INSTITUTO TECNOLÓGICO DE CHIHUAHUA

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

"ANÁLISIS Y COMPARACIÓN DE ALGORITMOS DE MODELADO DE FONDO DE SECUENCIAS DE VIDEO"

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS EN INGENIERÍA EN ELECTRÓNICA

PRESENTA:

ING. ABIMAEL GUZMÁN PANDO

DIRECTOR DE LA TESIS:

DR. MARIO IGNACIO CHACÓN MURGUÍA

CODIRECTOR DE LA TESIS:

DRA. GRACIELA MARÍA DE JESÚS RAMÍREZ ALONSO

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

CHIHUAHUA, CHIH., DICIEMBRE 2017

AGRADECIMIENTOS

A mis padres Efrén y Rosa, por el apoyo y formación que me brindaron cada día con su amor incondicional, por ser un ejemplo y una inspiración para lograr cada objetivo que me he propuesto. A mis hermanos Esmeralda y Aldahír, quienes siempre me han mostrado su apoyo y cariño.

A Marisol Ramírez, por inspirarme a ser una mejor persona y a dar lo mejor de mí todos los días a través de su amor incondicional, y por apoyarme en la realización de este trabajo de tesis.

Al Dr. Mario Chacón por brindarme la oportunidad de integrarme al equipo de trabajo del Laboratorio de Percepción Visual con Aplicaciones en Robótica (PVR), por el conocimiento, asesorías y consejos que me trasmitió a lo largo de este trabajo de investigación. Mi más sincero respeto y admiración hacia su trabajo y persona.

A mis profesores Dr. Francisco Corral, Dra. Didia Salas, M.C. José Robles, Dr. Pedro Acosta, por ser parte de mi formación académica y brindarme los conocimientos necesarios para concluir exitosamente el plan de estudios de la maestría en ciencias. En especial, al Dr. Javier Vega y Dr. Juan Ramírez, quiénes además, tomaron parte de su tiempo para revisar esta tesis y brindarme asesorías y consejos para realizar un trabajo de calidad.

Un especial agradecimiento a la Dra. Graciela Ramírez por su apoyo a lo largo de este trabajo de tesis, por sus consejos, y brindarme herramientas que fueron clave para el desarrollo de este trabajo. Reitero mi respeto y admiración hacia su trabajo y persona.

Al personal del DEPI, por ser una mano amiga en los trámites necesarios durante mi estancia en la maestría, especialmente a M.C. Alma Corral.

A todos mis compañeros y amigos del PVR y DEPI, Jorge, Luis, Miguel, Salvador, Nadia, Elisa, Moisés Miguel Ángel, Esteban, Alan Eruviel, Carlos, Eduardo, Brenda, Oscar, Manuel, José, Andrea, Daniel, Raúl, Rodolfo, Arnoldo, Daniel (Sosa), David, Daniel Vargas, María Magdalena, entre otros. Ellos hicieron que mi estancia en la maestría fuera más llevadera y amena brindándome su apoyo y amistad.

RESUMEN

ANÁLISIS Y COMPARACIÓN DE ALGORITMOS DE MODELADO DE FONDO DE SECUENCIAS DE VIDEO

Ing. Abimael Guzmán Pando
Maestría en Ciencias en Ingeniería Electrónica
División de Estudios de Posgrado e Investigación del
Instituto Tecnológico de Chihuahua
Chihuahua, Chih. 2017
Director de Tesis: Dr. Mario Ignacio Chacón Murguía
Codirector de Tesis: Dra. Graciela María de Jesús Ramírez Alonso

La detección de movimiento en video ha sido extensamente utilizada en aplicaciones como vigilancia, análisis de marcha humana, monitoreo, identificación de objetos, entre otras. Para desarrollar esta tarea uno de los métodos involucra obtener un modelo de fondo, el cual, luego es comparado contra el cuadro actual del video para así poder detectar los objetos en movimiento. El obtener el modelo de fondo no es una tarea sencilla de realizar debido a los problemas presentes en aplicaciones reales, tales como: cambio de iluminación, vibraciones de cámara, oclusión, fondos dinámicos (ej. las hojas de los árboles, lluvia o copos de nieve), camuflaje, entre otros factores. Debido a lo anterior, en los últimos años se ha desarrollado una gran variedad de algoritmos, aumentando considerablemente el conocimiento de la rama enfocada al análisis de video. Para su validación, existen bases de datos de evaluación de desempeño de algoritmos de detección de objetos dinámicos que utilizan modelado de fondo como *Change detection*, *SBMnet*, *BMC*, *SBI*, entre otras. Sin embrago, son procesos de evaluación no estandarizados por lo que no se pueden utilizar como una norma.

En este trabajo de tesis, se elabora una metodología de evaluación, comparación y selección de los mejores algoritmos de modelado de fondo y detección de movimiento vigentes, con el fin de determinar los potenciales existentes en el área de la detección de movimiento en los diferentes paradigmas actuales, y así poder proporcionar ideas útiles para el desarrollo de nuevo conocimiento y nuevos métodos. La metodología propuesta para la

selección de los mejores algoritmos, se realizó mediante cinco criterios referentes a evaluación, documentación, auto-adaptabilidad, desempeño, velocidad y actualidad. El análisis fue dividido en dos partes, una dirigida a algoritmos de detección de movimiento y la otra a algoritmos de modelado de fondo. En total, 112 algoritmos fueron considerados para el análisis enfocado a detección de movimiento y 34 algoritmos para el de modelado de fondo.

Debido a que los algoritmos son elaborados y probados en diferentes circunstancias, con factores que no están establecidos como una norma o regla a seguir, se realizaron dos experimentos. El primero consistió en evaluar el desempeño de cinco personas contra los *Ground truth* de la base de datos *Change detection*, esto con la finalidad de medir el grado de error humano o varianza que existe al realizar un *Ground truth.* El segundo experimento, evalúa el desempeño de los diez mejores algoritmos de detección de movimiento contra los resultados obtenidos por las segmentaciones humanas, con el propósito de obtener una referencia de cómo se sitúan los algoritmos respecto al humano, y si existían ocasiones en las que los algoritmos daban mejores resultados. Para ambos experimentos se incluyeron las métricas enfocadas a medir la calidad de segmentación SSIM y D-Score.

Tres de los primeros diez algoritmos de la base de datos *Change detection* no figuran dentro de los diez primeros de la evaluación propuesta; estos son: IUTIS-5, IUTIS-3 y SaliencySubsense. Por otro lado, en los experimentos realizados, los sujetos obtuvieron un promedio de 0.948 en la métrica F-Measure y 0.441 en PWC, el promedio humano en la métrica SSIM fue de 0.959 y 0.005 en D-Score, implicando que aún la más mínima variación en el *Ground truth*, produce impacto considerable en los resultados de las métricas. Como punto a destacar, un algoritmo obtuvo resultados por encima del promedio humano y los demás dentro de los primeros diez, obtuvieron un porcentaje entre el 84 y 89% de cercanía a los resultados del desempeño humano.

Los humanos, al igual o en mayor cantidad que los algoritmos tienden a fallar en categorías de video donde existen situaciones críticas de video tales como fondos dinámicos, camuflaje, problemas por desenfoque, entre otras. Por lo tanto, el proceso de realizar la evaluación respecto a un *Ground truth* elaborado por un humano resulta en una evaluación subjetiva. Tanto más subjetiva, en cuanto menos entrenamiento tenga el humano para realizar

tareas de segmentación. Por lo que, aún y cuando en bases de datos comúnmente usadas como *Change detection,* se plantea un método rápido de conocer y ubicar el impacto o importancia de un algoritmo, no es correcto considerarlas como la única alternativa para realizar comparaciones entre algoritmos. Por ello, la metodología propuesta en este trabajo de tesis pretende utilizar más criterios de evaluación para determinar los mejores algoritmos, en lugar de únicamente su desempeño, como usualmente se hace.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

GLOSARIO

Simbolo	Descripción
t	Variable temporal
α	Constante de rapidez de cambio
T	Constante de umbral
N	Numero de cuadros o imágenes posteriores a la actual
μ_t	Media en el instante de tiempo t que caracteriza a una distribución Gaussiana
σ_t^2	Varianza en el instante de tiempo t que caracteriza a una distribución Gaussiana
k	Numero de distribuciones Gaussianas
$I(x,y)_t$	Imagen actual I en el tiempo t
$B(x,y)_t$	Fondo actual B en el tiempo t
δ	Constante de proporción de aprendizaje
i	variable que representa iteraciones
c	Constante de umbral especifico
$i(x,y)_t$	Pixel en tiempo t, en la posición (x, y)
$w_{i,t}$	Peso asociado a la i-ésima Gaussiana en el tiempo t
$\mu_{i,t}$	Media asociada a la i-ésima Gaussiana en el tiempo t
$\Sigma_{i,t}$	Matriz de covarianza asociada a la i-ésima Gaussiana en el tiempo t
d	Número de canales usados en una distribución
x	Pixel
X	Historia reciente de un pixel x
$\eta(X_t \mid \mu, \sigma)$	Función de densidad de probabilidad Gaussiana
S	Variable aleatoria

$Kr(s)$	Función Kernel o ventana
$P(s_t)$	Probabilidad de observar un evento dada una variable aleatoria
h	Ancho de la función Kernel
U	El conjunto de algoritmos existentes enfocados a la detección de movimiento
A_A	Conjunto de algoritmos analizados
A_{BD}	Conjunto de algoritmos situados en las bases de datos
$Pr_{NùmV}$	Puntuación obtenida en el elemento "número de videos" en la categoría "procesamiento" en el criterio "documentación"
nm	Mínimo número de videos requeridos para lograr un punto en $Pr_{NùmV}$
Nv	Número de videos usados para realizar sus evaluaciones de desempeño del algoritmo
Pr_{Res}	Puntuación obtenida en el elemento "resolución" en la categoría "procesamiento" en el criterio "documentación"
Pr_{ResFPS}	Puntuación otorgada al elemento "resolución y cuadros por segundo (FPS)" en la categoría "procesamiento" en el criterio "documentación"
Pr_{EspC}	Puntuación otorgada al elemento "espacio de color" en la categoría "procesamiento" en el criterio "documentación"
Pr	Puntuación otorgada a la categoría "procesamiento" en el criterio "documentación"
$Ev_{NúmBD}$	Puntuación otorgada al elemento "base de datos" en la categoría "evaluación" en el criterio "documentación"
N_{BD}	Número de bases de datos usadas para realizar sus evaluaciones de desempeño del algoritmo
Ev_{BDCom}	Puntuación obtenida en el elemento "base de datos completa" en la categoría "evaluación" en el criterio "documentación"
Ev_{ECual}	Puntuación obtenida en el elemento "evaluación cualitativa" en la categoría "evaluación" en el criterio "documentación"
Ev_{ECuan}	Puntuación obtenida en el elemento "evaluación cuantitativa" en la categoría "evaluación" en el criterio "documentación"

$Ev_{NúmM}$	Puntuación obtenida en el elemento "número de métricas" en la categoría "evaluación" en el criterio "documentación"
N_M	Número de métricas reportadas al realizar la evaluación de desempeño del algoritmo.
Ev_{MPar}	Puntuación obtenida en el elemento "mismos parámetros para todos los videos" en la categoría "evaluación" en el criterio "documentación"
Ev	Puntuación obtenida en la categoría "evaluación" en el criterio "documentación"
Im_{Sof}	Puntuación obtenida en el elemento "software" en la categoría "implementación" en el criterio "documentación"
Im_{CPU}	Puntuación obtenida en el elemento "CPU/GPU" en la categoría "implementación" en el criterio "documentación"
Im_{RAM}	Puntuación obtenida en el elemento "RAM" en la categoría "implementación" en el criterio "documentación"
Im	Puntuación obtenida en la categoría "implementación" en el criterio "documentación"
$Co_{NúmA}$	Puntuación obtenida en el elemento "número de algoritmos" en la categoría "comparaciones" en el criterio "documentación"
Co	Puntuación obtenida en la categoría "comparaciones" en el criterio "documentación"
P_{DT}	Suma de las puntuaciones obtenidas en el criterio "documentación"
P_{Do}	Peso otorgado al criterio "documentación"
$P_{DOCUMENTACIÓN}$	Puntuación final obtenida en el criterio "documentación"
$Complex$	Puntuación obtenida en la categoría "complejidad" en el criterio "auto-adaptabilidad"
P_M	Número de parámetros manuales usados por un algoritmo
P_A	Número de parámetros automáticos usados por un algoritmo
$MaxP_M$	Máximo número de parámetros manuales
$MaxP_A$	Máximo número de parámetros automáticos
$Entr$	Puntuación obtenida en la categoría "entrenamiento" en el criterio "auto-adaptabilidad"

$GraIH$	Puntuación obtenida en la categoría “grado de intervención humana” en el criterio “auto-adaptabilidad”
M	Parámetro auxiliar utilizado en la puntuación GraIH
P_{AT}	Suma de las puntuaciones obtenidas en el criterio “auto-adaptabilidad”
P_{Aa}	Peso del criterio “auto-adaptabilidad”
$P_{AUTO\text{-}ADAPTABILIDAD}$	Puntuación final en el criterio “auto-adaptabilidad”
T_{PBD}	Puntuación obtenida para cada base de datos
NV_{BD}	Número de videos utilizados por las bases de datos
$GTPV_{BD}$	Número de GT utilizados por las bases de datos
NM_{BD}	Número de métricas utilizadas por las bases de datos
$PBD_j(n)$	Puntuación obtenida por el algoritmo en la posición del ranking n en la j-ésima base de datos
n	Posición del algoritmo en el ranking
j	j-ésima base de datos
PM_j	Puntuación máxima que puede obtener el algoritmo en la base de datos j
P_{De}	Peso otorgado al criterio “desempeño”
$P_{DESEMPEÑO}$	Puntuación final en el criterio “desempeño”
R_N	Factor de normalización
R_O	Resolución reportada en la publicación del algoritmo
FPS_O	Velocidad reportada en la publicación del algoritmo
V_N	Velocidad en cuadros por segundo normalizada
P_{Ve}	Peso otorgado al criterio “velocidad”
$P_{VELOCIDAD}$	Puntuación final en el criterio “velocidad”
β	Constante de rapidez de cambio
A_P	Año de publicación del articulo
P_{Ac}	Peso otorgado al criterio “actualidad”
$P_{ACTUALIDAD}$	Puntuación final en el criterio “actualidad”

P_{FINAL}	Suma de las puntuaciones obtenidas en los criterios: actualidad, velocidad, desempeño, auto-adaptabilidad y documentación
MA_{DM}	Conjunto de los primeros diez algoritmos de detección de movimiento
MCD_{DM}	Conjunto de los primeros diez algoritmos de detección de movimiento y con código disponible
C_{BD}	Conjunto de algoritmos en base de datos enfocadas en modelado de fondo
$S_{ACTUALIDAD}$	Puntuación final en el criterio "actualidad" para algoritmos enfocados en modelado de fondo
$S_{VELOCIDAD}$	Puntuación final en el criterio "velocidad" para algoritmos enfocados en modelado de fondo
$S_{DSESEMPEÑO}$	Puntuación final en el criterio "desempeño" para algoritmos enfocados en modelado de fondo
T_S	Puntuación obtenida para cada base de datos de modelado de fondo
NV_S	Número de videos utilizados por las bases de datos de modelado de fondo
$GTPV_S$	Número de GT utilizados por las bases de datos de modelado de fondo
NM_S	Número de métricas utilizadas por las bases de datos de modelado de fondo
l	l-ésima base de datos enfocada en modelado de fondo
SM_l	Puntuación máxima que puede obtener el algoritmo en la base de datos l
$SBD_l(n)$	Puntuación obtenida por el algoritmo en la posición del ranking n en la l-ésima base de datos
APs	Año de publicación del artículo del algoritmo enfocado en modelado de fondo
S_{FINAL}	Suma de las puntuaciones obtenidas en los criterios: actualidad, velocidad y desempeño

CAPÍTULO I. ANTECEDENTES

En la actualidad, los algoritmos de detección de movimiento de objetos en secuencias de video, se han vuelto un factor importante para implementar diversas aplicaciones de visión por computadora tales como: sistemas de vigilancia, navegación autónoma, identificación automática de objetivos, reconocimiento de actividad humana, monitoreo, entre otras [1]–[4]. La función principal de estos sistemas es proveer una interpretación automática de las escenas y predecir las acciones e interacciones de los objetos observados en base a la información previamente adquirida.

Los sistemas de detección de movimiento se basan principalmente en separar los objetos en movimiento, también llamados objetos dinámicos, del escenario del fondo ya sea estático o dinámico. Existe una gran cantidad de métodos para la detección de objetos en movimiento de los cuales, uno de los más simples es sustracción de fondo, en él se adquiere un escenario sin objetos en movimiento, llamado fondo y se compara contra el cuadro actual del video, de manera que, al restar el cuadro actual al fondo y aplicar una binarización en la imagen, obtendremos únicamente los objetos en movimiento, tal como se aprecia en la figura 1.1. Sin embargo, la aplicación de este método no es muy eficiente debido a la variedad de cambios que se pueden suscitar en un ambiente real o no controlado.

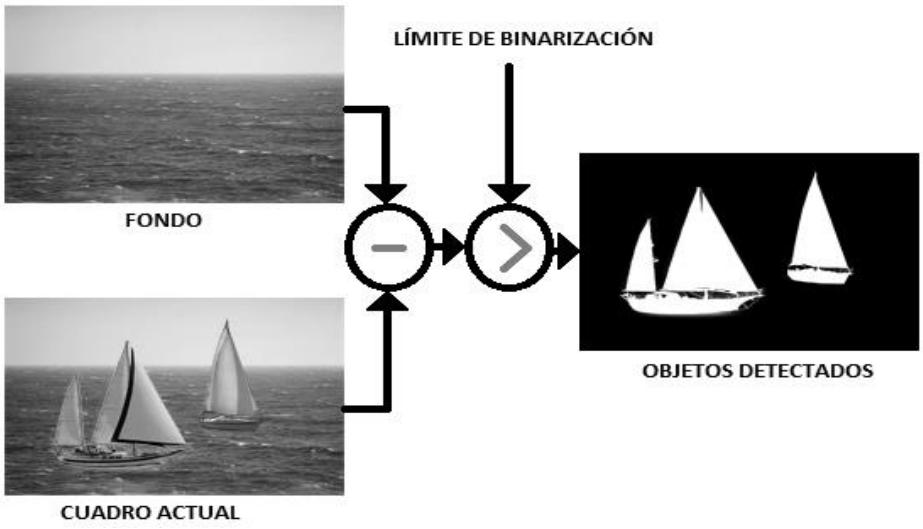


Figura 1.1 Sustracción de fondo simple.

Tener una iluminación constante, un fondo estático y una cámara fija son algunas características ideales que aseguran una detección fidedigna y funcional en los métodos de

sustracción de fondo. No obstante, en la práctica aparecen situaciones críticas que complican este proceso. En 1999, Kentaro Toyama [5] identificó 10 problemas en el campo de video vigilancia, tales como: las vibraciones de cámara, el cambio de iluminación gradual durante el día, el cambio repentino de luz, fondos dinámicos como árboles u ondulaciones en el agua, camuflaje, entre otros. Recientemente en el 2014 Thierry Bouwmans [6] extendió esta lista a 13 situaciones críticas, las cuales son listadas a continuación:

- **Ruido en la imagen (*noisy image*)**: problema debido a la pobre calidad de la imagen en cámaras con baja resolución o videos en formatos comprimidos.
- **Vibraciones o movimientos en cámara (*camera Jitter)***: movimiento no deseado debido al viento o falta de un posicionamiento estático al grabar.
- **Ajustes automáticos en cámara (*PTZ*)**: debido a que la mayoría de las cámaras modernas cuentan con ajuste de enfoque automático, brillo y contraste, hace que los niveles de color entre los cuadros de una secuencia de video se modifiquen dinámicamente siendo difícil su predicción.
- **Cambios de iluminación (*illumination changes*)**: cuando se tiene un escenario real, por lo general, la iluminación cambia gradual o repentinamente, originando falsas detecciones que pueden ser desde pequeños grupos de pixeles afectados hasta todos los pixeles. Por ejemplo, el cambio repentino de iluminación al prender y apagar un foco.
- **Mala inicialización de modelo de fondo *(bootstrapping)***: ocurre cuando en el periodo de entrenamiento, el fondo no puede ser correctamente representado, esto debido a que se encuentran presentes en el estado inicial del video, objetos en movimiento o alguna de las situaciones críticas mencionadas.
- **Camuflaje (*camouflage*)**: se da cuando no se pueden detectar los objetos dinámicos en el escenario de fondo, esto debido a que ambos presentan características muy similares en color o textura.
- **Apertura de primer plano (*foreground aperture)***: cuando los objetos tienen grandes regiones homogéneas, los cambios en los pixeles interiores no pueden ser detectados y por lo tanto, el objeto completo se vuelve parte del fondo.
- **Objetos del fondo movibles**: cuando inicialmente algunos objetos representan el fondo, luego son movidos y una vez que son cambiados de posición pasan a ser detectados como objetos en primer plano, sin embargo deberían seguir siendo considerados como fondo.

- **Objetos insertados en el fondo**: cuando se tiene un escenario de fondo y al paso del tiempo se introduce un nuevo objeto, el cual, no debe ser considerado como objeto en movimiento.
- **Fondos dinámicos (*dynamic background*)**: generalmente, en escenarios abiertos existen objetos moviéndose que no deben ser considerados como fondo, tal es el caso del movimiento de las ramas u hojas de los árboles, o el movimiento de las olas en el océano, ondulaciones, etc. Estas situaciones conforman un fondo dinámico y en todos los casos existe una gran cantidad de falsas detecciones de objetos dinámicos.
- **Efecto fantasma (*ghosting)***: cuando inicialmente un objeto se encuentra en el modelo de fondo y comienza a moverse, tanto el objeto, como las nuevas partes del fondo reveladas, son detectadas como objeto dinámico, llamándose a este efecto *ghosting*.
- **Objetos en primer plano estáticos**: ocurre cuando inicialmente un objeto se encuentra en movimiento y luego deja de moverse, al permanecer estático por un periodo de tiempo este es incorporado al fondo.
- **Sombras (*shadows)***: ocurre cuando las sombras son detectadas como objetos en primer plano, estas pueden provenir de objetos moviéndose o del fondo.

1.1. Modelos enfocados a detección de movimiento

Las situaciones críticas antes mencionadas, han sido la motivación de una gran cantidad de investigadores para proponer métodos nuevos que puedan lidiar con estos problemas con el fin de obtener una detección del objeto en movimiento lo más robusta posible. Algunas de las técnicas más representativas enfocadas a detección de movimiento son mostradas a continuación.

1.1.1 Modelos básicos

Los modelos básicos son métodos pioneros en el campo de detección de movimiento y debido a su facilidad de implementación siguen siendo utilizados en una gran cantidad de aplicaciones, aunque mayormente en ambientes controlados, ya que no son muy robustos para manejar un número considerable de situaciones críticas. La mayoría de las técnicas de detección de movimiento tienen el común denominador de proponer un modelo de fondo, es decir, un modelo del escenario que no contenga objetos en movimiento. Los modelos básicos

se caracterizan por su simplicidad para obtener el modelo del fondo. Algunas de las técnicas que caen dentro de esta categoría son filtro de mediana, filtro de promediado o por medio del análisis de histograma. Una vez obtenido el modelo del fondo, es comparado con el cuadro actual del video y la nueva imagen obtenida es binarizada a un cierto umbral para separar el fondo, de los objetos dinámicos, a manera que:

$$dif\left(I(x,y)_t, B(x,y)_t\right) > T \tag{1.1}$$

Donde $I(x,y)_t$ representa el cuadro actual de la imagen, $B(x,y)_t$ representa el fondo actual previamente modelado y T es una constante del límite de binarización.

1.1.1.1 Filtro de promedio temporal

Es un método que utiliza un promedio de N cuadros posteriores para proponer el modelo del fondo [7], mediante:

$$B(x,y)_t = \frac{1}{N}\sum_{i=1}^{N} I(x,y)_{t-i} \tag{1.2}$$

Donde $B(x,y)_t$ representa el nuevo modelo de fondo, N el número de cuadros posteriores sobre los cuales se obtiene el promedio e $I(x,y)_{t-i}$ es el cuadro de la imagen correspondiente al tiempo t-i.

El realizar el promedio de las N imágenes posteriores es una tarea relativamente sencilla, al usar una cantidad considerable de cuadros se pueden eliminar objetos que se mueven a gran velocidad. Sin embargo, al utilizar pocos cuadros, se produce un efecto peculiar, en el cual, las posiciones anteriores del objeto dejan una huella semitransparente en el modelo de fondo. Uno de los principales inconvenientes de este método es que al usar mayor número de cuadros también se limita la memoria, debido a la gran cantidad de datos con los que se tiene que realizar el promedio. En la figura 1.2 se ilustra el concepto de aplicar un filtro de promedio temporal.

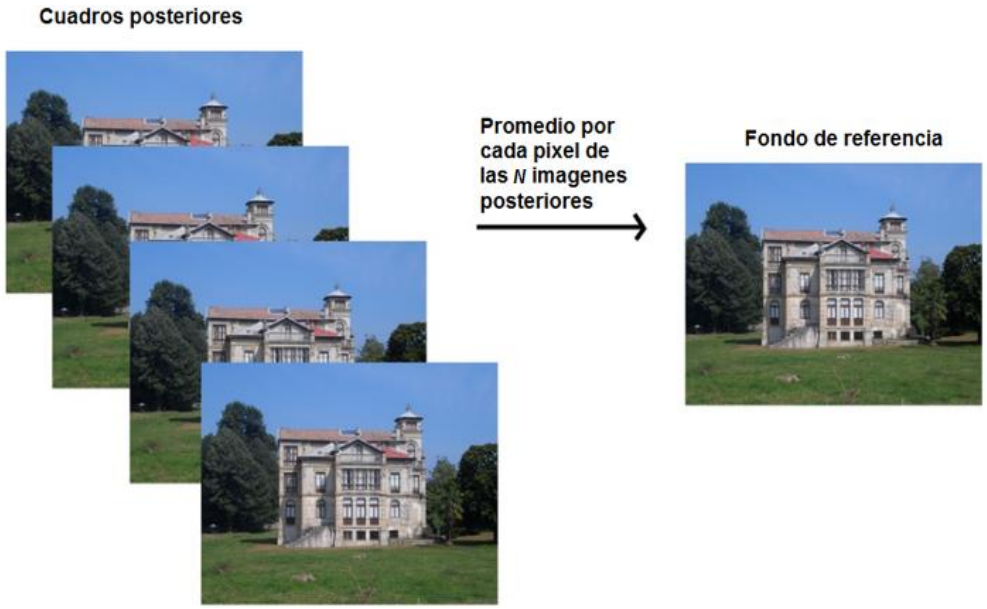


Figura 1.2 Modelado de fondo usando filtro de promedio.

1.1.1.2 Filtro de mediana temporal

Es un modelo similar al filtro de promedio temporal excepto que el fondo será determinado por la mediana de los píxeles de un *N* número de cuadros [8]. Su fórmula es la siguiente:

$$B_r(x,y)_t = Mediana(I(x,y)_{t-i}), \quad i = 0,1,2...,N \tag{1.3}$$

Donde $I(x,y)_{t-i}$ es el número de cuadros posteriores *t-i* sobre los que se realiza la operación de mediana, yendo desde *t* hasta *t-N* cuadros posteriores y $B(x,y)_t$ será el nuevo modelo de fondo.

1.1.1.3 Aproximación por filtro de mediana

Es uno de los métodos más comúnmente usado debido a su fácil implementación, a no requerir una gran cantidad de parámetros y su consumo bajo de recursos [9] y [10]. Se basa en modelar un fondo de referencia con base en el cálculo del valor de cada pixel de la imagen por medio de la siguiente ecuación:

$$B(x,y)_t = \begin{cases} B(x,y)_{t-1} + 1 & si \ \ I(x,y)_t > B(x,y)_{t-1} \\ B(x,y)_{t-1} - 1 & si \ \ I(x,y)_t < B(x,y)_{t-1} \end{cases} \tag{1.4}$$

Donde la imagen del cuadro actual $I(x,y)_t$ se compara con el cuadro del fondo previamente modelado $B(x,y)_{t-1}$ y se propone un nuevo fondo $B(x,y)_t$, sumando o restando uno, al valor de tono de gris del pixel, dependiendo de si la comparación es mayor o menor.

Este modelo es conocido como filtro de mediana debido a que el fondo calculado en cada iteración eventualmente converge a una estimación donde la mitad de los valores de los pixeles son mayores que el fondo y la otra mitad menores, lo cual se asemeja al comportamiento de la mediana estadística. El principal problema de este método es que la recuperación a los cambios repentinos en el escenario de fondo, es lenta.

Como una mejora a este método, en [11] se agrega una variable α como una constante de rapidez de cambio, de modo que no solo se añade o se sustrae el valor de uno a los pixeles del modelo del fondo. En este método se propone la relación uno sobre alfa. Al variar el valor de alfa, el modelo de fondo se actualiza, o más rápido, o más lento, permitiendo un manejo más flexible.

$$B(x,y)_t = \begin{cases} B(x,y)_{t-1} + \dfrac{1}{\alpha} & si \;\; \mathrm{I}(x,y)_t > B(x,y)_{t-1} \\ B(x,y)_{t-1} - \dfrac{1}{\alpha} & si \;\; \mathrm{I}(x,y)_t < B(x,y)_{t-1} \end{cases} \tag{1.5}$$

1.1.1.4 Aproximación por filtro de promedio

Es un método que calcula un modelo de fondo $B(x,y)_t$ para cada cuadro del video $I(x,y)_t$ en el tiempo *t*, a partir de la siguiente formula:

$$B(x,y)_t = \delta I(x,y)_t + (1-\delta) B(x,y)_{t-1} \tag{1.6}$$

Donde δ es una constante de proporción de aprendizaje.

Este método es relativamente preciso para la estimación del fondo cuando se tiene una iluminación constante, pero en cambios repentinos o fondos dinámicos resulta en un gran número de falsos positivos. El principal problema de este método radica en la velocidad de adaptación del modelo de fondo determinada por el factor δ ya que el método asume que todos los pixeles del cuadro actual tienen el mismo valor de δ, lo cual no es una buena opción

cuando existen cambios repentinos únicamente en algunas partes de la imagen, ya que los pixeles del modelo del fondo en zonas con cambios repentinos deberían ser actualizados más rápido y los de zonas constantes a un ritmo lento [12].

1.1.2 Modelos paramétricos

Son métodos que basan su modelado de fondo en funciones de densidad de probabilidad conocidas, cómo, por ejemplo, la distribución Gaussiana, uniforme, exponencial, logarítmica, entre otras. Estas funciones de distribución de probabilidad servirán para dar una aproximación al modelo de fondo ya que se asume que las variaciones de los pixeles en la imagen a través del tiempo tienen un comportamiento estocástico.

1.1.2.1 Modelos Gaussianos

Son modelos que utilizan un conjunto de datos aleatorios para hacer inferencias basadas en el cálculo de las probabilidades. En detección de movimiento, estos modelos basan su modelado de fondo en una función de densidad probabilística para obtener el valor correspondiente de cada pixel. Los modelos gaussianos asumen que el comportamiento de los valores que tienen los pixeles a lo largo del tiempo puede ser representado por un modelo de distribución Gaussiana. A continuación, se detallan los métodos más representativos de esta categoría.

Gaussiana simple [13]*:*

Es un método relativamente sencillo el cual usa una función Gaussiana para representar la variación de intensidad de color de cada pixel en el tiempo, es decir, cada píxel cuenta con su distribución de probabilidad Gaussiana caracterizada por una media y una varianza que son calculadas de la siguiente manera:

$$\mu_t = \frac{1}{N}\sum_{i=1}^{N} I\left(x,y\right)_{t-i} \tag{1.7}$$

$$\sigma_t^2 = \frac{1}{N-1}\sum_{i=1}^{N} (I\left(x,y\right)_{t-i} - \mu_t)^2 \tag{1.8}$$

Donde *N* es el número de cuadros posteriores considerados, μ_t y σ_t^2 representan la media y varianza de la distribución Gaussiana respectivamente, y $(I\left(x,y\right)_{t-i} - \mu_t)^2$ es la distancia

euclidiana entre la media de la distribución Gaussiana y el valor del pixel de la imagen en la posición *(x,y)*. La media, es el valor esperado del pixel en el espacio definido por la función, la varianza describe la dispersión o propagación de la posición del valor del tono de gris de los píxeles con respecto de la media. En cada instante *t* se determina si un pixel pertenece al modelo de fondo si el valor del pixel cae dentro de la Gaussiana definida por ese pixel, es decir, si la diferencia del valor actual del pixel y la media, es menor a un umbral $c\sigma_t$:

$$si\left|I\left(x,y\right)_t-\mu_t\right|<c\sigma_t\rightarrow B\left(x,y\right)_t \tag{1.9}$$

Siendo *c* es una constante de umbral específico y σ_t el valor de la desviación estándar.

Uno de los inconvenientes de este método, es que no es capaz de resolver fondos dinámicos, ya que los pixeles en las zonas de fondos dinámicos toman valores muy abruptos y una sola función Gaussiana no logra aproximar el comportamiento del pixel correctamente, por lo que, se obtendrá un comportamiento oscilante en la detección de movimiento, entre detectar el pixel como un objeto dinámico, y como parte del fondo.

Mezcla de Gaussianas [14]*:*

Este método surge a raíz de que el modelo gaussiano simple no puede manejar fondos dinámicos, por lo que se propone una solución en [15], donde se modelan los cambios del valor de los pixeles a través del tiempo (la historia reciente de cada pixel) a través de una mezcla de *k* distribuciones Gaussianas. El parámetro *k* por lo general es un número muy pequeño, de 3 a 5 dependiendo del escenario dinámico en análisis. Debido a que en el tiempo *t* lo que se conoce de cada pixel es su historia *X*, se tiene:

$$\left\{X_1,X_2,...,X_t\right\}=\left\{i(x,y)_i\quad donde:\ \ 1\le i\le t\right\} \tag{1.10}$$

Y la probabilidad de observar un valor de pixel determinado es:

$$P\left(X_t\right)=\sum_{i=1}^{k}w_{i,t}\eta(X_t\mid\mu_{i,t},\Sigma_{i,t}) \tag{1.11}$$

Donde $w_{i,t}$ es un peso asociado a la i-ésima Gaussiana en el tiempo *t* con una media $\mu_{i,t}$ y varianza $\Sigma_{i,t}$ y la función de densidad de probabilidad Gaussiana η queda definida mediante:

$$\eta(X_t \mid \mu, \sigma) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{|\Sigma|}} e^{-\frac{1}{2}\left((X_t-\mu)^T \Sigma^{-1} (X_t-\mu)\right)} \tag{1.12}$$

Donde cada pixel es caracterizado por una mezcla de k distribuciones Gaussianas de dimensión *d*, siendo la matriz de covarianza definida por la desviación estándar σ^2 y la matriz identidad *I*, mediante:

$$\Sigma_{i,t} = \sigma^2{}_{i,t} I \tag{1.13}$$

En el modelo de mezcla de Gaussianas, el modelo de fondo que representa la variación de cada pixel, será la combinación de las *k* distribuciones Gaussianas con mayor peso $w_{i,t}$ que superen un determinado umbral T :

$$B(x, y)_{t+1} = \sum_{i=1}^{k} w_{i,t} > T \tag{1.14}$$

El peso de una Gaussiana representa el número de veces que aparece el mismo estado (objeto en movimiento o fondo) por cada pixel. Entre más veces aparezca el estado, más peso adquirirá la Gaussiana; de esta manera, si se tiene un objeto dinámico que luego se queda estático, al principio el conjunto de Gaussianas no tendrá mucho peso. Sin embargo, conforme pase el tiempo, el peso aumentará proporcional al tiempo que el objeto permanezca estático, y a su vez el objeto se añadirá al fondo. Las distribuciones Gaussianas con menor peso $w_{i,t}$ son consideradas para representar los objetos dinámicos, mediante el uso de la distancia de Mahalanobis de la historia reciente del pixel X_t con respecto a su media $\mu_{i,t}$:

$$\sqrt{(X_t - \mu_{i,t})^T \Sigma_{i,t}^{-1} (X_t - \mu_{i,t})} < c\sigma_{i,t} \tag{1.15}$$

Donde *c* al igual que el modelo de gaussiano simple es una constante de umbral, generalmente igual a 2.5, teniendo la ecuación 1.15 anterior, dos casos pueden ocurrir:

1) Si la condición es satisfecha y esa distribución Gaussiana con media $\mu_{i,t}$ estaba previamente considerada como parte del modelo de fondo, entonces el pixel es clasificado como fondo, de otro modo será clasificado como objeto en movimiento.

2) Si no se satisface la condición el pixel será considerado como objeto en movimiento.

De esta manera se obtiene una máscara binara representando el fondo y los objetos dinámicos.

El método de mezcla de Gaussianas, es un método muy robusto para manejar fondos complejos y dinámicos. Sin embargo, cuando existen cambios rápidos en las escenas de video, se requiere incrementar el número de Gaussianas necesarias para lograr modelar el fondo correctamente, lo que conlleva a un costo computacional elevado. En la figura 1.3 se muestra el proceso de realizar el método mezcla de Gaussianas, donde el comportamiento de cada pixel es aproximado mediante una función Gaussiana multivariada.

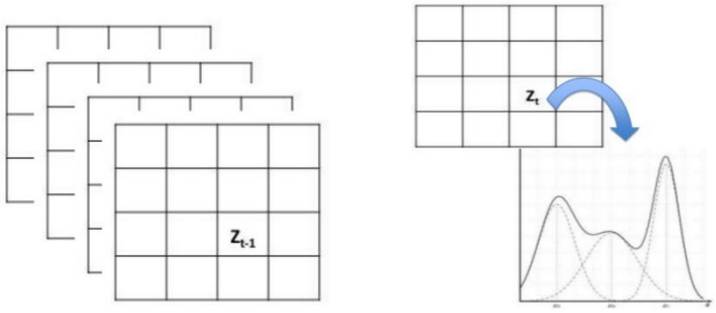


Figura 1.3 Función Gaussiana multivariada de los *n* cuadros posteriores.

1.1.3 Modelos no paramétricos

Son modelos estadísticos cuya distribución subyacente no se ajusta a los llamados criterios paramétricos. Su distribución no puede ser definida a priori, pues son los datos observados los que la determinan.

1.1.3.1 Kernel Density Estimation

Realiza una estimación no paramétrica de funciones de densidad que no siguen un modelo conocido como la función normal, binomial, exponencial, etc. La estimación, se realiza a partir de la información derivada de los datos, por lo que la estructura del modelo es suministrada por el número de muestras [16]. Se basa en el uso de funciones Kernel cuya denominación se refiere a toda función integrable, real definida no negativa, que cumpla:

$$\int_{-\infty}^{\infty} Kr(s)\,ds = 1 \tag{1.16}$$

Siendo *Kr(s)* la función Kernel y *s(1), s(2),…s(N)* el conjunto de muestras conocidas de una variable aleatoria *S*. La variable *S* se puede considerar, como el comportamiento del pixel a través del tiempo.

La estimación de la función de densidad probabilística (PDF por sus siglas en inglés) será la suma promediada de todas las funciones Kernel centradas en cada una de las muestras conocidas. Es decir:

$$P(s_t) = \frac{1}{Nm} \sum_{i=1}^{Nm} \frac{Kr(s - s_i)}{h} \tag{1.17}$$

Donde *h* es el ancho de la función Kernel utilizada, *Nm* es el conjunto de muestras de variables aleatorias utilizadas, $Kr(s\text{-}s_i)$ es la función Kernel evaluada en la muestra actual *s* menos la *i*-ésima muestra s_i.

El parámetro *Nm* es usado para normalizar la PDF resultante, debido a que existen *Nm* funciones Kernel cada una con un área 1 y estas serán sumadas y normalizadas con el fin de asegurar que el área de la PDF estimada resultante sea igual o menor a uno.

Normalmente las funciones utilizadas como Kernel son Gaussianas, la calidad de estimación de la PDF no reside principalmente en la función Kernel utilizada, si no en el ancho seleccionado, tal como se observa en la figura 1.4. A tamaños muy pequeños se obtiene una estimación demasiado variable con bastantes picos, y a tamaños muy grandes una estimación muy pobre, en la siguiente imagen se puede observar el tamaño idóneo de la función de Kernel.

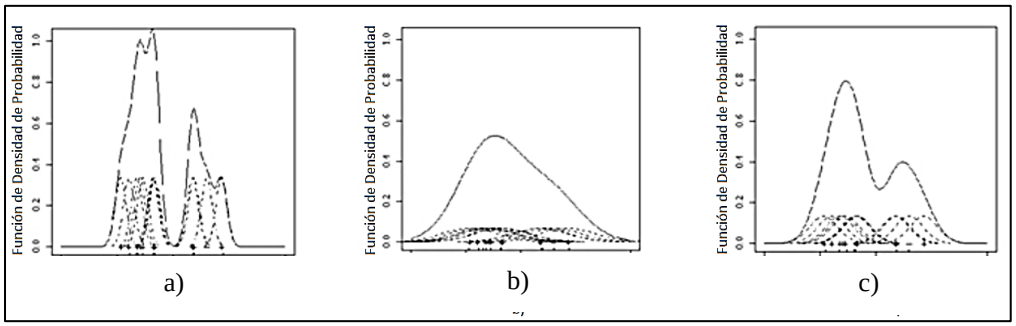


Figura 1.4 Tamaños de Kernel: a) ancho pequeño, b) ancho grande, c) ancho óptimo [14].

En particular, en algoritmos de modelado de fondo de secuencias de video, la estimación de la función de densidad de probabilidad del método Kernel Density Estimation queda definida mediante la ecuación 1.18 como un producto de los canales de color.

$$P(x) = \frac{1}{N}\sum_{i=1}^{N}\prod_{j=1}^{d}\frac{K\left(x_j - x_{i,j}\right)}{h_j} \tag{1.18}$$

Donde *d* representa el número de canales de color, x_j el pixel actual, $x_{i,j}$ la historia reciente de los valores de pixel de fondo y *N* el número de cuadros posteriores tomados en cuenta para moldear el fondo.

Para realizar la detección de movimiento, si la probabilidad de un pixel es mayor que un umbral *T*, se considera ese pixel como parte del modelo de fondo, en caso contrario será considerado como pixel de objeto dinámico.

Este método es útil para manejar fondos dinámicos, sin embargo, se debe escoger correctamente el número de muestras usadas en el modelo, ya que a mayor número de muestras, mejor será la estimación y se podrán manejar escenarios más complejos. Sin embargo, existirá un costo computacional elevado y se reducirá la velocidad de procesamiento.

1.2 Trabajos previos enfocados a análisis de algoritmos de detección de movimiento

En los últimos años se han tenido importantes aportaciones en el campo de detección de movimiento. Algoritmos y nuevos métodos han sido publicados y pueden encontrarse en diversos artículos y revistas en el estado del arte, e incluso hay autores que han ofrecido comparaciones y clasificaciones acerca de los modelos de detección.

Massimo Piccardi [17] en 2004, propuso un método de clasificación de algoritmos basado en su velocidad de ejecución, requerimientos de memoria y precisión. Algunos de los métodos de detección que consideró más importantes son el filtro de mediana, el promediado gaussiano, el estimador de densidad y el modelo de mezcla de Gaussianas (GMM). Cuatro años más tarde, Y. Benezeth et al. [18], ofrecen un trabajo enfocado a los métodos comúnmente utilizados en la detección de movimiento, tales como: el método básico,

gaussiano simple, GMM, MinMax y *Kernel density estimation*. Los autores presentan una evaluación y comparación de los métodos cuantitativa y cualitativamente a manera de medir el rendimiento y robustez en aplicaciones reales con video secuencias que representan diferentes retos de procesamiento. Luego, Norbert Buch [19] propone una clasificación de algoritmos de detección de movimiento en sistemas de monitoreo de tráfico urbano con base en su rendimiento y su posible aplicación en tiempo real, un aporte interesante que evalúa los métodos en escenarios urbanos y carreteras, para su implementación en aplicaciones de análisis de tráfico. Despues, Thierry Bouwmans [6] da una clasificación completa de algoritmos, dividiéndolos en dos categorías principales, modelos tradicionales y modelos recientes, en ellas se da una amplia información para elegir el mejor método de detección dependiendo de los problemas que se necesiten solucionar, tales como: cambios de iluminación, fondos dinámicos, sombras, ajustes automáticos de cámara, ruido en la imagen, etc. Thierry define los modelos tradicionales como algoritmos de fácil implementación y capaces de manejar cambios específicos siendo los primeros usados en el área de análisis de video. Por otro lado, los modelos recientes son considerados como algoritmos robustos y complicados que pueden manejar mayor cantidad de cambios, y la mayoría de ellos requieren mejoras para poder cumplir con los requerimientos de un procesamiento en tiempo real. Recientemente en el 2015, los investigadores Jaya S. Kulchandani y Kruti J. Dangarwala [20] describen y presentan las tendencias de investigación en el área de algoritmos enfocados en detección de objetos dinámicos, y proporcionan una tabla comparativa con sus respectivas limitaciones y ventajas. En la tabla 1.1 se observa los métodos que fueron tomados en cuenta por Jaya y Kruti para la realización de sus comparativas.

En este capítulo se presentaron los algoritmos y aspectos más significativos concernientes al estado del arte enfocado a detección de movimiento. Se describieron los métodos básicos de modelado de fondo y se detallaron sus principales ventajas y desventajas. Además, se hizo referencia a las recopilaciones o clasificaciones más recientes de métodos de detección de movimiento, lo anterior la finalidad de proporcionar información necesaria para discernir que método escoger dado un problema o situación determinada de video. También, con base en las limitaciones que presenta cada uno de los métodos se pueden establecer nuevas investigaciones que conduzcan a la mejora de dichos algoritmos e incluso el desarrollo de nuevos métodos con mejores desempeños.

Tabla 1.1 Comparativa de métodos realizada por Jaya S. y Kurt J [20].

Publicación	Año	Título	Resumen	Ventajas	Limitaciones
IEEE	2009	*Moving Object and Shadow Detection Based on RGB Color Space and Edge Ratio*	Separación de objetos, sombras y fondo, usando modelado de espacio de color RGB considerando cromaticidad y proporción de brillo combinado con modelo de proporción de borde para tratamiento de clasificaciones erróneas de objetos y sombras	*Objetos en movimiento y sombras son determinados de manera separada *Suficientemente rápido para utilización en tiempo real	*Áreas de sombras oscuras u objetos en movimiento con la misma información de color que la del fondo fallarán
ELSEVIER	2011	*Robust moving object detection against fast illumination change*	Identificación de objetos en movimiento bajo variaciones de iluminación rápida usando GMM para detección y modelado de proporción de cromaticidad y brillo para eliminación de falsas detecciones	*No requiere secuencia de entrenamiento *Ajuste automático de los parámetros	*Resultados degradados en ambientes como nieve, charcos, o superficies reflejantes
IEEE	2012	*Spatio-Temporal Traffic Scene Modeling for Object Motion Detection*	Vigilancia de tráfico usando método de fusión Bayesiano donde un Kernel de estimación de densidad es usado para modelar el fondo y una formulación Gaussiana en encargada del modelado de objetos en movimiento	*Requiere menos tiempo computacional *Trabaja bien con cambios rápidos y lentos en el fondo	*Características idénticas de objetos en el fondo son anuladas
IEEE	2013	*An Improved Moving Objects Detection Algorithm*	Una mejora al método de diferencia de tres cuadros combinada con información completa de ganancia relacionada con el objeto en movimiento	*Efecto fantasma eliminado *Algoritmo vence el fenómeno vacío y problemas de supresión de bordes de método diferencia de tres cuadros estándar	*El resultado no es ideal en ambientes con luz fuerte y sombras *Resultados degradados para fondo dinámico
IEEE	2013	*A Moving Target Detection Algorithm Based on Dynamic Scenes*	Diferencia de cinco cuadros combinada con método de sustracción de fondo para detección de blancos en movimiento	*Objetos en movimiento pueden ser extraídos con más precisión y completos de escenas dinámicas	*No puede eliminar revoloteos *No puede identificar múltiples blancos
ELSEVIER	2014	*The 3dSOBS+ algorithm for moving object detection*	Detección de objetos en movimiento por medio de un modelado de fondo neuronal el cual es automáticamente creado por su propio método de organización	*Trabaja bien con fondos dinámicos *Ajuste preciso con variaciones de iluminación graduales y sombras de objetos en movimiento	*No tiene precisión en caso de variación de iluminación repentina y refracción
IEEE	2014	*Image Processing Based Vehicle Detection and Tracking Method*	Reconocimiento y seguimiento de vehículos usando GMM y detección de blob	*Conteo de vehículos automático *Robusto para bajo y medio tráfico	*En caso de mucho tráfico el rendimiento falla
IEEE	2014	*Moving Object Detection Based on Temporal Information*	Hace uso de información temporal para generar una saliente de movimiento la cual es luego seguida por el método de entropía máxima y crecimiento difuso para identificar objetos en movimiento	*No necesita aprendizaje inicial *Robusto para pequeños movimientos de cámara *Ajuste automático de parámetros y trata eficientemente con perturbaciones de fondo	*La sombra se determina junto con el objeto en movimiento y puede ser clasificado erróneamente como el propio objeto
HINDAWI	2014	*Moving Object Detection and Shadow Removing under Changing Illumination Condition*	Modelo de proporción de intensidad local usado para la eliminación de sombras seguido por GMM para detección de objetos en movimiento	*Identificación de objetos en movimiento sin sombras y condiciones de cambios de iluminación	*No trabaja bien cuando el fondo y el objeto son parecidos *No puede adaptarse a cambios repentinos de luz

CAPÍTULO II. ELEMENTOS DE EVALUACIÓN DE DESEMPEÑO

En este capítulo se resumen los elementos usados actualmente para realizar las evaluaciones del desempeño de los algoritmos, con la finalidad de proveer un panorama más amplio acerca de las métricas usadas en las evaluaciones, el *Ground truth*, los resultados de clasificación extraídos de la matriz de confusión, así como las principales bases de datos encontradas en la literatura. El conocer este tipo de información es útil, para en un futuro, llegar a un consenso de las formas más convenientes o acertadas para realizar una evaluación justa entre algoritmos.

2.1 Ground truth y clasificaciones de pixeles generadas en la evaluación

El *Ground truth* o GT representa el resultado deseado o esperado y contiene información que nos ayuda a conocer que tan cercano está el resultado de un algoritmo del valor o característica ideal. En algoritmos de modelado de fondo el *Ground truth* puede hacer referencia a la imagen que contiene el modelo de fondo deseado, sin embargo, para algoritmos enfocados en detección de movimiento generalmente se usa una imagen binaria, la cual es elaborada manualmente, en esta imagen se identifican los pixeles que pertenecen a los objetos en movimiento con valor de 1 y el fondo con 0. El *Ground truth* al ser comparado con el resultado de detección del algoritmo (máscara binaria) se generan las siguientes clasificaciones de pixeles:

Falsos Positivos (FP): se producen cuando existen pixeles clasificados por el algoritmo con valor 1 que en el GT son 0, es decir; un pixel fue erróneamente detectado como parte de un objeto en movimiento cuando en realidad era parte del fondo.

Falsos Negativos (FN): se obtienen cuando existen pixeles clasificados por el algoritmo con valor 0 pero en el GT tienen valor de 1, es decir; un pixel fue erróneamente detectado como parte del fondo cuando debió ser detectado como objeto en movimiento.

Verdadero Positivo (VP): se producen cuando existen pixeles con valor 1 correctamente asignados. Los pixeles del objeto en movimiento son asignados correctamente.

Verdadero Negativo (VN): se producen cuando existen pixeles con valor 0 correctamente asignados. Los pixeles del fondo son asignados correctamente

En la figura 2.1 se muestran las clasificaciones de pixeles previamente descritas, al realizar la comparación del GT contra el resultado obtenido por un algoritmo. En la figura 2.1c, se observan unas zonas amarillas, las cuales pertenecen a los falsos negativos, las zonas azules a los falsos positivos, la zona roja a los verdaderos positivos y la zona negra los verdaderos negativos.

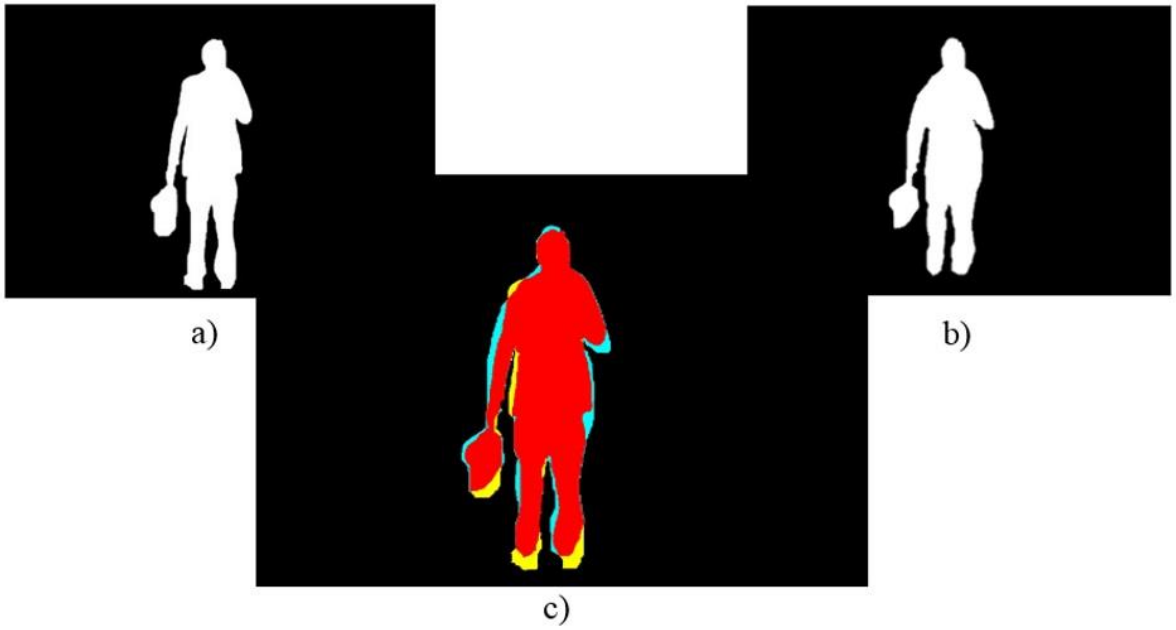


Figura 2.1 a) GT, b) resultado obtenido de algún algoritmo de detección de movimiento, c) comparación de la imagen a) con b), VP (zonas rojas), FP (zonas azules), FN (zonas amarillas) y VN (zonas negras).

Normalmente para dar una mejor visualización a las cuatro clasificaciones antes mostradas se utiliza una matriz de confusión. La matriz de confusión es una herramienta de evaluación que compara los valores reales contra los valores de predicción. Para el caso de un problema enfocado a detección de movimiento la matriz de confusión se compone de dos filas y dos columnas que reportan de manera cuantitativa el número de falsos positivos, falsos negativos, verdaderos positivos y verdaderos negativos. Esta matriz resume los resultados de las pruebas realizadas y sirve como una forma rápida de verificar el desempeño del algoritmo.

Lo anterior se observa en la tabla 2.1, por un lado, se tendrá la condición verdadera que en este caso es representada mediante el GT y por el otro lado se tendrá la condición de predicción, la cual será el resultado obtenido por el algoritmo de detección de movimiento.

Tabla 2.1 Matriz de confusión.

		Condición de predicción	
		Objeto en movimiento	Fondo
Condición verdadera	Objeto en movimiento	Verdadero Positivo (VP)	Falso Negativo (FN)
	Fondo	Falso Positivo (FP)	Verdadero Negativo (VN)

Las cuatro clasificaciones mostradas FP, FN, VP y VN son la base de algunas de las métricas enfocadas a medir el desempeño del algoritmo, las cuales son mostradas a continuación.

2.2 Métricas para evaluación de desempeño

Una forma cuantitativa de conocer que tan cercano es el resultado de un algoritmo con respecto al GT, es a través de métricas que permiten medir el desempeño del algoritmo. A continuación, se muestran algunas métricas reportadas en la literatura:

Especificidad (*Specificity*): cantidad o proporción de resultados negativos clasificados correctamente. La especificidad queda definida como el número de resultados negativos correctos divido sobre el número de todos los posibles resultados de una clasificación. La especificidad es lo contrario a la sensibilidad. A manera que:

$$Specificity = \frac{VN}{VN + FP} \tag{2.1}$$

Sensibilidad (*Sensitivity* ó *Recall)*: mide la capacidad de detectar resultados positivos correctamente. La sensibilidad se define como el número de resultados positivos dividido por el número de valores positivos verdaderos más los valores negativos incorrectamente clasificados. Por lo tanto, estará dada por la siguiente relación:

$$Recall = \frac{VP}{VP + FN} \tag{2.2}$$

Precisión (*Presicion*): definida como la capacidad de un sistema de dar el mismo resultado en mediciones diferentes realizadas en las mismas condiciones. La precisión queda definida como el número de resultados positivos correctos (VP) dividido por el número de todos los posibles resultados correctos de una clasificación. La precisión refleja la repetibilidad de la medida y dice que cantidad de pixeles son relevantes, por lo tanto:

$$Presicion = \frac{VP}{VP + FP} \tag{2.3}$$

Medida-F (*F-Measure*): es una medida de exactitud empleada en análisis estadístico de clasificación binaria, considera la precisión y la sensibilidad para dar un peso del promedio de estas dos medidas. Se define como el doble producto de la precisión por la sensibilidad dividido por la suma de las mismas. Mientras más cercano a 1 se encuentre el resultado de esta medida, mejor será el desempeño. Su fórmula queda definida por:

$$F\text{-}Measure = \frac{2*(Presicion * Recall)}{Presicion + Recall} \tag{2.4}$$

FPR (*False Positive Rate*): tasa de falsos positivos, es una métrica muy usada al realizar comparaciones, es conocida también como tasa de falsas alarmas. Se calcula a través de la relación del número de falsos positivos entre el número total de eventos negativos y se expresa por la siguiente relación:

$$FPR = \frac{FP}{FP + TN} \tag{2.5}$$

FNR (*False Negative Rate*): tasa de falsos negativos; métrica complementaria a la sensibilidad. Refleja la frecuencia con la que se producen clasificaciones erróneas. La sensibilidad se calcula como el número de falsos negativos dividido por el número de valores positivos verdaderos más los valores negativos incorrectamente clasificados. Se expresa mediante la siguiente ecuación:

$$FNR = \frac{FN}{VP + FN} \tag{2.6}$$

PWC (*Percentage of Wrong Classifications*): porcentaje de clasificación errónea. Es una métrica que mide la porción o número de falsos positivos y falsos negativos del total de una clasificación. Es muy usada para reportar comparaciones entre algoritmos debido a que toma en consideración todas las muestras mal clasificadas. Por lo tanto, a menor valor en esta métrica, mejor será el desempeño del algoritmo. Se calcula mediante:

$$PWC = \frac{100*(FN+FP)}{(VP+FN+FP+VN)} \tag{2.7}$$

MCC (*Matthews correlation coefficient*): métrica usada para medir la calidad de dos clases binarias. Devuelve un valor en el rango [-1,1], siendo +1 la predicción ideal, 0 una mala predicción y -1 la segmentación inversa ideal. Se calcula mediante:

$$MMC = \frac{(TP*TN)-(FP*FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{2.8}$$

SSIM (*Structural similarity*): métrica que compara los patrones locales de intensidad de los pixeles que han sido normalizados en luminiscencia y contraste [21]. Medida de calidad de la imagen desde la perspectiva de la formación o estructura de la imagen, se calcula mediante:

$$SSIM(I_X, I_Y) = \frac{(2\mu_X\mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)} \tag{2.9}$$

Donde, I_X e I_Y son las imágenes a comparar, siendo μ_X y μ_Y las medias y σ_X^2 *y* σ_Y^2 las varianzas de I_X e I_Y respectivamente, C_1 y C_2 son dos variables incluidas para evitar la inestabilidad, definidas por:

$$C_1 = (k_1 L)^2, \qquad C_2 = (k_2 L)^2 \tag{2.10}$$

Siendo, *L* el rango dinámico de valores de gris (255 para imágenes de 8 bits) y k_1, k_2 constantes pequeñas con valor menor a 1, comúnmente definidas con valores $k_1=0.01$ y $k2=0.03$.

D-Score: métrica enfocada a medir la calidad de la imagen, considera la localización de los errores en relación a la posición del GT [22]. Es una medida de similitud de imágenes binarias basada en una transformación de distancia, definida mediante:

$$D\text{-}Score(x) = \exp(-(\ln(2.DT(x)) - \alpha)^2) \tag{2.11}$$

Donde, α es un parámetro, el cual, entre más bajo, más cercano a cero será el resultado de la métrica, por lo que experimentalmente fue definido $\alpha=5/2$. *DT(x)* es la transformación de distancia, la cual es dada por:

$$DT(x) = min(d(p,x), \forall x \in X) \tag{2.12}$$

Siendo, *X* el conjunto de pixeles que pertenecen al GT, *d(p,x)* la distancia entre el pixel *p* de la imagen binaria estimada *P* contra el pixel *x* del GT. Por lo que, *DT(x)* es la distancia mínima entre *p* y *x*, para cada pixel $x \in X$.

Las siguientes métricas son usadas cuando el GT representa el modelo de fondo ideal para algoritmos enfocados a modelado de fondo y son reportadas en [23]:

PSNR (*Peak signal-to-noise ratio*): relación señal a ruido de pico. Es una métrica que define la relación entre la máxima energía posible de una señal y el ruido que afecta su representación fidedigna. Su uso más habitual es como medida cuantitativa de la calidad de compresión en imágenes. Usualmente se expresa en decibeles, donde a mayor valor de PSNR, mejor será la estimación de la imagen respecto al GT. Se define mediante:

$$PSNR = 10\log_{10}((M-1)/MSE) \tag{2.13}$$

Donde *M* es el máximo número de nivel de gris y *MSE* es el error cuadrático medio entre el GT y la imagen.

pCEPs (*Percentage of Clustered Error Pixels*): relación de CEPs (*Cluster Error Pixels*) con respecto al número total de pixeles *N* en una imagen. Su valor varía en el rango de 0 a 1. A valores más bajos mejor será la estimación del fondo respecto al GT. Se calcula mediante:

$$\text{pCEPs} = \frac{\text{CEPs}}{N} \tag{2.14}$$

Siendo, CEPs el número de pixeles erróneos cuyos vecinos con conectividad cuatro también son pixeles de error.

pEPs (*Percentage of Error Pixels*): relación de EPs (*Error Pixels*) con respecto al número total de pixeles *N* en una imagen. Los pixeles de error o EPs son definidos como aquellos cuyo valor en el modelo de fondo calculado difiere del modelo de GT. La métrica pEPs toma valores en el rango de [0, 1], a un valor más bajo, mejor será el fondo estimado. Se define mediante:

$$\text{pEPs} = \frac{\text{EPs}}{N} \tag{2.15}$$

AGE (*Average Gray-level Error*): promedio de error de nivel de gris. Es el promedio de la diferencia absoluta de los niveles de gris entre la imagen de referencia I_{GT} y el modelo de fondo calculado I_E. Sus valores se encuentran en el rango de [0, *L*-1], siendo *L* el máximo valor de nivel de gris.

$$\text{AGE} = mean(|\, I_{GT} - I_E \,|) \tag{2.16}$$

CQM (*Color Image Quality Measure*): Medida de calidad en imágenes a color. Esta métrica se basa en el cálculo de PSNR en las bandas únicas del espacio de color YUV, asume valores en decibelios y cuanto mayor es el valor de CQM, mejor es la estimación de fondo. Se define mediante:

$$\text{CQM} = (Y_{PSNR} * 0.9449) + \frac{U_{PSNR} + V_{PSNR}}{2 * 0.0551} \tag{2.17}$$

Donde, Y_{PSNR}, U_{PSNR}, V_{PSNR}, son los valores calculados de PSNR en cada una de las bandas Y, U y V respectivamente.

