

INSTITUTO TECNOLÓGICO DE CHIHUAHUA
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

***“ALGORITMOS PARA EL SEGUIMIENTO DE
MÚLTIPLES OBJETOS EN SECUENCIAS DE
VIDEO”***

TESIS

QUE PARA OBTENER EL GRADO DE

***MAESTRO EN CIENCIAS
EN INGENIERÍA ELECTRÓNICA***

PRESENTA:

ING. FRANCISCO JAVIER ÁLVAREZ PRIETO

**DIRECTOR DE LA TESIS:
DR. MARIO IGNACIO CHACÓN MURGUÍA**



SEP
SECRETARÍA DE
EDUCACIÓN PÚBLICA



TECNOLÓGICO NACIONAL DE MÉXICO



CHIHUAHUA, CHIH., ENERO 2021.



Chihuahua, Chih. 21 de enero de 2021

**FRANCISCO JAVIER ÁLVAREZ PRIETO
PRESENTE**

En cumplimiento con los requerimientos para la obtención del grado de Maestro en Ciencias en Ingeniería Electrónica y a propuesta de su Comité Tutorial, la División de Estudios de Posgrado e Investigación le concede la autorización para imprimir la tesis titulada *"Algoritmos para el seguimiento de múltiples objetos en secuencias de video"* dirigida por el Dr. Mario Ignacio Chacón Murguía con el siguiente contenido de capítulos:

- I Antecedentes
- II Modelo de apariencia (MA)
- III Modelo de movimiento (MM)
- IV Asociación de datos
- V Seguimiento de múltiples objetos
- VI Conclusiones y trabajo futuro

ATENTAMENTE

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

MTRO. ROGELIO ENRIQUE BARAY ARANA
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN





Chihuahua, Chih. 15 de enero de 2021

MTRO. ROGELIO ENRIQUE BARAY ARANA
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
PRESENTE

En cumplimiento con los requerimientos para la obtención del grado de Maestro en Ciencias en Ingeniería Electrónica, le notificamos que el documento de tesis del alumno FRANCISCO JAVIER ÁLVAREZ PRIETO, titulado *"Algoritmos para el seguimiento de múltiples objetos en secuencias de video"* dirigido por el Dr. Mario Ignacio Chacón Murguía ha sido aprobado y aceptado para su impresión.

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

Agradeciendo la atención a la presente, quedamos de usted:

ATENTAMENTE

Excelencia en Educación Tecnológica
"La Técnica por el Engrandecimiento de México"

DR. MARIO IGNACIO CHACÓN MURGUÍA
DIRECTOR DE TESIS



Juan Ramirez

DR. JUAN ALBERTO RAMÍREZ QUINTANA
MIEMBRO DEL JURADO DE EXAMEN

DR. JAVIER VEGA PINEDA
MIEMBRO DEL JURADO DE EXAMEN

DR. LUIS FRANCISCO CORRAL MARTÍNEZ
MIEMBRO DEL JURADO DE EXAMEN



CARTA CESIÓN DE DERECHOS

En la ciudad de Chihuahua el día 26 de enero de 2021, el que suscribe FRANCISCO JAVIER ÁLVAREZ PRIETO con número de control G19060583, de la Maestría en Ciencias en Ingeniería Electrónica, adscrita a la División de Estudios de Posgrado e Investigación del Instituto Tecnológico de Chihuahua, manifiesta que es autor intelectual del presente trabajo de Tesis bajo la dirección del Dr. Mario Ignacio Chacón Murguía y cede los derechos del trabajo titulado "Algoritmos para el seguimiento de múltiples objetos en secuencias de video", al Tecnológico Nacional de México y/o Instituto Tecnológico de Chihuahua para su difusión, divulgación, transmisión, reproducción, así como su digitalización con fines académicos y de investigación.

FRANCISCO JAVIER ÁLVAREZ PRIETO



DECLARACIÓN DE ORIGINALIDAD

En la ciudad de Chihuahua el día 26 de enero de 2021, el que suscribe FRANCISCO JAVIER ÁLVAREZ PRIETO de la Maestría en Ciencias en Ingeniería Electrónica, con número de control G19060583, adscrito a la División de Estudios de Posgrado e Investigación del Instituto Tecnológico de Chihuahua, manifiesta que es autor intelectual de la tesis titulada "Algoritmos para el seguimiento de múltiples objetos en secuencias de video" bajo la dirección el Dr. Mario Ignacio Chacón Murguía; que el contenido es original y que las fuentes de información consultadas para su fundamentación están debidamente citadas y referenciadas.

FRANCISCO JAVIER ÁLVAREZ PRIETO

Chihuahua, Chihuahua, a 17 de diciembre de 2020

Dra. María Elena Álvarez-Buylla Roces

Directora del CONACYT

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

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

A t e n t a m e n t e



Francisco Javier Álvarez Prieto
Exbecario CONACYT

RESUMEN

ALGORITMOS PARA EL SEGUIMIENTO DE MÚLTIPLES OBJETOS EN SECUENCIAS DE VIDEO

Ing. Francisco Javier Álvarez Prieto

Maestría en Ciencias en Ingeniería Electrónica

División de Estudios de Posgrado e Investigación del

Instituto Tecnológico de Chihuahua

Chihuahua, Chih. 2020

Director de Tesis: Dr. Mario I. Chacón Murguía

Al proceso de estimar a lo largo del tiempo la localización de uno o varios objetos de interés, se le conoce como seguimiento de múltiples objetos y su importancia radica en que es una característica fundamental que permite a las máquinas lograr ver, entender e interactuar con su entorno. Además, el seguimiento de múltiples objetos es un factor clave para el desarrollo de diversas aplicaciones como: sistemas de video vigilancia, manejo autónomo, análisis inteligente del comportamiento humano, entre algunas otras. Debido a su gran relevancia, en este proyecto se presenta el desarrollo de tres algoritmos para el seguimiento de múltiples objetos en secuencias de video, nombrados T-AM, T-MM y T-AMM. Los tres algoritmos están basados en el enfoque del *tracking-by-detection*, la diferencia entre los algoritmos está en la información que extraen de los objetos para realizar el seguimiento. Dicha información puede ser patrones de apariencia utilizando una red siamesa (T-AM), patrones de movimiento utilizando un filtro de Kalman (T-MM) o una combinación de ambos (T-AMM). Los tres algoritmos propuestos presentan un desempeño similar en las principales métricas de seguimiento de objetos, el cual es competitivo en comparación con los métodos publicados en la literatura reciente.

