	512 filtros de 3x3x512, L_S: [1 1] y L_P: [1 1]
29	relu5_2	Activación	Función de activación ReLU
30	conv5_3	Convolucional	512 filtros de 3x3x512, L_S: [1 1] y L_P: [1 1]
31	relu5_3	Activación	Función de activación ReLU
32	pool5	Agrupamiento	*Max Pooling* de 2x2, L_S: [2 2] y L_P: [0 0]
33	fc6	Totalmente conectada	4096
34	relu6	Activación	Función de activación ReLU
35	drop6	*Dropout*	50% dropout
36	fc7	Totalmente conectada	4096
37	relu7	Activación	Función de activación ReLU
38	drop7	*Dropout*	50% dropout
39	fc8	Totalmente conectada	1000
40	prob	*Softmax*	
41	output	Salida de clasificación	

De acuerdo con las características de nuestra aplicación, el cuadrante que mejor se ajusta es el cuatro, ya que se tienen pocos datos y una similitud alta con el conjunto de datos

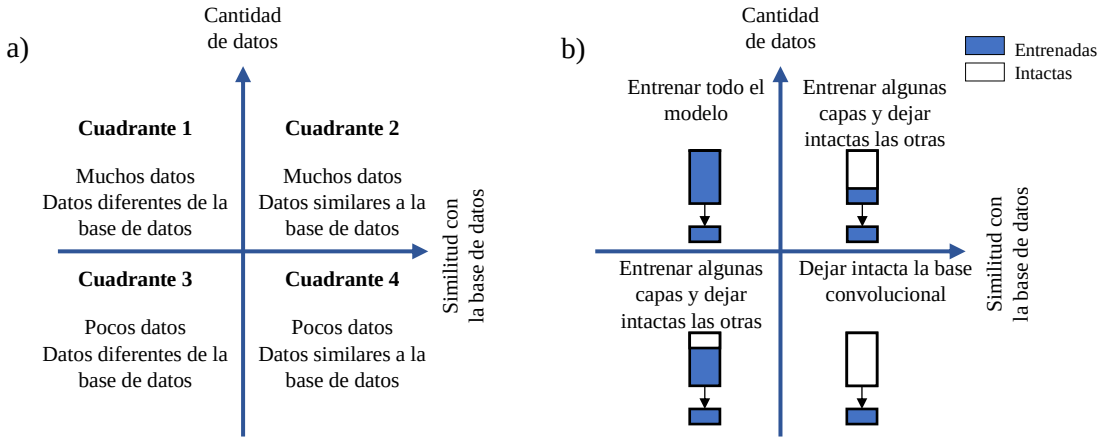


Figura 3.21. a) Tabla de similitud y tamaño y b) mapa de decisión, para el reajuste de modelos preentrenados [61].

de entrenamiento de VGG-16. Este cuadrante, de acuerdo con el mapa de decisión, recomienda mantener la base convolucional, es decir, la etapa de extracción de características en su forma original y reentrenar la etapa de clasificación. Sin embargo, para nuestra aplicación, la etapa de clasificación será removida y sustituida por el método de filtros de correlación que será explicado posteriormente.

Ahora que se ha definido a la red VGG-16 como mecanismo extractor de características se continuará con la explicación detallada de cómo es utilizada para extraer las características de $T(x,y,n)$.

3.3.3.3 Extracción de características con la red neuronal convolucional

Para propósitos de esta investigación, solo se utilizarán las capas que conforman la etapa de extracción de características de la red VGG-16, ya que a partir de estas se obtendrán las características que describirán al OI representado por $T(x,y,n)$.

De acuerdo con la Tabla 3.2, la capa input de la red VGG-16 requiere que el tamaño de la entrada sea de 224x224x3. Debido a que $T(x,y,n)$ puede tener diferentes tamaños se lleva a cabo una redimensión de manera que el tamaño final sea el requerido por la red. Si $T(x,y,n)$ está dada en RGB, significa que el requisito de la profundidad igual a 3 se cumple. Sin embargo, si está dada en escala de gris se realiza una conversión a RGB triplicando esta información para formar cada componente RGB.

Otra modificación que debe sufrir $T(x,y,n)$ es una normalización *zerocenter*. Esta normalización consiste en restarle a cada pixel de $T(x,y,n)$ la media de los datos con los que fue entrenada la red VGG-16. En la literatura se encontró que estos valores son 123.680 para el canal R, 116.779 para el canal G y 103.939 para el canal B.

Luego de este preprocesamiento $T(x,y,n)$ es ahora $T_r(x,y,n)$ siendo esta última la que es alimentada a la red VGG-16. En la Figura 3.22 se muestra el preprocesamiento descrito para la obtención de $T_r(x,y,n)$.

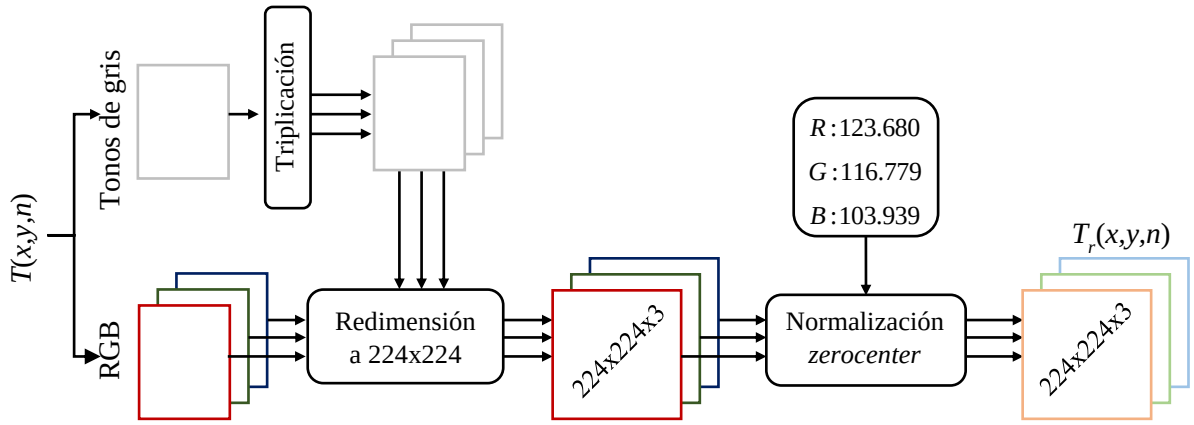


Figura 3.22. Preprocesamiento de $T(x,y,n)$ para ingresar a la red VGG-16.

Cuando $T_r(x,y,n)$ es ingresada a la red VGG-16 se inicia la etapa de extracción de características que va desde la capa 2 hasta la 32 de acuerdo con la Tabla 3.2. Estas 31 capas se dividen en:

- ✓ 13 capas convolucionales.
- ✓ 13 capas de activación con función ReLU.
- ✓ 5 capas de agrupamiento que utilizan el método de valor máximo.

Considere a $\xi_i(x,y,n)$ la entrada a alguna capa que se enlista en la Tabla 3.2, donde el subíndice i denota el número de la capa y solo para $i=\{1,2\}$ $\xi_i(x,y,n)$ es igual a $T_r(x,y,n)$.

Cada capa convolucional de la red VGG-16 obtiene las características de $\xi_i(x,y,n)$ al realizar la operación de convolución entre $\xi_i(x,y,n)$ y los filtros de la capa. La dimensión de los filtros son de 3x3 con profundidad *ch*, que corresponde con la profundidad de $\xi_i(x,y,n)$. En todas las

capas convolucionales, la operación de convolución utiliza un L_P igual a [1 1] y un L_S igual a [1 1] lo que indica que esta operación no cambiará el tamaño de $\xi_i(x,y,n)$.

Seguido de cada capa convolucional, se encuentra una capa de activación con función ReLU. Por otro lado, solo algunas capas de activación van seguidas por una capa de agrupación.

Las capas de agrupación, a diferencia de las capas convolucionales, tienen operadores que si afectarán el tamaño de $\xi_i(x,y,n)$ ya que su función es realizar un submuestreo haciendo que la resolución espacial de $\xi_i(x,y,n)$ se reduzca con el aumento de la profundidad en la red. Todos los operadores en estas capas de agrupamiento son de tamaño 2x2 con un L_S igual a [2 2] lo que hace que el tamaño de la salida en estas capas corresponda a la mitad del tamaño de su entrada.

Cada capa *i*, ya sea convolucional, activación o agrupación, genera una salida denotada como $O_i(x,y,n)$ que puede ser interpretada como un conjunto de mapas de características que describen el contenido de $T_r(x,y,n)$, como se ilustra en la Figura 3.23 que muestra la salida de la capa conv1_1.

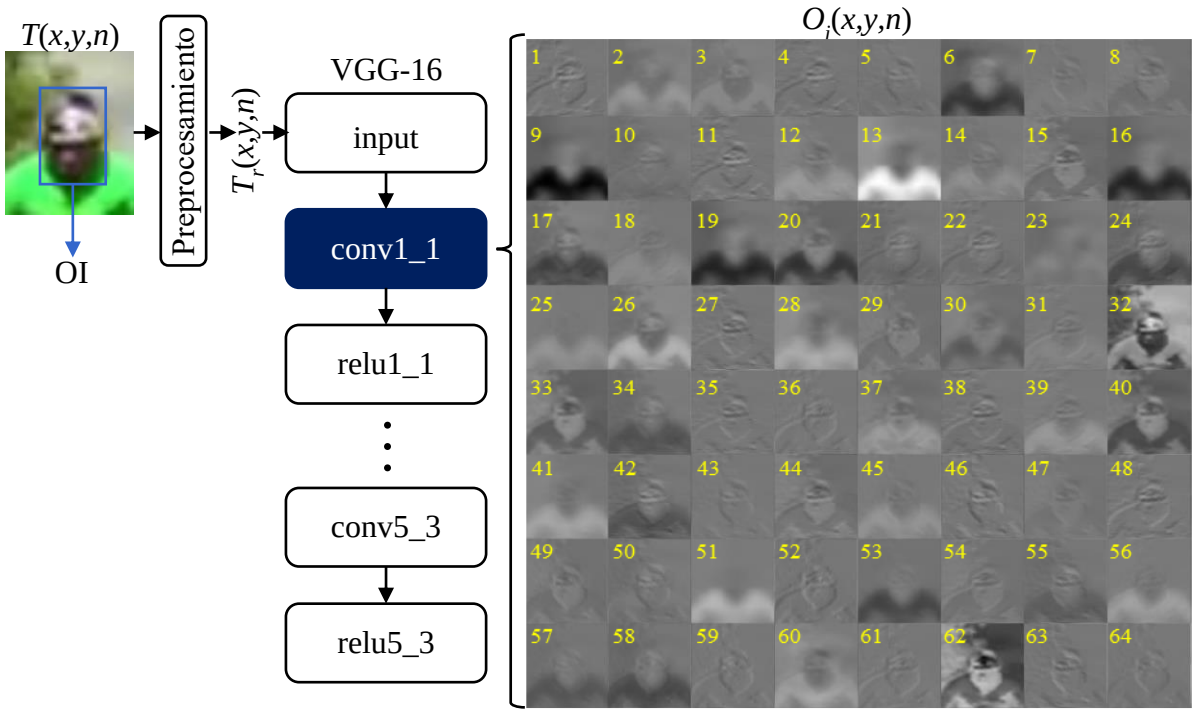


Figura 3.23. Propagación de $T_r(x,y,n)$ para la obtención de los mapas de características de la capa conv1_1. El cuadro pertenece a la secuencia de video *Biker* de la base de datos OTB [53].

En la Figura 3.23 la salida $O_i(x,y,n)$ es de tamaño de 224x224 con profundidad de 64. Cada uno de estos 64 elementos es un mapa de características que codifica un tipo de información. Al ser conv1_1 una de las primeras capas de la red, estos tipos de información incluye características de bajo nivel como bordes y colores. Por ejemplo, por observación de la Figura 3.23, el mapa de características 1 contiene información relacionada a bordes mientras que el mapa de características 13 contiene información acerca del color verde. Los pixeles brillantes y oscuros en estos mapas de características representan activaciones fuertes positivas y negativas, respectivamente, y los pixeles en tonos de gris indican que no existe una activación fuerte.

Para esta investigación se utilizaron los mapas de características generados por las capas de activación tal como en [64], ya que estos mapas contienen la información activada que prosigue a lo largo de la red. Sin embargo, el problema encontrado en los mapas de características que componen a $O_i(x,y,n)$ es que no todos ellos describen correctamente la información del OI. Para dar evidencia de esto considere la Figura 3.24 que muestra los mapas de características producidos en la capa relu1_1 cuando se propaga $T_r(x,y,n)$.

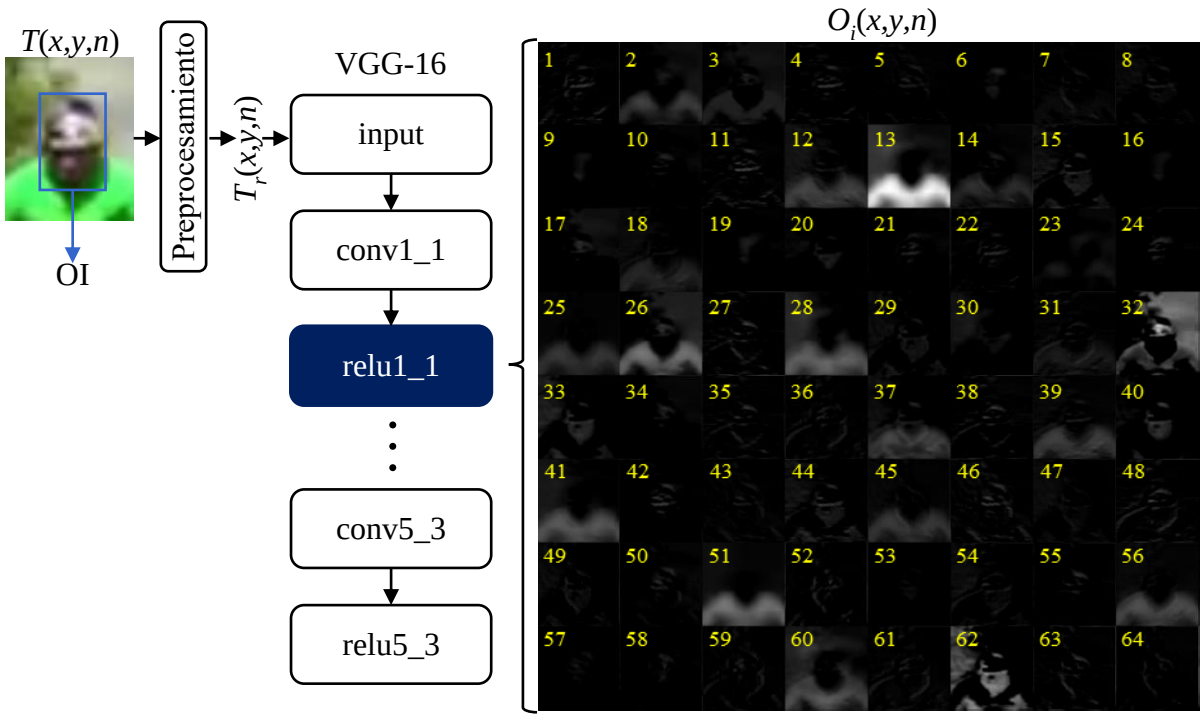


Figura 3.24. Propagación de $T_r(x,y,n)$ para la obtención de los mapas de características de la capa relu1_1. El cuadro pertenece a la secuencia de video *Biker* de la base de datos OTB [53].

En esta figura se puede visualizar que existen mapas de características que no producen suficiente información acerca del OI. Por ejemplo, por observación de la Figura 3.24, en el mapa de características 57 existe poca activación, en el 13 el fondo se activa con más intensidad que el OI y en el mapa de características 62 se activa tanto el fondo como el OI. Estos casos son indeseables ya que el conjunto de características no estaría describiendo al OI de forma esperada. Es por lo anterior que se propone una estrategia que permita utilizar los mapas de características que contienen información relevante acerca del OI.

Esta estrategia consiste en seleccionar aquellos filtros en cada capa convolucional que produzcan mapas de características discriminativos entre el OI y el fondo a partir del nivel de activación de aquellos generados por las capas de activación. Para aclarar lo anterior, considere el diagrama de la Figura 3.19, donde se aprecia que el origen de los mapas de características está en las capas convolucionales y que son generados por el cálculo de la convolución entre una entrada y varios filtros convolucionales. Entonces, si el objetivo es utilizar aquellos mapas de características generados en las capas de activación que contengan información relevante acerca del OI, deben de conocerse cuales son los filtros convolucionales que en su origen los generan.

Para esto, considere a $MP_{i,j}(x,y)$ como un mapa de características de $O_i(x,y,n)$ luego de ser evaluado por la capa de activación con función ReLU, por ejemplo, alguno como los mostrados en la Figura 3.24, entonces el nivel de activación $Na_{i,j}$ del OI en $MP_{i,j}(x,y)$ está definido por,

$$Na_{i,j} = \sum_x \sum_y MP_{i,j}(x,y) K_i(x,y) \tag{3.22}$$

donde los subíndices i,j denotan el número de capa y el número de mapa de características en $O_i(x,y,n)$, respectivamente y $K_i(x,y)$ es una máscara de dimensión igual que $MP_{i,j}(x,y)$ que inhibe la activación del fondo y mantiene la activación de la región del OI dada por,

$$K_i(x,y) = \cos\left(\frac{\sqrt{(x-cx)^2 + (y-cy)^2}}{\sqrt{cx^2 + cy^2}} \pi \right) \tag{3.23}$$

donde (cx,cy) es la coordenada del pixel central de $K_i(x,y)$. De esta forma, la máscara tendrá valor de 1 en (cx,cy) y conforme crece la distancia a este centro el valor decrece hasta un mínimo de -1 como se ilustra en la Figura 3.25.

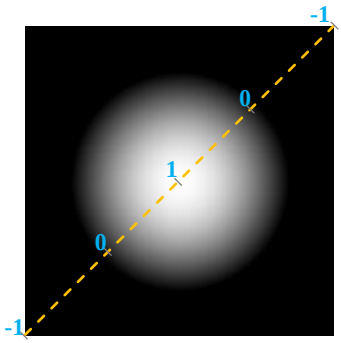


Figura 3.25. Máscara $K_i(x,y)$ para la selección de mapas de características.

Luego de obtener los $Na_{i,j}$ estos se agrupan de acuerdo con la capa i de la que provienen. Cada uno de estos grupos se ordena de mayor a menor de acuerdo con su valor Na y se seleccionan aquellos mp_i valores más grandes de cada grupo. Aquellos $Na_{i,j}$ elegidos hacen referencia a un conjunto de mapas de características en $MP_{i,j}(x,y)$ que fueron producidos a su vez por el conjunto de filtros convolucionales $F_{i,j}(x,y)$ que son los elegidos para producir los mapas de características discriminativos.

Cuando $T(x,y,n)$ sea alimentado a la red VGG-16 se producirán todos los mapas de características. Sin embargo, solo serán utilizados aquellos que provienen de las capas de activación y que en su origen fueron producidos por el conjunto de filtros $F_{i,j}(x,y)$ seleccionados. Cada mapa de características utilizado será denotado como $mc^l(x,y)$. El valor de mp_i será descrito en el capítulo IV.

3.4 Fundamentos de los filtros de correlación

Un filtro de correlación es una clase de clasificador comúnmente utilizado para la detección de objetos. Es una matriz en el dominio de la frecuencia que se diseña a partir de un conjunto de patrones representativos de una clase particular denominada como verdadera [65]. Cuando un patrón es evaluado por el filtro de correlación se produce una salida llamada mapa de respuesta. Si se produce un pico en este mapa entonces el patrón pertenece a la clase verdadera, mientras que si no se produce alguno entonces el patrón pertenece a la clase falsa [65].

Los filtros de correlación han sido muy utilizados dentro del área de seguimiento de objetos especialmente por su alta eficiencia computacional al utilizar la transformada rápida de Fourier. En el diseño de los filtros de correlación, los patrones de la clase verdadera son el conjunto de

características que describen al OI y el mapa de respuesta es el elemento que da referencia o indica la nueva posición de este a lo largo de una secuencia de video. Considere la Figura 3.26 que ilustra el diagrama de flujo del procedimiento para la estimación de la posición utilizando un filtro de correlación. En el primer cuadro de la secuencia de video se define cual es el OI a seguir. Su selección se lleva a cabo mediante un rectángulo de manera que el objeto se encuentre centrado en este. A esta porción de imagen $T(x,y,n)$ generada por el rectángulo se le obtiene la información que describe al objeto, ya sea color, bordes, niveles de intensidad de gris, etc., y que será utilizada en conjunto con la información de salida deseada para la inicialización del filtro de correlación. Luego el filtro es empleado para la determinación de la posición del objeto en los cuadros subsecuentes a través del mapa de respuesta. Este mapa se obtiene mediante la operación de correlación entre el filtro de correlación y la información obtenida en el cuadro actual. Una vez ubicado el OI, el filtro debe ser actualizado para capturar la nueva información de este.

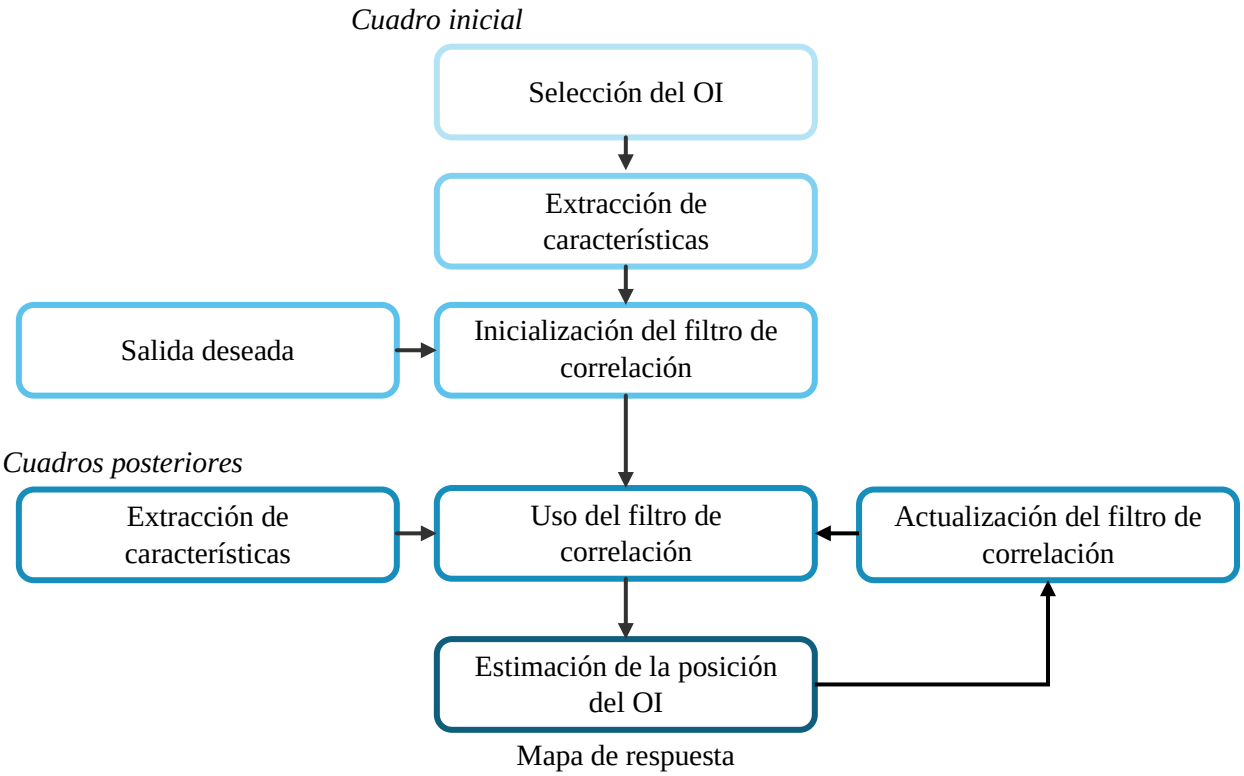


Figura 3.26. Diagrama de flujo del filtro de correlación para la estimación de la posición de objetos en secuencias de video.

Existen varios métodos para el diseño de filtros de correlación en la literatura, sin embargo, el principio de funcionamiento de estos es el mismo. A continuación se explicarán los fundamentos de dos de los métodos para el diseño de filtros de correlación más utilizados en la literatura, el primero de ellos se denomina MOSSE introducido por [66] y el segundo denominado como DSST introducido por [67].

3.4.1 MOSSE

MOSSE es el acrónimo de *minimum output sum of squared error* y se trata de un método para el diseño de filtros de correlación discriminativos. Un filtro de correlación discriminativo es aquel que busca separar la información del fondo y la información del OI, en otras palabras, es un filtro que busca clasificar la información. Los filtros de correlación diseñados por el método de MOSSE están dados en dos dimensiones, al igual que todos los elementos que intervienen en el diseño.

Por simplicidad, considere a q como las características extraídas del OI, ya sea tonos de gris, color, bordes, etc., y a r como la salida de correlación deseada. El primer paso en el diseño de un filtro de correlación es su inicialización. Para ello, el método MOSSE requiere de un número k de imágenes para generar los pares (q_k,r_k), es decir, a cada imagen se le obtienen sus características q_k y se le construye su salida de correlación deseada r_k. En el método MOSSE se utilizan como características los tonos de gris y como salida de correlación deseada una función gaussiana. No obstante, esta última puede tomar cualquier forma, lo que debe cumplirse es que la coordenada donde se encuentra el mayor valor coincida con el centro del OI.

La primera imagen es obtenida a partir de $T(x,y,n)$ mientras que el resto pueden construirse usando transformaciones aleatorias afines sobre $T(x,y,n)$ [66]. En la Figura 3.27 se ilustra un ejemplo del par (q_k,r_k), el cuadro pertenece a la secuencia *David* de la base de datos OTB [53].

Con el conjunto de pares (q_k,r_k) se lleva a cabo la inicialización del filtro de correlación que es conducida en el dominio de la frecuencia para aprovechar la relación elemento-elemento entre la entrada y salida, por lo tanto, todos los pares (q_k,r_k) deben compartir las mismas dimensiones.

De acuerdo con [66], el método MOSSE busca un filtro denotado como γ tal que mapee el conjunto de entrada q_k al conjunto de salida deseado r_k con el menor error posible.

$T(x,y,n)$ $q_k(x,y)$ $r_k(x,y)$

Figura 3.27. Ejemplo del par (q_k,r_k) obtenido de $T(x,y,n)$.

Sean Q_k, R_k y Γ las transformadas de Fourier de sus versiones en minúscula, el problema de diseño toma la forma de un problema de minimización de error expresado por la Ec. (3.24), donde la barra sobre una variable indica complejo conjugado y $\odot$ denota la multiplicación punto-punto.

$$\min_{\overline{\Gamma}} \sum_k \left| Q_k \odot \overline{\Gamma} - R_k \right|^2 \tag{3.24}$$

Un problema de minimización como el mostrado puede ser resuelto a través del método de gradiente descendiente. Debido a que la correlación en el dominio de Fourier es una multiplicación elemento a elemento, cada (x,y) de $\overline{\Gamma}$ puede ser optimizado independientemente. Derivando parcialmente la Ec. (3.24) respecto a $\overline{\Gamma}$ a nivel elemento e igualando a cero se tiene la Ec. (3.25) que expresa la minimización del error cuadrático.

$$\frac{\partial}{\partial \overline{\Gamma}(x,y)} \sum_k | Q_k(x,y)\overline{\Gamma}(x,y) - R_k(x,y) |^2 = 0 \tag{3.25}$$

La solución cerrada de la expresión anterior se muestra en la Ec. (3.26), donde se aprecia que el filtro de correlación $\overline{\Gamma}$ se compone de un numerador que es la correlación entre la entrada y la salida deseada, y un denominador que corresponde al espectro de energía de la entrada. Puede consultar el material suplementario de [66] para conocer el procedimiento completo de derivación.

$$\overline{\Gamma} = \frac{\sum_k R_k \odot \overline{Q_k}}{\sum_k Q_k \odot \overline{Q_k}} \tag{3.26}$$

Si solo se utiliza una imagen para el entrenamiento del filtro de correlación, es decir, solo existe el par (q_1, r_1), el método MOSSE producirá un filtro exacto.

Una vez que se ha entrenado el filtro de correlación, puede ser utilizado para la determinación de la posición del OI en cuadros posteriores. Para esto, se obtiene el mapa de respuesta ψ mediante la Ec. (3.27) en la que intervienen dos elementos: el filtro de correlación $\overline{\Gamma}$ y las características extraídas en el cuadro actual sobre el cual se estima la posición. Estas nuevas características denotadas por z son obtenidas de una región de interés (ROI) centrada en la posición del cuadro anterior, donde Z es la transformada de Fourier de z y $\Im^{-1}$ es la transformada inversa de Fourier. La nueva posición del OI estará definida como la coordenada en ψ donde se encuentra el valor máximo de correlación.

$$\psi = \Im^{-1}\left\{Z \odot \overline{\Gamma}\right\} \tag{3.27}$$

Hasta aquí se han tratado los procedimientos para la inicialización y uso del filtro de correlación descritos por las Ecs. (3.26) y (3.27), respectivamente. Durante el seguimiento de objetos es común que estos sufran cambios de apariencia continuamente, ya sea por factores externos como la variación de la iluminación o factores internos como deformaciones y rotaciones en el objeto. Es por ello que el filtro de correlación debe ser actualizado de manera que vaya incorporando esta nueva información. Este aprendizaje se realiza en cada cuadro n de la secuencia de video y puede llevarse a cabo mediante la actualización del numerador y denominador del filtro con la regla de aprendizaje descrita por las Ecs. (3.28) a (3.30), donde η es el parámetro de aprendizaje del filtro.

$$\overline{\Gamma_n} = \frac{A_n}{B_n} \tag{3.28}$$

$$A_n = \eta R_n \odot \overline{Z_n} + (1-\eta)A_{n-1} \tag{3.29}$$

$$B_n = \eta Z_n \odot \overline{Z_n} + (1-\eta)B_{n-1} \tag{3.30}$$

De esta manera, se le da mayor peso a la información de los cuadros recientes y permite que el efecto de cuadros antiguos decaiga con el tiempo.