FSD: métrica reportada en [24] cuyo nombre proviene del acrónimo de las tres métricas F-Measure, SSIM y D-Score. La métrica FSD combina las tres últimas en un valor de peso promediado, mediante:

$$FSD = \frac{F\text{-}Measure + SSIM + (1 - D\text{-}Score)}{3} \tag{2.18}$$

2.3 Bases de datos

Actualmente se encuentran disponibles un número considerable de bases de datos usadas para evaluar el desempeño de los algoritmos de detección de movimiento. Estas bases de datos, cuentan con una serie de videos con diferentes situaciones críticas que muy a menudo se presentan en aplicaciones reales de detección de movimiento, tales como: vibraciones, lluvia, nieve, sombras, fondo dinámico, entre otras. También, algunas bases de datos cuentan con una serie de *Ground truth*, los cuales, aunados a los videos, sirven para evaluar el desempeño del algoritmo.

Existen al menos tres clasificaciones en bases de datos; las enfocadas a detección de movimiento, las de modelado de fondo y las de tracking. Las de detección de movimiento, utilizan un *Ground truth* que representa la imagen binaria de detección ideal, donde se separan los objetos dinámicos del fondo, por lo general estas bases de datos reportan videos de larga duración con una gran cantidad de cuadros (aproximadamente de 500 hasta 8000) y el número de *Ground truth* varía desde uno por video, hasta uno cada cuadro de video. Por su parte, las bases de datos enfocadas en modelado de fondo reportan un *Ground truth* que representa el modelo de fondo deseado, ya no se evalúa la detección de movimiento sino el modelo de fondo obtenido por el algoritmo, normalmente estos videos son de duración más corta (aproximadamente de 6 a 2000 cuadros) y se reporta un *Ground truth* por video. Finalmente, las bases de datos enfocadas a tracking contienen *Ground truth* donde ubican el objeto en movimiento mediante cajas o recuadros que encierran al objeto, estas bases de datos al igual que las bases de datos de detección de movimiento, también cuentan con videos de larga duración y el número de *Ground truth* varía. Sin embargo, debido a la naturaleza de esta tesis, las bases de datos aquí presentadas se enfocan en detección de movimiento y modelado de fondo.

A continuación, en la tabla 2.2 se hace un listado de algunas de las bases de datos vigentes encontradas en la literatura con su respectiva descripción y la cantidad de *Ground truth* que reportan. Las bases de datos marcadas con una estrella pertenecen a bases de datos enfocadas a modelado de fondo.

Tabla 2.2 Bases de datos vigentes a la fecha diciembre del 2017.

Base de datos	Descripción	Ground truth
Wallflower	7 videos cortos con diferentes situaciones críticas. Objetos en movimiento, el cambio de iluminación, fondos dinámicos, camuflaje, etc.	1 GT por cada video.
ATON	11 videos: 4 con cámaras rectilíneas, 2 con cámaras multidireccionales y 5 de detección de sombra.	Únicamente 113 GT en un video de la categoría de detección de sombra
LIMU	8 videos con categorías como cambio de iluminación en una habitación, la parada de autobús en el día y en la tarde, tráfico y otros videos más prestados de PETS.	1 GT cada 15 cuadros por cada video.
PETS 2016	29 videos en diferentes escenarios, enfocados en detección y reconocimiento de patrones de comportamiento humano.	Un paquete de GT [1] por cada video.
UCSD	18 videos con diferentes temáticas (*Birds, Boats, Freeway,* etc.)	Un paquete de GT [1] por cada video.
CITIC RGB-D	4 videos (*Suitcase, Crossing, LCDScreen y LabDoor*)	Un paquete de GT [1] por cada video.
Change detection	53 videos divididos en 11 categorías con 4 o 6 videos cada una	Un paquete de GT [1] por cada video.
SBMnet ★	79 videos divididos en 8 categorías con 5 o 6 videos en cada una	1 GT [2] por video.
SBI ★	14 videos con diferentes temáticas	1 GT [2] por video.
Fish4knowledge	14 videos bajo el agua con 7 categorías diferentes representando diferentes retos complejos	Un paquete de GT [1] por cada video.
MAR	20 videos con tema marítimo para detección, clasificación y tracking	2 GT por video.
SZTAKI	5 videos: de los cuales 2 son tomados de ATON	Un paquete de GT [1] por cada video.
BMC	29 videos divididos en categorías: real, sintéticos y modo de aprendizaje	Un paquete de GT [1] por cada video.
OTCBVS	12 videos con diferentes temáticas, mostrando en su mayoría el espectro no visible infrarrojo.	Un paquete de GT [1] solo en algunos videos.
AVSS	18 videos: 3 videos en la categoría múltiples rostros, 5 para rostro simple, 3 para gente en audiovisual y 7 para detección de vehículos.	Un paquete de GT [1] solo en algunos videos.

[1] No cubierto en su totalidad (no todos los cuadros del video cuentan con GT ó no se encuentran disponibles al público).
[2] El GT corresponde al modelo de fondo deseado.

Últimamente algunos de los links que vinculaban a las bases de datos están caídos o fueron cambiados, por lo que en la tabla 2.3 se presenta la ubicación actual de cada una de las bases de datos vigentes a la fecha diciembre del 2017.

Tabla 2.3 Bases de datos vigentes a la fecha diciembre del 2017.

Base de datos	Localización
Wallflower	https://www.microsoft.com/en-us/research/project/test-images-for-wallflower-paper/
ATON	http://cvrr.ucsd.edu/aton/shadow/
LIMU	http://limu.ait.kyushu-u.ac.jp/dataset/en/
PETS	http://www.cvg.reading.ac.uk/PETS2016/
UCSD	http://www.svcl.ucsd.edu/projects/background_subtraction/
CITIC RGB-D	http://atcproyectos.ugr.es/mvision/index.php?option=com_content&view=article&id=45&Itemid=57
Change detection	http://www. changedetection.net
SBMnet	http://www.scenebackgroundmodeling.net
SBI	http://sbmi2015.na.icar.cnr.it/SBIdataset.html
Fish4knowledge	http://f4k.dieei.unict.it/datasets/bkg_modeling/
MAR	http://labrococo.dis.uniroma1.it/MAR/
SZTAKI	http://web.eee.sztaki.hu/~bcsaba/FgShBenchmark.htm
BMC	http://bmc.iut-auvergne.com
OTCBVS	http://vcipl-okstate.org/pbvs/bench/
AVSS	http://www.eecs.qmul.ac.uk/~andrea/avss2007_d.html

De las bases de datos mostradas anteriormente solo algunas proporcionan en el mismo sitio los resultados obtenidos de los algoritmos con base en sus métricas de evaluación de desempeño y ofrecen una tabla de posiciones para ubicar los algoritmos que mejor pueden enfrentar un número de situaciones críticas, dos de ellas enfocadas en detección de movimiento son *Change detection* y *BMC*. Mientras que para modelado de fondo dos de las bases de datos que ofrecen evaluaciones son *SBI* y *SBMnet*. Estas cuatro bases de datos serán analizadas y utilizadas en el capítulo cuatro como parte de la metodología usada para la selección de los mejores algoritmos.

CAPÍTULO III. ESTADO DEL ARTE ENFOCADO A DETECCIÓN DE MOVIMIENTO

En los últimos años se han desarrollado una gran cantidad de algoritmos enfocados en detección de movimiento, cada uno, con algún rasgo peculiar y con distintas características que buscan resolver diferentes situaciones críticas en video, como fondos dinámicos, vibraciones de cámara, cambios de iluminación, camuflaje, entre otras. Sin embargo, debido a la variedad de situaciones críticas presentes en la vida real, es difícil conseguir un algoritmo que cubra todos los problemas en su totalidad, por lo que muy a menudo se tiende a discernir qué algoritmo es mejor para cada circunstancia. Con la finalidad de destacar las características que poseen los diferentes enfoques de algoritmos de detección de movimiento, se realizó una búsqueda en la literatura considerando publicaciones del año 2013 a la actualidad. En total se encontraron 47 algoritmos, los cuales son presentados y analizados en este capítulo; se detallan sus características más relevantes y se propone una clasificación completa con base en la técnica que utilizan para realizar la detección de movimiento.

3.1 Algoritmos y técnicas utilizadas para la detección de movimiento

Para el desarrollo de la clasificación de los 47 algoritmos encontrados en la literatura, se tomó como referencia la clasificación propuesta por Thierry Bowmans en [6], en ella, se presentan más de 100 algoritmos, los cuales son divididos en dos categorías principales: los modelos tradicionales y los recientes, además, se proporcionan sub-categorías donde se indica qué tipo de técnica específica utilizan los algoritmos. Los modelos tradicionales son descritos como modelos básicos que permiten resolver algunas situaciones reducidas y generalmente son fáciles de implementar. Por su parte, los modelos recientes presentan características más sofisticadas para manejar de una manera robusta una gran cantidad de cambios en las escenas, sin embargo, la mayoría de ellos requiere de mejoras para alcanzar un procesamiento en tiempo real. La clasificación de Bouwmans, debido a que engloba una cantidad de técnicas y algoritmos, es una de las más completas encontradas en la literatura, por lo que, la clasificación propuesta en este capítulo se basa en ella.

En la Tabla 3.1 se muestra la clasificación propuesta de los 47 algoritmos encontrados en la literatura. En esta, se pueden observar que la columna “categorías” engloba de una manera

general el método utilizado para la detección de movimiento y las sub-categorías muestran de una manera más específica la técnica utilizada.

Tabla 3.1 Clasificación por tipo de técnica de los 47 algoritmos analizados.

CLASIFICACIÓN POR TIPO DE TECNICA			
TOTAL	CATERGORÍAS	TÉCNICAS	ARTÍCULOS
5	Básicos	Modelos basados en promedio	[25], [12]
		Modelos basados en mediana	[10], [26]
		Diferencia de cuadro	[27]
2	Combinados	Combina las características de los modelos Gaussianos, las redes SOM y un método que mide cambios de iluminación	[28]
		Combina Multi-view learning, Markov random field (MRF) y filtro de mediana temporal	[29]
2	Estadísticos	Modelo de mezclas Gaussianas auto-adaptivas (GMM)	[30]
		Kernel Density Estimation	[31]
11	Redes Neuronales	*Discrete Time Cellular Neural Network* (DTCNN)	[32]
		Self Organizing Maps(SOM)	[33], [34], [35], [36]
		Neural Response Mixture (NeRM)	[37]
		Convolutional Neural Network (CNN)	[38]
		Restricted Boltzmann Machine	[39]
		Weightless Neural Networks	[40]
		Hopfield-type neural networks	[41]
		Radial Basis Function Based	[42]
9	Modelos Estadísticos Avanzados	Modelo no paramétrico LBP (*Local Binary Patterns*)	[43]
		Modelos no paramétricos mediante ViBe (*Visual Background Extractor*)	[44], [45]
		Mejora al modelo no paramétrico SURF (*Speeded Up Robust Features*)	[46]
		Modelo no paramétrico basado en Sift flow	[47]
		Aproximación estocástica	[48]
		Modelo de aprendizaje compartido basado en GMM	[49]
		Modelo de mezcla de Gaussianas globalmente espaciadas e incertidumbres (SGGMM)	[50]
		SuBSENSE (*Self-Balanced SENsitivity SEgmenter*)	[51]
2	Modelos Difusos	Correlograma de *Kernel* difuso (KFC)	[52]
		Diferencia de histograma de color difuso (FCDH)	[53]
6	Modelos de subespacios	Método basado en RPCA (*Robust Principal Component Analysis*)	[54]
		Detección de movimiento mediante COROLA (*Contiguous Outliers Representation via Online Low Rank Approximation*)	[55]
		Modelo basado en descomposición ortogonal SVD (*Singular Value Descomposition*)	[56]
		Detección de movimiento mediante TVRPCA (*Total Variation Regualrized RPCA*)	[57]
		Graph cut y modelos Gaussianos	[58]
		Descomposición basada en Superpixeles	[59]
4	Diccionarios de aprendizaje	PAWCS (*Pixel-based- Adaptative Word Consensus*)	[60]
		BMTDL (*Background Modeling Through Dictionary Learning*)	[61]
		BREW-DLHPM (*Background REpre- sentation approach With Dictionary Learning and Historical Pixel Maintenance*)	[62]
		Modelo basado en codificación esparcida K-SVD	[63]
2	Modelos de transformación de dominio	Modelo basado en Wavelet BWM (*Background Wavelet Model*)	[64]
		Enfoque basado en memoria de diferencia de cuadros MFD y Wavelet	[65]
2	Modelos tensores robustos	Tensor de flujo con mezclas Gaussianas particionadas FTSG (*Flux Tensor with Split Gaussian models*)	[66]
		Tensor bayesiano robusto por medio de factorización BRTF (*Bayesian Robust Tensor Factorization*)	[67]
2	Otros	Modelo basado en programación genética	[68]
		Modelo basado en CTU (*Coding Tree Unit*)	[69]

A continuación, se da una descripción de cada una de las categorías mostradas en la tabla 3.1 y se resume la información más relevante de cada uno de los diferentes enfoques.

3.1.1 Modelos básicos

Son algoritmos que usan técnicas pioneras en el campo de detección de movimiento, son fáciles de implementar y actualmente siguen siendo muy utilizados debido a que no requieren una gran cantidad de parámetros y su consumo de recursos es bajo.

3.1.1.1 Modelos basados en el filtro de promedio

Utilizan la información temporal del video, tomando en consideración un número *n* de cuadros posteriores al actual, luego los *n* cuadros se promedian para generar una estimación del fondo, una vez obtenido la estimación del fondo se realiza una comparación contra el cuadro actual del video y mediante un umbral de binarización se logra detectar el movimiento. El principio de detección de movimiento se basa en que al realizar el promediado de los *n* cuadros posteriores, el fondo estático permanecerá sin sufrir cambios y por su parte los objetos en movimiento, debido a la naturaleza de su comportamiento no determinista, serán atenuados o eliminados de la estimación del modelo del fondo. También, existe una adaptación a este método llamada aproximación por filtro de promedio que permite realizar una estimación del modelo de fondo, únicamente conociendo el cuadro posterior y el actual. Estos dos métodos fueron previamente descritos en la sección 1.1.1.1 y 1.1.1.2. A continuación, se describen los métodos dentro de esta clasificación:

El algoritmo de aproximación por filtro de promedio, es relativamente acertado para modelar el fondo cuando la iluminación es constante. Sin embargo, en existencia de cambios repentinos de iluminación o fondos dinámicos, resulta en un alto número de falsos positivos. El problema principal radica en que se asume un parámetro de aprendizaje igual para todos los pixeles de la imagen. Sin embargo, en zonas con cambios de iluminación repentina o fondos dinámicos, el parámetro de aprendizaje debería ser mayor que el de los pixeles donde existan variaciones poco significativas. En [12] se propone una adaptación al método de aproximación por filtro de promedio en la cual el parámetro de aprendizaje es diferente para cada pixel. Este parámetro de aprendizaje también es usado para variar el umbral de binarización en el proceso de detección de movimiento.

En [25] se presenta un método que busca también hacer frente a los problemas debidos a cambios de iluminación. Se usan dos modelos de fondo con velocidades de adaptación distinta; baja y alta respectivamente. Los dos modelos de fondo son generados mediante el algoritmo de aproximación por filtro de promedio, variando únicamente el parámetro de adaptación para ambos modelos, mediante un parámetro llamado ganancia de compensación, que se ajusta dependiendo del grado de cambio de iluminación. La ganancia de compensación se calcula utilizando los cambios en los promedios de luminancia entre los modelos de fondo previos y los fondos actuales. La máscara binaria final, es obtenida mediante la multiplicación de las dos máscaras binarias, generadas por los modelos de fondo con baja y alta velocidad de adaptación. La velocidad de procesamiento del algoritmo muestra un bajo costo computacional siendo ideal para implementación en hardware.

3.1.1.2 Modelos basados en el filtro de mediana

Al igual que los modelos basados en promedio, los de mediana utilizan la información temporal del video, tomando en consideración un número *n* de cuadros posteriores al actual, luego se obtiene la mediana de los *n* cuadros y se genera la estimación del fondo. Existe también una adaptación llamada aproximación por filtro de mediana, que permite realizar la estimación del modelo de fondo, mediante la comparación del cuadro actual de la imagen, contra el cuadro del fondo previamente modelado. Ambos modelos fueron abordados previamente en las secciones 1.1.1.2-3. A continuación, se describen los métodos dentro de esta clasificación:

En [10] se propone un método de estimación de fondo y segmentación de objetos en primer plano basada en el uso del filtro de mediana, sobre un número consecutivo y limitado de cuadros del video. Este método cuenta con la desventaja de requerir ajuste de parámetros al cambiar de video y su uso es limitado a videos simples.

Con el propósito de resolver los problemas presentes en entornos con cambios de iluminación en [26] se propone un método basado en la información espacial de color RGB. Primero, se usa filtro de mediana como algoritmo de inicialización del modelo de fondo mediante los primeros 100 cuadros de video. Luego, se calcula la diferencia de la imagen del modelo de fondo realizada con el filtro de mediana y de la imagen actual. Después, una operación de umbral es aplicada para decidir si un pixel pertenece al fondo o al objeto en

movimiento. La mayoría de los algoritmos seleccionan el umbral mediante prueba y error, pero al tener un umbral estático para todos los pixeles de la imagen, algunos pixeles de objetos dinámicos son incorrectamente clasificados. Para resolver este problema se propone un umbral adaptativo. Este umbral será distinto para cada pixel en cada canal del espacio de color RGB y variará de cero a uno. Finalmente, Se calcula una máscara binaria para cada uno de los colores a través de la comparación del umbral para cada pixel, si el valor del umbral es aproximadamente uno, el pixel será tomado como objeto en movimiento de lo contrario será tomado como fondo. Las tres máscaras binarias de cada color son combinadas en una mediante una operación lógica OR. Los resultados experimentales del método muestran buenos resultados en problemas debidos a cambios de iluminación.

3.1.1.3 Modelo basado en cuadro de diferencia

El algoritmo propuesto en [27] se basa en la aplicación del método *Frame Difference Model* (FDM) y un método llamado *Background Difference Model* (BDM). En FDM se calcula la diferencia de los cuadros previos y el actual. Si el valor de los pixeles es mayor o igual a un umbral entonces serán considerados como objetos en movimiento, si el valor es menor entonces los pixeles serán considerados como parte del fondo. Por otro lado, BDM cuenta para cada pixel de la imagen actual, el número de veces que han sido considerados como fondo en *n* previas máscaras binarias almacenadas. Al inicio, el pixel es considerado objeto en movimiento, pero si la cuenta llega a ser igual o mayor que el de una constante de umbral, será considerado como parte del fondo. La máscara binaria de detección de movimiento es generada mediante una comparación del valor absoluto del cuadro actual menos el modelo de fondo para luego ser pasada por un umbral de binarización. El algoritmo fue implementado en secuencias de video en escenarios externos e internos basados en video vigilancia.

3.1.2 Modelos combinados

Estos modelos son una combinación de dos o más técnicas de detección de movimiento y debido a que cada técnica se le da un peso similar, resulta difícil dar una clasificación específica respecto a su técnica. Estos modelos, aprovechan los rasgos más sobresalientes de las técnicas ya existentes para generar métodos compuestos con mejores características.

3.1.2.1 Modelos gaussianos, redes auto-organizadas y método sensitivo a cambios de iluminación

Combinando características de algoritmos de modelado de fondo bien establecidos como el modelo de mezcla de Gaussianas, redes auto organizadas (SOM por sus siglas en inglés) y métodos sensitivos a la iluminación, el método presentado en [28] es capaz de manejar de una forma eficiente cambios de iluminación repentinos de fuentes de iluminación ya sea naturales o artificiales, las cuales muy probablemente causarían detecciones incorrectas en otros algoritmos de modelado fondo. El método maneja los cambios de iluminación mediante un algoritmo recursivo, donde se inicializa el modelo de fondo usando los primeros *n* cuadros del video, después, se verifica si hubo un cambio de iluminación. Si ocurrió, se actualiza y se agregan valores al modelo de fondo para corregir la segmentación del cuadro actual. Si no existió cambio de iluminación, no se toma acción y simplemente se pasa a evaluar el siguiente cuadro del video. Una ventaja significativa de esta propuesta es el hecho que no requiere ninguna inicialización o entrenamiento complejo. La información que se utiliza en el algoritmo son los valores de color RGB puros, sin ninguna información de uso temporal. Así, el método propuesto consigue una mayor precisión y mantiene una baja complejidad. Sin embargo, no alcanza un procesamiento en tiempo real para videos con resolución de 320x240.

3.1.2.2 Aprendizaje de múltiples vistas, filtro de mediana y MRF (*Markov Random Field*)

Esta propuesta [29], combina las características del filtro de mediana, la técnica *Markov Random Field* (MRF) y aprendizaje por múltiples vistas. El algoritmo se basa en tres principales pasos. Primero una imagen de fondo de referencia es generada por filtro de mediana temporal, donde múltiples características heterogéneas incluyendo variación de brillo, de cromaticidad y textura son extraídas de la secuencia de video. Luego, un aprendizaje con múltiples vistas es ideado para estimar la probabilidad condicional para ambos modelos; de fondo y objetos en primer plano. Finalmente, la técnica *Markov Random Field* es usada para incorporar el contexto del espacio temporal a la decisión de fondo u objeto en primer plano. El método propuesto fue implementado usando el software MATLAB y el tiempo de procesamiento en promedio fue de 4.3 segundos por cuadro en un video con resolución de 320x240. Las evaluaciones verifican la efectividad del método

propuesto para manejar sombras, cambios de iluminación y ruido suave. Sin embargo, en situaciones como movimientos repentinos de cámara y fondos dinámicos, se obtiene un bajo desempeño.

3.1.3 Modelos estadísticos

Los modelos estadísticos son una forma matemáticamente formalizada de realizar aproximaciones y hacer predicciones a partir dichas aproximaciones. Se dividen en modelos paramétricos y no paramétricos. Los modelos paramétricos, hacen uso de funciones de densidad de probabilidad con ciertos parámetros, variando estos parámetros se obtienen diferentes funciones, con distintas varianzas o tamaños que sirven para dar una aproximación al modelo de fondo. Mientras que los modelos no paramétricos, son modelos estadísticos cuya distribución subyacente no se ajusta a una distribución de probabilidad conocida, tal como la distribución normal, uniforme, exponencial, logarítmica, entre otras. Los modelos paramétricos y no paramétricos fueron abordados previamente en la sección 1.1.2 y 1.1.3.

3.1.3.1 Modelo paramétrico basado en mezcla de Gaussianas

Un modelo estadístico paramétrico es presentado en [30], es un algoritmo auto-adaptativo basado en un modelo de mezclas Gaussianas (GMM). El algoritmo ajusta sus parámetros de acuerdo a las características del escenario. El método auto-adaptivo es llamado SAG (*self-adaptative* GMM) y combina dos procesos: a nivel cuadro y a nivel pixel. Es basado en el modelo de mezclas Gaussianas propuesto por Zivkovic y Van Der Heijden [70], y el método *multi- dimensional Gaussian kernel density transform* (MDGKT) [71].

El método es compuesto de dos módulos principales. El primero modela el fondo mediante los siguientes pasos: se suprime el ruido de la imagen usando el filtro espacio-temporal MDGKT, luego un factor *g* de cambio de iluminación global es calculado entre el cuadro actual y el fondo de referencia, después el método SAG es aplicado. El segundo módulo detecta los objetos en movimiento contra el fondo, suprimiendo sombras y reflejos para luego rellenar espacios vacíos en la silueta binaria usando operaciones morfológicas. El factor *g* permite dar un seguimiento a los cambios de iluminación en la imagen, este será luego usado para calcular el mejor modelo de fondo aplicado en un GMM y también para escalar el brillo y la distorsión de cromaticidad en situaciones de sombras y cambios de iluminación.

El algoritmo fue implementado en MATLAB y mostró un desempeño de 8.3 cuadros por segundo y el algoritmo realizado en C++ tuvo un desempeño en promedio de 25 a 30 cuadros por segundo esto quiere decir que la implementación en C++ fue 10 veces más rápida que en MATLAB, excluyendo las operaciones morfológicas.

Una de las ventajas de este algoritmo es que puede adaptarse automáticamente a los cambios de iluminación ya que cuenta con un factor de aprendizaje que modela los cambios de iluminación global del escenario cuadro a cuadro. Los autores del método, son unos de los pocos que realizan las comparaciones de dos implementaciones en código distintas mostrando sus resultados para C++ y MATLAB.

3.1.3.2 Modelo no paramétrico mediante *textons* y *kernel density estimation*

En [31] se presenta un método que modela el fondo y los objetos dinámicos por medio de *textons*. Los *textons*, son elementos que describen texturas en pequeñas regiones con bajo contraste. En detalle, por cada pixel *P* se construye un vector de características que consiste en las coordenadas espaciales del pixel *(x, y)*, los tres canales de color (HSV, RGB ó Lab) y la energía del *texton* en una región *WxW* (5x5) centrada en el pixel *P*. Más específicamente, primero se extrae la magnitud del máximo gradiente de la imagen en la región *WxW* mediante el cálculo del vector propio correspondiente al valor propio mayor de la matriz Jacobina del vector del espacio de color. Luego, es obtenido el *texton.* En este método se propone el uso de 7 *textons* por la necesidad de capturar incluso las más pequeñas variaciones de textura. Una vez obtenida la imagen de textura en la región *WxW* se calcula la energía, que es usada junto con las coordenadas espaciales del pixel *(x, y)* y los tres canales de color (HSV, RGB ó Lab) para definir un vector de 6 características. Después, se obtienen las probabilidades de fondo y de objeto en movimiento mediante el algoritmo *kernel density estimation* usando el espacio de 6 características para posteriormente formar la máscara binaria.

En general este método propone construir el modelo de fondo y el modelo de objetos dinámicos a través de la integración de características de color y textura de la imagen. Para este método el espacio de color RGB demostró ser más adecuado para modelar variaciones de color que los espacios Lab o HSV, lo cual resulta contrario a lo reportado en [72]. Como trabajo futuro se piensa reducir el costo computacional debido a que para videos con

resolución de 320x240 pixeles se tuvo un tiempo de procesamiento a 18.5 cuadros por segundo en un CPU Intel i7 con 16 GB de RAM, implementado en código C++.

3.1.4 Modelos basados en redes neuronales

Son modelos matemáticos inspirados en el comportamiento de las redes neuronales biológicas. Debido a su naturaleza y fundamentos, las redes neuronales artificiales presentan un gran número de características semejantes a las del cerebro humano. Por ejemplo, son capaces de aprender de la experiencia, de generalizar, de extraer características a partir de entradas que representan información irrelevante, entre otras. Esto hace que ofrezcan numerosas ventajas y que este tipo de tecnología se aplique en múltiples áreas. A continuación, se describen algunos de los modelos encontrados en la literatura basados en redes neuronales enfocados en detección de movimiento.

3.1.4.1 Modelo *Discrete Time Cellular Neural Network* (DTCNN)

En [32] se presenta un modelo de segmentación para objetos dinámicos y estáticos. El método propuesto involucra tres principales bloques para la realización de la segmentación dinámica: registro de fondo dinámico, detección de objetos dinámicos y mejoras a la segmentación de objetos. Los tres bloques antes mencionados, son descritos a continuación:

El bloque de registro de fondo dinámico considera el primer cuadro del video como el fondo actual, después, un proceso de actualización comienza a incorporar pequeñas variaciones al fondo. Para distinguir variaciones entre los valores de pixeles se toma la distancia euclidiana de la imagen anterior y la actual en espacio de color HSV. Si el valor de la distancia euclidiana de algún pixel es menor a un umbral se le considera como pixel de no cambio y es registrado en una matriz de acumuladores. Cada elemento de la matriz indica el número de cuadros en los que un pixel ha permanecido sin cambio. La actualización de fondo se da cuando el valor del acumulador de un cierto pixel llega a un límite, entonces ese pixel será tomado como fondo debido a que ha permanecido sin alteración.

El segundo bloque, de detección de objetos dinámicos, se realiza por medio de una red neuronal *Discrete Time Cellular Neuronal Network* (DTCNN). Del bloque previo, una vez que el fondo ha sido actualizado, el siguiente paso es encontrar los objetos que están en posible movimiento. Los pixeles de objetos dinámicos son detectados mediante la distancia

Euclidiana $Ed_2(x,y,t)$ del cuadro de fondo y el cuadro actual. Esta distancia euclidiana es mapeada dentro del rango [-1,1] de esta manera puede ser usada para alimentar las entradas del modelo DTCNN y finalmente se usa una operación de umbral de peso ponderado para determinar qué pixeles son candidatos para ser objetos en movimiento, el resultado de la operación recae en dos posibles clases de pixeles que corresponden a objetos estáticos O_S o dinámicos O_D, mediante:

$$\overline{Ed_2}(x,y,t) : DTCNN \Rightarrow \begin{cases} O_S & \text{si} \quad \overline{Ed_2}(x,y,t) < Z_{ij} \\ O_D & \text{si} \quad \overline{Ed_2}(x,y,t) \geq Z_{ij} \end{cases} \tag{3.1}$$

Donde $\overline{Ed_2}(x,y,t)$ es el peso ponderado del valor de 8 pixeles vecinos de $Ed_2(x,y,t)$ y Z_{ij} es el umbral de clasificación.

Finalmente, en el tercer bloque, se realizan mejoras a la segmentación de objetos dinámicos mediante operaciones morfológicas de dilatación y erosión realizadas con redes DTCNN.

Una ventaja notoria del método es que no requiere ajuste manual de sus parámetros para procesar diferentes secuencias de video. Sin embargo, no fue comparado contra otros algoritmos y fue evaluado únicamente con 5 videos, por lo que su desempeño está limitado al número de experimentos que se reportaron en el artículo.

3.1.4.2 *Self Organizing Maps* (SOM)

El enfoque [33] presenta un nuevo sistema de modelado de fondo basado en dos mapas auto organizados (SOM por sus siglas en inglés) que se adaptan en forma paralela a diferentes velocidades. El algoritmo con arquitectura paralela auto-adaptiva es llamado AAPSA. Este puede identificar automáticamente el problema principal que afecta al escenario y a la detección de los objetos en movimiento, tal como: fondos estáticos o dinámicos, objetos dinámicos estacionarios, vibraciones de cámara, camuflaje, etc. La identificación es lograda mediante cuatro módulos diferentes. Los módulos son clasificados en: escenas con fondos estáticos, escenas normales, escenas dinámicas y escenas complejas.

AAPSA fue validado con la base de datos *Change Detection* y *BMC* usando los mismos parámetros iniciales en el modelo, demostrando su robustez con diferentes y complicados escenarios. Es el primer algoritmo reportado en la literatura que usa los mismos parámetros iniciales en dos diferentes bases de datos.

Otro enfoque que hace uso de redes SOM, es mostrado en [34]. El método es llamado *Retinotopic* SOM (RESOM) y se basa en una red neuronal inspirada en el mecanismo de la corteza visual humana. RESOM puede adaptar sus parámetros de aprendizaje basándose en el comportamiento del escenario, por lo que esta red tiene la ventaja de imitar la habilidad de percepción. Cuenta con dos capas, la primera capa se compone de los receptores de la retina que en este caso reciben el cuadro actual del video, y la segunda capa llamada V1 se constituye de un modelo de mapa topográfico llamado SOM *Retinotopic Map* (SOMRM).

SOMRM es un modelo que detalla cómo se realizan las interacciones retinotópicas en la corteza visual. Interacción retinotópica se refiere a la manera en cómo cada neurona tiene un peso propio conectado con los receptores de la retina. En otras palabras, SOMRM es un modelo neuronal usado para describir la corteza visual primaria como un mapa topográfico que forma una capa cortical, donde cada neurona gradualmente llega a ser selectiva a una determinada característica de entradas simultaneas.

Los pesos de las neuronas son actualizados en cada iteración con un esquema de regla de aprendizaje *Hebbiana*. La competencia de las neuronas es realizada mediante *winner takes all*, encontrando el máximo valor de cada producto punto entre el mapeo de pesos y la imagen de entrada.

Los resultados experimentales del método, muestran que el modelo se adapta adecuadamente para manejar escenarios con cambios de iluminación y fondos dinámicos. El modelo no requiere entrenamiento o ajuste manual de parámetros ya que automáticamente determina sus parámetros y es un método no supervisado, esto permite al modelo la habilidad de adaptarse a diferentes condiciones en los escenarios de video. Sin embargo, el método aun presenta problemas debido al camuflaje y a no poder procesar en tiempo real videos de resolución mayor a 128x160 pixeles.

Otro método bio-inspirado llamado SOM-CNN es propuesto en [35], se basa en los métodos *Self-Organized Maps* (SOM) y *Cellular Neural Networks* (CNN) para realizar la detección de objetos en movimiento en escenarios normales y complejos. El método es una mejora a la arquitectura RESOM [34] descrita anteriormente, la cual, es inspirada en el mecanismo de percepción de la corteza visual humana. También, se propone un esquema novedoso de umbralización llamado *Neighbor Threshold CNN* (NTCNN). El esquema general del SOM-CNN se compone de 4 bloques. En el primer bloque, se realiza la conversión de la imagen de entrada o actual del video en RGB, al espacio de color HSV, se elige el espacio HSV debido a que es posible obtener mejores resultados en los escenarios con fondos dinámicos y también para resolver situaciones de camuflaje. El segundo bloque, realiza un modelo de fondo mediante el método RESOM, la salida de esta etapa es la distancia euclidiana entre el cuadro actual y el modelo del fondo RESOM. El tercer bloque contiene un módulo NTCNN que es una red neuronal iterativa que elimina ruido y clasifica la distancia euclidiana obtenida del bloque anterior en fondo u objeto dinámico, por lo tanto, la salida de este bloque será la máscara binaria de la detección de movimiento. Finalmente, el último bloque es diseñado para ajustar los parámetros de las redes neuronales de los dos bloques anteriores, permitiendo una adaptación correcta del método ante distintos escenarios.

El método propuesto conserva las ventajas del ya visto algoritmo RESOM y se le añaden otras, en general puede manejar situaciones con cambios graduales y repentinos de iluminación, fondos dinámicos, camuflaje, vibraciones de cámara y objetos dinámicos detenidos. SOM-CNN trabaja mejor que el RESOM reduciendo el camuflaje, sin embargo, el problema se continúa presentando en videos con baja iluminación y contraste.

El último método que entra dentro de la categoría de redes SOM, es llamado *Stacked Multilayer Self-Organizing Map Background Model* (SMSOM-BM), se presenta en [36]. SMSOM-BM consiste en una capa de entrada y varias capas computacionales de salida para cada pixel de la imagen. El número de capas computacionales se puede elegir arbitrariamente. Para cada capa computacional, los pixeles de entrada se asocian con 9 nodos computacionales en esa capa. La entrada a la red corresponde a los tres valores de los canales del espacio de color HSV de cada uno de los pixeles. El proceso de modelar cada pixel con varias capas interconectadas con nueve nodos en cada una es llamado SMSOM y SMSOM-BM

corresponde al modelo de fondo completo que consta de todos los pixeles de la imagen. El principio del funcionamiento del algoritmo se basa en entrenarlo con diferentes imágenes de fondo, luego dividir los datos de entrenamiento en diferentes grupos, donde cada capa se utiliza para modelar una parte de las características para el fondo y la información jerárquica es capturada por un mecanismo de entrenamiento en capas. Además, los nodos computacionales en la misma capa cooperan entre sí para modelar las restricciones espaciales de las imágenes.

Como se mencionó, SMSOM-BM debe ser entrenado usando algunas imágenes de fondo primero, y luego gracias a este pre-entrenamiento cuando un cuadro nuevo llega, la detección de movimiento puede ser realizada. Para lograr que el algoritmo se adapte a los cambios de escenario durante la detección de movimiento, el modelo de fondo se actualiza usando cuadros de prueba *on-line*, este proceso es llamado mantenimiento de fondo.

Para su implementación se usó un CPU Intel con 6GB de RAM y una GPU NIVIDIA GeForce GTX 260, el algoritmo presenta un buen desempeño en tiempo real para la mayoría de los escenarios excepto aquellos casos con una resolución de imagen mayor o igual a 720x480. En cuanto a su desempeño, debido a que SMSOM-BM necesita imágenes de entrenamiento para ajustar un parámetro τ de aprendizaje del umbral y en algunos videos no se tiene imágenes de entrenamiento, el parámetro τ fue inicializado en 0.06 para la detección de movimiento. Sin embargo, debido a la falta del proceso de entrenamiento, no se puede proveer una buena estimación inicial, lo cual hace que el desempeño de SMSOM-BM decrezca.

3.1.4.3 *Neural Response Mixture* (NeRM)

En [37] se propone un nuevo método de detección de movimiento llamado *Neural Response Mixture* (NeRM) el cual, aprovecha las capacidades de las redes neuronales profundas (DNNs) estocásticas, manteniendo un bajo costo computacional para su aplicación en sistemas embebidos. En NeRM se extraen características para construir un modelo de mezcla de Gaussianas (GMM) de respuesta neuronal mixta, como fondo confiable que capture rasgos únicos de los objetos (personas, vehículos, animales, etc.) en entornos de vigilancia reales. Por lo que, se requiere un extenso entrenamiento a través de diferentes escenarios como cambios de iluminación, condiciones climatológicas (lluvia, nieve, granizo,

etc.), vibraciones de cámara, entre otros. Es necesario contar con la segmentación respectiva correspondiente a objetos en movimiento y fondo. La respuesta neuronal de las DNNs estocásticas provee características que describen los rasgos únicos de los objetos. Finalmente, el movimiento es detectado evaluando la probabilidad de pertenecer al fondo u objeto en movimiento basándose en el modelo GMM de respuesta neuronal construido.

Los resultados experimentales del método muestran que el algoritmo posee un buen desempeño, sobre todo en videos con escenarios complejos como mal clima, sombras y turbulencias. El modelo NeRM fue implementado en la tarjeta codificadora de vídeo AXIS Q7436. Los resultados experimentales mostraron que tomó alrededor de 470ms para procesar un cuadro de video de 352x240, lo anterior resulta en una velocidad de procesamiento cercana a dos cuadros por segundo, abriendo la posibilidad del uso de redes neuronales profundas en sistemas embebidos. Una de las desventajas de este método radica en el hecho de que requiere un extenso conjunto de entrenamiento, y el obtener una gran cantidad de videos para entrenar las DNNs con sus respectivos *Ground truth* es una tarea laboriosa y complicada.

3.1.4.3 *Convolutional Neural Network* (CNN)

En [38] se propone una mejora para el modelado de fondo a través de un algoritmo basado en características espaciales aprendidas con redes neuronales convolucionales. El modelo de fondo se realiza con una imagen en escala de gris que luego es usada para entrenar la red con un conjunto de datos específicos en el escenario. Este método es el primer intento por aplicar una red neuronal convolucionada para el modelado de fondo. Se compone principalmente por cuatro etapas: extracción de fondo de la imagen, generación del conjunto de datos, entrenamiento de la red y sustracción de fondo.

En la etapa de extracción de fondo se convierten las imágenes en espacio de color RGB al dominio de escala de gris, y se obtiene una imagen de fondo tras tomar los primeros 150 cuadros del video y realizar un filtro de mediana temporal.

Durante la generación del conjunto de datos específicos de escena que aprenderá la red se toman en consideración dos técnicas: generación automática e integración de etiquetas por personas expertas.

El entrenamiento de la red se logra mediante una arquitectura neural muy similar a la red LeNet-5 [73] usada para la clasificación de dígitos. La red utiliza campos receptivos de tamaño 5x5 con un paso de 1x1 para todas las capas convolucionales y un campo receptivo no superpuesto de 3x3 para todas las capas de agrupación. La primera capa convolucional tiene 6 mapas de características, y la segunda 16. La capa totalmente conectada tiene 120 unidades ocultas y la salida consta de una sigmoide. La red tiene un total de 20,243 pesos entrenados mediante el algoritmo de *Backpropagation* con una función de error de entropía cruzada. Las bias o polarizaciones son inicializadas en 0.1 mientras que los otros pesos son inicializados de manera aleatoria. Se usa una estrategia *RMSProp-optimization* para asegurar una rápida convergencia. Finalmente, la fase de entrenamiento es detenida después de las 10000 iteraciones. Los resultados experimentales del método muestran que el algoritmo tiene buenos resultados al manejar los problemas debido a sombras y videos nocturnos.

3.1.4.4 Máquinas de Boltzmann Restringidas Parcialmente-esparcidas

Las máquinas de Boltzmann o *Restricted Boltzmann Machine* (RBM) han sido ampliamente utilizadas para aprendizaje no supervisado. En [39] se propone un método de modelado de fondo a través del uso de una RBM. La máquina de Boltzmann restringida es una red neuronal de dos capas con estructura bipartida, donde la primera capa es definida como la capa visible, en la cual se usa la intensidad del pixel de la imagen, y la segunda capa es llamada oculta y funciona como un detector de características. En esta última todas las neuronas están totalmente conectadas. En la práctica RBM itera para actualizar los pesos durante el proceso de entrenamiento para minimizar la función de energía. Una vez que el entrenamiento es alimentado a la red RBM, la imagen es reconstruida a través de la combinación de los pesos conectados entre las capas de la red.

La idea general del funcionamiento del algoritmo es primero separar los datos de entrenamiento en diez grupos, luego cada grupo contendrá 50 imágenes. La imagen de entrada es reconstruida mediante RBM en cada grupo, luego las 10 salidas de cada uno de los grupos son almacenadas en una pila, donde las variaciones de la intensidad de los pixeles son calculadas a lo largo de la dirección de profundidad de la pila para formar un mapeo basado en variación. El mapeo anterior, es binarizado para formar una máscara de segmentación *SM* mediante el promedio de la variación de intensidad como umbral de corte.

La máscara *SM* distingue a los pixeles que contienen más variación. La comparación de la imagen de entrada con la de salida filtradas por la máscara *SM* generada, produce una imagen final que contendrá los objetos en movimiento.

Los experimentos del algoritmo muestran los mejores desempeños en categorías de video con fondos dinámicos, intermitentes e imágenes térmicas. El método tiene la ventaja de lograr una buena detección en diferentes videos sin tener que especificar configuraciones diferentes de los parámetros para cada caso. Existe potencial para poder mejorar la detección a través de incrementar las capas ocultas o ajustar el máximo de épocas. Sin embargo, una de las principales desventajas de RBM es que su desempeño es proporcional al número de capas ocultas, debido a que cada capa oculta representa una característica prototipo, la pregunta sería; ¿cuántas capas ocultas son suficientes para generalizar un sistema de vigilancia enfocado a detección de movimiento?.

3.1.4.5 Modelado de fondo por *Weightless Neural Networks*

Las *Weightless Neural Networks* (WNN) están basadas en redes de nodos de acceso a memoria aleatorio (RAM por sus siglas en inglés). El método propuesto de modelado de fondo en [40] es llamado BGWiS y explota las características de una red neuronal WiSARD en su núcleo, que es un tipo particular de WNN. WiSARD consiste en un modelo neuronal con un conjunto de neuronas RAM, las cuales almacenan información de la ocurrencia de patrones binarios de tamaño fijo durante el periodo de aprendizaje. El entrenamiento se realiza con un conjunto de datos representativos de cada clase o categoría.

El algoritmo es basado en las siguientes asunciones. El color del pixel (en cualquier espacio de color entre RGB, HSV y Lab) es representado por tres canales teniendo los valores de los pixeles en el rango de 0-255. A manera de usar WiSARD como el núcleo del modelo de fondo, una binarización de los pixeles es requerida. Los tres canales son escalados y discretizados en el rango 0,1…, W-1, lo cual representa un color con un patrón binario de tamaño 3xW. El valor de W puede ser variado. El patrón binario representará una imagen que alimentará la entrada del modelo WiSARD para entrenar y clasificar.

En un modelo WiSARD una clase patrón es representada por los contenidos instantáneos de las RAM a través del tiempo. Cada RAM graba el número de ocurrencias de sub-patrones

en la entrada binaria. En otras palabras, una RAM puede ser vista como un histograma del número de ocurrencias de todos los subpatrones binarios ocurridos durante el entrenamiento. En este caso de estudio se asume que el modelo de fondo de un pixel está formado por la combinación de los más frecuentes sub-patrones contenidos en las RAMs.

Este método fue probado previamente en la base de datos C*hange Detection* [74] obteniendo una segmentación robusta, demostrando que es uno de los pocos métodos que usa los mismo parámetros iniciales en toda las diferentes categorías de videos. Ahora en el artículo descrito en [40] se presenta la evaluación del método en la base de datos SBI, donde el GT que se busca es el modelo de fondo en sí, y no una máscara binaria como en el caso *de Change Detection*.

En todos los experimentos que se realizaron se optó por transformar los cuadros de las secuencias al espacio de color Lab como pre-procesamiento. Se usaron diferentes parámetros iniciales en los videos, en la mayoría de los casos se usó direccionamiento de 16 bits para las RAMs y a manera de generar el mejor modelo de fondo, el parámetro *W* fue inicializado en 256 para todos los experimentos, es decir, sin pérdida de información en la discretización de los patrones binarios.

3.1.4.6 *Gibbs–Markov random field* y Redes Hopfield

En [41] se presenta un esquema de detección de movimiento usando *Gibbs-Markov random field* (GMRF) y una red neuronal tipo *Hopfield* (HTNN). Para el modelo de fondo es considerada la técnica *Running Gaussian Average* sobre algunos cuadros posteriores. Para la segmentación se modela un vector de análisis de cambio (CVA) que será obtenido mediante:

$$y(a,b) = (\text{int})\sqrt{\sum_{p=1}^{3}\left(F_t(a,b)^p - B_{t-1}(a,b)^p\right)^2}\,p \qquad (3.2)$$

Donde (int) representa la transformación del valor a entero, *P* es el número de canales de color RGB, $B_{t-1}(a,b)$ es el fondo de referencia en la posición *(a, b)* y $F_t(a,b)$ el cuadro actual. Esta información de diferencias es moldeada mediante GMRF, la segmentación es resuelta

mediante un principio de estimación usando la probabilidad máxima *a posteriori* (MAP) y la red HTNN es usada para estimar la MAP.

En los resultados experimentales, el método mostró mejores resultados resolviendo situaciones de baja iluminación y cambios de iluminación repentinos.

3.1.4.7 Red Neuronal Basada en la Función Radial

El método propuesto en [42] se compone de dos principales módulos: uno que genera multi-fondos y otro enfocado a la detección de movimiento. El método se basa en una red neuronal de función radial (RBF) que se compone de 3 capas; de entrada, oculta y de salida. Al tener suficiente número de neuronas se puede mejorar el desempeño del algoritmo. Sin embargo, conforme aumenta la cantidad de neuronas, se incrementa el costo computacional. Por lo tanto, es importante construir un modelo probabilístico apropiado que pueda representar las neuronas ocultas. El módulo de generación de multi-fondos (MBG) realiza automáticamente un modelo de fondo probabilístico a través del cálculo de la distancia euclidiana de cada pixel en HSV del cuadro actual contra la correspondiente referencia de fondo, un factor de tolerancia empírica es usado para determinar si un pixel del cuadro actual pertenece, o no, a ser candidato de fondo. El MBG determina el número M y puntos centrales C_1,.., C_M de las capas ocultas en la red RBF y la imagen de entrada alimenta la capa de entrada. Finalmente el módulo de deteccion de movimiento se logra mediante la capa de salida de la red RBF que es usada para calcular la máscara binaria de deteccion de movmiento.

La red neuronal basada en una función radial posee una fuerte habilidad de mapeo no lineal y una plasticidad sináptica local de neuronas con una mínima estructura de red. Esto permite que sea adecuada para manejar escenarios dinámicos o estáticos. El algoritmo fue implementado usando programación en C en un CPU Intel Core Duo de 2GB RAM corriendo en un sistema operativo Windows 7. El método puede alcanzar una velocidad de 34 FPS con una resolución de 320x240 pixeles, lo cual es suficiente para aplicaciones en tiempo real.

3.1.5 Modelos estadísticos avanzados

Estos métodos son derivados de los modelos estadísticos convencionales usados en la detección de movimiento tales como las mezclas de Gaussianas (GMM) o el *método Kernel*

density estimation. No obstante, los modelos estadísticos avanzados realizan combinaciones entre modelos estadísticos convencionales y/o proponen el uso de otras distribuciones paramétricas y no paramétricas, con el propósito de crear algoritmos más robustos y con mejores características

3.1.5.1 Modelo de fondo mediante patrones locales binarios (LBP)

En [43] se presenta un método robusto para detección de movimiento que está orientado a videos de sistemas de vigilancia en alta resolución. Un sistema de vigilancia en videos de alta resolución debe de satisfacer dos características; una alta eficiencia computacional para manejar la gran cantidad de datos y una funcionalidad adecuada para analizar los objetos en movimiento. Con el fin de satisfacer las dos características mencionadas, este método propone una serie de módulos conectados cada uno con tareas específicas. Primero, se calcula la orientación de los vectores del movimiento, cada vector es clasificado dentro de una de ocho posibles orientaciones con el fin de reducir el espacio de memoria y la carga computacional. Los vectores de movimiento son detectados y analizados usando orientación de movimiento de histograma (MOH). En paralelo, se realizan las segmentaciones iniciales de los objetos en movimiento mediante el uso de cuadro de diferencia y patrones binarios locales (LBP) presentados en [75], los cuales son usados para discriminar texturas en escala de gris, siendo invariantes a la rotación. Los objetos en movimiento se segmentan por medio de un algoritmo de bloques adaptivos particionados. Se parte la entrada de la imagen en macro-bloques de 32x32 pixeles y luego cada bloque es dividido en bloques más pequeños para ajustarse a la segmentación del objeto en movimiento. Finalmente, se obtiene una segmentación de los objetos en movimiento con sus respectivas direcciones, todo en un procesamiento de video en tiempo real. En la figura 3.1 se puede apreciar el esquema general del método de detección de movimiento.

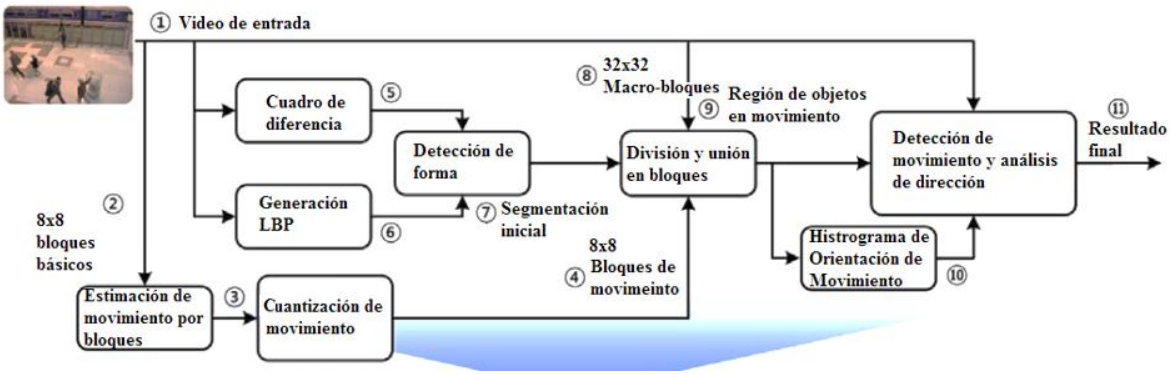


Figura 3.1 Detección de movimiento por medio de algoritmo en [38].

Los resultados experimentales muestran que el método logra buenos resultados en situaciones de inicialización del fondo con camuflaje u objetos presentes en los primeros cuadros de video. El método puede ser aplicado en tiempo real en videos con resolución de 768x576 a una velocidad de 83 FPS.

3.1.5.2 Modelos no paramétricos mediante ViBe (*Visual Background Extractor*)

Uno de los métodos no paramétricos más mencionados en los últimos años, es *Visual Background Extractor* (ViBe), que se basa en un análisis estocástico para discriminar los objetos en movimiento. Entre las ventajas de este método se encuentran: el bajo costo computacional, las altas tasas de detección y su robustez para la eliminación de ruido. Se realiza a nivel pixel almacenando *n* muestras de cuadros anteriores elegidos aleatoriamente y se determina que un pixel es fondo si es estocásticamente invariante en el tiempo.

En aras de realizar una mejora al método ViBe, en [44] se añaden ciertas características heurísticas. La primera corresponde a una estructura que indica el nivel de intermitencia (parpadeo) de cada pixel. Es decir, si un determinado pixel está en el límite entre el objeto en movimiento y el fondo, y además la máscara de clasificación para ese pixel es diferente a la del cuadro anterior, entonces se sospecha que puede ser un caso de parpadeo. Si un pixel contiene un nivel de parpadeo mayor o igual a 30, se remueve del modelo de fondo que servirá de actualización. En consecuencia, no se actualizarán pixeles que hayan sido detectados como objeto en movimiento, ni aquellos que presenten parpadeo. La segunda característica heurística, realiza una operación morfológica de cierre sobre la máscara binaria de la detección de movimiento, esto permite que los componentes que se encuentran muy cercanos entre sí, se terminen de conectar, en caso de no estarlo y también se rellenan los

contornos de los objetos en movimiento detectados. La tercera contribución corresponde a un módulo para evitar insertar objetos de interés en el modelo de fondo, es decir, aunque el objeto se encuentre estático durante un lapso de tiempo en la escena, se conserva la condición de ser objeto en movimiento. Este módulo se realiza mediante el cálculo del gradiente de los bordes internos de los objetos. Si el valor del gradiente es mayor a un umbral, se inhibe la propagación hacia ser detectado como fondo. Este umbral es definido en un valor de 30 e inhibe la propagación durante 180 cuadros. Por último, cada cinco cuadros se analiza la cantidad de puntos detectados como objetos en movimiento, se compara esta cantidad en los cinco cuadros previos contra los cinco más recientes. Si la resta de los más recientes respecto a los cuadros previos es mayor a la cantidad de pixeles que tiene la imagen ponderada por un factor de tamaño de objeto, se activa el módulo de inserción de objetos. Esto con el fin de afrontar las situaciones en la que es deseable insertar objetos dinámicos que antes eran estáticos al modelo.

El método fue implementado en una computadora portátil con un procesador i7 2.2 GHz. El código fue elaborado en C++ utilizando OpenCV. Entre las técnicas con las que se compara este enfoque se encuentran los algoritmos que sirvieron de base para el desarrollo del método como ViBe original y ViBe+. El algoritmo propuesto tiene la ventaja de poder realizar procesamiento en tiempo real a velocidades mayores a 30 FPS. Sin embargo, cuenta con dificultadas en situaciones de camuflaje.

Otro enfoque, basado en los algoritmos ViBe y *Graph cut* para realizar la detección de movimiento se presenta en [45] y es llamado GC STViBe (*Graph Cut* S/T ViBe). El *Graph* (grafo) usado en este método, es conformado como un conjunto de nodos definidos por los pixeles de la imagen de entrada y un conjunto indirecto de bordes que están conectados a los nodos. En el grafo hay dos terminales distintas, llamadas, de fuente y de descenso. Los bordes conectados a la fuente o al descenso son llamados *t-links* y los bordes conectados de modo bidireccional a dos pixeles vecinos se les conoce como *n-links*. Los pesos de estos *n-links* en dos direcciones no necesariamente son iguales. Un *cut* (corte) separará los nodos en dos partes, una parte pertenece a la fuente y la otra al descenso. El mínimo corte del grafo generará una segmentación óptima en la imagen.

El algoritmo [45] se resume en los siguientes puntos. Primero, se realiza una detección de objetos en movimiento para el cuadro actual basada en el método ViBe mejorado, con el fin de extraer la información de fondo y de objetos dinámicos. Segundo, para cada pixel en el cuadro actual, se etiqueta si el pixel corresponde al fondo o a objeto dinámico. Tercero, se obtienen los grupos de fondo y primer plano usando *mean shift clustering*. Cuarto, se inicializan el modelo de grafo con los pixeles correspondientes a la imagen actual como nodos. Quinto, se calculan los t-links y n-links para construir la probabilidad de la función de energía y se construye el modelo de corte del grafo. Sexto, se usan los criterios máximo flujo y mínimo corte para realizar una segmentación binaria y obtener una etiqueta en cada nodo del grafo, S o T. Si un nodo es etiquetado como T, entonces el pixel de la imagen pertenece al modelo de fondo, y si es etiquetado con S pertenecerá a objetos en primer plano. En la figura 3.2 se ilustra mejor el proceso.

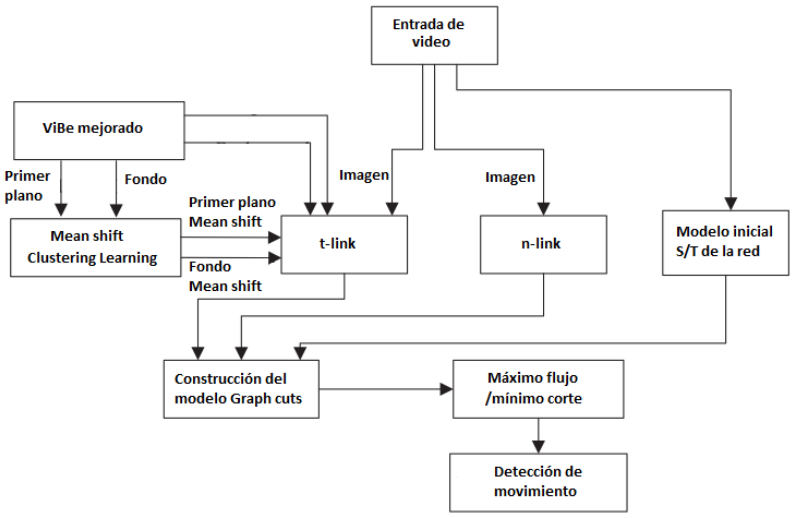


Figura 3.2 Diagrama de flujo método [40].

El método solo se probó con dos videos, por lo que no se puede generalizar su desempeño. Los tiempos de procesamiento y equipo usado para la implementación del método no son reportados en el artículo.

3.1.5.3 Modelo basado en mejora al método SURF

En este artículo [46] se presenta un enfoque que realiza una mejora al método original SURF [76]. Este usa una matriz Hessiana para encontrar puntos de interés, el determinante de la matriz, es una expresión de los cambios locales alrededor de un área. La mejora del

método propuesto se basa en limitar el número de puntos de características detectadas a través del uso de una ventana de supresión de tamaño 7x7x3 para detectar puntos de características solo en un vecindario. La segunda mejora, es un método de cálculo rápido de la orientación dominante de los puntos de características, mediante una ventana corrediza en ángulo de 60° que va cambiando alrededor de una región circular. Luego, se realiza el cálculo de la suma de las respuestas Haar Wavelet (horizontales y verticales), la suma de estas respuestas constituirá un vector y la orientación del vector más grande será también la orientación del punto de características.

El algoritmo se comparó contra el método SURF original y el enfoque SIFT para medir el número de puntos característicos que cada algoritmo usa para representar un objeto detectado y también el costo computacional que le toma a cada uno para representar el objeto. Los experimentos fueron realizados en un CPU Intel Core i5 de 4 GB de RAM y un GPU NVIDIA NVS 3100M. La resolución del video usada fue de 720x480, siendo el algoritmo propuesto el más rápido con 150 ms en promedio y usando un menor número de puntos característicos.

El algoritmo mejora la velocidad y precisión del original SURF, al reducir la complejidad de cálculo. Sin embargo, el método solo fue probado con un video y no se presentaron evaluaciones cuantitativas en cuanto al desempeño de la detección. Por lo que sus resultados quedan limitados, a únicamente, demostrar ser más veloz que los algoritmos SURF original y SIFT con el uso de menos puntos característicos.

3.1.5.4 Modelo no paramétrico basado en *Sift flow*

El articulo presentado en [47] propone un método basado en *Sift flow* [77] para modelar el fondo y detectar objetos en movimiento. Cada pixel es modelado con un grupo de descriptores adaptivos *Sift flow* que son calculados sobre una región rectangular alrededor del pixel. El algoritmo *Sift flow* consiste en relacionar las características *Sift (scale-invariant feature transform*) densamente muestreadas entre dos imágenes y al mismo tiempo preservar las discontinuidades espaciales. Por ejemplo, cada pixel de la imagen en un vecindario de 16x16 es dividido en matrices de 4x4, para cuantificar la orientación en 8 grupos en cada celda. El descriptor por pixel estará compuesto por una dimensión de vector 4x4x8=128. Luego, un modelo inspirado en flujo óptico, relaciona un par de imágenes con descriptores *Sift* a través de una correspondencia densa usando una función objetivo. El *Sift flow* calculado

sobre la región 16x16 es usado como vector de características. El modelo de fondo de los pixeles consiste en un grupo de *Sift flow* adaptivos. La velocidad del modelo es controlada a través de un parámetro de tasa de aprendizaje, en el cual, a mayor valor, mayor velocidad de adaptación. Los descriptores *Sift flow* de los pixeles serán relacionados directamente para calcular los pesos del modelo, luego estos pesos se usarán para realizar la comparación contra un umbral para conocer los modelos *Sift flow* del fondo. Después, los modelos *Sift flow* del fondo son comparados contra los actuales usando una medida de proximidad, si esta es más baja que un umbral de decisión, en al menos un componente, el pixel es clasificado como fondo. De otro modo será marcado como objeto en movimiento.

La efectividad del método fue probada en escenarios interiores y exteriores bajo distintas problemáticas, demostrando que puede trabajar con fondos dinámicos, cambios de iluminación, bajo contraste, vibraciones de cámara, introducción de objetos dinámicos, entre otros. La implementación se llevó a cabo en un CPU Genuine Intel con 2 GB de RAM, para videos de 160x120 se obtuvo una velocidad de 10 FPS y en videos a 320x240 de 3 FPS, esto debido a que el algoritmo requiere calcular el descriptor *Sift* para cada pixel, por lo que, la velocidad de procesamiento aumenta considerablemente.