CONTENIDO

CAPÍTULO 1 ANTECEDENTES	1
1.1 Visión por computadora.....	1
1.2 Seguimiento de múltiples objetos	2
1.2.1 Objetos de interés	2
1.3 Dificultades en el seguimiento de objetos	2
1.3.1 Dificultades en el seguimiento de múltiples objetos.....	5
1.4 Etapas en el seguimiento de múltiples objetos	6
1.5 Trabajos relacionados con el seguimiento de múltiples objetos.....	8
1.5.1 Métodos <i>Online</i>	8
1.5.2 Métodos <i>Offline</i>	10
1.6 Métricas de evaluación	11
1.7 Aplicaciones.....	13
CAPÍTULO 2 MODELO DE APARIENCIA (MA).....	15
2.1 Revisión de las principales características utilizadas en los modelos de apariencia	15
2.2 Red neuronal convolucional (CNN)	17
2.2.1 Capa de convolución	18
2.2.2 Capa de agrupación	20
2.2.3 Capa totalmente conectada (FC)	21
2.2.4 Hiperparámetros en una CNN	21
2.3 Análisis de CNN propuestas en la literatura	22
2.3.1 Selección de los modelos de CNN	23
2.3.2 Escenarios de prueba.....	24
2.3.3 Extracción de características	25
2.3.4 Función de similitud.....	26
2.3.5 Grado de discriminación G_d , MA actualizado.....	28
2.3.6 Grado de discriminación G_d , modelo no actualizado	31
2.3.7 Comparación entre modelos de CNN seleccionados	33

2.4 Propuesta del MA basado en una CNN en configuración siamesa.....	37
2.4.1 Diseño de la arquitectura del MA propuesto.....	38
2.4.2 Determinación del número de épocas de entrenamiento del MA propuesto.....	42
2.4.3 Comparación de SiamAM con VGG-16.....	43
CAPÍTULO 3 MODELO DE MOVIMIENTO (MM)	46
3.1 Revisión de los principales métodos utilizados en los modelos de movimiento	46
3.2 Filtro de Kalman	49
3.2.1 Algoritmo del Filtro de Kalman.....	50
3.3 El Filtro de Kalman enfocado al seguimiento de múltiples objetos	52
3.4 Configuración del Filtro de Kalman	54
CAPÍTULO 4 ASOCIACIÓN DE DATOS	56
4.1 Problema de asociación.....	56
4.1.1 Seguimiento de múltiples objetos como un problema de asociación.....	57
4.2 Método Húngaro	57
CAPÍTULO 5 SEGUIMIENTO DE MULTIPLES OBJETOS.....	64
5.1 Base de datos.....	64
5.2 Detector de objetos	66
5.2.1 Supresión no máxima modificada.....	68
5.3 Algoritmo para el seguimiento de múltiples objetos	71
5.3.1 Estructura general del algoritmo	71
5.3.2 Detalles de la implementación	74
5.4 Algoritmo T-AM.....	75
5.4.1 Modelado de objetos en el algoritmo T-AM.....	75
5.4.2 Evaluación del algoritmo T-AM	76
5.5 Algoritmo T-MM.....	76
5.5.1 Modelado de objetos en el algoritmo T-MM	77
5.5.2 Evaluación del algoritmo T-MM	77
5.6 Algoritmo T-AMM	78
5.6.1 Modelado de objetos en el algoritmo T-AMM	78
5.6.2 Evaluación del algoritmo T-AMM.....	79

5.7 Comparación entre los algoritmos T-AM, T-MM y T-AMM	80
5.8 Comparación del algoritmo T-AMM con la literatura.....	82
CAPÍTULO 6 CONCLUSIONES Y TRABAJO FUTURO	85

LISTA DE FIGURAS

Figura 1.1. Modelo de visión por computadora [2].	1
Figura 1.2. Ejemplos de objetivos de interés: (a) Personas; (b) Rostros [5].	2
Figura 1.3. Desafíos del seguimiento de objetos: (a) Oclusión; (b) Cambios de apariencia; (c) Fondos abarrotados; (d) Cambios de tamaño; (e) Cambios de iluminación; (f) Ruido en la imagen; (g) Objetos similares; (h) Movimientos complejos [18].	3
Figura 1.4. Pasos básicos para el seguimiento de objetos.	6
Figura 1.5. Áreas en donde se utiliza el seguimiento de múltiples objetos.	13
Figura 2.1. Porcentaje de trabajos en el que son utilizadas los diferentes tipos de características.	17
Figura 2.2. Ejemplo de una arquitectura de CNN.	18
Figura 2.3. Ejemplo de la generación de mapas de características en la l -ésima capa.	19
Figura 2.4. Ejemplo de la operación de <i>max-pooling</i> y <i>average pooling</i> .	20
Figura 2.5 Relación entre la exactitud y el tiempo de predicción de distintas CNN propuestas en la literatura.	23
Figura 2.6. Ejemplos de los objetivos que conforman los 6 escenarios de pruebas.	25
Figura 2.7. Procedimiento para obtener la medida de similitud entre los objetivos O_i^j y O_k^l .	26
Figura 2.8: Medida de similitud utilizando ρ , dE y los descriptores extraídos de la CNN Alexnet (a), (b) e Inception-ResNet-v2 (c), (d).	27
Figura 2.9. Procedimiento para obtener el grado de discriminación de cada CNN ante un determinado desafío.	29
Figura 2.10. Gd para las distintas CNN utilizando ρ (a) y dE (b), en un MA actualizado.	30
Figura 2.11. Gd para las distintas CNN utilizando ρ (a) y dE (b), en un MA no actualizado.	32
Figura 2.12. Conjunto de imágenes de la clase <i>plátano</i> , <i>naranja</i> y <i>persona</i> .	34
Figura 2.13. Promedio de similitud entre descriptores de las clases de prueba utilizando ρ (a) y dE (b).	35
Figura 2.14. MA propuesto basado en una CNN en configuración siamesa.	38
Figura 2.15. Evolución del Gd de los modelos ante diferente número de épocas de entrenamiento.	42