3.4.2 DSST

DSST es el acrónimo de *discriminative scale space tracking*, y al igual que MOSSE, se trata de un método para el diseño de filtros discriminativos. El método DSST es similar al método MOSSE, la diferencia se encuentra en que este último construye el filtro de correlación a partir de un conjunto de imágenes a las que se le extraen las características de tonos de gris, mientras que el método DSST lo construye a partir de las características multidimensionales obtenidas de una sola imagen. Bajo esta comparación, vemos que el método MOSSE hace uso de una característica (tonos de gris) y varias imágenes, y que el método DSST hace uso de características multidimensionales de una imagen.

A continuación, se describirá el método DSST de acuerdo con [67]. Considere un conjunto de características d-dimensional que describe al OI contenido en la porción rectangular $T(x,y,n)$. Se denota como q^l a un elemento en específico del conjunto, donde $l \in \{1,2,\ldots,d\}$, (Figura 3.28).

$T(x,y,n)$ Conjunto de características d-dimensional $r(x,y)$

Figura 3.28. Ejemplo del conjunto de características d-dimensional y salida de correlación deseada $r(x,y)$.

El objetivo del método DSST es encontrar un filtro de correlación γ que consiste en un filtro γ^l por cada dimensión de característica. Esto se logra mediante la minimización de la función de costo descrita por la Ec. (3.31), donde r es la salida de correlación deseada asociada a la muestra q^l, λ es un parámetro real positivo que controla el impacto del término de regularización y $\star$ denota la operación de correlación.

$$\left|\sum_{l=1}^{d}\gamma^l \star q^l - r\right|^2 + \lambda\sum_{l=1}^{d}\left|\gamma^l\right|^2 \tag{3.31}$$

La solución de la expresión anterior se muestra en la Ec. (3.32), donde se puede ver que se trata de una generalización del método MOSSE cuando este es inicializado con una imagen. Además, vemos que el parámetro de regularización λ mitiga el problema de las componentes de frecuencia cero en el espectro de *q* lo que llevaría a una división por cero.

$$\overline{\Gamma}^l = \frac{\overline{R}Q^l}{\sum_{k=1}^{d}\overline{Q^k}Q^k + \lambda} \tag{3.32}$$

Una vez inicializado el filtro de correlación, este puede ser utilizado para determinar la ubicación del OI al obtener el mapa de respuesta ψ. Este mapa se obtiene al evaluar la entrada *Z* con el filtro de correlación $\overline{\Gamma}$ tal como lo expresa la Ec. (3.33), donde *Z* es la transformada de Fourier del conjunto de características *d*-dimensional *z* extraído de la ROI del cuadro actual centrada en la posición anterior. La nueva posición del objeto será aquella donde se encuentre el valor máximo de correlación en ψ.

$$\psi = \Im^{-1}\left\{\frac{\sum_{l=1}^{d}A^l Z^l}{B+\lambda}\right\} \tag{3.33}$$

De manera similar al método MOSSE, la actualización del filtro se lleva a cabo de manera individual en el numerador y denominador de este, como lo muestran las Ecs. (3.34) a (3.36).

$$\overline{\Gamma}^l_n = \frac{A^l_n}{B_n} \tag{3.34}$$

$$A^l_n = \eta\overline{R_n}\odot Z^l_n + (1-\eta)A^l_{n-1} \tag{3.35}$$

$$B_n = \eta\sum_{k=1}^{d}\overline{Z^k_n}\odot Z^k_n + (1-\eta)B_{n-1} \tag{3.36}$$

3.5 Filtros de correlación en un enfoque de seguimiento de objetos

En esta sección se presentan tres implementaciones del filtro de correlación para el seguimiento de objetos. Cada implementación utiliza ya sea el histograma de gradientes

orientados, el histograma de color difuso o los mapas de características de la CNN para describir al OI. El objetivo de estas implementaciones es mostrar como cada una de las características descritas en la sección 3.3 pueden ser utilizadas en conjunto con los filtros de correlación para dar seguimiento a los objetos.

3.5.1 Filtro de correlación con el histograma de gradientes orientados

Aquí se explicará la implementación del filtro de correlación utilizando el HOG descrito en la sección 3.3.1. Este algoritmo es nombrado como *fc-hog*.

Los métodos para el diseño de filtros de correlación descritos previamente (MOSSE y DSST) relacionan directamente la estimación de la posición del OI con la coordenada en el mapa de respuesta ψ donde se tiene el valor máximo de correlación. Esto es posible por dos razones principales: la primera de ellas es que el diseño del filtro de correlación se da en dos dimensiones, lo que permite tener el par de coordenadas (x,y), la segunda es que las características utilizadas para describir al objeto contienen información sobre su distribución, lo que permite asociar un punto en ellas con el centro del OI. Por lo anterior, es posible intuir que existe una cierta desventaja en el uso del filtro de correlación con el HOG debido a que este se da en forma de vector y no como matriz. Sin embargo, el HOG si captura información espacial de cómo están distribuidos los gradientes, por lo que es posible asociar de manera aproximada una zona del HOG con el centro del OI, lo que es importante ya que el punto máximo en la salida de correlación deseada debe estar alineado con este centro.

Para empezar el diseño del filtro de correlación utilizando el HOG, se realizará un análisis más a detalle sobre la construcción de este, lo que ayudará a determinar la forma de la salida de correlación deseada que cumpla con los requisitos en el diseño de los filtros de correlación.

Considere la imagen de la Figura 3.29 que fue creada artificialmente con el propósito de facilitar la visualización de cómo se construye el HOG. La imagen es de 32x32 pixeles y contiene una figura blanca en la esquina superior izquierda. A primera vista se sabe que la imagen tendrá valores de gradiente diferentes de cero en la zona de la figura blanca.

Por conveniencia se han configurado los parámetros del HOG de la siguiente manera: S igual a 9, bloques de [2 2] celdas, celdas de [8 8] pixeles y con una superposición de [1 1] bloques. Además, la orientación del gradiente se toma sin signo.

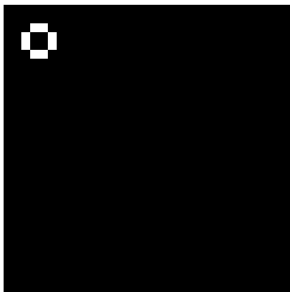


Figura 3.29. Imagen de 32x32 pixeles utilizada para el análisis del HOG.

De acuerdo con la Ec. (3.13), esta configuración hace que L_{hog} sea de 324, ya que se tienen 9 bloques formados por la superposición permitida, 4 celdas por bloque y 9 intervalos en cada histograma. En la Figura 3.30 se visualiza la división en celdas, bloques y la superposición de estos en la imagen de acuerdo con la configuración establecida.

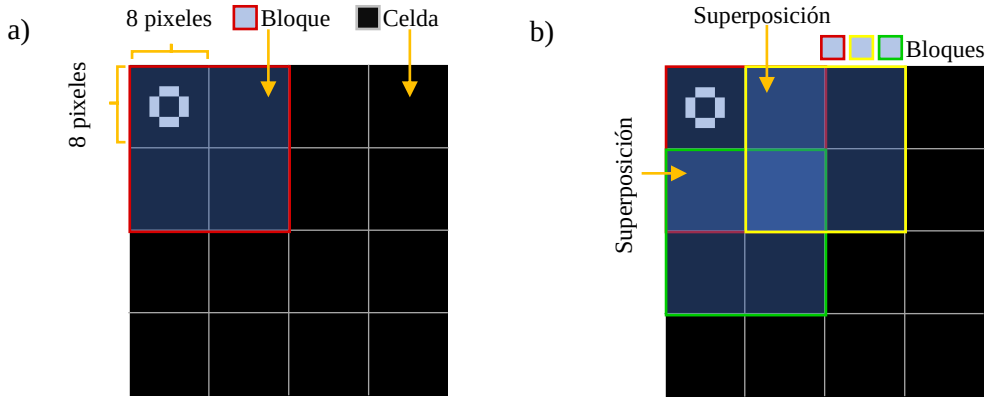


Figura 3.30. a) Celdas, bloques y b) superposición de bloques en la imagen.

Al hacer el cálculo del HOG de esta imagen, se obtendrá un vector cuyos primeros 36 elementos tendrán valores diferentes de cero como se visualiza en la Figura 3.31. Estos elementos corresponden a los 4 histogramas concatenados de las celdas contenidas en el bloque de la esquina superior izquierda, y dan como resultado 36 intervalos, ya que por cada histograma se tienen 9. Además, los primeros 9 elementos tienen mayor magnitud ya que la figura blanca está contenida en la primera celda de la imagen. Con todo lo anterior se concluye que el inicio de la construcción del HOG comienza en el bloque de la esquina superior izquierda y que además los histogramas se concatenan comenzando por la celda en la misma ubicación.

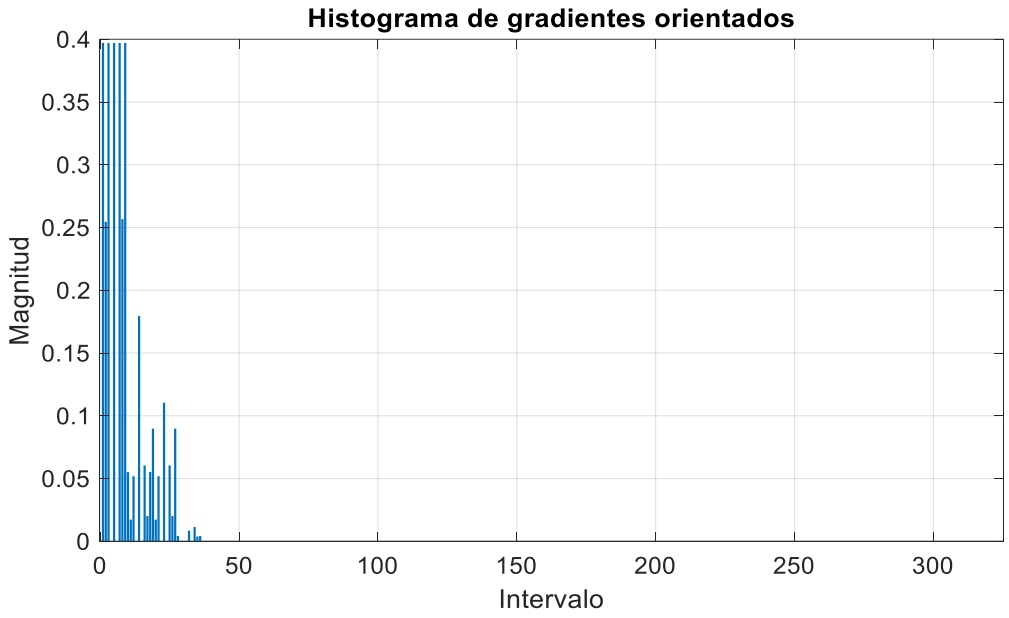


Figura 3.31. Resultado del cálculo del HOG de la imagen bajo estudio.

Si se desplaza la figura blanca a diferentes ubicaciones en la imagen se verá como la información del gradiente viaja a lo largo del vector HOG. En la Figura 3.32 y Figura 3.33 se visualiza este experimento y demuestra que la construcción del HOG se lleva a cabo de arriba hacia abajo, de izquierda a derecha, tanto en la concatenación de los histogramas dentro del bloque como en la concatenación de los histogramas entre bloques.

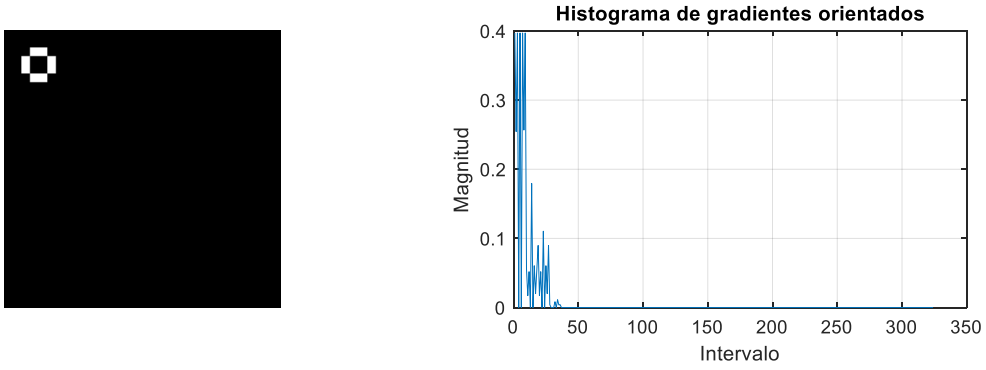


Figura 3.32. Resultados del cálculo del HOG en diferentes imágenes. (Parte 1/2).

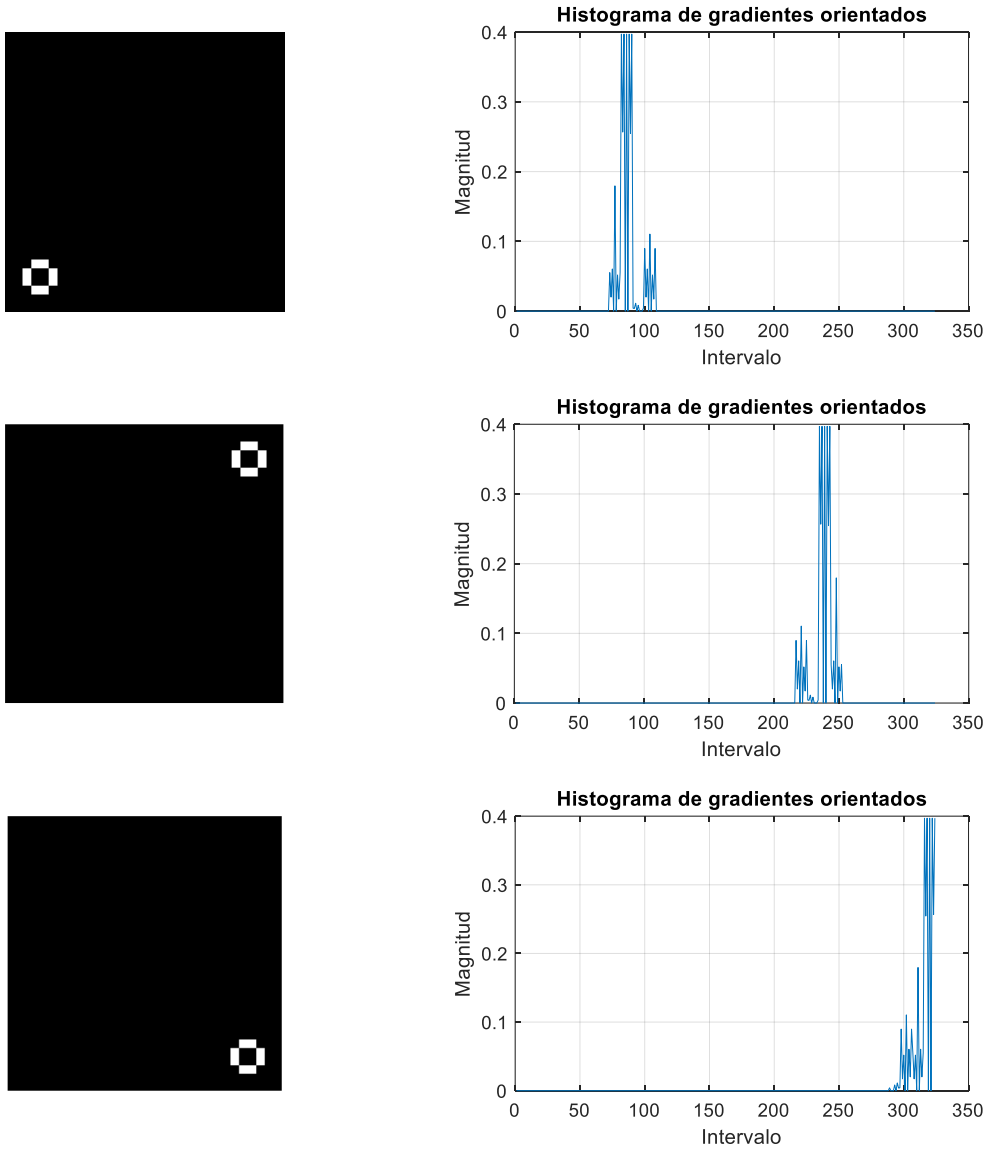


Figura 3.33. Resultados del cálculo del HOG en diferentes imágenes. (Parte 2/2).

Considere la Figura 3.34 que despliega la imagen bajo estudio dividida en las 16 celdas, cada una con su histograma denotado por hc_i donde $i \in \{1,\ldots,16\}$. La concatenación de los histogramas para la construcción del HOG está expresada por la Ec. (3.37).

$$h_{hog} = \{hc_1\ hc_2\ hc_5\ hc_6\ hc_2\ hc_3\ hc_6\ hc_7 \ldots hc_5\ hc_6\ hc_9\ hc_{10} \ldots hc_{11}\ hc_{12}\ hc_{15}\ hc_{16}\} \qquad (3.37)$$

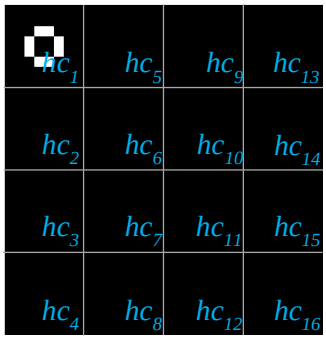


Figura 3.34. Imagen dividida en celdas para ilustrar el orden de construcción del HOG.

Este análisis llevado a cabo ayuda a definir la forma de la salida deseada r_{hog} para el diseño del filtro de correlación. Como ya se vio, hay histogramas en el HOG que capturan más información del OI que otros. Si el OI está centrado en $T(x,y,n)$, la información acerca de este es mayor en los histogramas centrales que en los histogramas de los extremos del HOG por la manera en que se van concatenando, como se muestra en la ilustración de la Figura 3.35.

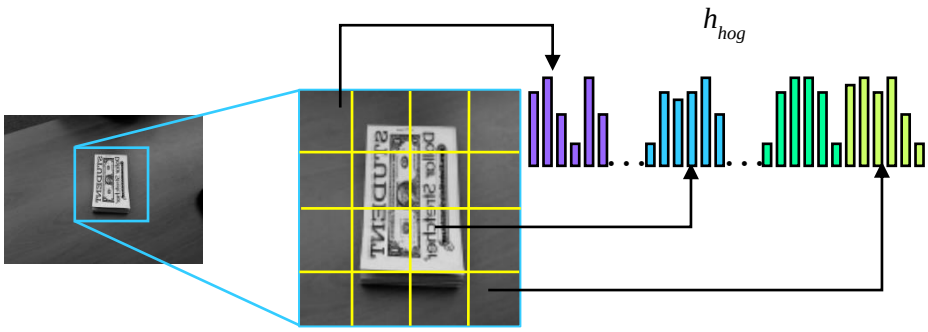


Figura 3.35. Los histogramas centrales capturan más información del OI que los histogramas en los extremos en el HOG. El cuadro pertenece a la secuencia de video *Coupon* de la base de datos OTB [53].

Este comportamiento es similar a una función gaussiana, por lo que se define la misma para r_{hog}. Además, debido a que el punto máximo en r_{hog} debe coincidir con el centro del OI, se establece que la media de la gaussiana μ_{hog} sea igual a $L_{hog}/2$ y que la desviación estándar σ_{hog} sea suficiente como para abarcar la información relevante. Sin embargo, se observa que en esta zona central del HOG hay histogramas que codifican información de fondo, es decir, son histogramas alejados del centro del OI. Para mitigar su efecto se propone que la amplitud A_{hog} de r_{hog} sea menor a uno. La Ec. (3.38) muestra la definición de r_{hog}.

$$r_{hog} = A_{hog} \exp\left(-\frac{\left(x - \mu_{hog}\right)^2}{2\sigma_{hog}^{\ 2}} \right) \tag{3.38}$$

Una vez que se tiene el HOG y la salida de correlación deseada, se puede inicializar el filtro de correlación $\overline{\Gamma_{hog}}$ al aplicar la Ec. (3.39),

$$\overline{\Gamma_{hog}} = \frac{R_{hog}\overline{H_{hog}}}{H_{hog}\overline{H_{hog}} + \lambda} = \frac{A_n}{B_n} \tag{3.39}$$

donde H_{hog} y R_{hog} es la transformada de Fourier de h_{hog} y r_{hog}, respectivamente.

Posterior a la inicialización del filtro de correlación se realiza el seguimiento del OI. A continuación, se enlistan los pasos para la determinación de la posición del objeto en el cuadro n denotado por $\rho(n)$:

1. Se genera un conjunto de muestras sobre el cuadro $I(x,y,n)$ a partir de una ventana deslizante en espiral, donde cada muestra es una porción rectangular de $I(x,y,n)$. La ventana deslizante tiene su origen $\rho(n\text{-}1)$ y se abre generando espirales, como se ilustra en la Figura 3.36. Los parámetros a configurar son el radio ra de apertura de cada espiral dado en pixeles, el ángulo α en radianes entre cada muestra tomada por la ventana deslizante y la cantidad de espirales es.

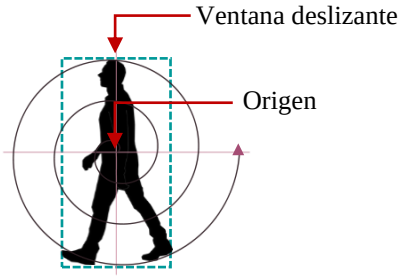


Figura 3.36. Ventana deslizante en espiral para la generación de muestras.

2. Se realiza el cálculo del HOG en cada una de las muestras, lo que genera varias z_{hog}. Luego se transforma cada z_{hog} al dominio de la frecuencia utilizando la transformada rápida de Fourier, obteniendo a Z_{hog}.
3. Cada Z_{hog} es evaluado con el filtro de correlación $\overline{\Gamma_{hog}}$ utilizando la Ec. (3.40). El resultado son una serie de mapas de respuesta ψ_{hog} en una dimensión.

$$\psi_{hog} = \mathfrak{F}^{-1}\left\{Z_{hog} \odot \overline{\Gamma_{hog}}\right\} \tag{3.40}$$

4. Se determina cuál ψ_{hog} representa a la muestra que más parecido tiene con el OI. El criterio propuesto consiste en seleccionar aquel ψ_{hog} que al ser evaluado en $L_{hog}/2$ posee el valor más alto. El ψ_{hog} elegido representa a la muestra *mt* que contiene el OI.
5. Se obtiene $\rho(n)$ al calcular el centroide de *mt*. Luego se utiliza el Z_{hog} de *mt* para actualizar el filtro de correlación a partir de las Ecs. (3.41) y (3.42).

$$A_n = \eta R_{hog_n} \odot \overline{Z_{hog_n}} + (1-\eta) A_{n-1} \tag{3.41}$$

$$B_n = \eta Z_{hog_n} \odot \overline{Z_{hog_n}} + (1-\eta) B_{n-1} \tag{3.42}$$

En la Figura 3.37 se muestra el diagrama de flujo que sigue el algoritmo *fc-hog*.

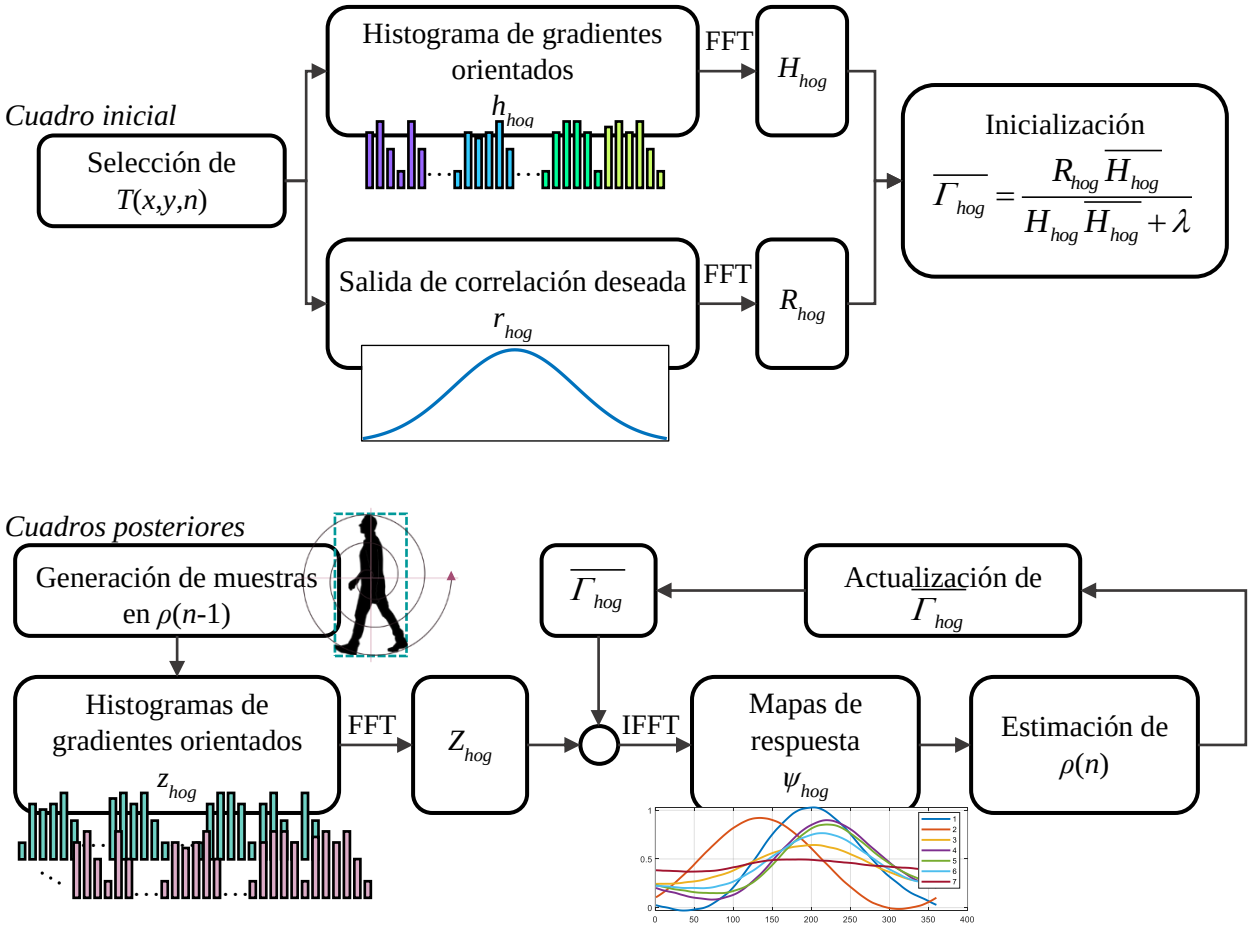


Figura 3.37. Diagrama de flujo del algoritmo *fc-hog*.

3.5.2 Filtro de correlación con el histograma de color difuso

En esta sección se presenta la implementación del filtro de correlación utilizando el histograma de color difuso descrito en la sección 3.3.2. Este algoritmo es nombrado como *fc-color* y su diseño es muy similar al presentado para *fc-hog*, ya que para ambos las características están dispuestas en forma de vector lo que modifica el diseño tradicional de filtro de correlación.

El primero paso en el diseño consiste en la inicialización del filtro de correlación. Para esto se requieren de dos elementos: la información que describe al OI y la información de la salida de correlación deseada. El primer elemento se obtiene a partir del histograma de color difuso y el segundo debe estar definido de tal manera que el punto de mayor valor coincida con el centro del OI.

A diferencia del HOG, el histograma de color difuso no contiene información sobre la distribución espacial del color, por lo que no es posible asociar una región en este con el centro del OI. Es por ello que se propone el diseño de un descriptor de la distribución espacial del color denotado por h_{dc}, inspirado en la construcción del HOG. A continuación, se describen los pasos para el cálculo de h_{dc}:

1. Se divide a $T(x,y,n)$ en celdas de tamaño fijo. Una celda es una región rectangular cuyo tamaño está dado en pixeles.
2. Se obtienen los histogramas de color difuso h_{color} para cada celda.
3. Se concatenan los h_{color}, lo que genera el h_{dc}.

En la Figura 3.38 se ilustra el procedimiento de construcción de h_{dc} donde la concatenación de los histogramas h_{color} está expresada por la Ec. (3.43). En este ejemplo la imagen se divide en 16 celdas por lo que se tienen 16 h_{color}. Como se aprecia, la concatenación es similar a la que se lleva a cabo en el HOG.

$$h_{dc} = \{h_1\ h_2\ h_3 \ldots h_{14}\ h_{15}\ h_{16}\} \tag{3.43}$$

El histograma h_{dc} describe el contenido del color en el OI y puede ser visto como un vector cuyo tamaño L_{dc} es 12 veces la cantidad de celdas formadas en $T(x,y,n)$, esto es porque cada histograma de color difuso está compuesto de 12 intervalos (ver sección 3.3.2).