3.1.5.5 Aproximación estocástica para modelado de fondo

En [48] se presenta un enfoque basado en aprendizaje por medio de aproximación estocástica o mezclas de probabilidad. El método fue ideado para trabajar con cámaras PTZ de zoom panorámico o cámaras estáticas que presentan variación por lo que, se asume que la cámara puede realizar acercamientos y moverse en ambas direcciones horizontal y vertical. Dos modelos son usados en este método, uno para seguir los movimientos de la cámara y otro para detectar los objetos en movimiento.

El modelo para seguir los movimientos de la cámara es basado en un algoritmo de aproximación estocástica, el cual, es usado para entrenar las mezclas probabilísticas, cuyas características son un vector de 24 elementos. Las primeras 3 características corresponden a los colores R, G y B de los canales del cuadro de entrada. Luego los canales RGB son normalizados y uzados como las características de la 4 a la 6. Las características de la 7 a la 12 son descriptores Haar-like de tamaño 9x9, con el fin de obtener robustez con respecto a

los cambios de iluminación. Las características 13 y 14, son componentes de gradiente de un operador Sobel, añadidos para proveer información de textura local. Luego, se considera la información de color y de pixeles vecinos como una indicación de textura de color, por lo que, la característica 15 es el canal rojo del pixel actual, normalizado con respecto el mismo y el pixel inmediato a su izquierda y la 16 es el canal verde del pixel inmediatamente en la parte inferior derecha del actual, normalizado con respecto el actual. A manera de incluir componentes más robustos al modelo se añaden versiones de las características de la 1 a las 6 procesadas con un filtro de mediana de ventana 5x5 que representan las características 17-22. Finalmente, se usan dos pequeños filtros de ventana 3x3 para extraer la información de textura en la imagen, estas serán las características 23 y 24.

El modelo probabilístico para realizar la detección de movimiento, cuenta con la combinación de una distribución Gaussiana para el modelo de fondo y una distribución uniforme para modelar los objetos en primer plano. El modelo probabilístico es alimentado con el vector de las 24 características antes mencionadas para cada pixel. Por su parte, en el modelo de fondo gaussiano se toma en consideración la matriz completa de covariancias, lo cual, hace al modelo suficientemente flexible para la mayoría de las distribuciones de fondo y mantiene una rápida velocidad de operación. Por otro lado, la distribución uniforme para los objetos en primer plano, modelan cualquier objeto entrante, sin importar cuán inesperadas sean sus características. Finalmente, la detección de movimiento se realiza mediante el cálculo la probabilidad del modelo, para cada pixel, de pertenecer al modelo de fondo o al de objetos en primer plano.

A manera de probar la eficiencia del método con cámaras con acercamiento o que presenten vibraciones, se escogieron varias secuencias de diferentes bases de datos con estas características. Con base en los resultados cualitativos y cuantitativos, el modelo de fondo probabilístico propuesto resulta ideal para cubrir las situaciones mencionadas. Sin embargo, una de las desventajas que presenta este método, es que se tienen que realizar ajustes a los parámetros para obtener mejores resultados en las diferentes secuencias de video con PTZ.

3.1.5.6 Modelo de aprendizaje compartido basados en GMM

En [49] se propone un novedoso método de compartición de modelos GMM. Donde, un mecanismo de compartición es presentado para modelar fondo y objetos en primer plano, mediante una relación muchos a uno entre pixeles y modelos. El mecanismo de compartición explota la relación espacio-temporal entre los pixeles, ya que dinámicamente, para cada pixel se busca un modelo óptimo de fondo u objeto en primer plano en una región NxN. Este tipo de algoritmo de modelos compartidos es una manera robusta de afrontar problemas con vibraciones de cámara, fondos dinámicos, entre otros. A los modelos compartidos para fondo y objetos dinámicos se les agregó un mecanismo de cambio, permitiendo que se puedan intercambiar los modelos de objetos en primer plano a modelos de fondo, en caso de que el modelo de objetos en primer plano sea usado por un largo periodo de tiempo. También, si un modelo compartido no es usado por un largo periodo de tiempo este es borrado. El tiempo de vida fue considerado de 500 cuadros para modelos de fondo y 50 para modelos de objetos en primer plano.

Los ruidos que resultan de pequeños movimientos locales en el video, son efectivamente eliminados a través de los modelos compartidos de fondo, mientras que la integridad de los objetos en movimiento es mejorada. La región compartida NxN ideal del modelo se estableció en 5x5 que es donde se obtuvo un mejor resultado en F-measure.

3.1.5.7 Mezcla de Gaussianas globalmente espaciada e incertidumbres

El método propuesto en [50] es una alternativa para la detección de objetos usando un modelo de mezclas Gaussianas innovador que usa los canales RGB y la incertidumbre de los pixeles para realizar un modelado de fondo. El método llamado SGGMM e incertidumbres (*Spatially Global Gaussian Mixture Models and uncertains*) y es capaz de afrontar de una manera eficaz fondos dinámicos y cambios rápidos de iluminación. El modelo de incertidumbre de píxeles es basado en parches, está enfocado en aumentar el color de fondo y es auxiliar al modelo SGGMM para obtener una detección robusta. Los parches o *patch* son regiones centradas en una localización de pixel (x,y) y compuestos de P^2 pixeles, donde el valor de P es elegido de acuerdo al máximo desplazamiento de pixeles.

Para la realización del método SGGMM se tienen dos modelos gaussianos: el de fondo y el de estimación. El primero, es representado por un modelo de *m* mezclas Gaussianas en tres dimensiones, donde, cada valor de pixel en la imagen es representado por un vector de características dados los canales R, G y B. Por su parte, el modelo de estimación, se realiza en una imagen que es el promedio de un *n* número de cuadros posteriores, en la cual, se asocia una función de distribución Gaussiana a cada pixel de la imagen promedio. Después, se calcula para cada pixel, la distancia de *Kullback-Leibler* (*KL-distance*) de los componentes de las Gaussianas del modelo de fondo y de las del modelo de estimación. La *KL-distance* es una medida de la diferencia entre dos distribuciones de probabilidad. Mediante el cálculo de esta, se obtiene un vector de distancias que posteriormente es usado para asignar grupos en la representación SGGMM que formaran parte de un mapeo de soporte. Después de modelar el mapa de soporte usando SGGMM en espacio de color RGB, el siguiente paso es estimar la incertidumbre de los pixeles. Finalmente, se obtiene un vector de características que incluye el color RGB y las incertidumbres, este vector es moldeado en una Gaussiana de 5 dimensiones y se obtiene un mapeo de soporte completo. La detección de objetos dinámicos se realiza evaluando la probabilidad de cada pixel, donde, si la probabilidad es mayor que un umbral T definido por el usuario, el pixel será tomado como fondo, de lo contrario se considerará como objeto en primer plano.

El método afronta problemas relativos a sombras, vibraciones y fondos dinámicos con buena precisión y una eficiencia computacional alta. Sin embargo, cuenta con el problema de que el usuario debe definir adecuadamente un parámetro T_{Cmap} que especifica el número de Gaussianas usadas en el modelo. Este parámetro varía significativamente de una base de datos a otra. Por lo tanto, una posible mejora sería hacer este parámetro flexible o adaptivo.

3.1.7.8 Enfoque SuBSENSE con LBSP (*Local Binary Similarity Patterns*)

En este artículo [51], es presentado un método de segmentación a nivel pixel que depende de las características espacio temporales binarias y de información de color para detectar movimiento. Esto permite que los objetos dinámicos camuflados sean detectados más rápido mientras que las variaciones de iluminación son ignoradas. En lugar de usar ajustes manuales, se usa un lazo de retroalimentación a nivel pixel para dinámicamente ajustar los parámetros internos del método sin intervención del usuario. Estos ajustes se basan en el continuo

monitoreo de la fidelidad del modelo y en los niveles de ruido en la segmentación local. La detección de movimiento se basa en un método de búsqueda de patrones binarios similares en una región local llamados *Local Binary Simalarity Patterns* (LBSP). Estos patrones son calculados en un área cuadriculada de 5x5, luego, se compara el valor del pixel central con cada uno de los valores de los vecinos. Si el valor de la diferencia es menor que un umbral T_d se le asignara un 1 al patrón binario, de lo contrario se le asignara un 0, es decir que aquellos pixeles vecinos que sean semejantes al valor central tendrán un valor de 1 y aquellos que su valor difiera serán etiquetados con un 0. El valor T_d se adapta automáticamente mediante una multiplicación de un valor constante T_r por el valor del pixel central i_x. El valor de T_r fue determinado experimentalmente a 0.3. Para realizar la máscara binaria de detección de movimiento se comparan los patrones LBSP-inter de cada uno de los pixeles en el cuadro actual contra los patrones LBSP-intra del cuadro anterior que son usados como referencia, si el número de patrones binarios que concuerdan son mayores a un cierto número, entonces se detectará como un objeto en movimiento, de lo contrario el pixel será parte del fondo.

El algoritmo SuBSENSE fue evaluado con la base de datos *Change Detection* en todas sus categorías y mediante la metodología de evaluación propuesta en la misma. SuBSENSE fue implementado en un CPU Inter Core i5 usando C++ en OpenCV. En la comparación con los otros algoritmos, SuBSENSE se posicionó dentro de los primeros diez algoritmos, obteniendo el tercer lugar en la evaluación realizada en *Change Detection 2012* y el sexto lugar en la evaluación *Change Detection 2014* esto indica que el método es extremadamente flexible y puede adaptarse incluso a los más difíciles escenarios. La velocidad de procesamiento de este algoritmo es de 45 FPS lo que lo hace ideal para aplicaciones en tiempo real. Como una posible mejora se planea reducir el número total de muestras en los modelos o controlar este parámetro dinámicamente, para favorecer el costo computacional.

3.1.6 Modelos difusos

Son modelos que permiten hacer inferencias con base en reglas heurísticas. Estos modelos extienden los conjuntos rígidos donde solamente un elemento podía o no pertenecer al conjunto. Para los conjuntos difusos existe asociada una función de pertenencia para sus elementos, que indica, en qué medida el elemento forma parte de ese conjunto difuso. Los

modelos basados en lógica difusa se basan en una adaptación matemática al lenguaje que percibimos y utilizamos comúnmente, el cual, es un lenguaje ambiguo e impreciso.

3.1.6.1 Ventana de correlograma multi-canal difusa

En [52] se propone el uso de un correlograma multi-canal difuso para la realización de detección de movimiento. El método es basado en el modelo propuesto de correlograma difuso en [78], el cual, captura las relaciones entre pixeles en una región NxN, incluyendo de este modo información espacial y de textura.

Para el desarrollo del algoritmo es necesaria una cuantización del color RGB a W-1 niveles para reducir el costo computacional. Después se calculan los correlogramas para cada pixel en la región NxN y se aplica una transformación difusa para mapear estos a un espacio de dimensión reducida llamado *kernel fuzzy correlogram*. Finalmente, una ventana difusa multi-canal (MKFC), es aplicada al espacio de dimensión reducida para modelar el fondo *Q*. Esto es usado después para realizar la detección de movimiento, donde el etiquetado $L^t(x)$ (de objeto en primer plano o fondo) de cada pixel *x* se basa en la similitud $S_{Q_b^x, Q_t^x}$ entre el modelo de fondo Q_b^x y el modelo Q_t^x del cuadro actual, comparada contra un umbral th, mediante:

$$L^t(x) = \begin{cases} 1 & si \quad S_{Q_b^x, Q_t^x} < th \\ 0 & cualquier\ otro \end{cases} \tag{3.3}$$

El tamaño de la región de ventana utilizada fue de 5x5. El método no requiere de cuadros de entrenamiento para el modelo de fondo inicial y puede ser inicializado incluso con objetos en movimiento. El método mantiene todos sus parámetros constantes excepto el umbral *th* y presenta la desventaja de un costo computacional elevado, aproximadamente 16 FPS en imágenes de tamaño 160x120 pixeles.

3.1.6.2 Diferencia de histograma de color difuso (FCDH)

En [53] se propone una técnica para el modelado de fondo a través de la combinación de la diferencia del histograma de color (CDH) [79] y lógica difusa. Primero se cuantifica la imagen RGB a *W*-1 niveles, para este algoritmo *W*=16 obteniendo *N*=4096 combinaciones posibles de color, después se convierte la imagen RGB cuantizada al espacio de color Lab,

luego se obtiene la diferencia de color entre un pixel y sus vecinos en una localidad reducida de 3x3, el uso de CDH disminuye significativamente el número de falsos positivos ocasionados por fondos dinámicos, cambios de iluminación repentinos o camuflaje. Después, la diferencia de color del histograma es fuzificada mediante una función de pertenecía Gaussiana. Luego se realiza un histograma de diferencias de color difuso para cada pixel en una región de 5x5. Finalmente es propuesto el uso de *Fuzzy c-means* para clasificar los *k* histogramas de todos los pixeles de la imagen en *c=16* grupos distintos reduciendo así la dimensionalidad del histograma.

El método se compone de cinco etapas principales: determinar la diferencia de color de histograma (CDH), emplear *Fuzzy c-means* para obtener una reducción en el espacio de características (FCDH), inicialización del modelo de fondo, detección de objetos en primer plano usando concordancia de similitudes y finalmente el mantenimiento del fondo, que es logrado al comparar la concordancia de similitud del FCDH del fondo y del cuadro actual contra un umbral.

Los resultados experimentales muestran que el método ofrece una solución a problemas debidos a la variación de iluminación y fondo dinámico. No obstante, falla en detectar la sombra proyectada por los objetos dinámicos y la presencia de estos durante el periodo de inicialización del modelo de fondo, provoca un efecto fantasma. El costo computacional no es reportado en el artículo.

3.1.7 Modelos de subespacios

Son modelos basados en una descomposición derivada de un espacio específico, algunos de los métodos que entran en esta categoría son: análisis de componentes principales, descomposición ortogonal, descomposiciones basadas en súper pixeles, matrices basadas en bajo-rango y descomposición esparcida, entre otros.

3.1.7.1 Modelo basado en *Robust Principal Component Analysis* (RPCA)

Un método de detección de movimiento mediante *Block-based RPCA* es presentado en [54]. El método está compuesto de cinco puntos principales: una segmentación inicial, construcción de *Block-based RPCA*, cálculo del parámetro de balance, resolver *Block-based RPCA* mediante IALM y el procesamiento final en la detección de movimiento. El primero,

es empleado para extraer las máscaras binarias de los objetos en movimiento, segmentados mediante la diferencia de tres cuadros, que es una mejora al método tradicional de cuadros de diferencia. El segundo supone que la matriz de entrada *D*, es compuesta de una matriz de fondo de bajo rango *L* y una matriz esparcida *S*. El tercero es el cálculo del parámetro de balance usado como ponderación de las contribuciones de *L* y *S* para minimizar la función objetivo. El cuarto propone una solución al problema de optimización convexa mediante multiplicadores aumentados de lagrange inexactos (IALM por sus siglas en inglés) resolviendo así la descomposición de matrices *L* y *S*. El último punto es un procesamiento final empleado para mejorar los resultados de la segmentación, usando operaciones morfológicas como dilatación o erosión.

El costo computacional y los medios en los que se llevó a cabo el algoritmo no son reportados en el artículo. El algoritmo se muestra robusto para suprimir sombras.

3.1.7.2 Detección de movimiento mediante COROLA

COROLA (*Contiguous Outliers Representation via Online Low Rank Approximation*) es presentando en [55]. Algunos métodos para detección de movimiento usan una matriz de descomposición de bajo rango para modelar el fondo y una matriz esparcida para detectar los objetos en movimiento. Sin embargo, la mayoría de estos métodos funcionan por lotes de imágenes, lo que impide su aplicación en tiempo real y videos de larga duración. Para solucionar este problema algunos métodos que trabajan *online* han sido propuestos. Sin embargo, estos cuentan con la desventaja de no mostrar resultados satisfactorios en escenarios complejos con fondos dinámicos o ruidosos. COROLA adopta una aproximación de matriz de descomposición de bajo rango en línea para poder resolver las situaciones mencionadas. El algoritmo es basado en los enfoques DECOLOR [80] y OR-PCA [81].

El código fue implementado en MATLAB y C++. A diferencia del método DECOLOR, el costo computacional de COROLA es independiente del número de imágenes procesadas debido a que el costo computacional viene del cálculo del SVD (*Singular Value Decomposition*) en cada iteración. Por lo que, solo se incrementa el tiempo al incrementar el tamaño de resolución espacial de la imagen. COROLA obtuvo el mejor desempeño en comparación a los algoritmos evaluados. A pesar de su desempeño satisfactorio COROLA comparte una desventaja con DECOLOR. Debido a que ambos métodos no cuentan con un

criterio de convergencia convexo, estos probablemente caerán en mínimos locales. El problema de mínimos locales no se presentó en los experimentos debidos a que los fondos dinámicos en las secuencias son aproximadamente similares, sin embargo, probablemente se presente en secuencias de video con cambios de iluminación repentina. En el futuro se planea realizar una mejora a este problema.

3.1.7.3 Modelo basado en descomposición ortogonal SVD

Inspirados por la observación, de que la descomposición ortogonal de un conjunto de intensidades de pixeles revela de una manera eficiente los cambios de iluminación, en [56] se propone un método llamado *Illumination-Invariant Structural Complexity* (IISC), que es simple y eficaz para modelar el fondo en videos bajo situaciones de variación de iluminación.

Para el modelado de fondo se adopta una descomposición en valores singulares o SVD por sus siglas en ingles. El método asume que los valores de los pixeles en una región local pequeña son principalmente determinados por la iluminación, por lo cual podemos suponer que su variación revelará en el valor singular más grande del esquema SVD. Por lo tanto, se piensa que los coeficientes SVD normalizados por el valor singular más grande proporcionan un espacio de características invariante a la iluminación. La diferencia entre los valores del modelo IISC del cuadro actual y el del modelo de fondo, es calculada para clasificar cada pixel; si pertenece al modelo de fondo u objeto en primer plano.

El método IISC propuesto fue evaluado usando dos bases de datos representativas PETS y OTCBVS las cuales contienen videos donde se cambia la iluminación de forma gradual y repentina. Basados en los resultados experimentales, el método propuesto es capaz de hacer frente a problemas de sombras que ocurren en un corto período de tiempo y es efectivo para manejar cambios de iluminación.

3.1.7.4 Detección de movimiento mediante TVRPCA

Tomando en cuenta la observación de que los fondos dinámicos están típicamente más esparcidos que los objetos en primer plano (dinámicos) y que además los objetos dinámicos deben estar contiguos temporal y espacialmente, en [57] se propone un algoritmo de detección de movimiento llamado *Total Variation Regularized RPCA* (TVRPCA), en el cual, se descompone el video analizado en tres partes: fondo estático de bajo nivel, fondo dinámico

esparcido y objetos en primer plano suavizados y esparcidos. El video es tratado como un tensor 3-D en una colección de imágenes partidas en rebanadas en direcciones horizontales y verticales, donde cada corte puede ser visto como una imagen 2-D donde una de las dimensiones es temporal, tal como se muestra en la figura 3.3.

TVRPCA es un algoritmo robusto para el manejo de ruidos del tipo Gaussiano, sal y pimienta, moteado y ruido poisson. Se comparó con algoritmos como RPCA y DECOLOR, mostrando que TVRPCA posee una clara ventaja, especialmente en videos con la presencia de mal clima o fondos complejos. Por otra parte, uno de los problemas que no fue resuelto en su totalidad es el camuflaje, así como también, se tienen problemas de detección o pérdidas cuando existen objetos pequeños que se mueven a gran velocidad, como automóviles que están lejanos a la cámara.

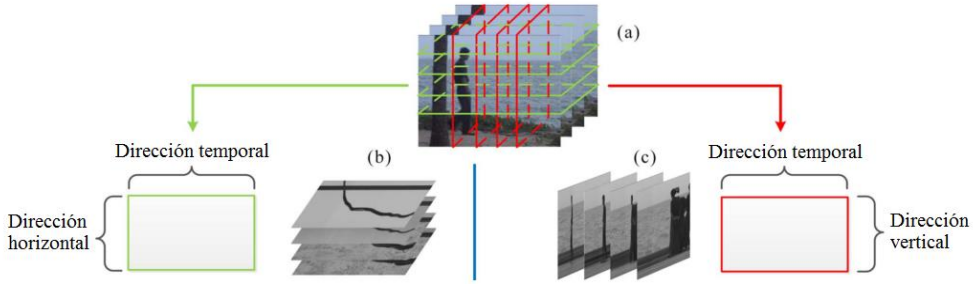


Figura 3.3. Representación del tensor. a) Tensor 3-D, b) Cortes en rebanadas horizontales, c) Cortes en rebanadas verticales.

3.1.7.5 *Graph cut* y modelos Gaussianos

En [58] se propone una función de energía para evaluar las relaciones espaciales en la segmentación. El método es basado en el uso de grafos con cortes normalizados (*Normalized Cut)*, donde se establece una optimización *Normalized Cut* que necesita una segmentación inicial basada en información espacial. Para la segmentación inicial se propone un modelo de sustracción de fondo GSM (*Gaussian Switch Model*). El método usa dos Gaussianas por pixel y por cada canal de color; en total, 6 Gaussianas por pixel. El uso de dos Gaussianas por canal asegura la alta eficiencia y un procesamiento estable. Los fondos generados de los tres canales son combinados por un algoritmo de votación para crear la segmentación de fondo completa. En general cualquier otro método de segmentación puede ser aplicado para crear la segmentación inicial pero solo unos pocos tienen la cualidad de poder ser ejecutados en tiempo real y al mismo tiempo poder ser paralelizados como el uso de Gaussianas.

Una vez obtenida la segmentación inicial, se usa un grafo con un vecindario Von Neuman. Los pesos de las esquinas son computados solamente basados en la información dada de la imagen, los pesos entre los nodos son definidos mediante la distancia Manhattan de los correspondientes valores de color. Los grafos son usados para segregar o clasificar mediante el cálculo de un corte en ellos. Para hacer esto, una función de energía tiene que ser definida para evaluar la calidad de cualquier corte arbitrario en ella. Existen varios enfoques para proponer una función de energía, sin embargo, en este algoritmo se usa *Normalized Cut*, que maximiza la asociación en las diferentes regiones mientras minimiza el corte entre ellas. Esto asegura que toda la información de la escena de video es tomada en cuenta cuando un corte es evaluado.

Existen algunos problemas al usar *Normalized Cut,* ya que al realizar la operación ésta debe asumir que siempre existen objetos en movimiento, esta asunción es errónea y al no tener objetos en movimiento no es posible realizar la operación porque resultaría en una división entre cero, también *Normalized Cut* tiende a segmentar la imagen en aproximadamente la misma cantidad de fondo y objetos dinámicos. Por lo que, para sobre llevar este problema se propone una modificación de *Normalized Cut* llamada N^2cut, la cual no tiene parcialidad para ningún conjunto de objetos en movimiento o fondo, las asociaciones usadas en la operación ahora son calculadas a través de la división del número de bordes en el conjunto, esto elimina la preferencia de largos conjuntos. La adicción de parámetros previene la división entre cero para conjuntos vacíos. Por lo tanto, el 100% del fondo y objetos en movimiento puede ser mapeado por esta función.

Para medir la exactitud, el método fue comparado con el enfoque *Markov Random Field* (MRF) y otras funciones de energía que son minimizadas del mismo modo, donde el uso de N^2cut es claramente la mejor opción. N^2cut logra eliminar el ruido obtenido del método GSM y se obtiene una segmentación en la que el objeto en movimiento está correctamente identificado, aunque los bordes pueden no estar perfectamente alineados con el objeto.

Métodos con precisión comparable al enfoque propuesto por lo general utilizan MRF lo cual inhabilita inherentemente cualquier capacidad de procesamiento en tiempo real debido a la complejidad de cálculos matemáticos. Sin embargo, la minimización por medio de N^2cut es dos veces más rápida y se puede utilizar en entornos en tiempo real, incluso en imágenes

en alta resolución en una computadora de escritorio estándar actual. El algoritmo fue probado en tiempo real en videos de alta definición tomados por una cámara GoPro, en una computadora con procesador i7-4770, la cual permite la paralización del método al usar los cuatro núcleos, obteniendo un desempeño de aproximadamente 20.486 imágenes por segundo.

3.1.7.6 Matriz de descomposición en línea basada en superpixeles

Un método comúnmente utilizado para detección de movimiento, es la descomposición de la matriz de datos en una matriz de bajo rango (correspondiente al fondo) y una matriz de esparcimiento (correspondiente a objetos dinámicos), este método ha mostrado ser muy eficiente para el manejo de fondos dinámicos y variaciones de luminosidad. Sin embargo, cuando el tamaño de los datos crece, el método no puede afrontar problemas en tiempo real y casi siempre muestra un desempeño débil. A manera de hacer frente a los problemas mencionados. En [59] se presenta una matriz de descomposición basada en superpixeles, junto con una norma de regularización de máximos (Max-Norm) y una estructura de restricciones esparcidas. Contrario a los esquemas comunes que descomponen toda la secuencia de video en dos matrices, una esparcida y otra de bajo rango, aquí se empieza por diseñar un esquema de segmentación basada en superpixeles para separar las matrices esparcidas y de bajo rango sobre cada región homogénea. Esto, hace el modelo más confiable y exacto con una cantidad limitada de pixeles.

Max-Norm se basa en una matriz de descomposición y es empleada en cada superpixel segmentado para separar el bajo rango y los valores atípicos iniciales. Luego, la estructura de restricciones esparcidas es usada para explotar la información estructural continua, ya que los pixeles pertenecientes a los objetos en movimiento están espacialmente conectados y esparcidos.

En general el método propuesto presenta un buen desempeño en el manejo de fondos dinámicos y cambios de iluminación. Sin embargo, tiene un bajo desempeño en objetos intermitentes o vibraciones de cámara, lo cual, se planea mejorar en trabajos futuros. En cuanto a costo computacional, se muestra un desempeño mejor en comparación con los demás algoritmos. Sin embargo, aún falta mejorar la parte de velocidad para que se pueda utilizar en una aplicación en tiempo real.

3.1.8 Diccionarios de aprendizaje

Los diccionarios de aprendizaje son un tipo de modelo esparcido o disperso donde la suposición básica de la representación dispersa, es que la imagen actual, se puede representar como una combinación lineal de vectores que forman un diccionario. A continuación, se presentan algunos métodos que caen dentro de esta categoría.

3.1.8.1 Enfoque auto-adaptivo PAWCS

En [60] se propone un método mediante el uso de un nuevo tipo de enfoque basado en *Word Consensus* llamado PAWCS (*Pixel-based Adaptive Word Consensus*). PAWCS es basado en la caracterización y monitoreo de las diferentes representaciones de fondo mediante el uso de *Word-based* aplicado en el nivel del pixel. La idea general es registrar las apariencias de los pixeles sobre el tiempo como *background words* en diccionarios locales usando características de color y textura. Las palabras *words* son consideradas como representaciones del fondo, son algo parecidas a las muy mencionadas *codewords* solo que estas son obtenidas mediante agrupación, a diferencia de *words* que son características únicas sin usar agrupación.

A manera de caracterizar y relacionar las representaciones de los pixeles con la sensibilidad adecuada, se integra el uso de características LBSP (*Local Binary Similarity Pattern*) las cuales, han demostrado ser más sensibles a cambios locales. El propósito de usar LBSP es optimizar el desempeño en situaciones de variaciones de iluminación y mejorar la coherencia o suavidad de la segmentación a través de una descripción de la textura local.

Para realizar la clasificación o máscara binaria, se asume que todos los pixeles son objetos en movimiento a menos que su descripción concuerde con un modelo de fondo. En otras palabras, se calcula la suma de persistencia de concordancia con cada palabra y se compara este valor con un umbral dinámico. También, se cuenta con un mecanismo que actualiza las palabras, eliminando gradualmente las palabras que no tienen suficiente persistencia. Este mecanismo es responsable de controlar el número de palabras activas (con mayor persistencia) en cada diccionario local basado en la complejidad asociada a la región de fondo.

El algoritmo fue implementador en C++ con un CPU Intel i5 trabajando a 22 cuadros por segundo y se encuentra disponible en línea como open-source. El artículo reporta que el método fue evaluado en la base de datos *Change detection* en comparación con otros 27 algoritmos y obtuvo la quinta posición en las evaluaciones.

3.1.8.2 Enfoque de códigos esparcidos para modelado de fondo BMTDL

En [61] se propone un enfoque para modelar el fondo llamado *Background Modeling Through Dictionary Learning* (BMTDL), el cual, se realiza mediante el uso de códigos esparcidos basados en diccionarios de aprendizaje que aprenden y se mantienen actualizados. Un diccionario de aprendizaje esparcido tiene el propósito de construir una representación de los datos a través de una combinación de unos pocos componentes seleccionados de un diccionario de elementos básicos llamados *atoms.* Primero, se divide cada imagen del video en una cuadrícula regular de parches del mismo tamaño. El procesamiento se lleva a cabo individualmente para cada parche permitiendo una alta tasa de paralelización. En la tapa de inicio el diccionario de aprendizaje es derivado de los primeros cuadros de la secuencia con los *k* parches normalizados mediante sus promedios, a manera de restringir los *atoms* a un valor menor que uno. Cada parche es aproximado con una combinación lineal esparcida de *atoms*. Si el error de reconstrucción es pequeño, el parche puede ser aproximado mediante el diccionario disponible, esto quiere decir que ese parche será parte del modelo de fondo. Si por el contrario, el error de reconstrucción es alto, una anomalía es detectada y ésta es posiblemente causada por un objeto en movimiento. En esencia cada parche puede tener cuatro posibles estados; dos de ellos son *NORMAL* y *ANOMALY*, que se refieren a las situaciones cuando nada está pasando y cuando el parche está siendo afectado por un objeto dinámico respectivamente. Los otros dos estados son *UPDATE* y *CLEAN*, los cuales son operaciones aplicadas para mantener el modelo de fondo actualizado y controlar su tamaño.

El método fue analizado enfocándose en situaciones de cambios de iluminación uniforme y no uniforme y variaciones de fondo estables con el fin de demostrar la robustez del algoritmo para adaptarse a este tipo de situaciones. BMTDL fue evaluado con las bases de datos *SABS* y *Change Detection*, en ambos casos, siguiendo la metodología de evaluación propuesta por los autores de las bases de datos. El método, basado intrínsecamente en parches, aprovecha la correlación espacial entre píxeles y muestra robustez y estabilidad a

través de los diferentes tipos de cambios incluyendo iluminación, *jitter*, fondo dinámico y escenarios interiores y exteriores. Sin embargo, el desempeño del algoritmo en cuanto a costo computacional está relacionado con el parámetro de frecuencia de *UPDATE* y el tamaño de los parches. Por lo tanto, si el tamaño del parche es muy grande, el costo computacional se ve afectado. No obstante, el algoritmo cuenta con la ventaja de que puede ser paralelizado usando sistemas multi-núcleo debido a la independencia computacional de cada parche. El diseño de una arquitectura apropiada para realizar procesamiento offline y online para cada parche se encuentra actualmente en investigación.

3.1.8.3 Enfoque de mantenimiento histórico de pixeles BREW-DLHPM

En [62] se propone una representación de fondo basada en diccionarios de aprendizaje y mantenimiento histórico de los pixeles, el método es abreviado BREW-DLHPM y se compone básicamente de cinco bloques. El primer módulo consiste en un muestreo temporal de los cuadros de video. El segundo módulo consiste en aplicar diccionarios de aprendizaje sobre la imagen actual. Cada cuadro del video es representado mediante sus principales características en un diccionario (vector) con el número de átomos menor al número de pixeles, teniendo dos restricciones de esparcimiento *w1* y *w2*, siendo la restricción en el coeficiente del vector para el modelado del fondo y la restricción para objetos dinámicos respectivamente. Dichas restricciones son usadas para calcular el componente de fondo del vector y el componente de objetos dinámicos, ambos componentes son luego considerados para el aprendizaje del modelo de fondo. El tercero módulo trabaja en paralelo junto con el segundo y consta de un mantenimiento de fondo a nivel pixel. El cuarto es una estimación del modelo de fondo dinámico. Finalmente, el último módulo es una toma de decisión para detectar objetos en movimiento.

La implementación del método fue llevada a cabo en un CPU Intel Core i5 con 8GB de RAM. El aprendizaje del modelo de fondo usa 6 iteraciones, suficientes para lograr una correcta representación. El número de iteraciones en el fondo del diccionario de aprendizaje es un parámetro crucial que controla qué cantidad de información común o detallada será aprendida dentro de los átomos del diccionario. También, otro parámetro crítico que se debe considerar es el de la velocidad de aprendizaje del diccionario definida en 0.1.

Los resultados experimentales muestran que BREW-DLHPM puede trabajar en tiempo real en resoluciones de 320x240 y 432x288, obteniendo 61.01 y 36.42 FPS respectivamente. Sin embargo, a más resolución espacial mayor será el costo computacional. Como trabajo futuro se plantea el cálculo automático de los parámetros críticos como la velocidad de aprendizaje o el número de iteraciones.

3.1.8.4 Modelo basado en codificación esparcida K-SVD

En [63] se propone un método de modelado de fondo compuesto de una combinación lineal esparcida de elementos de un diccionario de aprendizaje K-SVD [82] y un conjunto de coeficientes asociados a cada locación de la imagen. A manera de reducir los problemas introducidos por fondos dinámicos es añadido un modelo Gaussiano en los coeficientes. Para obtener la máscara binaria de detección de movimiento se usan dos algoritmos de umbral. El primero se usa para eliminar los errores debido a fondos dinámicos y posible ruido, y el segundo tiene la tarea de finalizar la segmentación y es calculado mediante *K-means*.

En la etapa de aprendizaje del diccionario se consideró un diccionario de 200 elementos. Los parches de entrenamiento fueron seleccionados al azar de un conjunto de 100 imágenes, sub-muestreadas con una tasa de 10 imágenes. El número de parches aleatorios seleccionados es de 1000 por imagen. Se consideraron parches de 7x7 pixeles. El número de coeficientes considerados en la etapa de entrenamiento fue 5.

3.1.9 Modelos de transformación de dominio

El principio de estos modelos se basa en separar el fondo y los objetos dinámicos mediante la ayuda de una transformación de dominio. Para esto, diferentes transformaciones pueden ser usadas, tal como *Wavelet*, *Fast Fourier Transform*, *Hadamard*, entre otras.

3.1.9.1 Enfoque basado en Wavelet (BWM)

A manera de reducir el efecto de las variaciones de luz en [64] se plantea un algoritmo de modelado de fondo que puede ser dividido en los siguientes puntos. Primero se define un modelo de fondo usando aproximación por coeficientes de *Wavelet* y es llamado *Background Wavelet Model* (BWM), luego se usa una norma H2 a manera de extraer los coeficientes de energía entre el cuadro actual y el cuadro de fondo. La norma H2 es usada para medir la covarianza en estado estable o la energía de la respuesta de salida. El siguiente paso es aplicar

una técnica de umbral a los coeficientes de energía para generar la máscara binaria. Por último, se actualizan los *buffers* y el modelo BWM.

Los experimentos fueron corridos en una CPU Intel Core Duo con 1GB de RAM, mediante el software MATLAB y una cámara con resolución 576x720 en RGB. La cámara fue fijada durante los experimentos y estos fueron realizados en tiempo real. El método no fue extensamente probado en distintas circunstancias, por lo tanto, no se puede generalizar su desempeño.

3.1.9.2 Enfoque basado en memoria de diferencia de cuadros MFD y Wavelet

En [65] se presenta un método que utiliza el análisis de *Wavelet* a manera de mejorar el desempeño de un algoritmo de diferencia de cuadros basado en memoria MFD (*Memory-based Frame Differencing*), y de esta manera poder detectar objetos dinámicos lentos y pequeños en diferentes escenarios y aún con cambios de iluminación. La idea básica de los algoritmos de detección basados en memoria es filtrar el ruido a través del uso de una memoria, que mantiene la información acerca de los cuadros previos. Esto permite que el detector de movimiento ignore los cambios causados por el ruido, usando como principio que los cambios ocurrieron de manera aleatoria sobre la secuencia de video.

En el artículo se proponen las siguientes tres variaciones en el método:

La primera propuesta, usa la transformada 2D DWT (*Bidimensional Discrete Wavelet Transform*) en el paso de pre-procesamiento, como un filtro pasa bajas, para obtener una versión suave de los cuadros de entrada, en los cuales, los componentes de alta frecuencia (ruido, fondos dinámicos) de la imagen son excluidos. Después, los coeficientes de aproximación de wavelet son usados para reconstruir los cuadros de entrada en las versiones suavizadas, luego estas imágenes suavizadas son usadas como entrada al algoritmo MFD. Finalmente, operaciones morfológicas son aplicadas para mejorar la segmentación binaria de detección.

En la segunda, se aplica 2D DWT, posteriormente los coeficientes de aproximación son usados directamente en el algoritmo MFD, esto es posible debido a que Wavelet divide cada cuadro en cuatro diferentes bandas de frecuencias sin perder información espacial. En seguida, las salidas del MFD son reconstruidas y pasadas por un umbral para lograr la

detección final. Después, al igual que la primera propuesta se aplican operaciones morfológicas a la máscara binaria.

La tercera variación usa 2D DWT para post-procesamiento. Primero se aplica MFD, después se realizan las operaciones morfológicas y por último se suaviza el ruido en la imagen binaria mediante 2D DWT, la reconstrucción de las aproximaciones wavelet generará la máscara binaria final.

Los resultados experimentales muestran que el método logra afrontar exitosamente los problemas para detectar objetos dinámicos pequeños y lentos, también objetos con bajo contraste y en ambientes con cambios de iluminación. Todos los experimentos fueron implementados usando MATLAB en un CPU Intel Core Duo. Como trabajo futuro se planea realizar un análisis con diferentes resoluciones e investigar otras posibles variaciones utilizando 2D DWT para mejorar el desempeño del algoritmo MDF. El tiempo de procesamiento no fue reportado en el artículo.

3.1.10 Modelos tensores robustos

Los modelos tensores se basan en la asunción de que no todas las relaciones en la naturaleza son lineales, pero la mayoría son diferenciables y de este modo se pueden aproximar localmente con sumas de funciones multi-lineales. Así la mayoría de las magnitudes en física se pueden expresar como tensores. Matemáticamente, un tensor es una entidad algebraica representada por un cierto número de componentes que generaliza los conceptos escalar, vector y matriz de cualquier sistema de coordenadas elegido.

3.1.10.1 *Flux Tensor with Split Gaussian Models* (FTSG)

En [66] se presenta un sistema de detección llamado *Flux Tensor with Split Gaussian models* (FTSG) que aprovecha los beneficios de fusionar un método basado en un tensor espacio-temporal, un novedoso modelado (de fondo y primer plano) y una comparación de apariencia multi-señal. FTSG consiste en tres módulos principales. El primero, es un módulo de detección de movimiento a nivel pixel con dos métodos complementarios. Una detección de movimiento basada en un tensor de flujo y un modelo Gaussiano dividido, basado en sustracción de fondo, ambos corren de manera separada con las imágenes de entrada y producen detecciones de movimiento. El segundo módulo es una fusión, los resultados de

detección del tensor de flujo y la Gaussiana dividida son fusionados usando un sistema basado en reglas, para producir mejores resultados reduciendo los errores debido a cambios de iluminación, ruido o efecto halo. Por último, se incluye un módulo de clasificación a nivel objeto, donde objetos removidos o detenidos son considerados. Los bordes de los objetos estáticos en el modelo de primer plano son comparados con los bordes de los objetos en el cuadro actual del video y también mediante un modelo de fondo que usa concordancia en los chaflanes de la segmentación.

El algoritmo prototipo del sistema completo fue implementado en MATLAB obteniendo una velocidad de procesamiento de 10 FPS en videos con resolución de 320x240. Actualmente FTSG se encuentra dentro de los primeros 10 métodos en la evaluación del 2014 en *Change Detection*.

Una de las ventajas de este sistema híbrido, es que puede manejar situaciones de fondo dinámico, objetos removidos o detenidos, sombras y cambios de iluminación. El sistema ofrece tres clasificaciones en la máscara de detección: objetos en movimiento, objetos estáticos y sombras o iluminación. Una desventaja es su tiempo de procesamiento no apto para tiempo real.

3.1.10.2 *Bayesian Robust Tensor Factorization* (BRTF)

El método en [67] propone un modelo generado por medio de la factorización de un tensor bayesiano robusto (BRTF). Se basa en el uso del tensor de bajo rango CANDECOMP/PARAFAC (CP) [83], que captura la información global, mientras que un tensor esparcido captura la información local, de este modo se obtiene una predicción robusta de la distribución. El tensor de bajo rango CP es modelado por interacciones multi-lineales entre diferentes factores latentes, donde la columna de esparcimiento es forzada por prioridad jerárquica. El tensor esparcido es moldeado por vistas jerárquicas de distribuciones *T-Student* que asocian un hiper-parámetro individual a cada elemento independientemente. Para el modelo de aprendizaje, se desarrolló una inferencia de variación bajo un enfoque completamente bayesiano, el cual, puede prevenir efectivamente el sobre entrenamiento. El método tiene la característica de que todos los parámetros del modelo pueden ser aprendidos

automáticamente de los datos observados, contrario a los demás métodos de tensores de factorización que requieren, o predefinir un rango o ajustar sus parámetros.

Para la detección de movimiento se toma en consideración que el fondo está altamente correlacionado a través de los cuadros del video, por lo tanto, puede ser moldeado mediante un tensor de bajo rango, mientras que los objetos en movimiento pueden ser identificados mediante un tensor esparcido.

Un inconveniente presentado en este artículo es que se muestran evaluaciones cualitativas en lo que al modelado de fondo se refiere, pero no se proporciona una descripción de las evaluaciones cuantitativas.

3.1.11 Otros

En esta clasificación, se contemplan algoritmos nuevos, que no se basan en ningún modelo de los descritos anteriormente o algoritmos basados en modelos poco estudiados, a manera que resulta complicado incorporarlos en alguna clasificación.

3.1.11.1 Enfoque basado en programación genética (GP)

En este artículo [68] se investiga cómo se pueden combinar los mejores algoritmos de detección de movimiento aprovechando sus características y particularidades individuales, con el fin de generar un algoritmo más robusto. Para la elaboración del algoritmo se usa programación genética (GP). La entrada del algoritmo se alimenta con las máscaras binarias correspondientes a las salidas de los algoritmos de detección de movimiento y un conjunto de funciones unarias, binarias y n-arias que se encargan de realizar la combinación de las máscaras binarias mediante operadores AND y OR lógicos y del post-procesamiento a través de filtros.

El uso de la programación genética tiene tres principales ventajas. La primera, se puede elegir automáticamente los algoritmos que muestran los mejores resultados sobre un conjunto de predefinido. La segunda, se pueden generar máscaras binarias con mejor desempeño detección de movimiento a través de la combinación de máscaras binarias obtenidas de otros algoritmos. Por último, se puede proporcionar automáticamente una mejora a las máscaras binarias mediante funciones unarias, binarias y n-arias.

El algoritmo fue evaluado usando todas las métricas y videos proporcionados en la base de datos *Change Detection*. Se propusieron tres estrategias de combinación con los diferentes métodos; un esquema de votos mayoritarios, el método STAPLE y un algoritmo probabilístico de índice Rand. Se realizaron diferentes combinaciones y se observó que los algoritmos llamados IUTIS-3 y IUTIS-5 tuvieron los mejores resultados. Siendo IUTIS-5 el primer lugar cuando fue publicado, en comparación contra los 29 algoritmos de detección de movimiento presentes en la base de datos *Change Detection.*

3.1.11.2 Uso de características Coding Tree Unit (CTU) en videos HEVC

En [69] se propone un enfoque para modelado de fondo mediante el uso de bloques de características espacio-temporales correlacionadas y extraídas directamente de videos comprimidos en HEVC. Típicamente HEVC usa el espacio de color *YCbCr*, donde *Y* conocido como *luma,* es el componente que representa la luminosidad, *Cb* y *Cr* representan los componentes de cromaticidad, es decir, la medida en la que el color se desvía del gris hacia el azul y el rojo respectivamente. Un video HEVC comprimido consiste de una secuencia de cuadros, los cuales están divididos en bloques que no se traslapan llamados *Coding Tree Unit* (CTU). CTU es la unidad compresión fundamental que mantiene la información por cada componente de color en estructuras conocidas como *Coded Tree Block* (CTB).

En un video comprimido HEVC, los coeficientes de trasformación de un CTU registran cambios incrementales que ocurren entre cuadros adyacentes. Por lo que se asume que los cambios mayores corresponden a objetos dinámicos, mientras que los otros son debidos a elementos dinámicos pertenecientes al fondo. El desarrollo del modelo de fondo propuesto involucra decisiones binarias en dos niveles. El primer nivel realiza una segmentación a nivel bloque, de cada cuadro, mediante la selección de un conjunto de bloques CTU (*Coding Tree Unit*) potenciales que se encuentren ocupados total o parcialmente por partes de objetos en movimiento. El segundo nivel realiza una segmentación más fina a nivel pixel, eliminando los pixeles de las CTU seleccionadas que tienen intensidades similares al modelo de fondo. El conjunto de los 200 primeros cuadros del video es usado para el entrenamiento de los parámetros de fondo.

Los resultados experimentales del método propuesto, muestran un buen desempeño para situaciones de fondos dinámicos. El método también posee la ventaja de que es relativamente rápido, el promedio de procesamiento esta entre 25-30 FPS en resoluciones de video de 720×420, permitiendo un procesamiento en tiempo real. En general, el desempeño del algoritmo es comparable con otros algoritmos del estado del arte y mantiene un bajo costo computacional, lo cual lo hace ideal para aplicaciones en tiempo real. Por otro lado, existe una *trade-off* entre el tamaño de bloque CTU usado *L* y el desempeño en F-Measure y velocidad; a mayor tamaño *L* mejor velocidad, pero menor F-Measure y a menor tamaño *L* menor velocidad, pero mejor F-Measure, por lo que, este parámetro deber ser elegido cuidadosamente.

En este capítulo se resumieron 47 enfoques diferentes para realizar la detección de movimiento, es interesante notar que los enfoques más utilizados son redes neuronales y los modelos estadísticos avanzados. Lo cual, puede ser un indicio de la tendencia que se tendrá en un futuro en el desarrollo de nuevos algoritmos enfocados a la detección de movimiento.

Los enfoques aquí presentados, se utilizarán en capítulos posteriores para ser evaluados y comparados, con la finalidad de seleccionar los mejores algoritmos y destacar sus principales ventajas para resolver los problemas de detección de movimiento en secuencias de video.

CAPÍTULO IV. ANÁLISIS, COMPARACIÓN Y SELECCIÓN DE LOS MEJORES ALGORITMOS

Una de las partes esenciales del desarrollo de este tema de tesis es el análisis y comparación de algoritmos para la selección de aquellos que presenten las mejores características. Para el desarrollo de la metodología de selección se consideraron cinco criterios; documentación, auto-adaptabilidad, desempeño, velocidad y actualidad. La documentación revela aspectos referentes a la reproducibilidad del algoritmo, se evalúa la información puesta en las publicaciones y se destacan aquellos algoritmos que otorguen los elementos necesarios para poder realizar comparaciones y repetir el experimento. La auto-adaptabilidad es un criterio donde se evalúa la capacidad que posee un algoritmo de aprender automáticamente y cambiar sus parámetros de acuerdo al escenario de video sobre el que se trabaje. El desempeño indica cuantitativamente qué tan robusto es un algoritmo a diferentes situaciones críticas de video. La velocidad evalúa y destaca a los algoritmos que logren cumplir con un procesamiento en tiempo real. Finalmente, el criterio actualidad toma en consideración la fecha de publicación de los algoritmos con el fin de destacar los más recientes.

El análisis es dividido en dos partes, una enfocada en algoritmos de detección de movimiento y la otra en algoritmos de modelado de fondo. En las siguientes secciones se explica a detalle la metodología realizada en cada caso.

4.1 Análisis enfocado en detección de movimiento

Se partió del conjunto universo *U*, que se define como todo el conjunto de algoritmos existentes enfocados a la detección de movimiento,

$$U = \{x \mid x \text{ es un algoritmo enfocado a detección de movimiento}\} \tag{4.1}$$

y debido a que resulta impráctico hacer un análisis y detallar las características de todo el universo *U* de algoritmos, se consideró un estudio más reducido con base en los algoritmos encontrados en la literatura y los algoritmos evaluados en las bases de datos *Change Detection 2012, Change Detection 2014 y BMC*. De aquí se obtuvieron los conjuntos A_A y A_{BD} sobre los cuales se realizó la selección de los mejores algoritmos. A_A se refiere al conjunto

de algoritmos analizados y A_{BD} se define como el conjunto de algoritmos situados en las bases de datos. Ambos conjuntos son definidos mediante la ecuación 4.2.

$$\begin{aligned} A_A &= \{a \mid a \text{ es un algoritmo analizado}\} \\ A_{BD} &= \{b \mid b \text{ es uno de los algoritmos en las base de datos} \\ &\quad CD2012, CD2014 \text{ ó } BMC\} \end{aligned} \tag{4.2}$$

En el conjunto A_A se realiza un análisis a fondo de los algoritmos de acuerdo a lo reportado en sus artículos o publicaciones. A_A se compone de los 47 algoritmos encontrados en la literatura que fueron descritos en el capítulo 3, y se le añaden los primeros 10 lugares del ranking de desempeño de cada base de datos, esto con el fin de no dejar fuera del grupo A_A a los mejores algoritmos de A_{BD}. Cabe destacar que no todos los algoritmos dentro de los primeros diez lugares de las bases de datos pudieron ser agregados debido a que no se encontraron sus artículos. Por su parte, A_{BD} es compuesto por todos los algoritmos que aparecen evaluados en las bases de datos *Change detection 2012*, *Change detection 2014* y *BMC*. Se consideran únicamente estas tres bases de datos debido a que en ellas se reporta una comparación o evaluación completa entre algoritmos y ofrecen un ranking con base en los mejores desempeños.

En la figura 4.1 se ilustran los conjuntos sobre los que se realizó la metodología de selección de los mejores algoritmos. En total, 112 algoritmos fueron considerados. El conjunto A_A contiene 67 algoritmos, A_{BD} 72 y 27 algoritmos se encuentran en ambos conjuntos.

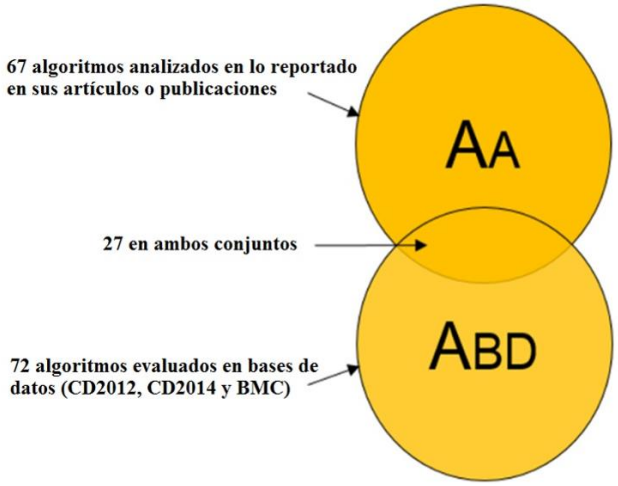


Figura 4.1 Conjuntos de algoritmos usados en análisis.

4.1.1 Metodología de selección de los mejores algoritmos

Con el propósito de determinar cuáles algoritmos de detección de movimiento poseen las mejores características, se propone una metodología basada en evaluar los algoritmos $A_A \cup A_{BD}$ mediante cinco criterios: documentación, auto-adaptabilidad, velocidad, desempeño y actualidad. Luego, con la finalidad de generar una puntuación se les otorga un peso específico. El peso representa la cantidad máxima de puntos que puede alcanzar un algoritmo en cada uno de los criterios. Se repartieron 100 puntos, de manera que a los pesos P_{De} y P_{Ve} de los criterios desempeño y velocidad respectivamente, se les otorga un peso mayor, debido a que en aplicaciones reales es deseable tener algoritmos robustos, con un buen desempeño pero que además puedan realizar un procesamiento en tiempo real. Por su parte, el peso P_{Aa} del criterio auto-adaptabilidad también tiene un peso significativo debido a que entre más automático sea un algoritmo mayor será su flexibilidad a cambios de escenario. Al peso del criterio documentación P_{Do} se le otorga una puntuación menor ya que va enfocado a la reproducibilidad de los resultados y no específicamente en evaluar alguna característica que posea el algoritmo y finalmente el peso actualidad P_{Ac} se lleva la puntuación menos significativa. Los puntos asignados por criterio se muestran en la tabla 4.1.

Tabla 4.1 Pesos otorgados a los criterios de evaluación.

Pesos otorgados a criterios	
Criterios de evaluación	**Puntos máximos por criterio**
Actualidad (P_{Ac})	10
Documentación (P_{Do})	15
Auto-adaptabilidad (P_{Aa})	20
Velocidad (P_{Ve})	25
Desempeño (P_{De})	30
TOTAL	100

Cada uno de los pesos descritos anteriormente serán usados para generar una puntuación a nivel criterio, siendo $P_{ACTUALIDAD}$, $P_{DOCUMENTACIÓN}$, $P_{AUTO\text{-}ADAPTABILIDAD}$, $P_{VELOCIDAD}$ y $P_{DESEMPEÑO}$ las puntuaciones otorgadas a los algoritmos en los criterios actualidad, documentación, auto-adaptabilidad, velocidad y desempeño respectivamente. Después, mediante la suma de estas puntuaciones se obtendrán las puntaciones finales de los algoritmos P_{FINAL} y finalmente se ordenan de mayor a menor puntuación para seleccionar los diez métodos con mejores características. En la figura 4.2 se ilustra el diagrama de selección de los mejores algoritmos.

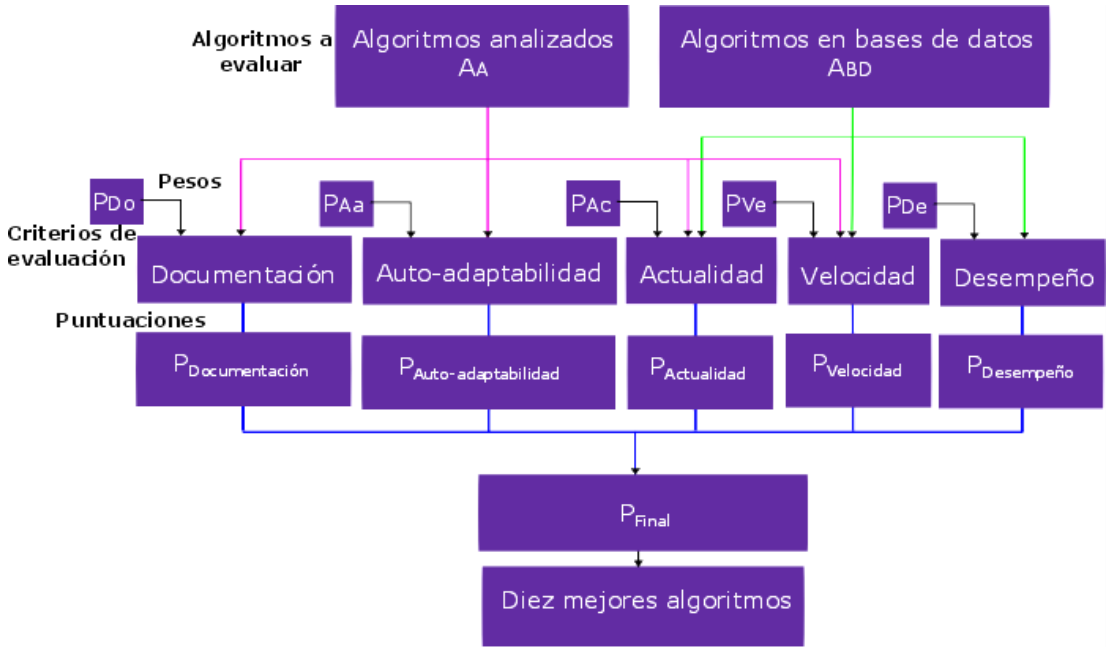


Figura 4.2 Diagrama de selección de los mejores algoritmos.

En el diagrama de la figura 4.2, se puede observar que el conjunto A_A se evalúa mediante los criterios documentación, auto-adaptabilidad, actualidad y velocidad, pero no con el criterio desempeño, esto debido a que los algoritmos de A_A fueron encontrados en la literatura, por lo tanto, no necesariamente se encuentran evaluados con alguna base de datos, incluso en algunos casos sus evaluaciones se realizan mediante un par de videos con situaciones específicas y una cantidad reducida de métricas. Por lo tanto, debido a que no se encuentra estandarizado el proceso de evaluación del desempeño para los algoritmos A_A, no es factible dar una puntuación que sea acertada y justa en el criterio desempeño.

Por otro lado, el conjunto A_{BD} no se evalúa mediante los criterios de documentación y auto-adaptabilidad, debido a que estas evaluaciones provienen de la lectura de los artículos de los algoritmos. Por lo que, realizar la evaluación de estos dos criterios resultaría impráctico por el tiempo necesario en llevarla a cabo en los 70 algoritmos de las bases de datos. No obstante, se incluyen los primeros diez algoritmos de cada base de datos al conjunto A_A con el propósito de que al menos estos algoritmos se evalúen mediante todos los criterios.

A continuación, en las siguientes secciones se describen los criterios de evaluación y se detalla cómo se generan las puntuaciones otorgadas a los algoritmos en cada uno de ellos.

4.1.1.1 Documentación

Uno de los mayores problemas en los artículos publicados de los algoritmos es que no se revelan detalles de muchos de los aspectos del proceso del desarrollo, incluyendo bases de datos, parámetros, diseño experimental, evaluaciones, etc. La falta de información provoca que no se puedan replicar los resultados que se reportan y complica el desarrollo de nuevas propuestas. Por lo tanto, este criterio es evaluado con la finalidad de premiar a los algoritmos que reporten una buena documentación en sus artículos.

Para evaluar la documentación se toma en consideración el conjunto A_A, se evalúan puntos referentes a la reproducibilidad del método de acuerdo a lo reportado en sus artículos. Se resume información pertinente a la metodología de evaluación y desarrollo del método con el fin de tener un panorama más amplio acerca de aquellas características que son y no son reportadas a la hora de documentar la elaboración de algún algoritmo. La evaluación se realiza otorgando una puntuación a cada aspecto contenido dentro de 4 categorías. Las categorías son: procesamiento, evaluación, implementación y comparaciones.

A continuación, se da una descripción de cada categoría y los puntos que se otorgan en cada uno de los elementos a evaluar de la misma.

1) Procesamiento. – Abarca los elementos que influyen en la recolección y manipulación de los datos para producir una salida significativa:

• Número de videos. Cuantos videos son considerados por el autor para realizar la implementación y evaluación del algoritmo. Si se reporta una cantidad pequeña de videos, el algoritmo pudo haber sido sobre entrenado para dar buenos resultados sobre esos videos específicos. A más cantidad de videos mayor generalización se tendrá a las diferentes circunstancias presentes en aplicaciones reales.