Figura 2.16. Comparación entre el Gd de SiamAM y la CNN VGG-16 utilizando ρ , en un MA actualizado (a) y un MA no actualizado (b).	44
Figura 3.1. Porcentaje en que sea a utilizado cada método con base en los trabajos consultados.	48
Figura 3.2. El ciclo del filtro de Kalman.	50
Figura 3.3. Visión completa del FK.	52
Figura 3.4. Predicción de la posición de un objetivo en un instante futuro.....	53
Figura 3.5. Comparación entre las predicciones del FK y la trayectoria real del objetivo, utilizando el centroide del objetivo.	53
Figura 4.1. Problema de asociación entre orígenes y destinos (a), solución que maximiza el beneficio (b).....	56
Figura 4.2. Problema de asociación entre detecciones en los cuadros t y $t+1$ (a), solución óptima (b).	57
Figura 4.3. Esquema general del funcionamiento del método Húngaro.	58
Figura 4.4. Generación de la matriz de costo C_{ij}	58
Figura 4.5. Preprocesamiento de la matriz C_{ij}	59
Figura 4.6. Proceso para obtener la matriz reducida.	60
Figura 4.7. Algoritmo de asociación.	61
Figura 4.8. Resultado del algoritmo de asociación para obtener la asociación inicial (a), interpretación de la asociación inicial (b).....	61
Figura 4.9. Pasos para generar la nueva matriz de asociación.....	62
Figura 4.10. Resultado del algoritmo de asociación (a), interpretación de la nueva asociación (b).....	62
Figura 4.11. Problema de asociación entre detecciones considerando errores en el detector de objetos (a), solución óptima (b).	63
Figura 5.1. Muestras de las secuencias de entrenamiento de la base de datos MOT16 y MOT17.	65
Figura 5.2. Muestras de las secuencias de prueba de la base de datos MOT16 y MOT17.	66
Figura 5.3. Ejemplo de detecciones proporcionadas por un detector en dos cuadros de video consecutivos.....	67

Figura 5.4. Ejemplo de detecciones redundantes alineadas (a) y no alineadas (b).....	69
Figura 5.5. Detecciones antes (a) y después (b) de aplicar el algoritmo NMS modificado.	70
Figura 5.6. Estructura general del algoritmo para el seguimiento de múltiples objetos.....	71
Figura 5.7. Ejemplo de la creación de los candidatos (C) a partir de las detecciones (D) para generar las trayectorias de los objetos (O).....	73
Figura 5.8. Generación de la matriz de costo en el algoritmo T-AM para objetos (a) y candidatos (b).	75
Figura 5.9. Velocidad relativa de procesamiento del algoritmo T-AM.....	76
Figura 5.10. Generación de la matriz de costo en el algoritmo T-MM para objetos (a) y candidatos (b).	77
Figura 5.11. Velocidad relativa de procesamiento del algoritmo T-MM.....	78
Figura 5.12. Generación de la matriz de costo en el algoritmo T-MM para objetos.....	79
Figura 5.13. Velocidad relativa de procesamiento del algoritmo T-AMM.	80
Figura 5.14.Comparación entre algoritmos utilizando las métricas MOTA, MOTP e IDF1 en MOT17.	81
Figura 5.15. Comparación entre algoritmos utilizando las métricas MOTA, MOTP e IDF1 en MOT16.	81
Figura 5.16. Comparación entre algoritmos utilizando la velocidad relativa de procesamiento.	81

LISTA DE TABLAS

Tabla 1.1. Principales métricas dentro del seguimiento de múltiples objetos.	11
Tabla 2.1. Revisión de las características más utilizadas para el desarrollo de modelos de apariencia.	16
Tabla 2.2. Gd para las distintas CNN utilizando ρ en un MA actualizado.	29
Tabla 2.3. Gd para las distintas CNN utilizando dE en un MA actualizado.	30
Tabla 2.4. Gd para las distintas CNN utilizando ρ en un MA no actualizado.	32
Tabla 2.5. Gd para las distintas CNN utilizando dE en un MA no actualizado.	32
Tabla 2.6. Similitud entre los descriptores del conjunto de imágenes de prueba utilizando ρ y la CNN Alexnet.	34
Tabla 2.7. Promedio de similitud entre descriptores de las clases de prueba utilizando ρ	36
Tabla 2.8. Promedio de similitud entre descriptores de las clases de prueba utilizando dE	36
Tabla 2.9. Resultados del Gd para determinar el tipo de normalización que se utiliza en el MA propuesto.	39
Tabla 2.10. Resultados del Gd promedio para determinar el tipo de normalización que se aplicará a la imagen de entrada en el MA propuesto.	40
Tabla 2.11. Resultados del Gd promedio para determinar los hiperparámetros del MA propuesto.	40
Tabla 2.12 Arquitectura del MA propuesto basado en una CNN en configuración siamesa. ...	41
Tabla 2.13. Gd de los 10 modelos entrenados con 1000 épocas.	43
Tabla 3.1. Revisión de los métodos más utilizados para el desarrollo de modelos de movimiento.	47
Tabla 3.2. Comparativa de los MM basados en filtros de Kalman y RNN (LSTM).	49
Tabla 3.3. Configuración del FK enfocado al seguimiento de múltiples objetos.	54
Tabla 5.1. Descripción de las secuencias de video de la base de datos MOT16 y MOT17 para entrenamiento y prueba.	65
Tabla 5.2. Detectores públicos proporcionados en MOT17.	66
Tabla 5.3. Comparación entre detectores públicos proporcionados por la base de datos MOT17.	67