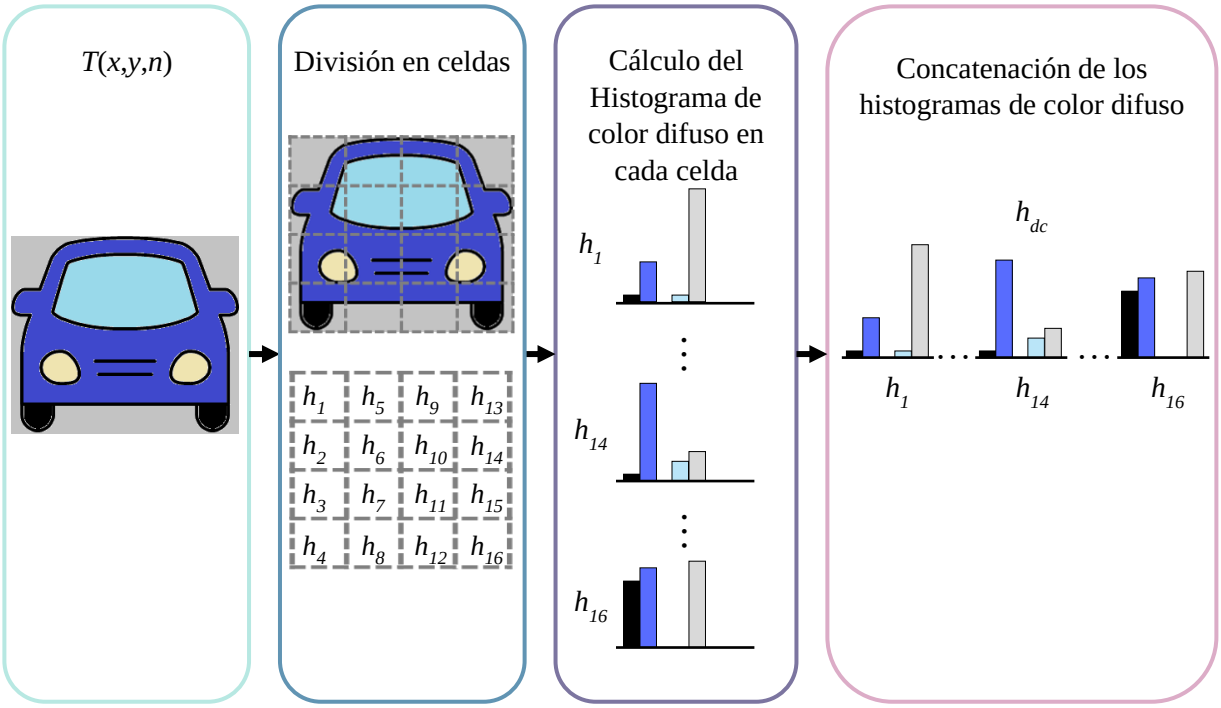


Figura 3.38. Procedimiento para la obtención del descriptor de la distribución espacial del color h_{dc}.

Por la manera en que se concatenan los histogramas de color difuso, la información acerca del OI es mayor en los histogramas centrales que en los histogramas de los extremos de h_{dc}. Este comportamiento se asemeja a una función gaussiana, por lo cual se define la misma para la salida de correlación deseada r_{dc} que se expresa por la Ec. (3.44),

$$r_{dc} = A_{dc} \exp\left(-\frac{\left(x-\mu_{dc}\right)^2}{2\sigma_{dc}^{\ 2}} \right) \tag{3.44}$$

donde A_{dc}, μ_{dc} y σ_{dc} son la amplitud, media y desviación estándar de r_{dc}. Una vez que se tiene h_{dc} y r_{dc}, se inicializa el filtro de correlación $\overline{\Gamma_{dc}}$ mediante la Ec. (3.45),

$$\overline{\Gamma_{dc}} = \frac{R_{dc}\overline{H_{dc}}}{H_{dc}\overline{H_{dc}} + \lambda} = \frac{A_n}{B_n} \tag{3.45}$$

donde H_{dc} y R_{dc} son la transformada de Fourier de h_{dc} y r_{dc}, respectivamente.

Una vez inicializado el filtro de correlación, este se utiliza para determinar ρ(n) en los cuadros subsecuentes. El procedimiento para llevar a cabo lo anterior es el mismo que el aplicado para *fc-hog*:

1. Se genera un conjunto de muestras sobre el cuadro $I(x,y,n)$ a partir de una ventana deslizante en espiral cuyo origen está en $\rho(n\text{-}1)$.
2. Se calcula el descriptor de la distribución espacial del color en cada muestra, lo que genera varias z_{dc}. Se transforma cada z_{dc} al dominio de la frecuencia utilizando la transformada rápida de Fourier, obteniendo a Z_{dc}.
3. Se evalúa cada Z_{dc} con el filtro de correlación $\overline{\Gamma_{dc}}$ mediante la Ec. (3.46), lo que genera el conjunto de mapas de respuesta ψ_{dc}.

$$\psi_{dc} = \Im^{-1}\left\{Z_{dc} \odot \overline{\Gamma_{dc}}\right\} \tag{3.46}$$

4. Se analizan los mapas de respuesta ψ_{dc} para determinar cuál de estos representa a la muestra que más parecido tiene con el OI. El criterio propuesto consiste en seleccionar aquel ψ_{dc} que al ser evaluado en $L_{dc}/2$ posee el valor más alto. El ψ_{dc} elegido representa a la muestra *mt* que contiene el OI.
5. Se obtiene $\rho(n)$ al calcular el centroide de *mt*. Luego se utiliza el Z_{dc} de *mt* para actualizar el filtro de correlación a partir de las Ecs. (3.47) y (3.48).

$$A_n = \eta R_{dc_n} \odot \overline{Z_{dc_n}} + (1-\eta)A_{n-1} \tag{3.47}$$

$$B_n = \eta Z_{dc_n} \odot \overline{Z_{dc_n}} + (1-\eta)B_{n-1} \tag{3.48}$$

En la Figura 3.39 se muestra el diagrama de flujo que sigue el algoritmo *fc-color*.

3.5.3 Filtros de correlación con las características de la red neuronal convolucional

En esta sección se presenta la implementación del filtro de correlación utilizando las características extraídas de la red neuronal convolucional VGG-16, que fueron detalladas en la sección 3.3.3.3. Este algoritmo es nombrado como *fc-cnn* e implementa los filtros de correlación en dos dimensiones.

El primer paso en el diseño de los filtros de correlación es la inicialización de los mismos. Para ello se requiere la salida de correlación deseada y las características que describen al OI. La salida de correlación deseada r_{cnn} tiene forma gaussiana y está definida por la Ec. (3.49), donde A_{cnn} es la amplitud, σ_{cnn} es la desviación estándar y (cx,cy) es el pixel central en r_{cnn}.

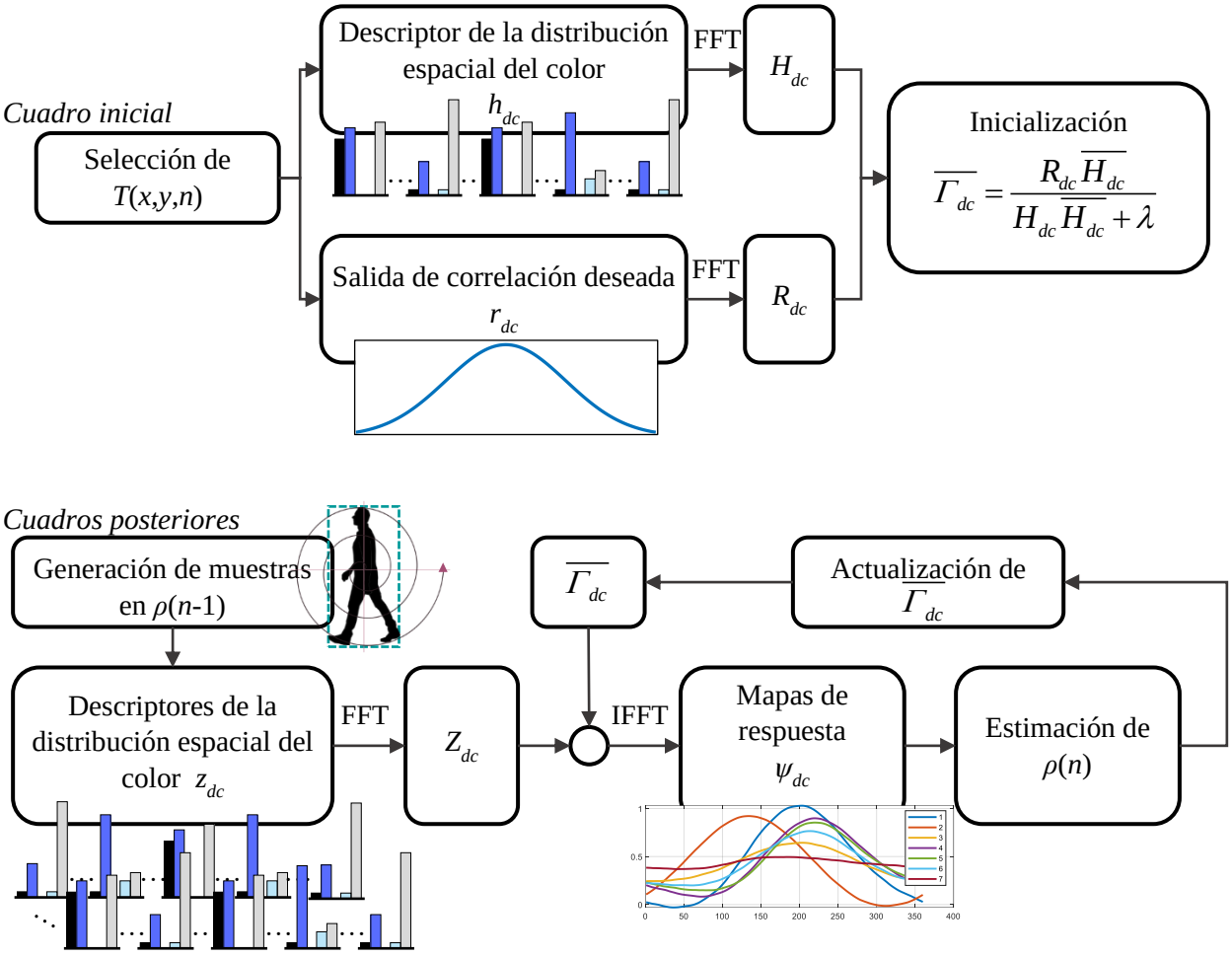


Figura 3.39. Diagrama de flujo del algoritmo *fc-color*.

$$r_{cnn} = A_{cnn} \exp\left(-\frac{(x-cx)^2 + (y-cy)^2}{2\sigma_{cnn}^{\ 2}} \right) \tag{3.49}$$

Por otro lado, las características que describen al OI son los mapas de características mc^l obtenidos de las capas de activación en la red VGG-16, donde $l \in \{1,2,\ldots,d\}$ y d es el número total de mapas de características, que equivale a la suma de la cantidad de estos seleccionados en cada capa (mp_i). Por ejemplo, si se seleccionan 3 mapas de características de 5 capas, entonces d será igual a 15.

Un aspecto importante sobre la transformada de Fourier es que en su teoría está implícita la periodicidad de la función a transformar. Esto es, cuando se transforma una imagen con Fourier

se asume que esta se repite de forma periódica. En general, las orillas de la imagen, por ejemplo, la orilla izquierda y derecha o la orilla de arriba y abajo no son iguales. Esta desigualdad en las orillas genera una discontinuidad a la hora de considerar a la imagen como periódica, lo que ocasiona que sobre el espectro se cree una cruz central sobreimpuesta sobre el resto de la información [68].

Esta cruz aparece al transformar los mapas de características al dominio de Fourier y puede afectar el desempeño de los filtros de correlación. Es por lo anterior que se utiliza el método de ventaneo, que consiste en multiplicar la imagen a transformar con una ventana que atenúe las diferencias en las orillas. En la Figura 3.40a se muestra el espectro de un mapa de características sin utilizar el método de ventaneo, por lo que aparece la cruz central sobreimpuesta. Por otro lado, en la Figura 3.40b se visualiza como el método de ventaneo con la ventana tipo Hann [69] elimina la cruz. Por observación, el mapa de características contiene información acerca de los bordes de la imagen.

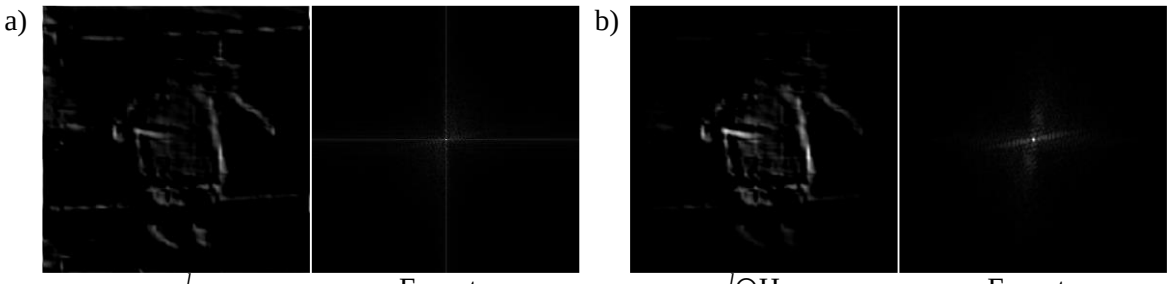


Figura 3.40. Espectro del mapa de características cuando a) no se utiliza la ventana Hann y b) cuando se utiliza la ventana Hann.

Luego de multiplicar los mapas de características con la ventana Hann [69], se realiza la inicialización del filtro de correlación $\overline{\Gamma^l_{cnn}}$ mediante la Ec. (3.50), donde MC^l es el mapa de características y R_{cnn} es la salida de correlación deseada, ambos en el dominio de la frecuencia.

$$\overline{\Gamma^l_{cnn}} = \frac{\overline{R_{cnn}}\, MC^l}{\sum_{k=1}^{d} \overline{MC^l} MC^l + \lambda} = \frac{A^l_n}{B_n} \tag{3.50}$$

De acuerdo con la Ec. (3.50), se tendrán *d* filtros de correlación, uno por cada mapa de características mc^l.

Posteriormente, estos filtros de correlación son utilizados para la estimación de $\rho(n)$ a partir de los siguientes pasos:

1. Se genera una ROI centrada en la posición $\rho(n\text{-}1)$.
2. Se extraen los mapas de características z^l_{cnn} de la ROI y se transforman al dominio de la frecuencia utilizando la transformada rápida de Fourier. A partir de esta operación se obtiene a Z^l_{cnn}.
3. Se evalúa Z^l_{cnn} con el filtro de correlación $\overline{\Gamma^l_{cnn}}$ mediante la Ec. (3.51), lo que permite obtener el mapa de respuesta ψ_{cnn}.

$$\psi_{cnn} = \Im^{-1}\left\{\frac{\sum_{l=1}^{d} A^l Z^l_{cnn}}{B+\lambda}\right\} \tag{3.51}$$

4. Se obtiene la coordenada (x,y) en ψ_{cnn} donde sucede el valor máximo de correlación.
5. Se calcula $\rho(n)$ mediante la Ec. (3.52),

$$\rho(n) = (x_{roi} + x, y_{roi} + \mathrm{y}) - 1 \tag{3.52}$$

donde (x_{roi}, y_{roi}) son las coordenadas de la esquina superior izquierda de la ROI sobre el cuadro $I(x,y,n)$.

6. Se genera una ROI centrada en la posición $\rho(n)$ calculada.
7. Se extraen los mapas de características z^l_{cnn} de la nueva ROI y se transforman al dominio de la frecuencia utilizando la transformada rápida de Fourier. A partir de esta operación se obtiene a Z^l_{cnn}.
8. Se actualiza el filtro de correlación con la nueva información Z^l_{cnn} a partir de las Ecs. (3.53) y (3.54).

$$A^l_n = \eta\overline{R_{cnn_n}} \odot Z^l_{cnn_n} + (1-\eta)A^l_{n-1} \tag{3.53}$$

$$B_n = \eta\sum_{k=1}^{d}\overline{Z^k_{cnn_n}} \odot Z^k_{cnn_n} + (1-\eta)B_{n-1} \tag{3.54}$$

En la Figura 3.41 se muestra el diagrama de flujo que sigue el algoritmo *fc-cnn*.

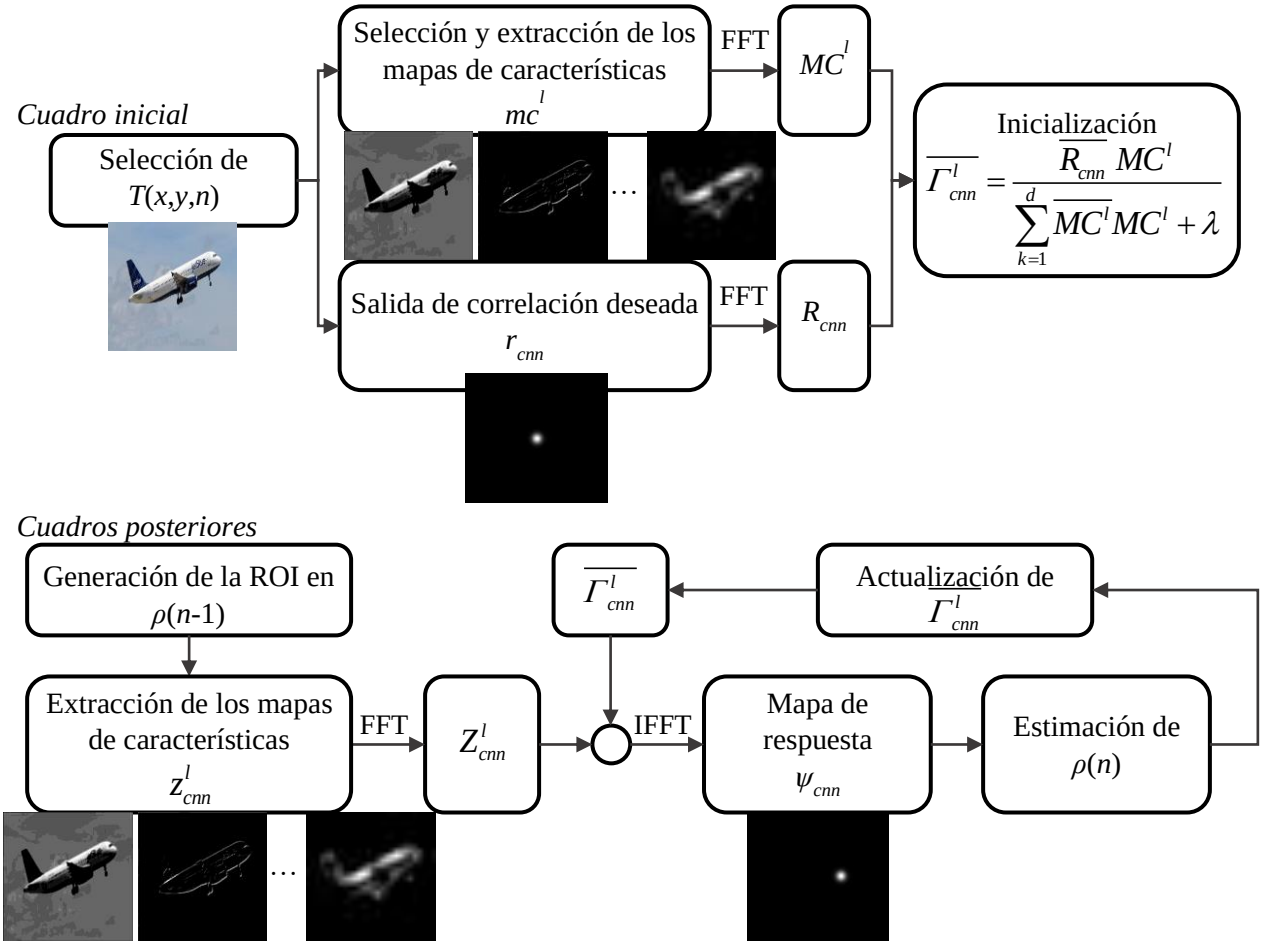


Figura 3.41. Diagrama de flujo del algoritmo *fc-cnn*.

3.6 Métodos adicionales para el algoritmo *fc-cnn*

El seguimiento de objetos es un problema desafiante debido a que los OI tienden a ser afectados por alteraciones internas como deformaciones y rotaciones, y por alteraciones externas como variaciones en la iluminación, fondos abarrotados y oclusiones. Estas alteraciones, llamadas situaciones críticas o desafíos crean un entorno más real sobre el cual el algoritmo de seguimiento de objetos operará, por lo que es común que se diseñen formas de enfrentar y manejar estos desafíos de tal manera que sea posible seguir a un objeto aún bajo estas condiciones.

En esta sección se describen aquellos elementos incorporados al algoritmo *fc-cnn* con el propósito de mejorar su desempeño en un mayor número de secuencias de video y por lo tanto un mayor número de desafíos. De forma general, estos elementos incluyen: una configuración

de la desviación estándar σ_{cnn} de la salida de correlación deseada, una nueva máscara selectora de mapas de características, una combinación lineal de los mapas de respuesta de cada capa y un método para la detección de oclusiones y deformaciones basado en la medición de la razón del pico y la región del lóbulo lateral (*peak-to-sidelobe ratio*, PSR). En las secciones posteriores se tratarán con más detalles estas modificaciones.

3.6.1 Desviación estándar en la salida de correlación deseada

La salida de correlación deseada r_{cnn} en el algoritmo *fc-cnn* tiene forma gaussiana. En r_{cnn}, el punto máximo está alineado con el centro del OI y se configura el valor de la desviación estándar σ_{cnn} con un valor fijo, de forma similar a como se sugiere en [66].

Lo que hace σ_{cnn} es definir una región en torno al punto máximo, que al ser proyectada sobre la porción rectangular $T(x,y,n)$, indica la presencia del OI sobre esa región.

Si se utiliza un valor fijo en σ_{cnn} implica que para cualquier OI se estaría creando la misma región. Esto hace que en algunos OI se desaproveche información sobre estos, ya que su tamaño es tan grande que la región creada por σ_{cnn} se vuelve pequeña en comparación. Por ejemplo, considere la Figura 3.42 donde el OI en a) es más pequeño que el OI en b). Para ambos se utilizó un valor de σ_{cnn} igual a 3 en sus salidas de correlación deseada. La salida de correlación deseada c) es adecuada para a), no obstante, la salida de correlación deseada d) no es adecuada para b), ya que gran parte de la información sobre el rostro en b) se desaprovecha.

Es por ello que se decide configurar el valor de σ_{cnn} de manera que se considere el tamaño del OI en el cuadro de video utilizado en la inicialización del filtro de correlación. Sea el ancho y alto del OI denotado por N y M, respectivamente, el valor de σ_{cnn} queda denotado por,

$$\sigma_{cnn} = f_{\sigma}\sqrt{MN} \tag{3.55}$$

donde f_{σ} es un factor utilizado para el cálculo de la desviación estándar σ_{cnn}. De esta manera, σ_{cnn} es proporcional al tamaño del OI convertido a un cuadrado. El valor de f_{σ} será explicado en el capítulo IV.

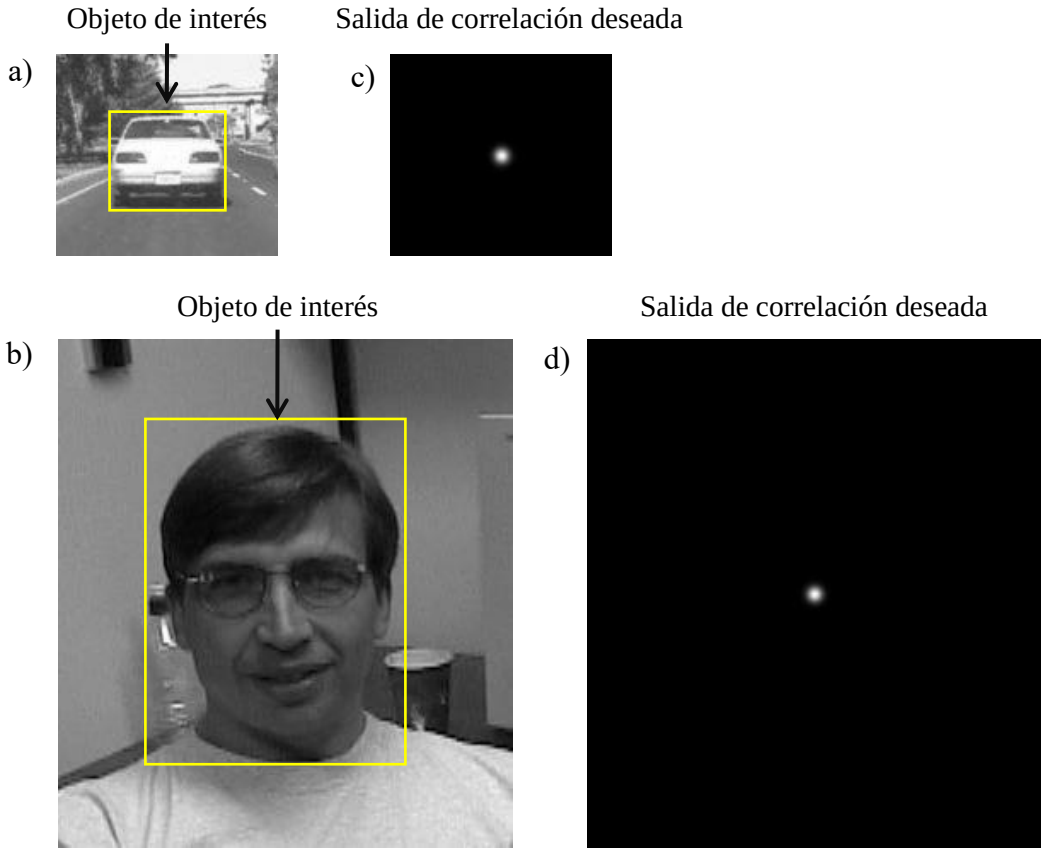


Figura 3.42. Salidas de correlación deseadas para diferentes OI, pero conservando un valor fijo en σ_{cnn}. Los cuadros fueron tomados de las secuencias a) *Car2* y b) *Dudek* de la base de datos OTB [53].

3.6.2 Máscara super-gaussiana o de orden superior

Con el propósito de realizar una mejor selección de los mapas de características (sección 3.3.3.3) se llevó a cabo una modificación de la máscara $K_i(x,y)$.

Originalmente, el algoritmo *fc-cnn* utiliza una máscara $K_i(x,y)$ con forma cosenoidal, esto es, el valor en el pixel central (cx,cy) es 1 y conforme crece la distancia a este centro el valor decrece hasta un mínimo de -1. El problema con esta definición es que en la máscara existe un solo elemento con el valor de 1, este elemento es el pixel central (cx,cy), haciendo que durante el cálculo del nivel de activación se le dé mayor importancia únicamente al pixel central del mapa de características, y deja en desventaja al resto de los pixeles que podrían describir de igual manera al OI. Es decir, la máscara $K_i(x,y)$ considera que el OI está totalmente definido

por el pixel central en el mapa de características cuando en realidad el OI abarca una región de este.

Por ello, se realiza una nueva definición en $K_i(x,y)$ mediante una función super-gaussiana o de orden superior, tal como se expresa en la Ec. (3.56).

$$K_i(x,y) = 2\exp\left(-\left(\frac{(x-cx)^2}{2\sigma_x^{\,2}} + \frac{(y-cy)^2}{2\sigma_y^{\,2}}\right)^2\right) - 1 \tag{3.56}$$

Esta nueva definición genera una región donde los valores son cercanos a 1 y cuyo alcance está definido por el par de desviaciones estándar σ_x y σ_y. Para lograr que la región abarque el contenido del OI sobre los mapas de características, se define el valor de σ_x y σ_y en función del ancho N y alto M del OI, respectivamente. Además, se conserva la característica original de la máscara en donde el rango de valores varia de -1 a 1, donde valores cercanos a -1 se dan sobre las orillas de la máscara y valores cercanos a 1 se dan en la zona central de esta. Esta característica permite que se mantenga la zona activada del OI y se inhiba la activación del fondo haciendo que el cálculo del nivel de activación de un mapa de características se refiera al nivel de activación del OI. En la Figura 3.43 se ilustran las máscaras cosenoidal y super-gaussiana.