Debido a que existían algoritmos evaluados con más de 60 videos y otros con sólo uno o dos videos. Se definió un parámetro *nm* =5 videos, como lo mínimo aceptable para lograr un punto en la evaluación. Por lo tanto, la puntuación otorgada en este elemento de evaluación $Pr_{NúmV}$ dependerá del número de videos utilizados *Nv* y del parámetro *nm*, lo anterior se define mediante la ecuación 4.3.

$$Pr_{NúmV} = \frac{Nv}{nm} \tag{4.3}$$

• Resolución. Dimensiones espaciales que manejan los videos en análisis. Este aspecto es muy importante debido a que no todos los videos cuentan con la misma resolución y en un momento dado que se quiera conocer el comportamiento del algoritmo con base en algún tipo de tamaño de video, es útil tener este tipo de información. La puntuación Pr_{Res} otorgada para este punto se define en la ecuación 4.4.

$$Pr_{Res} = \begin{cases} 1 \quad si \rightarrow & \text{se muestra explícitamente alguna resolución de video utilizada} \\ 0 \quad si \rightarrow & \text{no se muestra explícitamente alguna resolución de video utilizada} \end{cases} \tag{4.4}$$

• Resolución y cuadros por segundo (FPS por sus siglas en inglés). Es la velocidad de procesamiento, el número de imágenes que se pueden procesar en un segundo. Esta velocidad indica si se puede trabajar en aplicaciones en tiempo real o fuera de línea, por lo que, es importante incluir la resolución del video para la cual se obtiene dicha velocidad. La puntuación Pr_{ResFPS} se define en la siguiente ecuación:

$$Pr_{ResFPS} = \begin{cases} 1 \quad si \rightarrow & \text{se muestra explícitamente alguna velocidad con su resolución} \\ 0 \quad si \rightarrow & \text{no se muestra explícitamente alguna velocidad con su resolución} \end{cases} \tag{4.5}$$

• Espacio de color. Normalmente se asume que una imagen a color tiene un espacio de color RGB. Sin embargo, en muchas ocasiones se obtienen mejores resultados trabajando en algún otro espacio de color. Este aspecto debe ser especificado para evitar ambigüedad. Su puntuación Pr_{EspC} se muestra en la ecuación 4.6.

$$Pr_{EspC} = \begin{cases} 1 \quad si \rightarrow \quad se\ muestra\ explícitamente\ el\ espacio\ de\ color \\ \\ 0 \quad si \rightarrow \quad no\ se\ muestra\ explícitamente\ el\ espacio\ de\ color \end{cases} \tag{4.6}$$

Finalmente, la puntuación para la categoría procesamiento *Pr* se obtiene mediante suma de las cuatro puntuaciones otorgadas a los elementos de la misma, lo anterior se define en la ecuación 4.7.

$$Pr = Pr_{NúmV} + Pr_{Res} + Pr_{ResFPS} + Pr_{EspC} \tag{4.7}$$

2) Evaluación. – Contiene los elementos utilizados para realizar la verificación y validación del desempeño del algoritmo:

• Base de datos. Se toma en consideración si se proporciona el nombre de la base de datos donde el algoritmo es evaluado y se cuenta el número de bases de datos que se reportan. Las bases de datos contienen videos con diferentes categorías que tratan de cubrir algunas de las posibles situaciones críticas, tales como: fondos dinámicos, vibraciones de cámara, camuflaje, oclusiones, sombras, mal clima, entre otras. A mayor cantidad de bases de datos usadas por el algoritmo más completa será la evaluación. La puntuación $Ev_{NúmBD}$ de este elemento de evaluación se define en la ecuación 4.8 donde se otorgará un punto por cada base de datos N_{BD} que se reporte.

$$Ev_{NúmBD} = N_{BD} \tag{4.8}$$

• Base de datos completa. En este elemento de evaluación se verifica si se reporta el uso total del número de videos disponibles en la base de datos. Por lo que, si en el elemento anterior se evaluó el número de bases de datos que se reportan, en este elemento se evalúa si estas fueron usadas en su totalidad. La puntuación otorgada a este elemento de evaluación Ev_{BDCom} para cada base de datos, se define por la ecuación 4.9.

$$Ev_{BDCom} = \begin{cases} 1 \quad si \rightarrow \quad Se\ utilizó\ la\ base\ de\ datos\ completa \\ \\ 0 \quad si \rightarrow \quad No\ se\ utilizó\ la\ base\ de\ datos\ completa \end{cases} \tag{4.9}$$

• Evaluación cualitativa. Comparación o demostración del desempeño del algoritmo en forma visual. Puede ser mediante el uso de imágenes o un número de cuadros del video mostrando el resultado obtenido a la salida del algoritmo en comparación al resultado esperado o *Ground truth*, este tipo de evaluación es subjetiva y particularizada. La puntuación otorgada para este elemento de evaluación Ev_{ECual} se obtiene mediante la ecuación 4.10.

$$Ev_{ECual} = \begin{cases} 1 \quad si \rightarrow \quad Se\ muestra\ evaluación\ cualitativa \\ \\ 0 \quad si \rightarrow \quad No\ se\ muestra\ evaluación\ cualitativa \end{cases} \tag{4.10}$$

• Evaluación cuantitativa. Comparación o demostración del desempeño del algoritmo en forma numérica. Se utiliza un conjunto de métricas que muestran el resultado obtenido por el algoritmo en comparación a un *Ground truth*. La puntuación Ev_{ECuan} es otorgada mediante la siguiente ecuación:

$$Ev_{ECuan} = \begin{cases} 1 \quad si \rightarrow \quad Se\ muestra\ evaluación\ cuantitativa \\ \\ 0 \quad si \rightarrow \quad No\ se\ muestra\ evaluación\ cuantitativa \end{cases} \tag{4.11}$$

• Número de métricas. Las métricas son herramientas para calcular el desempeño del algoritmo que indican numéricamente que tan lejano o cercano se encuentra el resultado obtenido del algoritmo al valor deseado. A mayor número de métricas mejor será la descripción de la evaluación del algoritmo. Debido a que el número de métricas que se reportaban en las evaluaciones de los algoritmos no era uniforme y variaban de manera considerable de un algoritmo a otro. Se consideró que la puntuación de este elemento $Ev_{NúmM}$ se realizara mediante la ecuación 4.12, donde TH es un valor de umbral igual a tres, que representa el mínimo número de métricas que se deben usar para que una evaluación se considere significativa y N_M es el número de métricas reportadas al realizar la evaluación de desempeño del algoritmo.

$$Ev_{NúmM} = \begin{cases} 0 \quad si \rightarrow \quad N_M = 0 \\ 1 \quad si \rightarrow \quad 1 \leq N_M < TH \\ 2 \quad si \rightarrow \quad N_M \geq TH \end{cases} \tag{4.12}$$

• Mismos parámetros para todos los videos. En este punto se revisa si el algoritmo utiliza los mismos parámetros al realizar las evaluaciones con distintos videos. La puntuación Ev_{MPar} de este elemento de evaluación se realiza mediante:

$$Ev_{MPar} = \begin{cases} 1 \quad si \rightarrow \quad Se\ usan\ los\ mismos\ parametros \\ \\ 0 \quad si \rightarrow \quad No\ se\ usan\ los\ mismos\ parametros \end{cases} \tag{4.13}$$

La puntuación para la categoría evaluación *Ev* se define en la ecuación 4.14. Se realiza mediante la suma de cada una de las puntuaciones de los elementos de la misma.

$$Ev = Ev_{NúmBD} + Ev_{BDCom} + Ev_{ECual} + Ev_{ECuan} + Ev_{NúmM} + Ev_{MPar} \tag{4.14}$$

3) Implementación. – Se considera la infraestructura necesaria para la realización o ejecución del algoritmo:

• Software. Conjunto de componentes lógicos necesarios que hacen posible la realización de tareas específicas, tal como un procesador de texto o compilador. Se debe reportar el software que se utilizó en la elaboración del algoritmo para facilitar el proceso de replicar el experimento. La puntuación Im_{Sof} otorgada a este elemento de evaluación es realizada mediante la siguiente ecuación:

$$Im_{Sof} = \begin{cases} 1 \quad si \rightarrow \quad Se\ reporta\ el\ software\ utilizado \\ \\ 0 \quad si \rightarrow \quad No\ se\ reporta\ el\ software\ utilizado \end{cases} \tag{4.15}$$

• CPU/GPU. En este punto se debe reportar el uso de CPU o GPU de ser necesario para mejorar la velocidad y costo computacional del algoritmo. La puntuación de este elemento Im_{CPU} se define en la ecuación 4.16. El documentar el uso de GPU o CPU nos revela detalles que facilitan la comparación entre métodos en cuanto a la velocidad y el número de recursos necesarios para la ejecución del algoritmo.

$$Im_{CPU} = \begin{cases} 1 \quad si \rightarrow \quad Se\ reporta\ el\ CPU\ ó\ GPU\ utilizado \\ \\ 0 \quad si \rightarrow \quad No\ se\ reporta\ el\ CPU\ ó\ GPU\ utilizado \end{cases} \tag{4.16}$$

• RAM. Memoria de acceso aleatorio, se utiliza como un espacio de almacenamiento para el sistema operativo del ordenador y la mayor parte del software. Este punto se debe reportar con el fin de conocer que capacidad de memoria suficiente para ejecutar el algoritmo. La puntuación Im_{RAM} del elemento en evaluación RAM, se define mediante la ecuación 4.17.

$$Im_{RAM} = \begin{cases} 1 & si \rightarrow \quad Se\ reporta\ la\ RAM\ utilizada \\ 0 & si \rightarrow \quad No\ se\ reporta\ la\ RAM\ utilizada \end{cases} \tag{4.17}$$

La puntuación para la categoría implementación se da mediante la ecuación 4.18.

$$Im = Im_{Sof} + Im_{CPU} + Im_{RAM} \tag{4.18}$$

4) Comparaciones. – Esta categoría considera los algoritmos usados para realizar las comparaciones con el método propuesto y demostrar su desempeño:

• Número de algoritmos. En este punto se considera el número total de algoritmos con los que se realiza la comparación del algoritmo propuesto. Al tener un mayor número de comparaciones la evaluación se torna más robusta y generalizada.

Debido a que existían métodos evaluados con más de 30 algoritmos y otros con solamente uno, se consideró un umbral TH_2 =3 como el mínimo número de algoritmos con los que se debía comparar el algoritmo propuesto para al menos dar una referencia aproximada del desempeño del algoritmo, por lo que la puntuación $Co_{NúmA}$ otorgada a este elemento quedó definida mediante la ecuación 4.19. Siendo N_A es el número de algoritmos reportados en el artículo contra los que se compara el algoritmo propuesto.

$$Co_{NúmA} = \begin{cases} 0 & si \rightarrow \quad N_A = 0 \\ 1 & si \rightarrow \quad 1 \le N_A < TH_2 \\ 2 & si \rightarrow \quad N_A \ge TH_2 \end{cases} \tag{4.19}$$

La puntuación final para la categoría comparaciones debido a que solo se tiene un elemento en evaluación será asignada mediante:

$$Co = Co_{NúmA} \tag{4.20}$$

Por lo que, la puntuación total P_{DT} será la suma de las puntuaciones de las categorías procesamiento *Pr*, evaluación *Ev*, implementación *Im* y comparaciones *Co*. Definida mediante la ecuación 4.21.

$$P_{DT} = Pr + Ev + Im + Co \tag{4.21}$$

En la tabla 4.2 se resume la puntuación otorgada a cada uno de los elementos de las categorías.

Tabla 4.2 Puntuaciones otorgadas a los elementos de las categorías.

Puntuación en análisis				
	Categoría	**Elementos**	**Puntos**	**Variables**
Total $P_{DT} = Pr + Ev + Im + Co$	Procesamiento $Pr = Pr_{NúmV} + Pr_{Res} + Pr_{ResFPS} + Pr_{EspC}$	Número de videos:	$Pr_{NúmV} = \frac{Nv}{nm}$	nm→ mínimo número aceptable de videos Nv→ número de videos utilizados
		Resolución:	$Pr_{Res} = \begin{cases} 1 & si \rightarrow \text{se muestra explícitamente alguna resolución de video utilizada} \\ 0 & si \rightarrow \text{no se muestra explícitamente alguna resolución de video utilizada} \end{cases}$	
		Resolución/FPS:	$Pr_{ResFPS} = \begin{cases} 1 & si \rightarrow \text{se muestra explícitamente alguna velocidad con su resolución} \\ 0 & si \rightarrow \text{no se muestra explícitamente alguna velocidad con su resolución} \end{cases}$	
		Espacio de color:	$Pr_{EspC} = \begin{cases} 1 & si \rightarrow \text{se muestra explícitamente el espacio de color} \\ 0 & si \rightarrow \text{no se muestra explícitamente el espacio de color} \end{cases}$	
	Evaluación $Ev = Ev_{NúmBD} + Ev_{BDCom} + Ev_{ECual} + Ev_{ECuan} + Ev_{NúmM} + Ev_{MPar}$	Numero bases de datos:	$Ev_{NúmBD} = N_{BD}$	N_{BD}→ número de bases de datos
		¿Base de datos Completa?:	$Ev_{BDCom} = \begin{cases} 1 & si \rightarrow \text{Se utilizó la base de datos completa} \\ 0 & si \rightarrow \text{No se utilizó la base de datos completa} \end{cases}$	
		Evaluación cuantitativa:	$Ev_{ECual} = \begin{cases} 1 & si \rightarrow \text{Se muestra evaluación cualitativa} \\ 0 & si \rightarrow \text{No se muestra evaluación cualitativa} \end{cases}$	
		Evaluación cualitativa:	$Ev_{ECuan} = \begin{cases} 1 & si \rightarrow \text{Se muestra evaluación cuantitativa} \\ 0 & si \rightarrow \text{No se muestra evaluación cuantitativa} \end{cases}$	
		Número de métricas:	$Ev_{NúmM} = \begin{cases} 0 & si \rightarrow N_M = 0 \\ 1 & si \rightarrow 1 \le N_M < TH \\ 2 & si \rightarrow N_M \ge TH \end{cases}$	TH→ mínimo número aceptable de métricas N_M→ número de métricas utilizados
		Mismos parámetros para todos los videos:	$Ev_{MPar} = \begin{cases} 1 & si \rightarrow \text{Se usan los mismos parametros} \\ 0 & si \rightarrow \text{No se usan los mismos parametros} \end{cases}$	
	Implementación $Im = Im_{Sof} + Im_{CPU} + Im_{RAM}$	Software:	$Im_{Sof} = \begin{cases} 1 & si \rightarrow \text{Se reporta el software utilizado} \\ 0 & si \rightarrow \text{No se reporta el software utilizado} \end{cases}$	
		CPU/GPU:	$Im_{CPU} = \begin{cases} 1 & si \rightarrow \text{Se reporta el CPU ó GPU utilizado} \\ 0 & si \rightarrow \text{No se reporta el CPU ó GPU utilizado} \end{cases}$	
		RAM:	$Im_{RAM} = \begin{cases} 1 & si \rightarrow \text{Se reporta la RAM utilizada} \\ 0 & si \rightarrow \text{No se reporta la RAM utilizada} \end{cases}$	
	Comparaciones $Co = Co_{NúmA}$	Número de algoritmos:	$Co_{NúmA} = \begin{cases} 0 & si \rightarrow N_A = 0 \\ 1 & si \rightarrow 1 \le N_A < TH_2 \\ 2 & si \rightarrow N_A \ge TH_2 \end{cases}$	TH_2→ mínimo número aceptable de algoritmos N_A→ número de algoritmos utilizados

Las puntuaciones en cada categoría para cada uno de los 67 algoritmos analizados A_A son mostradas en el apéndice A.

Los puntos P_{DT} obtenidos por los algoritmos se ordenaron de mayor a menor puntuación, luego se seleccionaron los diez primeros lugares, estos algoritmos representan los algoritmos mejor documentados y son mostrados a continuación en la tabla 4.3.

Tabla 4.3 Algoritmos mejor documentados.

Algoritmos mejor documentados		
Ranking	**Métodos**	**Puntuación P_{DT}**
1	*Multimode Background Subtraction*(MBS)	28.2
2	Bewis	28.0
3	AAPSA	27.4
4	DeepBS	27.0
5	Cascade CNN	26.6
6	TEDV	26.0
7	SuBSENSE	25.6
	WeSamBE	25.6
8	*Multimode Background Subtraction Version 0* (MBSV 0)	24.6
9	SharedModel	23.6
10	FTSG	22.6

Para obtener las puntuaciones finales del criterio documentación $P_{DOCUMENTACIÓN}$, se realizó una normalización de los puntos P_{DT} mediante:

$$P_{DOCUMENTACIÓN} = \frac{P_{Do} * P_{DT}}{max(P_{DT})} \tag{4.22}$$

Donde, $max(P_{DT})$ es la puntuación más alta de P_{DT} que corresponde al primer lugar en el ranking, en este caso *Multimode Background Subtraction(MBS)*, por lo que $max(P_{DT})$ =28.2 y P_{Do}=20 corresponde al valor otorgado en el peso del criterio documentación (véase tabla 4.1).

4.1.1.2 Auto-adaptabilidad

A diferencia de los algoritmos convencionales que requieren ajustes y/o actualizaciones manuales de sus parámetros, los algoritmos auto-adaptivos poseen características que les

permiten aprender por si mismos mediante el análisis de la información que están procesando, por lo tanto, sus parámetros o estructura se adaptan al entorno en tiempo real.

El criterio auto-adaptabilidad es propuesto a valoración para resaltar aquellos algoritmos que cuenten con la capacidad de ser autónomos y cambiar su comportamiento con respecto al tipo de video o escenario del cual se detecte el movimiento. Para evaluar este criterio se toma en consideración los algoritmos A_A. Se examinan aspectos referentes a la adaptabilidad del algoritmo de acuerdo a lo reportado en sus publicaciones. Se analizan los parámetros o variables que posee el algoritmo y el comportamiento del mismo.

Para generar las puntuaciones de este criterio se evalúan tres categorías: complejidad, entrenamiento y grado de intervención humana. A continuación, se da una descripción de cada categoría y los puntos que se le otorgan.

1) Complejidad. –Implica el grado de dificultad necesario para determinar los valores de los parámetros manuales y que el algoritmo a su vez, logre sintonizar sus parámetros automáticos para dar los resultados adecuados. Por lo que, entre mayor número de parámetros use el algoritmo, se considera más complejo y se le otorgara una puntuación más baja en el análisis.

Para evaluar esta categoría se obtiene el número máximo de parámetros usados en todos los algoritmos para parámetros automáticos y manuales, siendo $MaxP_A$ y $MaxP_M$ respectivamente, donde $MaxP_A$ =9 y $MaxP_M$ =10. Luego, se propuso la ecuación 4.23,

$$Complex = \left(2 - \left(\frac{P_M}{MaxP_M} \right) - \left(\frac{P_A}{MaxP_A} \right) \right) \tag{4.23}$$

en donde a mayor número de parámetros manuales P_M o automáticos P_A usados por el algoritmo, menor será la puntuación *Complex* que se otorgue debido a que se tendrá un grado de complejidad mayor. Esta ecuación se elaboró de tal modo que la puntuación máxima que pudiera tener un algoritmo fuera de 2 puntos. Es decir, un punto si el algoritmo no usa parámetros manuales y otro punto si el algoritmo no usa parámetros automáticos. Sin embargo, debido a que todos los algoritmos cuentan con al menos un parámetro de ajuste manual o automático, obtener la puntuación máxima en esta categoría resulta prácticamente imposible.

2) Entrenamiento. Se evalúa si el algoritmo requiere o no del uso de cuadros previos, para entrenar el algoritmo y lograr dar un buen desempeño cada vez que es probado con un video nuevo. El uso de cuadros de entrenamiento es muy común en algoritmos con base en redes neuronales y algoritmos que modelan un fondo de referencia inicial. La puntuación *Entr* para esta categoría se define en la ecuación 4.24, en la cual se otorga un punto si no requiere entrenamiento y cero puntos de lo contrario.

$$Entr = \begin{cases} 0 \quad si \rightarrow \quad Requiere\ entrenamiento \\ \\ 1 \quad si \rightarrow \quad No\ requiere\ entrenamiento \end{cases} \tag{4.24}$$

3) Grado de intervención humana. – Se considera el número de parámetros manuales que utilizan los algoritmos con el fin de medir el grado de intervención humana. A menor número de parámetros manuales se tendrá un comportamiento más auto-adaptivo por ende el algoritmo requerirá de menor intervención humana. Se considera que un parámetro es de ajuste manual, cuando se inicializa el parámetro o es modificado manualmente en repetidas ocasiones con el fin de generar un mejor desempeño.

Para evaluar esta categoría y otorgar una puntuación, se propuso la ecuación 4.25.

$$GraIH = M * \left(1 - \left(\frac{P_M}{MaxP_M}\right)\right) \tag{4.25}$$

Siendo *GraIH* la puntuación de la categoría, P_M el número de parámetros manuales utilizados por el algoritmo, $MaxP_M$ el número máximo de parámetros manuales, el cual fue mencionado previamente en la ecuación 4.23 y M=15 representa la puntación máxima para esta categoría. El parámetro *M* fue seleccionado de manera experimental con el fin de crear una separación más marcada entre la puntuación de aquellos algoritmos que no requerían de intervención humana contra los que sí. Este parámetro es analizado más adelante♦ después de definir la puntuación total de las categorías, ya que esta puntuación fue usada para observar el comportamiento de los algoritmos al variar el parámetro *M*.

Entonces, la puntuación total de las categorías evaluadas del criterio auto–adaptabilidad P_{AT} se define mediante la ecuación 4.26.

$$P_{AT} = Complex + Entr + GraIH \tag{4.26}$$

En la tabla 4.4 se resume la puntuación otorgada en las diferentes categorías.

Tabla 4.4 Puntuaciones otorgadas en las categorías del criterio auto-adaptabilidad.

	Categoría	Puntos	Variables
Total $P_{AT} = Complex + Entr + GraIH$	Complejidad	$Complex = \left(2 - \left(\frac{P_M}{MaxP_M}\right) - \left(\frac{P_A}{MaxP_A}\right)\right)$	P_M→ Número de parámetros manuales $MaxP_M$→ Máximo número de parámetros manuales P_A→ Número de parámetros automáticos $MaxP_A$→ Máximo número de parámetros automáticos
	Entrenamiento	$Entr = \begin{cases} 0 \;\; si \rightarrow \;\; Requiere\ entrenamiento \\ 1 \;\; si \rightarrow \;\; No\ requiere\ entrenamiento \end{cases}$	
	Grado de intervención humana	$GraIH = M * \left(1 - \left(\frac{P_M}{MaxP_M}\right)\right)$	M→ Puntuación máxima de la categoría P_M→ Número de parámetros manuales $MaxP_M$→ Máximo número de parámetros manuales

♦Análisis del parámetro M

Debido que se pretendía que los primeros lugares de puntuación P_{AT} destacaran por su comportamiento autónomo a diferentes circunstancias de video, sin importar el número de parámetros (manuales o automáticos) que estos usaran o el entrenamiento que fuese necesario, se le dio un peso mayor, a la categoría grado de intervención humana, en la cual, tendrán mayor puntuación los algoritmos que menor intervención humana tengan. El peso de esta categoría recae sobre el parámetro *M* definido en la ecuación 4.25. Este parámetro fue variado de cinco en cinco por cuestiones prácticas hasta dar los resultados deseados en las puntuaciones ya que, si el parámetro *M* era muy pequeño, los algoritmos con menor cantidad de parámetros (menos complejos) pero no necesariamente los más auto-adaptivos, se posicionaban en los primeros lugares.

En la figura 4.3 se muestran las puntuaciones P_{AT} obtenidas por los primeros diez algoritmos, ordenados de mayor a menor puntuación para un valor *M*=5. Los cinco algoritmos que no requieren de ningún parámetro manual son *DMIEC*, *Block-based RPCA*, *AAPSA*, *RESOM* y *SOM-CNN*, estos se encuentran marcados con una estrella. Como se puede observar al tener un parámetro *M*=5, los algoritmos *AAPSA*, *RESOM* y *SOM-CNN* no se encuentran dentro de los primeros cinco lugares, aún y cuando estos poseen la ventaja de no requerir ajuste manual de parámetros, en la categoría complejidad no obtuvieron una buena puntuación por la cantidad de parámetros automáticos con los que cuentan.

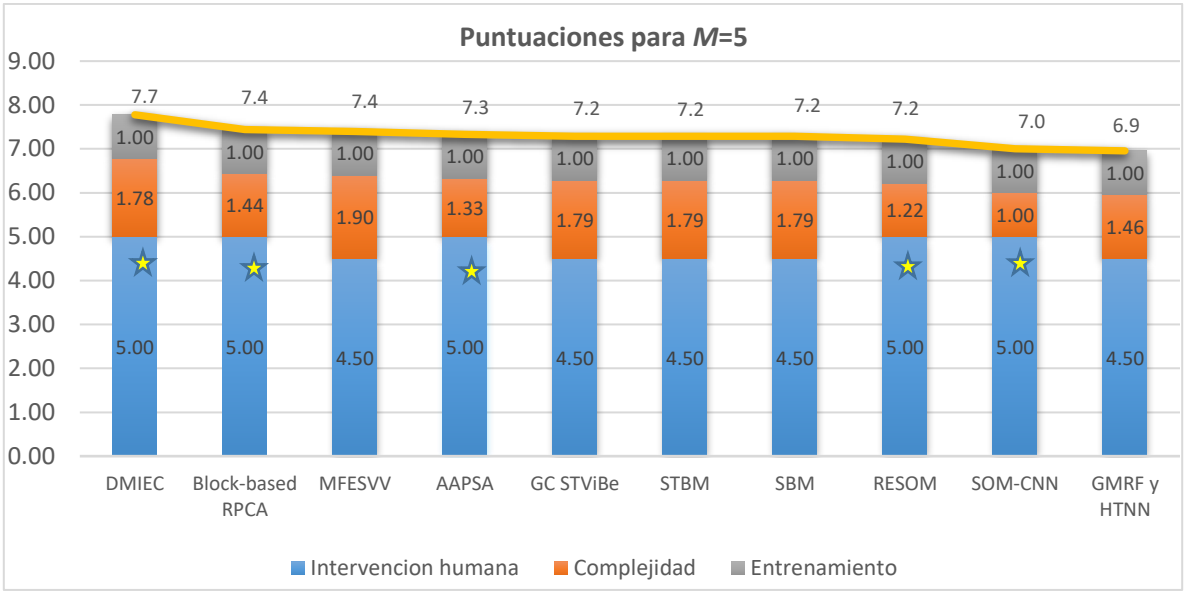


Figura 4.3 Primeros 10 algoritmos auto-adaptivos para *M*=5.

En la figura 4.4 se observan las puntuaciones P_{AT} de los diez primeros algoritmos para un valor *M*=10, los algoritmos marcados con una estrella mencionados anteriormente logran situarse dentro de los primeros cinco lugares. Sin embargo, del quinto lugar al sexto, existe una diferencia de 0.1 únicamente. Por lo que, aún no se observa una gran diferencia entre los algoritmos marcados con la estrella y el resto. Incluso se puede observar que la tendencia de los datos (línea amarrilla) sigue conservando un comportamiento lineal.

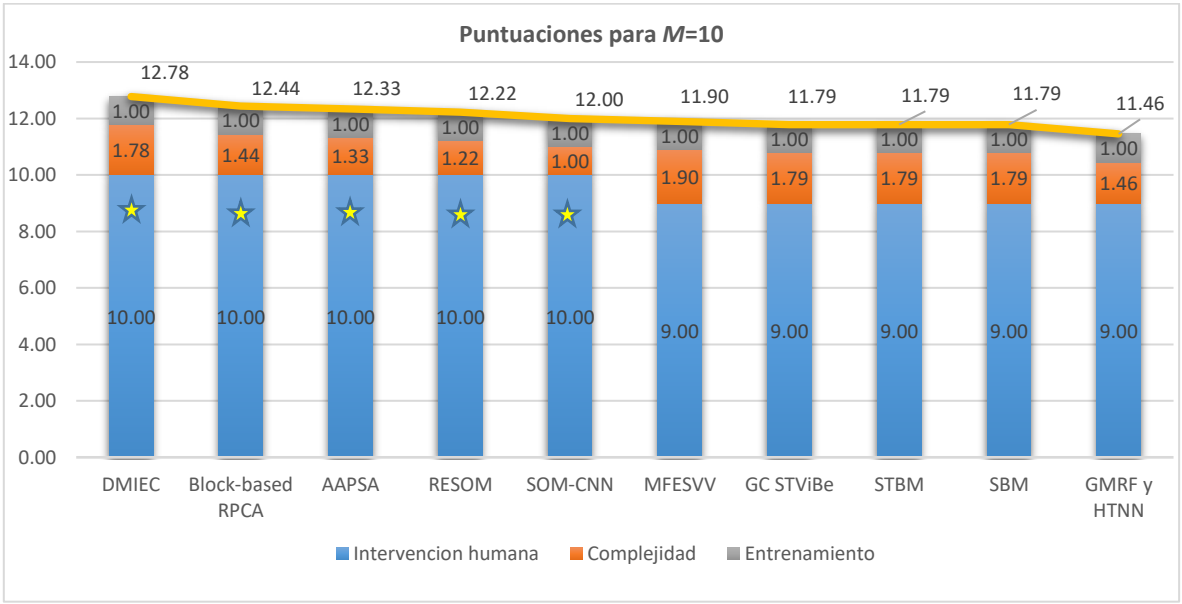


Figura 4.4 Primeros 10 algoritmos auto-adaptivos para *M*=10.

Finalmente, la figura 4.5 muestra las puntuaciones obtenidas con un valor *M*=15. En esta gráfica se observa que en la tendencia de los datos (línea amarrilla) existe una diferencia bastante marcada entre los algoritmos que no cuentan con parámetros manuales (marcados con estrella) y los que sí. Por lo que, se consideró que el valor *M*=15 era el óptimo para lograr los resultados deseados.

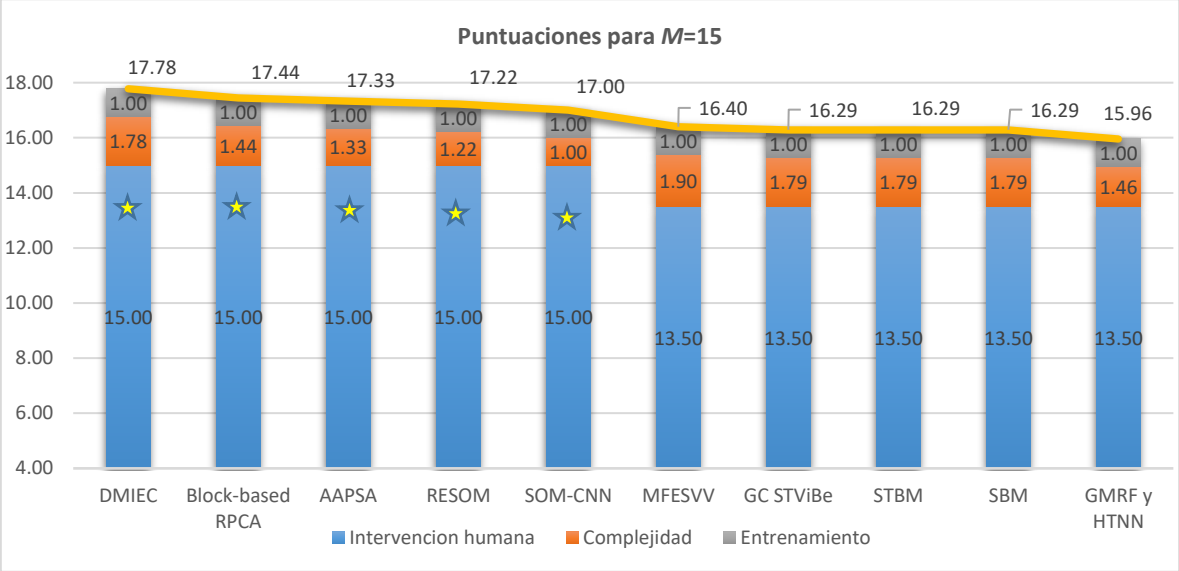


Figura 4.5 Primeros 10 algoritmos auto-adaptivos para *M*=15.

La evaluación completa para todos los algoritmos en el criterio auto-adaptabilidad se observa en el Apéndice B y los diez primeros lugares del ranking son mostrados a continuación en la tabla 4.5.

Tabla 4.5 Algoritmos más auto-adaptivos.

Algoritmos más auto-adaptivos		
Ranking	**Métodos**	**Puntuación P_{AT}**
1	DMIEC	17.778
2	*Block-based* RPCA	17.444
3	AAPSA	17.333
4	RESOM	17.222
5	SOM-CNN	17.000
6	MFESVV	16.400
7	GC STViBe	16.289
	STBM	16.289
	SBM	16.289
8	GMRF y HTNN	15.956
9	EVIEVV	15.178
10	SGMM-SOD	14.800

Para continuar con la metodología de evaluación de los algoritmos, la puntuación final del criterio auto-adaptabilidad $P_{AUTO\text{-}ADAPTABILIDAD}$ se define mediante:

$$P_{AUTO-ADAPTABILIDAD} = \frac{P_{Aa} * P_{AT}}{max(P_{AT})} \tag{4.27}$$

Donde, P_{AT} fue la suma de la puntuación total de los algoritmos en las diferentes categorías, $max(P_{AT})$ es la puntuación más alta de P_{AT} que corresponde a la del algoritmo en primer lugar en el ranking, en este caso $max(P_{AT})$=17.778 y P_{Aa} es el peso del criterio auto-adaptabilidad con valor igual a 20 (véase tabla 4.1).

4.1.1.3 Desempeño

El medir el desempeño de un algoritmo no es una tarea fácil. Previamente existía una tendencia de evaluar y publicar los resultados del algoritmo usando bases de datos propias. Sin embargo, este proceso se volvía subjetivo y no se llegaba a conclusiones precisas debido al uso de videos no representativos, por lo que no se lograba la generalización del algoritmo. Otro problema que se presenta al tratar de medir el desempeño de un algoritmo es el tipo y cantidad de métricas que serán tomadas en cuenta para realizar la evaluación ya que aún no existe una norma o regla sobre que métricas se deben usar para dar como resultado una evaluación fidedigna.

Para valuar este criterio se analizan los algoritmos del conjunto A_{BD} debido a estos algoritmos se encuentran evaluados mediante una misma cantidad de métricas y videos. Las bases de datos utilizadas por A_{BD} son *Change Detection* 2012, *Change Detection* 2014 y BMC debido a que solo ellas cuentan con evaluaciones completas de los algoritmos y ofrecen un ranking de desempeño.

Change detection 2014 cuenta con 11 categorías de video diferentes con 4 o 6 escenarios en cada una. Las categorías comprenden objetos intermitentes, objetos dinámicos, tomas nocturnas, turbulencia, vibraciones de cámara, etc. Se utilizan 7 métricas en total y se ordenan los resultados de los algoritmos con base al promedio obtenido en las métricas, a través de todas las categorías de video. *Change detection* 2012 fue la predecesora de *Change detection* 2014. La principal diferencia entre ambas, radica en el número de categorías de video, reducida a 7 para la base de datos *Change detection 2012*. BMC realiza una evaluación con

19 videos diferentes y cuenta con 6 métricas de las cuales *D-Score* y *SSIM* no figuran en Change Detection. *D-Score* y *SSIM* son métricas con el objetivo de realizar mediciones cualitativas de tal manera que, si un algoritmo reporta falsos positivos y falsos negativos, estas métricas contribuyen a especificar si realmente estos errores afectan a la identificación del objeto de interés.

Para la evaluación del criterio desempeño, primero se realizó un análisis con las bases de datos previamente descritas, con el fin de conocer cual poseía mayor cantidad de elementos para realizar sus evaluaciones y de esta manera poder definir una puntuación para identificar las bases de datos más robustas. Luego, el peso del criterio desempeño P_{De} (véase tabla 4.1) es repartido en las tres bases de datos para otorgar una puntuación máxima a los algoritmos de acuerdo a la base de datos donde se encuentren evaluados. La idea de repartir los puntos, es que la puntuación máxima que pueda alcanzar un algoritmo dependa de la cantidad de bases de datos que se evaluó y desempeño obtenido en las mismas. En la figura 4.6 se ejemplifica este proceso de repartición de puntos.

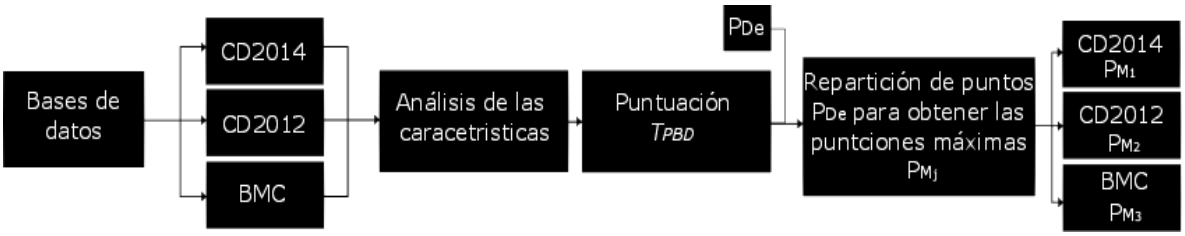


Figura 4.6 Proceso de repartición de puntos de acuerdo a las características de la base de datos.

En el bloque de análisis de las características de la figura 4.6, se examinaron los aspectos de: número de videos NV_{BD}, *Ground Truth* por cada cuadro de video $GTPV_{BD}$ y número de métricas utilizadas para realizar las evaluaciones NM_{BD}, después se consideró cada uno de los aspectos como una cantidad de puntos y mediante la suma de ellos se generó la puntuación total por base de datos T_{PBD} mostrada en la ecuación 4.28

$$T_{PBD} = NV_{BD} + GTPV_{BD} + NM_{BD} \tag{4.28}$$

Los puntos obtenidos en cada aspecto NV_{BD}, $GTPV_{BD}$ y NM_{BD} se observa en la tabla 4.6, así como la puntuación T_{PBD} obtenida para cada una de las bases de datos.

Tabla 4.6 Puntuaciones otorgadas a las bases de datos de acuerdo a sus características.

Base de datos	Número de videos NV_{BD}	GT por cada cuadro de video, usados para evaluar el desempeño de los algoritmos $GTPV_{BD}$	Número de métricas NM_{BD}	Puntuación total T_{PBD}
CD2014	53	1	7	61
CD2012	31	1	7	39
BMC	19	1	6	26

Una vez que se obtuvo la puntuación T_{PBD} de las bases de datos y se identificó qué bases de datos eran más robustas, se procedió a repartir el valor del peso P_{De} (véase tabla 4.1) en cada una de las tres bases de datos para otorgar una puntuación máxima PM_j, siendo j un índice para diferenciar la base de datos sobre la que se trabaja, a manera que PM_1, PM_2 y PM_3 representan a las puntuaciones máximas de las bases de datos CD2014, CD2012 y BMC respectivamente. La repartición de estas puntuaciones es mostrada en la tabla 4.7.

Tabla 4.7 Puntuaciones máximas en cada base de datos.

Índice j	BMC	CD2012	CD2014	Puntuación máxima (PM_j)
1			x	20
2		x		15
3	x			10

Debido a que CD2014 es una extensión de CD2012 y un algoritmo podía evaluarse en ambas bases de datos. Las puntuaciones fueron otorgadas de manera qué si algún algoritmo estaba evaluado en ambas bases de datos, únicamente se le otorgaban los puntos de CD2014. En la tabla 4.8 se muestran los puntos máximos que pueden llegar a obtener los algoritmos de acuerdo al número de bases de datos con las que se evalúen.

Tabla 4.8 Puntos máximos de acuerdo al número de bases de datos utilizadas.

BMC	CD2012	CD2014	Puntuación máxima
	x	x	20
x		x	30
x	x		25
x	x	x	30

Continuando con el proceso de evaluación, para definir las puntuaciones que se les otorgo a los algoritmos se utilizó el ranking o posición que poseen en las bases de datos. Cabe destacar que las tres bases de datos utilizadas para este análisis, muestran a los algoritmos ordenados con base a su desempeño explícitamente. Por lo que la puntuación del algoritmo para cada base de datos se define mediante la ecuación 4.29.

$$PBD_j(n) = \begin{cases} PM_j & si \quad n = 1 \\ PBD_j(n-1) * (1-\alpha) & si \quad n > 1 \end{cases} \qquad (4.29)$$
$$Para \quad j = \{1, 2, 3\}$$

Donde $PBD_j(n)$ es la puntuación otorgada al algoritmo en la posición del ranking *n*, en la *j*-ésima base de datos y PM_j es la puntuación máxima que puede obtener el algoritmo en la base de datos *j* (véase tabla 4.7 para consulta). Por su parte, el parámetro α es una constante con valor de 0.01 definida experimentalmente, utilizada a manera de que el decremento en las puntuaciones se realice suavemente, de esta forma por ejemplo; el algoritmo en primer lugar de la base de datos BMC obtendrá 10 puntos y el que está en segundo lugar disminuirá proporcionalmente a los puntos de su predecesor por la multiplicación de 10*(1- α) obteniendo una puntuación de 9.9 puntos con α=0.01.

El hecho de proponer una disminución suave en las puntuaciones de los algoritmos $PBD_j(n)$ de acuerdo a su ranking *n* es porque si se eligiera un valor de α muy grande, los algoritmos en últimos lugares de las bases de datos no se llevarían ningún punto. También, cabe destacar que no se utilizó ninguna de las métricas colocadas en las bases de datos para otorgar la puntuación $PBD_j(n)$, debido a que no cuentan con las mismas métricas y el rango de variación que existen entre el desempeño de los primeros lugares en las bases de datos contra los últimos, haría que nuevamente se cayera en la situación en que los algoritmos en los primeros lugares se llevaran todos los puntos.

A continuación, en las tablas 4.9-4.11 se observan los puntos otorgados a los algoritmos para cada base de datos.

Tabla 4.9 Puntuaciones otorgadas a los algoritmos en Change detection 2014 a la fecha mayo del 2017.

Base de datos CD2014		
Nombre	**Ranking**	**Puntos(PBD_1)**
Cascade CNN	1	20.00
IUTIS-5	2	19.80
IUTIS-3	3	19.60
DeepBS	4	19.41
PAWCS	5	19.21
SuBSENSE	6	19.02
WeSamBE	7	18.83
SharedModel	8	18.64
FTSG	9	18.45
SaliencySuBSENSE	10	18.27
M4CD Version 2.0	11	18.09
Superpixel Strengthen Background Subtraction	12	17.91
CwisarDRP	13	17.73
C-EFIC	14	17.55
M4CD Version 1.0	15	17.37
Multimode Background Subtraction	16	17.20
BGWiS o CWisarDH	17	17.03
Multimode Background Subtraction Version 0 (MBS V0)	18	16.86
EFIC	19	16.69
Spectral-360	20	16.52
Sample based background subtractor (SBBS)	21	16.36
IUTIS-2	22	16.19
BMOG	23	16.03
AMBER	24	15.87
IUTIS-1	25	15.71
AAPSA	26	15.56
GraphCutDiff	27	15.40
SC_SOBS	28	15.25
Mahalanobis distance	29	15.09
SOBS_CF	30	14.94
RMoG (Region-based Mixture of Gaussians)	31	14.79
KDE - ElGammal	32	14.65
CP3-online	33	14.50
GMM | Stauffer & Grimson	34	14.35
DCB	35	14.21
GMM | Zivkovic	36	14.07
Multiscale Spatio-Temporal BG Model	37	13.93
Euclidean distance	38	13.79

Tabla 4.10 Puntuaciones otorgadas a los algoritmos de la base de datos BMC a la fecha mayo del 2017.

Base de datos BMC		
Nombre	**Ranking**	**Puntos(PBD_3)**
Statistical local difference pattern	1	10.00
GMM & SURF combination	2	9.90
VuMeter	3	9.80
BC, or Bayesian classification	4	9.70
Temporal saliency	5	9.61
GMM | Zivkovic	6	9.51
One-class classification	7	9.41
GMM | KaewTraKulPong	8	9.32
Robust low rank matrix decomposition	9	9.23
NA, or Naive approach	10	9.14
CB, or Codebooks	11	9.04

Tabla 4.11 Puntuaciones otorgadas a los algoritmos en Change detection 2012 a la fecha mayo del 2017.

Base de datos CD2012		
Nombre	**Ranking**	**Puntos(PBD_2)**
Cdet	1	15.00
PAWCS	2	14.85
SuBSENSE	3	14.70
STBM	4	14.55
Multimode Background Subtraction(MBS)	5	14.41
Spectral-360	6	14.26
Multimode Background Subtraction Version 0 (MBS V0)	7	14.12
SBM	8	13.98
SGMM-SOD	9	13.84
PBAS-PID	10	13.70
SBBS	11	13.57
PBAS	12	13.43
GPRMF	13	13.30
DPGMM	14	13.16
CwisarD	15	13.03
PSP-MRF	16	12.90
SOBS_CF	17	12.77
SC-SOBS	18	12.64
CDPS	19	12.52
Chebyshev prob. with Static Object detection	20	12.39
GRBM	21	12.27
RMoG (Region-based Mixture of Gaussians)	22	12.15
Multi-Layer Background Subtraction	23	12.02
SOBS	24	11.90
KNN	25	11.79
SGMM	26	11.67
GRBM_without tuning	27	11.55
KDE - Integrated Spatio-temporal Features	28	11.44
GMM | KaewTraKulPong	29	11.32
KDE - ElGammal	30	11.21
KDE - Spatio-temporal change detection	31	11.10
Bayesian Background	32	10.98
GMM | Stauffer & Grimson	33	10.87
Local-Self similarity	34	10.77
GMM | RECTGAUSS-Tex	35	10.66
GMM | Zivkovic	36	10.55
Prost	37	10.45
TUBITAK UZAY	38	10.34
Histogram	39	10.24
Mahalanobis distance	40	10.14
Euclidean distance	41	10.03

Los mejores algoritmos del criterio desempeño, en cada una de las bases de datos previamente descritas se pueden observar en la tabla 4.12

Tabla 4.12 Mejores algoritmos del criterio desempeño en cada una de las bases de datos.

Algoritmos con mejores desempeños			
Ranking	**CD2014**	**CD2012**	**BMC**
1	Cascade CNN(supervised method)	Cdet	Statistical local difference pattern
2	IUTIS-5	PAWCS	GMM & SURF combination
3	IUTIS-3	SuBSENSE	VuMeter
4	DeepBS (supervised method)	STBM	BC, or Bayesian classification
5	PAWCS	Multimode Background Subtraction(MBS)	Temporal saliency
6	SuBSENSE	Spectral-360	GMM | Zivkovic
7	WeSamBE	Multimode Background Subtraction Version 0 (MBS V0)	One-class classification
8	SharedModel	SBM	GMM | KaewTraKulPong
9	FTSG	SGMM-SOD	Robust low rank matrix decomposition
10	SaliencySuBSENSE	PBAS-PID	NA, or Naive approach

Para concluir con el criterio desempeño la puntuación final $P_{DESEMPEÑO}$ otorgada a cada algoritmo será la suma de las puntuaciones PBD_j obtenidas de cada una en las bases de datos como se muestra en la ecuación 4.30.

$$P_{DESEMPEÑO} = \sum_{j=1}^{3} PBD_j \tag{4.30}$$

4.1.1.4 Velocidad

La velocidad de procesamiento es una medida relacionada al consumo de recursos computacionales por un algoritmo en tiempo de ejecución. Aquí, un algoritmo es considerado eficiente si su consumo de recursos le permite cumplir con un procesamiento en tiempo real. En algoritmos enfocados a detección de movimiento normalmente se usa la unidad FPS o cuadros por segundo por sus siglas en inglés para denotar la velocidad del algoritmo y se considera tiempo real a un procesamiento por encima de los 25 FPS.

Para la evaluación del criterio velocidad se consideran los conjuntos A_A y A_{BD}, y se obtiene el tiempo de procesamiento reportado en sus publicaciones. No obstante, debido a que existen diferentes resoluciones de video y cada autor reporta la velocidad del algoritmo con respecto a alguna resolución específica, se realiza una normalización de los FPS a una resolución de 320x240 como se muestra en la ecuación 4.31, de esta manera se garantiza una comparación justa y uniforme.

$$
\begin{aligned}
R_N &= 320x240 = 76800 \\
R_O &= (ancho)*(alto) \\
V_N &= \frac{(R_O)*(FPS_O)}{R_N}
\end{aligned}
\tag{4.31}
$$

Donde R_N es el factor de normalización, R_O y FPS_O son la resolución y velocidad reportadas en las publicaciones de los algoritmos respectivamente, y V_N es la velocidad en cuadros por segundo normalizada. Una vez normalizadas las velocidades de cada algoritmo se filtran los resultados de acuerdo a aquellos que cumplan con un procesamiento en tiempo real y finalmente se ordenan de mejor a peor tiempo de procesamiento, estos algoritmos son mostrados en la tabla 4.13.

Tabla 4.13 Algoritmos que cumplen con un procesamiento en tiempo real.

Algoritmos más veloces		
Ranking	**Métodos**	**FPS → Resolución (320x240)**
1	AMBER	843
2	CTU	665
3	OMHBP	450
4	PSP-MRF	140
5	GMM | KaewTraKulPong	139
6	Variación ViBe	137
7	Chebyshev prob. with Static Object detection	129
8	CDPS	128
9	KNN	112
10	Mahalanobis distance	103
11	BMOG	102
12	GMM | Stauffer & Grimson	94
13	BMTDL	84
14	GRBM	72
15	GRBM_without tuning	72
16	Euclidean distance	70
17	SBM	66
18	SGMM	65
19	BREW-DLHPM	61
20	GMM | Zivkovic	50
21	PBAS	48
22	KDE - ElGammal	40
23	Cdet	40
24	CwisarDRP	37
25	SharedModel	35
26	RBFMD	34
27	SMSOM-BM	34
28	SGMM-SOD	34
29	SGGMM and Uncertains	33
30	SuBSENSE	30
31	ISURF	29
32	DPGMM	28
33	PAWCS	27
34	KDE - Spatio-temporal change detection	25

La puntuación final $P_{VELOCIDAD}$ para el criterio velocidad es otorgada mediante la ecuación 4.32, considerando el ranking de los algoritmos que cumplen un procesamiento en tiempo real mostrados en la tabla 4.13.

$$P_{VELOCIDAD}(n) = \begin{cases} P_{Ve} & si \quad n = 1 \\ P_{VELOCIDAD}(n-1) * (1-\beta) & si \quad n > 1 \end{cases} \tag{4.32}$$

Donde, n es la posición en el ranking del algoritmo, P_{Ve} es el peso otorgado al criterio velocidad con valor 25 (véase tabla 4.1) y el parámetro β es una constante con valor de 0.1 definida experimentalmente, utilizada a manera de que el decremento en las puntuaciones $P_{VELOCIDAD}$ se realice suavemente, es decir sin cambios bruscos de algoritmo a algoritmo, su comportamiento es similar al del parámetro α mostrado previamente en la ecuación 4.29 para el criterio desempeño. Si su valor fuera muy grande, los algoritmos en los últimos lugares del ranking no se llevarían ningún punto y si es muy pequeño, prácticamente se les estaría otorgando a todos los algoritmos la puntuación más alta P_{Ve}.

4.1.1.5 Actualidad

Para la evaluación del criterio actualidad se consideran los algoritmos analizados A_A y A_{BD}, estos algoritmos son clasificados con respecto al año de su publicación y se filtran del periodo 2013-2017. Este criterio fue pensado para favorecer a las ideas nuevas ya que los algoritmos más recientes son los menos estudiados.

Para otorgar la puntuación final $P_{ACTUALIDAD}$ al criterio actualidad, se propuso la ecuación 4.32 mostrada a continuación:

$$P_{ACTUALIDAD} = \begin{cases} P_{Ac} & si \rightarrow \quad A_P \geq 2013 \\ 0 & si \rightarrow \quad A_P < 2013 \end{cases} \tag{4.32}$$

Donde A_P representa el año de publicación del artículo y P_{Ac} el peso del criterio actualidad mostrado previamente en la tabla 4.1. En esta evaluación se les dio el mismo valor a todos los algoritmos dentro del periodo 2013-2017.

De un total de 112 algoritmos en el análisis, 74 algoritmos quedaron dentro de la clasificación 2013-2017, estos se observan en la tabla 4.14.

Tabla 4.14 Algoritmos dentro del periodo 2013-2017.

Algoritmos más recientes			
Año	**Métodos**		**Puntos finales $P_{ACTUALIDAD}$**
2017	Multimode Background Subtraction(MBS)	CwisarDRP	10 Puntos generales
	SBBS	DeepBS	
	BMOG	WeSamBE	
2016	Cascade CNN	EAMV	
	M4CD Version 1.0	ERCI	
	M4CD Version 2.0	FCDH	
	AAPSA	IISC	
	BREW-DLHPM	NeRM	
	ConvNets	SGGMM and Uncertains	
	DMW	TVRPCA	
	DRCIMF	BRTF	
2015	Multimode Background Subtraction Version 0 (MBS V0)	Block-based RPCA	
	PAWCS	BMTDL	
	SuBSENSE	COROLA	
	BGWiS o CWisarDH	CTU	
	C-EFIC	DMIEC	
	CP3-online	FMR	
	EFIC	GMRF y HTNN	
	GraphCutDiff	MDLS	
	IUTIS-1	Normalized cut	
	IUTIS-3	OMHBP	
	IUTIS-5	RESOM	
	SaliencySuBSENSE	SMSOM-BM	
	SharedModel	SOM-CNN	
	AEMF		
2014	PBAS-PID	DTCNN	
	pROST	MFESVV	
	SBM	MKFC	
	STBM	SAG	
	AMBER	SiftFlow	
	FTSG	TEDV	
	Multiscale Spatio-Temporal BG Model	Variación ViBe	
	AFPA		
2013	RMoG (Region-based Mixture of Gaussians)	GC STViBe	
	CDPS	ISURF	
	CwisarD	K-SVD	
	KDE - Spatio-temporal change detection	PS-RBM	
	EVIEVV	RBFMD	

4.1.1.6 Selección de los mejores algoritmos en detección de movimiento

En secciones anteriores se destacaron los algoritmos mejor documentados, más auto-adaptivos, más veloces, con mejores desempeños y los más recientes. Sin embargo, aún resulta difícil el definir cuales algoritmos presentan las mejores características considerando todos los criterios. Por lo que, de acuerdo a las puntuaciones obtenidas para cada uno de los criterios, se definió una puntuación final P_{FINAL} mediante la ecuación 4.33.

$$P_{FINAL} = P_{DOCUMENTACIÓN} + P_{AUTO-ADAPTABILIDAD} + P_{DESEMPEÑO} + P_{VELOCIDAD} + P_{ACTUALIDAD} \quad (4.33)$$

Para el desarrollo de la metodología de selección de los mejores algoritmos también se realizó una búsqueda enfocada en detectar los algoritmos que tuvieran código disponible con el propósito de obtener dos clasificaciones, una donde refleje los métodos con mejores características y otra donde se les diera más importancia a los algoritmos de código disponible.

En la búsqueda de código disponible se localizaron dos librerías enfocadas a detección de movimiento, estas librerías cuentan con una cantidad considerable de algoritmos y proporcionan una vía para obtener la máscara de detección de cada método y poder evaluar su comportamiento en diferentes circunstancias de video. También, por otro lado, se realizó una búsqueda en la web enfocada a encontrar los algoritmos que no se localizaran en las librerías. Las librerías encontradas y los resultados de la búsqueda web son descritas a continuación:

- BGSLibrary[1] (*Background Subtraction Library*)

Desarrollada por Andrews Sobral provee una interfaz fácil de usar desarrollada en C++ basada en OpenCV para separar los objetos dinámicos del fondo. BGSLibrary es compatible con OpenCV 2.x y 3.x y compila en Windows, Linux y Mac OS X. Actualmente la librería cuenta con 43 algoritmos. La librería es de código abierto para propósitos académicos. En la figura 4.7 se observa la interfaz gráfica de la librería.

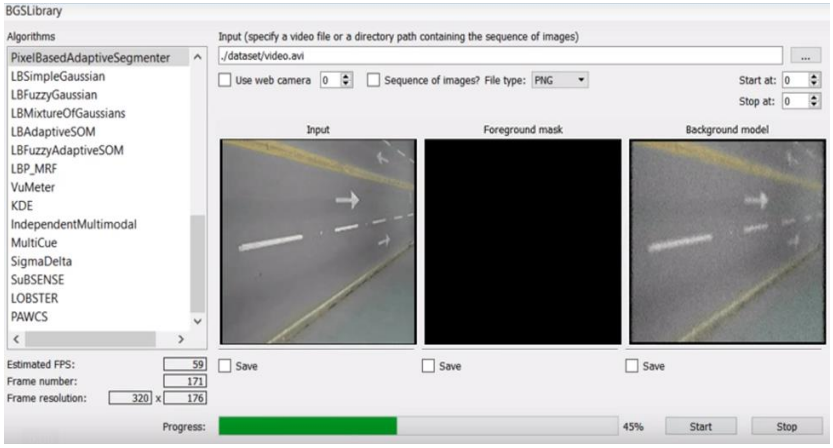


Figura 4.7 Interfaz gráfica BGSLibrary.

[1] Disponible en: https://github.com/andrewssobral/bgslibrary

- LRSLibrary[2] (*Low-Rank and Sparse tool for Background Modeling and Subtraction in Videos*)

Desarrollada también Andrews Sobral provee una colección de algoritmos de descomposición en matrices de bajo rango y esparcidas, en el software computacional MATLAB. La librería fue diseñada para segmentación de objetos dinámicos, pero puede ser adaptada para otros propósitos de sistemas de visión. LRSLibrary es compatible con MATLAB R2013, R2014, R2015 y R2016 para versiones x86 y x64. Actualmente la librería cuenta con 104 algoritmos. La interfaz gráfica es mostrada en la figura 4.8.

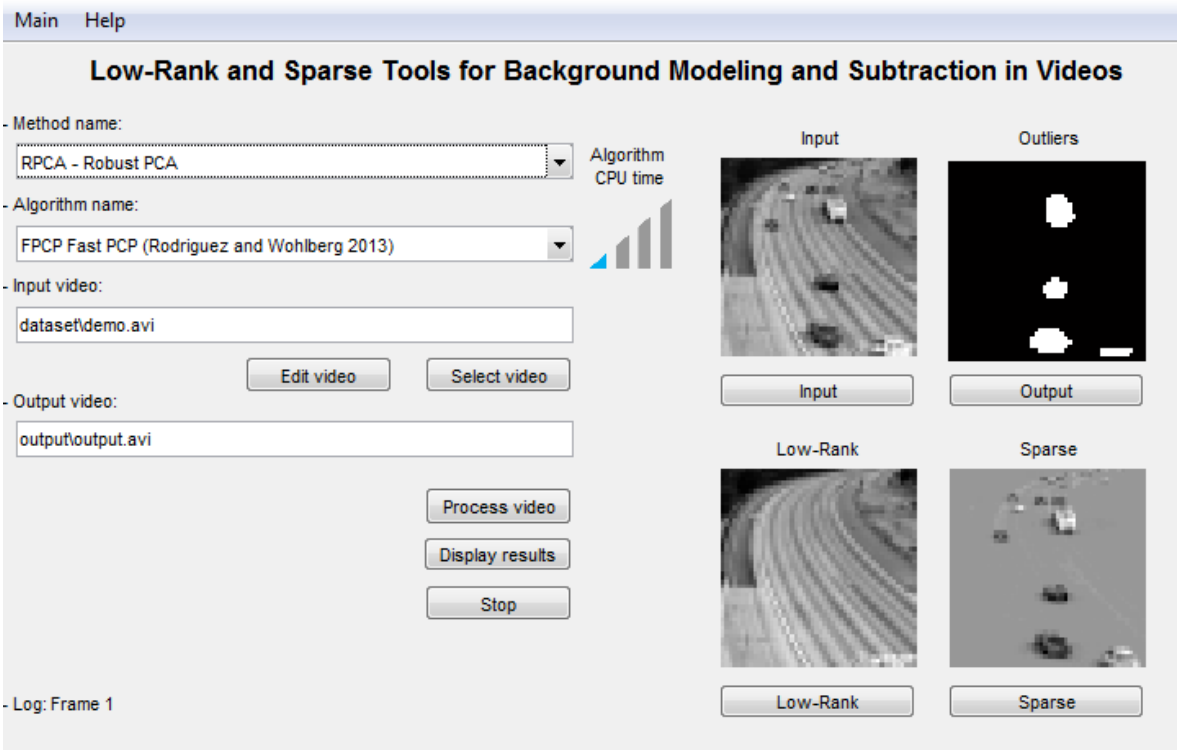


Figura 4.8 Interfaz gráfica LRSLibrary.

- Búsqueda en la web

Se realizó una búsqueda en diferentes páginas web y se encontraron 14 algoritmos con código disponible, de los cuales 3 de ellos ya estaban incluidos en las librerías. La tabla 4.15 muestra los códigos encontrados y su ubicación.

[2] Disponible en: https://github.com/andrewssobral/lrslibrary

Tabla 4.15 Algoritmos con código disponible encontrados en la web.

Algoritmos encontrados en la web			
Incluidos en librerías	**Método**	**Autor (Año)**	**Ubicación**
SI	SOBS	Madalena et al. (2008)	http://www.na.icar.cnr.it/~maddalena.l/MODLab/SoftwareSOBS.html
SI	DECOLOR	X. Zhou et al. (2013)	https://fling.seas.upenn.edu/~xiaowz/dynamic/wordpress/projects/
NO	PBAS	Martin Hofmann et al. (2012)	https://sites.google.com/site/pbassegmenter/download-1
NO	COROLA	Moein Shakeri et al. (2016)	https://github.com/shakeri542/COROLA_Algorithm
NO	laBGen	B. Laugraud et al. (2016)	https://github.com/benlaug/labgen-of
SI	PAWCS	P.-L. St-Charles et al. (2015)	https://bitbucket.org/pierre luc st charles/pawcs
NO	SMSOM-BM	Zhao, Zhenjie et al. (2015)	https://github.com/zhaozj89/SMSOM
NO	Variación ViBe	L. Gervasoni et al. (2014)	No disponible (fue conseguido con el autor)
NO	Cascade CNN	Wang Y. et al. (2016)	https://github.com/zhimingluo/MovingObjectSegmentation
NO	Bewis	M. Giordano et al. (2015)	https://github.com/giordamaug/BEWiS
NO	IMBS-MT	Domenico Daniele et al. (2016)	https://github.com/dbloisi/imbs-mt
NO	labgen-p	B. Laugraud et al. (2016)	https://github.com/benlaug/labgen-p
NO	AAPSA	G. Ramírez et al. (2016)	No disponible (fue conseguido con el autor)
NO	BE-AAPSA	G. Ramírez et al. (2017)	No disponible (fue conseguido con el autor)

Finalmente, de la búsqueda de código, en total se encontraron 146 algoritmos de código disponible.

La evaluación y puntuación final P_{FINAL} de los algoritmos con respecto a los criterios en análisis e incluyendo si estos cuentan con código disponible o no, se muestra en la tabla 4.16. En el criterio desempeño se observan también las puntuaciones obtenidas en cada una de las bases de datos PBD_j y la puntuación final del criterio $P_{DESEMPEÑO}$, esto con la finalidad de que se observe en cuales bases de datos se evaluó el algoritmo y que puntuaciones obtuvo. Es importante señalar nuevamente que a los algoritmos evaluados en las bases de datos CD2012 y CD2014 únicamente se les otorgó la puntuación de CD2014, debido a que esta última es una extensión de la otra.

Tabla 4.16-A Algoritmos del 1 al 10 ordenados de acuerdo a sus puntuaciones obtenidas.

Ranking	**Documentación** $P_{DOCUMENTACIÓN}$	**Adaptabilidad** $P_{AUTO\text{-}ADAPTABILIDAD}$	**Velocidad** $P_{VELOCIDAD}$	**Actualidad** $P_{RECIENTES}$	**Desempeño**				**Puntuación final** P_{FINAL}	**Código disponible**	**Lista de algoritmos**
					CD2012 PBD_2	**CD2014** PBD_1	**BMC** PBD_3	$P_{DESEMPEÑO}$			
1	13.62	16.15	1.18	10.00	14.70	19.02	0.00	19.02	59.96	✔	SuBSENSE [51]
2	14.15	15.53	0.00	10.00	0.00	20.00	0.00	20.00	59.67	✔	Cascade CNN [84]
3	14.57	19.50	0.00	10.00	0.00	15.56	0.00	15.56	59.63	✔	AAPSA [33]
4	12.55	15.90	1.99	10.00	0.00	18.64	0.00	18.64	59.09	✘	SharedModel [49]
5	13.62	16.53	0.00	10.00	0.00	18.83	0.00	18.83	58.97	✘	WeSamBE [85]
6	14.36	15.15	0.00	10.00	0.00	19.41	0.00	19.41	58.92	✘	DeepBS [86]
7	11.70	18.33	4.63	10.00	13.98	0.00	0.00	13.98	58.64	✘	SBM [87]
8	11.49	15.90	0.86	10.00	14.85	19.21	0.00	19.21	57.46	✔	PAWCS [88]
9	12.02	16.03	0.00	10.00	0.00	18.45	0.00	18.45	56.50	✘	FTSG [66]
10	13.09	15.03	0.00	10.00	14.12	16.86	0.00	16.86	54.97	✘	MBSV 0 [89]

Tabla 4.16-B Algoritmos del 11 al 62 ordenados de acuerdo a sus puntuaciones obtenidas.