Tabla 5.4. Evaluación del algoritmo T-AM en la base de datos MOT16 y MOT17.....	76
Tabla 5.5. Evaluación del algoritmo T-MM en la base de datos MOT16 y MOT17.	78
Tabla 5.6. Evaluación del algoritmo T-AMM en la base de datos MOT16 y MOT17.	80
Tabla 5.7. Comparación del algoritmo T-AMM con la literatura en la base de datos MOT17 (Resultados del 17/noviembre/2020).....	83

GLOSARIO DE TÉRMINOS

Acrónimo	Descripción
A	Matriz de transición de estado para el filtro de Kalman
AS	Apariencias similares
AT	Área de traslape entre un objetivo y su respectivo <i>ground-truth</i>
b	Factor del <i>bias</i>
C	Candidatos
C_{det}	Confidencialidad asignada por el detector de objetos a cada detección
Cs	Matriz de costo para el método Húngaro
CA	Cambios de apariencia
CI	Cambios de iluminación
CT	Cambios de tamaño
CNN	Red neuronal convolucional
Cnf	Valor de confidencialidad de objetos y candidatos
D	Detecciones
Dc	Detecciones que no se asignan con ningún objeto
DF	Vector de características profundas
dE	Distancia Euclidiana
DNN	Red neuronal profunda
DPM	<i>Deformable Part Model</i>
FK	Filtro de Kalman
FM	Mapa de características
$f(\cdot)$	Función de activación no lineal
Fn	Número de falsos negativos
Fp	Número de falsos positivos
FPS	<i>Frame per Second</i>
$FR-CNN$	<i>Faster R-Convolutional neural network</i>
Gd	Grado de discriminación
GT	Número de <i>ground-truth</i>
H	Matriz de medición para el filtro de Kalman
HOG	Histograma de gradientes orientados
h	Altura del mapa de características
I	Matriz identidad
$IDF1$	Métrica de cambios de identidad

Acrónimo	Descripción
ID_{sw}	Número de cambios de identidad
K	Ganancia de Kalman
MA	Modelo de apariencia
ML	Métrica de seguimiento (<i>Mostly Lost</i>)
MM	Modelo de movimiento
MOT	<i>Multiple Object Tracking</i>
MT	Métrica de seguimiento (<i>Mostly Tracked</i>)
$MOTA$	Métrica de exactitud (<i>Multi Object Tracking Accuracy</i>)
$MOTP$	Métrica de precisión (<i>Multi Object Tracking Precision</i>)
Ms	Medida de similitud
$MsOI$	Medida de similitud del objetivo de interés consigo mismo
$MsOP$	Medida de similitud del objetivo de interés con otros objetos
NMS	<i>Non maximum suppression</i>
nR	Número total de relaciones
O	Objetos
OCL	Oclusiones
P	Matriz de la covarianza del error para el filtro de Kalman
Q	Matriz de la covarianza del ruido del proceso para el filtro de Kalman
R	Matriz de la covarianza del ruido de la medición para el filtro de Kalman
$ReLU$	Función de unidad lineal rectificada
RI	Ruido en la imagen
$RU_{Di,Dj}$	Intersección sobre la unión entre detecciones
$RM_{Di,Dj}$	Intersección sobre area minima entre detecciones
SDP	<i>Scale-Dependent Pooling</i>
Se	Margen de discriminación
$SiamAM$	Modelo de apariencia propuesto (<i>Siamese Appearance Model</i>)
$T-AM$	<i>Tracker - Appearance Model</i>
$T-AMM$	<i>Tracker - Appearance Motion Model</i>
$T-MM$	<i>Tracker - Motion Model</i>
Th_{Af}	Umbral de afinidad
W	Matriz de pesos de una capa totalmente conectada
w	Filtros de una capa convolucional
X	Vector de estado para el filtro de Kalman
Z	Vector de medición para el filtro de Kalman
As	Función de similitud
ρ	Coefficiente de correlación cruzada
ω	Ancho del mapa de características

INTRODUCCIÓN

El seguimiento de múltiples objetos se puede definir como el proceso de estimar la posición de uno o más objetos de interés mientras se mueven alrededor de una escena. Este proceso realizado de manera automática propicia el desarrollo de novedosas aplicaciones como: sistemas de video vigilancia, manejo autónomo, análisis deportivos, análisis inteligente del comportamiento humano, entre otras. Sin embargo, el seguimiento de múltiples objetos aun en la actualidad es una tarea muy desafiante, debido a situaciones críticas que ocurren durante este proceso, tales como: oclusiones, cambios de apariencia, fondos abarrotados, cambios de tamaño, ruido en la imagen, cambios de iluminación, objetos similares, movimientos complejos, entre otras.