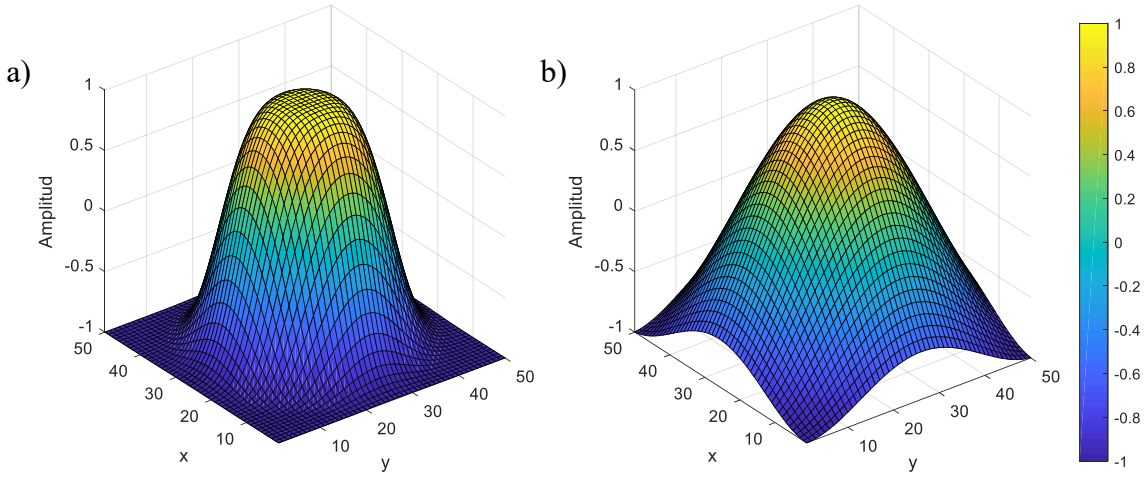


Figura 3.43. a) Máscara super-gaussiana y b) máscara cosenoidal.

3.6.3 Combinación lineal de los mapas de respuesta de los filtros de correlación

Una particularidad de los mapas de características de una red neuronal convolucional es que la complejidad de las características que contienen está relacionada con la profundidad de la capa que las genera. Por ejemplo, aquellos mapas de características que provienen de las primeras capas de la red codifican características de bajo nivel como bordes, brillo o color mientras que aquellos mapas de características que provienen de capas más profundas codifican características de mayor nivel como formas y siluetas.

En la Figura 3.44 se ilustra como la profundidad de la red neuronal convolucional se relaciona con la complejidad de las características contenidas en los mapas de características.

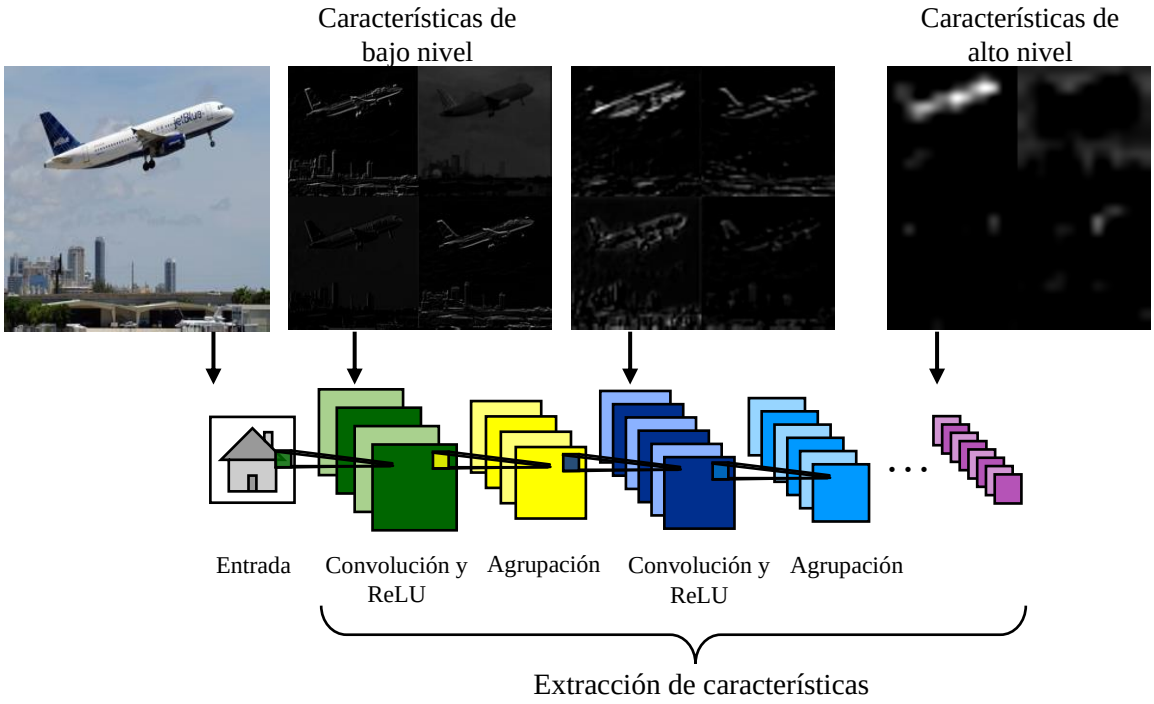


Figura 3.44. Mapas de características a lo largo de una red neuronal convolucional.

Esta particularidad puede ser aprovechada para mitigar el impacto que tienen las deformaciones del OI sobre el mapa de respuesta que se obtiene por medio de los filtros de correlación.

Originalmente, el algoritmo *fc-cnn* implementa *d* filtros de correlación $\overline{\Gamma^{l}_{cnn}}$, uno por cada mapa de características. Cada uno de estos filtros genera un mapa de respuesta ψ^{l}. De la Ec. (3.51), estos mapas son combinados para generar a ψ_{cnn} mediante la Ec. (3.57).

$$\psi_{cnn} = \Im^{-1}\left\{\frac{\sum_{l=1}^{d} A^{l} Z^{l}_{cnn}}{B+\lambda}\right\} = \sum_{l=1}^{d} \psi^{l} \tag{3.57}$$

En la expresión anterior, los mapas de respuesta ψ^{l} tienen la misma importancia cuando son combinados para generar a ψ_{cnn}, y por ende los mapas de características z^{l}_{cnn} son tratados por igual. Ahora, de manera similar que en [33], se le dará una ponderación a los mapas de respuesta ψ^{l} proporcional al nivel de profundidad que tiene la capa que produce el mapa de características.

Para esto, considere a Ψ como el mapa de respuesta final que será utilizado para la determinación de la posición del OI dado por,

$$\Psi = \sum_{i} \omega^{i} \psi^{i}_{cnn} \tag{3.58}$$

donde ω^{i} es un valor real positivo que pondera al mapa de respuesta ψ^{i}_{cnn} de la capa *i*. Los mapas de respuesta ψ^{i}_{cnn} se obtienen mediante,

$$\psi^{i}_{cnn} = \sum_{l=1}^{mp_i} \psi^{l} \tag{3.59}$$

donde mp_i el número de mapas de características extraídos de la capa *i*, y ψ^{l} es el mapa de respuesta obtenido a partir de un mapa de características.

La ponderación ω^{i} de la Ec. (3.58) debe ser proporcional a la profundidad de la capa *i*, de manera que aquellos mapas de respuesta provenientes de capas profundas tengan un valor de ponderación más alto que aquellos mapas de respuesta que provienen de capas tempranas. Se propone que los valores de ω^{i} estén linealmente distribuidos comenzando en 0.25 hasta 1 en pasos de 0.0625.

La razón de darle menor ponderación a los mapas de respuesta obtenidos de capas tempranas es que estos cambian con más facilidad cuando se presenta alguna deformación sobre el OI, ya que son producidos por la evaluación de mapas de características que modelan información de bajo nivel. Esta distorsión en el mapa de respuesta vuelve menos precisa la estimación de la

posición del OI. En cambio, las características de alto nivel, es decir, aquellas obtenidas de capas profundas, no cambian con tal facilidad debido a que su información ya está a un nivel semántico, por lo tanto, los mapas de respuesta generados por capas profundas son menos distorsionados por las deformaciones. Bajo este punto de vista parece que es mejor utilizar solo características de alto nivel, sin embargo, las características de bajo nivel se mantienen en el diseño ya que estas ayudan a separar el OI de distractores similares en la escena [70].

Debido a que se utilizan 13 capas de activación se tendrán 13 mapas de respuesta $\psi^{i}{}_{cnn}$ y por lo tanto 13 valores de ponderación ω^{i}. En la Figura 3.45 se muestra un esquema que ilustra como se realiza la combinación lineal para la generación del mapa de respuesta final Ψ.

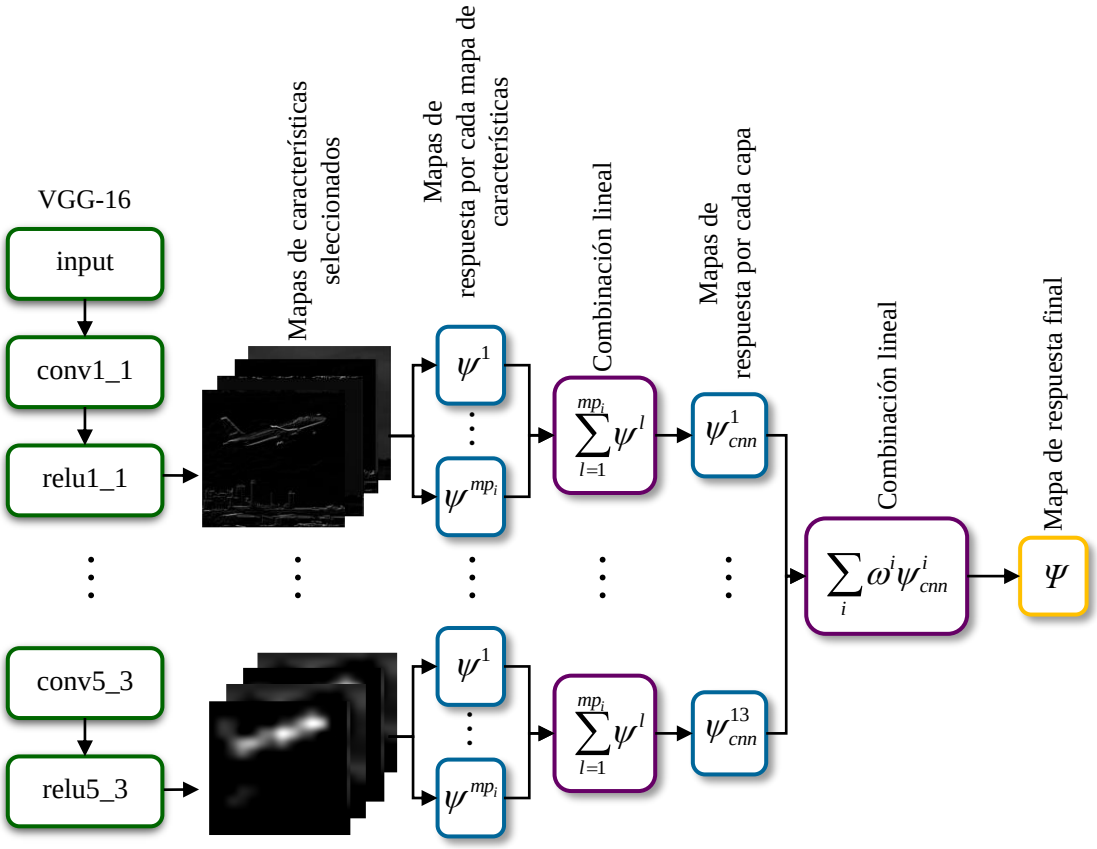


Figura 3.45. Combinación lineal de los mapas de respuesta de los filtros de correlación.

3.6.4 Método para la detección de oclusiones y deformaciones

Las oclusiones se refieren al evento donde los atributos clave utilizados para dar seguimiento al OI dejan de estar disponibles debido a que se encuentra obstruido por algún elemento del escenario. Por otro lado, una deformación se refiere al evento donde el OI se ve alterado debido a un cambio de forma o apariencia. Aunque sus definiciones son distintas, las oclusiones y deformaciones provocan el mismo comportamiento sobre el mapa de respuesta que genera el filtro de correlación ya que en ambos casos ocurre el mismo evento sobre el cuadro de video: un cambio en el valor de los pixeles.

Cuando el filtro de correlación se inicializa, este aprende acerca del aspecto que tiene el OI en el cuadro de video de inicialización. En cuadros posteriores, si el OI sufre pequeñas deformaciones el mapa de respuesta será muy parecido a la salida de correlación deseada con la que se inicializó el filtro de correlación. En cambio, si el OI sufre deformaciones considerables u oclusiones, el mapa de respuesta tendrá grandes variaciones respecto a la salida de correlación deseada. En la Figura 3.46 se ilustra como el mapa de respuesta cambia cuando se presenta una oclusión o deformación.

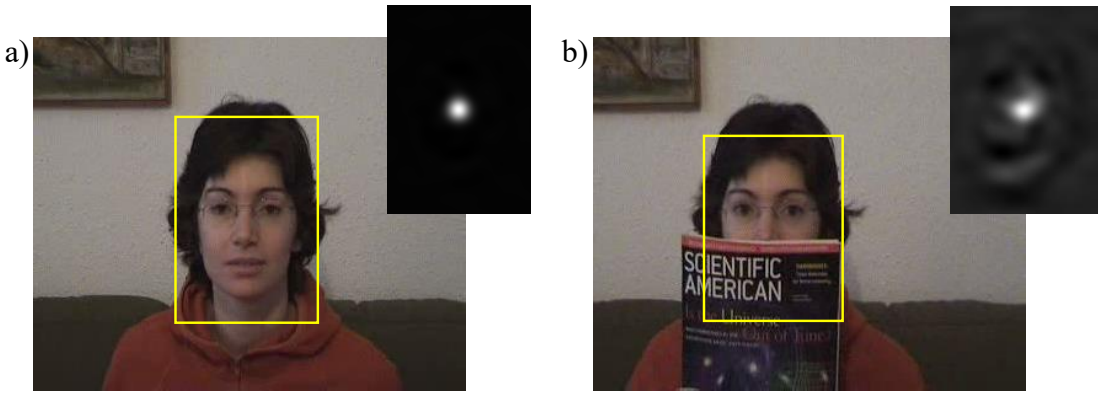


Figura 3.46. Ejemplo del mapa de respuesta cuando el OI está a) libre de oclusiones y deformaciones y b) cuando se presentan estas.

Este comportamiento en el mapa de respuesta puede ser utilizado en la detección de oclusiones y deformaciones. Para esto se usa la razón entre el pico y la región del lóbulo lateral (R_{sl}) del mapa de respuesta, esta métrica es conocida como *PSR* y mide la fuerza que tiene el pico de correlación. Su expresión está dada por la Ec. (3.60),

$$PSR(n) = \frac{\psi_{max}(n) - \mu_{sl}(n)}{\sigma_{sl}(n)} \tag{3.60}$$

donde ψ_{max} es la magnitud del pico de correlación en el mapa de respuesta final, μ_{sl} y σ_{sl} es la media y desviación estándar de R_{sl} y n es el número de cuadro de video. En la Figura 3.47 se ilustra el mapa de respuesta final y a R_{sl} que está centrada en la coordenada donde sucede ψ_{max} y que excluye una región cercana a esta equivalente a 3 σ_{cnn}. El tamaño de R_{sl} es 0.3 veces el tamaño del mapa de respuesta final Ψ.

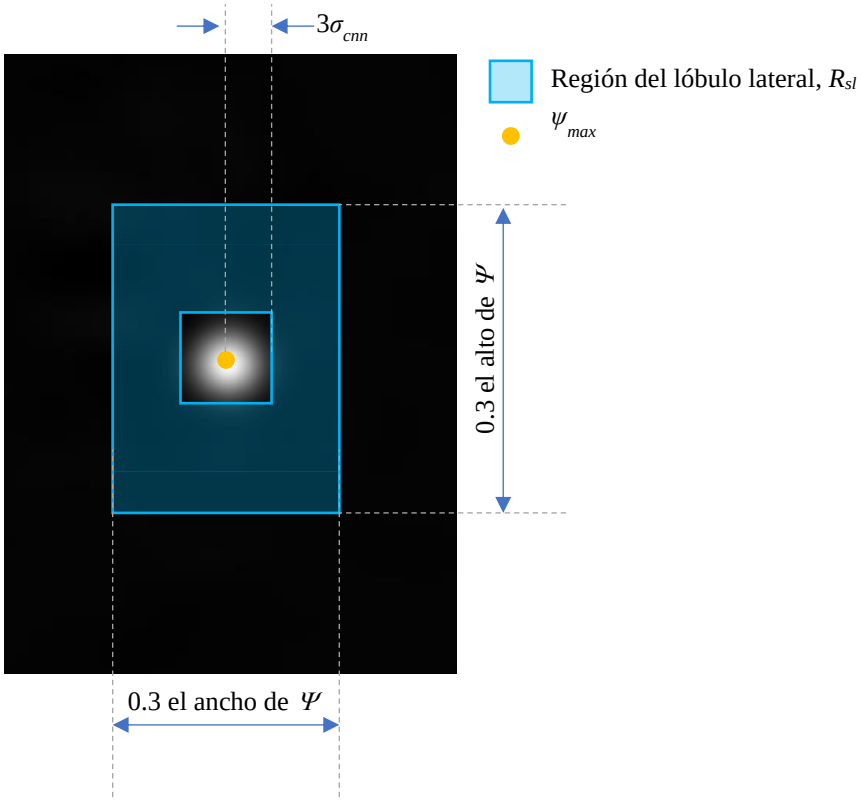


Figura 3.47. Mapa de respuesta final y región del lóbulo lateral.

Lo que hace *PSR* es medir que tan diferente es ψ_{max} de los valores de la región del lóbulo lateral: si ψ_{max} es mayor que μ_{sl}, y σ_{sl} es pequeña significa que la región de lóbulo lateral tiene valores pequeños y cercanos entre si haciendo que el valor de *PSR* sea alto, por otro lado, si ψ_{max} es parecido a μ_{sl}, y σ_{sl} es grande entonces significa que la región de lóbulo lateral tiene fluctuaciones haciendo que el valor de *PSR* sea bajo. En conclusión, entre más grande sea el valor de *PSR* mejor mapa de respuesta se tiene.

Una forma simple de detectar oclusiones y deformaciones es utilizando un umbral que indique el nivel mínimo en el valor *PSR* para considerar que el OI no está siendo ocluido ni se ha deformado. Sin embargo, establecer un umbral sobre el valor de *PSR* no es lo más conveniente ya que diferentes OI en distintas secuencias de video generan distintos valores de *PSR* aceptables.

Es por lo anterior que se utiliza un método similar al presentado en [71] donde se toma en cuenta los valores *PSR* de cuadros donde no se presentaron oclusiones ni deformaciones.

Considere que el OI no presenta oclusiones ni deformaciones en el cuadro utilizado para la inicialización del filtro de correlación, la detección de oclusiones y deformaciones se realiza en los cuadros siguientes y empieza con el cálculo de $P\hat{S}R$, que es el promedio ponderado de los valores de *PSR* de los mapas de respuesta de un C_{cf} número de cuadros recientes donde no se presentaron oclusiones ni deformaciones y que se obtiene por la Ec. (3.61),

$$PSR = \frac{1}{C_{cf}} \sum_{m=1}^{C_{cf}} \omega_{PSR}(m) PSR_{cf}(m) \tag{3.61}$$

donde ω_{PSR} son las ponderaciones obtenidas por la Ec. (3.62) y PSR_{cf} son los valores *PSR* de los mapas de respuesta de cuadros donde no se presentaron oclusiones ni deformaciones. De esta manera, la ponderación de los valores de *PSR* de cuadros más antiguos es menor que los de cuadros recientes.

$$\omega_{PSR}(m) = 1 - e^{-m+1} \qquad m = 1, 2, ..., C_{cf} \tag{3.62}$$

Luego, se obtiene el nivel de incertidumbre N_{inc} que está definido por la Ec. (3.63).

$$N_{inc}(n) = \frac{PSR}{PSR(n)} \tag{3.63}$$

El nivel de incertidumbre N_{inc} indica el grado de desconfianza que se tiene sobre el mapa de respuesta. Si *PSR*(*n*) tiene un valor parecido o mayor que $P\hat{S}R$ entonces N_{inc} será cercano o menor a 1, lo que es bueno ya que significa que el mapa de respuesta se comporta de forma similar o mejor en comparación con los mapas de respuesta de cuadros recientes donde no se presentaron oclusiones ni deformaciones. Por otro lado, si *PSR*(*n*) tiene un valor menor que $P\hat{S}R$ entonces N_{inc} tendrá un valor mayor, lo que indica la presencia de oclusiones y deformaciones ya que el mapa de respuesta actual es diferente de lo establecido como normal.

Por lo anterior, considere a *occdef* el estado lógico de la presencia de oclusiones y deformaciones. Su definición queda expresada por la Ec. (3.64), donde τ_{inc} es el umbral que define cuando se detecta o no una oclusión o deformación.

$$occdef(n) = \begin{cases} 1 & \text{si } N_{inc}(n) > \tau_{inc} \\ 0 & \text{de otra forma} \end{cases} \tag{3.64}$$

Durante la práctica se encontró que un valor de 2.5 para τ_{inc} detecta adecuadamente las oclusiones y deformaciones en el OI. En la Figura 3.48 se ilustran la gráfica del nivel de incertidumbre N_{inc} a lo largo de una sección de la secuencia de video *faceocc1* de la base de datos OTB [53].

Cuando el estado *occdef* en el cuadro actual tiene valor de 1 significa que el OI está siendo ocluido o presenta alguna deformación considerable. Ante este evento se establece que el valor del parámetro de aprendizaje η sea igual a 0, esto es, que el filtro de correlación no aprenda y que además se active un módulo que permita el seguimiento del OI bajo estas condiciones.

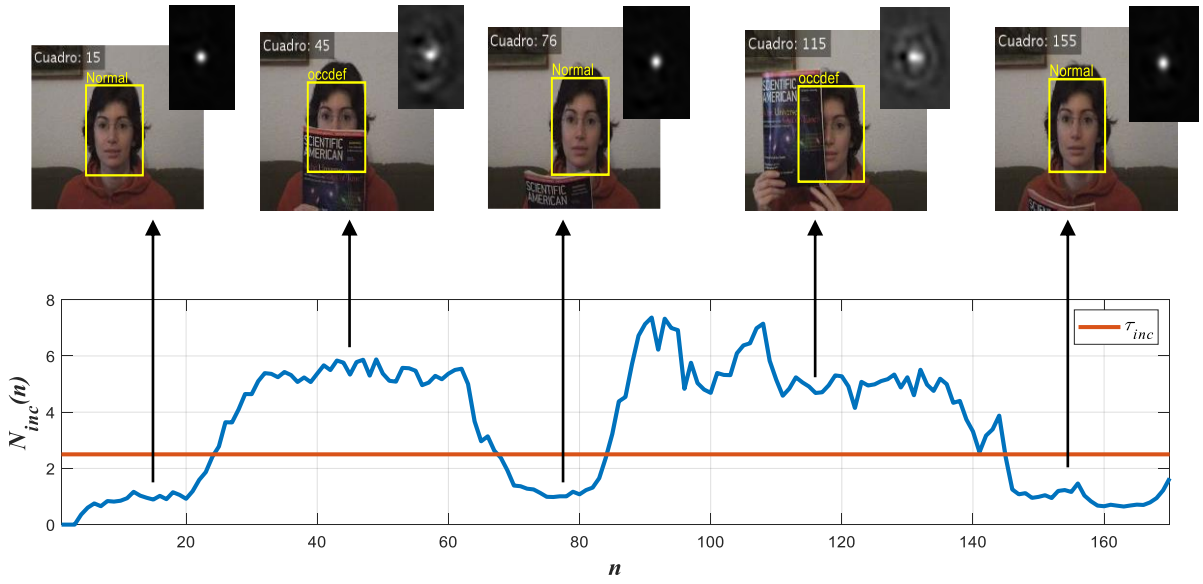


Figura 3.48. Funcionamiento del método de detección de oclusiones y deformaciones.

Para esta investigación se definieron dos versiones de este módulo, la primera de ellas utiliza un filtro de Kalman, que es conocido por su capacidad de estimación del estado de un proceso [69]-[70], que, para este caso, es la posición de OI. La segunda versión consiste en un

incremento del tamaño de la región de interés con el propósito de abarcar una mayor área durante la posible oclusión a la espera de la reaparición del OI. Ambas versiones incorporan todos los elementos descritos en esta sección, por lo que se generan los algoritmos nombrados como *fc-cnn_kf* y *fc-cnn_inc*, respectivamente.

El filtro de Kalman fue implementado con la ayuda de la función de Matlab® `configureKalmanFilter()` y es configurado con los siguientes valores: se estableció un modelo de movimiento en velocidad constante, la localización inicial se definió como la posición del OI en el cuadro de inicialización, se configuró un error inicial de $1x10^5$, un ruido del movimiento en 10 para ambas direcciones y un error de medición en 5. Por otro lado, para el incremento de ROI de definió que esta sucediera a una escala de 1.1 respecto al tamaño de la ROI original.

Los valores de C_{cf} y η cuando no hay oclusión ni deformación son definidos en el capítulo IV.

CAPÍTULO IV. RESULTADOS Y CONCLUSIONES

En este capítulo se presenta un resumen de los resultados obtenidos en el capítulo II acerca de la evaluación de métodos aplicados al seguimiento de objetos utilizando el instrumento de evaluación propuesto. Estos resultados dieron a conocer el enfoque principal que lleva el algoritmo de seguimiento de objetos de esta investigación. Posteriormente, y con la finalidad de probar el funcionamiento de los algoritmos presentados en el capítulo III, se realizó una evaluación de estos para medir su desempeño en términos de precisión, éxito y velocidad de procesamiento ante un mismo conjunto de secuencias de video. Los resultados de esta evaluación conducen a la obtención del algoritmo FCCNN y son mostrados en este capítulo.

4.1 Evaluación de métodos aplicados al seguimiento de objetos con el instrumento de evaluación

En el capítulo II se dio a conocer un instrumento de evaluación de métodos aplicados al seguimiento de objetos que considera tres criterios: documentación, desempeño y desafíos superados. Cada uno de estos está compuesto de varios subcriterios y métricas que otorgan un puntaje determinado. El propósito del instrumento es definir qué métodos presentan las mejores características y a su vez establecer cuál es el enfoque que otorga mejores resultados en el área de seguimiento de objetos.

Se seleccionaron un total de 29 métodos aplicados al seguimiento de objetos de literatura reciente (Tabla 2.1 y Tabla 2.2.) que implementan distintos enfoques como filtros de partículas, filtros de correlación y redes neuronales convolucionales. Los resultados de la evaluación (Figura 2.1) muestran que de los 10 primeros lugares 5 utilizan un enfoque de filtros de correlación, algunos de ellos combinados con enfoques de redes neuronales convolucionales, mientras que los 5 restantes se distribuyen entre filtros de partículas, modelos de probabilidad, filtros de Kalman y clasificadores.

Para comprobar la congruencia de estos resultados se llevó a cabo una comparación con los resultados del posicionamiento presentado por VOT2017 *challenge* [39], donde los 10 primeros lugares son ocupados igualmente por métodos que utilizan filtros de correlación y redes neuronales convolucionales (Tabla 2.3).

Adicionalmente, los 10 primeros lugares de VOT2017 *challenge* fueron evaluados con el instrumento de evaluación para comparar los posicionamientos que obtienen ambos. Esto dio lugar a 2 experimentos: en el primero se evaluaron 5 métodos de VOT2017 *challenge* ya que solo estos tienen una fuente de información escrita necesaria para utilizar el instrumento de evaluación. En el segundo experimento se evaluaron los 10 algoritmos de VOT2017 *challenge* dentro del criterio *De* en la métrica de éxito De_e. Los resultados de ambos experimentos (Tabla 2.5 a Tabla 2.8) muestran congruencia con el posicionamiento de VOT2017 *challenge*, ya que 3 de los 5 algoritmos del primer experimento y 5 de los 10 del segundo, están posicionados dentro de los 10 primeros lugares.

Los resultados del instrumento de evaluación fueron de apoyo para la determinación del enfoque en el algoritmo de seguimiento de objetos de esta investigación el cual es el de filtros de correlación.

4.2 Evaluación de los algoritmos de seguimiento de objetos desarrollados

En esta sección se presenta la evaluación de los algoritmos de seguimiento de objetos que se muestran en la Figura 4.1. Los rectángulos verdes y rojos corresponden a aquellos que fueron descritos en el capítulo III, mientras que los rectángulos amarillos son varias configuraciones de *fc-cnn_inc*. En total se tienen 9 algoritmos de seguimiento de objetos, todos ellos basados en filtros de correlación. Con esta evaluación se determinará cuál de los 9 obtiene mejores resultados y por lo tanto cuál corresponde al algoritmo FCCNN de esta investigación.

La metodología de evaluación utilizada es la presentada por Wu *et al.* [53] llamada evaluación de una sola pasada (OPE) que consiste en inicializar los algoritmos con la información del primer cuadro de video y ejecutarlos hasta que finalice la secuencia. Las métricas para medir el desempeño son la precisión y el éxito cuyos resultados se darán a conocer por medio de sus correspondientes gráficas que han sido explicadas en la sección 3.1.1.

La base de datos utilizada es la presentada por Wu *et al.* [53] llamada OTB que está compuesta por 100 secuencias de video y que ha sido descrita en la sección 3.2.

Todos los algoritmos evaluados fueron ejecutados sobre el mismo equipo con las siguientes características: procesador Ryzen 5 2600x 6 núcleos @4.2 GHz CPU, 16GB de RAM @3200 MHz, GPU GTX 1070 Ti 8 GB GDDR5 y SSD.