Ranking	Documentación $P_{DOCUMENTACIÓN}$	Adaptabilidad $P_{AUTO-ADAPTABILIDAD}$	Velocidad $P_{VELOCIDAD}$	Actualidad $P_{RECIENTES}$	Desempeño				Puntuación final P_{FINAL}	Código disponible	Lista de algoritmos
					CD2012 PBD_2	CD2014 PBD_1	BMC PBD_3	$P_{DESEMPEÑO}$			
11	3.30	14.48	16.40	0.00	11.32	0.00	9.32	20.64	54.82	✔	GMM | KaewTraKulPong [90]
12	11.70	18.33	0.00	10.00	14.55	0.00	0.00	14.55	54.58	✘	STBM [87]
13	14.89	11.93	0.00	10.00	0.00	17.03	0.00	17.03	53.85	✘	BGWiS o CWisarDH [74]
14	8.62	11.55	22.50	10.00	0.00	0.00	0.00	0.00	52.67	✘	CTU [69]
15	10.96	11.55	0.00	10.00	0.00	19.80	0.00	19.80	52.31	✘	IUTIS-5 [68]
16	10.96	11.55	0.00	10.00	0.00	19.60	0.00	19.60	52.11	✘	IUTIS-3 [68]
17	0.00	0.00	25.00	10.00	0.00	15.87	0.00	15.87	50.87	✘	AMBER [91]
18	5.74	13.05	20.25	10.00	0.00	0.00	0.00	0.00	49.04	✘	OMHBP [43]
19	10.96	11.55	0.00	10.00	0.00	15.71	0.00	15.71	48.22	✘	IUTIS-1 [68]
20	15.00	6.28	0.00	10.00	14.41	0.00	0.00	14.41	45.68	✘	Multimode Background Subtraction(MBS) [92]
21	5.96	14.60	0.00	10.00	13.70	0.00	0.00	13.70	44.26	✘	PBAS-PID [93]
22	2.45	14.48	3.38	0.00	10.55	14.07	9.51	23.58	43.88	✔	GMM | Zivkovic [94]
23	6.81	9.45	14.76	10.00	0.00	0.00	0.00	0.00	41.02	✔	Variación ViBe [44]
24	9.57	19.38	0.00	10.00	0.00	0.00	0.00	0.00	38.95	✘	RESOM [34]
25	13.83	14.85	0.00	10.00	0.00	0.00	0.00	0.00	38.68	✘	TEDV [31]
26	12.02	12.43	3.75	10.00	0.00	0.00	0.00	0.00	38.20	✘	BREW-DLHPM [62]
27	9.57	11.43	7.06	10.00	0.00	0.00	0.00	0.00	38.06	✘	BMTDL [61]
28	8.62	19.13	0.00	10.00	0.00	0.00	0.00	0.00	37.74	✘	SOM-CNN [35]
29	10.53	16.15	0.00	10.00	0.00	0.00	0.00	0.00	36.68	✘	MDLS [59]
30	4.26	16.65	1.45	0.00	13.84	0.00	0.00	13.84	36.20	✘	SGMM-SOD [95]
31	8.94	15.03	1.62	10.00	0.00	0.00	0.00	0.00	35.58	✔	SMSOM-BM [36]
32	0.00	0.00	8.72	10.00	0.00	16.03	0.00	16.03	34.75	✘	BMOG [96]
33	4.89	19.63	0.00	10.00	0.00	0.00	0.00	0.00	34.52	✘	Block-based RPCA [54]
34	0.00	0.00	11.96	10.00	12.52	0.00	0.00	12.52	34.48	✘	CDPS [97]
35	4.26	20.00	0.00	10.00	0.00	0.00	0.00	0.00	34.26	✘	DMIEC [26]
36	5.64	17.95	0.00	10.00	0.00	0.00	0.00	0.00	33.59	✘	GMRF y HTNN [41]
37	5.00	18.45	0.00	10.00	0.00	0.00	0.00	0.00	33.45	✘	MFESVV [27]
38	9.04	14.23	0.00	10.00	0.00	0.00	0.00	0.00	33.27	✘	TVRPCA [57]
39	5.96	17.08	0.00	10.00	0.00	0.00	0.00	0.00	33.03	✘	EVIEVV [64]
40	7.34	15.78	0.00	0.00	0.00	0.00	9.90	9.90	33.02	✘	GMM & SURF combination [98]
41	6.81	14.35	1.31	10.00	0.00	0.00	0.00	0.00	32.47	✘	SGGMM and Uncertains [50]
42	3.94	18.33	0.00	10.00	0.00	0.00	0.00	0.00	32.26	✘	GC STViBe [45]
43	7.66	14.60	0.00	10.00	0.00	0.00	0.00	0.00	32.26	✘	IISC [56]
44	5.43	16.53	0.00	10.00	0.00	0.00	0.00	0.00	31.95	✘	DMW [65]
45	3.83	16.40	1.06	10.00	0.00	0.00	0.00	0.00	31.29	✘	ISURF [46]
46	0.00	0.00	18.23	0.00	12.90	0.00	0.00	12.90	31.13	✘	PSP-MRF [99]
47	4.57	16.40	0.00	10.00	0.00	0.00	0.00	0.00	30.97	✘	AFPA [12]
48	4.57	15.53	0.00	10.00	0.00	0.00	0.00	0.00	30.10	✘	BRTF [67]
49	0.00	0.00	2.22	10.00	0.00	17.73	0.00	17.73	29.94	✘	CwisarDRP [100]
50	4.68	15.15	0.00	10.00	0.00	0.00	0.00	0.00	29.83	✘	PS-RBM [39]
51	3.30	16.53	0.00	10.00	0.00	0.00	0.00	0.00	29.82	✘	FMR [10]
52	7.45	11.80	0.00	10.00	0.00	0.00	0.00	0.00	29.25	✘	ERCI [28]
53	4.04	14.73	0.00	10.00	0.00	0.00	0.00	0.00	28.77	✘	DRCIMF [25]
54	6.28	12.93	0.00	0.00	0.00	0.00	9.41	9.41	28.62	✘	One-class classification [101]
55	0.00	0.00	0.00	10.00	0.00	18.27	0.00	18.27	28.27	✘	SaliencySuBSENSE [no ref]
56	0.00	0.00	0.00	10.00	0.00	18.09	0.00	18.09	28.09	✘	M4CD Version 2.0 [no ref]
57	3.62	15.15	0.00	0.00	0.00	0.00	9.23	9.23	27.99	✘	Robust low rank matrix decomposition [102]
58	3.19	14.73	0.00	10.00	0.00	0.00	0.00	0.00	27.92	✘	DTCNN [32]
59	5.96	11.93	0.00	10.00	0.00	0.00	0.00	0.00	27.88	✘	ConvNets [38]
60	0.00	0.00	0.00	10.00	0.00	17.55	0.00	17.55	27.55	✘	C-EFIC [103]
61	8.09	9.45	0.00	10.00	0.00	0.00	0.00	0.00	27.54	✘	EAMV [29]
62	10.32	7.15	0.00	10.00	0.00	0.00	0.00	0.00	27.47	✔	COROLA [55]

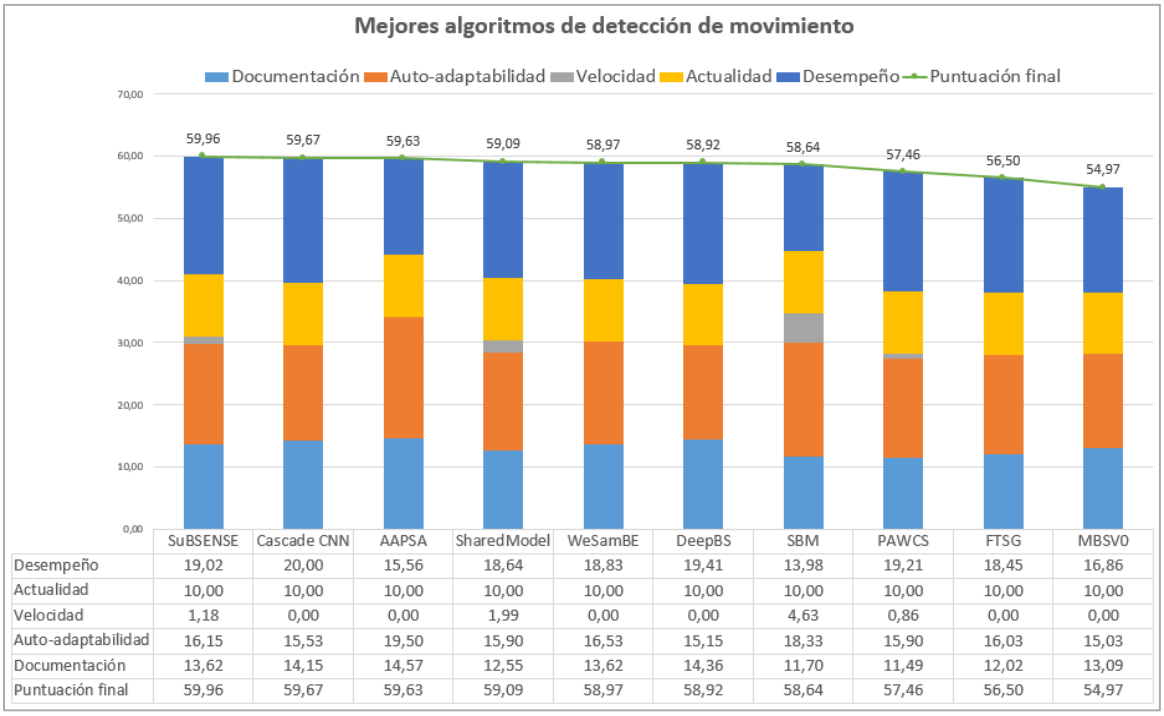
Tabla 4.16-C Algoritmos del 63 al 112 ordenados de acuerdo a sus puntuaciones obtenidas.

Ranking	Documentación $P_{DOCUMENTACIÓN}$	Adaptabilidad $P_{AUTO-ADAPTABILIDAD}$	Velocidad $P_{VELOCIDAD}$	Actualidad $P_{RECIENTES}$	Desempeño				Puntuación final P_{FINAL}	Código disponible	Lista de algoritmos
					CD2012 PBD_2	CD2014 PBD_1	BMC PBD_3	$P_{DESEMPEÑO}$			
63	4.79	12.93	0.00	0.00	0.00	0.00	9.70	9.70	27.42	✗	BC, or Bayesian classification [104]
64	0.00	0.00	0.00	10.00	0.00	17.37	0.00	17.37	27.37	✗	M4CD Version 1.0 [no ref]
65	6.28	11.00	0.00	10.00	0.00	0.00	0.00	0.00	27.28	✗	SiftFlow [47]
66	6.28	10.88	0.00	0.00	0.00	0.00	10.00	10.00	27.15	✗	Statistical local difference pattern [105]
67	7.02	8.33	1.79	10.00	0.00	0.00	0.00	0.00	27.14	✗	RBFMD [42]
68	7.55	9.45	0.00	10.00	0.00	0.00	0.00	0.00	27.00	✗	FCDH [53]
69	0.53	16.53	0.00	0.00	0.00	0.00	9.80	9.80	26.86	✓	VuMeter [106]
70	0.00	0.00	0.00	10.00	0.00	16.69	0.00	16.69	26.69	✗	EFIC [107]
71	4.79	11.80	0.00	10.00	0.00	0.00	0.00	0.00	26.59	✗	K-SVD [63]
72	0.00	0.00	13.29	0.00	12.39	0.00	0.00	12.39	25.68	✗	Chebyshev prob. with Static Object detection [108]
73	5.74	9.88	0.00	10.00	0.00	0.00	0.00	0.00	25.62	✗	AEMF [48]
74	0.00	0.00	0.00	10.00	0.00	15.40	0.00	15.40	25.40	✗	GraphCutDiff [109]
75	0.00	0.00	0.00	10.00	12.15	14.79	0.00	14.79	24.79	✗	RMoG [110]
76	0.00	0.00	9.69	0.00	10.14	15.09	0.00	15.09	24.78	✗	Mahalanobis distance [111]
77	0.00	0.00	0.00	10.00	0.00	14.50	0.00	14.50	24.50	✗	CP3-online [112]
78	0.00	0.00	0.00	10.00	0.00	13.93	0.00	13.93	23.93	✗	Multiscale Spatio-Temporal BG Model [113]
79	0.00	0.00	0.00	10.00	13.57	0.00	0.00	13.57	23.57	✗	SBBS [no ref]
80	0.00	0.00	0.00	10.00	13.03	0.00	0.00	13.03	23.03	✗	CwisarD [no ref]
81	0.00	0.00	10.76	0.00	11.79	0.00	0.00	11.79	22.55	✓	KNN [70]
82	0.00	0.00	7.85	0.00	10.87	14.35	0.00	14.35	22.20	✓	GMM | Stauffer & Grimson [114]
83	0.00	0.00	0.77	10.00	11.10	0.00	0.00	11.10	21.87	✗	KDE - Spatio-temporal [115]
84	10.64	0.00	0.00	10.00	0.00	0.00	0.00	0.00	20.64	✗	NeRM [37]
85	0.00	0.00	0.00	10.00	10.45	0.00	0.00	10.45	20.45	✓	pROST [116]
86	7.55	2.25	0.00	10.00	0.00	0.00	0.00	0.00	19.80	✗	MKFC [52]
87	7.87	2.25	0.00	0.00	0.00	0.00	9.61	9.61	19.73	✗	Temporal saliency [117]
88	7.34	2.30	0.00	10.00	0.00	0.00	0.00	0.00	19.64	✗	SAG [30]
89	0.00	0.00	5.15	0.00	10.03	13.79	0.00	13.79	18.94	✗	Euclidean distance [111]
90	0.00	0.00	6.35	0.00	12.27	0.00	0.00	12.27	18.62	✗	GRBM [no ref]
91	0.00	0.00	0.00	0.00	0.00	17.91	0.00	17.91	17.91	✗	Superpixel Strengthen Background Subtraction [no ref]
92	0.00	0.00	2.46	0.00	15.00	0.00	0.00	15.00	17.46	✗	Cdet [no ref]
93	0.00	0.00	2.74	0.00	11.21	14.65	0.00	14.65	17.38	✓	KDE – ElGammal [118]
94	0.00	0.00	5.72	0.00	11.55	0.00	0.00	11.55	17.27	✗	GRBM_without tuning [no ref]
95	5.85	1.13	0.00	10.00	0.00	0.00	0.00	0.00	16.98	✗	Normalized cut [58]
96	0.00	0.00	0.00	0.00	14.26	16.52	0.00	16.52	16.52	✗	Spectral-360 [119]
97	0.00	0.00	3.04	0.00	13.43	0.00	0.00	13.43	16.47	✓	PBAS [120]
98	0.00	0.00	4.17	0.00	11.67	0.00	0.00	11.67	15.84	✗	SGMM [121]
99	0.00	0.00	0.00	0.00	12.77	14.94	0.00	14.94	14.94	✗	SOBS_CF [122]
100	0.00	0.00	0.00	0.00	0.00	14.21	0.00	14.21	14.21	✗	DCB [no ref]
101	0.00	0.00	0.95	0.00	13.16	0.00	0.00	13.16	14.12	✗	DPGMM [121]
102	0.00	0.00	0.00	0.00	13.30	0.00	0.00	13.30	13.30	✗	GPRMF [123]
103	0.00	0.00	0.00	0.00	12.02	0.00	0.00	12.02	12.02	✗	Multi-Layer Background Subtraction [124]
104	0.00	0.00	0.00	0.00	11.90	0.00	0.00	11.90	11.90	✓	SOBS [125]
105	0.00	0.00	0.00	0.00	11.44	0.00	0.00	11.44	11.44	✗	KDE – ISF [126]
106	0.00	0.00	0.00	0.00	10.98	0.00	0.00	10.98	10.98	✗	Bayesian Background [127]
107	0.00	0.00	0.00	0.00	10.77	0.00	0.00	10.77	10.77	✗	Local-Self similarity [128]
108	0.00	0.00	0.00	0.00	10.66	0.00	0.00	10.66	10.66	✗	GMM | RECTGAUSS-Tex [129]
109	0.00	0.00	0.00	0.00	10.34	0.00	0.00	10.34	10.34	✗	TUBITAK UZAY 1 [130]
110	0.00	0.00	0.00	0.00	10.24	0.00	0.00	10.24	10.24	✗	Histogram [131]
111	0.00	0.00	0.00	0.00	0.00	0.00	9.14	9.14	9.14	✗	NA, or Naive approach [no ref]
112	0.00	0.00	0.00	0.00	0.00	0.00	9.04	9.04	9.04	✗	CB, or Codebooks [14]

A continuación, se presentan los mejores algoritmos de detección de movimiento de acuerdo al ranking de la evaluación de los criterios de análisis en la tabla 4.16, estos algoritmos son representados mediante el conjunto MA_{DM} definido en la ecuación 4.34.

$$MA_{DM} = \{SuBSENSE,\ Cascade\ CNN,\ AAPSA,\ FTSG,\ MBSV\ 0 \\ WeSamBE,\ DeepBS,\ SBM,\ PAWCS,\ SharedModel\} \tag{4.34}$$

En la figura 4.9 se muestran las puntuaciones obtenidas por los algoritmos MA_{DM} en cada uno de los criterios de evaluación. Se puede observar que en general, estos diez algoritmos comparten la característica de ser actuales y tener una buena documentación. También, en cuanto a velocidad únicamente los cuatro algoritmos *SuBSENSE*, *SharedModel*, *SBM* y *PAWCS* pueden realizar un procesamiento en tiempo real, siendo *SBM* el más veloz. De los primeros tres lugares se observa que *SuBSENSE* y *Cascade CNN* tienen muy buenos desempeños en comparación a *AAPSA*, sin embargo, *AAPSA* logra colocarse en la tercera posición por su comportamiento auto-adaptivo. En cuanto a las puntuaciones obtenidas en el criterio desempeño, ninguno de los algoritmos logro tener más de 20 puntos, siendo 30 la máxima, lo que implica que ninguno evaluó su desempeño en las tres bases de datos CD2012, CD2014 y BMC.



	SuBSENSE	Cascade CNN	AAPSA	SharedModel	WeSamBE	DeepBS	SBM	PAWCS	FTSG	MBSV0
Desempeño	19,02	20,00	15,56	18,64	18,83	19,41	13,98	19,21	18,45	16,86
Actualidad	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00	10,00
Velocidad	1,18	0,00	0,00	1,99	0,00	0,00	4,63	0,86	0,00	0,00
Auto-adaptabilidad	16,15	15,53	19,50	15,90	16,53	15,15	18,33	15,90	16,03	15,03
Documentación	13,62	14,15	14,57	12,55	13,62	14,36	11,70	11,49	12,02	13,09
Puntuación final	59,96	59,67	59,63	59,09	58,97	58,92	58,64	57,46	56,50	54,97

Figura 4.9 Puntuaciones finales de los algoritmos MA_{DM}.

Continuando con los resultados obtenidos, ahora se presentan los diez mejores algoritmos con código disponible a manera de destacar los métodos con las mejores características pero que además cuenten con la ventaja de tener la manera de realizar experimentos y pruebas con ellos, con el fin de seguir desarrollando nuevas técnicas, efectuar comparaciones y promover el avance del desarrollo científico.

Los mejores algoritmos con código disponible son representados mediante el conjunto MCD_{DM} definido en la ecuación 4.35.

$$MCD_{DM} = \{SuBSENSE, CascadeCNN, AAPSA, PAWCS, SMSOM\text{-}BM, COROLA, GMM\text{-}KaewTraKulPong, GMM\text{-}Zivkovic, VariacionViBE, VuMeter\} \quad (4.35)$$

A continuación, en la tabla 4.17 se muestran los algoritmos con código disponible MCD_{DM} acomodados de mayor a menor desempeño. Se puede observar que existe una diferencia muy marcada entre el primero y el último lugar en la puntación total, debido a que el algoritmo *VuMeter* no cuenta con una buena puntuación en las categorías documentación, velocidad, actualidad y desempeño. Por otro lado, *SuBSENSE* muestra un buen desempeño en todas las categorías. Es importante señalar que los primeros cuatro algoritmos MCD_{DM} también se encuentran dentro de los algoritmos MA_{DM}

Tabla 4.17 Puntuaciones obtenidas por los algoritmos MCD_{DM}.

Ranking	Documentación $P_{DOCUMENTACIÓN}$	Adaptabilidad $P_{AUTO\text{-}ADAPTABILIDAD}$	Velocidad $P_{VELOCIDAD}$	Actualidad $P_{RECIENTES}$	Desempeño				Puntuación final P_{FINAL}	Código disponible
					CD2012 PBD_2	CD2014 PBD_1	BMC PBD_3	$P_{DESEMPEÑO}$		
1	13.62	16.15	1.18	10.00	14.70	19.02	0.00	19.02	59.96	SuBSENSE
2	14.15	15.53	0.00	10.00	0.00	20.00	0.00	20.00	59.67	Cascade CNN
3	0.00	0.00	0.00	10.00	0.00	0.00	0.00	0.00	10.00	AAPSA
8	11.49	15.90	0.86	10.00	14.85	19.21	0.00	19.21	57.46	PAWCS
11	3.30	14.48	16.40	0.00	11.32	0.00	9.32	20.64	54.82	GMM | KaewTraKulPong
22	2.45	14.48	3.38	0.00	10.55	14.07	9.51	23.58	43.88	GMM | Zivkovic
23	6.81	9.45	14.76	10.00	0.00	0.00	0.00	0.00	41.02	Variación ViBe
31	8.94	15.03	1.62	10.00	0.00	0.00	0.00	0.00	35.58	SMSOM-BM
62	10.32	7.15	0.00	10.00	0.00	0.00	0.00	0.00	27.47	COROLA
69	0.53	16.53	0.00	0.00	0.00	0.00	9.80	9.80	26.86	VuMeter

4.2 Análisis enfocado en modelado de fondo

El análisis enfocado en algoritmos de modelado de fondo, se realizó de una manera más reducida que el de detección de movimiento debido a que los algoritmos de modelado de fondo a diferencia de los de detección de movimiento no realizan la segmentación binaria que identifica al fondo de los objetos en movimiento, únicamente se centran en obtener una estimación del modelo de fondo y en la mayoría de los casos se evalúan con videos de duración corta donde se pretende llegar a un fondo de referencia. Por lo tanto, mientras los algoritmos de detección de movimiento se evalúan para probar su capacidad de enfrentarse a diferentes situaciones críticas de video en aplicaciones reales, los algoritmos de modelado de fondo se evalúan para probar su habilidad de obtener un modelo de fondo inicial o de referencia que sea útil para facilitar la detección de movimiento.

Para el desarrollo del análisis de los algoritmos enfocados en detección de movimiento únicamente se consideraron los algoritmos dentro de las bases de datos que reportan evaluaciones completas, tal es el caso de *SBMnet* y *SBI*.

El conjunto de algoritmos en análisis C_{BD} quedó definido mediante la ecuación 4.36:

$$C_{BD} = \{c \mid c \text{ es uno de algoritmos en las bases de datos SBMnet ó SBI}\} \tag{4.36}$$

C_{BD} se conforma por 24 algoritmos de la base de datos *SBMnet* y 15 algoritmos de *SBI*. Ambos conjuntos comparten 5 algoritmos en común y en total se tienen 34 algoritmos en análisis, lo anterior se ilustra en la figura 4.10

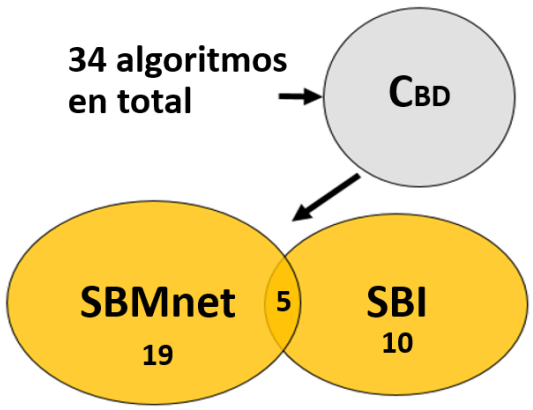


Figura 4.10 Composición del conjunto en análisis C_{BD}.

4.2.1 Metodología de selección de los mejores algoritmos

Para el desarrollo de la metodología de selección de los algoritmos de modelado de fondo C_{BD} que tuvieran los mejores desempeños, se consideraron los criterios velocidad, desempeño y actualidad. No se utilizaron los criterios auto-adaptabilidad y documentación ya que el tiempo necesario en realizarla las evaluaciones sería muy extenso. Al igual que la metodología de detección de movimiento en la sección 4.1.1, la selección de los mejores algoritmos se realizó mediante el uso de pesos en los criterios en evaluación. Recapitulando, los puntos asignados a los pesos de los criterios se muestran en la tabla 4.18.

Tabla 4.18 Pesos otorgados a los criterios de evaluación para los algoritmos de modelado de fondo.

Pesos otorgados a criterios	
Criterios de evaluación	**Puntos máximos por criterio**
Código Disponible	**NA**
Actualidad (P_{Ac})	10
Velocidad (P_{Ve})	25
Desempeño (P_{De})	30
TOTAL	65

En la figura 4.11 se muestra el proceso de selección de los mejores algoritmos donde se tienen las puntuaciones $S_{ACTUALIDAD}$, $S_{VELOCIDAD}$ y $S_{DESEMPEÑO}$ derivadas de los criterios actualidad, velocidad y desempeño respectivamente. Después, mediante la suma de estas puntuaciones se obtendrán las puntaciones finales de los algoritmos S_{FINAL} y finalmente se ordenan de mayor a menor puntuación para seleccionar los diez métodos con mejores características

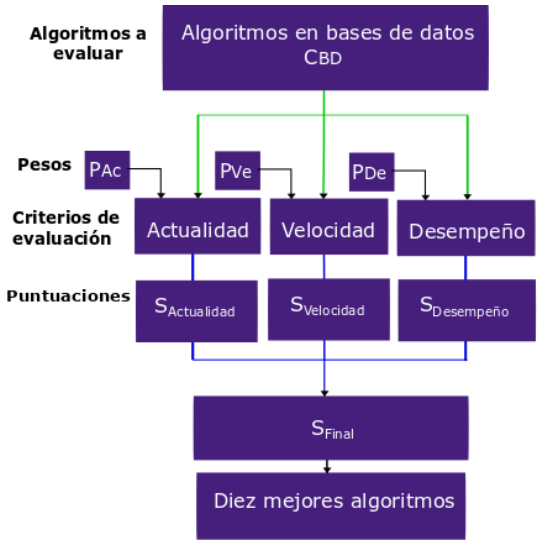


Figura 4.11 Metodología de selección de los mejores algoritmos.

A continuación, en las siguientes secciones se describen los criterios de evaluación y se detalla cómo se generan las puntuaciones otorgadas a los algoritmos en cada uno de ellos.

4.2.1.1 Desempeño

El criterio desempeño evalúa los algoritmos C_{BD}, los cuales están situados en las bases de datos SMBnet y SBI. SBMnet provee una colección de 79 videos divididos en 8 categorías con diversas situaciones críticas, movimiento intermitente, vibraciones, cambios de iluminación, fondos dinámicos, entre otros. Los videos fueron obtenidos con diferentes cámaras de baja a mediana resolución, como consecuencia la resolución espacial varia de 240x240 a 800x600. Algunos videos fueron tomados de bases de datos públicas como *Change detection*, BMC2012, VSSN, SABS, LASIESTA, LIMU, CMU, ICRA, IPPR, CIRL, ATON, UCF, MIT, Fish4Knowledge y PETS. La evaluación se realiza mediante 6 métricas distintas y se otorga un ranking con base en el promedio general.

SBI por su parte cuenta con 14 videos con situaciones más específicas, un hombre entrando y saliendo de la habitación, hombre moviéndose enfrente de un pizarrón generando sombras, carros estacionados en la mayoría de los cuadros y luego empiezan a moverse, etc. Las resoluciones de video varían de 144x144 a 800x600. Las métricas usadas en la evaluación son las mismas que reporta SBMnet, AGE, pEPs, pCEPs, PSNR, MS-SSIM y CQM. Al igual que SBMnet los videos son obtenidos de diversas bases de datos públicas, PBI, Candela, CAVIAR, COST 211, ATON, son algunas de las que figuran.

Para la evaluación del criterio desempeño, primero se realizó un análisis con las bases de datos previamente descritas, con el fin de conocer cual poseía mayor cantidad de elementos para realizar sus evaluaciones y de esta manera poder definir una puntuación para identificar las bases de datos más robustas. Luego, el peso del criterio desempeño P_{De} (véase tabla 4.18) es repartido en las dos bases de datos para otorgar una puntuación máxima a los algoritmos de acuerdo a la base de datos donde se encuentren evaluados. La idea de repartir los puntos, es que la puntuación máxima que pueda alcanzar un algoritmo dependa de la cantidad de bases de datos que se evaluó y el desempeño obtenido en las mismas.

En el análisis de las bases de datos se consideraron los aspectos de: número de videos NV_S, *Ground truth* por video usados para evaluar el desempeño del algoritmo $GTPV_S$ y número de

métricas utilizadas *NMs*, después se sumaron las puntuaciones y se obtuvo la puntuación total T_S por base de datos, mostrada en la ecuación 4.37.

$$T_S = NV_S + GTPV_S + NM_S \tag{4.37}$$

Los puntos T_S obtenidos por cada base de datos se observan en la tabla 4.19.

Tabla 4.19 Puntos otorgados a las bases de datos de modelado de fondo de acuerdo a sus características.

Descripción de base de datos de modelado de fondo				
Base de datos	Número de videos NV_S	GT por video, usados para evaluar el desempeño de los algoritmos $GTPV_S$	Número de métricas NM_S	Puntuación total T_S
SBMnet	79	1	6	86
SBI	14	1	6	21

Luego, se procedió a repartir el valor del peso P_{De} en cada una de las dos bases de datos para otorgar una puntuación máxima SM_l, siendo l un índice para diferenciar la base de datos sobre la que se trabaja, a manera que SM_1 y SM_2 representan a las puntuaciones máximas de las bases de datos SBMnet y SBI respectivamente. La repartición de estas puntuaciones es mostrada en tabla 4.20. Los puntos fueron otorgados de manera proporcional de acuerdo a las puntuaciones T_S que obtuvieron las bases de datos, siendo SBMnet superior a SBI. Por lo que, si el algoritmo se evaluó en SBMnet puede alcanzar los 20 puntos, en SBI únicamente 10 y si usa ambas bases de datos 30.

Tabla 4.20 Puntuaciones de acuerdo al número de bases de datos utilizadas.

Índice l	SBI	SBMnet	Puntuación máxima SM_l
1		x	20
2	x		10
Total	x	x	30

Continuando con el proceso de evaluación, para definir las puntuaciones que se les otorgo a los algoritmos se utilizó el ranking o posición que poseen en sus respectivas bases de datos donde se evalúan. Por lo que la puntuación del algoritmo para cada base de datos se define mediante la ecuación 4.38.

$$SBD_l(n) = \begin{cases} SM_l & si \quad n = 1 \\ SBD_l(n-1) * (1-\alpha) & si \quad n > 1 \end{cases} \qquad (4.38)$$
$$Para \quad l = \{1, 2\}$$

Donde $SBD_l(n)$ es la puntuación otorgada al algoritmo en la posición del ranking n, en la l-ésima base de datos y SM_l es la puntuación máxima que puede obtener el algoritmo en la base de datos l (véase tabla 4.26 para consulta). Por su parte, el parámetro α es una constante con valor de 0.01 la cual fue definida experimentalmente, utilizada a manera de que el decremento en las puntuaciones se realice suavemente.

Finalmente, se suman las puntuaciones obtenidas de cada base de datos SBD_l y se obtiene la puntuación final $S_{DESEMPEÑO}$ otorgada a cada algoritmo, mediante la ecuación 4.39.

$$S_{DESEMPEÑO} = \sum_{l=1}^{2} SBD_l \qquad (4.39)$$

A continuación, en la tabla 4.21 se muestran las puntuaciones obtenidas por los algoritmos.

Tabla 4.21-A Puntuaciones de los algoritmos del 1 al 10 en el criterio desempeño a la fecha mayo del 2017.

Ranking	Lista de algoritmos	Desempeño		
		SBMnet SBD_1	SBI SBD_2	$S_{DESEMPEÑO}$
1	LaBGen	19.60	10.00	29.60
2	Bewis	19.80	9.80	29.60
3	Photomontage	19.21	9.51	28.72
4	BE-AAPSA	18.45	8.95	27.41
5	RSL2011	17.37	9.32	26.70
6	MSCL	20.00	0.00	20.00
7	LaBGen-P	19.41	0.00	19.41
8	SC-SOBS-C4	19.02	0.00	19.02
9	MAGRPCA	18.83	0.00	18.83
10	Temporal median filter	18.64	0.00	18.64

Tabla 4.21-B Puntuaciones de los algoritmos del 11 al 34 en el criterio desempeño a la fecha mayo del 2017.

Ranking	Lista de algoritmos	Desempeño		
		SBMnet SBD_1	SBI SBD_2	$S_{DESEMEPEÑO}$
11	Bidirectional Analysis	18.27	0.00	18.27
12	Bidirectional Analysis and Consensus Voting	18.09	0.00	18.09
13	TMFG	17.91	0.00	17.91
14	FC-FlowNet	17.73	0.00	17.73
15	RMAMR	17.55	0.00	17.55
16	AAPSA	17.20	0.00	17.20
17	RMR	17.03	0.00	17.03
18	3-Term Decomposition	16.86	0.00	16.86
19	DECOLOR	16.69	0.00	16.69
20	SSGoDec	16.52	0.00	16.52
21	GOSUS	16.36	0.00	16.36
22	BRTF	16.19	0.00	16.19
23	GRASTA	16.03	0.00	16.03
24	RFSA	15.87	0.00	15.87
25	SC-SOBS_1	0.00	9.90	9.90
26	IMBS-MT	0.00	9.70	9.70
27	SC-SOBS_2	0.00	9.61	9.61
28	WS2006	0.00	9.41	9.41
29	LRGeomCG	0.00	9.23	9.23
30	Tmac	0.00	9.14	9.14
31	MOG2	0.00	9.04	9.04
32	Color Median	0.00	8.86	8.86
33	KNN	0.00	8.78	8.78
34	Mean	0.00	8.69	8.69

4.2.1.2 Velocidad

El criterio velocidad se evaluó sobre los algoritmos C_{BD} con el mismo procedimiento que el descrito para el análisis en detección de movimiento. Primero se normaliza la velocidad a una resolución de 320x240 y luego se ordenan los algoritmos de mayor a menor velocidad de procesamiento. Luego, se filtran con base aquellos que puedan cumplir con un procesamiento en tiempo real.

Por lo tanto, la puntuación final $S_{VELOCIDAD}$ para el criterio velocidad es otorgada mediante la ecuación 4.40, considerando el ranking de los algoritmos que cumplen un procesamiento en tiempo real mostrados en la tabla 4.22.

Tabla 4.22 Algoritmos que pueden realizar procesamiento en tiempo real.

Ranking	FPS (320x240)	Nombre algoritmos	Autor
1	5248	LaBGen	B. Laugraud , S. Pierard , M. Van Droogenbroeck
2	504	LaBGen-P	B. Laugraud, S. Piérard, M. Van Droogenbroeck
3	112	KNN	Z. Zivkovic , F. van der Heijden
4	100	Temporal median filter	Piccardi, Massimo

Entonces:

$$S_{VELOCIDAD}(n) = \begin{cases} P_{Ve} & si \quad n = 1 \\ S_{VELOCIDAD}(n-1) * (1-\beta) & si \quad n > 1 \end{cases} \tag{4.40}$$

Donde, n es la posición en el ranking del algoritmo, P_{Ve} es el peso otorgado al criterio velocidad con valor 25 (véase tabla 4.18) y el parámetro β es una constante con valor de 0.1 definida experimentalmente, utilizada a manera de que el decremento en las puntuaciones $S_{VELOCIDAD}$ se realice suavemente. Si su valor es muy grande, los algoritmos en los últimos lugares del ranking no se llevarían ningún punto y si es muy pequeño, prácticamente se les estaría otorgando a todos los algoritmos la puntuación más alta P_{Ve}.

En la tabla 4.23 se muestran las puntuaciones finales $S_{VELOCIDAD}$ otorgadas en el criterio velocidad.

Tabla 4.23 Puntuaciones de los algoritmos más veloces.

Ranking	FPS (320x240)	Nombre algoritmos	Autor	$S_{VELOCIDAD}$
1	5248	LaBGen	B. Laugraud , S. Pierard , M. Van Droogenbroeck	25
2	504	LaBGen-P	B. Laugraud, S. Piérard, M. Van Droogenbroeck	22,5
3	112	KNN	Z. Zivkovic , F. van der Heijden	20,25
4	100	Temporal median filter	Piccardi, Massimo	18,225

El algoritmo LaBGen reporta un tiempo de procesamiento de 1312 FPS a una resolución de 640x480 en la base de datos SBMnet por lo que al realizar la normalización a la resolución de 320x240 su tiempo de procesamiento será de 5248 FPS, una cantidad bastante elevada al considerar que no se reporta el uso de GPU. Hay que destacar también que LabGen-P del mismo autor obtiene el segundo lugar con una velocidad de 504 FPS.

4.2.1.3 Actualidad

Para la evaluación del criterio actualidad se consideran los algoritmos analizados C_{BD}, estos son clasificados con respecto al año de su publicación y se filtran del periodo 2013-2017.

Para otorgar la puntuación final $S_{ACTUALIDAD}$ al criterio actualidad, se tiene la ecuación 4.41 mostrada a continuación:

$$S_{ACTUALIDAD} = \begin{cases} P_{Ac} & si \rightarrow \quad AP_S \geq 2013 \\ 0 & si \rightarrow \quad AP_S < 2013 \end{cases} \tag{4.41}$$

Donde AP_S representa el año de publicación del artículo y P_{Ac} el peso del criterio actualidad mostrado previamente en la tabla 4.18. En esta evaluación se les dio el mismo valor a todos los algoritmos dentro del periodo 2013-2017. De un total de 34 algoritmos en el análisis, 21 algoritmos quedaron dentro de la clasificación 2013-2017. Los cuales se muestran a continuación en la tabla 4.24.

Tabla 4.24 Puntuación otorgada a los algoritmos dentro del periodo 2013-2017.

Algoritmos más recientes			
Año	**Métodos**	**Autor**	$S_{ACTUALIDAD}$
2017	BE-AAPSA	G. Ramirez-Alonso , J. Ramirez-Quintana , M. Chacon-Murguia	10 puntos generales
	Bewis	M. De Gregorio and M. Giordano	
	IMBS-MT	D. Bloisi , A. Pennisi , L. Iocchi , Parallel	
	LaBGen	B. Laugraud , S. Pierard , M. Van Droogenbroeck	
2016	AAPSA	G. Ramírez et al.	
	Bidirectional Analysis	T. Minematsu, A. Shimada and R-I Taniguchi	
	Bidirectional Analysis and Consensus Voting	T. Minematsu, A. Shimada and R-I Taniguchi	
	BRTF	Q. Zhao et al.	
	FC-FlowNet	I. Halfaoui, F. Bouzaraa and O. Urfalioglu	
	LaBGen-P	B. Laugraud, S. Piérard, M. Van Droogenbroeck	
	MAGRPCA	S Javed, A. Mahmmod	
	MSCL	to be submitted	
	RMR	D. Ortego, J. C. SanMiguel, J. M. Martínez	
	SC-SOBS-C4	L. Maddalena and A.Petrosino	
	TMFG	W. Liu, Y. Cai, M. Zhang, H. Li and H. Gu	
2015	RMAMR	X. Ye	
	Tmac	Y. Xu , R. Hao , W. Yin , Z. Su , Parallel	
2014	RFSA	X. Guo et al	
2013	DECOLOR	X. Zhou, C. Yang, W. Yu	
	GOSUS	J. Xu et al.	
	LRGeomCG	B. Vandereycken	

4.2.1.4 Selección de los mejores algoritmos de modelado de fondo

Mediante las puntuaciones obtenidas para cada uno de los criterios, se definió una puntuación final S_{FINAL} que se observa en la ecuación 4.33.

$$S_{FINAL} = S_{DESEMPEÑO} + S_{VELOCIDAD} + S_{ACTUALIDAD} \tag{4.33}$$

A continuación, en la tabla 4.25, se observan las puntuaciones obtenidas por los algoritmos en cada uno de los criterios y la puntuación final S_{FINAL}.

Tabla 4.25 Puntuaciones finales otorgadas a los algoritmos en análisis.

Ranking	Velocidad $S_{VELOCIDAD}$	Actualidad $S_{ACTUALIDAD}$	Desempeño			S_{FINAL}	Código disponible	Lista de algoritmos nueva
			SBMnet, SBD_1	SBI, SBD_2	$S_{DESEMPEÑO}$			
1	25.00	10.00	19.60	10.00	29.60	64.60	✔	LaBGen [132]
2	22.50	10.00	19.41	0.00	19.41	51.91	✔	LaBGen-P [132]
3	0.00	10.00	19.80	9.80	29.60	39.60	✔	Bewis [40]
4	0.00	10.00	18.45	8.95	27.41	37.41	✔	BE-AAPSA [133]
5	18.23	0.00	18.64	0.00	18.64	36.87	✘	Temporal median filter [134]
6	0.00	10.00	20.00	0.00	20.00	30.00	✘	MSCL [135]
7	20.25	0.00	0.00	8.78	8.78	29.03	✔	KNN [70]
8	0.00	10.00	19.02	0.00	19.02	29.02	✘	SC-SOBS-C4 [136]
9	0.00	10.00	18.83	0.00	18.83	28.83	✘	MAGRPCA [137]
10	0.00	0.00	19.21	9.51	28.72	28.72	✘	Photomontage [138]
11	0.00	10.00	18.27	0.00	18.27	28.27	✘	Bidirectional Analysis [139]
12	0.00	10.00	18.09	0.00	18.09	28.09	✘	Bidirectional Analysis and Consensus Voting [139]
13	0.00	10.00	17.91	0.00	17.91	27.91	✘	TMFG [140]
14	0.00	10.00	17.73	0.00	17.73	27.73	✘	FC-FlowNet [141]
15	0.00	10.00	17.55	0.00	17.55	27.55	✔	RMAMR [142]
16	0.00	10.00	17.20	0.00	17.20	27.20	✔	AAPSA [33]
17	0.00	10.00	17.03	0.00	17.03	27.03	✘	RMR [143]
18	0.00	0.00	17.37	9.32	26.70	26.70	✘	RSL2011 [144]
19	0.00	10.00	16.69	0.00	16.69	26.69	✔	DECOLOR [80]
20	0.00	10.00	16.36	0.00	16.36	26.36	✔	GOSUS [145]
21	0.00	10.00	16.19	0.00	16.19	26.19	✘	BRTF [67]
22	0.00	10.00	15.87	0.00	15.87	25.87	✘	RFSA [146]
23	0.00	10.00	0.00	9.70	9.70	19.70	✔	IMBS-MT [147]
24	0.00	10.00	0.00	9.23	9.23	19.23	✔	LRGeomCG [148]
25	0.00	10.00	0.00	9.14	9.14	19.14	✘	Tmac [149]
26	0.00	0.00	16.86	0.00	16.86	16.86	✔	3-Term Decomposition [148]
27	0.00	0.00	16.52	0.00	16.52	16.52	✔	SSGoDec [150]
28	0.00	0.00	16.03	0.00	16.03	16.03	✔	GRASTA [151]
29	0.00	0.00	0.00	9.90	9.90	9.90	✘	SC-SOBS_1 [152]
30	0.00	0.00	0.00	9.61	9.61	9.61	✘	SC-SOBS_2 [152]
31	0.00	0.00	0.00	9.41	9.41	9.41	✘	WS2006 [153]
32	0.00	0.00	0.00	9.04	9.04	9.04	✘	MOG2 [94]
33	0.00	0.00	0.00	8.86	8.86	8.86	✘	Color Median [no ref]
34	0.00	0.00	0.00	8.69	8.69	8.69	✘	Mean [no ref]

Dentro de los mejores algoritmos de modelado de fondo se destacan los primeros tres *LaBGen*, *LaBGen-P* y *Bewis*. *LaBGen* cuenta con el primer lugar en velocidad, se encuentra evaluado en ambas bases de datos con reportando un buen desempeño, cuenta con la ventaja de tener código disponible aun siendo un método de reciente creación. Por su parte *LaBGen-P* es una modificación a su homónimo, contando con una buena puntuación en tiempo de procesamiento y en la base de datos SBMnet, su puntuación es menor debido a que no se encuentra evaluado en SBI. *Bewis* logra casi un desempeño a la par de *LaBGen*, sin embargo, cuenta con la desventaja de no poder realizar un procesamiento en tiempo real, lo cual impacta en la puntuación final.

A continuación, en la figura 4.12 se observan las mejores características y puntuaciones que poseen los primeros diez algoritmos de modelado de fondo con base en los criterios, desempeño, velocidad y actualidad.

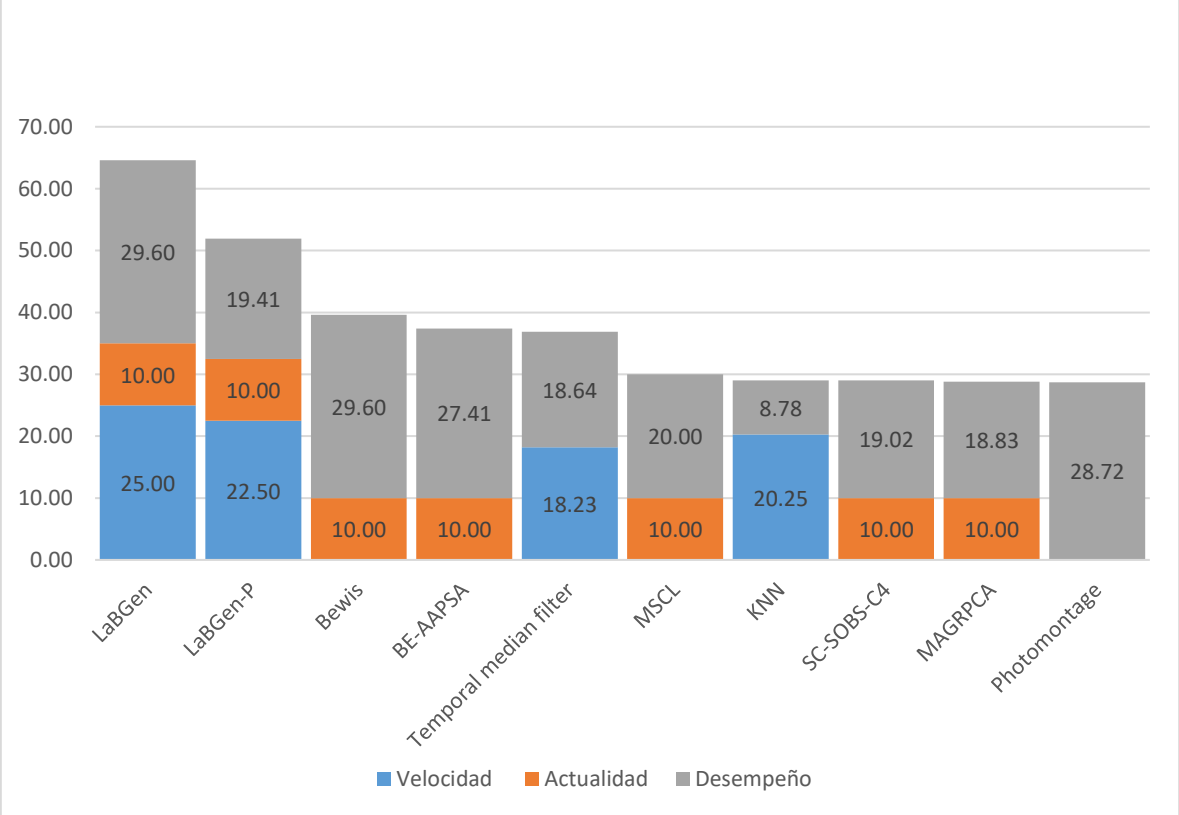


Figura 4.12 Los diez mejores algoritmos de modelado de fondo

4.3 Comparación de los mejores algoritmos obtenidos de la evaluación propuesta contra algoritmos de Change detection

Para concluir el capítulo, se realizó una comparación entre los resultados obtenidos de la evaluación propuesta en este trabajo de tesis mediante los cinco criterios (documentación, auto-adaptabilidad, velocidad, desempeño y actualidad) contra los resultados en la base de datos *Change detection*, con el objetivo de exponer algunas de las situaciones que ocurren cuando se trata de medir el impacto de un algoritmo, centrando su análisis únicamente en el desempeño obtenido en una base de datos o un número de métricas.

A continuación, en la tabla 4.26, se muestran los primeros diez lugares de la base de datos *Change detection* con las puntuaciones y ranking obtenidos en la evaluación propuesta. En rojo se resalta los criterios de evaluación donde no se obtuvieron puntos, en verde se resaltan los diez primeros algoritmos de la base de datos *Change detection* que también están dentro de los diez primeros de la evaluación propuesta, y la flecha al lado del ranking, muestra si el desempeño del algoritmo de *Change detection* subió (↑), bajo (↓) o fue igual (↔), respecto a los resultados de la evaluación propuesta.

Tabla 4.26 Puntuaciones finales otorgadas para los diez primeros lugares de *Change detection*.

Diez primeros lugares CD2014		**Puntuaciones obtenidas en evaluación propuesta**										
							Desempeño					
Ranking CD214	**Algoritmos en CD2014**	**Ranking obtenido en evaluación propuesta**	**Documentación** $P_{DOCUMENTACIÓN}$	**Adaptabilidad** $P_{AUTO\text{-}ADAPTABILIDAD}$	**Velocidad** $P_{VELOCIDAD}$	**Actualidad** $P_{RECIENTES}$	**CD2012** PBD_2	**CD2014** PBD_1	**BMC** PBD_3	$P_{DESEMPEÑO}$	**Puntuación final** P_{FINAL}	**Código disponible**
1	Cascade CNN	2↓	14.15	15.53	0.00	10.00	0.00	20.00	0.00	20.00	59.67	✔
2	IUTIS-5	15↓	10.96	11.55	0.00	10.00	0.00	19.80	0.00	19.80	52.31	✘
3	IUTIS-3	16↓	10.96	11.55	0.00	10.00	0.00	19.60	0.00	19.60	52.11	✘
4	DeepBS	6↓	14.36	15.15	0.00	10.00	0.00	19.41	0.00	19.41	58.92	✘
5	PAWCS	8↓	11.49	15.90	0.86	10.00	14.85	19.21	0.00	19.21	57.46	✔
6	SuBSENSE	1↑	13.62	16.15	1.18	10.00	14.70	19.02	0.00	19.02	59.96	✔
7	WeSamBE	5↑	13.62	16.53	0.00	10.00	0.00	18.83	0.00	18.83	58.97	✘
8	SharedModel	4↑	12.55	15.90	1.99	10.00	0.00	18.64	0.00	18.64	59.09	✘
9	FTSG	9↔	12.02	16.03	0.00	10.00	0.00	18.45	0.00	18.45	56.50	✘
10	SaliencySuBSENSE	55↓	0.00	0.00	0.00	10.00	0.00	18.27	0.00	18.27	28.27	✘

De la tabla anterior, se observa que de los diez mejores algoritmos de la base de datos *Change detection* (mostrados en la segunda columna de la parte izquierda de la tabla), los algoritmos IUTIS-5, IUTIS-3 y SaliencySuBSENSE, no figuran dentro de los diez primeros de la evaluación propuesta. Debido a que IUTIS-5 y IUTIS-3, no tienen muy buenas puntuaciones en los criterios documentación y auto-adaptabilidad (Apéndices A y B

respectivamente). En la parte de documentación existen varios detalles que no son reportados en sus artículos, tales como tiempo de procesamiento, resoluciones de video utilizadas y aspectos que conciernen en la parte de la implementación del método. Por otro lado, en la parte de auto-adaptabilidad, se identificó que ITUIS-5 y IUTIS-3 no eran en sí, algoritmos enfocados a detección de movimiento, son algoritmos de programación genética que dependen de las máscaras binarias de resultados de los mejores métodos de detección de movimiento para obtener un mejor resultado mediante la combinación de estas. Por lo que, IUTIS-3 es la combinación de métodos como SuBSENSE, FTSG y CwisarDH, y IUTIS-5 es la combinación de las máscaras binarias de SuBSENSE, FTSG, CwisarDH, Spectral-360 y AMBER. Por lo tanto, ambos métodos no son auto-adaptivos, cuentan con algunos parámetros automáticos, sin embargo, dependen de parámetros como el número de algoritmos usados para realizar el experimento y el número máximo de generaciones o combinaciones posibles, así como una gran cantidad de entrenamiento; usando el video de duración más corta en cada una de las categorías de video de la base de datos Change detection para entrenar el algoritmo y luego probándose con el resto.

Por otro lado, el método SaliencySuBSENSE muestra en la base de datos Change detection que su artículo fue subido a CVPR2015, cuyos autores son Zhenkun huang Ruimin Hu and Shihong Chen. Sin embargo, no fue localizado para realizar su evaluación completa en los criterios documentación, auto-adaptabilidad y velocidad, por lo que, quedo en ceros en las tres evaluaciones al no poder confirmarse la fuente.

El resto de los primeros diez algoritmos en Change detection, Cascade CNN, DeepBS, PAWCS, SuBSENSE, WeSamBE, SharedModel y FTSG también figuraron dentro de los diez primeros algoritmos por la evaluación propuesta, solo que, con posiciones en el ranking distintas, debido a que la evaluación propuesta considera otros aspectos además del desempeño del algoritmo. Su ranking dentro de la evaluación propuesta se observa en la parte izquierda de la tabla 4.32 en color verde.

La evaluación propuesta con respecto a Change detection tuvo 7 algoritmos de 10, dentro de los mejores algoritmos de detección de movimiento, lo que representa un 70% de coincidencia. Implicando que aún y cuando Change detection es una base de datos comúnmente usada como un método rápido de conocer y ubicar el impacto e importancia de

un algoritmo, no es correcto considerarla como la única alternativa para realizar comparaciones entre algoritmos, ya que en aplicaciones reales importan de igual o mayor manera aspectos como la velocidad del algoritmo, la auto-adaptabilidad y su documentación. Sin una buena documentación el algoritmo carece de relevancia científica, ya que no se existe forma de demostrar aspectos concernientes a la problemática que resuelve y/o el impacto de la aportación, lo cual impide a su vez el avance científico. Por su parte, la auto-adaptabilidad implica el grado de autonomía que tiene un algoritmo para resolver diferentes situaciones críticas sin la intervención humana, lo cual es enfocado en que se traten de proponer algoritmos que logren emular el pensamiento humano para hacerle frente a los problemas que se suscitan en aplicaciones reales de video. Por ejemplo, de nada serviría que un algoritmo tenga el mejor desempeño en detectar el movimiento cuando existen condiciones ambientales de luz de día, si en condiciones de noche o lluvia falla y se tiene que corregir el error cambiando sus parámetros manualmente, esto solo llevaría a tener un sistema de detección de movimiento impráctico y engorroso. Finalmente, la velocidad es una característica que implica si un algoritmo puede ser usado o no en aplicaciones en tiempo real, lo cual es un rasgo que se busca en un algoritmo. Por lo tanto, para generar comparaciones acertadas y justas entre algoritmos, se deben considerar cuando menos los aspectos mencionados anteriormente.

En este capítulo se explicó la metodología realizada para la selección de los mejores algoritmos de detección y modelado de fondo, proponiendo el uso de cinco criterios de evaluación: documentación, auto-adaptabilidad, velocidad, desempeño y actualidad. Después, se obtuvieron los mejores algoritmos de cada evaluación otorgando puntuaciones en cada uno de los criterios. Las puntuaciones finales se pueden observar en las tablas 4.16 y 4.31, respectivamente para los algoritmos de detección y modelado de fondo. Finalmente se compararon los resultados obtenidos de los diez mejores algoritmos MA_{DM} en el análisis de detección de movimiento contra la base de datos *Change detection*. Lo anterior se muestra en la tabla 4.32, en la cual, siete de los primeros diez algoritmos de la base de datos *Change detection* están dentro de los primeros diez lugares de la evaluación propuesta.

CAPÍTULO V. EVALUACIÓN DE ALGORITMOS CONTRA CRITERIO DE SEGMENTACIÓN HUMANA

Por mucho tiempo, la evaluación de los algoritmos orientados a la detección de movimiento se ha realizado mediante métricas enfocadas a medir la cantidad de pixeles clasificados de manera correcta con respecto a una imagen de referencia o *Ground truth*. La imagen de referencia se realiza normalmente de forma manual por algún experto, donde se discriminan los objetos en movimiento del escenario de fondo. Sin embargo, este proceso se vuelve subjetivo, arrojando resultados relativos a la observación del experto.

Hoy en día existe una gran variedad de bases de datos que cuentan con diferentes escenas en la**s** que se incluyen situaciones críticas, presentes en circunstancias reales de toma de video. Una de las bases de datos más reportada en la literatura es *Change detection*, debido a que encapsula y contiene un riguroso compendio de videos divididos en once categorías, provee un amplio agregado de *Ground truth*, y además realiza evaluaciones y comparaciones entre algoritmos mediante siete métricas enfocadas a medir el desempeño. Sin embargo, aún y cuando bases de datos como *Change detection* ofrecen una evaluación completa y extendida, existe una tendencia por parte de los investigadores a evaluar el algoritmo y reportar resultados de acuerdo a las bases de datos sin el pleno conocimiento o certeza de si la información puesta en ellas o si las métricas utilizadas para medir el desempeño del algoritmo son las correctas.

Con el objetivo de atender la problemática antes descrita, en este capítulo, se proponen dos experimentos. El primero consiste en evaluar el desempeño humano contra los *Ground truth* de la base de datos *Change detection* con la finalidad de medir el grado de error humano o varianza que existe al realizar un *Ground truth*. El segundo evalúa el desempeño de los diez mejores algoritmos de detección de movimiento MA_{DM} (obtenidos en la sección 4.1.1.6) contra los resultados obtenidos por las segmentaciones humanas con el propósito de obtener una referencia de cómo se sitúan los algoritmos respecto al humano y si existen ocasiones en las que los algoritmos dan mejores resultados. Además, para realizar la evaluación del desempeño en ambos experimentos, se propuso el uso de dos métricas enfocadas a medir la calidad de segmentación *D-Score* [22] y *SSIM* [21], y la métrica poco estudiada *FSD* [24], la cual permite combinar información estadística y estructural mediante la unión de las métricas

SSIM, *D-Score* y *F-Measure*, lo anterior con el objetivo de realizar una evaluación más extendida y medir el impacto que se obtiene sobre las demás métricas reportadas en la base de datos *Change detection*. A continuación, se explica la metodología usada en la realización de los experimentos.

- **Metodología propuesta para la realización de los experimentos**

Para el desarrollo de los experimentos se optó por elegir la base de datos *Change detection* ya que, tal como se vio en la sección 4.1.1.3 es la más completa y reportada en la literatura, debido a que utiliza 7 métricas para medir el desempeño y encapsula 53 videos divididos en 11 categorías, con 4 o 6 videos en cada una, además, otorga los resultados (imágenes binarias de detección) obtenidos en cada uno de los videos por los algoritmos ahí evaluados. Por lo tanto, debido a que se usaron los algoritmos MA_{DM} para realizar los experimentos y nueve de ellos se encuentran evaluados en esta base de datos, fue ventajoso el contar con sus resultados de detección para elegir un video por cada categoría para realizar los experimentos. Se decidió seleccionar un video por cada categoría, debido a que sería muy extensa la evaluación al usar todos y cada uno de ellos. Los videos fueron elegidos de acuerdo a la moda de los mejores desempeños obtenidos por los algoritmos MA_{DM} para cada categoría de video. Después, para cada video seleccionado se eligieron aleatoriamente tres cuadros. Los cuadros y videos seleccionados son mostrados en la tabla 5.1.

Tabla 5.1 Videos y cuadros elegidos en orden aleatorio usados para realizar las evaluaciones.

SELECCIÓN VIDEOS Y CUADROS A SEGMENTAR				
Categoría	**Video**	**Cuadro 1**	**Cuadro 2**	**Cuadro 3**
Baseline	office	1932	724	1406
Dynamic Background	fountain02	694	745	1236
Camera Jitter	badminton	1100	879	1125
Intermittent Object Motion	streetLight	288	728	1735
Shadow	backdoor	1380	1719	1897
Thermal	library	2377	4156	1524
Bad Weather	skating	1371	1003	1476
Low Framerate	turnpike_0_5fps	809	1036	979
Nigth Videos	tramStation	1575	593	1530
PTZ	twoPositionPTZCam	973	1188	1295
Turbulence	turbulence3	1120	1008	1259

Luego, cinco personas fueron designadas para realizar las segmentaciones correspondientes a los cuadros elegidos. Por lo tanto, cada sujeto segmentó un total de 33 cuadros, tres por cada uno de los 11 videos elegidos en cada una de las categorías. Estas segmentaciones fueron usadas para realizar las comparaciones y evaluaciones contra los

Ground truth de *Change detection*. Los *Ground truth*, al igual que los videos y resultados de los algoritmos MA_{DM} para realizar los experimentos, fueron obtenidos de *Change detection*.

Los *Ground truth* de *Change detection* idealmente deberían ser realizados por un número de diferentes personas y los resultados de las segmentaciones deberían ser promediados para poder obtener una segmentación representativa. Sin embargo, debido a la cantidad de videos y cuadros con los que cuenta, esto resulta impráctico, por el tiempo y recursos necesarios para realizarlas. Además, es muy difícil para una persona producir un *Ground truth* que no cree controversia, debido a la distorsión por desenfoque de movimiento y objetos parcialmente opacos ya que los pixeles en esas zonas contendrán objetos en movimiento y fondo al mismo tiempo y como consecuencia existirá un cierto nivel de incertidumbre. A manera de resolver esta y otras situaciones se originaron zonas con distinto nivel de gris dentro de los *Ground truth* de *Change detection* con las siguientes etiquetas:

- Estáticas: valor en escala de gris 0.
- Sombras: valor en escala de gris 50.
- Zonas de no interés: valor en escala de gris 85.
- Desconocido: valor en escala de gris 170.
- Movimiento: valor en escala de gris 255.

En la figura 5.1 se ven representadas las diferentes zonas en escala de gris en los *Ground truth* de *Change detection*.

Figura 5.1 Interpretación de los niveles de gris dentro de los *Ground truth* de *Change detection*.

Como parte de la metodología usada en la realización de los experimentos, se siguió el esquema establecido en *Change detection* y sólo fueron consideradas las zonas de movimiento y estáticas como *Ground truth*. Además, tanto en las segmentaciones humanas como en los resultados de los algoritmos, no se consideran las zonas de no interés para realizar las evaluaciones.

En resumen, en el primer experimento se evalúan y analizan los resultados de las segmentaciones humanas contra los *Ground truth* de *Change detection* en los cuadros y videos elegidos (mostrados en la tabla 5.1), y el segundo experimento corresponde a los resultados de los algoritmos MA_{DM} y las segmentaciones humanas contra los *Ground truth* de la misma base de datos. La metodología para el desarrollo de los experimentos antes descrita, se encuentra representada en la figura 5.2. En las siguientes secciones se analizan los resultados obtenidos de ambos experimentos.