En esta tesis se presenta el desarrollo de tres algoritmos para el seguimiento de múltiples objetos, para su implementación en el análisis de secuencias de video. Con el objetivo de que puedan ser utilizados para el desarrollo de nuevas aplicaciones en futuras investigaciones dentro del Instituto Tecnológico de Chihuahua. Los tres algoritmos desarrollados en esta tesis están basados en el enfoque del *trackig-by-detection*, que consiste en la incorporación de un algoritmo para la detección de objetos en cada cuadro de la secuencia de video. El seguimiento de objetos en los tres algoritmos se logra mediante la asociación de dichas detecciones en cuadros de video subsecuentes, utilizando el método Húngaro. La diferencia entre los algoritmos está en la información que extraen de los objetos para realizar el seguimiento. Dicha información puede ser patrones de apariencia utilizando una red siamesa, patrones de movimiento utilizando un filtro de Kalman o una combinación de ambos.

El contenido de esta tesis está dividido en 6 capítulos. En donde en el capítulo 1 se presentan los antecedentes y conceptos relacionados con el seguimiento de múltiples objetos. En lo referente al modelo de apariencia y el modelo de movimiento desarrollados en esta tesis, estos se describen en los capítulos 2 y 3 respectivamente. Por su parte, en el capítulo 4 se muestra el algoritmo utilizado para la asociación de datos. Posteriormente, en el capítulo 5 se detallan los algoritmos propuestos para el seguimiento de múltiples objetos, mientras que las conclusiones y áreas de oportunidad para trabajos futuros se muestran en el capítulo 6.

CAPÍTULO 1

ANTECEDENTES

El objetivo principal de esta tesis es presentar un algoritmo enfocado en el seguimiento de múltiples objetos, para su implementación en el análisis de secuencias de video. Para llevar a cabo este proyecto será necesario la aplicación e integración de técnicas y conocimientos específicos sobre algunas áreas de la ciencia y la ingeniería, por lo cual, en esta sección se definirá una serie de conceptos relacionados con el tema de interés.

1.1 Visión por computadora

El propósito de la visión por computadora consiste en desarrollar métodos y técnicas basadas en algoritmos programables que permitan a una computadora reconocer elementos del entorno real a través de imágenes y extraer información de éstas para su posterior estudio, análisis de comportamiento, clasificación o reconstrucción de situaciones. Para poder cumplir con su propósito, un modelo de visión por computadora debe definir los niveles y procesos que se realizan sobre una escena para llegar a su interpretación [1]. Dichos niveles se pueden dividir en tres tipos: nivel bajo, nivel intermedio y nivel alto como se muestra en la Figura 1.1.

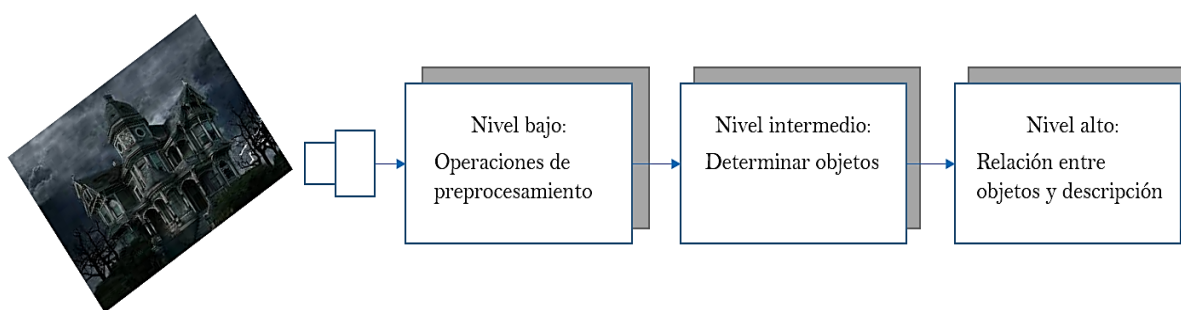


Figura 1.1. Modelo de visión por computadora [2].

Dentro del nivel bajo se encuentran acciones sobre la imagen correspondientes a suavizado, umbralización, eliminación de ruido, definición de bordes, análisis de textura, etc. En el nivel intermedio en cambio se tienen acciones como definición de límites, regiones, superficies, que están relacionadas a generar objetos presentes en la imagen. Por último, en el nivel alto se entablarán relaciones entre los objetos para realizar la interpretación de la escena [2].

1.2 Seguimiento de múltiples objetos

Una característica fundamental que permite a las máquinas lograr ver, entender e interactuar con su entorno es su capacidad para detectar y seguir objetos de interés.

El seguimiento de múltiples objetos o MOT por sus siglas en inglés se define como el proceso de estimar a lo largo del tiempo la localización de uno o varios objetos de interés, manteniendo de manera consistente sus identidades para posteriormente generar sus respectivas trayectorias individuales [3]–[5].

1.2.1 Objetos de interés

La definición de los objetos de interés también conocidos como objetivos depende específicamente de la aplicación, es decir, para el desarrollo de aplicaciones de video vigilancia los objetivos podrían ser personas o vehículos, Figura 1.2 (a), mientras que para el desarrollo de aplicaciones de entretenimiento interactivo los objetivos se convertirían solo en las manos, la cara o alguna otra parte en específico del cuerpo, Figura 1.2 (b).