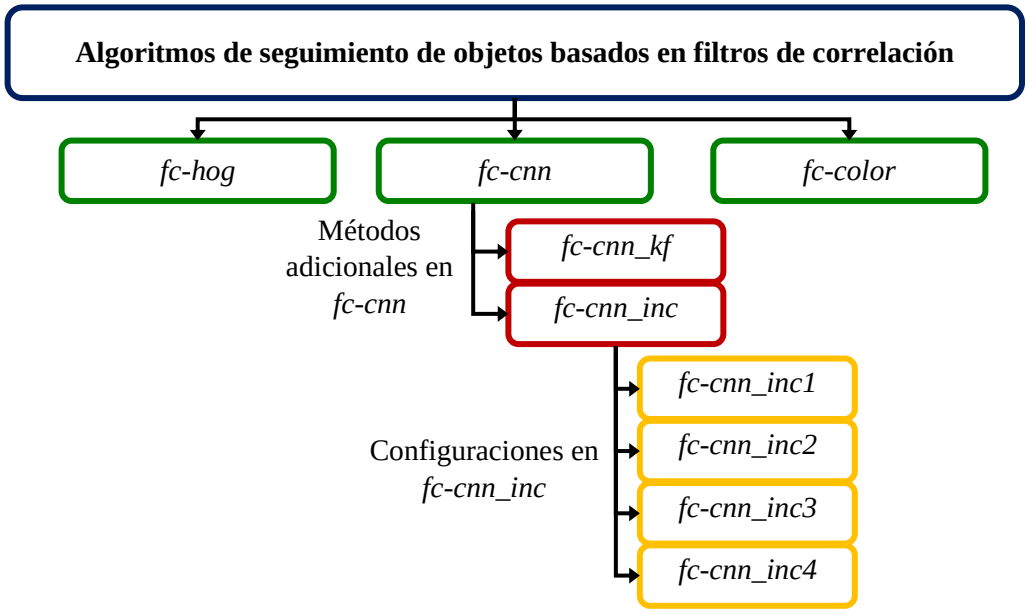


Figura 4.1. Algoritmos de seguimiento de objetos basados en filtros de correlación.

A lo largo de esta sección se describirán las características de los algoritmos y la configuración de sus parámetros.

Primero fueron evaluados los algoritmos *fc-hog*, *fc-color* y *fc-cnn*. Como ya se vio en el capítulo III, los algoritmos *fc-hog* y *fc-color* implementan un filtro de correlación en una dimensión utilizando el histograma de gradientes orientados y el descriptor de la distribución espacial del color, mientras que *fc-cnn* implementa los filtros de correlación en dos dimensiones con las características extraídas de una red CNN.

El propósito de esta primera evaluación es definir cuál de los algoritmos representa una opción con oportunidad de producir mejores resultados si se sigue trabajando y mejorando y el de explorar la posibilidad de una fusión entre estos.

Se seleccionaron 20 secuencias de video de la base de datos OTB [53] para la evaluación de estos algoritmos, cuya dificultad va desde una baja hasta moderada. En la dificultad baja se tienen pequeñas variaciones en la iluminación, deformaciones leves, entre otras, y en la dificultad moderada se tienen oclusiones parciales y deformaciones considerables.

La configuración de parámetros en cada uno de los algoritmos se muestra en la Tabla 4.1. Algunos de estos valores fueron elegidos mediante un análisis empírico que consiste en un

ajuste de los parámetros guiado por el resultado cualitativo obtenido en la práctica, mientras que otros fueron tomados de la literatura.

Tabla 4.1. Configuración de parámetros de los algoritmos *fc-hog*, *fc-color* y *fc-cnn*.

fc-hog		
Parámetro	**Valor**	
Tamaño de la celda	[8 8] pixeles	[54]
Tamaño de la superposición	[0 0] bloques	
C	[1 1] celdas	
S	9 intervalos	[54]
Orientación	Sin signo	[54]
A_{hog}	0.8	
μ_{hog}	0.5 L_{hog}	
σ_{hog}	0.25 L_{hog}	
λ	0	
η	0.125	
ra	3 pixeles	
α	π/10 radianes	
es	2	

fc-color	
Parámetro	**Valor**
Tamaño de la celda	[8 8] pixeles
A_{dc}	0.8
μ_{dc}	0.5 L_{dc}
σ_{dc}	0.125 L_{dc}
λ	0.001
η	0.001
ra	5 pixeles
α	π/3 radianes
es	1

fc-cnn		
Parámetro	**Valor**	
A_{cnn}	1	
σ_{cnn}	1	
λ	0.001	
η	0.01	[33]
mp	6 para cada capa	

En la Figura 4.2 se muestran las gráficas de precisión y éxito de cada algoritmo evaluado en las 20 secuencias de video, donde RP es razón de precisión y RE es razón de éxito. El algoritmo con mejor desempeño es *fc-cnn* y alcanza valores promedio de 0.7 para precisión y 0.604 para éxito. Seguido de *fc-cnn*, está *fc-color* y obtiene valores promedio de 0.411 en precisión y 0.321 en éxito. Finalmente, el algoritmo *fc-hog* obtiene un promedio de 0.31 en precisión y 0.211 en éxito.

Esta diferencia entre los desempeños de *fc-cnn* y los algoritmos *fc-hog* y *fc-color* tiene su origen en el método de ventana deslizante aplicado a estos últimos. Debido a que están diseñados en una dimensión, no es posible obtener un par de coordenadas (x,y) de los mapas de respuesta que indique la posición del OI, por lo que el ventaneo es una estrategia implementada para la solución de esta desventaja. La estrategia consiste en asignar la posición del OI como el centroide de aquella muestra con el mayor parecido al OI (vea secciones 3.5.1 y 3.5.2). Sin

embargo, el ventaneo tiene un error agregado perteneciente al muestreo y para obtener mejores resultados debe existir al menos una muestra que logre encerrar por completo al OI. Esto podría lograrse al incrementar el número de estas, sin embargo, este aumento implica un mayor tiempo de procesamiento. De acuerdo con la configuración de parámetros mostrada en la Tabla 4.1, el algoritmo *fc-color* genera un total de 7 muestras y el algoritmo *fc-hog* genera 41 muestras, contando aquella que se crea en el origen del muestreo.

Por otra parte, *fc-cnn* si permite obtener el par de coordenadas (*x*,*y*) del mapa de respuesta y asignarlo directamente como la posición del OI.

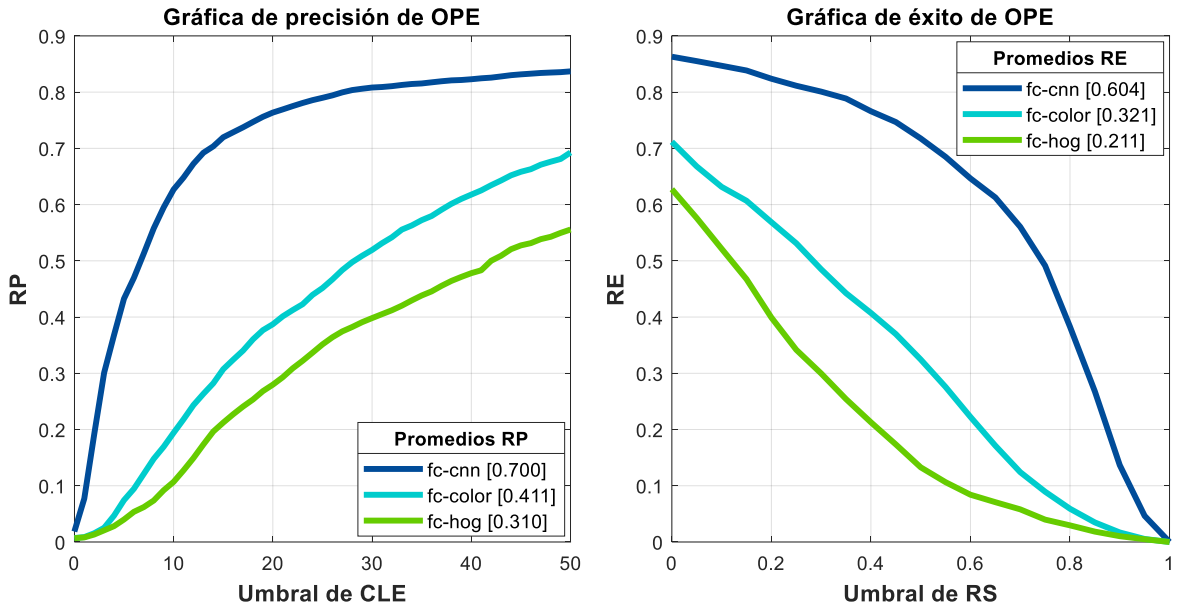


Figura 4.2. Gráficas de precisión y éxito de los algoritmos *fc-hog*, *fc-color* y *fc-cnn* para 20 secuencias de video.

En la Tabla 4.2 se enlistan los resultados de precisión y éxito para cada uno de los algoritmos y cada secuencia de video utilizada en esta evaluación. A pesar de que *fc-cnn* es el que obtuvo mejores resultados, vemos que existen secuencias para las cuales los algoritmos *fc-color* y *fc-hog* logran resultados de desempeño competentes.

Para la comparación entre *fc-cnn* y *fc-color* se tiene que las secuencias de *biker*, *dog*, *skater* y *vase* producen resultados de desempeño de precisión y éxito similares. Estas secuencias

presentan los desafíos de rotaciones sobre el plano y deformaciones leves lo que indica un buen manejo sobre estos desafíos por parte de los algoritmos.

Tabla 4.2. Resultados de precisión y éxito para cada una de las 20 secuencia de video.

	Nombre de la secuencia	*fc-cnn*		*fc-color*		*fc-hog*	
		Precisión	**Éxito**	**Precisión**	**Éxito**	**Precisión**	**Éxito**
1	Biker	0.466	0.350	0.429	0.257	0.288	0.087
2	Boy	0.770	0.650	0.137	0.089	0.238	0.136
3	CarDark	0.969	0.881	0.466	0.195	0.714	0.312
4	Coupon	0.908	0.846	0.368	0.426	0.308	0.268
5	Crossing	0.945	0.767	0.659	0.284	0.656	0.254
6	Dancer	0.851	0.773	0.617	0.638	0.138	0.224
7	Dancer2	0.840	0.780	0.337	0.537	0.370	0.271
8	David2	0.850	0.574	0.625	0.271	0.541	0.245
9	Dog	0.694	0.361	0.672	0.315	0.167	0.095
10	Fish	0.897	0.839	0.087	0.079	0.116	0.178
11	Freeman1	0.450	0.320	0.109	0.072	0.677	0.401
12	Gym	0.094	0.077	0.716	0.659	0.356	0.381
13	Human2	0.128	0.295	0.046	0.132	0.066	0.100
14	Human8	0.932	0.793	0.680	0.527	0.064	0.032
15	Man	0.952	0.835	0.322	0.184	0.496	0.242
16	Mhyang	0.827	0.740	0.328	0.307	0.174	0.164
17	Panda	0.497	0.305	0.336	0.119	0.383	0.163
18	Skater	0.615	0.596	0.600	0.561	0.128	0.260
19	Toy	0.680	0.628	0.143	0.193	0.187	0.221
20	Vase	0.642	0.675	0.548	0.583	0.142	0.185
Promedio		**0.700**	**0.604**	**0.411**	**0.321**	**0.310**	**0.211**
Desviación estándar		**0.260**	**0.236**	**0.223**	**0.199**	**0.208**	**0.096**
Velocidad de procesamiento (FPS)		**4.435**		**0.806**		**15.775**	

De igual forma sucede en la comparación de *fc-color* y *fc-hog* en la métrica de precisión para las secuencias de *panda*, *crossing*, y *dancer2*. Estas secuencias presentan baja resolución, deformaciones y variaciones en la iluminación. El buen manejo de esta última por parte del algoritmo *fc-hog* se debe principalmente a que durante la construcción del histograma de gradientes orientados se realiza una normalización sobre la magnitud del gradiente para minimizar las variaciones de la iluminación.

También se puede apreciar que existen secuencias en las que los algoritmos *fc-color* y *fc-hog* destacan por encima de *fc-cnn*, tal es el caso de las secuencias *gym* y *freeman1*. Estas secuencias

tienen la característica de que presentan deformaciones considerables, lo que indica que el algoritmo *fc-cnn* encuentra complicado el manejo de esta situación. Esto se debe a que los mapas de respuesta de las capas tempranas de la CNN cambian con facilidad ante las deformaciones, haciendo que el mapa de respuesta final se altere y que la estimación de la posición sea menos exacta.

Por último, en la comparación de tiempos de procesamiento, el algoritmo *fc-hog* obtiene el mejor resultado alcanzando los 15.775 FPS, luego está *fc-cnn* con 4.435 FPS y finalmente *fc-color* con 0.806 FPS. Estos resultados se deben principalmente a las operaciones involucradas en la extracción de características de cada algoritmo, donde para *fc-hog* se tiene el cálculo del gradiente y su acomodo en un histograma, para *fc-cnn* se realiza la propagación de los datos sobre la red VGG-16 y la selección de los mapas de características y para *fc-color* se tiene la fusificacion, varias operaciones en el plano difuso y por último la defusificación.

En la Figura 4.3 y Figura 4.4 se muestra la evaluación cualitativa de cada algoritmo utilizando las secuencias de video *CarDark*, *Crossing*, *David2* y *Gym*.

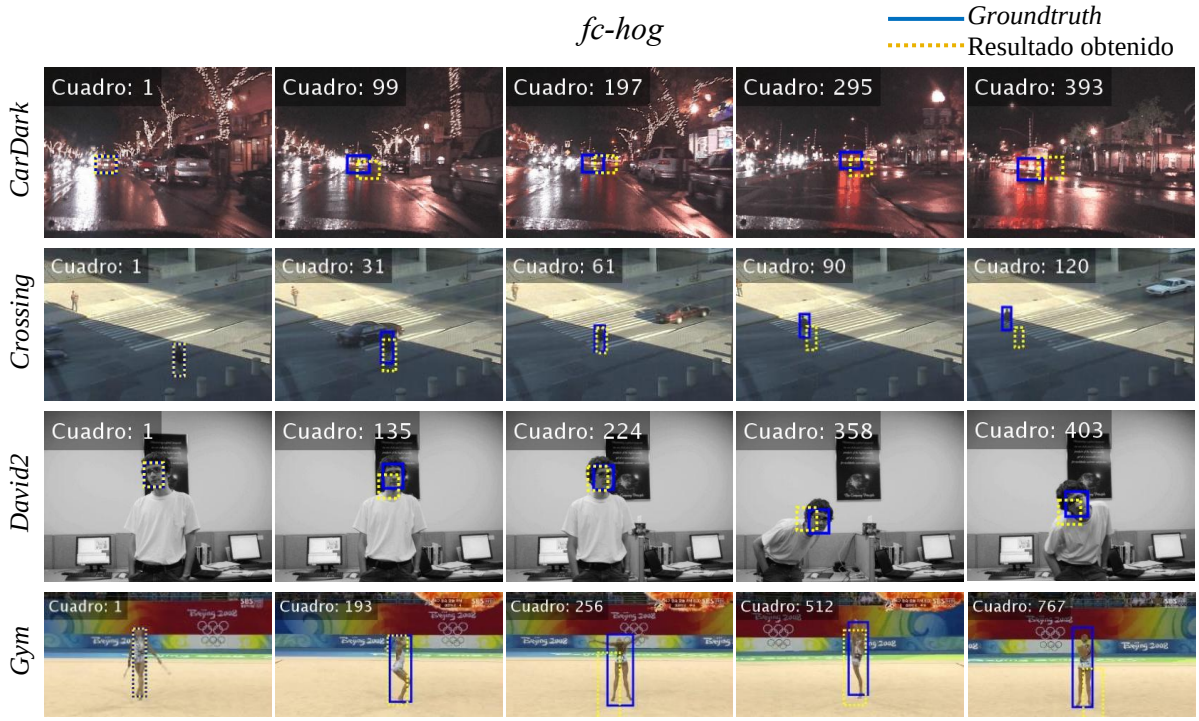


Figura 4.3. Evaluación cualitativa del algoritmo *fc-hog*.

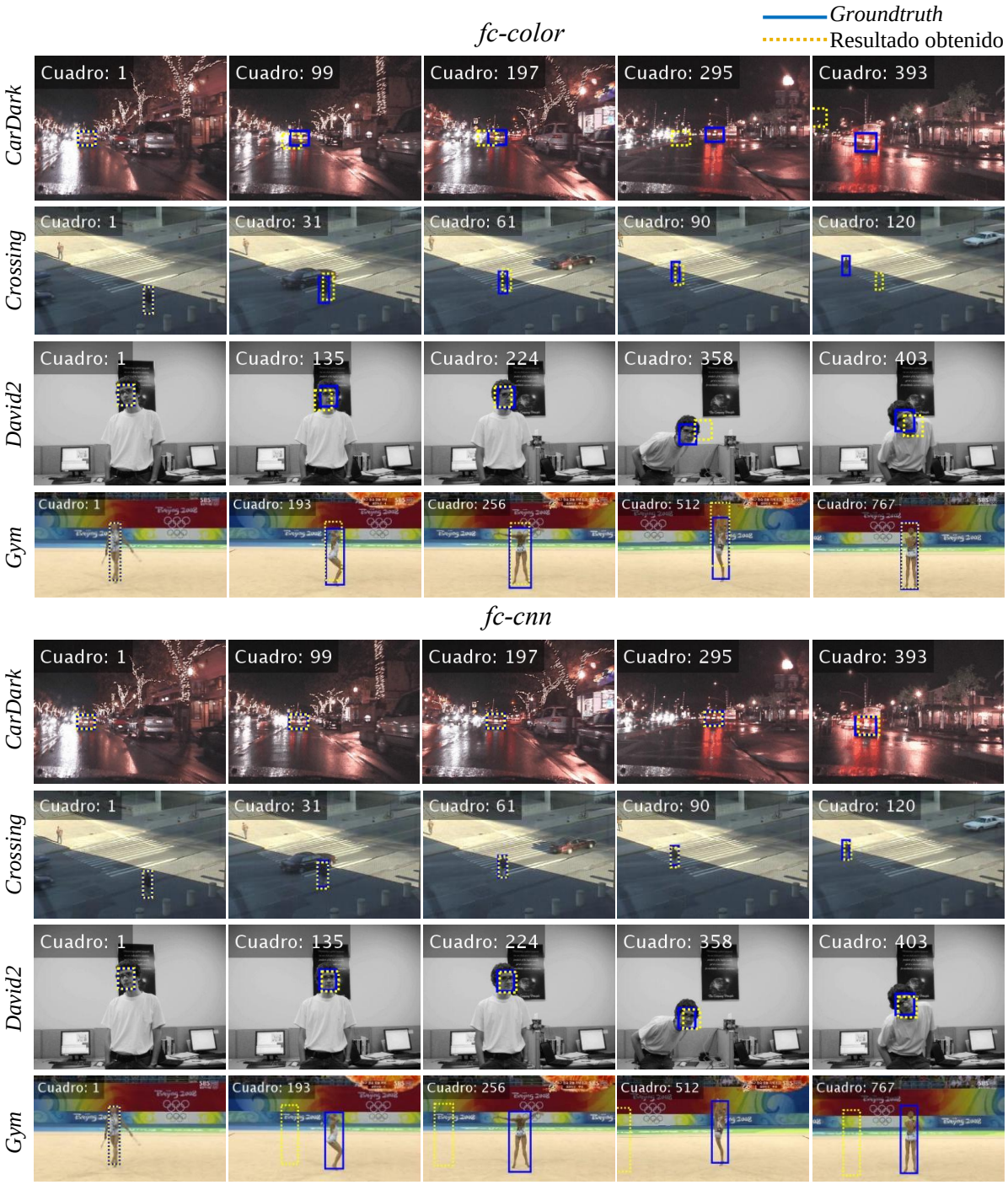


Figura 4.4. Evaluación cualitativa de los algoritmos *fc-color* y *fc-cnn*.

Debido a que el algoritmo *fc-cnn* obtiene los mejores resultados en las métricas de precisión y éxito, se decidió continuar la investigación sobre este con el propósito de mejorar el

desempeño del algoritmo sobre un mayor número de secuencias de video y por lo tanto un mayor número de desafíos. Es por ello que a partir de este se generan los algoritmos *fc-cnn_kf* y *fc-cnn_inc* que incorporan todos los elementos descritos en la sección 3.6. La diferencia entre estos es el módulo que incorporan para el seguimiento del OI cuando se detectan oclusiones o deformaciones. El algoritmo *fc-cnn_kf* implementa este módulo con un filtro de Kalman mientras que *fc-cnn_inc* utiliza en su lugar un incremento del tamaño de la ROI.

Con la finalidad de dar a conocer la mejora de *fc-cnn_kf* y *fc-cnn_inc* respecto a *fc-cnn* se llevó a cabo una comparación de los resultados de desempeño de los 3 algoritmos utilizando las mismas 20 secuencias de video con las que se evaluó *fc-cnn*. La configuración de los parámetros para *fc-cnn* se muestra en la Tabla 4.1 y para *fc-cnn_kf* y *fc-cnn_inc* se muestra en la Tabla 4.3. Estos valores fueron elegidos mediante un análisis empírico, que consiste en un ajuste de los parámetros guiado por el resultado cualitativo obtenido en la práctica.

Tabla 4.3. Configuración de parámetros de los algoritmos *fc-cnn_kf* y *fc-cnn_inc*.

Parámetro	Valor
A_{cnn}	1
f_σ	0.027
λ	0.0001
η	0.005
mp	6 para cada capa
C_{cf}	50
τ_{inc}	2.5

En la Figura 4.5 se muestran las gráficas de precisión y éxito de los algoritmos *fc-cnn*, *fc-cnn_kf* y *fc-cnn_inc* para las 20 secuencias de video. El algoritmo *fc-cnn_inc* logra el mejor desempeño tanto en precisión como éxito obteniendo los valores promedio de 0.746 y 0.641, respectivamente. Seguido de este, se encuentra *fc-cnn* alcanzando un promedio de 0.7 para la métrica de precisión y 0.604 para la métrica de éxito. Por último, se encuentra *fc-cnn_kf* con valores promedio de 0.558 y 0.474 en precisión y éxito, respectivamente.

Este bajo desempeño en *fc-cnn_kf* se debe a lo siguiente: cuando se detecta una oclusión o deformación se activa el filtro de Kalman, quien comienza a estimar la posición del OI. Debido

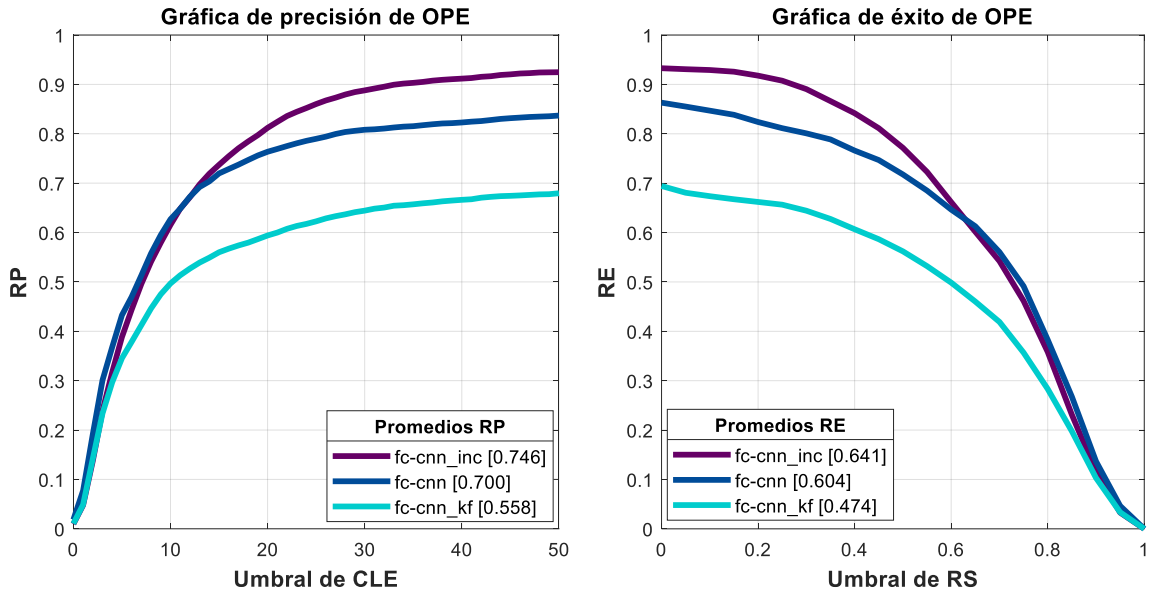


Figura 4.5. Gráficas de precisión y éxito de los algoritmos *fc-cnn*, *fc-cnn_kf* y *fc-cnn_inc* para 20 secuencias de video.

a la naturaleza lineal del sistema dinámico que se modela (sistema dinámico de velocidad constante) la estimación de la posición tendrá una tendencia lineal, haciendo que este resultado se vaya alejando gradualmente de la posición real del OI ya que el filtro de Kalman puede durar activado varios cuadros consecutivos. No obstante, el algoritmo *fc-cnn_inc* si obtiene un mejor desempeño al ser comparado con su versión original *fc-cnn*. La diferencia entre estos algoritmos es que *fc-cnn* no incorpora los métodos explicados en la sección 3.6 como la combinación lineal de los mapas de respuesta y el manejo de los eventos de oclusiones y deformaciones. Por lo tanto, se llega a la deducción que la integración de estos métodos es la principal causa del incremento en el desempeño para *fc-cnn_inc*. Para conocer los resultados individuales obtenidos en cada secuencia para cada algoritmo evaluado, consulte el Apéndice C-1.

Como el algoritmo *fc-cnn_inc* presenta el mejor desempeño, se decidió llevar a cabo un análisis sobre la configuración de sus parámetros, de tal manera que se defina una que brinde resultados superiores. Los parámetros que se analizaron fueron los siguientes:

- ✓ La cantidad mp_i de mapas de características que se seleccionan de cada capa i para la generación de los filtros de correlación.
- ✓ El factor f_σ de la desviación estándar.

Primero se realizó un experimento que consiste en variar manualmente estos parámetros, dejando constantes el resto para conocer el impacto que tienen de forma individual en los resultados cualitativos del seguimiento de objetos. De esta manera, se eligieron aquellos valores que producen mejores resultados, dando lugar a 4 configuraciones que fueron nombradas como *fc-cnn_inc1, fc-cnn_inc2, fc-cnn_inc3, fc-cnn_inc4* y cuyos parámetros se muestran en la Tabla 4.4.

Para conocer cuál de estas configuraciones da mejor desempeño se llevó a cabo una evaluación completa utilizando las 100 secuencias de video de la base de datos OTB [53]. Además, para dar un panorama general del desempeño también se evaluaron los algoritmos *fc-cnn_kf* y *fc-cnn_inc* con las mismas 100 secuencias de video. La configuración de parámetros de estos se conserva igual que la descrita en la Tabla 4.3.

Tabla 4.4. Configuración de parámetros de *fc-cnn_inc1, fc-cnn_inc2, fc-cnn_inc3* y *fc-cnn_inc4*.

Parámetro	**Valor**			
	fc-cnn_inc1	*fc-cnn_inc2*	*fc-cnn_inc3*	*fc-cnn_inc4*
A_{cnn}	1	1	1	1
f_σ	0.04	0.027	0.04	0.04
λ	0.0001	0.0001	0.0001	0.0001
η	0.005	0.005	0.005	0.005
mp	6 para cada capa	10 para cada capa	10 para cada capa	15 para cada capa
C_{cf}	50	50	50	50
τ_{inc}	2.5	2.5	2.5	2.5

En la Tabla 4.5 se muestra un resumen del desempeño obtenido de cada algoritmo luego de evaluarlos con las 100 secuencias de video. En la métrica de velocidad de procesamiento, los algoritmos alcanzan en promedio una velocidad de 14.288 FPS. Esta velocidad es superior a la alcanzada por *fc-cnn* que tiene un valor de 4.435 FPS (Tabla 4.2). Este aumento se debe a que el código de los algoritmos de la Tabla 4.5 fue optimizado al aprovechar la eficiencia que tiene el lenguaje Matlab® en las operaciones con matrices y al reemplazo del *toolbox* de Deep Learning de Matlab® por el *toolbox* MatConvNetv [74] para el manejo de la CNN, lo que permite que las operaciones implementadas por los filtros de correlación y la extracción de características sean llevadas a cabo de forma más rápida.

Tabla 4.5. Resumen de desempeño de los algoritmos evaluados para 100 secuencias de video.

Algoritmo / Métrica	*fc-cnn_kf*	*fc-cnn_inc*	*fc-cnn_inc1*	*fc-cnn_inc2*	*fc-cnn_inc3*	*fc-cnn_inc4*
Precisión	0.51	0.68	0.65	0.68	0.70	0.69
Éxito	0.45	0.60	0.59	0.61	0.62	0.62
Velocidad de procesamiento (FPS)	15.906	15.907	15.665	13.412	13.479	11.396

A continuación, se enlistarán las observaciones que se tuvieron de los resultados del experimento:

→ El algoritmo *fc-cnn_inc* se diferencia de *fc-cnn_inc1* en el valor configurado de f_σ, donde para el primero es de 0.027 y para el segundo es de 0.04. Este cambio originó que el desempeño en *fc-cnn_inc* sea de 0.68 para precisión y 0.60 para éxito, siendo estos valores mayores que los obtenidos para *fc-cnn_inc1*.