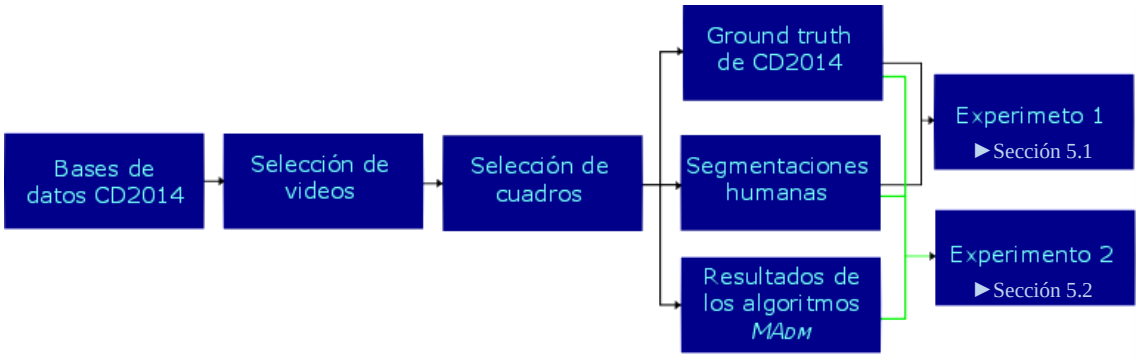


Figura 5.2 Diagrama de la metodología usada para la realización de los experimentos.

5.1 Segmentaciones humanas contra *Change detection*

Las segmentaciones humanas realizadas por los cinco sujetos se evaluaron con respecto a los *Ground truth* de *Change detection* mediante las métricas colocadas en la misma base de datos, más las métricas enfocadas a medir la calidad de segmentación *D-Score* y *SSIM*, y la métrica *FSD* con el fin de realizar una evaluación más extendida y evaluar el comportamiento de estas métricas, con respecto a las que comúnmente son reportadas en la literatura para realizar las comparaciones entre algoritmos.

Los resultados generales por cada uno de los cinco sujetos en todos los videos, se muestran a continuación en la tabla 5.2 ordenados mediante el ranking promedio obtenido sobre todas

las métricas. En verde se muestran los resultados más altos para cada métrica y en rojo los más bajos. También, se agregaron dos filas al final de la tabla con la finalidad de facilitar el análisis (remarcadas en amarillo). La fila de promedio, indica el promedio de desempeño humano para cada una de las métricas y la fila variación muestra la diferencia entre el mejor y peor resultado de los sujetos para cada una de las métricas (en azul se muestran las métricas con mayor variación).

Tabla 5.2 Resultados de cada uno de los sujetos sobre todos los videos.

Segmentaciones humanas vs *Change detection*											
Sujetos	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	SSIM	D-Score	FSD	Average Ranking
Sujeto5	0.968	0.999	0.001	0.032	0.299	0.963	0.966	0.961	0.004	0.974	1.700
Sujeto1	0.970	0.997	0.003	0.031	0.419	0.929	0.946	0.958	0.005	0.967	2.700
Sujeto2	0.952	0.997	0.003	0.048	0.514	0.939	0.942	0.962	0.004	0.966	3.200
Sujeto3	0.969	0.997	0.003	0.031	0.500	0.929	0.946	0.957	0.005	0.966	3.500
Sujeto4	0.974	0.996	0.004	0.026	0.473	0.917	0.942	0.956	0.006	0.964	3.900
Promedio	0.967	0.997	0.003	0.034	0.441	0.936	0.948	0.959	0.005	0.967	
Variación	0.021	0.003	0.003	0.021	0.215	0.046	0.024	0.006	0.002	0.010	

Como se observa en la tabla 5.2, el comportamiento de los cinco sujetos de prueba fue muy similar, a excepción del sujeto cinco el cual muestra una mejor puntuación en 7 de las 10 métricas disponibles. A manera general, existe una variación más alta en los resultados de los sujetos para las métricas *Recall*, *FNR, Precision*, *F-Measure* y *PWC* en comparación a las demás métricas. En la métrica *F-Measure*, los sujetos dos y cuatro obtuvieron el menor desempeño con 0.942 en comparación con el sujeto cinco, el cual obtuvo el desempeño más alto con 0.966, esto resulta debido a que la métrica *F-Measure* es derivada de las métricas *Recall* y *Precision*, siendo los sujetos dos y cuatro los que obtuvieron menores desempeños para cada una respectivamente. Por su parte PWC es la métrica que muestra la mayor variación entre el mejor y más bajo desempeño obtenido por los sujetos, teniendo una variación de 0.215, donde nuevamente el sujeto cinco se posiciona en primer lugar en esta métrica y el sujeto dos en último. Cabe destacar que el sujeto cinco tenía conocimientos previos acerca de la realización de *Ground truth*, mientras que los otros cuatro sujetos eran inexpertos, lo que explica el porqué de las variaciones entre el desempeño más alto y más bajo son tan marcadas para algunas de las métricas mencionadas. Para el resto de las métricas existe una variación, por debajo de 0.01 por lo que, es muy poco significativa.

A continuación, en la figura 5.3 se observa el comportamiento de las segmentaciones humanas con respecto a las métricas, donde el sujeto cinco visiblemente se encuentra por encima del promedio humano, en especial en las métricas *PWC, Precision* y *F-Measure.*

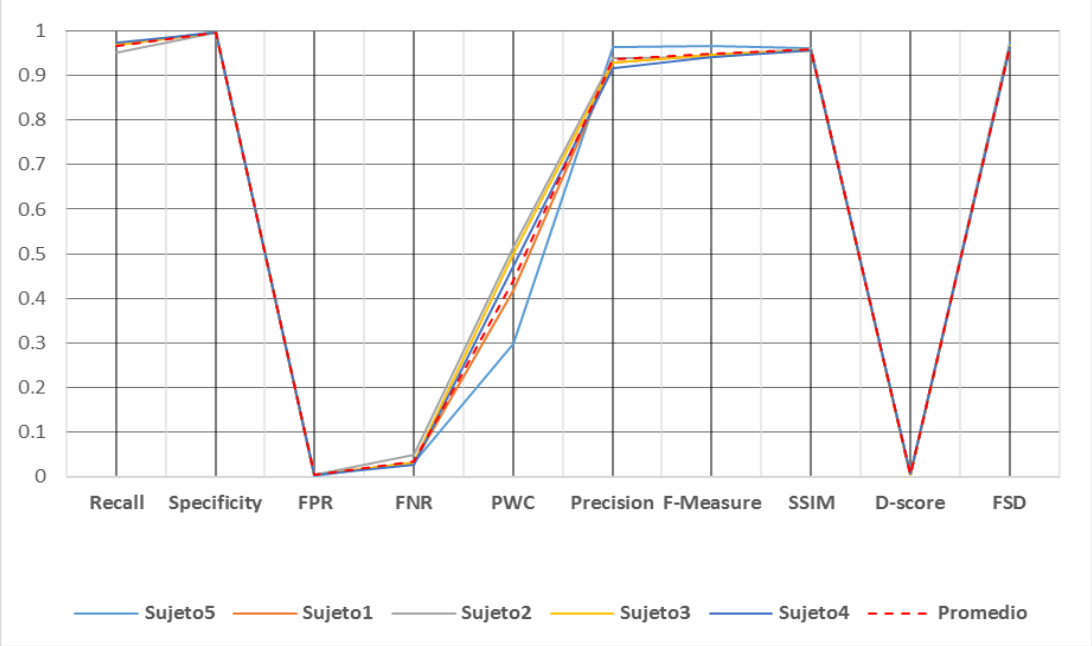


Figura 5.3 Comportamiento de las segmentaciones humanas con respecto a las métricas.

A continuación, se realiza un análisis respecto a dos de las métricas más comúnmente usadas en la literatura *PWC* y *F-Measure* contra las métricas enfocadas en medir la calidad de la segmentación *SSIM* y *D-Score*.

La métrica *PWC* es una métrica enfocada a medir el porcentaje de clasificación errónea, por lo que, cuanto más cercano a cero su valor, mejor será el desempeño del algoritmo. *F-Measure* es una métrica que mide el desempeño combinando *Precision* y *Recall*. Donde la *Precision* representa la repetitividad de la medida y qué cantidad de pixeles son relevantes, y el *Recall*, mide la relevancia de los mismos y la capacidad de detectar resultados positivos correctamente, en ambas métricas entre más cercanas a uno, mejor será el desempeño. La situación ideal sería que los resultados de las métricas *Recall* y *Precision* fueran uno, sin embargo, en situaciones reales siempre existirá un *trade-off* entre ellas. Con el objetivo de ponderar y observar que tan lejanas se encuentran ambas medidas del valor ideal, se suele combinar los resultados de ambas en la métrica llamada *F-Measure*, la cual, entre más

cercano a uno sea su valor mejor será el desempeño obtenido. De acuerdo a los resultados obtenidos de los cinco sujetos en la tabla 5.2, en promedio se obtuvo 0.441 y 0.948 respectivamente para las métricas *PWC* y *F-Measure*, lo que implicar que cualquier resultado obtenido por encima de estos valores, caerá dentro del desempeño de lo que se considera error humano. Lo anterior es congruente con lo reportado en el artículo [84], en él, se realiza un experimento similar, donde se seleccionaron 77 cuadros representativos de *Change detection* y tres personas realizaron las segmentaciones. Los resultados del experimento fueron que ninguna de las personas obtuvo un *F-Measure* mayor a 0.96 y en promedio se obtuvo un *F-Measure* de 0.948 y un PWC de 0.87. También en el mismo artículo toman el *Ground truth* y lo dilatan o erosionan en valores de 1 o 2 pixeles, con el propósito de demostrar que una simple dilatación (o erosión) de un pixel en la segmentación, puede no ser muy significativa para afectar la calidad de la imagen, sin embargo, da como resultado un *F-Measure* de 0.94 y un PWC de 0.31. Los resultados de las dilataciones y erosiones reportadas en el artículo [84], se muestran en la tabla 5.3.

Tabla 5.3 Resultados al erosionar o dilatar el Ground truth en el experimento de [84].

METODO	#PIXEL	F-Measure	PWC
Dilatación	1	0.94	0.31
	2	0.88	0.73
Erosión	1	0.93	0.33
	2	0.86	0.63

Por su parte, en los resultados de la tabla 5.2 para las métricas *SSIM* y *D-Score*, ninguno de los sujetos obtuvo un resultado mayor a 0.96 en SSIM o menor de 0.004 en *D-Score* y el promedio las segmentaciones humanas fue 0.959 y 0.005 respectivamente. *SSIM* está enfocada a medir la calidad de la segmentación mediante la información estructural, bajo la asunción de que la precepción visual humana está altamente adaptada a extraer información estructural de la escena. *D-Score* se enfoca en evaluar la distancia de la localización de los pixeles mal clasificados de acuerdo a la posición real del objeto de interés en el *Ground truth*, es una métrica de similitud para imágenes binarias basada en una transformación de distancia. Para las métricas SSIM y *D-Score* no se tiene una gran variación entre los resultados de los cinco sujetos (véase tabla 5.2) y los resultados obtenidos en ambas métricas son casi siempre superiores a los resultados de las métricas PWC y F-Measure, lo cual indica que, a nivel

estructural, la calidad de la segmentación entre los sujetos es muy similar y más cercana al *Ground truth*. Por otro lado, las métricas *PWC* y *F-Measure* miden a nivel pixel las variaciones que existen entre las segmentaciones humanas y el *Ground truth*, por lo que, es congruente que se presente más variación en los resultados de estas métricas y un desempeño menor. Para ejemplificar los resultados obtenidos en las métricas anteriores, se analiza la figura 5.4, donde se tomó al azar un video de los seleccionados (véase tabla 5.1) y a su vez un cuadro del mismo. Luego, se evaluaron las métricas de ese cuadro con respecto a la segmentación del sujeto 3, el cual fue elegido aleatoriamente, esto con el fin de mostrar las diferencias entre lo que toman en consideración las cuatro métricas anteriores. Los resultados obtenidos por el sujeto 3 de acuerdo a la figura 5.4 se muestran en la tabla 5.4.

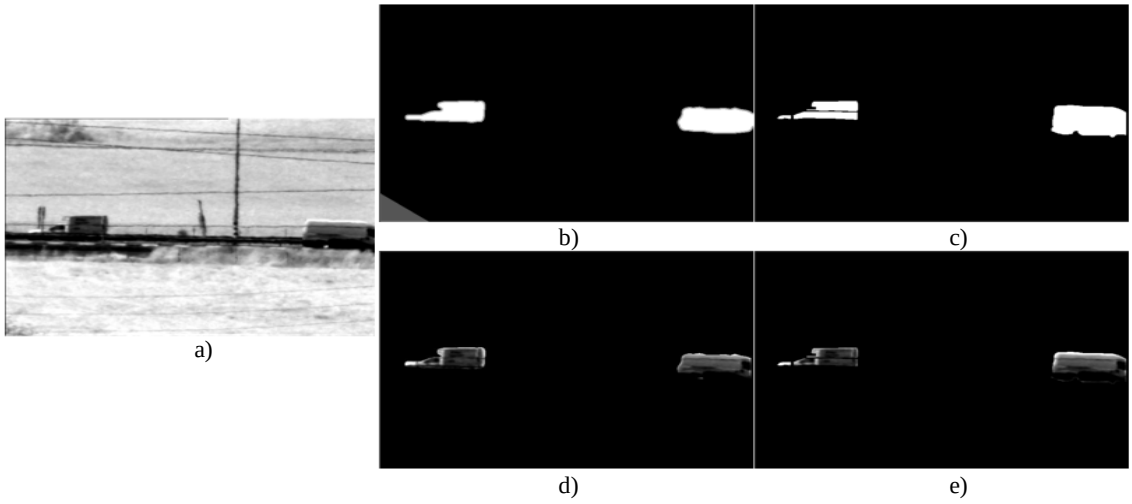


Figura 5.4 a) Cuadro 1008 del video turbulence3, b) Ground truth CD2014, c) Segmentación humana, d) Ground truth CD2014 b) multiplicado por el cuadro de video a), e) Segmentación humana c) multiplicada por el cuadro de video a).

Tabla 5.4 Resultados del sujeto tres en el cuadro 1008 del video turbulence3.

Métrica	Valor
SSIM	0.972
F-Measure	0.876
D-Score	0.006
PWC	0.913

En la figura 5.4c se puede observar la segmentación realizada por el sujeto 3, en la cual se pueden apreciar que existe una diferencia respecto al *Ground truth* de *Change detection* de la figura 5.4b y es fácil apreciar las zonas donde existen pixeles mal clasificados. No obstante, a nivel estructural, la imagen del sujeto 3 (figura 5.4e) contra la imagen del *Ground truth*

(figura 5.4d) es un poco más similar y visualmente es fácil identificar los automóviles en la escena. De la figura 5.4, las imágenes *d)* y *e)* son el resultado de la multiplicación de *b)* y *c)* por la imagen *a)* respectivamente.

Los resultados obtenidos del experimento en la tabla 5.4 muestran que en las métricas a nivel pixel *F-Measure* y *PWC* se obtuvo un desempeño de 0.876 y 0.913 respectivamente, y en las métricas enfocadas a medir la calidad se obtiene un desempeño mayor con 0.972 y 0.006 respectivamente de *SSIM* y *D-Score*. Por lo tanto, si se toma en consideración una evaluación a nivel pixel de las imágenes *c)* y *b)* de la figura 5.4 existirá un desempeño más bajo en los resultados, ya que se evalúa y se toma en consideración el total de pixeles correctamente clasificados con respecto a un *Ground truth* y resulta muy difícil obtener un resultado idéntico. Así que, cualquier variación mínima produce un impacto considerable en los resultados. Sin embargo, si la evaluación es a nivel estructural y se toman como ejemplo las imágenes *d)* y *e)* de la figura 5.4, se obtendrán mejores resultados, ya que lo que se evalúa en sí, es la facilidad con la que se puede identificar los objetos de interés con respecto al *Ground truth* y en particular estas imágenes resultan visualmente similares. Por lo tanto, las métricas *SSIM* y *D-Score* aunque no son tan estrictas como las métricas F-Measure o PWC que evalúan el número total de pixeles correctamente clasificados, deberían de ser más reportadas en las evaluaciones de algoritmos ya que finalmente lo que se busca de un algoritmo de detección de movimiento no es que cantidad de pixeles puede detectar correctamente, si no que los pixeles que detecte sean significativos y visualmente se logre identificar el objeto de interés. Después de todo, en la mayoría de los casos, la detección de movimiento es el paso previo al seguimiento (tracking) y el realizar el seguimiento es un proceso más flexible, donde únicamente se requiere un grupo de pixeles (blob) que identifiquen al objeto de interés, sin importar el número de pixeles sobrantes o faltantes del mismo.

Como parte de otro análisis, se obtuvo el promedio general de los cinco sujetos para cada una de las categorías de video con el fin de generalizar su comportamiento e identificar las categorías de video que resultaban más fáciles de segmentar y en las cuales se tenía mayores problemas. Después, se obtuvo el ranking promedio sobre todas las métricas y finalmente se

ordenó de menor a mayor. La tabla 5.5, muestra el promedio de las segmentaciones humanas con respecto las categorías de video ordenadas con base al ranking promedio.

Tabla 5.5 Resultados del promedio de los cinco sujetos en cada categoría de video.

Promedio humano en cada categoría de video												
Categorías	Videos	Promedio										
		Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	SSIM	D-Score	FSD	Average Ranking
Baseline	Office	1.000	0.999	0.001	0.000	0.072	0.990	0.995	0.943	0.005	0.978	2.400
Intermi…	SreetLight	0.996	0.997	0.003	0.004	0.260	0.966	0.980	0.986	0.001	0.989	4.100
PTZ	TwoPositionPTZCam	0.973	0.998	0.002	0.027	0.234	0.942	0.957	0.985	0.002	0.980	5.000
Shadow	Backdoor	0.979	0.999	0.001	0.021	0.276	0.982	0.980	0.942	0.005	0.972	5.100
CameraJ…	Badminton	0.999	0.998	0.002	0.001	0.209	0.933	0.965	0.955	0.005	0.971	5.200
Dynamic…	Fountain02	0.905	0.999	0.001	0.095	0.222	0.937	0.920	0.980	0.003	0.966	6.000
BadWeat…	Skating	0.993	0.998	0.002	0.007	0.289	0.980	0.986	0.933	0.006	0.971	6.200
Turbule…	Turbulence	0.866	0.999	0.001	0.134	0.341	0.903	0.883	0.988	0.003	0.956	7.200
Thermal	Library	0.957	0.998	0.002	0.043	1.143	0.993	0.975	0.935	0.007	0.967	7.300
LowFram…	Trunpike_0_5fps	0.991	0.996	0.004	0.009	0.422	0.964	0.977	0.921	0.010	0.963	8.100
NigthVi…	Tramstation	0.972	0.987	0.013	0.028	1.385	0.701	0.811	0.979	0.005	0.928	9.400
Promedio		0.966	0.997	0.003	0.034	0.441	0.936	0.948	0.959	0.005	0.967	

Los resultados de la tabla 5.5 indican que el video *Office* de la categoría *Baseline* fue donde los sujetos de prueba obtuvieron el mejor desempeño. Este video contiene una escena donde una persona toma un libro y lo hojea; el movimiento en la escena es mínimo y no se observan problemas debidos a objetos dinámicos. Por lo que, no representó un gran problema el realizar la segmentación de este video. Los únicos inconvenientes fueron partes donde se observa un cierto camuflaje en la escena entre el pantalón de la persona y una mesa, por lo que en estas zonas fue difícil el clasificar cuales pixeles pertenecían al objeto en movimiento y cuales al fondo. Lo anterior se observa en la figura 5.5.

Por otro lado, el peor desempeño se obtuvo en el video *Tramstation* de la categoría *NigthVideos*. En él, se observa una escena de noche, la cual incluye problemas de sombras con automóviles en una carretera y peatones. En este video es evidente que el mayor inconveniente que tuvieron los sujetos al realizar la segmentación se encuentra en el cuadro 593 mostrado en la figura 5.6, ya que la mayoría de los sujetos consideraron gran parte de la sombra del automóvil como objeto en movimiento, el único que claramente no la consideró fue el sujeto cinco, quien contaba con experiencia previa al realizar los *Ground truth.* También, en el mismo video se observan algunos problemas debido a camuflaje debido a que

es una escena de noche y los peatones caminan con ropa oscura, por lo que cada sujeto segmentó lo que para su juicio consideró que encerraba al objeto de interés.

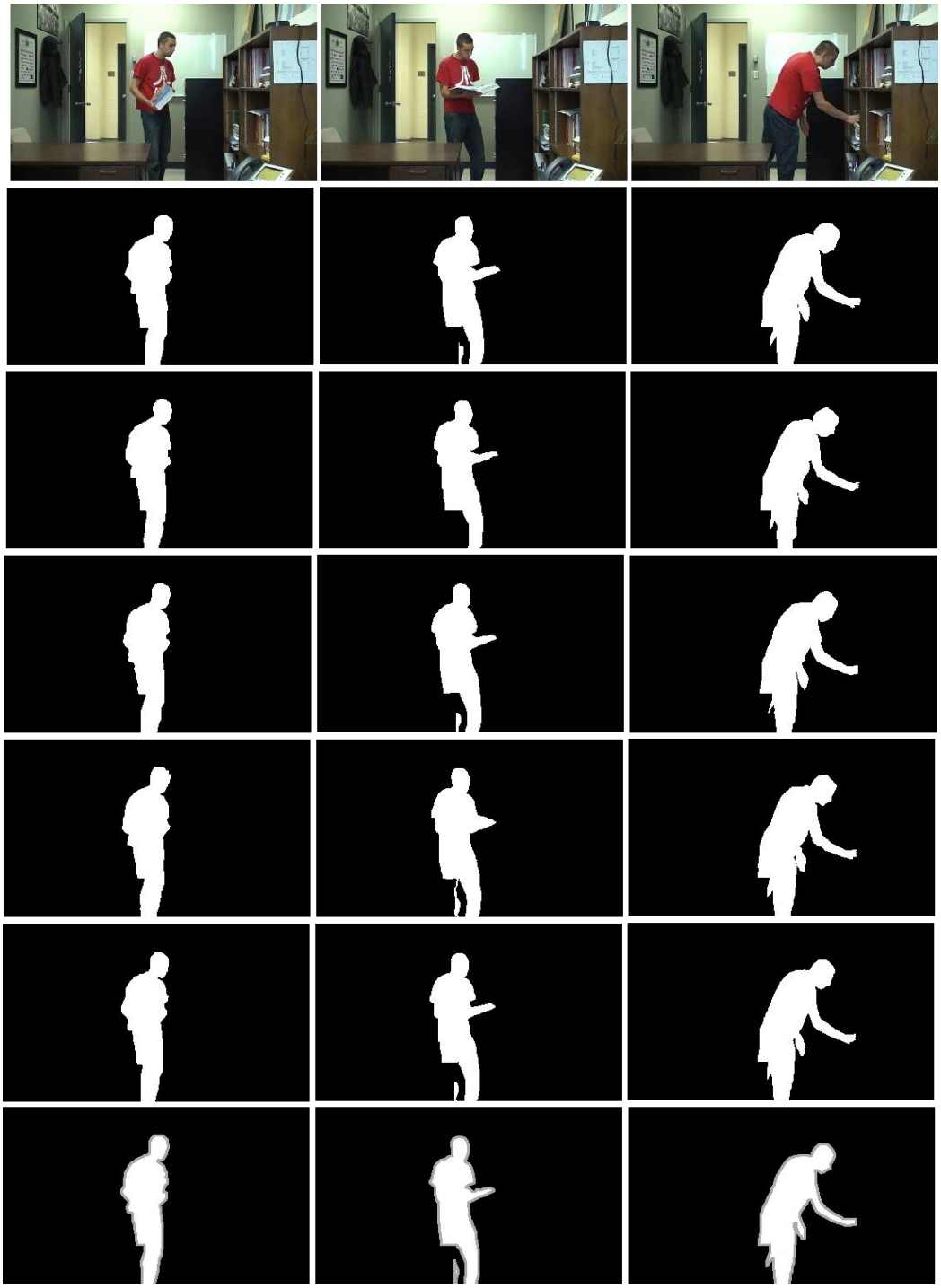


Figura 5.5 Segmentaciones manuales contra *Ground truth* de *Change detection* en el video *Office*. En la primera fila se observa los cuadros 1932, 724 y 1406 del video *Office* en RGB, las filas consecutivas corresponden a las segmentaciones de los sujetos del primero al quinto, en ese orden, finalmente la última fila pertenece al *Ground truth*.

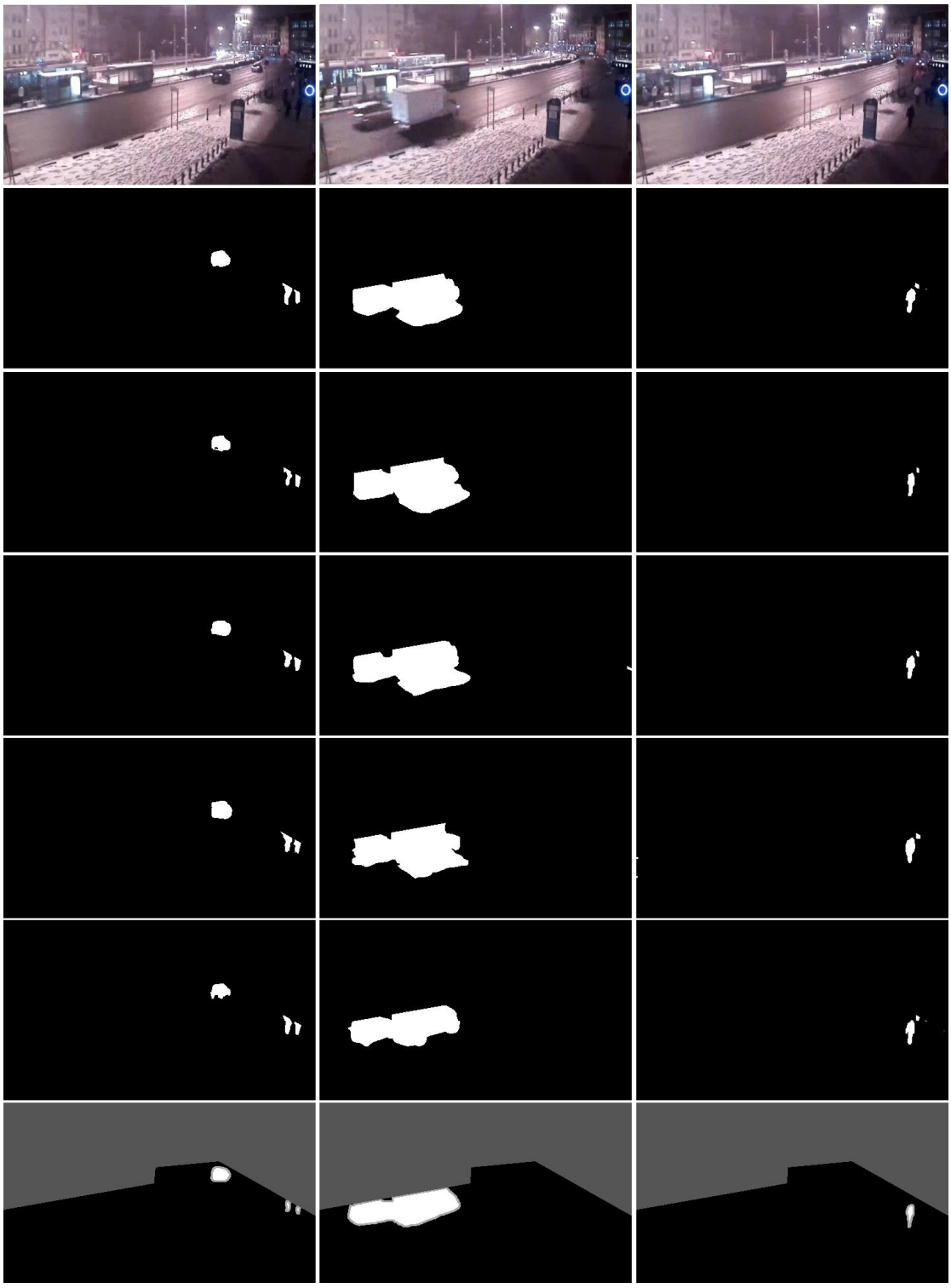


Figura 5.6 Segmentaciones manuales contra *Ground truth* de *Change detection* en el video *Tramstation*. En la primera fila se observa los cuadros 1575, 593 y 1530 del video en RGB, las filas consecutivas corresponden a las segmentaciones de los sujetos del primero al quinto, en ese orden, finalmente la última fila pertenece al *Ground truth.*

Como anteriormente se mencionó, los *Ground truth* de *Change detection* contienen partes en tono de gris que son consideradas como zonas fuera de los límites de interés, de manera que las segmentaciones humanas fueron procesadas para omitir estas zonas, en la figura 5.6 se observa además de lo descrito anteriormente, el resultado de este proceso.

De la tabla 5.5 y las figuras anteriores 5.5 y 5.6, se observa que el proceso de realizar un *Ground truth* siempre llevará consigo un cierto grado de error, debido a que cada persona tiene un juicio o percepción distinta de lo que consideran debería ser clasificado como objeto en movimiento y lo que debería ser fondo. Estas diferencias entre cada persona se presentan con mayor grado en escenas donde existen camuflajes, sombras, vibraciones, fondos dinámicos, entre otras, ya que en estas situaciones se tiene un mayor grado de incertidumbre. Por lo que, estas categorías de video no figuran dentro de los tres primeros lugares de la tabla.

En resumen general de esta sección, en la tabla 5.2, se mostró que los sujetos que realizaron los GT tuvieron una variación más alta en sus resultados para las métricas *Recall*, *FNR, Precision, F-Measure* y *PWC.* El sujeto cinco que contaba con entrenamiento previo acerca de la realización de *Ground truth* se posicionó en primer lugar en el ranking, obteniendo los mejores desempeños en siete de las diez métricas disponibles en evaluación. Por otro lado, se realizó una comparación de las métricas F-Measure y PWC contra las métricas enfocadas en medir la calidad de la imagen SSIM y *D-Score*. Los resultados de estas comparaciones muestran que las métricas *F-Measure* y *PWC* realizan una evaluación muy estricta, ya que evalúan el número de pixeles (de fondo u objetos dinámicos) correctamente clasificados. Por su parte las métricas SSIM y *D-Score* son más flexibles ya que se enfocan en medir la calidad de segmentación de la imagen, tomando en consideración información estructural, y finalmente lo que se busca de un algoritmo de detección de movimiento no es que cantidad de pixeles puede detectar correctamente, si no que los pixeles que detecte sean significativos y visualmente se logre identificar el objeto de interés. Por lo que estas dos métricas deberían de ser más reportadas en las evaluaciones de algoritmos. Por último, se realizó un análisis enfocado en detectar aquellas situaciones donde los humanos tuvieran mayores problemas para realizar las segmentaciones y se encontró que al igual que los algoritmos, los humanos tienen problemas al segmentar situaciones críticas en video, en especial en camuflajes y sombras.

5.2 Comparación de resultados de segmentación de algoritmos y humanos contra *Change detection*

En esta sección se evalúa el desempeño de los diez mejores algoritmos de detección de movimiento MA_{DM}, mediante las mismas métricas y cuadros de video elegidos del experimento anterior (véase tabla 5.1). La idea general es evaluar y analizar el desempeño de los algoritmos al incluir las métricas SSIM y *D-Score*, las cuales, no son reportadas en la base de datos *Change detection*. También, los resultados de los algoritmos son comparados con el desempeño humano, con el propósito de contrastar la variación que existe entre ambos e identificar si existen categorías de video en las que los algoritmos logran mejores resultados al del humano.

A continuación, se detallan los resultados generales obtenidos por los algoritmos sobre los videos seleccionados. La tabla 5.6 muestra los resultados ordenados con base en el ranking promedio sobre todas las métricas, en verde se resaltan los mejores resultados en cada métrica y en rojo los más bajos. También, se muestran los resultados de las segmentaciones de los sujetos y el promedio de los cinco, con el fin de realizar una comparativa.

La evaluación del algoritmo SBM no se realizó debido a la carencia de los resultados de su detección (imágenes binarias) para todos los videos, ya que SBM sólo se evaluó en *Change detection 2012*.

Tabla 5.6 Resultados generales de los algoritmos y sujetos sobre todos los videos.

Algoritmo o sujeto	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	SSIM	D-Score	FSD	Average Ranking
Sujeto 5	0.968	0.999	0.001	0.032	0.299	0.963	0.965	0.961	0.004	0.974	1.90
Cascade CNN	0.984	0.997	0.003	0.016	0.409	0.946	0.964	0.959	0.006	0.972	2.80
Promedio humano	0.966	0.997	0.003	0.034	0.441	0.936	0.948	0.959	0.005	0.967	4.40
Sujeto 1	0.969	0.997	0.003	0.031	0.419	0.929	0.946	0.958	0.005	0.967	4.60
Sujeto 2	0.952	0.997	0.003	0.048	0.514	0.939	0.942	0.962	0.004	0.966	4.60
Sujeto 3	0.969	0.997	0.003	0.031	0.500	0.929	0.946	0.957	0.005	0.966	5.50
Sujeto 4	0.974	0.996	0.004	0.026	0.473	0.917	0.942	0.956	0.006	0.964	6.50
PAWCS	0.852	0.997	0.003	0.148	1.068	0.918	0.882	0.955	0.007	0.943	9.50
SuBSENSE	0.889	0.995	0.005	0.111	1.265	0.901	0.891	0.954	0.008	0.946	10.50
DeepBs	0.751	0.998	0.002	0.249	2.654	0.955	0.807	0.949	0.007	0.916	10.60
MBSV 0	0.878	0.994	0.006	0.122	1.135	0.874	0.874	0.955	0.007	0.941	11.30
Shared Model	0.903	0.990	0.010	0.097	1.315	0.886	0.890	0.952	0.010	0.944	11.60
FTGS	0.827	0.995	0.005	0.173	1.402	0.909	0.854	0.952	0.007	0.933	11.80
AAPSA	0.783	0.995	0.005	0.217	1.677	0.850	0.808	0.957	0.006	0.919	12.10
WesSamBe	0.860	0.995	0.005	0.140	1.527	0.899	0.873	0.950	0.008	0.938	12.30
SBM	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	

En la tabla anterior se observa que el algoritmo Cascade CNN obtuvo el mejor desempeño en comparación a los demás algoritmos, incluso está por encima del desempeño promedio de segmentación humano. No obstante, el sujeto cinco se posiciona por encima de éste. Cabe destacar que el sujeto cinco contaba con experiencia en la realización de segmentaciones manuales, por lo que es comprensible que se encuentre en primer lugar de la evaluación. También, al comparar el desempeño de los algoritmos con respecto al ranking promedio, se observa que existe una separación muy marcada entre el desempeño de los algoritmos y la segmentación de los sujetos, a excepción del algoritmo Cascade CNN, el cual cae dentro de lo que podría considerarse como comportamiento humano de sujetos sin entrenamiento. En la figura 5.7 se observa lo anterior con más detalle; en color azul se resaltan los algoritmos, en amarillo las segmentaciones manuales y en naranja el promedio humano.

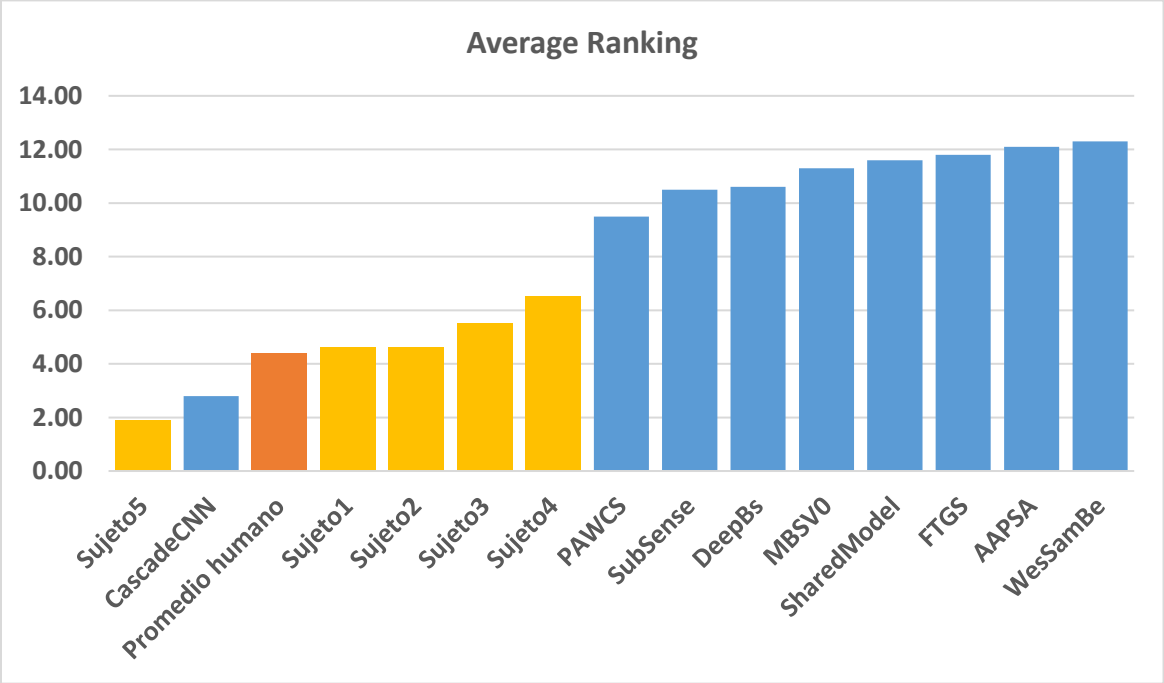


Figura 5.7 Resultados de los algoritmos y segmentaciones manuales respecto al ranking promedio.

Con el propósito de analizar el comportamiento de los algoritmos y el humano con respecto a dos de las métricas más reportadas en la literatura *PWC* y *F-Measure*, y las métricas enfocadas en la calidad *SSIM* y *D-Score*, las cuales se abordaron previamente en la sección 5.1, se graficaron sus resultados por separado en la figura 5.8. En estas figuras se muestran los resultados del desempeño de los algoritmos (en color azul) y el promedio humano (color amarrillo) con respecto a las métricas.

Se observa que, en general, el algoritmo Cascade CNN y el promedio de segmentación humana se mantienen dentro de los primeros tres mejores resultados en todas las métricas. Sin embargo, existen algoritmos que logran posicionarse dentro de los primeros tres lugares en alguna métrica, pero al evaluarse en alguna otra, su desempeño decae considerablemente. En particular, el algoritmo AAPSA se posiciona en tercer lugar dentro de la evaluación de las métricas *SSIM* y *D-Score* pero obtiene el penúltimo lugar en *F-Measure* y *PWC*. Esto implica una diferencia muy marcada en los resultados, la cual, es ocasionada debido a que mientras las métricas *PWC* y *F-Measure* se basan en contar el número de pixeles (fondo u objetos dinámicos) correctamente clasificados, la métrica *SSIM* toma en consideración información estructural de la imagen y *D-Score* la distancia entre los pixeles mal clasificados y el *Ground truth*.

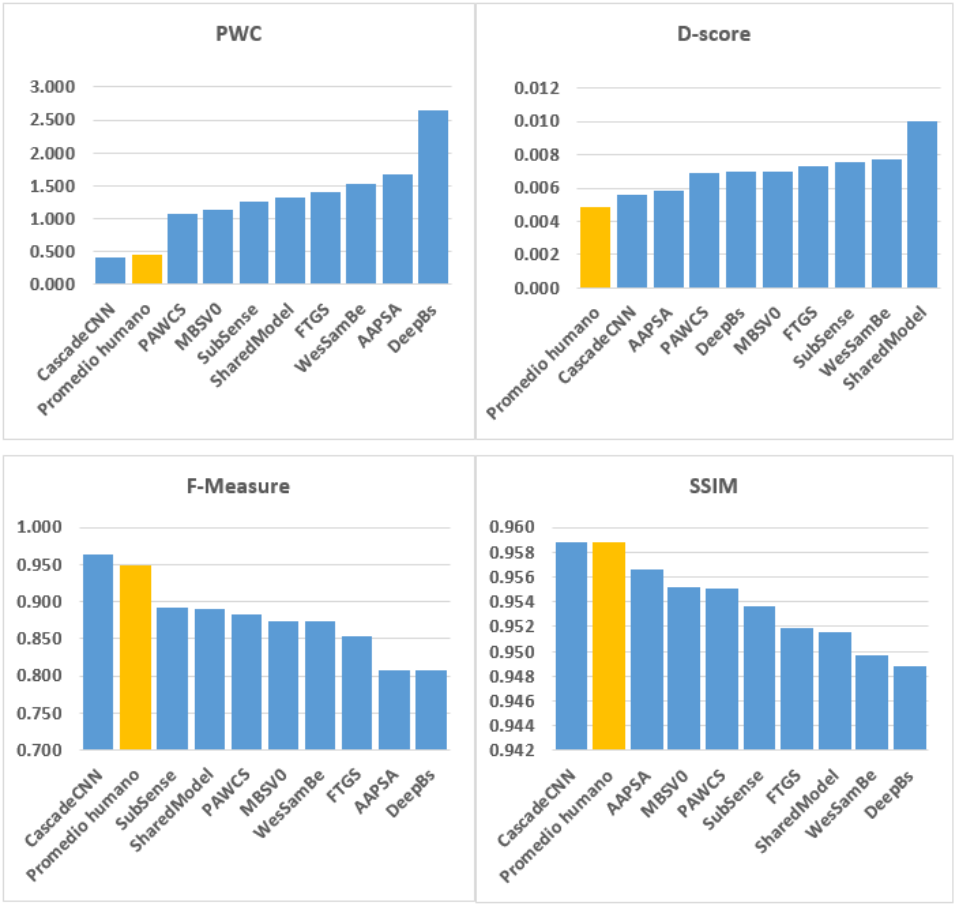


Figura 5.8 Resultados de los algoritmos y humanos en las métricas PWC, D-Score, F-Measure y SSIM.

Uno de los motivos por los que el algoritmo AAPSA obtiene un bajo desempeño en las métricas *F-Measure* y *PWC* es debido a que existen zonas dentro de la segmentación de

objetos dinámicos que no son cubiertas en su totalidad. Al no existir el relleno completo del objeto en movimiento se generan una gran cantidad de falsos negativos que impactan considerablemente en el desempeño de estas métricas. A continuación, en la figura 5.9 se ejemplifica lo anterior en algunos cuadros de diferentes videos.

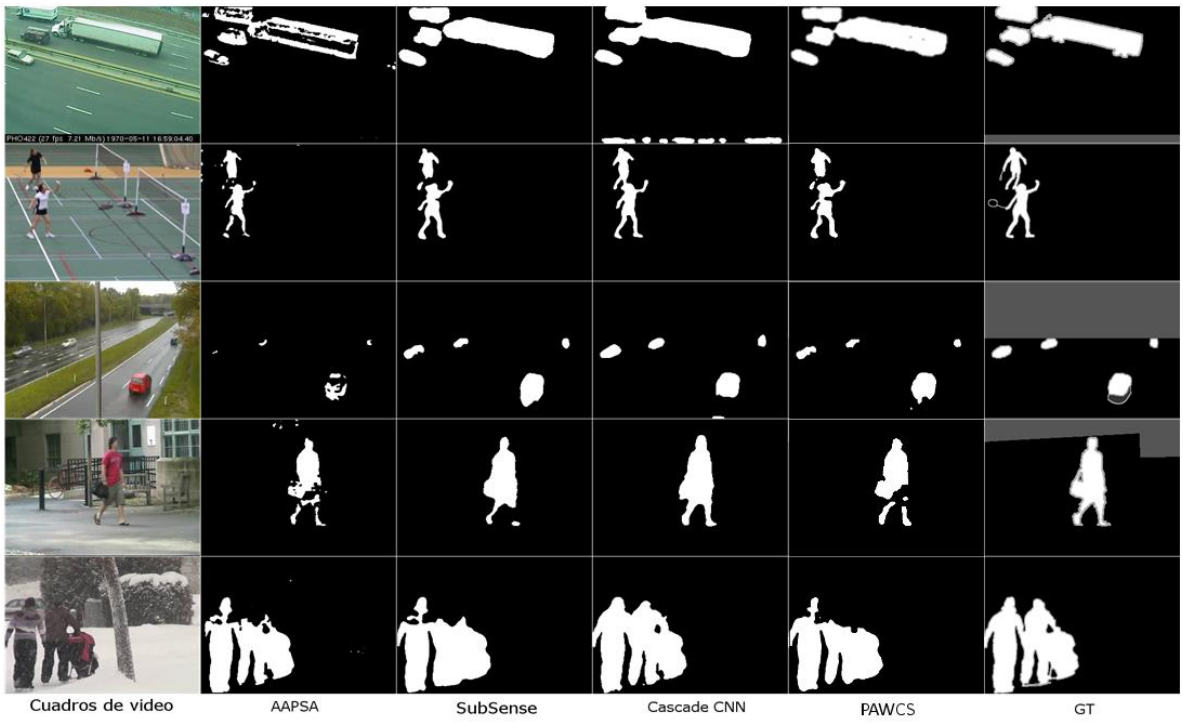


Figura 5.9 Resultados de los algoritmos en los primeros tres lugares en las métricas SSIM, F-Measure y D-Score. Cuadros de video 809, 1125, 1188, 1719 y 1371 de los videos turnpike_0_5fps, badminton, TwoPosition-PTZCam, backdoor y skating respectivamente, ordenados en filas de arriba hacia abajo, en ese orden.

Se puede observar en la imagen anterior que AAPSA realiza una segmentación que resalta la figura o contorno del objeto, en comparación de SuBSENSE o PAWCS en donde se pierden algunos detalles como los pies de las personas en el cuadro 1371 del video *skating* y existen distorsiones donde se observa demasiada dilatación en los objetos. Lo cual a nivel estructural puede implicar él porque AAPSA se encuentra por encima de SuBSENSE en cuanto a *D-Score* y *SSIM* se refiere. Por su parte el algoritmo Cascade CNN tiene muy buenos desempeños en todas las métricas y visiblemente se aprecia que sus resultados de detección son muy semejantes a los cuadros de *Ground truth.*

A continuación, se da un breve resumen de cada uno de los resultados obtenidos en las diferentes métricas en evaluación para la tabla 5.6.

Respecto a los resultados de las métricas mostradas en la tabla 5.6 el desempeño más alto en *Recall* lo obtuvo el algoritmo Cascade CNN con 0.984 logrando sobre pasar incluso el resultado obtenido por el sujeto cinco y el desempeño menor lo obtuvo el algoritmo DeepBs con 0.751, lo que implica que este último clasifica una gran cantidad de pixeles como objeto en movimiento de los cuales solo unos pocos tienen relevancia para la segmentación del objeto de interés. En la métrica *Specificity* los algoritmos y humanos obtuvieron más de 0.99, lo cual es de esperarse debido a que esta métrica se enfoca en evaluar la fracción de verdaderos negativos y por lo general se tiene mayor cantidad de pixeles de fondo que de objetos en movimiento clasificados correctamente. Para la métrica *Precision* el desempeño más bajo fue del algoritmo AAPSA con 0.850 y el desempeño más alto lo obtuvo el sujeto cinco con 0.963, el resultado bajo de AAPSA en esta métrica es consecuencia de que en los resultados de sus detecciones cuenta con un gran número de falsos negativos, lo cual se pudo observar en la figura 5.9 donde los resultados del algoritmo no alcanzan a rellenar por completo los objetos de interés. Las métricas *FPR* y *FNR* son complementarias a las métricas *Specificity* y *Recall* respectivamente, por lo tanto, estas dos métricas no generan mucho impacto en la evaluación, ya que, por ejemplo, si un algoritmo obtuvo el mejor desempeño en *Recall* por consiguiente también lo tendrá en *FNR*. Por lo tanto, no es necesario reportarlas en la evaluación, debido a que se torna algo redundante.

Para las métricas *F-Measure* y *PWC*, las cuales fueron analizadas en la sección 5.1 se demostró que cualquier variación mínima entre el *Ground truth* y la segmentación de los sujetos producía un impacto considerable en los resultados de estas métricas, por lo que, en promedio, el humano solo alcanzó obtener 0.948 y 0.441 respectivamente para *F-Measure* y *PWC*. Por lo tanto, es natural que exista tanta variación entre los mejores y peores desempeños para estas métricas en la tabla 5.6. Por su parte las métricas SSIM y *D-Score* muestran una forma distinta de realizar las evaluaciones donde ya no es tan importante el número de pixeles (de fondo u objeto dinámico) clasificados correctamente, sino que, la segmentación de la imagen produzca de una manera visual o estructural, una semejanza con respecto a un *Ground truth.* Por lo tanto, se puede tener una imagen en la que el objeto dinámico no se encuentre completamente bien segmentado, pero, si se encuentran segmentadas partes que son clave para dar una buena estructura al objeto de interés con respecto a la imagen de referencia, se puede decir que la imagen tendrá una buena calidad

visual. El promedio humano para estas dos últimas métricas fue de 0.959 y 0.005 respectivamente en *SSIM* y *D-Score*.

Finalmente, la métrica FSD es una combinación de las métricas *D-Score*, *F-Measure* y *SSIM*, que intenta unir la información estructural de las métricas *SSIM* y *D-Score* con la información estadística de la métrica *F-Measure*. Sin embargo, esta métrica presenta el problema que, al ser un promedio de las tres anteriores, la métrica F-Measure tendrá el mayor peso, debido a que la variación natural que existe en sus resultados es muy grande en comparación a las otras dos.

Para ejemplificar lo anterior, se muestra la tabla 5.7 donde se observa el desempeño y ranking obtenido por el promedio humano y los algoritmos para cada métrica, exceptuando las métricas FNR y FPR debido a lo antes mencionado de ambas métricas. En la tabla se observa que la variación natural en los resultados de la métrica *D-Score* entre el mejor y peor desempeño es de 0.005 y para la métrica SSIM es de 0.010, por lo tanto, al no existir cambios tan abruptos en los resultados de ambas métricas, no tendrán una gran influencia en el promedio realizado por la métrica FSD. No obstante, en comparación de *SSIM* y *D-Score*, los resultados de la métrica *F-Mesure* generan una variación más marcada (de 0.157). La influencia de la métrica *F-Measure* sobre la métrica FSD se puede observar en las columnas del ranking (marcadas en amarillo), las cuales muestran los mismos resultados para ambas.

Tabla 5.7 Desempeño y ranking obtenido por el promedio humano y los algoritmos para cada métrica exceptuando las métricas FNR y FPR.

Algoritmo	Recall	Specificity	PBC	Precision	F-Measure	SSIM	D-Score	FSD	Ranking Recall	Ranking Specificity	Ranking PBC	Ranking Precision	Ranking F-Measure	Ranking SSIM	Ranking D-Score	Ranking FSD	Average Ranking
CascadeCNN	0.984	0.997	0.409	0.946	0.964	0.959	0.006	0.972	1	2	1	2	1	1	2	1	1.38
Promedio humano	0.966	0.997	0.441	0.936	0.948	0.959	0.005	0.967	2	3	2	3	2	2	1	2	2.13
PAWCS	0.852	0.997	1.068	0.918	0.882	0.955	0.007	0.943	7	4	3	4	5	5	4	5	4.63
SuBSENSE	0.889	0.995	1.265	0.901	0.891	0.954	0.008	0.946	4	8	5	6	3	6	8	3	5.38
MBSV 0	0.878	0.994	1.135	0.874	0.874	0.955	0.007	0.941	5	9	4	9	6	4	6	6	6.13
SharedModel	0.903	0.990	1.315	0.886	0.890	0.952	0.010	0.944	3	10	6	8	4	8	10	4	6.63
FTGS	0.827	0.995	1.402	0.909	0.854	0.952	0.007	0.933	8	5	7	5	8	7	7	8	6.88
DeepBs	0.751	0.998	2.654	0.955	0.807	0.949	0.007	0.916	10	1	10	1	10	10	5	10	7.13
AAPSA	0.783	0.995	1.677	0.850	0.808	0.957	0.006	0.919	9	6	9	10	9	3	3	9	7.25
WesSamBe	0.860	0.995	1.527	0.899	0.873	0.950	0.008	0.938	6	7	8	7	7	9	9	7	7.50
Variación	0.233	0.008	2.245	0.106	0.157	0.010	0.005	0.056									

Por lo tanto, la métrica FSD tiene que ser más estudiada para lograr que cada una de las métricas que involucra, obtengan el mismo peso, ya que la diferencia entre los rangos de variación de cada una de las métricas no está normalizada, siendo más notorio el caso para la métrica *D-Score*.

Dejando de lado lo anterior y pasando a otro punto, ya a que se tienen los resultados de las segmentaciones realizadas por los sujetos (Véase tabla 5.6), se proponen un rango de valores que pueden ser aceptables de acuerdo al desempeño humano para cada una de las métricas, esto tomando en consideración el menor desempeño de los sujetos y el desempeño ideal en cada métrica, lo anterior se observa en la tabla 5.8.

Tabla 5.8 Rango de valores aceptables de acuerdo al desempeño humano promedio.

	Recall	Specificity	PWC	Precision	F-Measure	SSIM	D-Score	FSD
Menor desempeño de las segmentaciones humanas	0.952	0.996	0.514	0.917	0.942	0.956	0.006	0.964
Desempeño ideal	1	1	0	1	1	1	0	1
Rango	0.952-1	0.996-1	0-0.514	0.917-1	0.942-1	0.956-1	0-0.006	0.964-1

Una vez establecidos los rangos, en la tabla 5.9, se observar en color verde, los resultados de las métricas por algoritmo que caen dentro del rango propuesto de lo que se considera desempeño humano.

Tabla 5.9 Resultados de las métricas por algoritmo cuyo desempeño está dentro del rango humano.

Algoritmo	Recall	Specificity	PWC	Precision	F-Measure	SSIM	D-Score	FSD
CascadeCNN	0.984	0.997	0.409	0.946	0.964	0.959	0.006	0.972
PAWCS	0.852	0.997	1.068	0.918	0.882	0.955	0.007	0.943
SuBSENSE	0.889	0.995	1.265	0.901	0.891	0.954	0.008	0.946
DeepBs	0.751	0.998	2.654	0.955	0.807	0.949	0.007	0.916
MBSV 0	0.878	0.994	1.135	0.874	0.874	0.955	0.007	0.941
SharedModel	0.903	0.990	1.315	0.886	0.890	0.952	0.010	0.944
FTGS	0.827	0.995	1.402	0.909	0.854	0.952	0.007	0.933
AAPSA	0.783	0.995	1.677	0.850	0.808	0.957	0.006	0.919
WesSamBe	0.860	0.995	1.527	0.899	0.873	0.950	0.008	0.938
SBM	NA	NA	NA	NA	NA	NA	NA	NA

De la tabla anterior, es evidente que el algoritmo Cascade CNN logra estar dentro del rango del desempeño humano en todas las métricas, mientras que PAWCS y DeepBs lo hacen para las métricas *Specificity* y *Precision*, y finalmente AAPSA cae dentro del rango del desempeño humano en las métricas *SSIM* y *D-Score*.

Debido a que ya se estableció un rango de valores de lo que se considera desempeño humano promedio, también se puede visualizar el porcentaje de qué tan cercanos se encuentran los algoritmos de alcanzar dicho rango de valores para cada una de las métricas. Esto, se observa en la tabla 5.10, la cual muestra los porcentajes obtenidos de los algoritmos en cada una de las métricas, respecto al rango de valores del desempeño humano promedio, donde los valores mínimos para caer dentro del rango humano fueron mostrados en la tabla 5.8.

Tabla 5.10 Porcentajes de desempeño de los algoritmos en cada una de las métricas respecto al rango que se considera desempeño humano promedio.

Algoritmo	Recall	Specificity	PWC	Precision	F-Measure	SSIM	D-Score	FSD	Porcentaje promedio del desempeño de los algoritmos respecto al rango humano
CascadeCNN	103	100	126	103	102	100	100	101	104
PAWCS	89	100	48	100	94	100	86	98	89
MBSV 0	92	100	45	95	93	100	86	98	89
SuBSENSE	93	100	41	98	95	100	75	98	87
FTGS	87	100	37	99	91	100	86	97	87
AAPSA	82	100	31	93	86	100	100	95	86
WesSamBe	90	100	34	98	93	99	75	97	86
SharedModel	95	99	39	97	94	100	60	98	85
DeepBs	79	100	19	104	86	99	86	95	84
SBM	NA	NA	NA	NA	NA	NA	NA	NA	NA

De la tabla anterior, en la última columna se observa que el algoritmo Cascade CNN esta 4% arriba del valor mínimo del rango de desempeño humano. El resto de los algoritmos no logran alcanzar el 90%. Sin embargo, logran situarse en un rango entre el 84 y 89% esto es un indicador de que estos algoritmos no están tan alejados respecto a lo que se puede considerar comportamiento humano. También, se puede observar que la métrica que reporta menor porcentaje es PWC con excepción del algoritmo Cascade CNN. Por lo que, aún queda trabajo para lograr que los algoritmos tengan un buen desempeño en todas las métricas, sin embargo, parece que los nuevos algoritmos van en buen camino, tal es el caso de Cascade CNN.

Por otro parte, con el propósito de analizar si existían ocasiones en las que los algoritmos lograban resultados mejores que el promedio humano para algunas categorías de video, se evaluó su desempeño con respecto al ranking obtenido en cada categoría de video. En la tabla

5.11 se muestran los rankings de los algoritmos y del promedio humado con respecto a cada una de las categorías de video.

Tabla 5.11 Ranking de los algoritmos y el promedio humano obtenido sobre cada categoría de video.

Ranking sobre cada categoría de video												
Segmentación humana y algoritmos	Categorías											Average Ranking
	PTZ	Bad Weather	Baseline	Camera Jitter	Dynamic Background	Intermitent Object Motion	Low Framerate	Night Videos	Shadow	Thermal	Turbulence	
	Videos											
	twoPositionPTZ Cam	skating	office	badminton	fountain02	streetLight	turnpike_0_5fps	tramStation	backdoor	library	turbulence3	
	Ranking											
Cascade CNN	2	3	3	1	1	5	1	1	3	2	1	2.09
Promedio humano	1	2	2	3	9	9	3	3	1	1	2	3.27
DeepBs	4	1	1	2	5	7	9	4	2	10	3	4.36
SuBSENSE	6	9	5	7	2	1	5	5	4	8	5	5.18
PAWCS	5	6	8	8	3	6	2	2	10	6	10	6.00
Wesambe	3	10	9	9	6	2	4	7	6	9	4	6.27
SharedModel	8	7	7	6	4	3	6	8	9	7	7	6.55
MBSV 0	9	5	4	10	7	8	7	6	7	4	6	6.64
AAPSA	10	4	6	4	10	4	10	10	8	3	9	7.09
FTSG	7	8	10	5	8	10	8	9	5	5	8	7.55

De la tabla anterior se puede observar que el algoritmo Cascade CNN se mantiene superior al promedio de desempeño humano. DeepBS logra el tercer lugar, teniendo el mejor desempeño en las categorías *Bad Weather* y *Baseline*, siendo su peor resultado en las categorías *Low Framerate* y *Thermal*. Por otro lado, el promedio de segmentación humana obtuvo el primer lugar únicamente en tres ocasiones, para los videos *twoPositionPTZCam*, *library* y *backdoor*, las cuales son escenas que no representan mucho problema, debido a que no existen oclusiones, ni fondos dinámicos. Por otro lado, en dos ocasiones el promedio humano se posicionó en noveno lugar, para los videos *fountain02* y *streetlight* de las categorías *Dynamic Background* e *Intermitent Object Motion* respectivamente.

Del video *fountain02* se observa una variación muy marcada entre las segmentaciones realizadas por cada uno de los sujetos, específicamente en los cuadros donde existe oclusión del objeto en movimiento, en este caso un automóvil que pasa por atrás de un árbol, el cual a su vez tiene movimiento en sus hojas y ramas. Por lo que, cada sujeto a su juicio, segmentó

lo que consideraron debía ser parte del automóvil, dejando fuera de la segmentación el tronco y ramas del árbol. Esto se puede observar en la figura 5.10. Por su parte, los algoritmos en esta categoría de video tienen un comportamiento muy semejante entre ellos, desempeñándose bien ante los fondos dinámicos, la única excepción fue el algoritmo AAPSA el cual no logra eliminar por completo la fuente de agua que se observa en primer plano de la escena. Los resultados de detección de los algoritmos pueden verse en la figura 5.11.

Para el video *Streetlight* los sujetos segmentaron todos los objetos en movimiento, sin contar los que fueran considerados como fondos dinámicos, ya que no se les comentó que existían zonas de no interés en las escenas. Sin embargo, las zonas grises del *Ground truth* de *Change detection* no fueron consideradas para la evaluación. En la figura 5.12 se observa la segmentación realizada por los cinco sujetos, la cual, debido a que se presentaban zonas donde se tenía desenfoque debido al movimiento, hubo una variedad de resultados distintos. En el cuadro 1735 en particular se observa que la segmentación realizada a dos personas que pasaban por una carretera tuvo varias interpretaciones, debido a que existían pixeles que no se distinguía si eran de fondo o del objeto en movimiento. De la figura 5.13 se observa que el algoritmo Cascade CNN, muestra visiblemente la peor detección en comparación a los demás algoritmos, ya que en este video se aprecian una gran cantidad de falsos positivos ubicados a los costados de la carretera. Sin embargo, al no ser considerada la zona gris del *Ground truth* su desempeño en la evaluación no se vio afectado.

En conclusión, para este análisis, se puede decir que los humanos, al igual o en mayor cantidad que los algoritmos tienden a fallar en categorías de video donde existen situaciones críticas de video tales como fondos dinámicos, camuflaje, problemas por desenfoque, entre otras. Por lo tanto, el proceso de realizar la evaluación respecto a un *Ground truth* realizado por un humano resulta en una evaluación subjetiva. Tanto más subjetiva, en cuanto menos entrenamiento tenga el humano. Sin embargo, nos puede dar un indicio de que tan cercanos se encuentran los resultados de detección de los algoritmos respecto a lo que se considera un desempeño humano promedio.

A continuación, se muestran las figuras de la 5.10 a la 5.13 antes descritas.