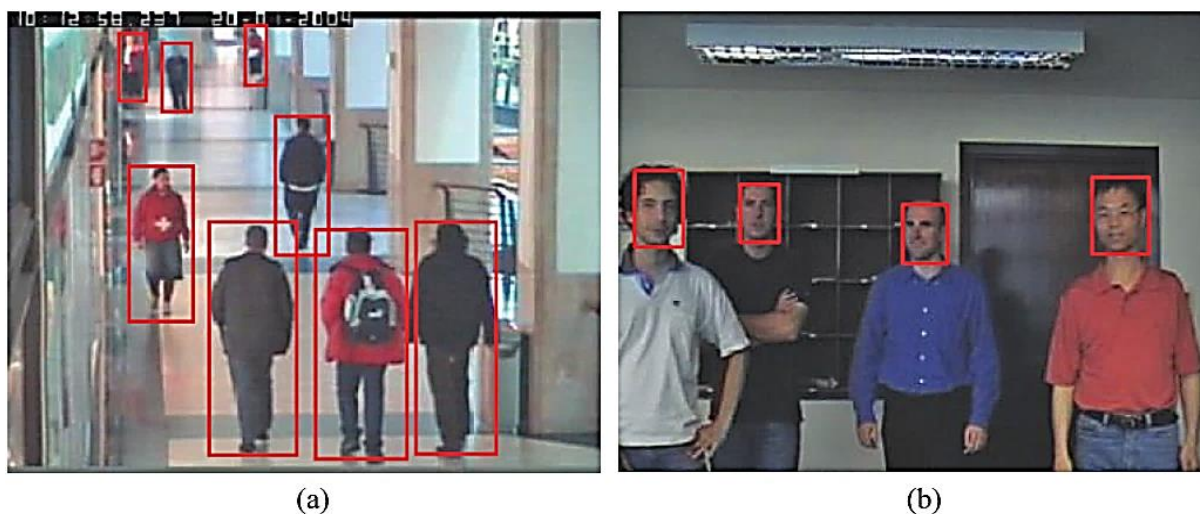


Figura 1.2. Ejemplos de objetivos de interés: (a) Personas; (b) Rostros [5].

1.3 Dificultades en el seguimiento de objetos

Investigaciones realizadas en las últimas 4 décadas han aportado grandes logros en el campo del seguimiento de objetos. Sin embargo, dicha tarea continúa siendo un gran desafío

tecnológico aun en la actualidad [6]–[12], esto debido a situaciones críticas que ocurren durante este proceso, tales como: oclusiones, cambios de apariencia, fondos abarrotados, cambios de tamaño, ruido en la imagen, cambios de iluminación, objetos similares, movimientos complejos, entre otros [8], [13]–[18]. Los desafíos antes mencionados se ilustran en la Figura 1.3 y serán definidos a continuación.

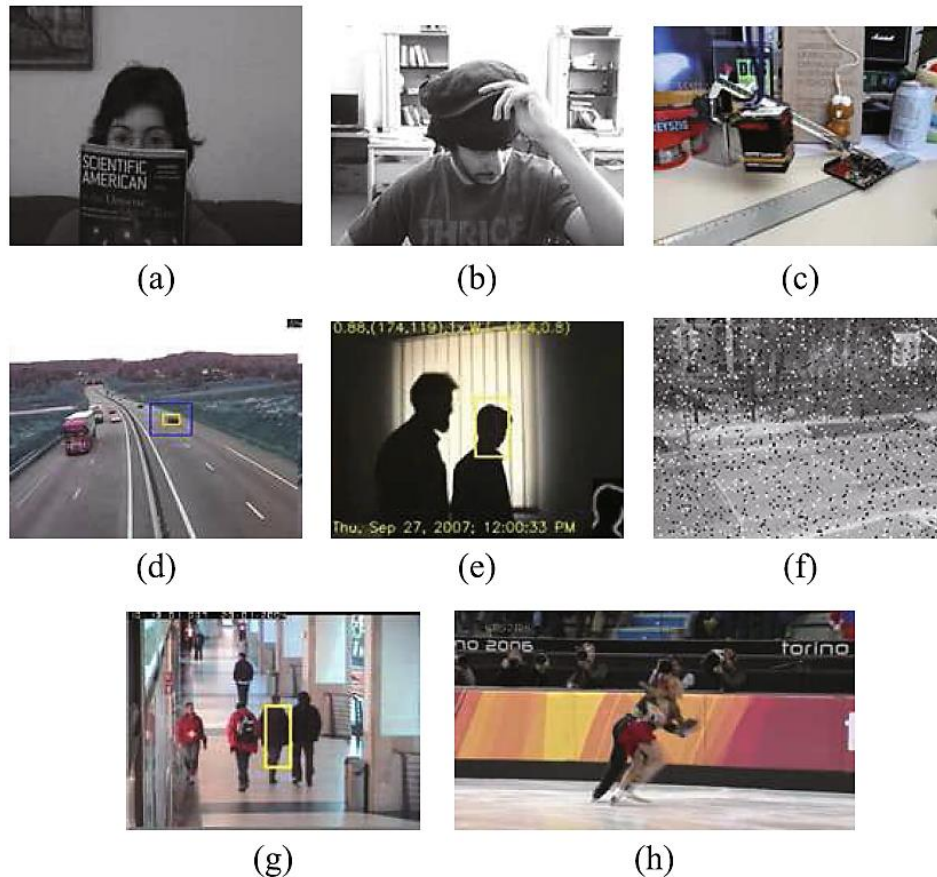


Figura 1.3. Desafíos del seguimiento de objetos: (a) Oclusión; (b) Cambios de apariencia; (c) Fondos abarrotados; (d) Cambios de tamaño; (e) Cambios de iluminación; (f) Ruido en la imagen; (g) Objetos similares; (h) Movimientos complejos [18].

Oclusión: Es el estado en que el objeto de interés es cubierto parcial o totalmente por otro objeto en la escena [18], [5]. Una oclusión se presenta frecuentemente debido a que:

- El objetivo se mueve detrás de un objeto estático, como una columna, una pared o un escritorio.
- Otro objeto en movimiento se interpone entre el objetivo.

La oclusión es un problema común durante el seguimiento de objetos, para el cual no existe una técnica universal para tratarlo. Por lo tanto, diversas estrategias se han propuesto de acuerdo con la naturaleza del objetivo y del entorno en el que se desenvuelve.