→ El algoritmo *fc-cnn_inc2* se diferencia de *fc-cnn_inc3* en el valor de f_σ que cambia de 0.027 a 0.04. Esta última configuración otorga mejores resultados llegando a 0.70 en precisión y 0.62 en éxito.

→ De las comparaciones anteriores se tiene la teoría de que se obtienen mejores resultados si se incrementa tanto f_σ como *mp*.

→ El algoritmo *fc-cnn_inc* es diferente a *fc-cnn_inc2* en el valor de *mp*, donde para el primero es de 6 y para el segundo es de 10. Esto origina que el desempeño de *fc-cnn_inc2* sea de 0.68 para precisión y 0.61 para éxito, siendo mejores valores que los obtenidos por *fc-cnn_inc*. Se cree que un aumento en *mp* da lugar a un mejor modelado de apariencia acerca del OI, lo que permite que los filtros de correlación realicen una mejor estimación de la posición. Sin embargo, el aumentar *mp* tiene un costo al disminuir el tiempo de procesamiento en 2.495 FPS.

→ El algoritmo *fc-cnn_inc1* es distinto a *fc-cnn_inc3* en el valor de *mp* que cambia de 6 a 10. Esto origina que el desempeño de *fc-cnn_inc3* sea mejor obteniendo 0.7 para precisión y 0.62 para éxito. El resultado de esta comparación apoya la suposición del punto anterior.

→ Ante una mejora aparente en el desempeño al incrementar *mp* se estableció la configuración de *fc-cnn_inc4* que se diferencia de *fc-cnn_inc3* al incrementar *mp* de

10 a 15 mapas de características por capa. El resultado de *fc-cnn_inc4* fue de 0.69 para precisión y 0.62 para éxito con un descenso de tiempo de procesamiento de 2.08 FPS. Esta comparación apoya la teoría del punto tres, que indica que se logra un mejor resultado si los parámetros f_σ y *mp* se incrementan proporcionalmente.

→ Como era de esperarse, el algoritmo *fc-cnn_kf* es el de menor desempeño, esto por las razones citadas anteriormente.

En la Figura 4.6 se muestran las gráficas de precisión y éxito de los algoritmos evaluados. En estas gráficas se visualiza que existe una cierta competencia por el primer lugar entre *fc-cnn_inc3 y fc-cnn_inc4.* Sin embargo, se decide que la versión *fc-cnn_inc3* es preferible ya que tiene mejor velocidad de procesamiento. Para conocer los resultados individuales de cada secuencia en cada algoritmo evaluado, consulte el Apéndice C-2.

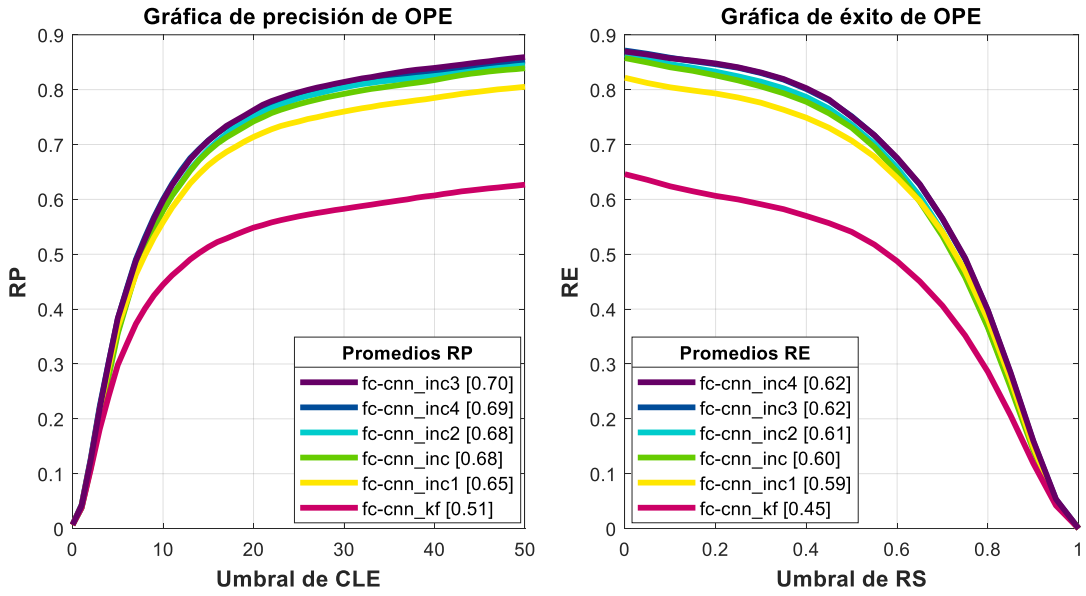


Figura 4.6. Gráficas de precisión y éxito de los algoritmos *fc-cnn_kf*, *fc-cnn_inc*, *fc-cnn_inc1*, *fc-cnn_inc2*, *fc-cnn_inc3* y *fc-cnn_inc4*.

En la Figura 4.7 y Figura 4.8 se muestra la evaluación cualitativa de los algoritmos *fc-cnn_kf*, *fc-cnn_inc* y *fc-cnn_inc3* utilizando las secuencias de video *BlurBody*, *Box*, *David* y *Dog*.

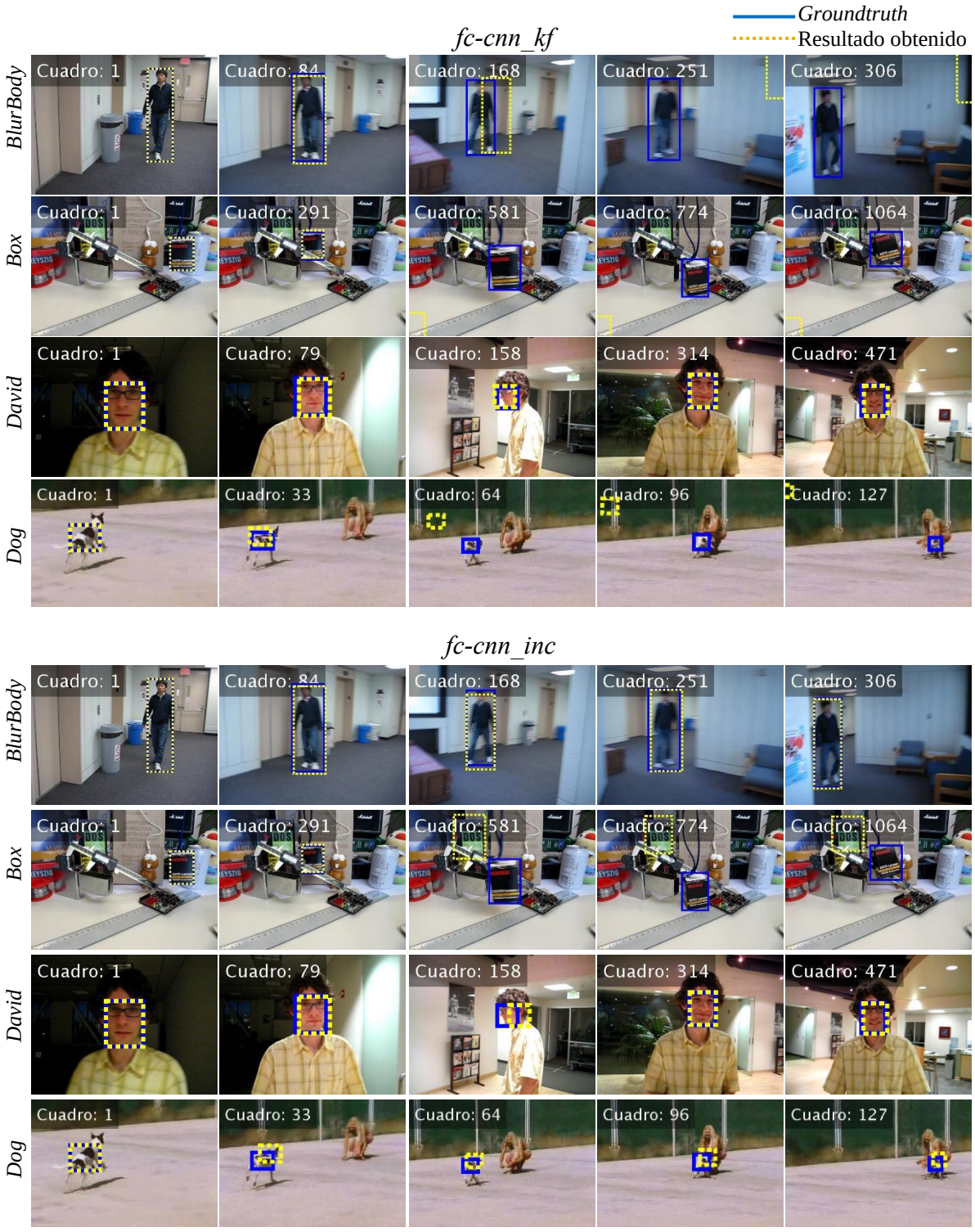


Figura 4.7. Evaluación cualitativa de los algoritmos *fc-cnn_kf* y *fc-cnn_inc*.

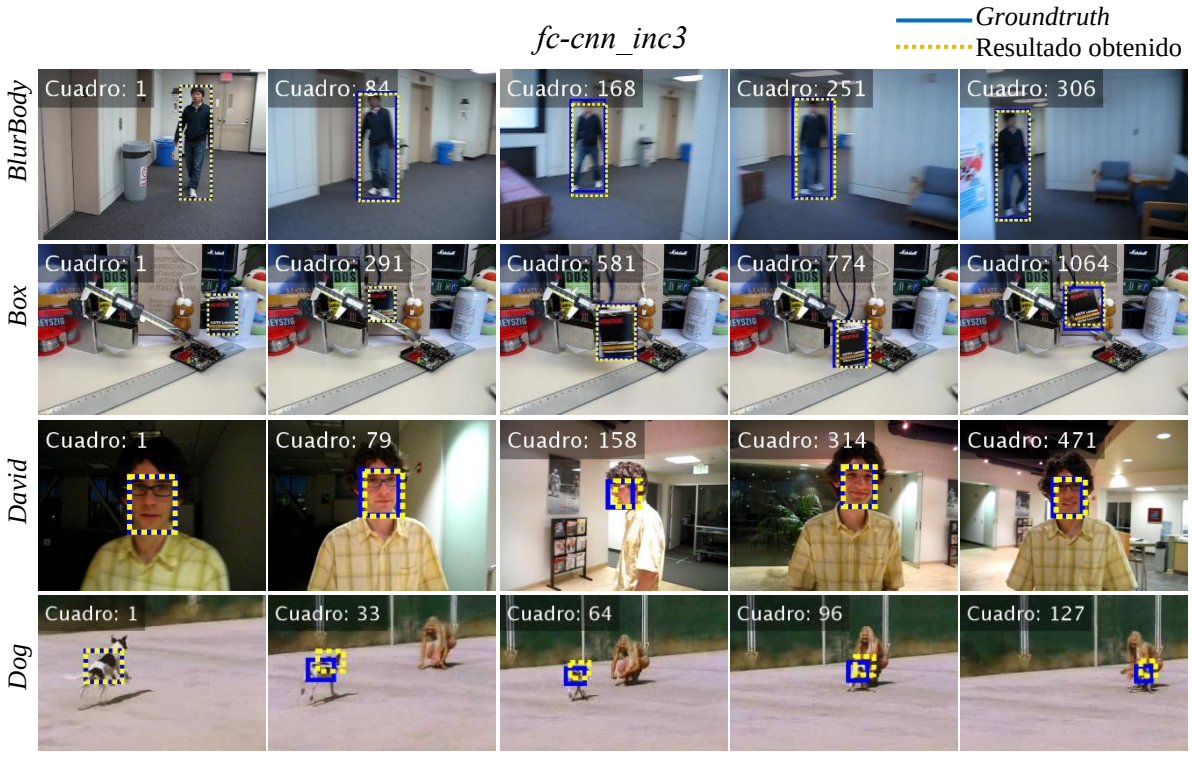


Figura 4.8. Evaluación cualitativa del algoritmo *fc-cnn_inc3*.

Para analizar las fortalezas y debilidades del algoritmo *fc-cnn_inc3* se obtuvieron los resultados en los 11 atributos que contienen las secuencias de video de OTB [53], donde cada atributo representa un desafío para el área de seguimiento de objetos.

En la Tabla 4.6 se muestran los resultados de precisión y éxito de cada desafío. El mejor resultado en precisión es el desafío de baja resolución LR con 0.77 mientras que el más bajo es el de fondo abarrotado BC con 0.64. Los desafíos con mejor resultado en éxito son movimiento rápido FM y borroso MB con 0.63 mientras que el más bajo es el desafío de fondo abarrotado BC con 0.56.

Tabla 4.6. Resultados de *fc-cnn_inc3* en cada desafío.

	Desafío	**IV**	**OPR**	**SV**	**OCC**	**DEF**	**MB**	**FM**	**IPR**	**OV**	**BC**	**LR**
fc-cnn_inc3	**Cantidad de secuencias con el desafío**	38	63	64	49	44	29	39	51	14	31	9
	Precisión	0.65	0.69	0.67	0.68	0.66	0.66	0.67	0.66	0.67	0.64	0.77
	Éxito	0.59	0.61	0.61	0.60	0.59	0.63	0.63	0.58	0.60	0.56	0.58

Para los desafíos de oclusión OCC y deformación DEF se tiene una precisión de 0.68 y 0.66, y un éxito de 0.60 y 0.59, respectivamente. Esto demuestra que la detección de oclusiones y deformaciones se da de forma correcta y que, aunque el módulo para el manejo de estos desafíos sigue una metodología simple, brinda resultados aceptables, como se puede visualizar en la Figura 4.9, donde se muestran las secuencias *Jogging-2*, *Woman* y *David3*.

De forma general, estos resultados indican que el manejo de los desafíos es consistente ya que los resultados de precisión y éxito son cercanos entre sí.

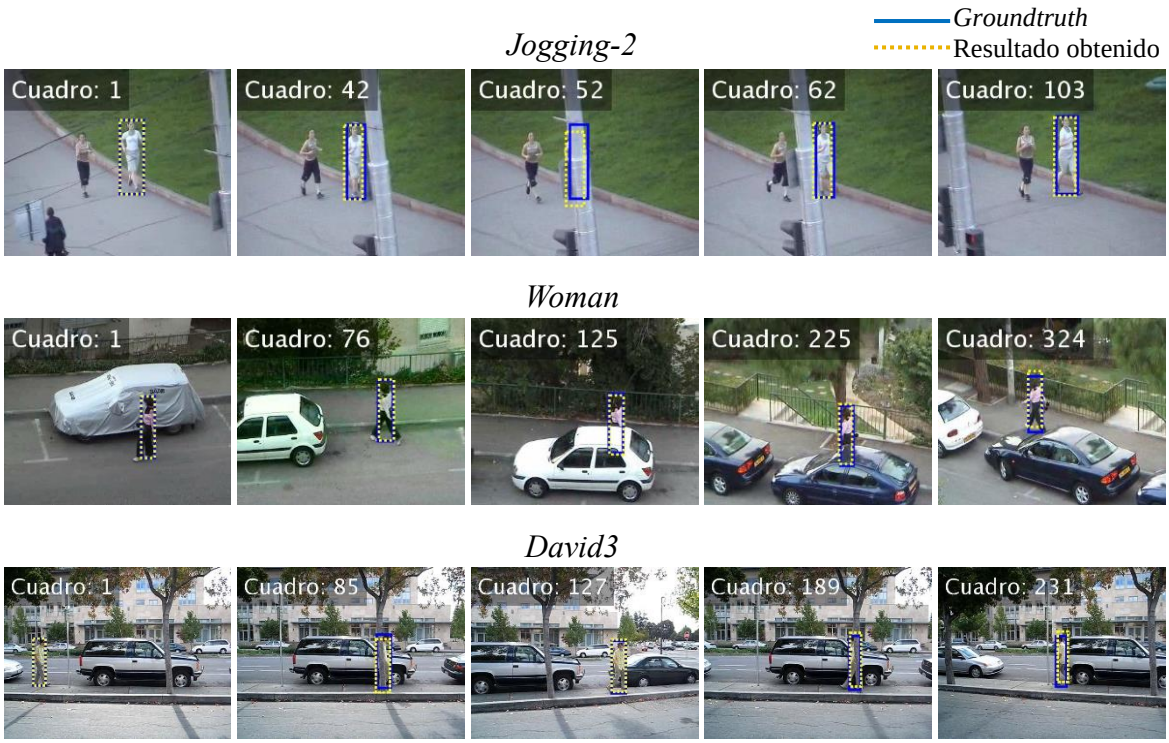


Figura 4.9. Evaluación cualitativa de los atributos de deformación y oclusión.

Los resultados de *fc-cnn_inc3* han sido los mejores de entre los demás algoritmos desarrollados. Esto hace que *fc-cnn_inc3* se convierta en el algoritmo FCCNN de esta investigación. En la sección siguiente se llevará a cabo una comparación de FCCNN con algunos algoritmos de la literatura.

4.3 Comparación de FCCNN con la literatura

En esta sección se presenta la comparación de FCCNN con algunos algoritmos de la literatura reciente que utilizan de igual manera la base de datos OTB [53] y que además realizan una evaluación OPE con las métricas de precisión y éxito. Los resultados mostrados a continuación fueron tomados directamente de la fuente de información escrita de los algoritmos. En la Figura 4.10 se ilustran los resultados de precisión y éxito de los algoritmos de la literatura, así como también el resultado de FCCNN.

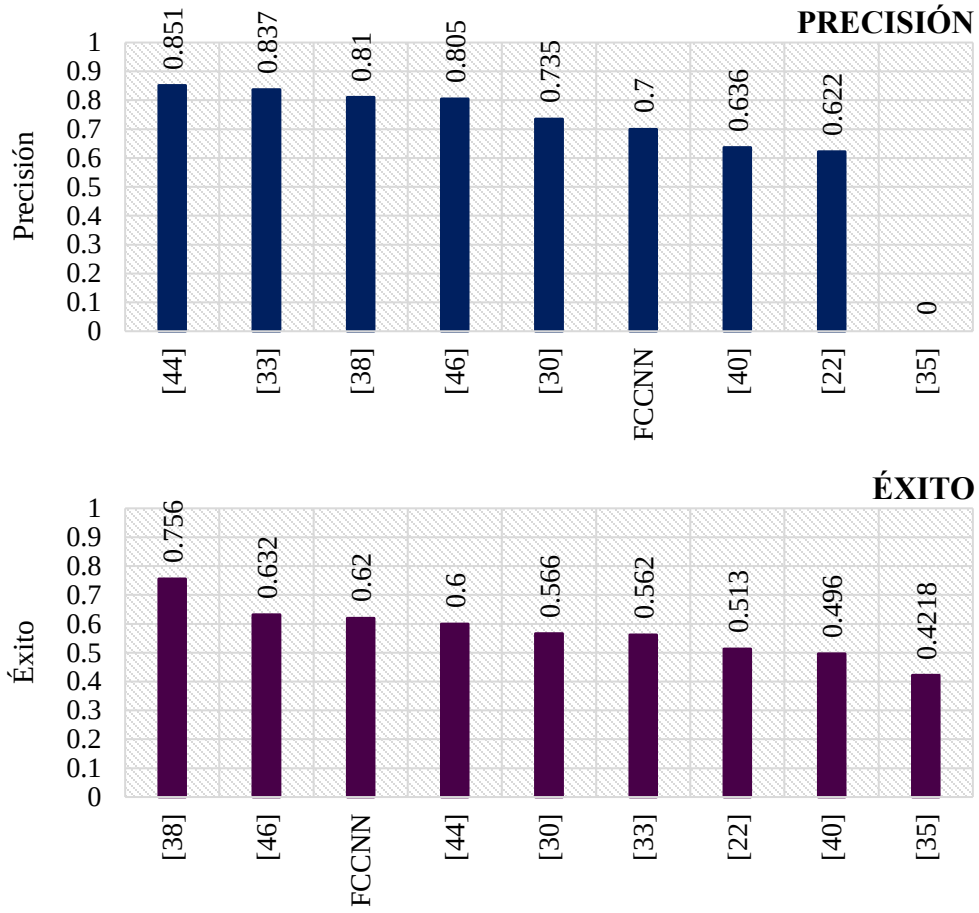


Figura 4.10. Comparación del algoritmo FCCNN con la literatura.

De acuerdo con estos resultados, FCCNN se posiciona en el sexto lugar en precisión y tercer lugar en éxito. Sin embargo, se tienen las siguientes observaciones que ponen en ventaja a FCCNN:

1. El algoritmo presentado en [33] utiliza la combinación de filtros de correlación con características extraídas de la red neuronal convolucional VGG-19. Este enfoque es similar al presentado para FCCNN, sin embargo, en [33] se utilizan un total de 1280 mapas de características, que es una cantidad mayor si lo comparamos con FCCNN que solo utiliza 130 mapas de características que codifican información acerca del OI. De lo anterior se deduce que el algoritmo FCCNN requiere menos datos para construir el modelo de apariencia del OI, y por lo tanto los filtros de correlación realizan menor cantidad de operaciones.
2. El algoritmo presentado en [38] realiza el entrenamiento de un clasificador discriminativo durante el primer cuadro de la secuencia de video, y lo actualiza cuando los resultados del filtro de correlación no son adecuados. Esta actualización consiste en la maximización de una función de probabilidad. De forma similar, FCCNN inicializa el filtro de correlación en el primer cuadro de video y lo actualiza durante el procesamiento. Sin embargo, las operaciones de actualización en FCCNN son más simples que las requeridas por [38], ya que se trata de una suma de productos.

En la Tabla 4.7 se muestra la velocidad de procesamiento de cada algoritmo de la literatura comparada con FCCNN, así como el equipo y el lenguaje utilizado.

Tabla 4.7. Velocidad de procesamiento, equipo y lenguaje de programación utilizado para FCCNN y los algoritmos de la literatura.

Ref.	Velocidad de procesamiento (FPS)	Equipo	Lenguaje de programación
[35]	149.63	Intel Xeon CPU E5-2696 v4 @2.20 GHz, Nvidia Titan X (Pascal)	-
FCCNN	13.479	Ryzen 5 2600x 6 núcleos @4.2 GHz CPU, 16GB de RAM @3200 MHz, GPU GTX 1070 Ti 8 GB GDDR5 y SSD.	Matlab
[33]	10.7	Intel I7-4770, 3.40 GHz CPU, 32 GB de RAM, GeForce GTX Titan GPU.	Matlab
[40]	10	Intel i7 @2 GHz CPU.	C++
[38]	7	Intel Xeon @3.10 GHz CPU, 8 GB de RAM.	Matlab
[44]	6	Intel Core i7-4790 CPU @3.60 GHz, 16 GB de RAM.	Matlab
[30]	5	Intel i5 CPU @3.3 GHz, 8 GB de RAM	Python
[22]	-	-	-
[46]	-	Intel Core i7-4710HQ CPU @ 2.5 GHz.	Matlab

Los resultados muestran que el algoritmo de [35] obtiene una velocidad excelente utilizando un procesador de menor velocidad y una GPU de capacidad similar en comparación con el algoritmo FCCNN, no obstante los resultados de [35] en la métrica de éxito lo posicionan en el último lugar, lo que representa una desventaja para este algoritmo.

Adicionalmente, se comparó FCCNN con los resultados de los algoritmos que se sitúan en los 10 primeros lugares de los posicionamientos realizados en el 2013 por [75] y en el 2015 por [53]. Ambos posicionamientos utilizan la base de datos OTB ya sea en su totalidad o una porción de esta.

En [75] se utilizan 50 secuencias de video y los resultados presentados son el éxito y la precisión a un umbral de 20 pixeles, por lo que se calculó esta última para FCCNN obteniendo un valor de 0.762. En la Figura 4.11 se muestra la comparación entre FCCNN y los 10 primeros lugares de [75], donde para ambas métricas el algoritmo FCCNN logra situarse en primer lugar.

Adicionalmente, en la Tabla 4.8 se muestran los resultados de la velocidad de procesamiento de los 10 algoritmos en [75] ejecutados en un equipo con Intel i7 3770 CPU @3.4 GHz. FCCNN obtiene el sexto lugar, no obstante, aunque los algoritmos como CSK tienen una velocidad de procesamiento sobresaliente, el desempeño en cuanto a precisión y éxito son menores que los obtenidos por FCCNN.

Tabla 4.8. Velocidad de procesamiento de FCCNN y los algoritmos de los posicionamientos realizados en el 2013 [75] y 2015 [53].

Algoritmo	CSK	TLD	OAB	Struck	CXT	FCCNN	DFT	ASLA	VTD	VTS	LSK	LOT	SCM
Velocidad de procesamiento (FPS)	362	28.1	22.4	20.2	15.3	13.479	13.2	8.5	5.7	5.7	5.5	0.7	0.51

Por último, en [53] se utilizaron 50 y 100 secuencias de video de la base de datos OTB. Sin embargo, solo se dan a conocer los resultados en éxito, por lo que la comparación solo se realiza sobre esta métrica.

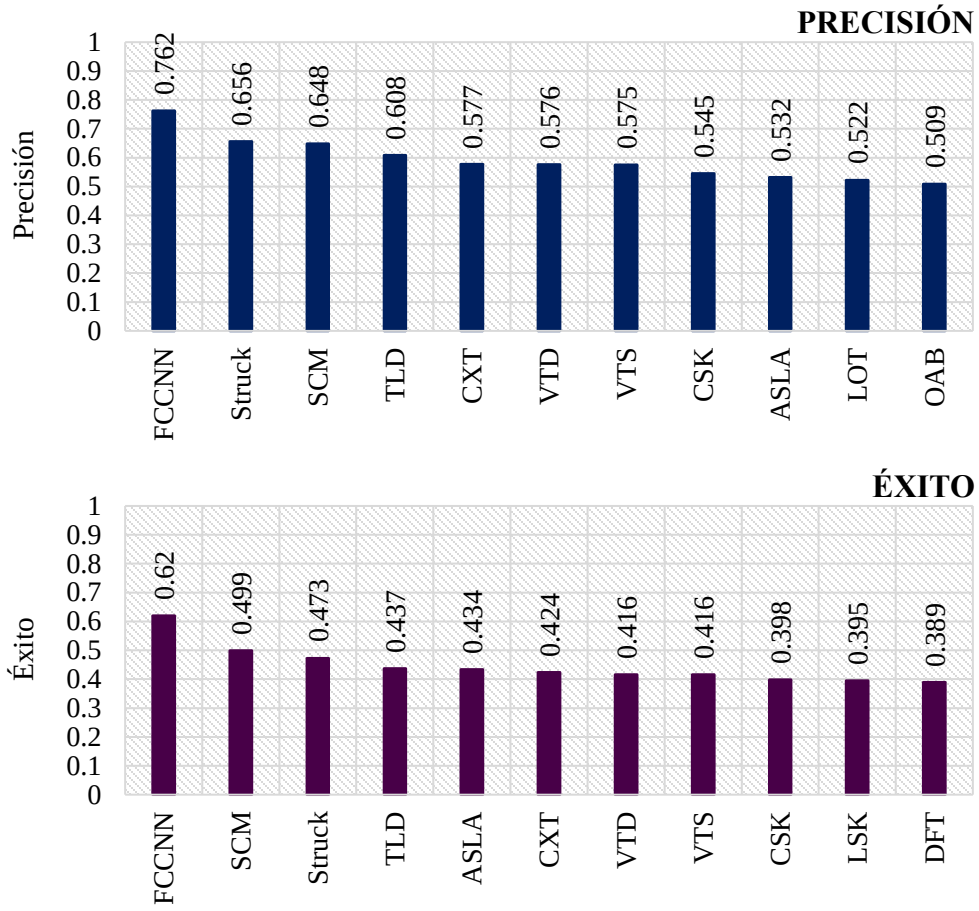


Figura 4.11. Comparación del algoritmo FCCNN con el posicionamiento realizado en el 2013 por [75].

En [53] intervienen los mismos algoritmos que los presentados para [75]. Los resultados en la velocidad de procesamiento son los mismos que los mostrados en la Tabla 4.8. Sin embargo, aquellos en cuanto al éxito no concuerdan con los obtenidos en [75], aun cuando se trate de los mismos algoritmos. Esto puede deberse a que es complicado realizar una reproducción exacta de los resultados de estos en diferentes tiempos, equipos o incluso entre investigadores [76].

En la Figura 4.12 y Figura 4.13 se muestra el éxito de los algoritmos en las 50 y 100 secuencias de video. Estos muestran que el algoritmo FCCNN logra posicionarse nuevamente en el primer lugar.

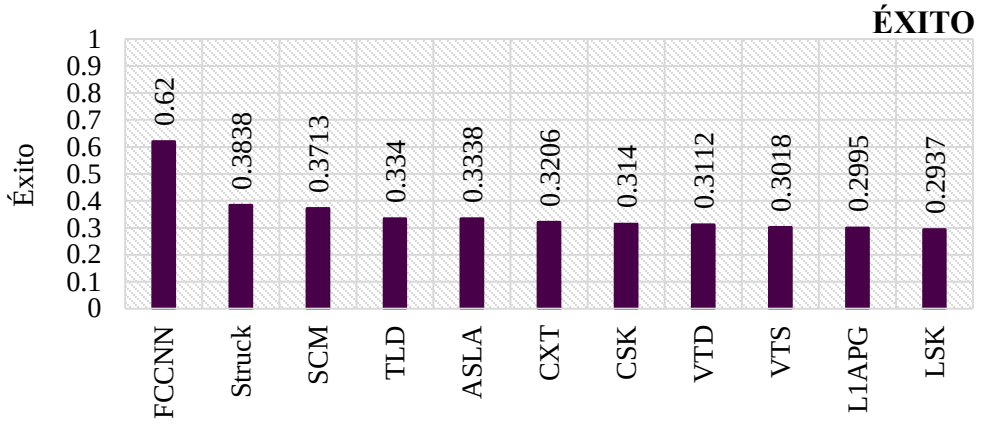


Figura 4.12. Comparación del algoritmo FCCNN con el posicionamiento realizado en el 2015 por [53] para 50 secuencias de video.