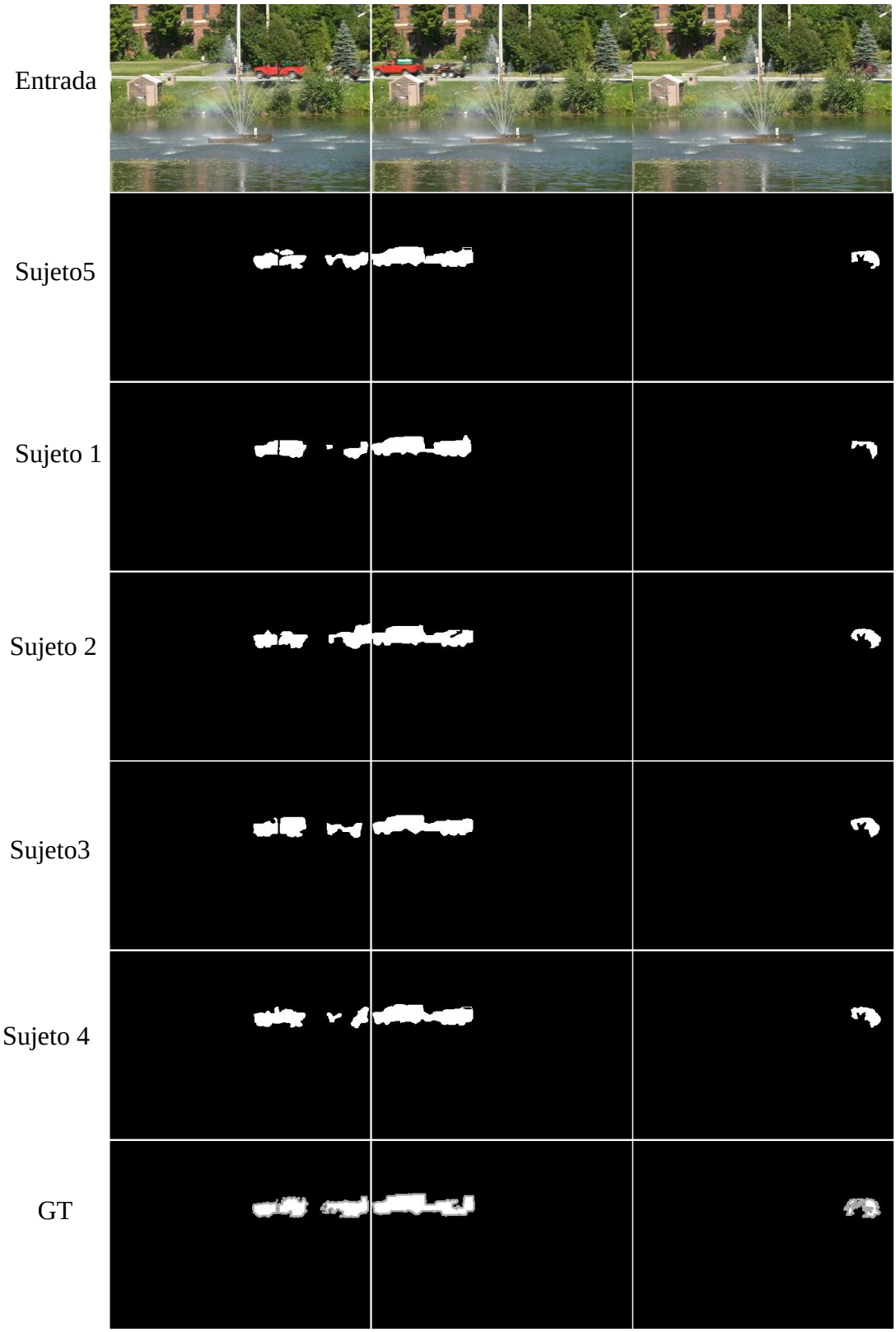


Figura 5.10 Segmentaciones humanas en video *fountain02*, cuadros de video 694, 745 y 1236 ordenados de izquierda a derecha.

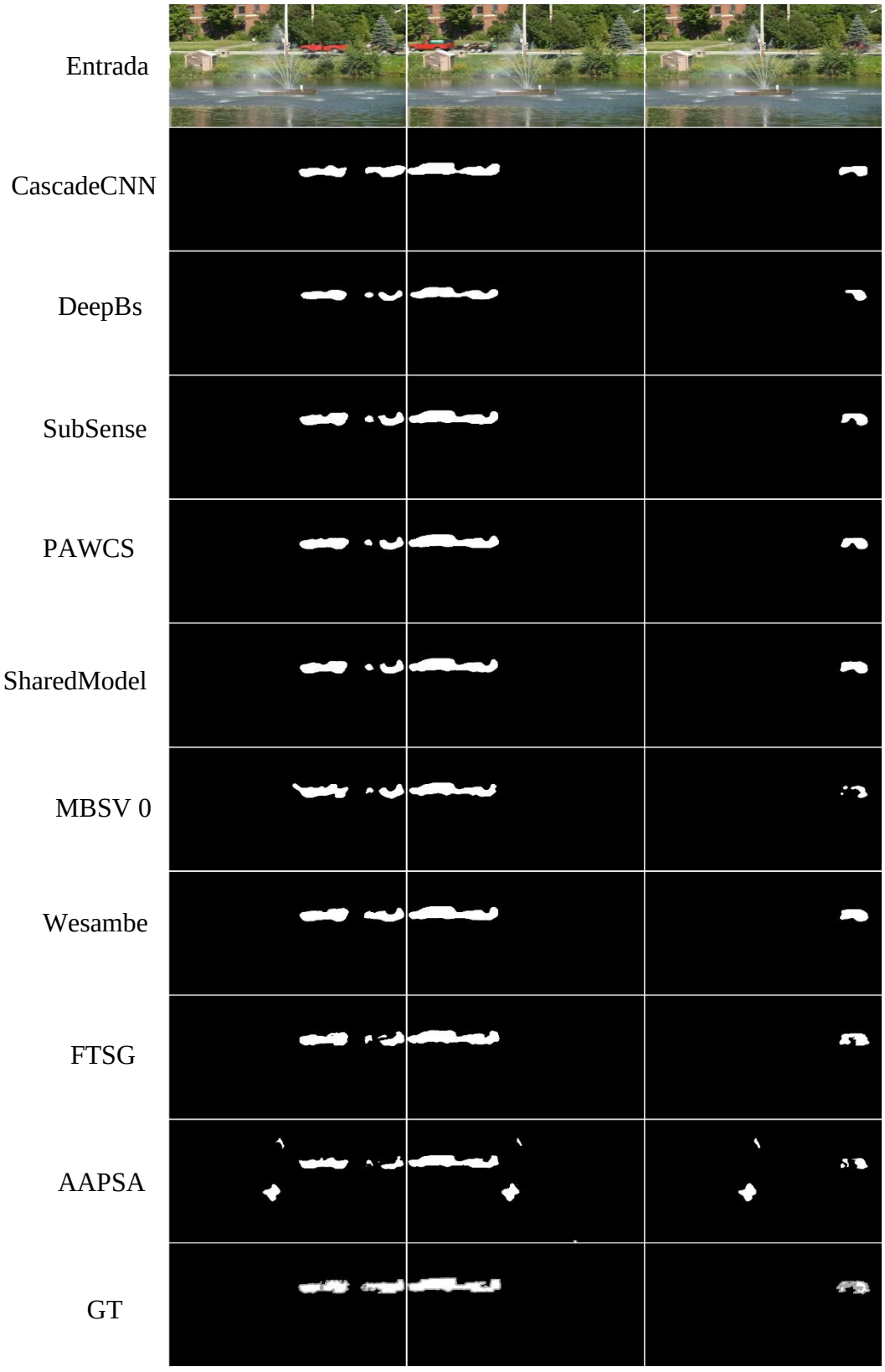


Figura 5.11 Resultados de los algoritmos en el video *fountain02*, cuadros de video 694, 745 y 1236 ordenados de izquierda a derecha.

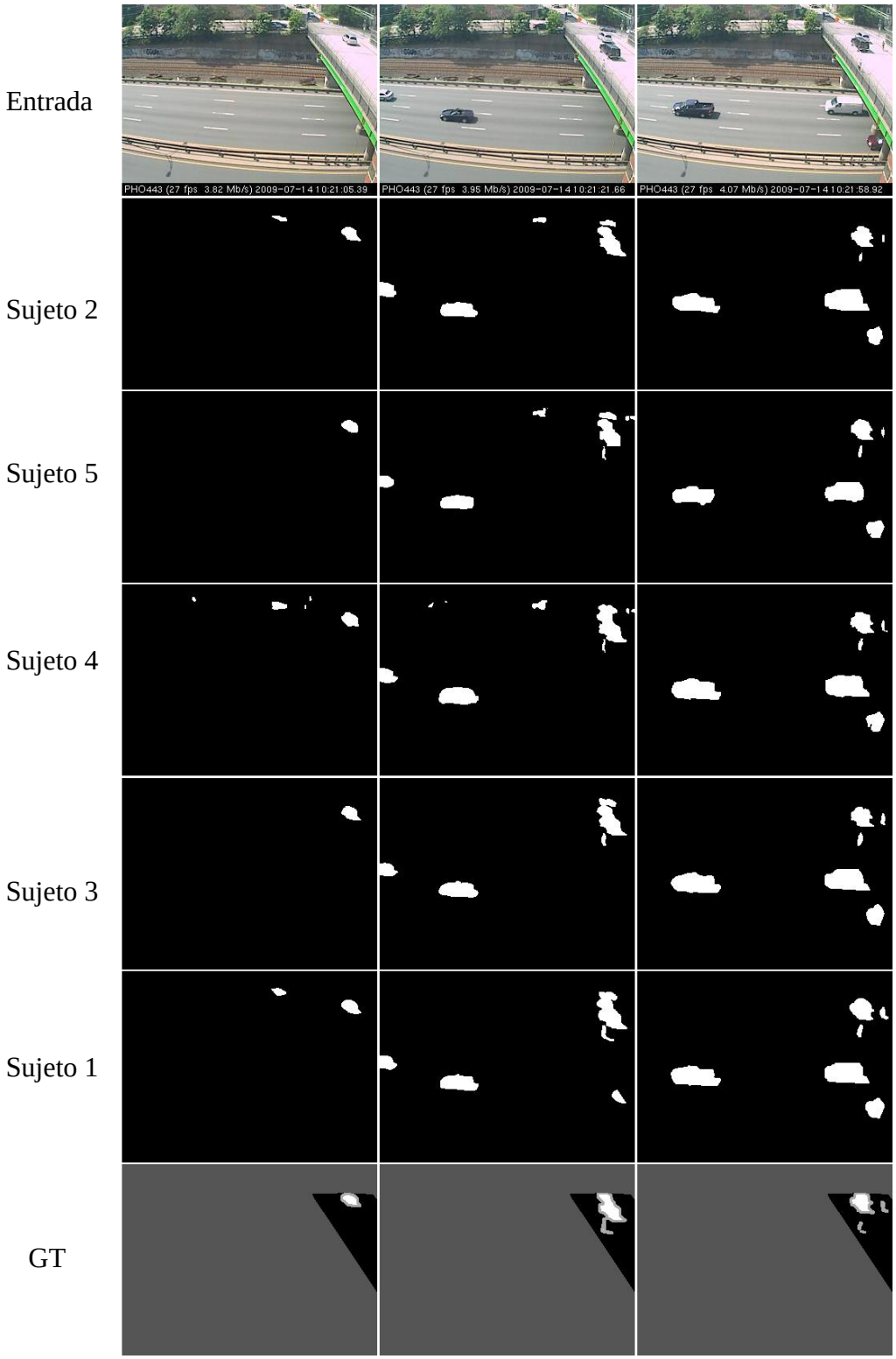


Figura 5.12 Segmentaciones humanas en el video *Streetlight*, en los cuadros de video 288, 728 y 1735 ordenados de izquierda a derecha. Segmentaciones ordenadas de mayor a menor desempeño. La zona gris que se puede apreciar el Ground Truth (GT), no es zona de interés, por lo que no fue considerada en ninguna de las segmentaciones para realizar las evaluaciones.

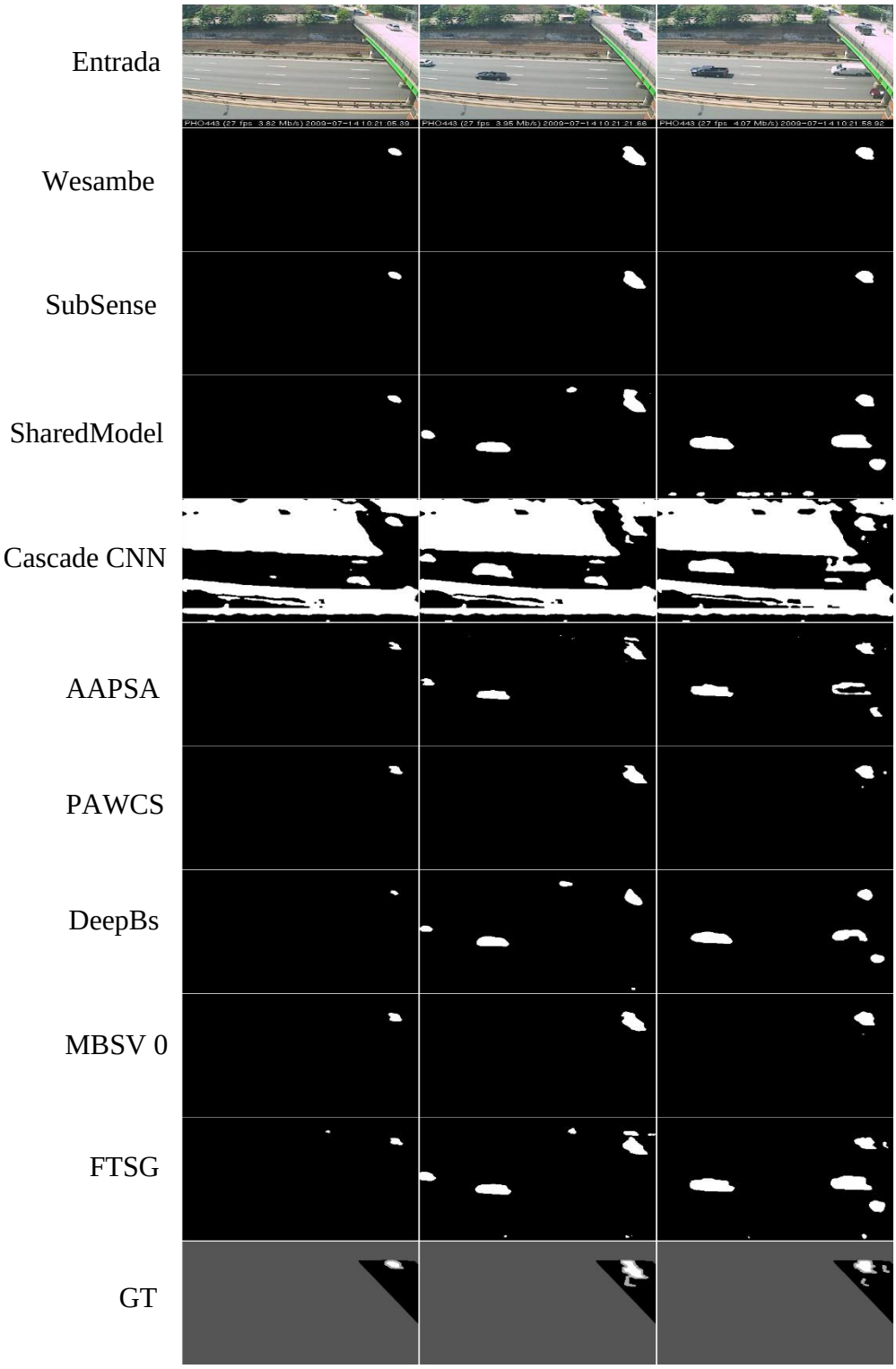


Figura 5.13 Resultados de los algoritmos en el video *Streetlight*, en los cuadros de video 288, 728 y 1735 ordenados de izquierda a derecha. Segmentaciones ordenadas de mayor a menor desempeño. La zona gris que se puede apreciar el Ground Truth (GT), no es zona de interés, por lo que no fue considerada en ninguno de los resultados de los algoritmos para realizar las evaluaciones.

5.3 Análisis por tipo de técnica

Para concluir este capítulo, se realizó un análisis de los algoritmos respecto al tipo de técnica que utilizan para realizar la detección de movimiento, esto con el fin de obtener un panorama de qué técnicas resultan ser las mejores para afrontar las diferentes circunstancias o categorías de video. Por lo que, primero se procedió a identificar el tipo de técnica de cada uno de los algoritmos. Una vez identificado el tipo de técnica se procedió a evaluar en qué posición de ranking sobre cada categoría de video se posicionaban y finalmente se realiza el análisis y comparativa de los resultados.

A continuación, en la tabla 5.12 se observan los algoritmos con su tipo de técnica para realizar la detección de movimiento.

Tabla 5.12 Algoritmos y tipo de técnica de detección de movimiento.

Algoritmo	Tipo de técnica
Cascade CNN	Redes Convolucionales
DeepBs	Redes Convolucionales
PAWCS	Diccionario de Aprendizaje
SharedModel	Modelo Estadístico Avanzado basado en GMM
AAPSA	Redes neuronales Auto-organizadas
SuBSENSE	Modelo Estadístico Avanzado LBSP (Local Binary Similarity Patterns)
MBSV 0	Modelo Estadístico Avanzado con mega pixeles
Wesambe	Modelo Estadístico Avanzado con estrategia de refuerzo y penalización de pesos
FTSG	Modelo Tensor Robusto Flux tensor con Gaussianas divididas

Enseguida, en la tabla 5.13 se muestra el ranking obtenido de cada técnica respecto a las once categorías de video. Se puede observar que las redes neuronales convolucionales usadas por los algoritmos DeepBs y Cascade CNN, se posicionan en primer lugar en casi todas las categorías de video, sólo en la categoría de objetos intermitentes este tipo de técnica se sitúa en la quita posición. Las redes convolucionales han tomado gran auge en los últimos años por su capacidad de dar buenos resultados con grandes cantidades de datos. Cuentan con la ventaja de no requerir el definir características, ya que, por sí solas extraen las características. Sin embargo, sus principales desventajas son que requieren una gran cantidad de datos de entrenamiento y en la mayoría de los casos su tiempo de procesamiento es muy elevado, como el algoritmo CNN el cual procesa a 12 FPS y el algoritmo DeepBS a 10 FPS en videos con resolución de 320x240, lo cual, es impráctico para realizar aplicaciones en tiempo real. No obstante, para aplicaciones fuera de línea las redes convolucionales plantean una buena opción con un alto nivel de desempeño.

Tabla 5.13 Ranking del tipo de técnica sobre las categorías de video.

Ranking	Categorías de video									
	PTZ	***Bad Weather***	***Baseline***	***Camera Jitter***	***Dynamic Background***	***Intermittent Object Motion***	***Low Frame Rate***	***Night Videos***	***Shadow***	***Thermal***	***Turbulence***
1	Redes convolucionales (Cascade CNN)	Redes convolucionales (DeepBs)	Redes convolucionales (DeepBs)	Redes convolucionales (Cascade CNN)	Redes convolucionales (Cascade CNN)	Modelos Estadísticos Avanzados (Wesambe)	Redes convolucionales (Cascade CNN)	Redes convolucionales (Cascade CNN)	Redes convolucionales (DeepBs)	Redes convolucionales (Cascade CNN)	Redes convolucionales (Cascade CNN)
2	Modelos Estadísticos Avanzados (Wesambe)	Redes convolucionales (Cascade CNN)	Redes convolucionales (Cascade CNN)	Redes convolucionales (DeepBs)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Estadísticos Avanzados (SuBSENSE)	Diccionario de aprendizaje (PAWCS)	Diccionario de aprendizaje (PAWCS)	Redes convolucionales (Cascade CNN)	Redes neuronales (AAPSA)	Redes convolucionales (DeepBs)
3	Redes convolucionales (DeepBs)	Redes neuronales (AAPSA)	Modelos Estadísticos Avanzados (MBSV 0)	Redes neuronales (AAPSA)	Diccionario de aprendizaje (PAWCS)	Modelos Estadísticos Avanzados (SharedModel)	Modelos Estadísticos Avanzados (Wesambe)	Redes convolucionales (DeepBs)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Estadísticos Avanzados (MBSV 0)	Modelos Estadísticos Avanzados (Wesambe)
4	Diccionario de aprendizaje (PAWCS)	Modelos Estadísticos Avanzados (MBSV 0)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Tensores Robustos (FTSG)	Modelos Estadísticos Avanzados (SharedModel)	Redes neuronales (AAPSA)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Estadísticos Avanzados (MBSV 0)	Modelos Tensores Robustos (FTSG)	Modelos Tensores Robustos (FTSG)	Modelos Estadísticos Avanzados (SuBSENSE)
5	Modelos Estadísticos Avanzados (SuBSENSE)	Diccionario de aprendizaje (PAWCS)	Redes neuronales (AAPSA)	Modelos Estadísticos Avanzados (SharedModel)	Redes convolucionales (DeepBs)	Redes convolucionales (Cascade CNN)	Modelos Estadísticos Avanzados (SharedModel)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Estadísticos Avanzados (Wesambe)	Diccionario de aprendizaje (PAWCS)	Modelos Estadísticos Avanzados (MBSV 0)
6	Modelos Tensores Robustos (FTSG)	Modelos Estadísticos Avanzados (SharedModel)	Modelos Estadísticos Avanzados (SharedModel)	Diccionario de aprendizaje (PAWCS)	Modelos Estadísticos Avanzados (Wesambe)	Diccionario de aprendizaje (PAWCS)	Modelos Tensores Robustos (FTSG)	Modelos Estadísticos Avanzados (Wesambe)	Modelos Estadísticos Avanzados (MBSV 0)	Modelos Estadísticos Avanzados (SharedModel)	Modelos Estadísticos Avanzados (SharedModel)
7	Modelos Estadísticos Avanzados (SharedModel)	Modelos Tensores Robustos (FTSG)	Diccionario de aprendizaje (PAWCS)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Estadísticos Avanzados (MBSV 0)	Redes convolucionales (DeepBs)	Modelos Estadísticos Avanzados (MBSV 0)	Modelos Estadísticos Avanzados (SharedModel)	Redes neuronales (AAPSA)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Tensores Robustos (FTSG)
8	Modelos Estadísticos Avanzados (MBSV 0)	Modelos Estadísticos Avanzados (SuBSENSE)	Modelos Estadísticos Avanzados (Wesambe)	Modelos Estadísticos Avanzados (Wesambe)	Modelos Tensores Robustos (FTSG)	Modelos Estadísticos Avanzados (MBSV 0)	Redes convolucionales (DeepBs)	Modelos Tensores Robustos (FTSG)	Modelos Estadísticos Avanzados (SharedModel)	Modelos Estadísticos Avanzados (Wesambe)	Redes neuronales (AAPSA)
9	Redes neuronales (AAPSA)	Modelos Estadísticos Avanzados (Wesambe)	Modelos Tensores Robustos (FTSG)	Modelos Estadísticos Avanzados (MBSV 0)	Redes neuronales (AAPSA)	Modelos Tensores Robustos (FTSG)	Redes neuronales (AAPSA)	Redes neuronales (AAPSA)	Diccionario de aprendizaje (PAWCS)	Redes convolucionales (DeepBs)	Diccionario de aprendizaje (PAWCS)

Otras de las técnicas que se situaron dentro de los primeros y segundos lugares en la tabla 5.13 son los diccionarios de aprendizaje usados por el algoritmo PAWCS, las redes neuronales de AAPSA y los modelos estadísticos avanzados usados por los algoritmos SuBSENSE y Wesambe. Los modelos estadísticos avanzados obtuvieron el primer y segundo lugar en la categoría de video *Intermittent Object Motion,* y segundo lugar también, en las categorías *PTZ y Dynamic Background.* La categoría *Intermittent Object Motion* contiene el video *Streetlight* el cual se observó previamente en la figura 5.11 donde el método Wesambe obtuvo los mejores resultados. Por su parte, la técnica de diccionarios de aprendizaje usada por el algoritmo PAWCS logra obtener el segundo lugar en las categorías de video *Low Frame Rate* y *Night Videos* y la técnica de redes neuronales de AAPSA obtuvo el segundo lugar en la categoría *Thermal.* Los terceros lugares se encuentran distribuidos dentro varias de las técnicas; redes convolucionales para *PTZ* y *Night Videos*; modelos estadísticos avanzados para *Baseline*, *Thermal*, *Intermittent Object Motion*, *Turbulence*, *Shadow* y *Low Frame Rate*; redes neuronales para *Bad Weather* y *Camera Jitter* y finalmente diccionarios de aprendizaje para *DynamicBackground*. A manera general los tres primeros lugares sobre todas las categorías de video se reparten como se muestra en la figura 5.14. Donde, redes convolucionales aparece el 52% de las veces dentro de los tres primeros lugares, los modelos estadísticos avanzados 30%, los diccionarios de aprendizaje 9% y las técnicas redes neuronales 9%. Cabe destacar que dentro de los tres primeros lugares no aparecen los modelos tensores robustos.

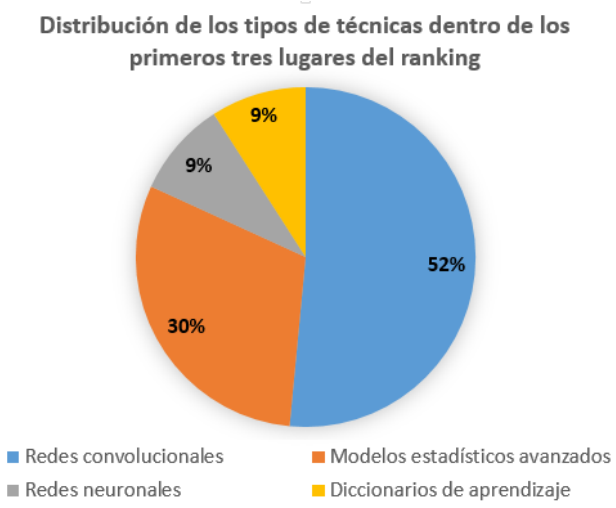


Figura 5.14 Porcentaje de veces que aparecen ciertos tipos de técnicas dentro de los tres primeros lugares del ranking del tipo de técnica sobre categoría de video.

Es importante notar que el tipo de técnica de redes convolucionales usada por el algoritmo Cascade CNN plantea un futuro prometedor en el área de detección de movimiento, ya que esta técnica como se analizó en la sección previa, logró tener un desempeño dentro del rango del desempeño humano promedio y además logra afrontar correctamente todas las categorías de video excepto la categoría de videos de objetos con movimiento intermite. Aunque, también técnicas de modelos estadísticos con LBSP y diccionarios de aprendizaje con LBSP usadas por los algoritmos SuBSENSE y PAWCS respectivamente, logran situarse dentro de las primeras tres técnicas que mejor pueden afrontar situaciones de fondos dinámicos, objetos con movimiento intermitente, baja tasa de velocidad de cuadros, tomas nocturnas y sombras. Además, PAWCS y SuBSENSE como se mostró en la sección 4.1.1.4 pueden trabajar en tiempo real. Por último, una de las técnicas que resalta de los modelos estadísticos avanzados es el algoritmo Wesambe, este logra colocarse dentro de las tres primeras técnicas de detección de movimiento y afronta situaciones de cámaras PTZ, objetos con movimiento intermitente, baja tasa de velocidad de cuadros y turbulencia. A manera general, las técnicas mencionadas anteriormente, podrían ser aún más explotadas para desarrollar nuevos métodos con mejores características. De ellas, la más destacada es redes convolucionales, la cual, aunque tiene los mejores desempeños, no cuenta con un procesamiento en tiempo real, por lo que, se requiere más investigación en esta área para tratar de mejorar sus características con el fin de ser aplicadas en tiempo real.

En resumen, en este capítulo se analizó la variación que existe en los humanos al realizar un *Ground truth*, tomando en consideración cinco sujetos, donde en promedio, en métricas como *F-Measure* y *SSIM* se obtuvo 0.948 y 0.959 respectivamente. Después se realizó una comparación de las métricas F-Measure y PWC contra las métricas enfocadas en medir la calidad de la imagen SSIM y *D-Score*. Los resultados de estas comparaciones mostraron que las métricas *F-Measure* y *PWC* realizan una evaluación muy estricta, ya que evalúan el número de pixeles (de fondo u objetos dinámicos) correctamente clasificados y por su parte las métricas *SSIM* y *D-Score* son más flexibles ya que enfocan en medir la calidad de segmentación de la imagen, tomando en consideración información estructural, y finalmente lo que se busca de un algoritmo de detección de movimiento no es que cantidad de pixeles puede detectar correctamente, si no que los pixeles que detecte sean significativos y visualmente se logre identificar el objeto de interés. Por lo que estas dos métricas deberían

de ser más reportadas en las evaluaciones de algoritmos. Por otro lado, se estableció un rango de valores sobre cada una de las métricas que se podría considerar dentro del rango humano, para después evaluar cuales algoritmos lograban situarse dentro del rango humano propuesto para cada métrica, y se obtuvo que el algoritmo Cascade CNN logro situarse en todas las métricas dentro del rango humano, el algoritmo AAPSA en las métricas de *SSIM* y *D-Score* y en *Specificity* y *Precision* los algoritmos DeepBs y PAWCS. Luego se analizó que tan cercanos estaban los algoritmos de lograr el desempeño humano y se obtuvo que, en porcentaje promedio general sobre todas las métricas, todos los algoritmos lograban tener un porcentaje dentro del 84 a 89 % de proximidad a estar dentro del rango de lo que se considera desempeño humano, excepto el algoritmo Cascade que incluso rebasó el valor mínimo para del rango del desempeño humano. Finalmente, se realizó un análisis por tipo de técnica para conocer que técnicas son las que presentan más facilidad para afrontarse a diferentes circunstancias críticas de video. Donde se obtuvo que las técnicas con mejores desempeños son redes convolucionales, los modelos estadísticos avanzados y los diccionarios de aprendizaje.

CAPÍTULO VI. RESULTADOS Y CONCLUSIONES

Este capítulo se divide en tres secciones. En la primera se da una conclusión de la búsqueda realizada en la literatura de algoritmos enfocados en detección de movimiento, en el periodo 2013 a 2017. La segunda detalla los resultados obtenidos en la metodología propuesta para realizar la comparación entre algoritmos. Y la última, describe los resultados obtenidos en la evaluación de los algoritmos contra el criterio de segmentación humana.

6.1 Estado del arte enfocado a detección de movimiento en el periodo 2013-2017

Del análisis realizado a la literatura en el periodo del año 2013 a 2017 mostrado en el Capítulo III, se estudiaron 47 algoritmos, los cuales fueron resumidos y clasificados por tipo de técnica. En total, se formaron once categorías de acuerdo al tipo de técnica, las cuales fueron: modelos básicos, combinados, estadísticos, redes neuronales, modelos estadísticos avanzados, modelos difusos, modelos de subespacios, diccionarios de aprendizaje, modelos de transformación de dominio, modelos tensores robustos y otros modelos. De los 47 algoritmos, las tres técnicas más utilizadas corresponden a redes neuronales con 24%, modelos estadísticos avanzados con 19% y los modelos de subespacios con 13%, esto indica que estas tres últimas técnicas han tomado popularidad en los últimos cuatro años. Lo anterior se observa en la figura 6.1.

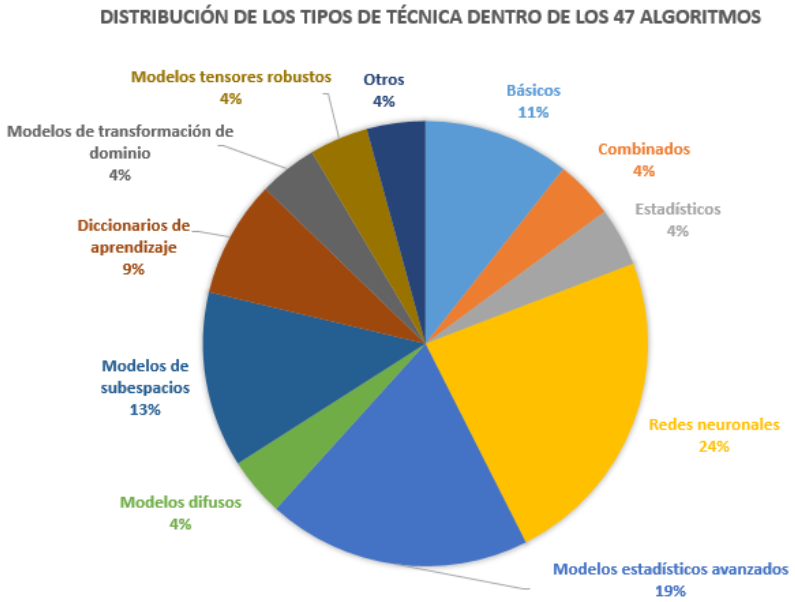


Figura 6.1 Porcentaje de uso de los diferentes tipos de técnicas dentro de los 47 algoritmos encontrados en el periodo 2013-2017.

Los modelos de redes neuronales comparten muchas características semejantes a las del cerebro humano, son capaces de aprender de la experiencia, de generalizar, de extraer características a partir de entradas que representan información irrelevante, entre otras. Por lo que, no es difícil imaginar porque son tan utilizados. Entre algunas de sus desventajas se encuentra que requieren entrenamiento y en ocasiones no pueden procesar en tiempo real a resoluciones mayores de 320x240 pixeles, tal es el caso de [33]–[35], [37] y [38]. Sin embargo, estos problemas se encuentran en constante mejora o estudio, como el método [42] basado en una red neuronal con función radial que logra realizar un procesamiento en tiempo real y el algoritmo [36] que hace uso de una red neuronal SOM en conjunto con un GPU. Por lo que, muy probablemente en un futuro, las desventajas de requerir un gran número de cuadros de entrenamiento y el no poder procesar en tiempo real, serán reducidas al grado de no representar un problema. Por su parte, los modelos estadísticos avanzados, se han vuelto muy populares, debido a que, el realizar estimaciones del fondo mediante métodos más sofisticados que un simple modelo de mezcla de Gaussianas GMM o *Kernel density estimation* ha desencadenado el desarrollo de algoritmos más robustos y con mejores características, tal es el caso del método basado en LBP [43], la mejora al método ViBe [44] y el algoritmo basado en LBSP [51] los cuales, son modelos robustos e ideales para aplicaciones en tiempo real. Por último, los modelos de subespacios involucran en su mayoría, métodos basados en descomposiciones en matrices de bajo rango (correspondiente al fondo) y una matriz de esparcimiento (correspondiente a objetos dinámicos), estos métodos han mostrado ser muy eficientes para el manejo de fondos dinámicos y variaciones de luminosidad [54], [57]. Sin embargo, la mayoría funcionan por lotes de cuadros y cuando el tamaño de los datos crece, estos métodos no pueden afrontar problemas en tiempo real. Por lo que, en aras de mejorar estos métodos se han desarrollado aproximaciones de matrices de descomposición en línea, tal es el caso de COROLA [55] y el método basado en superpixeles [59], sin embargo, estos últimos aunque logran reducir el costo computacional y dan buenos resultados, aun requieren ser más estudiados para que logren un procesamiento en tiempo real.

En cuanto a las demás técnicas no significa que no den buenos resultados, ya que incluso en ocasiones superan a las tres técnicas anteriores, es soló que probablemente no han sido aún tan explotadas. De estas técnicas, es interesante notar, que el 11% de utilización entre los

47 algoritmos analizados pertenece a modelos básicos, lo que significa que estos modelos continúan siendo aplicados aún y cuando existen métodos nuevos con mejores características, probablemente esto se deba a que su funcionamiento es de lo más simple, y solo se requiere el uso de algunos parámetros para realizar la detección, en comparación de las técnicas nuevas que involucran mayor complejidad. Otro de los métodos, que parece ser, será más utilizado en los próximos años es diccionarios de aprendizaje, el cual obtuvo un 9% de uso dentro de los 47 algoritmos analizados, y en cuanto al resto de las técnicas mantuvieron un empate con el 4%.

6.1 Metodología de evaluación y comparación de algoritmos

En este trabajo de tesis se propuso una metodología para evaluar y comparar los distintos enfoques existentes para realizar la detección de movimiento y modelado de fondo. La metodología de evaluación, se llevó a cabo mediante cinco criterios distintos propuestos a evaluación, documentación, auto-adaptabilidad, desempeño, velocidad y actualidad. En total, 112 algoritmos fueron considerados para realizar el análisis enfocado a detección de movimiento y 34 algoritmos para el análisis de modelado de fondo.

En el criterio documentación, se evaluó la calidad del contenido puesto en las publicaciones de los artículos. Este criterio evidencia la problemática existente debido a que la mayoría de los autores, no reportan muchos de los aspectos del diseño y desarrollo experimental del algoritmo. Por lo que, los algoritmos que cumplieron con una buena documentación obtuvieron mejores puntuaciones. Los diez primeros lugares de este criterio fueron mostrados en la tabla 4.3 y la evaluación completa se puede observar en el apéndice A. La importancia de este criterio, radica, en que sin una buena documentación el algoritmo carece de relevancia científica, ya que no existe forma de demostrar aspectos concernientes a la problemática que resuelve y/o el impacto de la aportación, lo cual impide a su vez, el avance científico.

El criterio auto-adaptabilidad fue propuesto a valoración para resaltar aquellos algoritmos que cuentan con la capacidad de ser autónomos y cambiar su comportamiento con respecto al tipo de video o escenario del cual se detecte el movimiento. En este criterio se examinaron aspectos referentes a la adaptabilidad del algoritmo, de acuerdo a lo reportado en sus publicaciones. Se evaluaron aspectos como: la cantidad de parámetros manuales y

automáticos que utilizan, si son capaces de dar un buen resultado sin la necesidad de requerir cuadros de entrenamiento y su complejidad. Los diez primeros lugares de este criterio se observan en la tabla 4.5 y la evaluación completa se encuentra en el apéndice B. La auto-adaptabilidad implica el grado de autonomía que tiene un algoritmo para resolver diferentes situaciones críticas sin la intervención humana, lo cual es enfocado en que se traten de proponer algoritmos que logren emular el pensamiento humano para hacerle frente a los problemas que se suscitan en aplicaciones reales de video.

Para el criterio desempeño, se consideraron las bases de datos más completas y representativas, Change detection, BMC, SBMnet y SBI. Luego, se propuso una evaluación donde se tomó en consideración, tanto el desempeño obtenido por el algoritmo en alguna base de datos, como el número de bases de datos usadas para demostrar su desempeño. Los algoritmos con mejores resultados de este criterio fueron mostrados en las tablas 4.27-A y 4.12, para el análisis enfocado en modelado de fondo y detección de movimiento respectivamente.

En el criterio velocidad se evalúa el tiempo de procesamiento reportado por los algoritmos. La velocidad es una característica que implica si un algoritmo puede ser usado o no en aplicaciones en tiempo real, lo cual es un rasgo que se busca en un algoritmo. Se consideró que los algoritmos que tuvieran un procesamiento mayor o igual a 25 FPS, sí cumplían con esta característica. Los algoritmos que cumplieron con un procesamiento en tiempo real en el análisis enfocado a algoritmos de detección de movimiento, se muestran en la tabla 4.13, representando el 38% de un total de 112 algoritmos. Por su parte, los resultados de este criterio en el análisis enfocado en modelado de fondo se muestran en la tabla 4.28, de los cuales, únicamente el 1.3% de un total de 34 algoritmos cumplieron con un procesamiento en tiempo real.

El último criterio en evaluación, fue actualidad, en él se toma en consideración la fecha de publicación del algoritmo. Este criterio fue pensado para favorecer a las ideas nuevas ya que los algoritmos más recientes son los menos estudiados. Se consideraron algoritmos recientes, aquellos que se publicaron dentro de los años 2013 al 2017. A manera general, un 74% de los algoritmos enfocados en detección de movimiento estuvieron dentro del periodo mencionado; es decir, 74 algoritmos de un total de 112. Lo anterior se muestra en la tabla

4.14. Por su parte, en los algoritmos enfocados a modelado de fondo, se obtuvo un 61%, es decir; 21 algoritmos de un total de 34, fueron clasificados como de reciente creación, considerando el periodo establecido.

La evaluación y puntuación completa de los algoritmos de modelado de fondo se muestra en la tabla 4.25. Los mejores diez algoritmos fueron LabGen, LabGen-P, Bewis, Temporal median filter, BE-AAPSA, MSCL, SC-SOBS-C4, MAGRPCA, Photomontage y Bidirectional Analysis. Por su parte, los diez mejores algoritmos de detección de movimiento que resultaron de la evaluación de los cinco criterios mencionados anteriormente, fueron: SuBSENSE, Cascade CNN, AAPSA, SharedModel, WeSamBE, DeepBS, SBM, PAWCS, FTSG y MBSV 0, y sus puntuaciones obtenidas a lo largo de la evaluación se muestran en la tabla 4.16-A. De estos algoritmos, siete, también se encuentran dentro de los diez primeros de la base de datos *Change detection*, esto se observa en la tabla 4.26. Dicho de otro modo, tres de los primeros diez algoritmos de la base de datos *Change detection* no figuran dentro de los diez primeros de la evaluación propuesta; estos son: IUTIS-5, IUTIS-3 y SaliencySuBSENSE. De los dos primeros, se detectó que no contaban con buenas puntuaciones en los criterios de documentación, velocidad y auto-adaptabilidad, ya que existen varios detalles que no son reportados en sus artículos tales como tiempo de procesamiento, resoluciones de video utilizadas y aspectos que conciernen en la parte de la implementación del método. Por otro lado, en la parte de auto-adaptabilidad, se identificó que ITUIS-5 y IUTIS-3 no eran en sí algoritmos de detección de movimiento, sino que son algoritmos de programación genética, que combinan los resultados de los mejores algoritmos de detección de movimiento para generar mejores resultados. Por otro lado, el artículo del algoritmo SaliencySuBSENSE no fue localizado para realizar su evaluación completa en los criterios documentación, auto-adaptabilidad y velocidad, por lo que, quedó en ceros en las tres evaluaciones, al no poder confirmarse la fuente. De estos resultados, se concluye que aún y cuando *Change detection* es una base de datos comúnmente usada como un método rápido de conocer y ubicar el impacto o importancia de un algoritmo, no es correcto considerarla como la única alternativa para realizar comparaciones entre algoritmos, ya que, para aplicaciones reales y fines científicos, importan de igual o mayor manera aspectos como la velocidad del algoritmo, la auto-adaptabilidad y su documentación.

6.2 Evaluación de los algoritmos y criterio de segmentación humana

Actualmente existe una gran controversia en la forma de evaluación que se implementa con cada método, debido a que los algoritmos son elaborados y probados en diferentes circunstancias donde están presentes diversos factores que no están establecidos como una norma o regla a seguir. Durante el desarrollo de esta tesis se identificaron tres principales problemas al realizar la evaluación de un algoritmo de detección de movimiento, los cuales son: los videos usados para probar el algoritmo, las métricas usadas en la evaluación y el *Ground truth*. La primera situación ocurre debido a que existe una gran cantidad de bases de datos de videos y cada autor usa y hace referencia a alguna de su agrado en específico. Cada video distinto presenta diferentes circunstancias y retos para la detección de movimiento, sin embargo, al cerrar la evaluación a una sola base de datos, el proceso se vuelve hermético a solo ciertas circunstancias, obteniendo una evaluación débil y dudosa. El segundo factor se presenta al no haber un consenso acerca de cuáles son las métricas más adecuadas para realizar las evaluaciones, lo que origina una carente interpretación de los datos. Por último, el *Ground truth* o la imagen de referencia, cuenta con un proceso de elaboración que es bastante cuestionable, por el hecho de que generalmente una persona es la que realiza esta imagen de referencia, y como es bien sabido cualquier proceso manual va acompañado de un cierto grado de error humano.

Con la finalidad de atender la problemática antes descrita, se realizaron dos experimentos. El primero consistió en evaluar el desempeño de cinco personas contra los *Ground truth* de la base de datos *Change detection*, esto con la finalidad de medir el grado de error humano o varianza que existe al realizar un *Ground truth*. En el segundo experimento se evaluó el desempeño de los diez mejores algoritmos de detección de movimiento MA_{DM} (véase tabla 4.16-A) contra los resultados obtenidos por las segmentaciones humanas, con el propósito de obtener una referencia de cómo se sitúan los algoritmos respecto al humano y si existían ocasiones en las que los algoritmos daban mejores resultados. En los experimentos descritos, se utilizaron las métricas reportadas en las bases de datos *Change detection* (Recall, Specificity, FPR, *F-Measure*, FNR, PWC, Precision) y se añadieron métricas enfocadas a medir la calidad de la segmentación que no son comúnmente reportadas en la literatura, *SSIM*, *D-Score* y FSD la cual, es una combinación de las dos últimas y F-Measure.

Las evaluaciones obtenidas de las segmentaciones humanas, los resultados y el promedio para cada métrica, se muestran en la tabla 5.2. De los sujetos en prueba, el sujeto cinco tenía conocimientos previos en realización de *Ground truth*, por lo que, presentó el mejor resultado, esto se observa en la figura 5.3, en la cual, claramente se ve que el sujeto cinco se encuentra por encima del promedio, en especial en las métricas *PWC*, *F-Measure* y *Precision.* Además, se realizó una comparación de dos de las métricas más reportadas en la literatura F-Measure y PWC contra las métricas enfocadas en medir la calidad de la imagen *SSIM* y *D-Score*. Los resultados de estas comparaciones mostraron que las métricas *F-Measure* y *PWC* realizan una evaluación muy estricta, ya que evalúan a nivel pixel, el número de pixeles (de fondo u objetos dinámicos) correctamente clasificados, por lo que, cualquier variación mínima respecto al *Ground truth* produce un impacto considerable en los resultados. Lo anterior se ve reflejado en que ninguno de los sujetos logró un resultado mayor a 0.966 en F-Measure y menor de 0.299 en PWC y en promedio, se obtuvo 0.948 y 0.441 para F-Measure y PWC respectivamente. Por su parte, las métricas SSIM y *D-Score* son más flexibles, ya que se enfocan en medir la calidad de segmentación de la imagen tomando en consideración la información estructural de la escena. El promedio humano en la métrica SSIM fue de 0.959 y 0.005 en *D-Score*. Se piensa que estas dos últimas métricas deberían ser más consideradas, a manera de incluirlas en las evaluaciones de algoritmos, debido a que son inspiradas bajo el proceso de precepción visual, y finalmente lo que se busca de un algoritmo de detección de movimiento, no es qué cantidad de pixeles puede detectar correctamente, si no que los pixeles que detecte sean significativos y visualmente se logre identificar el objeto de interés.

Los resultados obtenidos en la comparación de los mejores algoritmos de detección de movimiento MA_{DM} contra las segmentaciones manuales de cada uno de los sujetos se mostraron en la tabla 5.6, es interesante notar que el algoritmo Cascade CNN obtuvo el mejor desempeño en comparación a los demás algoritmos e incluso está por encima del desempeño promedio de segmentación humano. No obstante, el sujeto cinco se posiciona por encima de éste, cabe destacar que el sujeto cinco contaba con entrenamiento previo en realización de *Ground truth*. Por lo que, se puede concluir que de los diez algoritmos MA_{DM}, únicamente el algoritmo Cascade CNN, cae dentro de lo que podría considerarse como comportamiento humano de sujetos sin entrenamiento.

Basado en el análisis anterior, se propuso un rango de valores que pueden ser aceptables de acuerdo al desempeño humano para cada una de las métricas, esto tomando en consideración el menor desempeño de los sujetos y el desempeño ideal en cada métrica, este rango se muestra en la tabla 5.8. Una vez establecido el rango se observó qué tan cercanos estaban los algoritmos de caer dentro del rango humano (véase tabla 5.10), donde, el algoritmo Cascade CNN obtuvo 104%, es decir, que estuvo 4% arriba del rango de desempeño humano sin entrenamiento y el resto de los algoritmos lograron tener un porcentaje entre el 84% y 89% de proximidad a estar dentro del rango del desempeño humano. Estos rangos conducen a cuestionar nuevamente el cómo evaluar el desempeño de algoritmos de detección de movimiento y la posibilidad de incorporar criterios de desempeño humano para mejorar las formas de evaluar los algoritmos en las bases de datos, ya que los porcentajes anteriores logrados por los algoritmos son un indicador de que los desempeños obtenidos, no se encuentran tan alejados respecto a lo que se puede considerar comportamiento humano, y parece que los nuevos algoritmos van por buen camino, tal es el caso de Cascade CNN, el cual es un algoritmo de redes neuronales convolucionales que plantea un futuro prometedor en el área de detección de movimiento, ya que esta técnica logró tener un desempeño dentro del rango del desempeño humano promedio y además, logra afrontar correctamente todas las categorías de video excepto la categoría de videos de objetos con movimiento intermite. Aunque, también técnicas con diccionarios de aprendizaje y LBSP usadas por los algoritmos PAWCS y SuBSENSE logran situarse dentro de las primeras tres técnicas que mejor pueden afrontar situaciones de fondos dinámicos, objetos con movimiento intermitente, baja tasa de velocidad de cuadros, tomas nocturnas y sombras. Además, PAWCS y SuBSENSE como se mostró en la sección 4.1.1.4 pueden trabajar en tiempo real.

Los humanos, al igual o en mayor cantidad que los algoritmos tienden a fallar en categorías de video donde existen situaciones críticas de video tales como fondos dinámicos, camuflaje, problemas por desenfoque, entre otras. Por lo tanto, el proceso de realizar la evaluación respecto a un *Ground truth* realizado por un humano resulta en una evaluación un tanto subjetiva. Tanto más subjetiva, en cuanto menos entrenamiento tenga el humano. Sin embargo, nos puede dar un indicio de qué tan cercanos se encuentran los resultados de detección de los algoritmos respecto a lo que se considera un desempeño humano promedio.

Las técnicas para realizar la detección de movimiento que obtuvieron los mejores desempeños del análisis de la sección 5.3, fueron redes convolucionales, diccionarios de aprendizaje con LBSP y los modelos estadísticos no paramétricos. Por lo que, el uso en estas técnicas debería ser extendido con el fin de generar algoritmos con mejores características y desempeños similares a los de un humano promedio.

CAPÍTULO VII. TRABAJO FUTURO

Este trabajo de tesis propone una metodología de evaluación y comparación de algoritmos, tomando en consideración aspectos referentes a la documentación del método, su auto-adaptabilidad, velocidad, desempeño y actualidad. Por lo que, un trabajo futuro sería aplicar esta metodología como una regla general para realizar las comparaciones, de esta manera se utilizarían más criterios de evaluación para obtener los mejores algoritmos, en lugar de únicamente su desempeño, como usualmente se hace. También, este trabajo proporciona una vía para conocer cuáles técnicas de detección de movimiento tienen los mejores desempeños y para qué situaciones críticas de video. Por lo que, como trabajo futuro estas técnicas podrían ser más estudiadas y usadas con el fin de desarrollar mejores algoritmos. Por último, la metodología propuesta para la evaluación y comparación de algoritmos de detección de movimiento, podría ser extendida para proponer una metodología nueva para analizar y comparar algoritmos de seguimiento.

CAPÍTULO VIII. REFERENCIAS

[1] M. I. Chacon-Murguia, R. Sandoval-Rodriguez, y O. Arias-Enriquez, "Human gait feature extraction including a kinematic analysis toward robotic power assistance", *Int. J. Adv. Robot. Syst.*, vol. 9, pp. 1–9, 2012.

[2] T. Kimura, M. Ohashi, K. Crailsheim, T. Schmickl, R. Odaka, y H. Ikeno, "Tracking of Multiple Honey Bees on a Flat Surface", en *Fifth International Conference on Emerging Trends in Engineering and Technology*, 2012, pp. 36–39.

[3] M. Shakeri y H. Zhang, "Real-time bird detection based on background subtraction", *Proc. World Congr. Intell. Control Autom.*, pp. 4507–4510, 2012.

[4] M. Baillieul y P. Scheunders, "On-line determination of the velocity of simultaneously moving organisms by image analysis for the detection of sublethal toxicity", *Water Res.*, vol. 32, núm. 4, pp. 1027–1034, 1998.

[5] K. Toyama, J. Krumm, B. Brumitt, y B. Meyers, "Wallflower: principles and practice of background maintenance", *Comput. Vision, 1999. Proc. Seventh IEEE Int. Conf.*, pp. 255–261, 1999.

[6] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview", *Comput. Sci. Rev.*, vol. 11–12, núm. 1, pp. 31–66, 2014.

[7] Q. Zhou y J. K. Aggarwal, "Tracking and classifying moving objects from video", *Proc. IEEE Work. Perform. Eval. Track. Surveill.*, vol. 12, pp. 46–54, 2001.

[8] M.-H. Hung, J.-S. Pan, y C.-H. Hsieh, "A Fast Algorithm of Temporal Median Filter for Background Subtraction", *J. Inf. Hiding Multimed. Signal Process.*, vol. 5, núm. 1, pp. 33–40, 2014.

[9] G. M. Rao, "Object Tracking System Using Approximate Median Filter , Kalman Filter and Dynamic Template Matching", *Int. J. Intell. Syst. Appl.*, vol. 6, núm. 5, pp. 83–89, 2014.

[10] F. A. Díaz González y D. A. Arévalo Suárez, "Recursive median filter for background estimation and foreground segmentation in surveillance videos ", *Comput. y Sist.*, vol. 19, núm. 2, pp. 283–293, 2015.

[11] H. Al-Khateeb y M. Petrou, "Automatic Change Detection Method for an Indoor Environment", en *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, 2010, pp. 53–58.

[12] B. Nikolov y N. Kostov, "Motion Detection Using Adaptive Temporal Averaging Method", *Radioengineering*, vol. 23, núm. 2, pp. 652–658, 2014.

[13] T. S. Caetano, S. D. Olabarriaga, y D. A. C. Barone, "Performance evaluation of single and multiple-Gaussian models for skin color modeling", en *Conference: Brazilian Symposium of Computer Graphic and Image Processing*, 2002, pp. 275–282.

[14] K. Kim, T. H. Chalidabhongse, D. Harwood, y L. Davis, "Real-time foreground-

background segmentation using codebook model", *Real-Time Imaging*, vol. 11, núm. 3, pp. 172–185, 2005.

[15] T. Bouwmans, F. El Baf, y B. Vachon, "Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey", *Recent Patents Comput. Sci.*, vol. 1, núm. 3, pp. 219–237, 2008.

[16] D. Giordano, S. Palazzo, y C. Spampinato, "Kernel Density Estimation Using Joint Spatial-Color-Depth Data for Background Modeling", *2014 22nd Int. Conf. Pattern Recognit.*, pp. 4388–4393, 2014.

[17] M. Piccardi, "Background subtraction techniques: a review", *IEEE Int. Conf. Syst. Man Cybern. (IEEE Cat. No.04CH37583)*, vol. 4, pp. 3099–3104, 2004.

[18] Y. Benezeth, P. M. Jodoin, B. Emile, H. Laurent, y C. Rosenberger, "Review and evaluation of commonly-implemented background subtraction algorithms", en *2008 19th International Conference on Pattern Recognition*, 2008, pp. 1–4.

[19] N. Buch, S. A. Velastin, y J. Orwell, "A review of computer vision techniques for the analysis of urban traffic", *IEEE Trans. Intell. Transp. Syst.*, vol. 12, núm. 3, pp. 920–939, 2011.

[20] J. S. Kulchandani y K. J. Dangarwala, "Moving object detection: Review of recent research trends", *Pervasive Comput. (ICPC), 2015 Int. Conf.*, vol. 1, núm. c, pp. 1–5, 2015.

[21] Z. Wang, A. C. Bovik, H. R. Sheikh, y E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity", *IEEE Trans. Image Process.*, vol. 13, núm. 4, pp. 600–612, 2004.

[22] C. Lallier, E. Reynaud, L. Robinault, y L. Tougne, "A testing framework for background subtraction algorithms comparison in intrusion detection context", *2011 8th IEEE Int. Conf. Adv. Video Signal Based Surveillance, AVSS 2011*, pp. 314–319, 2011.

[23] T. Bouwmans, L. Maddalena, y A. Petrosino, "Scene background initialization: A taxonomy", *Pattern Recognit. Lett.*, vol. 0, pp. 1–9, 2016.

[24] A. Vacavant, L. Tougne, y L. Robinault, "Evaluation of Background Models with Synthetic and Real Data", en *Background Modeling and Foreground Detection for Video Surveillance*, 2015, pp. 1–14.

[25] S. Mahmoudpour y M. Kim, "Robust foreground detection in sudden illumination change", *Electron. Lett.*, vol. 52, núm. 6, pp. 441–443, 2016.

[26] O. El Harrouss, D. Moujahid, y H. Tairi, "Motion detection based on the combining of the background subtraction and spatial color information", en *2015 Intelligent Systems and Computer Vision, ISCV 2015*, 2015, pp. 2–5.

[27] W. Hossain y M. N. Das, "Moving Object Detection in Dynamic Backgrounds for Surveillance Systems", en *IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, 2014, núm. 978,

pp. 1476–1479.

[28] K. K. Delibasis, T. Goudas, y I. Maglogiannis, “A novel robust approach for handling illumination changes in video segmentation”, *Eng. Appl. Artif. Intell.*, vol. 49, pp. 43–60, 2016.

[29] K. Wang, Y. Liu, C. Gou, y F. Wang, “A Multi-view Learning Approach to Foreground Detection for Traffic Surveillance Applications”, *IEEE Trans. Veh. Technol.*, vol. 65, núm. 6, pp. 4144–4158, 2016.

[30] Z. Chen y T. Ellis, “A self-adaptive Gaussian mixture model”, *Comput. Vis. Image Underst.*, vol. 122, núm. 1, pp. 35–46, 2014.

[31] C. Spampinato, S. Palazzo, y I. Kavasidis, “A texton-based kernel density estimation approach for background modeling under extreme conditions”, *Comput. Vis. Image Underst.*, vol. 122, pp. 74–83, 2014.

[32] M. I. Chacon-Murguia y D. Urias-Zavala, “A DTCNN Approach on Video Analysis: Dynamic and Static Object Segmentation”, *Stud. Comput. Intell.*, vol. 551, pp. 315–336, 2014.

[33] G. Ramírez-Alonso y M. I. Chacón-Murguía, “Auto-Adaptive Parallel SOM Architecture with a modular analysis for dynamic object segmentation in videos”, *Neurocomputing*, vol. 175, núm. November, pp. 990–1000, 2016.

[34] J. A. Ramirez-Quintana y M. I. Chacon-Murguia, “An Adaptive Unsupervised Neural Network Based on Perceptual Mechanism for Dynamic Object Detection in Videos with Real Scenarios”, *Neural Process. Lett.*, vol. 42, núm. 3, pp. 665–689, 2015.

[35] J. A. Ramirez-Quintana y M. I. Chacon-Murguia, “Self-adaptive SOM-CNN neural system for dynamic object detection in normal and complex scenarios”, *Pattern Recognit.*, vol. 48, núm. 4, pp. 1133–1145, 2015.

[36] Z. Zhao, X. Zhang, Y. Fang, y S. Member, “Stacked Multi-layer Self-Organizing Map for Background Modeling”, *IEEE Trans. IMAGE Process.*, vol. 7149, núm. c, pp. 1–10, 2015.

[37] M. J. Shafiee, P. Siva, P. Fieguth, y A. Wong, “Embedded Motion Detection via Neural Response Mixture Background Modeling”, en *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, pp. 19–26.

[38] M. Braham y M. Van Droogenbroeck, “Deep background subtraction with scene-specific convolutional neural networks”, en *International Conference on Systems, Signals, and Image Processing*, 2016, pp. 3–6.

[39] R. Guo y H. Qi, “Partially-sparse restricted boltzmann machine for background modeling and subtraction”, en *2013 12th International Conference on Machine Learning and Applications, ICMLA 2013*, 2013, pp. 209–214.

[40] M. Giordano y M. De Gregorio, “Background Modeling by Weightless Neural Networks”, *New Trends Image Anal. Process. -- ICIAP 2015 Work. Lect. Notes Comput. Sci.*, vol. 9281, núm. Springer, pp. 493–501, 2015.

[41] B. N. Subudhi, S. Ghosh, y A. Ghosh, “Application of Gibbs–Markov random field and Hopfield-type neural networks for detecting moving objects from video sequences captured by static camera”, *Soft Comput.*, vol. 19, núm. 10, pp. 2769–2781, 2015.

[42] S.-C. Huang y B.-H. Do, “Radial Basis Function Based Neural Network for Motion Detection in Dynamic Scenes.”, *IEEE Trans. Cybern.*, vol. 44, núm. 1, pp. 114–125, 2013.

[43] S. Lee, N. Kim, K. Jeong, I. Paek, H. Hong, y J. Paik, “Multiple moving object segmentation using motion orientation histogram in adaptively partitioned blocks for high-resolution video surveillance systems”, *Optik (Stuttg).*, vol. 126, núm. 19, pp. 2063–2069, 2015.

[44] L. Gervasoni y R. Barbuzza, “Aplicación de un método de sustracción de fondo a partir de imágenes de vídeo-vigilancia”, en *15th Argentine Symposium on Technology*, 2014, pp. 25–36.

[45] J. Dou y J. Li, “Moving object detection based on improved VIBE and graph cut optimization”, *Opt. - Int. J. Light Electron Opt.*, vol. 124, núm. 23, pp. 6081–6088, 2013.

[46] J. Pan, W. Chen, y W. Peng, “A new moving objects detection method based on improved SURF algorithm”, en *2013 25th Chinese Control and Decision Conference (CCDC)*, 2013, pp. 901–906.

[47] J. Dou y J. Li, “Modeling the background and detecting moving objects based on Sift flow”, *Optik (Stuttg).*, vol. 125, núm. 1, pp. 435–440, 2014.

[48] F. J. López-Rubio y E. López-Rubio, “Foreground detection for moving cameras with stochastic approximation”, *Pattern Recognit. Lett.*, vol. 68, pp. 161–168, 2015.

[49] H. Lu, Yingying Chen, Jinqiao Wang, “Learning Sharable Models For Robust Background Subtraction”, en *IEEE International Conference on Multimedia and Expo (ICME)*, 2015, pp. 1–6.

[50] R. Azzam, M. S. Kemouche, N. Aouf, y M. Richardson, “Efficient visual object detection with spatially global Gaussian mixture models and uncertainties”, *J. Vis. Commun. Image Represent.*, vol. 36, núm. C, pp. 90–106, 2016.

[51] P.-L. St-Charles, G.-A. Bilodeau, y R. Bergevin, “SuBSENSE: A Universal Change Detection Method With Local Adaptive Sensitivity”, *IEEE Trans. Image Process.*, vol. 24, núm. 1, pp. 359–373, 2015.

[52] P. Chiranjeevi y S. Sengupta, “Detection of moving objects using multi-channel kernel fuzzy correlogram based background subtraction”, *IEEE Trans. Cybern.*, vol. 44, núm. 6, pp. 870–881, 2014.

[53] D. K. Panda y S. Meher, “Detection of Moving Objects Using Fuzzy Color Difference Histogram Based Background Subtraction”, *IEEE Signal Process. Lett.*, vol. 23, núm. 1, pp. 45–49, 2016.

[54] B. Yang y L. Zou, “Robust foreground detection using block-based RPCA”, *Optik*

(Stuttg)., vol. 126, núm. 23, pp. 4586–4590, 2015.

[55] M. Shakeri y H. Zhang, "COROLA: A sequential solution to moving object detection using low-rank approximation", *Comput. Vis. Image Underst.*, vol. 146, pp. 27–39, 2015.