Cambios de apariencia: Muchos de los objetivos, especialmente los objetos no rígidos, cambian su apariencia al estar en movimiento [18], [5]. Lo que hace que surja la necesidad de que el modelo del objetivo se adapte a dichos cambios si el seguimiento se va a realizar durante un largo periodo. Sin embargo, debido a la actualización frecuente de dicho modelo, pequeñas imprecisiones se pueden agregar a éste, las cuales, se acumulan y con el paso del tiempo pueden provocar un seguimiento inestable. Si por el contrario se opta por mantener un modelo fijo, es decir sin actualizaciones, esto podría ocasionar que el seguidor pierda por completo el objetivo, en consecuencia, es necesario que exista un balance en las actualizaciones de dicho modelo.

Fondos abarrotados: Es el fenómeno que ocurre cuando una escena contiene demasiados objetos que no son de interés [5]. Dicha situación puede ser fácilmente solucionada si se trabaja en un entorno conocido o en escenarios estáticos. No obstante, si dicha situación se presenta cuando el entorno es dinámico y desconocido la solución a dicho problema se complica considerablemente.

Cambios de tamaño: Este efecto se produce cuando el objeto de interés se dirige hacia la cámara o se aleja de ella, provocando que su tamaño en la imagen se incremente o se reduzca respectivamente [18].

Cambios de iluminación: La dirección, intensidad y color de la luz del entorno influye directamente en la apariencia de los objetivos [5]. Por esta razón los cambios en la iluminación son un problema muy importante. Esto debido principalmente a que las características prominentes que se pueden extraer del objetivo con una buena iluminación se tornan inutilizables cuando la iluminación es muy baja, ya que afecta la forma en que las cámaras observan a los objetos.

Ruido en la imagen: El ruido cuando se trata de imágenes, se puede considerar como aquella información no deseada, que obstaculiza el manejo de la información útil presente en dicha imagen [2]. Este fenómeno puede presentarse a causa de distintos factores, siendo el más

frecuente el ruido que se introduce durante el proceso de adquisición y que depende directamente del sensor que se utilice [5]. El efecto más dañino que produce el ruido es la distorsión de la imagen, afectando así el rendimiento del seguidor. Afortunadamente la mayoría de las formas en que se puede presentar dicho fenómeno se pueden solventar.

Objetos similares: Ocurre cuando existen objetos con una apariencia muy similar a la del objeto de interés y se torna difícil diferenciar y seguir a dicho objetivo.

Movimientos complejos: Se presenta cuando el objetivo de interés realiza movimientos muy complicados, es decir, cambios muy abruptos en su velocidad o dirección. Complicando la tarea del seguimiento a causa de la inexactitud que genera en la aproximación del modelado del objetivo [18].

1.3.1 Dificultades en el seguimiento de múltiples objetos

Es posible encontrar aplicaciones en las que no es suficiente seguir a un solo objeto de interés, sino que es necesario realizar esta misma tarea con dos o más objetivos. Cuando esta situación ocurre, es inevitable considerar una serie de aspectos adicionales, complicando aún más dicha tarea [9], [19].

El seguimiento simple y el seguimiento de múltiples objetos se diferencian principalmente debido a que en este último, además de los desafíos mencionados en la sección 1.3, se agregan las siguientes dos dificultades [20].

1. No existe un número fijo de objetivos, es decir los objetos de interés continuamente pueden aparecer y desaparecer de la escena.
2. La identidad de los objetivos debe ser consistente mientras el objetivo permanezca en la escena.

Dentro de estos dos desafíos, los cambios de identidad son el problema más frecuente durante el seguimiento de múltiples objetos [21]-[22] ya que este problema se produce por la combinación de los desafíos del seguimiento simple como: oclusiones, apariencias similares, fondos abarrotados, entre otros.

1.4 Etapas en el seguimiento de múltiples objetos

El seguimiento de objetos involucra una serie de pasos básicos como lo son: detección, modelado y seguimiento de los objetos de interés [23], como se muestra en la Figura 1.4.

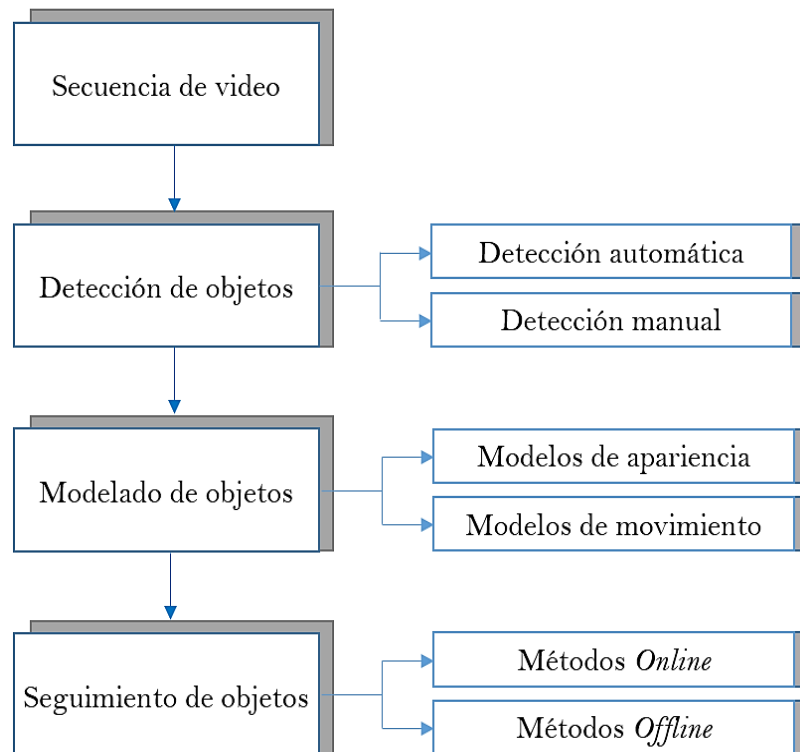


Figura 1.4. Pasos básicos para el seguimiento de objetos.

Secuencias de video: Se refiere al conjunto de imágenes digitales que se suministra al sistema para su análisis.