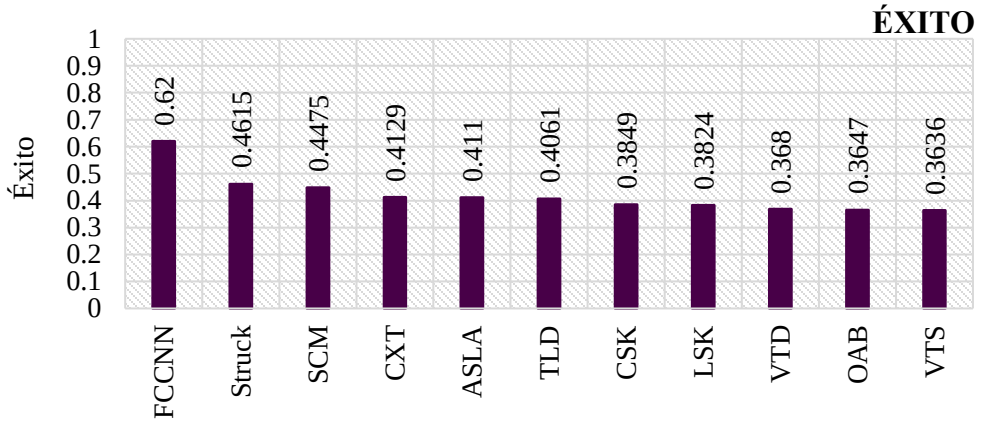


Figura 4.13. Comparación del algoritmo FCCNN con el posicionamiento realizado en el 2015 por [53] para 100 secuencias de video.

4.4 Conclusiones

En esta investigación se presentó una metodología para el diseño de algoritmos de seguimiento de objetos que inicia con la determinación del enfoque principal y continua con el desarrollo de varios algoritmos que conducen a la versión final nombrada como FCCNN.

Para establecer el enfoque principal, se diseñó un instrumento de evaluación de métodos aplicados al seguimiento de objetos que determina cuáles presentan las mejores características y cuáles son los enfoques que dan mejores resultados en el área. Luego de la evaluación de 29

métodos y una comparación con otros posicionamientos en la literatura, como el de VOT2017 *challenge*, se establece a los filtros de correlación como el enfoque que produce mejores resultados.

El instrumento de evaluación resulta ser una herramienta confiable, que produce resultados que pueden ser utilizados en la toma de decisiones, como puede ser el enfoque principal de un algoritmo de seguimiento de objetos. Además, es una alternativa para evaluar los métodos cuando no se posee el código fuente de estos, permitiendo verificar la congruencia de sus resultados y la validez de los mismos.

Posteriormente, se desarrollaron varios algoritmos de seguimiento de objetos basados en filtros de correlación y se evaluaron por etapas utilizando la base de datos OTB, con la finalidad de encontrar aquel que obtiene mejores resultados en las métricas de precisión, éxito y velocidad de procesamiento para convertirlo en el algoritmo FCCNN de esta investigación. Luego de realizar las evaluaciones correspondientes, se concluye que *fc-cnn_inc3* obtiene el mejor desempeño lo que lo hace el algoritmo FCCNN.

FCCNN implementa los filtros de correlación en dos dimensiones en conjunto con las características extraídas de la red neuronal convolucional preentrenada VGG-16 e incorpora la combinación lineal de los mapas de respuesta y la detección de oclusiones y deformaciones, entre otros métodos. Al ser evaluado con 100 secuencias de video, FCCNN logra una precisión de 0.70, un éxito de 0.62 y una velocidad de procesamiento de 13.479 FPS. Estos resultados demuestran que FCCNN es competente al ser comparado con varios algoritmos de la literatura reciente. Las evaluaciones cualitativas de la Figura 4.8 y Figura 4.9 ilustran que el algoritmo FCCNN es capaz de manejar situaciones desafiantes como deformaciones, oclusiones, borrosidad por movimiento, fondos abarrotados y movimientos rápidos.

Debido a su desempeño, el algoritmo FCCNN puede llegar a ser más apto para ciertas aplicaciones como interacción humano-computadora, robótica o análisis de comportamiento y movimiento.

De forma general, la metodología de diseño presentada en esta investigación resulta ser efectiva, permitiendo generar algoritmos competentes con la literatura reciente y con la posibilidad de ser utilizados dentro de aplicaciones más complejas.

REFERENCIAS

[1] L. E. Sucar y G. Gómez, "Vision Computacional", *Instituto Nacional de Astrofísica, Óptica y Electrónica*, 2011. [En línea]. Disponible en: http://ccc.inaoep.mx/~esucar/Libros/vision-sucar-gomez.pdf. [Consultado: 13-ago-2019].

[2] K. A. Joshi y D. G. Thakore, "A Survey on Moving Object Detection and Tracking in Video Surveillance System", *Int. J. Soft Comput. Eng.*, vol. 2, núm. 3, pp. 44–48, 2012.

[3] S. R. Balaji y S. Karthikeyan, "A survey on moving object tracking using image processing", en *2017 11th International Conference on Intelligent Systems and Control (ISCO)*, 2017, pp. 469–474.

[4] A. Yilmaz, O. Javed, y M. Shah, "Object tracking: A Survey", *ACM Comput. Surv.*, vol. 38, núm. 4, pp. 1–45, 2006.

[5] S. A. Daneshyar y M. Nahvi, "Moving objects tracking based on improved particle filter algorithm by elimination of unimportant particles", *Optik (Stuttg).*, vol. 138, pp. 455–469, 2017.

[6] H. Liu y F. Sun, "Efficient visual tracking using particle filter with incremental likelihood calculation", *Inf. Sci. (Ny).*, vol. 195, pp. 141–153, 2012.

[7] R. Fan, F.-L. Zhang, M. Zhang, y R. R. Martin, "Robust tracking-by-detection using a selection and completion mechanism", *Comput. Vis. Media*, vol. 3, núm. 3, pp. 285–294, 2017.

[8] F. Alamdar y M. Keyvanpour, "A New Color Feature Extraction Method Based on QuadHistogram", *Procedia Environ. Sci.*, vol. 10, Part A, pp. 777–783, 2011.

[9] M. C. Sanchez-Cuevas, R. M. Aguilar-Ponce, y J. L. Tecpanecatl-Xihuitl, "A Comparison of Color Models for Color Face Segmentation", *Procedia Technol.*, vol. 7, pp. 134–141, 2013.

[10] K. Cannons, "A Review of Visual Tracking", *Dep. Comput. Sci. Eng. Cent. Vis. Res.*

Tech. Rep. CSE-2008-07.

[11] R. S. Choras, “Image Feature Extraction Techniques and Their Applications for CBIR and Biometrics Systems”, *Int. J. Biol. Biomed. Eng.*, vol. 1, núm. 1, pp. 6–16, 2007.

[12] P. Li, D. Wang, L. Wang, y H. Lu, “Deep visual tracking: Review and experimental comparison”, *Pattern Recognit.*, vol. 76, pp. 323–338, 2018.

[13] N. Wang, J. Shi, D.-Y. Yeung, y J. Jia, “Understanding and Diagnosing Visual Tracking Systems”, en *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3101–3109.

[14] L. Nanni, S. Ghidoni, y S. Brahnam, “Handcrafted vs. non-handcrafted features for computer vision classification”, *Pattern Recognit.*, vol. 71, pp. 158–172, 2017.

[15] P. Mirunalini, S. M. Jaisakthi, y R. Sujana, “Tracking of Object in Occluded and Non-occluded Environment using SIFT and Kalman Filter”, en *TENCON 2017 - 2017 IEEE Region 10 Conference*, 2017, pp. 1290–1295.

[16] J. F. Chacón Hinojos, “Estimación de trayectorias de objetos dinámicos para sistemas inteligentes de video”, M.C. Tesis, Instituto Tecnológico de Chihuahua, CUU, MX, 2011.

[17] A. Ali *et al.*, “Visual object tracking — classical and contemporary approaches”, *Front. Comput. Sci.*, vol. 10, núm. 1, pp. 167–188, 2016.

[18] B. Y. Lee, L. H. Liew, W. S. Cheah, y Y. C. Wang, “Occlusion handling in videos object tracking: A survey”, *IOP Conf. Ser. Earth Environ. Sci.*, vol. 18, pp. 1–6, 2014.

[19] J. J. Athanesious y P. Suresh, “Systematic Survey on Object Tracking Methods in Video”, *Int. J. Adv. Res. Comput. Eng. Technol.*, vol. 1, núm. 8, pp. 242–247, 2012.

[20] J. A. J. y P. Suresh, “Implementation and Comparison of Kernel and Silhouette Based Object Tracking”, *Int. J. Adv. Res. Comput. Eng. Technol.*, vol. 2, núm. 3, pp. 1298–1303, 2013.

[21] J. Sun, F. He, Y. Chen, y X. Chen, “A multiple template approach for robust tracking of fast motion target”, *Appl. Math. J. Chinese Univ.*, vol. 31, núm. 2, pp. 177–197, 2016.

[22] J. Li, X. Zhou, S. Chan, y S. Chen, “Object tracking using a convolutional network and a structured output SVM”, *Comput. Vis. Media*, vol. 3, núm. 4, pp. 325–335, 2017.

[23] Z. Liu, H. Shen, G. Feng, y D. Hu, “Tracking objects using shape context matching”, *Neurocomputing*, vol. 83, pp. 47–55, 2012.

[24] S. Kanagamalliga y S. Vasuki, “Contour-based object tracking in video scenes through optical flow and gabor features”, *Optik (Stuttg).*, vol. 157, pp. 787–797, 2018.

[25] M. M. Naushad Ali, M. Abdullah-Al-Wadud, y S.-L. Lee, “Multiple object tracking with partial occlusion handling using salient feature points”, *Inf. Sci. (Ny).*, vol. 278, pp. 448–465, 2014.

[26] G. Choe, T. Wang, F. Liu, C. Choe, y H. So, “Visual tracking based on particle filter with spline resampling”, *Multimed. Tools Appl.*, vol. 74, núm. 17, pp. 7195–7220, 2015.

[27] M. D. Jenkins, P. Barrie, T. Buggy, y G. Morison, “Selective Sampling Importance Resampling Particle Filter Tracking With Multibag Subspace Restoration”, *IEEE Trans. Cybern.*, vol. 48, núm. 1, pp. 264–276, 2018.

[28] I. Elafi, M. Jedra, y N. Zahid, “Unsupervised detection and tracking of moving objects for video surveillance applications”, *Pattern Recognit. Lett.*, vol. 84, pp. 70–77, 2016.

[29] Z. Zhou, M. Zhou, y J. Li, “Object tracking method based on hybrid particle filter and sparse representation”, *Multimed. Tools Appl.*, vol. 76, núm. 2, pp. 2979–2993, 2017.

[30] K. Chen, W. Tao, y S. Han, “Visual object tracking via enhanced structural correlation filter”, *Inf. Sci. (Ny).*, vol. 394–395, pp. 232–245, 2017.

[31] J. Fourie, S. Mills, y R. Green, “Harmony filter: A robust visual tracking system using the improved harmony search algorithm”, *Image Vis. Comput.*, vol. 28, núm. 12, pp. 1702–1716, 2010.

[32] W.-C. Hu, C.-H. Chen, T.-Y. Chen, D.-Y. Huang, y Z.-C. Wu, "Moving object detection and tracking from video captured by moving camera", *J. Vis. Commun. Image Represent.*, vol. 30, pp. 164–180, 2015.

[33] C. Ma, J.-B. Huang, X. Yang, y M.-H. Yang, "Hierarchical convolutional features for visual tracking", en *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 3074–3082.

[34] P. Zhang, T. Zhuo, W. Huang, K. Chen, y M. Kankanhalli, "Online object tracking based on CNN with spatial-temporal saliency guided sampling", *Neurocomputing*, vol. 257, pp. 115–127, 2017.

[35] D. Gordon, A. Farhadi, y D. Fox, "Re3 : Real-Time Recurrent Regression Networks for Visual Tracking of Generic Objects", *IEEE Robot. Autom. Lett.*, vol. 3, núm. 2, pp. 788–795, 2018.

[36] J. Wang, H. Zhu, S. Yu, y C. Fan, "Object tracking using color-feature guided network generalization and tailored feature fusion", *Neurocomputing*, vol. 238, pp. 387–398, 2017.

[37] X. Yun y G. Xiao, "Spiral visual and motional tracking", *Neurocomputing*, vol. 249, pp. 117–127, 2017.

[38] X. Zhang, G.-S. Xia, Q. Lu, W. Shen, y L. Zhang, "Visual object tracking by correlation filters and online learning", *ISPRS J. Photogramm. Remote Sens.*, vol. 140, pp. 77–89, 2018.

[39] M. Kristan *et al.*, "The Visual Object Tracking VOT2017 Challenge Results", en *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017, pp. 1949–1972.

[40] K. M. Yi, H. Jeong, B. Lee, y J. Y. Choi, "Visual tracking in complex scenes through pixel-wise tri-modeling", *Mach. Vis. Appl.*, vol. 26, núm. 2–3, pp. 205–217, 2015.

[41] S. Xiaoyan y C. Faliang, "Moving object tracking with feature learning and inheriting",

en *2017 Chinese Automation Congress (CAC)*, 2017, pp. 899–903.

[42] W. Li y D. Powers, "Multiple Object Tracking Using Motion Vectors from Compressed Video", en *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2017, pp. 1–5.

[43] D. Oiwa, S. Fukui, Y. Iwahori, B. Kijsirikul, T. Nakamura, y M. K. Bhuyan, "Tracking with Extraction of Moving Object under Moving Camera Environment", *Procedia Comput. Sci.*, vol. 112, pp. 1479–1487, 2017.

[44] B. Liu, Z. Zhu, y Y. Yang, "Convolutional neural networks based scale-adaptive kernelized correlation filter for robust visual object tracking", en *2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, 2017, pp. 423–428.

[45] I. A. Iswanto y B. Li, "Visual Object Tracking Based on Mean-shift and Particle-Kalman Filter", *Procedia Comput. Sci.*, vol. 116, pp. 587–595, 2017.

[46] S. Pang, J. J. del Coz, Z. Yu, O. Luaces, y J. Díez, "Deep learning to frame objects for visual target tracking", *Eng. Appl. Artif. Intell.*, vol. 65, pp. 406–420, 2017.

[47] H. Yu, L. Qin, Q. Huang, y H. Yao, "Online multiple object tracking via exchanging object context", *Neurocomputing*, vol. 292, pp. 28–37, 2018.

[48] E. Gundogdu y A. A. Alatan, "Good Features to Correlate for Visual Tracking", *IEEE Trans. Image Process.*, vol. 27, núm. 5, pp. 2526–2540, 2018.

[49] M. Danelljan, G. Bhat, F. Shahbaz Khan, y M. Felsberg, "ECO: Efficient convolution operators for tracking", en *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6931–6939.

[50] M. Danelljan, A. Robinson, F. Shahbaz Khan, y M. Felsberg, "Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking", *ECCV*, pp. 472–488, 2016.

[51] A. Lukežič, T. Vojíř, L. Čehovin Zajc, J. Matas, y M. Kristan, "Discriminative Correlation Filter with Channel and Spatial Reliability", en *2017 IEEE Conference on*

Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4847–4856.

[52] T. Zhang, C. Xu, y M.-H. Yang, "Multi-task correlation particle filter for robust object tracking", en *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4819–4827.

[53] Y. Wu, J. Lim, y M.-H. Yang, "Object Tracking Benchmark", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, núm. 9, pp. 1834–1848, 2015.

[54] N. Dalal y B. Triggs, "Histograms of Oriented Gradients for Human Detection", en *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005, vol. 1, pp. 886–893.

[55] S. Zhang y X. Wang, "Human Detection and Object Tracking Based on Histograms of Oriented Gradients", en *2013 Ninth International Conference on Natural Computation (ICNC)*, 2013, pp. 1349–1353.

[56] MathWorks, "rgb2gray", 2019. [En línea]. Disponible en: https://la.mathworks.com/help/matlab/ref/rgb2gray.html. [Consultado: 26-sep-2019].

[57] MathWorks, "histeq", 2019. [En línea]. Disponible en: https://la.mathworks.com/help/images/ref/histeq.html. [Consultado: 26-sep-2019].

[58] M. Ju, C. Ouyang, y H. Chang, "Mean shift tracking using fuzzy color histogram", en *2010 International Conference on Machine Learning and Cybernetics*, 2010, pp. 2904–2908.

[59] O. Küçüktunç, U. Güdükbay, y Ö. Ulusoy, "Fuzzy color histogram-based video segmentation", *Comput. Vis. Image Underst.*, vol. 114, núm. 1, pp. 125–134, 2010.

[60] MathWorks, "Redes Neuronales Convolucionales, Tres cosas que es necesario saber". [En línea]. Disponible en: https://la.mathworks.com/solutions/deep-learning/convolutional-neural-network.html. [Consultado: 19-mar-2019].

[61] P. Marcelino, "Transfer learning from pre-trained models", *Towards Data Science*, 2018. [En línea]. Disponible en: https://towardsdatascience.com/transfer-learning-from-

pre-trained-models-f2393f124751. [Consultado: 19-mar-2019].

[62] K. Simonyan y A. Zisserman, “Very deep convolutional networks for large-scale image recognition”, en *ICLR 2015*, pp. 1–14.

[63] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge”, *Int. J. Comput. Vis.*, vol. 115, núm. 3, pp. 211–252, 2015.

[64] M. Danelljan, H. Gustav, F. S. Khan, y M. Felsberg, “Convolutional Features for Correlation Filter Based Visual Tracking”, en *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, 2015, pp. 621–629.

[65] V. N. Boddeti, “Advances in Correlation Filters : Vector Features, Structured Prediction and Shape Alignment”, Ph. D. Tesis, Carnegie Mellon, Pittsburgh, PA, EUA, 2012.

[66] D. S. Bolme, J. R. Beveridge, B. A. Draper, y Y. M. Lui, “Visual Object Tracking using Adaptive Correlation Filters”, en *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2544–2550.

[67] M. Danelljan, G. Häger, F. S. Khan, y M. Felsberg, “Accurate Scale Estimation for Robust Visual Tracking”, en *British Machine Vision Conference (BMVC)*, 2014, pp. 1–11.

[68] M. I. Chacón Murguía, “Procesamiento digital de imágenes”, *Trillas*, 2007.

[69] MathWorks, “hann”, 2019. [En línea]. Disponible en: https://la.mathworks.com/help/signal/ref/hann.html. [Consultado: 08-sep-2019].

[70] Y. Kuai, G. Wen, y D. Li, “Hyper-Siamese network for robust visual tracking”, *Signal, Image Video Process.*, vol. 13, núm. 1, pp. 35–42, 2019.

[71] A. Lukežič, L. Č. Zajc, T. Vojíř, J. Matas, y M. Kristan, “FuCoLoT – A Fully-Correlational Long-Term Tracker”, en *Asian Conference on Computer Vision (ACCV)*, 2018, pp. 595–611.

[72] P. S. Maybeck, *Stochastic models , estimation and control*, vol. 1. 1979.

[73] G. Welch y G. Bishop, "An Introduction to the Kalman Filter", *Dep. Comput. Sci. Univ. North Carolina Chapel Hill*, pp. 1–16, 2006.

[74] The MatConvNet Team, "MatConvNet: CNNs for MATLAB", 2017. [En línea]. Disponible en: http://www.vlfeat.org/matconvnet/. [Consultado: 13-ago-2019].

[75] Y. Wu, J. Lim, y M. Yang, "Online Object Tracking : A Benchmark", en *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2411–2418.

[76] M. Baker, "1,500 scientists lift the lid on reproducibility", *Nature*, vol. 533, pp. 452–454, 2016.

APÉNDICES

A. RESULTADOS DE LA EVALUACIÓN DE LOS MÉTODOS APLICADOS AL SEGUIMIENTO DE OBJETOS

A-1. Puntuaciones para el criterio de *Do*, subcriterios Do_{Pr} y Do_C

Ref.	Do_{Pr}													Do_C
	Pr_v	V_{cc}	V	Pr_{cc}	V_r	V	Pr_r	Pr_{ec}	Pr_{tr}	Pr_{ee}	P_{max}	Pa	Pr_{cp}	
[5]	0	5	5	1	5	5	1	0	0	1	5	1	1	0.5
[6]	1	7	7	1	4	7	0.571	1	0	1	5	2	1	0.5
[7]	0	2	6	0.333	2	6	0.333	0	1	0	5	1	1	1
[15]	0	15	15	1	5	15	0.333	1	0	1	5	-	0	0.5
[21]	1	5	5	1	5	5	1	0	1	1	5	4	1	1
[22]	1	50	50	1	50	50	1	1	0	0	5	3	1	1
[24]	0	4	4	1	4	4	1	0	0	0	5	2	1	0.5
[25]	0	3	3	1	3	3	1	0	0	1	5	3	1	0.5
[26]	0	0	22	0	15	22	0.681	0	0	1	5	3	1	0.5
[27]	1	20	20	1	20	20	1	0	0	1	5	18	0.277	1
[28]	0	6	6	1	6	6	1	1	0	1	5	1	1	1
[29]	1	2	3	0.666	2	3	0.666	0	0	1	5	5	1	0.5
[30]	1	51	51	1	51	51	1	0	0	1	5	10	0.5	1
[31]	0	1	3	0.333	3	3	1	1	0	1	5	10	0.5	0.5
[32]	1	7	7	1	7	7	1	0	1	1	5	5	1	1
[33]	1	100	100	1	100	100	1	0	0	0	5	6	0.833	1
[34]	1	31	31	1	31	31	1	0	0	0	5	9	0.555	1
[35]	1	135	135	1	135	135	1	0	1	0	5	7	0.714	1
[36]	0	5	6	0.833	5	6	0.833	0	0	1	5	1	1	1
[37]	0	12	12	1	10	12	0.833	0	0	1	5	10	0.5	1
[38]	1	51	51	1	51	51	1	0	0	0	5	13	0.384	1
[40]	1	51	54	0.944	51	54	0.944	0	0	1	5	9	0.555	1
[41]	0	1	3	0.333	1	3	0.333	1	0	1	5	7	0.714	0.5
[42]	0	5	5	1	0	5	0	0	0	1	5	5	1	0
[43]	1	2	5	0.4	5	5	1	0	1	0	5	7	0.714	1
[44]	1	162	162	1	162	162	1	0	0	0	5	10	0.5	1
[45]	1	8	8	1	3	8	0.375	1	0	1	5	2	1	0.5
[46]	0	50	50	1	50	50	1	1	1	0	5	14	0.357	1
[47]	1	36	36	1	36	36	1	1	0	0	5	8	0.625	1
[48]	1	160	160	1	160	160	1	1	0	0	5	2	1	1
[49]	1	411	411	1	411	411	1	0	0	0	5	9	0.555	1
[50]	0	288	288	1	288	288	1	1	1	0	5	9	0.555	1
[51]	1	160	160	1	160	160	1	1	1	0	5	7	0.714	1
[52]	1	228	228	1	228	228	1	0	0	0	5	7	0.714	1

A-2. Puntuaciones para el criterio de *Do*, subcriterios Do_R y Do_{Ev}.

Ref.	Do_R								Do_{Ev}						
	pro	*ram*	*cg*	*sl*	R_{in}	P_{cfg}	*Pa*	R_{cfg}	*bdc*	*bdi*	*bd*	Ev_{bd}	Ev_{cn}	Ev_{cl}	Ev_{cfg}
[5]	0	0	0	0	0	1	1	1	0	3	3	0.5	1	0	0
[6]	0	0	0	0	0	2	2	1	0	3	3	0.5	1	1	1
[7]	1	0	1	0	2	0	1	0	0	1	1	0.5	1	1	1
[15]	0	0	0	0	0	-	-	0	0	1	1	0.5	1	0	0
[21]	1	1	0	1	3	4	4	1	0	1	1	0.5	1	1	0
[22]	0	0	0	0	0	0	3	0	1	0	1	1	1	1	1
[24]	0	0	0	0	0	0	2	0	0	2	2	0.5	1	0	0
[25]	0	0	0	0	0	3	3	1	0	3	3	0.5	1	1	0
[26]	0	0	0	0	0	3	3	1	0	1	1	0.5	1	1	0
[27]	0	0	0	0	0	18	18	1	0	1	1	0.5	1	1	1
[28]	0	0	0	0	0	1	1	1	0	2	2	0.5	1	1	1
[29]	1	1	0	1	3	5	5	1	0	1	1	0.5	1	1	0
[30]	1	1	1	1	4	10	10	1	1	0	1	1	1	1	0
[31]	0	0	0	0	0	10	10	1	0	1	1	0.5	1	1	1
[32]	1	1	1	1	4	4	5	0.8	0	0	0	0	1	1	0
[33]	1	1	1	1	4	6	6	1	1	0	1	1	1	1	1
[34]	1	1	0	1	3	9	9	1	0	2	2	0.5	1	1	0
[35]	1	0	1	0	2	7	7	1	4	0	4	1	1	1	0
[36]	0	0	0	0	0	1	1	1	0	1	1	0.5	1	1	1
[37]	1	1	1	1	4	10	10	1	0	1	1	0.5	1	1	0
[38]	1	1	1	1	4	13	13	1	1	0	1	1	1	1	1
[40]	1	0	1	1	3	9	9	1	1	1	2	0.75	1	1	0
[41]	0	0	0	0	0	7	7	1	0	1	1	0.5	0	1	0
[42]	0	0	0	1	1	4	5	0.8	0	0	0	0	0	0	0
[43]	1	0	0	0	1	7	7	1	0	1	1	0.5	1	1	1
[44]	1	1	1	1	4	10	10	1	2	0	2	1	1	1	1
[45]	1	1	1	1	4	2	2	1	0	1	1	0.5	1	1	0
[46]	1	1	1	1	4	14	14	1	1	0	1	1	1	1	0
[47]	0	0	0	0	0	8	8	1	2	0	2	1	1	1	0
[48]	0	0	0	0	0	0	2	0	3	0	3	1	1	0	0
[49]	0	0	1	1	2	9	9	1	4	0	4	1	1	1	1
[50]	0	0	1	1	2	9	9	1	3	0	3	1	1	0	0
[51]	1	0	1	1	3	7	7	1	3	0	3	1	1	1	1
[52]	1	1	1	1	4	7	7	1	3	0	3	1	1	1	1

A-3. Puntuaciones para el criterio de *De*, métricas De_{ve}, De_p y De_e

Ref.	*fps*	fps_{max}	De_{ve}	Precisión						Éxito				
				w	P_P	P_P'	P_U	P_A	De_p	*w*	E_P	E_U	E_A	De_e
[5]	0	20	0						0					0
[6]	0	20	0						0					0
[7]	0	20	0						0	A			0.873	1.873
[15]	0	20	0						0					0
[21]	24	20	1						0					0
[22]	4.1	20	0	A				0.622	1.622	A			0.513	1.513
[24]	0	20	0						0					0
[25]	0	20	0						0					0
[26]	0	20	0						0					0
[27]	9	20	0.45	P	23.72	0.209			0.909	P	0.564			1.264
[28]	0	20	0						0					0
[29]	5	20	0.25						0					0
[30]	5	20	0.25	U			0.735		1.635	A			0.566	1.566
[31]	0	20	0						0					0
[32]	24	20	1						0					0
[33]	10.7	20	0.535	U	19.25	0.358	0.864		1.764	A		0.697	0.562	1.562
[34]	5	20	0.25	P	17.45	0.418			1.118	P	0.66			1.36
[35]	149.6	20	1						0					0
[36]	0	20	0	P	4.096	0.863			1.563	P	0.711			1.411
[37]	0	20	0	P	8.25	0.725			1.425	P	0.905			1.605
[38]	7	20	0.35	U	29.2	0.026	0.81		1.71	U		0.756		1.656
[40]	10	20	0.5	A				0.636	1.636	A			0.496	1.496
[41]	0	20	0						0					0
[42]	0	20	0						0					0
[43]	33.73	20	1						0					0
[44]	6	20	0.3	U			0.851		1.751	A			0.6	1.6
[45]	0	20	0						0					0
[46]	0	20	0	U			0.805		1.705	A			0.632	1.632
[47]	1.8	20	0.09						0					0
[48]	1.344	20	0.067	U			0.910		1.810	A			0.685	1.685
[49]	6	20	0.3						0	A			0.614	1.614
[50]	0	20	0						0	A		0.764	0.631	1.631
[51]	13	20	0.65	A				0.733	1.733	A			0.587	1.587
[52]	0.56	20	0.028	U			0.854		1.754	A			0.616	1.616

A-4. Puntuaciones para el criterio de *De*, métricas De_{eex}, De_m, De_d, De_v y De_a

Ref.	M_u	M_E	E	De_{eex}	Me	M_{max}	De_m	D	D_{max}	De_d	V	V_{max}	De_v	Al	A_{max}	De_a
[5]	0	0	0	0	1	10	0.1	2	14	0.142	5	162	0.030	1	35	0.028
[6]	0	0	0	0	1	10	0.1	5	14	0.357	7	162	0.043	1	35	0.028
[7]	0	0	0	0	1	10	0.1	2	14	0.142	6	162	0.037	6	35	0.171
[15]	0	0	0	0	2	10	0.2	2	14	0.142	15	162	0.092	1	35	0.028
[21]	1	8	1	0.125	2	10	0.2	5	14	0.357	5	162	0.030	10	35	0.285
[22]	2	8	1	0.25	0	10	0	11	14	0.785	50	162	0.308	6	35	0.171
[24]	0	0	0	0	7	10	0.7	0	14	0	4	162	0.024	3	35	0.085
[25]	0	0	0	0	3	10	0.3	2	14	0.142	3	162	0.018	3	35	0.085
[26]	0	0	0	0	1	10	0.1	2	14	0.142	22	162	0.135	4	35	0.114
[27]	0	0	0	0	2	10	0.2	13	14	0.928	20	162	0.123	14	35	0.4
[28]	2	8	1	0.25	0	10	0	6	14	0.428	6	162	0.037	7	35	0.2
[29]	0	0	0	0	1	10	0.1	5	14	0.357	3	162	0.018	7	35	0.2
[30]	6	8	1	0.75	1	10	0.1	6	14	0.428	51	162	0.314	12	35	0.342
[31]	0	0	0	0	3	10	0.3	8	14	0.571	3	162	0.018	2	35	0.057
[32]	0	0	0	0	5	10	0.5	4	14	0.285	7	162	0.043	3	35	0.085
[33]	6	8	1	0.75	2	10	0.2	9	14	0.642	100	162	0.617	12	35	0.342
[34]	0	0	0	0	2	10	0.2	8	14	0.571	31	162	0.191	19	35	0.542
[35]	1	8	1	0.125	3	10	0.3	4	14	0.285	135	162	0.833	20	35	0.571
[36]	0	0	0	0	2	10	0.2	5	14	0.357	6	162	0.037	7	35	0.2
[37]	0	0	0	0	2	10	0.2	9	14	0.642	12	162	0.074	5	35	0.142
[38]	6	8	1	0.75	3	10	0.3	14	14	1	51	162	0.314	13	35	0.371
[40]	2	8	1	0.25	1	10	0.1	9	14	0.642	54	162	0.333	28	35	0.8
[41]	0	0	0	0	0	10	0	3	14	0.214	3	162	0.018	2	35	0.057
[42]	0	0	0	0	1	10	0.1	1	14	0.071	5	162	0.030	0	35	0
[43]	0	0	0	0	3	10	0.3	2	14	0.142	5	162	0.030	6	35	0.171
[44]	2	8	1	0.25	3	10	0.3	8	14	0.571	162	162	1	14	35	0.4
[45]	0	0	0	0	3	10	0.3	3	14	0.214	8	162	0.049	3	35	0.085
[46]	2	8	1	0.25	2	10	0.2	11	14	0.785	50	162	0.308	35	35	1
[47]	2	2	1	1	10	10	1	3	14	0.214	36	162	0.222	14	35	0.4
[48]	2	8	2	0.625	1	10	0.1	13	14	0.928	160	162	0.987	78	35	1
[49]	1	8	2	0.5625	2	10	0.2	0	14	0	411	162	1	29	35	0.828
[50]	3	8	2	0.520	1	10	0.1	11	14	0.785	288	162	1	20	35	0.571
[51]	2	8	2	0.625	4	10	0.4	5	14	0.357	160	162	0.987	165	35	1
[52]	2	8	1	0.25	1	10	0.1	11	14	0.785	228	162	1	70	35	1

A-5. Puntuaciones para el criterio de *DS*, lingüístico (LG), cualitativo (CL), cuantitativo (CN) y no superado (X).