[56] W. Kim y Y. Kim, "Background Subtraction Using Illumination-Invariant Structural Complexity", *IEEE Signal Process. Lett.*, vol. 23, núm. 5, pp. 634–638, 2016.

[57] X. Cao, L. Yang, y X. Guo, "Total Variation Regularized RPCA for Irregularly Moving Object Detection Under Dynamic Background", *IEEE Trans. Cybern.*, vol. 46, núm. 4, pp. 1014–1027, 2016.

[58] M. Radolko, F. Farhadifard, E. Gutzeit, y U. F. Von Lukas, "Real time video segmentation optimization with a modified Normalized Cut", en *9th International Symposium on Image and Signal Processing and Analysis, ISPA 2015*, 2015, pp. 31–36.

[59] S. Javed, S. H. Oh, A. Sobral, T. Bouwmans, y S. K. Jung, "Background Subtraction via Superpixel-Based Online Matrix Decomposition with Structured Foreground Constraints", en *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, 2015, pp. 930–938.

[60] P. L. St-Charles, G. A. Bilodeau, y R. Bergevin, "A self-adjusting approach to change detection based on background word consensus", en *IEEE Winter Conference on Applications of Computer Vision, WACV 2015*, 2015, pp. 990–997.

[61] A. Staglianò, N. Noceti, A. Verri, y F. Odone, "Online space-variant background modeling with sparse coding", *IEEE Trans. Image Process.*, vol. 24, núm. 8, pp. 2415–2428, 2015.

[62] P. Dong, S. Wang, Y. Xia, D. Liang, y D. D. Feng, "Foreground detection with simultaneous dictionary learning and historical pixel maintenance", *IEEE Trans. Image Process.*, vol. 25, núm. 11, pp. 5035–5049, 2016.

[63] C. David y V. Gui, "Sparse Coding and Gaussian Modeling of Coefficients Average for Background Subtraction", en *2013 8th International Symposium on Image and Signal Processing and Analysis (ISPA)*, 2013, pp. 223–228.

[64] J. Šoda, I. Kuzmanić, y I. Vujović, "Stabilising illumination variations in motion detection for surveillance applications", *IET Image Process.*, vol. 7, núm. 7, pp. 671–678, 2013.

[65] M. N. A. F, M. A. S. M, H. M. Ebeid, y M. F. Tolba, "Wavelet-Enhanced Detection of Small / Slow Object Movement in Complex Scenes", en *2016 11th International Conference on Computer Engineering & Systems (ICCES)*, 2016, pp. 172–180.

[66] R. Wang, F. Bunyak, G. Seetharaman, y K. Palaniappan, "Static and moving object detection using flux tensor with split gaussian models", en *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 420–424.

[67] Q. Zhao, G. Zhou, L. Zhang, A. Cichocki, y S. I. Amari, "Bayesian Robust Tensor Factorization for Incomplete Multiway Data", *IEEE Trans. Neural Networks Learn. Syst.*, vol. 27, núm. 4, pp. 736–748, 2016.

[68] S. Bianco, G. Ciocca, y R. Schettini, "How Far Can You Get By Combining Change Detection Algorithms?", *IEEE Trans. Image Process.*, vol. 2, pp. 1–10, 2015.

[69] B. Dey y M. K. Kundu, "Efficient Foreground Extraction from HEVC Compressed Video for Application to Real-Time Analysis of Surveillance 'Big' Data", *IEEE Trans. Image Process.*, vol. 24, núm. 11, pp. 3574–3585, 2015.

[70] Z. Zivkovic y F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction", *Pattern Recognit. Lett.*, vol. 27, núm. 7, pp. 773–780, 2006.

[71] Z. Chen y T. Ellis, "Self-adaptive Gaussian mixture model for urban traffic monitoring system", en *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2011, pp. 1769–1776.

[72] M. Narayana, A. Hanson, y E. Learned-miller, "Background Modeling Using Adaptive Pixelwise Kernel Variances in a Hybrid Feature Space", en *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2104–2111.

[73] Y. LeCun, L. Bottou, Y. Bengio, y P. Haffner, "Gradient-based learning applied to document recognition", *Proc. IEEE*, vol. 86, núm. 11, pp. 2278–2323, 1998.

[74] M. De Gregorio, I. Cibernetica, E. C. I. Cnr, M. Giordano, P. Icar, y V. P. Castellino, "Change Detection with Weightless Neural Networks", en *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 409–413.

[75] T. Ojala, M. Pietikäinen, y T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, núm. 7, pp. 971–987, 2002.

[76] H. Bay, T. Tuytelaars, y L. Van Gool, "SURF: Speeded up robust features", en *European Conference on Computer Vision- ECCV*, 2006, pp. 404–417.

[77] C. Liu, J. Yuen, y A. Torralba, "Sift flow: Dense correspondence across scenes and its applications", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, núm. 5, pp. 978–994, 2011.

[78] C. Pojala y S. Sengupta, "Detection of moving objects using fuzzy correlogram based background subtraction", en *IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2011, pp. 255–259.

[79] G.-H. Liu y J.-Y. Yang, "Content-based image retrieval using color difference histogram", *Pattern Recognit.*, vol. 46, núm. 1, pp. 188–198, 2013.

[80] X. Zhou, C. Yang, y W. Yu, "Moving object detection by detecting contiguous outliers in the low-rank representation", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, núm. 3, pp. 597–610, 2013.

[81] J. Feng y H. Xu, "Online Robust PCA via Stochastic Optimization", *Adv. Neural Inf.*

Process. Syst., vol. 26, pp. 1–9, 2013.

[82] M. Aharon, M. Elad, y A. M. Bruckstein, "K-Svd : Design of Dictionaries for Sparse Representation", *Spars'05*, vol. 1, pp. 9–12, 2005.

[83] T. G. Kolda y B. W. Bader, "Tensor Decompositions and Applications", *SIAM Rev.*, vol. 51, núm. 3, pp. 455–500, 2009.

[84] Y. Wang, Z. Luo, y P. M. Jodoin, "Interactive deep learning method for segmenting moving objects", *Pattern Recognit. Lett.*, vol. 0, pp. 1–10, 2016.

[85] S. Jiang y X. Lu, "WeSamBE: A Weight-Sample-Based Method for Background Subtraction", *IEEE Trans. Circuits Syst. Video Technol.*, vol. PP, núm. 99, pp. 1–11, 2017.

[86] M. Babaee, D. T. Dinh, y G. Rigoll, "A Deep Convolutional Neural Network for Background Subtraction", *Comput. Res. Repos.*, vol. abs/1702.0, pp. 1–28, 2017.

[87] M. Chen, Q. Yang, Q. Li, G. Wang, y M. Yang, "Spatiotemporal Background Subtraction Using Minimum Spanning Tree and Optical Flow", en *Computer Vision–ECCV 2014*, 2014, pp. 521–534.

[88] P. L. St-Charles, G. A. Bilodeau, y R. Bergevin, "A self-adjusting approach to change detection based on background word consensus", *Proc. - 2015 IEEE Winter Conf. Appl. Comput. Vision, WACV 2015*, pp. 990–997, 2015.

[89] H. Sajid y S.-C. S. Cheung, "Background Subtraction for Static & Moving Camera", en *International Conference on Image Processing (ICIP)*, 2015, pp. 4530–4534.

[90] P. K. King y R. Bowden, "An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection", en *Video-Based Surveillance Systems*, 2002, pp. 135–144.

[91] B. Wang y P. Dudek, "A Fast Self - tuning Background Subtraction Algorithm", en *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 4321–4324.

[92] H. Sajid y S.-C. S. Cheung, "Universal Multimode Background Subtraction", *IEEE Trans. Image Process.*, vol. 26, núm. 7, pp. 3249–3260, 2017.

[93] P. Tiefenbacher, M. Hofmann, D. Merget, y G. Rigoll, "PID-based regulation of background dynamics for foreground segmentation", en *IEEE International Conference on Image Processing, ICIP*, 2014, pp. 3282–3286.

[94] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction", en *Proceedings of the 17th International Conference on Pattern Recognition*, 2004, pp. 28–31.

[95] R. Heras Evangelio y T. Sikora, "Complementary background models for the detection of static and moving objects in crowded environments", en *8th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS*, 2011, pp. 71–76.

[96] J. L. Martins, I., Carvalho, P., Corte-Real, L., Alba-Castro, "BMOG: Boosted Gaussian Mixture Model with Controlled Complexity", en *Pattern Recognition and Image Analysis. IbPRIA 2017*, 2017, pp. 50–57.

[97] F. J. Hernandez-Lopez y M. Rivera, "Change detection by probabilistic segmentation from monocular view", *Mach. Vis. Appl.*, vol. 25, núm. 5, pp. 1175–1195, 2014.

[98] M. Shah, J. D. Deng, y B. J. Woodford, "Illumination invariant background model using mixture of Gaussians and SURF features", en *Computer Vision (ACCV 2012 Workshops)*, vol. 7728, 2012, pp. 308–314.

[99] A. Schick, M. Bauml, y R. Stiefelhagen, "Improving Foreground Segmentation with Probabistic Superpixel Markov Random Fields", en *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2012, pp. 27–31.

[100] M. De Gregorio y M. Giordano, "WiSARDRP for Change Detection in Video Sequences", en *25th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2017, pp. 453–458.

[101] A. Glazer, M. Lindenbaum, y S. Markovitch, "One-class background model", en *Computer Vision (ACCV 2012 Workshops)*, vol. 7728, 2012, pp. 301–307.

[102] C. Guyon, T. Bouwmans, y E. H. Zahzah, "Foreground detection via robust low rank matrix decomposition including spatio-temporal constraint", *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7728 LNCS, núm. PART 1, pp. 315–320, 2013.

[103] G. Allebosch, D. Van Hamme, F. Deboeverie, P. Veelaert, y W. Philips, *C-EFIC: Color and Edge Based Foreground Background Segmentation with Interior Classification*, vol. 598. 2015.

[104] L. Li, W. Huang, I. Y. H. Gu, y Q. Tian, "Foreground object detection from videos containing complex background", en *Proceedings of the eleventh ACM international conference on Multimedia MULTIMEDIA 03*, 2003, pp. 2–10.

[105] T. R. -i. Yoshinaga S. ,Shimada A., Nagahara H., "Background Model Based on Statistical Local Difference Pattern", en *Computer Vision (ACCV 2012 Workshops)*, vol. 7728, 2012, pp. 327–332.

[106] Y. Goyat, T. Chateau, L. Malaterre, y L. Trassoudaine, "Vehicle trajectories evaluation by static video sensors", en *2006 IEEE Intelligent Transportation Systems Conference*, 2006, pp. 864–869.

[107] G. Allebosch, F. Deboeverie, P. Veelaert, y W. Philips, "EFIC: Edge Based Foreground Background Segmentation and Interior Classification for Dynamic Camera Viewpoints", *Adv. Concepts Intell. Vis. Syst. Lect. Notes Comput. Sci.*, vol. 9386, pp. 130–141.

[108] A. Morde, X. Ma, y S. Guler, "Learning a background model for change detection", en *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 15–20.

[109] A. Miron y A. Badii, “Change detection based on graph cuts”, en *International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2015, pp. 273–276.

[110] S. Varadarajan, P. Miller, y H. Zhou, “Spatial mixture of Gaussians for dynamic background modelling”, en *10th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2013*, 2013, pp. 63–68.

[111] Y. Benezeth, P. Jodoin, B. Emile, y C. Rosenberger, “Comparative study of background subtraction algorithms”, *J. Electron. Imaging, Soc. Photooptical Instrum. Eng.*, vol. 19, núm. 3, pp. 1–30, 2012.

[112] D. Liang, S. Kaneko, M. Hashimoto, K. Iwata, y X. Zhao, “Co-occurrence probability-based pixel pairs background model for robust object detection in dynamic scenes”, *Pattern Recognit.*, vol. 48, núm. 4, pp. 1374–1390, 2015.

[113] X. Lu, “A multiscale spatio-temporal background model for motion detection”, en *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 3268–3271.

[114] C. Stauffer y W. E. L. Grimson, “Adaptive background mixture models for real-time tracking”, en *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999, pp. 246–252.

[115] S. Yoshinaga, A. Shimada, H. Nagahara, y R. Taniguchi, “Background Model Based on Intensity Change Similarity Among Pixels”, en *19th Japan-Korea Joint Workshop on Frontiers of Computer Vision*, 2013, pp. 276–280.

[116] F. Seidel, C. Hage, y M. Kleinsteuber, “pROST - A Smoothed Lp-norm Robust Online Subspace Tracking Method for Realtime Background Subtraction in Video”, *Mach. Vis. Appl.*, vol. 25, pp. 1227–1240, 2014.

[117] H. R. Tavakoli, E. Rahtu, y J. Heikkilä, “Temporal Saliency for Fast Motion Detection”, *Comput. Vis. (ACCV 2012 Work.*, vol. 7728, pp. 321–326, 2012.

[118] A. Elgammal, D. Harwood, y L. Davis, “Non-parametric model for background subtraction”, en *6th European Conference on Computer Vision, Dublin*, 2000, pp. 751–767.

[119] S. M. H. Ismail, “Object Segmentation Using Full-Spectrum Matching of Albedo Derived from Colour Images”, *UK patent application no. 0822953.6 16.12.2008 GB, 2008, PCT patent application international application no. PCT/GB2009/002829, EP2374109, 2009, US patent no. 2374109 12.10.2011 US*. p. , 2011.

[120] M. Hofmann, P. Tiefenbacher, y G. Rigoll, “Background segmentation with feedback: The Pixel-Based Adaptive Segmenter”, en *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 38–43.

[121] R. H. Evangelio, M. Pätzold, y S. T., “Splitting gaussians in mixture models”, en *IEEE 9th International Conference on Advanced Video and Signal-Based Surveillance, AVSS 2012*, 2012, pp. 300–305.

[122] L. Maddalena y A. Petrosino, “A Fuzzy Spatial Coherence-based Approach to

Background/ Foreground Separation for Moving Object Detection", *Neural Comput. Appl.*, vol. 19, pp. 179–186, 2010.

[123] N. Wang, T. Yao, y J. Wang, "A probabilistic approach to robust matrix factorization[M]", en *Computer Vision–ECCV 2012. Springer Berlin Heidelberg*, 2012, pp. 126–139.

[124] J. Yao y J. Odobez, "Multi-Layer Background Subtraction Based on Color and Texture", en *IEEE Conference on Computer Vision and Pattern Recognition, CVPR '07.*, 2007, pp. 1–8.

[125] L. Maddalena y A. Petrosino, "A Self-organizing approach to background subtracion for visual surveillance applications", *IEEE Trans. IMAGE Process.*, vol. 17, núm. 7, pp. 1168–1177, 2008.

[126] R. T. Y.Nonaka, A. Shimada, H.Nagahara, "Evaluation Report of Integrated Background Modeling Based on Spatio-temporal Features", en *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, 2012, pp. 9–14.

[127] F. Porikli y O. Tuzel, "Bayesian background modeling for foreground detection", en *the third ACM international workshop on Video surveillance & sensor networks - VSSN '05*, 2005, p. 55.

[128] J. Jodoin y G. Bilodeau, "Background subtraction based on Local Shape", *Arxiv*, vol. 1204.6326, pp. 5–8, 2012.

[129] D. Riahi, P.-L. St-Onge, y G.-A. Bilodeau, "RECTGAUSS-Tex : Block-based Background Subtraction", 2012.

[130] D. S. Lee, "Effective Gaussian mixture learning for video background subtraction", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, núm. 5, pp. 827–832, 2005.

[131] M. E. H. J. Zheng, Y. Wang, N.L. Nihan, "Extracting roadway background image : Mode-based approach", *Transp. Res. Rec. J. Transp. Res. Board*, vol. 10.3141/19, pp. 82–88, 2006.

[132] B. Laugraud, S. Piérard, y M. Van Droogenbroeck, "LaBGen-P: A pixel-level stationary background generation method based on LaBGen", en *International Conference on Pattern Recognition*, 2017, pp. 107–113.

[133] G. Ramirez-Alonso, J. A. Ramirez-Quintana, y M. I. Chacon-Murguia, "Temporal weighted learning model for background estimation with an automatic re-initialization stage and adaptive parameters update", *Pattern Recognit. Lett.*, vol. 96, pp. 34–44, 2017.

[134] M. Piccardi, "Background subtraction techniques: a review", *IEEE Int. Conf. Syst. Man Cybern. (IEEE Cat. No.04CH37583)*, vol. 4, pp. 3099–3104, 2004.

[135] S. Javed, A. Mahmood, T. Bouwmans, y S. K. Jung, "Background-Foreground Modeling Based on Spatiotemporal Sparse Subspace Clustering", *IEEE Trans. Image Process.*, vol. 26, núm. 12, pp. 5840–5854, 2017.

[136] L. Maddalena y A. Petrosino, “Extracting a background image by a multi-modal scene background model”, en *International Conference on Pattern Recognition*, 2017, pp. 143–148.

[137] S. Javed, S. K. Jung, A. Mahmood, y T. Bouwmans, “Motion-Aware Graph Regularized RPCA for background modeling of complex scenes”, en *Proceedings - International Conference on Pattern Recognition*, 2017, pp. 120–125.

[138] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, y M. Cohen, “Interactive digital photomontage”, *ACM Trans. Graph.*, vol. 23, núm. 3, p. 294, 2004.

[139] A. S. and R. i. T. T. Minematsu, “Background initialization based on bidirectional analysis and consensus voting”, en *23rd International Conference on Pattern Recognition (ICPR), Cancun*, 2016, pp. 126–131.

[140] W. Liu, Y. Cai, M. Zhang, H. Li, y H. Gu, “Scene background estimation based on temporal median filter with Gaussian filtering”, en *International Conference on Pattern Recognition*, 2017, pp. 132–136.

[141] I. Halfaoui, F. Bouzaraa, y O. Urfalioglu, “CNN-based initial background estimation”, en *International Conference on Pattern Recognition*, 2017, pp. 101–106.

[142] X. Ye, J. Yang, X. Sun, K. Li, C. Hou, y Y. Wang, “Foreground-Background Separation from Video Clips via Motion-Assisted Matrix Restoration”, *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, núm. 11, pp. 1721–1734, 2015.

[143] J. M. M. D. Ortego, J. C. SanMiguel, “Rejection based multipath reconstruction for background estimation in video sequences with stationary objects”, *Comput. Vis. Image Underst.*, vol. 147, pp. 23–37, 2016.

[144] V. Reddy, C. Sanderson, y B. C. Lovell, “A Low-Complexity Algorithm for Static Background Estimation from Cluttered Image Sequences in Surveillance Contexts”, *Eurasip J. Image Video Process.*, vol. 2011, pp. 1–13, 2011.

[145] J. Xu, V. K. Ithapu, L. Mukherjee, J. M. Rehg, y V. Singh, “GOSUS: Grassmannian online subspace updates with structured-sparsity”, en *IEEE International Conference on Computer Vision*, 2013, pp. 3376–3383.

[146] M. Y. Guo X., Wang X., Yang L., Cao X., “Robust Foreground Detection Using Smoothness and Arbitrariness Constraints”, *Comput. Vis. – ECCV 2014. ECCV 2014. Lect. Notes Comput. Sci.*, vol. 8695, pp. 535–550, 2014.

[147] D. D. Bloisi, A. Pennisi, y L. Iocchi, “Parallel Multi-modal Background Modeling”, *Pattern Recognit. Lett.*, vol. 96, pp. 45–54, 2017.

[148] B. Vandereycken, “Low-Rank Matrix Completion by Riemannian Optimization”, *SIAM J. Optim.*, vol. 23, núm. 2, pp. 1214–1236, 2013.

[149] Z. S. Y. Xu , R. Hao , W. Yin, “Parallel matrix factorization for low-rank tensor completion”, *Inverse Probl. Imaging*, vol. 9, núm. 2, pp. 601–624, 2013.

[150] T. Zhou y D. Tao, “GoDec : Randomized Low-rank & Sparse Matrix Decomposition

in Noisy Case", en *28th International Conference on Machine Learning, Bellevue,WA, USA*, 2011, p. 8.

[151] J. He, L. Balzano, y A. Szlam, "Incremental gradient on the Grassmannian for online foreground and background separation in subsampled video", en *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1568–1575.

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

[153] H. Wang y D. Suter, "A novel robust statistical method for background initialization and visual surveillance", en *Computer Vision (ACCV 2006 Workshop)*, 2006, pp. 328–337.

CAPÍTULO IX. APÉNDICES

APÉNDICE A. CRITERIO DOCUMENTACIÓN.

A-1. Puntuaciones otorgadas a los algoritmos del 1 a 20 en el elemento procesamiento *Pr*.

Método		Procesamiento (*Pr* = Pr_{ResFPS} + Pr_{Res} + Pr_{EspC} + $Pr_{NúmV}$)						
Ranking	Identificación	$Pr_{NúmV}$	Videos	Pr_{ResFPS} + Pr_{Res}	Resolución	Resolución->FPS	Pr_{EspC}	Espacio de color
1	Multimode Background Subtraction(MBS)	11.2	56	2	320 × 240	320 × 240->10	1	RGB,YCbCr
2	BGWiS o CWisarDH	12	60	2	352x240 320x240 200x136 200x148 146x150	320×240->18 720x480->4	1	RGB HSV Lab
3	AAPSA	12.4	62	1	NE	240x320->20	1	HSV
4	DeepBS (Supervised method)	12	60	1	320 × 240	NE	1	RGB
5	Cascade CNN(supervised method)	10.6	53	2	320 × 240	320 ×240->12.5	1	RGB
6	TEDV	7	35	2	160x120 320x240 640x480	320x240->18.5	1	RGB HSV Lab
8	SuBSENSE	10.6	53	1	NE	Promedio general->45	1	RGB
7	WeSamBE	10.6	53	2	320 × 240	320 ×240->2	1	RGB
9	Multimode Background Subtraction Version 0 (MBS V0)	10.6	53	2	320x240	320x240->8.5	1	RGB YCbCr
10	SharedModel	10.6	53	1	NE	320x240->35	1	RGB
11	FTSG	10.6	53	1	NE	320x240->10	1	RGB
12	BREW-DLHPM	10.6	53	1	NE	320x240->61.01 432x288->36.42 720x480->13.04	0	NE
13	STBM	8	40	2	800x600 320x240	320x240->1.018 320x240->12 (GPU)	0	NE
14	SBM	8	40	2	800x600 320x240	320x240->66.6	0	NE
15	PAWCS	10.6	53	0	NE	NE	1	RGB
16	IUTIS-5	10.6	53	0	NE	NE	1	Binary
17	IUTIS-3	10.6	53	0	NE	NE	1	Binary
18	IUTIS-1	10.6	53	0	NE	NE	1	Binary
19	NeRM	9	45	1	NE	352x240->2	0	NA
20	MDLS	6.8	34	2	128x160	128x160->12.31	0	NA

A-2. Puntuaciones otorgadas a los algoritmos del 21 a 39 en el elemento procesamiento *Pr*.

Método		Procesamiento ($Pr = Pr_{ResFPS} + Pr_{Res} + Pr_{EspC} + Pr_{NúmV}$)						
Ranking	Identificación	$Pr_{NúmV}$	Videos	$Pr_{ResFPS} + Pr_{Res}$	Resolución	Resolución->FPS	Pr_{EspC}	Espacio de color
21	COROLA	4.4	22	2	160x128 176x144 160x128 160x130 320x256 320x240 160x120	320x240->12	0	NE
22	RESOM	3	15	2	120×160 128x160 240x320 244x380 288x352 288x432 480x640 480x720	120×160->65 128x160->58 240x320->15 244x380->12 288x352->11 288x432->10 480x640->4 480x720->4	1	RGB
23	BMTDL	8	40	1	NE	640×480->(0.77-21.10)	0	NE
24	TVRPCA	3	15	1	NE	128x160-> 0.9	1	GraysScale
25	SMSOM-BM	3.8	19	2	320x240 352x288 360x240 432x288 720x576 720x480	320x240->34 352x288->26.6 360x240->27.2 432x288->20 720x576->7 720x480->7.2	1	HSV
26	SOM-CNN	1.2	6	2	120x160	120x160->35	1	HSV
27	CTU	1.2	6	2	720×420	720×420->169.1	1	YCbCr
28	EAMV	1.2	6	1	NE	320x240->0.23	1	RGB
29	Temporal saliency	3.8	19	2	320x240	320x240->14.2	1	LMS
30	IISC	0.4	2	2	320x240	320x240->(2.5-6.25)	1	RGB
31	MKFC	2.2	11	1	NE	160x120->16	1	RGB
32	FCDH	1.2	6	0	NE	NE	1	Lab
33	ERCI	1	5	2	480x640 240x320	240x320gray_nb.2->10.42 240x320gray_nb.4->15.21 240x320gray_nb.8->16.51 240x320RGB_nb.8->7.31	1	GraysScale RGB
34	SAG	1.8	9	2	640x480 320x240	320x240->8.3	1	RGB
35	GMM & SURF combination	3.8	19	0	NE	NE	1	YUV
36	RBFMD	1.2	6	2	320×240 160x128	320×240->34 160x128->54	1	HSV
37	SGGMM and Uncertains	0.8	4	2	320x240 480x320 640x480 720x480	320x240->33.3 480x320->25 640x480->16.7 720x480->13	1	RGB
38	Variación ViBe	0.8	4	2	320x240	320x240->137 360x240->125	1	RGB
39	SiftFlow	0.8	4	2	160x120 320x240	160 × 120-> 10 320x240-> 3	1	RGB

A-3. Puntuaciones otorgadas a los algoritmos del 40 a 67 en el elemento procesamiento *Pr*.

Método		Procesamiento (*Pr* = Pr_{ResFPS} + Pr_{Res} + Pr_{EspC} + $Pr_{NúmV}$)						
Ranking	Identificación	$Pr_{NúmV}$	Videos	Pr_{ResFPS} + Pr_{Res}	Resolución	Resolución->FPS	Pr_{EspC}	Espacio de color
40	Statistical local difference pattern	3.8	19	0	NE	NE	0	NE
41	One-class classification	3.8	19	0	NE	NE	1	RGB
42	EVIEVV	0.2	1	2	576 × 720	576 × 720->2	1	RGB
43	ConvNets	4.2	21	0	NE	NE	1	GraysScale
44	PBAS-PID	6.2	31	0	NE	NE	0	NE
45	Normalized cut	1	5	2	120x160 1080x1920	120x160->20.48 1080x1920->6	1	RGB
46	OMHBP	0.8	4	2	768x576 1920x1080	768x576->78.21 1920x1080->17.53	0	NA
47	AEMF	1.8	9	0	NE	NE	1	RGB
48	GMRF y HTNN	0.6	3	0	NE	NE	0	NE
49	DMW	1.2	6	0	NE	NE	0	NE
50	MFESVV	0.4	2	1	NE	Promedio->(8.3-10)	0	NE
51	Block-based RPCA	1.2	6	0	NE	NE	0	NE
52	K-SVD	1	5	0	NE	NE	1	GrayScale
53	BC, or Bayesian classification	1	5	0	NE	NE	1	RGB
54	PS-RBM	0.8	4	0	NE	NE	0	NE
55	BRTF	0.6	3	1	256x320 144x176 128x160	NE	1	RGB
56	AFPA	0.6	3	1	768x576	NE	0	NE
57	DMIEC	1	5	0	NE	NE	1	RGB
58	SGMM-SOD	1	5	0	NE	NE	0	NE
59	DRCIMF	0.6	3	1	NE	Promedio->218	0	NE
60	GC STViBe	0.4	2	0	NE	NE	1	CIE-Luv
61	ISURF	0.2	1	2	720X480	720X480->6.6	0	NE
62	Robust low rank matrix decomposition	1.8	9	0	NE	NE	0	NE
63	FMR	0.2	1	1	320 x 160	NE	1	GraysScale
64	GMM | KaewTraKulPong	0.2	1	1	192x144	NE	1	RGB
65	DTCNN	1	5	0	NE	NE	1	HSV
66	GMM | Zivkovic	0.6	3	2	320×240	320×240->50.76	1	RGB
67	VuMeter	0	0	0	NE	NE	1	RGB

A-4. Puntuaciones otorgadas a los algoritmos del 1 a 10 en el elemento evaluación *Ev*.

Método		Evaluación *Ev* = $Ev_{NúmBD}$+ Ev_{BDCom} + $Ev_{NúmM}$ + Ev_{ECual} + Ev_{ECuan}+ Ev_{MPar}									
Ranking	Identificación	$Ev_{NúmBD}$+ Ev_{BDCom}	Nombre	Completa	Ev_{ECual}+ Ev_{ECuan}	Ev. Cual	Ev. Cuan	$Ev_{NúmM}$	Metricas	Ev_{MPar}	Mismos parametros p/t video
1	Multimode Background Subtraction(MBS)	4	Change Detection 2014, ESI	SI	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	1	SI
2	BGWiS o CWisarDH	4	SBI, Change Detection 2014	Si	2	✔	✔	2	AGE, pEPs, pCEPs, PSNR, MS-SSIM, CQM	0	No
3	AAPSA	4	Change Detection 2014, BMC	Si	2	✔	✔	2	Recall,Especificidad ,FPR, FNR PWC, Precisión, F-measure	1	Si
4	DeepBS (Supervised method)	6	Change Detection, Wallflower, PETS 2009	SI	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	1	SI
5	Cascade CNN(supervised method)	4	Change Detection 2014, SBI2015	SI	2	✔	✔	2	F-measure, PWC, FPED, FNED	1	Si
6	TEDV	6	I2R, Fish4Knowledge, BMC	Si	2	✔	✔	2	F-measure, Precisión, Recall, ROC, PSNR, D-Score, SSIM	1	Si
8	SuBSENSE	4	Change Detection 2012, Change Detection 2014	Si	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	1	Si
7	WeSamBE	2	Change Detection 2014	SI	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	1	Si
9	Multimode Background Subtraction Version 0 (MBS V0)	2	Change Detection 2014	SI	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	0	No
10	SharedModel	2	Change Detection 2014	Si	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	1	Si

A-5. Puntuaciones otorgadas a los algoritmos del 11 a 21 en el elemento evaluación *Ev*.

Método		Evaluación $Ev = Ev_{NúmBD} + Ev_{BDCom} + Ev_{NúmM} + Ev_{ECual} + Ev_{ECuan} + Ev_{MPar}$									
Ranking	Identificación	$Ev_{NúmBD} + Ev_{BDCom}$	Nombre	Completa	$Ev_{ECual} + Ev_{ECuan}$	Ev. Cual	Ev. Cuan	$Ev_{NúmM}$	Metricas	Ev_{MPar}	Mismos parametros p/t video
11	FTSG	2	Change Detection 2014	Si	2	✔	✔	2	Recall ,Especificidad, FPR, FNR PWC, Precisión, F-measure	1	Si
12	BREW-DLHPM	2	Change Detection 2014	Si	2	✔	✔	2	Recall Especificidad FPR, FNR PWC, Precisión F-measure	1	Si
13	STBM	4	Change Detection 2012, SABS	SI	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	0	NE
14	SBM	4	Change Detection 2012, SABS	SI	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	0	NE
15	PAWCS	2	Change Detection 2014	Si	2	✔	✔	2	Recall Especificidad FPR, FNR PWC, Precisión F-measure	1	Si
16	IUTIS-5	2	Change Detection 2014	Si	2	✔	✔	2	Recall ,Especificidad FPR, FNR, PWC, Precisión, F-measure	1	Si
17	IUTIS-3	2	Change Detection 2014	Si	2	✔	✔	2	Recall ,Especificidad FPR, FNR, PWC, Precisión, F-measure	1	Si
18	IUTIS-1	2	Change Detection 2014	Si	2	✔	✔	2	Recall ,Especificidad FPR, FNR, PWC, Precisión, F-measure	1	Si
19	NeRM	1	Change Detection 2014	No	2	✔	✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	0	NE
20	MDLS	2	PTIS, Change Detection 2014	No	2	✔	✔	1	F-measure	1	Si
21	COROLA	3	Change Detection 2014, I2R, Wallflower	No	2	✔	✔	2	Recall, Precision, F-measure	1	Si

A-6. Puntuaciones otorgadas a los algoritmos del 22 a 35 en el elemento evaluación *Ev*.

Método		Evaluación $Ev = Ev_{NúmBD} + Ev_{BDCom} + Ev_{NúmM} + Ev_{ECual} + Ev_{ECuan} + Ev_{MPar}$									
Ranking	Identificación	$Ev_{NúmBD} + Ev_{BDCom}$	Nombre	Completa	$Ev_{ECual} + Ev_{ECuan}$	Ev. Cual	Ev. Cuan	$Ev_{NúmM}$	Metricas	Ev_{MPar}	Mismos parametros p/t video
22	RESOM	3	I2R, Wallflower, PETS	No	2	✔	✔	2	Precisión, Recall, Similitud, F-measure	1	Si
23	BMTDL	2	SABS, Change Detection 2012	No	1		✔	2	Recall ,Especificidad, FPR, FNR, PWC, Precisión, F-measure	1	Si
24	TVRPCA	3	SABS Dataset, I2R Dataset, Change Detection 2014	No	2	✔	✔	2	Precisión, Recall, F-measure	0	No
25	SMSOM-BM	1	Change Detection 2012	No	2	✔	✔	2	Precisión, Recall, F-measure	1	Si
26	SOM-CNN	2	Limu, I2R	No	2	✔	✔	2	Precisión, Recall, Similitud, F-measure	1	Si
27	CTU	2	Change Detection 2012, BMC	No	2	✔	✔	2	Precisión, Recall, F-measure	1	Si
28	EAMV	2	Change Detection 2014, AVSS	No	2	✔	✔	2	Recall, Precision, F-measure	1	Si
29	Temporal saliency	2	BMC	SI	2	✔	✔	2	Recall ,Precision, F-measure, PSNR ,D-Score, SSIM	0	NE
30	IISC	2	PETS2001, OTCBVS	No	2	✔	✔	1	F-measure	1	Si
31	MKFC	3	CAVIAR, I2R, Wallflower	No	2	✔	✔	1	AVG, F-measure	0	No
32	FCDH	3	I2R Dataset, Wallflower, Change Detection 2012	No	2	✔	✔	2	MCC, PCC, ROC ,PR	0	NE
33	ERCI	1	ATON	No	2	✔	✔	2	Recall ,Especificidad, Exactitud, Precisión, F-measure ,MCC	0	No
34	SAG	1	BMC	No	2	✔	✔	2	Precisión, Recall, PSNR, DS, SSIM, F-measure	0	No
35	GMM & SURF combination	2	BMC	SI	1		✔	2	Recall ,Precision, F-measure, PSNR ,D-Score, SSIM	1	Si

A-7. Puntuaciones otorgadas a los algoritmos del 36 a 55 en el elemento evaluación *Ev*.

Método		Evaluación $Ev = Ev_{NúmBD} + Ev_{BDCom} + Ev_{NúmM} + Ev_{ECual} + Ev_{ECuan} + Ev_{MPar}$									
Ranking	Identificación	$Ev_{NúmBD} + Ev_{BDCom}$	Nombre	Completa	$Ev_{ECual} + Ev_{ECuan}$	Ev. Cual	Ev. Cuan	$Ev_{NúmM}$	Metricas	Ev_{MPar}	Mismos parametros p/t video
36	RBFMD	0	NE	NE	2	✔	✔	2	Precisión, Recall, Similitud, F-measure	1	Si
37	SGGMM and Uncertains	1	Change Detection 2012	No	2	✔	✔	2	Precisión, Recall, Similitud, F-measure	0	No
38	Variación ViBe	1	Change Detection 2012	No	2	✔	✔	1	Precisión , Recall	1	Si
39	SiftFlow	1	Wallflower	No	2	✔	✔	1	Recall, FNR	0	No
40	Statistical local difference pattern	2	BMC	SI	2	✔	✔	2	Precisión, Recall, F-measure	1	Si
41	One-class classification	2	BMC	SI	1		✔	2	Recall ,Precision, F-measure, PSNR ,D-Score, SSIM	0	NO
42	EVIEVV	0	NA	NA	1		✔	1	PCC, FPR	1	Si
43	ConvNets	1	Change Detection 2014	No	2	✔	✔	1	F-measure	0	No
44	PBAS-PID	2	Change Detection 2012	Si	1		✔	1	F-measure, PWC	1	SI
45	Normalized cut	1	Wallflower	No	2	✔	✔	1	Errores ,porcentaje de mejora	0	NE
46	OMHBP	2	PETS 2006, AVSS	No	1	✔		0		1	Si
47	AEMF	2	UMA	Si	2	✔	✔	1	F-measure	0	No
48	GMRF y HTNN	3	PETS2006, I2R, Wallflower	No	2	✔	✔	2	Precisión, Recall, F-measure	0	No
49	DMW	2	ACV, CAVIAR	No	1		✔	1	TPR, FPR	1	Si
50	MFESVV	0	NE	NE	2	✔	✔	2	Precisión, Recall, Similitud, F-measure	0	NE
51	Block-based RPCA	1	I2R	No	2	✔	✔	2	Recall, Precision, F-measure	1	Si
52	K-SVD	1	I2R	No	2	✔	✔	2	Recall, Precision, F-measure, PSNR	0	No
53	BC, or Bayesian classification	2	I2R	SI	2	✔	✔	1	FNE, FPE	1	Si
54	PS-RBM	1	Change Detection 2012	No	2	✔	✔	2	Precisión, Recall, F-measure	1	Si
55	BRTF	1	I2R	No	1	✔		0		1	Si

A-8. Puntuaciones otorgadas a los algoritmos del 56 a 67 en el elemento evaluación *Ev*.

Método		Evaluación $Ev = Ev_{NúmBD} + Ev_{BDCom} + Ev_{NúmM} + Ev_{ECual} + Ev_{ECuan} + Ev_{MPar}$									
Ranking	Identificación	$Ev_{NúmBD} + Ev_{BDCom}$	Nombre	Completa	$Ev_{ECual} + Ev_{ECuan}$	Ev. Cual	Ev. Cuan	$Ev_{NúmM}$	Metricas	Ev_{MPar}	Mismos parametros p/t video
56	AFPA	0	NA	NA	2	✔	✔	2	F-measure, Precisión, Recall, ROC	0	No
57	DMIEC	0	NE	NE	2	✔	✔	2	Precisión, Recall, Similitud, F-measure	1	Si
58	SGMM-SOD	3	i-LIDS, PETS2006, CAVIAR	No	1	✔		1	Ninguna	0	NO
59	DRCIMF	2	I2R Dataset, LIMU	No	1		✔	1	Similitud, F-measure	0	NE
60	GC STViBe	1	I2R	No	2	✔	✔	1	TP, FP	0	NE
61	ISURF	0	NA	NA	1	✔		0		0	NE
62	Robust low rank matrix decomposition	1	BMC	NO	2	✔	✔	2	Recall ,Precision, F-measure, PSNR	0	NE
63	FMR	0	NA	NA	2	✔	✔	1	FP, FN	0	No
64	GMM | KaewTraKulPong	0	NE	NE	1	✔		1	NE	1	SI
65	DTCNN	0	NE	NE	2	✔	✔	2	Precisión, Recall, Similitud, F-measure	0	No
66	GMM | Zivkovic	0	NE	NE	0			0		0	NE
67	VuMeter	0	NE	NE	0			0		0	NE

A-9. Puntuaciones otorgadas a los algoritmos del 1 a 12 en los elementos implementación *Im* y comparaciones *Co*, y puntuación final P_{DT}.

Método		Implementación Im= Im_{Sof} + Im_{CPU} + Im_{RAM}				Comparaciones Co= $Co_{NúmA}$		Puntuación P_{DT}
Ranking	Identificación	Im_{Sof} + Im_{CPU} + Im_{RAM}	Software	CPU	RAM	$Co_{NúmA}$	Algoritmos	Ordenados de mejor a menor desempeño
1	Multimode Background Subtraction(MBS)	3	MATLAB	Intel core i5 PC	8.0 GB	2	FTSG, SuBSENSE, CwisarDH, Spectral, AAPSA, BinWang, KNN, SC_SOBS, RMoG, KDE, SOBS_CF, Mahala, etc.	28.2
2	BGWiS o CWisarDH	3	C++	Intel Core i7 3.4GHz	8.0 GB	2	FTSG, SuBSENSE, Spectral, AAPSA, BinWang, KNN, SC_SOBS, RMoG, KDE, SOBS_CF, Mahala, etc.	28
3	AAPSA	2	MATLAB	Intel Xeon 2.4 GHz	NE	2	FTSG, SuBSENSE, CwisarDH, Spectral, AAPSA, BinWang, KNN, SC_SOBS, RMoG, KDE, SOBS_CF, Mahala, etc.	27.4
4	DeepBS (Supervised method)	0	NE	NE	NE	2	PBAS, PAWCS,SuBSENSE, MBS,GMM, RMoG, Spectral-360,ETC.	27
5	Cascade CNN(supervised method)	2	Matlab	GTX970 GPU	NE	2	FCN, IUTIS-5 PAWCS , SuB- SENSE	26.6
6	TEDV	3	C++	Intel i7 3.4 Ghz	16.0 GB	2	P-finder,GMM, ZGMM, EIGEN, ML-BKG, KDE-RGB, ViBe	26
8	SuBSENSE	2	C++	Intel i5 CPU at 3.3 GHz	NE	2	CwisarD, Spectral 360, DPGMM, SGMM-SOD, PBAS, PSP-MRF, SC-SOBS, ViBe+, kde, ViBe, GMM	25.6
7	WeSamBE	3	C++ (OpenCV)	Intel i5 3470 2.2 GHz	6.0 GB	2	FTSG, SharedModel, SuBSENSE, Spectral-360, AMBER, etc.	25.6
9	Multimode Background Subtraction Version 0 (MBS V0)	3	MATLAB	Intel Core i5	8.0 GB	2	FTSG, SuBSENESE, CwisarDH, Spectral-360, Bin Wang Apr 2014, SC_SOBS, etc	24.6
10	SharedModel	2	C++	Intel i7 3.4 Ghz	NE	2	GMM, Adaptative GMM, RMoG	23.6
11	FTSG	1	MATLAB	NE	NE	2	KNN, GMM, KDE, MahaD, GMM, EucD	22.6
12	BREW-DLHPM	2	NE	Intel Core i5 3.20 GHz	8.0 GB	2	Adaptative SOM, DECOLOR, EigenBkg, FTSG, GOSUS, GRASTA, MOG, MultiLayer, PBAS, BS-RDL, ViBe, etc.	22.6

A-10. Puntuaciones otorgadas a los algoritmos del 13 a 24 en los elementos implementación *Im* y comparaciones *Co*, y puntuación final P_{DT}.

Método		Implementación Im= Im_{Sof}+ Im_{CPU} + Im_{RAM}				Comparaciones Co= $Co_{NúmA}$		Puntuación P_{DT}
Ranking	Identificación	Im_{Sof}+ Im_{CPU} + Im_{RAM}	Software	CPU	RAM	$Co_{NúmA}$	Algoritmos	Ordenados de mejor a menor desempeño
13	STBM	2	NE	Intel Core i7 2.4Ghz	4.0 GB	2	McFarlane,Stauffer, Oliver, McKenna, Li, Kim, Zikovic, Maddalena, etc.	22
14	SBM	2	NE	Intel Core i7 2.4Ghz	4.0 GB	2	McFarlane,Stauffer, Oliver, McKenna, Li, Kim, Zikovic, Maddalena, etc.	22
15	PAWCS	1	C++	NE	NE	2	DPGMM, SGMM-SOD, PBAS, LOBSTER, PSPMRF, SCSOBS, ViBe+, SOBS, Cheby, KNN,etc	21.6
16	IUTIS-5	0	NE	NE	NE	2	IUTIS-3,PAWCS, SuBSENSE, FTSG, AAPSA, EFIC, AMBER, RMoG, KNN, Spectral 360, GraphCutDiff, SOBS_CF, M4CD, etc	20.6
17	IUTIS-3	0	NE	NE	NE	2	IUTIS-3,PAWCS, SuBSENSE, FTSG, AAPSA, EFIC, AMBER, RMoG, KNN, Spectral 360, GraphCutDiff, SOBS_CF, M4CD, etc	20.6
18	IUTIS-1	0	NE	NE	NE	2	IUTIS-3,PAWCS, SuBSENSE, FTSG, AAPSA, EFIC, AMBER, RMoG, KNN, Spectral 360, GraphCutDiff, SOBS_CF, M4CD, etc	20.6
19	NeRM	3	C++	Axis Q7436 (ARTPEC-5 chip-set) Encoder	NA	2	RGB, Chist, NeRM	20
20	MDLS	3	MATLAB	Intel core i5 processor 3.40 Ghz	4.0 GB	2	OR-PCA with MRF, COROLA, DECOLOR, LR-FSO, GOSUS, BS-GFLFM, RPCA,SemiSoftGoDec	19.8
21	COROLA	3	C++	Intel i7 3.4 GHz	16.0 GB	2	MOG, GRASTA, OPRMF, ORPCA, ORPCA+MRF	19.4
22	RESOM	2	MATLAB	Intel Xeon	NE	2	SSOM, CNN, SOBS, BNN, Li et al	18
23	BMTDL	1	NE	Intel 2.40GHz	NE	2	Stauffer, Oliver, McKenna, Li, Kim, Zivkovic, Maddalena, Barnich	18
24	TVRPCA	3	MATLAB con C++	Intel Core i7-2600 3.4 GHz	16.0 GB	2	RPCA, DECOLOR, TVRPCA, GMM, SOBS, GRASTA, SPDM, SAC	17

A-11. Puntuaciones otorgadas a los algoritmos del 25 a 41 en los elementos implementación *Im* y comparaciones *Co*, y puntuación final P_{DT}.

Método		Implementación Im= Im_{Sof}+ Im_{CPU} + Im_{RAM}				Comparaciones Co= $Co_{NúmA}$		Puntuación P_{DT}
Ranking	Identificación	Im_{Sof}+ Im_{CPU} + Im_{RAM}	Software	CPU	RAM	$Co_{NúmA}$	Algoritmos	Ordenados de mejor a menor desempeño
25	SMSOM-BM	2	NE	2.67GHz Intel CPU and a NVIDIA GeForce GTX 260 GPU,	6.0 GB	2	SOBS, KDE, GMM	16.8
26	SOM-CNN	3	MATLAB	Intel Xeon	4.0 GB	2	SSOM, CNN, SOBS, BNN, Li et al	16.2
27	CTU	3	C++	Intel Core i7-2600 3.40 GHz	16.0 GB	2	CBR, Reddy et al, SOBS, AFABS, DPGMM	16.2
28	EAMV	3	MATLAB	Intel Core i5-3210M CPU	4.0 GB	2	GMM, ViBe, Lam, Rounding foreground posterior	15.2
29	Temporal saliency	2	MATLAB 2012a	2.4GHz (Windows 7)	NE	0	Ninguno	14.8
30	IISC	3	C++	Intel i7 3.4 GHz	16.0 GB	2	Recursive MoG, Ensamble MoG, Phase-based, Kernel score, Kernel density, Low rank	14.4
31	MKFC	2	NE	Intel Core 2 Duo 2.53 GHz	2.0 GB	2	Improved MoG, CoodeBook, PSM, SAB, SOBS, LBPH	14.2
32	FCDH	3	MATLAB	Intel Core i5-2400 3.10 GHz	4.0 GB	2	GMM, LBP, STLBP, LIBS, FBS, FST, MCC,FCH, FCDH	14.2
33	ERCI	3	MATLAB	Intel(R) Core i5-2430 2.40 GHz	4.0 GB	2	AFM, IS, SOM, GMM,ViBe, ViBe-BMRI	14
34	SAG	2	C++ Matlab	Intel dual core 3.0 GHz Pentium	NE	2	TSFM, SLDP, OCBM, IIBM, RLRM, ZHG, SAM	13.8
35	GMM & SURF combination	1	NE	Intel Core 2 (2.26GHz Quad Core CPU)	NE	2	MoG, AMoG, MoG-SH, IIBM	13.8
36	RBFMD	2	NE	Intel Core2Quad 2.33GHz	2.0 GB	2	RBFMD, GMM, SOBS, CB, ViBe	13.2
37	SGGMM and Uncertains	2	NE	Intel Core i5-2430M 2.4 GHz	8.00 GB	2	SGGMM basado en color,SGGMM W.P.U , KDE, GMM Stauffer, DECOLOR, LOBSTER, Spectral 360	12.8
38	Variación ViBe	2	C++	Portatil i7 2.2 GHz	NE	2	ViBe, ViBe+, GMM-Z, GMM-KTP	12.8
39	SiftFlow	2	NE	Genuine Intel® 1.73 GHz	2.0 GB	2	ACMLI, GMM, CD, LBP, NPPDF, ViBe	11.8
40	Statistical local difference pattern	0	NE	NE	NE	1	GMM, RRF	11.8
41	One-class classification	2	NE	Intel Duo-core 2.4GHz	4.0 GB	0	Ninguno	11.8

A-12. Puntuaciones otorgadas a los algoritmos del 42 a 67 en los elementos implementación *Im* y comparaciones *Co*, y puntuación final P_{DT}.

Método		Implementación $Im = Im_{Sof} + Im_{CPU} + Im_{RAM}$				Comparaciones $Co = Co_{NúmA}$		Puntuación P_{DT}
Ranking	Identificación	$Im_{Sof} + Im_{CPU} + Im_{RAM}$	Software	CPU	RAM	$Co_{NúmA}$	Algoritmos	Ordenados de mejor a menor desempeño
42	EVIEVV	3	MATLAB	Intel Core 2 Duo 2.00 GHz	1.0 GB	2	MBMD, WTH1, WTH2, WTH3, WTH4	11.2
43	ConvNets	0	NE	NE	NE	2	ConvNet-GT, IUTIS-5, SuBSENSE, PAWCS, PSP-MRF, ConvNet-IUTIS, EFIC, Spectral 360, SC_SOBS, GMM, GraphCut, KDE	11.2
44	PBAS-PID	0	NE	NE	NE	0	Ninguno	11.2
45	Normalized cut	1	NE	Intel Core i7-4770 3.4 GHz	NE	2	GSM, GSM MRF, GSM MinCut, GSM Ncut	11
46	OMHBP	2	NE	Intel 3.07 GHz	4.0 GB	2	GMM, IGMM, ViBe	10.8
47	AEMF	1	MATLAB	NE	NE	2	ViBe, FVS, MoSeg	10.8
48	GMRF y HTNN	2	NE	Intel i5 CPU at 3.3 GHz	4.0 GB	1	Codebook, GMRF and GraphCut	10.6
49	DMW	2	MATLAB	Intel Core 2 2.16 GHz	NE	2	MFD, Proposed V1, Proposed V2, Proposed V3, Background Substraction, FD	10.2
50	MFESVV	2	NE	Intel Core i5 2.8 GHz	4.0 GB	2	FDM, BSM, ABSM	9.4
51	Block-based RPCA	0	NE	NE	NE	2	MG, FCI, KDE, SG	9.2
52	K-SVD	0	NE	NE	NE	2	ViBe, Adaptative MoG, MoG, TM	9
53	BC, or Bayesian classification	0	NE	NE	NE	1	MoG, RBS	9
54	PS-RBM	0	NE	NE	NE	2	DPGMM, KDE, KNN, SOBS	8.8
55	BRTF	1	MATLAB	NE	NE	2	PCP, GoDec, DRMF, RegL1ALM, VBRPCA, PRMF, BRMF, DECOLOR, BRTF	8.6
56	AFPA	1	Matlab	NE	NE	2	TAM, Adaptative TAM Var.A, Adaptative TAM Var.B, MoG	8.6
57	DMIEC	0	NE	NE	NE	1	SD, MDPS	8
58	SGMM-SOD	1	NE	Intel Core 2 3GHz	NE	1	System1, System2	8
59	DRCIMF	0	NE	NE	NE	2	MoG, KDE, ViBe, DBG	7.6
60	GC STViBe	0	NE	NE	NE	2	ACMLI, GMM, ViBe	7.4
61	ISURF	3	C++	Intel Core i5 M520 2.4GHz NVIDIA NVS 3100M	4.0 GB	1	SURF, SIFT	7.2
62	Robust low rank matrix decomposition	0	NE	NE	NE	0	Ninguno	6.8
63	FMR	1	MATLAB	NE	NE	0	Ninguno	6.2
64	GMM | KaewTraKulPong	0	NE	NE	NE	1	Grimson et al.	6.2
65	DTCNN	0	NE	NE	NE	0	Ninguno	6
66	GMM | Zivkovic	1	NE	2 GHz PC	NE	0	Ninguno	4.6
67	VuMeter	0	NE	NE	NE	0	Ninguno	1

APÉNDICE B. CRITERIO AUTO-ADAPTABILIDAD.

B-1. Puntuaciones otorgadas a los algoritmos del 1 a 36 en los elementos grado de inervación humana *GraIH* y complejidad *Complex*.

Método		Grado de intervención humana *GraIH*			Complejidad *Complex*
Ranking	Identificación	*GraIH*	Ajuste manual	Ajuste automático	*Complex*
1	DMIEC	15	Ninguno	th (x,y)	1.78
2	Block-based RPCA	15	Ninguno	Y0 , S0 , k ,μ, λ	1.44
3	AAPSA	15	ninguno	ThDCh,Thstable,ThsceneCh, ThSFo,ThLLR,ThHLR	1.33
4	RESOM	15	ninguno	αo, σs, σβ, α, β(m)z, Tf, ta	1.22
5	SOM-CNN	15	ninguno	a1,a2, αo, σs, σβ, α, β(m)z, Tf, ta	1.00
6	MFESVV	13.5	tr	Ninguno	1.90
7	GC STViBe	13.5	Ro	mHg	1.79
8	STBM	13.5	α	Vp	1.79
9	SBM	13.5	α	Vp	1.79
10	GMRF y HTNN	13.5	α	μc,μu,σ2c ,σ2u	1.46
11	EVIEVV	13.5	L	σth_buf, σth2	1.68
12	SGMM-SOD	12	k,T	NE	1.80
13	DMW	12	τ,L	th	1.69
14	FMR	12	FA,FM	r	1.69
15	VuMeter	12	α,k	Pi	1.69
16	WeSamBE	12	Ro_color(x),N	ci(x)	1.69
17	AFPA	12	C,β	T,α	1.58
18	ISURF	12	σ,N	number of feature points,Dx	1.58
19	SuBSENSE	12	αST,αLT	T,R,Dmin(x),v(x)	1.36
20	MDLS	12	k, N	V,U, σ , wij in	1.36
21	FTSG	12	Tb, Tf	Dmin,μt,σt,wit,M	1.24
22	SharedModel	12	α, N	B, F, μB, σB, μF, Σf	1.13
23	PAWCS	12	Ro color,Ro desc	T(x),R(x),W(x),Dmin(x),Rcolor(x),Rdesc(x)	1.13
24	GMM & SURF combination	12	T, N	W, μkI , μkC, SigmaKI ,SigmaKC, α, λ	1.02
25	BRTF	12	X, n	a(n),V(n),Cm,Dm, am,bm,λ, γ ,τ	0.80
26	Cascade CNN(supervised method)	12	# cuadros para entrenar, # epocas	NE	1.80
27	PS-RBM	12	α,β	bi ,ci,W	1.47
28	Robust low rank matrix decomposition	12	α, β	Ck, C'k, Bk	1.47
29	DeepBS (Supervised method)	12	learning rate, epochs	Fs, bm, α	1.47
30	SMSOM-BM	12	αtrain,αtest	ωh(l),ωs(l),ωv(l),τ	1.36
31	Multimode Background Subtraction Version 0 (MBS V0)	12	t, N	CPDN, CPFDN, CFDN, CiFDN	1.36
32	TEDV	10.5	w,α,γ	NE	1.70
33	DRCIMF	10.5	TR, TL , αL	βd	1.59
34	DTCNN	10.5	Lm,th1,zij	L	1.59
35	IISC	10.5	α,T,Block size	fB(x, y)	1.48
36	PBAS-PID	10.5	Kp, Ki, Kd	T(xi), R(xi)	1.48

B-2. Puntuaciones otorgadas a los algoritmos del 37 a 67 en los elementos grado de inervación humana *GraIH* y complejidad *Complex*.

Método		Grado de intervención humana *GraIH*			Complejidad *Complex*
Ranking	Identificación	*GraIH*	Ajuste manual	Ajuste automático	*Complex*
37	GMM | Zivkovic	10.5	cthr, cf, T	Pim, μm, Sigmam	1.37
38	GMM | KaewTraKulPong	10.5	α, T, L	Wk, μk,SigmaK	1.37
39	SGGMM and Uncertains	10.5	Tseg,Tcmap,Tseg pixel and uncertain	Sigmak,μk,βk,wk	1.26
40	TVRPCA	10.5	λ1(dinamico/normal) ,λ2, λ3	Fk+1,Ek+1, Mk+1,Bk+1,γ t+1	1.14
41	OMHBP	9	T_teta, TD, Th,Tc	Ninguno	1.60
42	One-class classification	9	th, m, n, b	αi	1.49
43	BC, or Bayesian classification	9	T, σ1, delta, n	σ2	1.49
44	BREW-DLHPM	9	Kmax,Jmax, α,β	c,λ,zj+1,Φ,x	1.04
45	BGWiS o CWisarDH	9	β,φ,Ro,n	NE	1.60
46	ConvNets	9	bias,learning rate, pesos, #neuronas	NE	1.60
47	ERCI	9	nB,nBmax,α,t	Mcij(kij)	1.49
48	K-SVD	9	dictionary size,patch size, # dictionary training, number of descomposition coefficientes	α avg	1.49
49	IUTIS-5	9	Population size ,w1,w2, max number of generations	Max tree depth,Crossver rate, Mutation rate	1.27
50	IUTIS-3	9	Population size ,w1,w2, max number of generations	Max tree depth,Crossver rate, Mutation rate	1.27
51	IUTIS-1	9	Population size ,w1,w2, max number of generations	Max tree depth,Crossver rate, Mutation rate	1.27
52	CTU	9	CTUs ,tY,tC,YCbCr	α, μidx,Eidx	1.27
53	BMTDL	9	λ, patch, K,overlapping	atoms,Up,Dp,K	1.16
54	SiftFlow	7.5	TP,αb,αw,k, Tb	ωk,mk	1.28
55	Statistical local difference pattern	7.5	N,TB ,α ,T, r	Wjk, μjk ,Sigmajk	1.17
56	AEMF	7.5	ε , τ , F , λ1, λ2	Cback,x(n)	1.28
57	EAMV	6	γ,ϕ,σI,τCV,τTV,τBV	Ninguno	1.40
58	FCDH	6	σ,α, M,R,th,W	Ninguno	1.40
59	Variación ViBe	6	cMin, R, N, φ, β, κ	Ninguno	1.40
60	RBFMD	6	β, E, S ,α, w0 ,η	Ninguno	1.40
61	COROLA	4.5	α,m,n,β1,β2,r,γ	U,A,B,v	0.86
62	Multimode Background Subtraction(MBS)	4.5	corr_th, prob_th, M, colorthreshold, channel_th, th, m	U, μn	1.08
63	SAG	1.5	ht,r0,number of Gaussians,cT,γ1,cf,dc,γ2,α hs	σm,wm,bm,μm,cm	0.54
64	MKFC	0	l, R, r, d, k, σ, α, β, γ1, γ2	Ninguno	1.00
65	Temporal saliency	0	NE	Ninguno	2.00
66	Normalized cut	0	NE	NE	0.00
67	NeRM	0	NE	NE	0.00

B-3. Puntuaciones otorgadas a los algoritmos del 1 a 35 en el elemento entrenamiento *Entr* y puntuación total P_{AT}.

Método		Entrenamiento *Entr*		Puntuación P_{AT}
Ranking	Identificación	*Entr*	¿Requiere cuadros entrenamiento?	Ordenados de mejor a menor desempeño
1	DMIEC	1	No	17.778
2	Block-based RPCA	1	No	17.444
3	AAPSA	1	No	17.333
4	RESOM	1	No	17.222
5	SOM-CNN	1	No	17.000
6	MFESVV	1	No	16.400
7	GC STViBe	1	No	16.289
8	STBM	1	No	16.289
9	SBM	1	No	16.289
10	GMRF y HTNN	1	No	15.956
11	EVIEVV	0	Si	15.178
12	SGMM-SOD	1	No	14.800
13	DMW	1	No	14.689
14	FMR	1	No	14.689
15	VuMeter	1	No	14.689
16	WeSamBE	1	No	14.689
17	AFPA	1	No	14.578
18	ISURF	1	No	14.578
19	SuBSENSE	1	No	14.356
20	MDLS	1	No	14.356
21	FTSG	1	No	14.244
22	SharedModel	1	No	14.133
23	PAWCS	1	No	14.133
24	GMM & SURF combination	1	No	14.022
25	BRTF	1	No	13.800
26	Cascade CNN(supervised method)	0	Si	13.800
27	PS-RBM	0	Si	13.467
28	Robust low rank matrix decomposition	0	SI	13.467
29	DeepBS (Supervised method)	0	Si	13.467
30	SMSOM-BM	0	Si	13.356
31	Multimode Background Subtraction Version 0 (MBS V0)	0	Si	13.356
32	TEDV	1	No	13.200
33	DRCIMF	1	No	13.089
34	DTCNN	1	No	13.089
35	IISC	1	No	12.978

B-4. Puntuaciones otorgadas a los algoritmos del 36 a 67 en el elemento entrenamiento *Entr* y puntuación total P_{AT}.

Método		Entrenamiento *Entr*		Puntuación P_{AT}
Ranking	Identificación	*Entr*	¿Requiere cuadros entrenamiento?	Ordenados de mejor a menor desempeño
36	PBAS-PID	1	No	12.978
37	GMM | Zivkovic	1	No	12.867
38	GMM | KaewTraKulPong	1	No	12.867
39	SGGMM and Uncertains	1	No	12.756
40	TVRPCA	1	No	12.644
41	OMHBP	1	No	11.600
42	One-class classification	1	No	11.489
43	BC, or Bayesian classification	1	No	11.489
44	BREW-DLHPM	1	No	11.044
45	BGWiS o CWisarDH	0	Si	10.600
46	ConvNets	0	Si	10.600
47	ERCI	0	Si	10.489
48	K-SVD	0	Si	10.489
49	IUTIS-5	0	Si	10.267
50	IUTIS-3	0	Si	10.267
51	IUTIS-1	0	Si	10.267
52	CTU	0	Si	10.267
53	BMTDL	0	Si	10.156
54	SiftFlow	1	No	9.778
55	Statistical local difference pattern	1	No	9.667
56	AEMF	0	Si	8.778
57	EAMV	1	No	8.400
58	FCDH	1	No	8.400
59	Variación ViBe	1	No	8.400
60	RBFMD	0	Si	7.400
61	COROLA	1	No	6.356
62	Multimode Background Subtraction(MBS)	0	Si	5.578
63	SAG	0	Si	2.044
64	MKFC	1	No	2.000
65	Temporal saliency	0	Si	2.000
66	Normalized cut	1	No	1.000
67	NeRM	0	Si	0.000