Ref.	Oclusión total	Oclusión parcial	Cambios de apariencia o deformaciones	Fondos abarrotados	Escalamiento	Variación en la iluminación	Presencia de ruido	Objetos similares	Movimientos erráticos o complejos	Videos de baja resolución	Entrada y salida de escena	Fondo dinámico	Movimiento rápido	Rotación fuera del plano	Rotación en el plano	Borrosidad por movimiento
[5]																
[6]	LG,CL	LG,CL			LG,CL											
[7]	1	1														
[15]	CL,CN	CL,CN														
[21]	LG,CL	LG,CL			LG,CL	LG,CL							LG,CL			
[22]		X	LG	CN	CN	CL,CN				X	X		X	CL,CN	CL,CN	CN
[24]																
[25]		1						1								
[26]		LG,CL										LG,CL				
[27]	LG,CL	LG,CL	LG,CL	LG,CL	X	LG,CL		LG,CL		LG,CL		LG,CL	LG,CL	LG,CL	LG,CL	LG,CL
[28]		LG,CL	CL	CL,CN	LG,CL	1			LG,CL							
[29]	1	1	LG			LG,CL		1								
[30]	LG,CL	LG,CL	1	LG,CL	LG,CL	LG,CL										
[31]	LG,CL	LG,CL	LG,CL			X		LG,CL	LG,CL	X	X					
[32]	LG,CL				LG,CL							LG,CL	LG			
[33]	X	CL,CN	CL,CN	1	1	CL,CN				CL,CN			CL,CN		CL,CN	
[34]	LG,CL	LG,CL	LG,CL	LG,CL	LG,CL	LG,CL							X	LG,CL		
[35]	1	1	LG									LG				
[36]		1	CL,CN	1				1					LG,CL			
[37]	1	1	1	1		1		1	1	1						1
[38]	1	1	1	LG,CN	1	1		CL	CL	CN		CL	1	LG	LG	LG,CN
[40]	1	1	CL	1		CL		LG,CL				1		CL	CL	
[41]			LG,CL		LG	LG,CL										
[42]																
[43]		CL										CL				
[44]	X	1		CN	CN	CL						X		CN		CN
[45]	LG,CL	LG,CL							LG,CL							
[46]	X	CL	X	X	LG,CL	LG				X			X	LG	LG	X
[47]		LG,CL			LG,CL							1				
[48]	CN	CN	CN	X	CN	CN			X	CN		CN	CN	CN	CN	CN
[49]																
[50]																
[51]		LG,CN			LG,CN	X			LG,CN			LG,CN				
[52]	X	X	X	LG,CL	LG,CL	CL				LG,CL			CL	1	1	X

A-6. Puntuaciones finales

Ref.	Criterios			*Do + 3 De + 2 DS*	*PF* (%)	Posicionamiento
	Do	*De*	*DS*			
[5]	7	0.302	0	7.906	10.008	32
[6]	10.571	0.528	1.5	15.158	19.187	27
[7]	9.166	2.324	2	20.139	25.493	19
[15]	5.333	0.464	1.7	10.125	12.816	30
[21]	13.5	1.998	2.5	24.496	31.007	17
[22]	10	4.85	4.2	32.967	41.730	12
[24]	5	0.81	0	7.431	9.406	33
[25]	8	0.547	2	13.641	17.267	28
[26]	6.681	0.492	1	10.16	12.861	29
[27]	9.777	4.275	6	34.604	43.803	10
[28]	10.5	0.915	3.7	20.646	26.135	23
[29]	11.333	0.925	3.65	21.41	27.101	22
[30]	13.5	5.387	3.5	36.661	46.407	9
[31]	8.833	0.947	2.5	16.674	21.107	26
[32]	13.8	1.914	1.65	22.843	28.916	20
[33]	13.833	6.413	7.1	47.275	59.842	2
[34]	11.055	4.233	3.5	30.757	38.933	13
[35]	11.714	3.115	2.3	25.66	32.481	15
[36]	9.166	3.769	4.35	29.174	36.929	14
[37]	11.833	4.089	9	42.102	53.294	3
[38]	13.384	6.452	9.15	51.041	64.609	1
[40]	12.194	5.758	5.9	41.269	52.239	6
[41]	6.38	0.289	1.15	9.55	12.089	31
[42]	4.8	0.202	0	5.406	6.844	34
[43]	10.614	1.645	0.7	16.949	21.455	25
[44]	13.5	6.172	3.35	38.717	49.009	8
[45]	13.375	0.649	1.5	18.323	23.193	24
[46]	13.357	5.881	1.3	33.601	42.533	11
[47]	9.625	2.926	2	22.404	28.36	21
[48]	8	7.203	5.5	40.611	51.407	7
[49]	11.555	4.505	0	25.07	31.735	16
[50]	10.555	4.609	0	24.383	30.865	18
[51]	14.714	7.339	2.6	41.933	53.08	4
[52]	13.714	6.534	4.2	41.718	52.808	5

B. INFORMACIÓN DE LA BASE DE DATOS OTB

	Nombre	Cantidad de Cuadros		Nombre	Cantidad de Cuadros		Nombre	Cantidad de Cuadros
1	Basketball	725	36	Dog	127	71	Mhyang	1490
2	Biker	142	37	Dog1	1350	72	MotorRolling	164
3	Bird1	408	38	Doll	3872	73	MountainBike	228
4	Bird2	99	39	DragonBaby	113	74	Panda	1000
5	BluCar1	742	40	Dudek	1145	75	RedTeam	1918
6	BluCar3	357	41	FaceOcc1	892	76	Rubik	1997
7	BluCar4	380	42	FaceOcc2	812	77	Shaking	365
8	BlurBody	334	43	Fish	476	78	Singer1	351
9	BlurCar2	585	44	FleetFace	707	79	Singer2	366
10	BlurFace	493	45	Football	362	80	Skater	160
11	BlurOwl	631	46	Football1	74	81	Skater2	435
12	Board	698	47	Freeman1	326	82	Skating1	400
13	Bolt	350	48	Freeman3	460	83	Skating2-1	473
14	Bolt2	293	49	Freeman4	283	84	Skating2-1	473
15	Box	1161	50	Girl	500	85	Skiing	81
16	Boy	602	51	Girl2	1500	86	Soccer	392
17	Car1	1020	52	Gym	767	87	Subway	175
18	Car2	913	53	Human2	1128	88	Surfer	376
19	Car24	3059	54	Human3	1698	89	Suv	945
20	Car4	659	55	Human4	667	90	Sylvester	1345
21	CarDark	393	56	Human5	713	91	Tiger1	354
22	CarScale	252	57	Human6	792	92	Tiger2	365
23	ClifBar	472	58	Human7	250	93	Toy	271
24	Coke	291	59	Human8	128	94	Trans	124
25	Couple	140	60	Human9	305	95	Trellis	569
26	Coupon	327	61	Ironman	166	96	Twinnings	472
27	Crossing	120	62	Jogging-1	307	97	Vase	271
28	Crowds	347	63	Jogging-2	307	98	Walking	412
29	Dancer	225	64	Jump	122	99	Walking2	500
30	Dancer2	150	65	Jumping	313	100	Woman	597
31	David	471	66	KiteSurf	84			
32	David2	537	67	Lemming	1336			
33	David3	252	68	Liquor	1741			
34	Deer	71	69	Man	134			
35	Diving	215	70	Matrix	100			

C.RESULTADOS DE LA EVALUACIÓN OPE

C-1. Desempeños de los algoritmos *fc-cnn*, *fc-cnn_kf* y *fc-cnn_inc* en 20 secuencias de video de OTB [53].

	Nombre de la secuencia	*fc-cnn*		*fc-cnn_kf*		*fc-cnn_inc*	
		Precisión	**Éxito**	**Precisión**	**Éxito**	**Precisión**	**Éxito**
1	Biker	0.466	0.350	0.478	0.380	0.912	0.675
2	Boy	0.770	0.650	0.143	0.130	0.923	0.795
3	CarDark	0.969	0.881	0.913	0.689	0.913	0.689
4	Coupon	0.908	0.846	0.306	0.276	0.360	0.334
5	Crossing	0.945	0.767	0.946	0.779	0.344	0.264
6	Dancer	0.851	0.773	0.860	0.805	0.860	0.805
7	Dancer2	0.840	0.780	0.880	0.831	0.876	0.830
8	David2	0.850	0.574	0.938	0.796	0.933	0.782
9	Dog	0.694	0.361	0.289	0.194	0.791	0.472
10	Fish	0.897	0.839	0.059	0.059	0.823	0.742
11	Freeman1	0.450	0.320	0.275	0.195	0.804	0.610
12	Gym	0.094	0.077	0.616	0.595	0.616	0.595
13	Human2	0.128	0.295	0.027	0.038	0.668	0.794
14	Human8	0.932	0.793	0.910	0.772	0.922	0.783
15	Man	0.952	0.835	0.940	0.813	0.891	0.755
16	Mhyang	0.827	0.740	0.721	0.615	0.721	0.615
17	Panda	0.497	0.305	0.831	0.498	0.837	0.499
18	Skater	0.615	0.596	0.256	0.265	0.692	0.659
19	Toy	0.680	0.628	0.684	0.645	0.616	0.595
20	Vase	0.642	0.675	0.084	0.097	0.424	0.517
Promedio		**0.700**	**0.604**	**0.558**	**0.474**	**0.746**	**0.641**
Velocidad de procesamiento (FPS)		**4.435**		**15.9059**		**15.9072**	

C-2. Desempeños de los algoritmos *fc-cnn_kf*, *fc-cnn_inc*, *fc-cnn_inc1*, *fc-cnn_inc2*, *fc-cnn_inc3*, y *fc-cnn_inc4* en las 100 secuencias de video de OTB [53]. (Parte 1/4)

	Nombre de la secuencia	Cantidad de cuadros	*fc-cnn_kf*		*fc-cnn_inc*		*fc-cnn_inc1*		*fc-cnn_inc2*		*fc-cnn_inc3*		*fc-cnn_inc4*	
			Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito
1	Basketball	725	0.08	0.06	0.08	0.06	0.08	0.06	0.03	0.02	0.69	0.61	0.44	0.38
2	Biker	142	0.48	0.38	0.91	0.68	0.91	0.66	0.92	0.70	0.92	0.69	0.93	0.71
3	Bird1	408	0.36	0.24	0.28	0.20	0.35	0.23	0.41	0.25	0.37	0.23	0.32	0.22
4	Bird2	99	0.47	0.44	0.85	0.79	0.80	0.74	0.86	0.79	0.80	0.74	0.80	0.74
5	BlurBody	334	0.46	0.49	0.79	0.83	0.79	0.82	0.80	0.84	0.80	0.83	0.80	0.83
6	BlurCar1	742	0.02	0.02	0.36	0.37	0.11	0.13	0.77	0.77	0.86	0.86	0.83	0.84
7	BlurCar2	585	0.89	0.88	0.89	0.88	0.89	0.88	0.90	0.89	0.90	0.89	0.90	0.89
8	BlurCar3	357	0.23	0.22	0.89	0.84	0.91	0.86	0.87	0.82	0.91	0.85	0.91	0.86
9	BlurCar4	380	0.87	0.89	0.87	0.89	0.86	0.88	0.90	0.90	0.89	0.90	0.89	0.90
10	BlurFace	493	0.74	0.71	0.85	0.83	0.85	0.83	0.84	0.82	0.85	0.82	0.85	0.82
11	BlurOwl	631	0.03	0.03	0.65	0.62	0.88	0.84	0.68	0.65	0.89	0.85	0.89	0.85
12	Board	698	0.04	0.12	0.78	0.85	0.73	0.83	0.79	0.85	0.76	0.84	0.76	0.84
13	Bolt	350	0.82	0.65	0.82	0.65	0.75	0.60	0.82	0.64	0.81	0.64	0.82	0.65
14	Bolt2	293	0.02	0.01	0.02	0.01	0.02	0.01	0.51	0.37	0.02	0.01	0.02	0.01
15	Box	1161	0.35	0.34	0.34	0.34	0.76	0.76	0.73	0.74	0.78	0.78	0.77	0.78
16	Boy	602	0.14	0.13	0.92	0.80	0.92	0.79	0.91	0.77	0.92	0.80	0.92	0.78
17	Car1	1020	0.95	0.76	0.95	0.77	0.96	0.80	0.95	0.77	0.96	0.81	0.96	0.81
18	Car2	913	0.96	0.90	0.93	0.84	0.94	0.86	0.93	0.84	0.94	0.86	0.95	0.87
19	Car24	3059	0.54	0.36	0.58	0.36	0.51	0.33	0.56	0.36	0.48	0.32	0.46	0.31
20	Car4	659	0.95	0.90	0.95	0.90	0.95	0.90	0.94	0.90	0.95	0.91	0.95	0.91
21	CarDark	393	0.91	0.69	0.91	0.69	0.91	0.67	0.91	0.69	0.91	0.67	0.91	0.68
22	CarScale	252	0.84	0.82	0.85	0.82	0.87	0.82	0.88	0.84	0.87	0.84	0.86	0.82
23	ClifBar	472	0.15	0.14	0.18	0.17	0.31	0.28	0.15	0.15	0.30	0.28	0.16	0.15
24	Coke	291	0.09	0.08	0.71	0.56	0.71	0.57	0.71	0.56	0.72	0.57	0.72	0.57
25	Couple	140	0.72	0.58	0.81	0.66	0.81	0.66	0.72	0.57	0.81	0.64	0.78	0.63
26	Coupon	327	0.31	0.28	0.36	0.33	0.36	0.34	0.36	0.34	0.36	0.34	0.36	0.34
27	Crossing	120	0.95	0.78	0.34	0.26	0.95	0.78	0.95	0.80	0.95	0.79	0.95	0.80

C-2. Desempeños de los algoritmos *fc-cnn_kf*, *fc-cnn_inc*, *fc-cnn_inc1*, *fc-cnn_inc2*, *fc-cnn_inc3*, y *fc-cnn_inc4* en las 100 secuencias de video de OTB [53]. (Parte 2/4)

	Nombre de la secuencia	Cantidad de cuadros	*fc-cnn_kf*		*fc-cnn_inc*		*fc-cnn_inc1*		*fc-cnn_inc2*		*fc-cnn_inc3*		*fc-cnn_inc4*	
			Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito
28	Crowds	347	0.87	0.60	0.87	0.60	0.86	0.58	0.89	0.65	0.89	0.70	0.86	0.59
29	Dancer	225	0.86	0.81	0.86	0.81	0.87	0.81	0.86	0.81	0.87	0.81	0.87	0.81
30	Dancer2	150	0.88	0.83	0.88	0.83	0.88	0.84	0.87	0.83	0.88	0.84	0.88	0.84
31	David	471	0.91	0.81	0.90	0.78	0.91	0.81	0.90	0.78	0.91	0.81	0.92	0.81
32	David2	537	0.94	0.80	0.93	0.78	0.94	0.79	0.94	0.80	0.90	0.69	0.90	0.70
33	David3	252	0.82	0.71	0.83	0.71	0.83	0.72	0.82	0.71	0.85	0.73	0.84	0.73
34	Deer	71	0.29	0.27	0.89	0.82	0.89	0.82	0.88	0.81	0.89	0.83	0.89	0.83
35	Diving	215	0.32	0.30	0.62	0.53	0.39	0.34	0.72	0.60	0.31	0.28	0.33	0.31
36	Dog	127	0.29	0.19	0.79	0.47	0.79	0.48	0.76	0.44	0.78	0.46	0.79	0.47
37	Dog1	1350	0.82	0.74	0.82	0.74	0.82	0.75	0.85	0.77	0.83	0.76	0.83	0.76
38	Doll	3872	0.73	0.69	0.69	0.67	0.66	0.65	0.68	0.66	0.67	0.66	0.65	0.65
39	DragonBaby	113	0.78	0.70	0.78	0.70	0.54	0.49	0.75	0.67	0.59	0.53	0.58	0.52
40	Dudek	1145	0.73	0.79	0.73	0.79	0.76	0.81	0.72	0.79	0.76	0.81	0.75	0.81
41	FaceOcc1	892	0.03	0.04	0.74	0.79	0.68	0.75	0.77	0.81	0.72	0.77	0.72	0.78
42	FaceOcc2	812	0.38	0.36	0.80	0.76	0.80	0.76	0.81	0.76	0.82	0.77	0.81	0.77
43	Fish	476	0.06	0.06	0.82	0.74	0.85	0.77	0.83	0.75	0.84	0.76	0.86	0.78
44	FleetFace	707	0.56	0.69	0.56	0.69	0.57	0.70	0.48	0.63	0.53	0.68	0.51	0.67
45	Football	362	0.17	0.15	0.86	0.68	0.16	0.15	0.17	0.15	0.86	0.67	0.17	0.15
46	Football1	74	0.09	0.08	0.09	0.08	0.09	0.08	0.71	0.51	0.68	0.46	0.66	0.52
47	Freeman1	326	0.28	0.19	0.80	0.61	0.36	0.28	0.83	0.62	0.84	0.63	0.83	0.61
48	Freeman3	460	0.89	0.67	0.89	0.67	0.88	0.67	0.88	0.68	0.89	0.68	0.89	0.68
49	Freeman4	283	0.30	0.17	0.52	0.24	0.47	0.21	0.82	0.51	0.50	0.27	0.50	0.27
50	Girl	500	0.92	0.78	0.92	0.77	0.92	0.76	0.92	0.76	0.91	0.76	0.92	0.76
51	Girl2	1500	0.05	0.05	0.73	0.67	0.58	0.59	0.68	0.63	0.58	0.59	0.67	0.66
52	Gym	767	0.62	0.60	0.62	0.60	0.60	0.61	0.63	0.62	0.61	0.61	0.53	0.58
53	Human2	1128	0.03	0.04	0.67	0.79	0.67	0.79	0.64	0.79	0.67	0.79	0.68	0.79
54	Human3	1698	0.76	0.62	0.73	0.59	0.64	0.53	0.68	0.58	0.61	0.53	0.60	0.52

C-2. Desempeños de los algoritmos *fc-cnn_kf*, *fc-cnn_inc*, *fc-cnn_inc1*, *fc-cnn_inc2*, *fc-cnn_inc3*, y *fc-cnn_inc4* en las 100 secuencias de OTB [53]. (Parte 3/4)

	Nombre de la secuencia	Cantidad de cuadros	*fc-cnn_kf*		*fc-cnn_inc*		*fc-cnn_inc1*		*fc-cnn_inc2*		*fc-cnn_inc3*		*fc-cnn_inc4*	
			Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito
55	Human4	667	0.18	0.16	0.18	0.16	0.18	0.16	0.18	0.16	0.19	0.16	0.47	0.40
56	Human5	713	0.23	0.18	0.78	0.68	0.82	0.67	0.78	0.65	0.82	0.66	0.83	0.66
57	Human6	792	0.73	0.70	0.73	0.70	0.75	0.72	0.72	0.70	0.74	0.72	0.74	0.72
58	Human7	250	0.77	0.69	0.44	0.40	0.44	0.40	0.44	0.40	0.44	0.40	0.65	0.59
59	Human8	128	0.91	0.77	0.92	0.78	0.94	0.82	0.93	0.79	0.94	0.82	0.95	0.83
60	Human9	305	0.49	0.38	0.48	0.38	0.48	0.38	0.46	0.38	0.75	0.65	0.43	0.38
61	Ironman	166	0.04	0.03	0.04	0.03	0.04	0.03	0.08	0.07	0.11	0.10	0.11	0.10
62	Jogging-1	307	0.91	0.81	0.91	0.81	0.91	0.82	0.90	0.81	0.91	0.83	0.91	0.82
63	Jogging-2	307	0.17	0.15	0.90	0.82	0.92	0.83	0.90	0.82	0.92	0.82	0.92	0.83
64	Jump	122	0.68	0.52	0.68	0.52	0.06	0.06	0.69	0.53	0.06	0.06	0.06	0.06
65	Jumping	313	0.10	0.09	0.77	0.58	0.13	0.10	0.12	0.10	0.12	0.11	0.75	0.59
66	KiteSurf	84	0.37	0.27	0.93	0.72	0.93	0.74	0.44	0.34	0.44	0.34	0.43	0.35
67	Lemming	1336	0.01	0.01	0.70	0.67	0.73	0.70	0.47	0.45	0.81	0.77	0.72	0.69
68	Liquor	1741	0.17	0.16	0.89	0.86	0.88	0.85	0.86	0.84	0.88	0.86	0.88	0.86
69	Man	134	0.94	0.81	0.89	0.75	0.93	0.80	0.94	0.81	0.94	0.81	0.94	0.82
70	Matrix	100	0.24	0.17	0.24	0.17	0.28	0.20	0.37	0.28	0.66	0.51	0.58	0.47
71	Mhyang	1490	0.72	0.62	0.72	0.62	0.82	0.72	0.73	0.62	0.82	0.73	0.83	0.73
72	MotorRolling	164	0.14	0.25	0.14	0.25	0.41	0.46	0.17	0.28	0.42	0.45	0.44	0.47
73	MountainBike	228	0.78	0.67	0.79	0.68	0.79	0.68	0.77	0.66	0.79	0.67	0.79	0.68
74	Panda	1000	0.83	0.50	0.84	0.50	0.85	0.52	0.85	0.52	0.83	0.50	0.83	0.50
75	RedTeam	1918	0.91	0.67	0.91	0.67	0.89	0.64	0.90	0.65	0.89	0.64	0.90	0.65
76	Rubik	1997	0.49	0.54	0.50	0.55	0.72	0.70	0.50	0.55	0.63	0.64	0.63	0.64
77	Shaking	365	0.87	0.80	0.87	0.80	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
78	Singer1	351	0.94	0.90	0.94	0.90	0.93	0.89	0.94	0.90	0.93	0.88	0.93	0.88
79	Singer2	366	0.24	0.26	0.41	0.44	0.34	0.38	0.69	0.65	0.37	0.39	0.63	0.63
80	Skater	160	0.26	0.27	0.69	0.66	0.71	0.69	0.73	0.69	0.74	0.71	0.74	0.71
81	Skater2	435	0.72	0.66	0.72	0.66	0.71	0.65	0.72	0.66	0.72	0.66	0.72	0.66

C-2. Desempeños de los algoritmos *fc-cnn_kf*, *fc-cnn_inc*, *fc-cnn_inc1*, *fc-cnn_inc2*, *fc-cnn_inc3*, y *fc-cnn_inc4* en las 100 secuencias de datos OTB [53]. (Parte 4/4)

	Nombre de la secuencia	Cantidad de cuadros	*fc-cnn_kf*		*fc-cnn_inc*		*fc-cnn_inc1*		*fc-cnn_inc2*		*fc-cnn_inc3*		*fc-cnn_inc4*	
			Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito	Precisión	Éxito
82	Skating1	400	0.25	0.17	0.60	0.59	0.31	0.31	0.35	0.35	0.39	0.34	0.55	0.54
83	Skating2-1	473	0.71	0.64	0.71	0.64	0.72	0.65	0.70	0.63	0.72	0.64	0.71	0.64
84	Skating2-2	473	0.53	0.57	0.52	0.56	0.52	0.57	0.51	0.55	0.52	0.57	0.52	0.56
85	Skiing	81	0.17	0.10	0.17	0.10	0.15	0.10	0.14	0.08	0.14	0.09	0.16	0.10
86	Soccer	392	0.10	0.10	0.11	0.11	0.21	0.19	0.16	0.14	0.16	0.15	0.16	0.15
87	Subway	175	0.92	0.74	0.92	0.73	0.92	0.74	0.92	0.73	0.91	0.74	0.91	0.74
88	Surfer	376	0.90	0.73	0.90	0.73	0.91	0.74	0.90	0.73	0.92	0.75	0.92	0.75
89	Suv	945	0.88	0.82	0.52	0.49	0.51	0.49	0.87	0.81	0.81	0.76	0.88	0.82
90	Sylvester	1345	0.81	0.67	0.83	0.69	0.82	0.67	0.83	0.69	0.81	0.65	0.82	0.66
91	Tiger1	354	0.06	0.06	0.62	0.59	0.62	0.59	0.60	0.57	0.67	0.63	0.67	0.63
92	Tiger2	365	0.25	0.28	0.18	0.21	0.19	0.22	0.56	0.51	0.64	0.58	0.64	0.57
93	Toy	271	0.68	0.64	0.62	0.60	0.53	0.51	0.37	0.40	0.42	0.42	0.40	0.42
94	Trans	124	0.35	0.49	0.33	0.60	0.38	0.63	0.34	0.61	0.43	0.66	0.38	0.64
95	Trellis	569	0.05	0.05	0.80	0.68	0.93	0.87	0.80	0.67	0.82	0.71	0.82	0.71
96	Twinnings	472	0.39	0.34	0.54	0.48	0.55	0.49	0.54	0.48	0.72	0.63	0.54	0.49
97	Vase	271	0.08	0.10	0.42	0.52	0.41	0.51	0.45	0.54	0.44	0.53	0.46	0.54
98	Walking	412	0.94	0.80	0.93	0.79	0.93	0.79	0.93	0.79	0.93	0.80	0.94	0.81
99	Walking2	500	0.88	0.79	0.86	0.76	0.90	0.81	0.87	0.76	0.90	0.81	0.90	0.81
100	Woman	597	0.14	0.12	0.81	0.64	0.80	0.63	0.81	0.66	0.80	0.66	0.84	0.67
Promedio			**0.51**	**0.45**	**0.68**	**0.60**	**0.65**	**0.59**	**0.68**	**0.61**	**0.70**	**0.62**	**0.69**	**0.62**
Velocidad de procesamiento (FPS)			**15.9059**		**15.9072**		**15.6646**		**13.4121**		**13.4787**		**11.3958**	