Detección de objetos: Es el paso inicial en el proceso de analizar un video, se encarga de identificar los objetos de interés y de agrupar el conjunto de píxeles que los conforman [24].

Esta etapa suele ser un criterio para clasificar los métodos para el seguimiento de múltiples objetos, ya que es posible encontrar métodos bajo los siguientes dos enfoques.

1. Detección automática: A este enfoque se le conoce como *Tracking-by-detection* y consiste en utilizar un algoritmo de detección de objetos en cada cuadro de la secuencia de video y posteriormente asociar las detecciones obtenidas en cuadros

consecutivos. Su principal desventaja es que los algoritmos para la detección de objetos son susceptibles a cometer errores que afectan en gran medida el desempeño del seguidor de objetos. Por otra parte, este enfoque es indispensable para aplicaciones que operan en tiempo real y aquellas en donde los objetivos aparecen y desaparecen de la escena de manera aleatoria.

2. Detección manual: Los métodos que siguen este enfoque reciben la información de los objetivos mediante detecciones realizadas manualmente durante los primeros cuadros de la secuencia de video, evitando los errores que generan los algoritmos de detección. Sin embargo, tienen la desventaja de prescindir de una inicialización manual.

Modelado de objetos: Una vez que un nuevo objeto ha sido detectado, el siguiente paso es crear un modelo del objetivo que permita distinguirlo fácilmente del fondo y los demás objetos presentes en la escena, para realizar esta tarea frecuentemente son utilizados modelos de apariencia y movimiento.

1. Modelos de apariencia: Son aquellos que mediante la extracción de características visuales de los objetivos modelan la proyección de su aspecto, para posteriormente calcular la similitud entre las predicciones y las observaciones con el fin de definir si se trata del mismo objetivo o no.
2. Modelos de movimiento: Se basan en el comportamiento dinámico de un objeto para predecir su posición en cuadros futuros. Generalmente se asume que los objetos se desplazan con una velocidad constante, aunque hay excepciones en donde se modelan desplazamientos no lineales.

Seguimiento de objetos: Como paso final se genera la trayectoria de cada objetivo delimitando su posición en cada cuadro de la secuencia de video manteniendo su identidad de manera consistente [25]. En esta etapa los métodos para realizar el seguimiento de objetos se clasifican por medio de un segundo criterio basado en la forma en que la información es procesada, en donde es posible encontrar métodos *online* y *offline*, cuya definición se proporcionará en la sección 1.5.

1.5 Trabajos relacionados con el seguimiento de múltiples objetos

El seguimiento de múltiples objetos es un área de investigación muy activa que cuenta con un amplio contenido en la literatura [26]. En la última década se han propuesto diversos métodos con enfoques totalmente diferentes, que se adaptan según la aplicación en la que se esté trabajando [3]. Por lo que resulta realmente complicado cubrir todos los diferentes enfoques que existen [26], por tal motivo en esta sección solo se mencionarán trabajos de relevancia realizados en un contexto similar al de esta tesis, dando preferencia a aquellos trabajos con fecha de publicación entre los años 2015 - 2020.

Los trabajos presentados se clasifican en métodos *online* y *offline*, ya que esta clasificación suele ser más justa, debido a que los métodos *online* únicamente se basan en la información obtenida en el cuadro actual y cuadros pasados, entregando resultados de manera secuencial, mientras que los métodos *offline* utilizan todo el contenido de la secuencia de video, es decir consideran información de cuadros futuros para entregar resultados. Esta forma en que los métodos *offline* procesan la información les da una amplia ventaja provocando una notoria diferencia en su desempeño.

1.5.1 Métodos Online

Tanto en los métodos *online* como *offline*, el enfoque más popular es el *tracking-by-detection* debido a los grandes avances que se han realizado en los métodos de detección de objetos. Sin embargo, estos detectores aun suelen cometer errores como falsas alarmas o pérdida de objetos de interés a causa de oclusiones o escenas muy concurridas. Por tal motivo, Henschel *et al* [27] incorporan un segundo detector entrenado para localizar articulaciones del cuerpo humano, lo que permite recuperar personas perdidas por el detector principal. No obstante, en escenarios muy concurridos la detección de articulaciones se dificulta considerablemente, por lo que en un segundo trabajo Henschel *et al* [28] proponen un detector entrenado para localizar la cabeza de las personas, esto debido a que suele mantenerse con mayor grado de visibilidad que otras partes del cuerpo, para obtener las trayectorias formula el seguimiento como un problema de grafo ponderado, en donde resuelve un problema de relación binaria cuadrática utilizando el algoritmo Frank-Wolfe.

Chen *et al* [29] incorporan un método para la predicción de la posición de los objetos que en combinación con un detector de objetos ayuda a disminuir la pérdida de objetivos.

De manera contradictoria a los métodos anteriores, Bergmann *et al* [30] consideran que un solo detector de objetos cuenta con la información suficiente para generar trayectorias en una gran cantidad de escenarios desafiantes, es decir que no es necesario incorporar un algoritmo especial para realizar el seguimiento de objetos. Por lo que propone un nuevo paradigma llamado “Tractor” en donde únicamente se vale de la regresión de un detector de objetos para realinear temporalmente los cuadros delimitadores de cada detección.

Por otra parte, en los métodos *online* es muy común el uso de análisis de movimiento y el aprendizaje de apariencia utilizando una asociación de datos jerárquica y modelos Bayesianos.

Dentro de los modelos Bayesianos los más utilizados son los filtros de partículas [31]–[42] y los filtros de Kalman en su versión simplificada [43]–[46] y extendida [47]–[49].

Debido a los grandes avances en el área del aprendizaje profundo los modelos de apariencia han sido muy utilizado recientemente, como se muestra en los siguientes trabajos.

Hong *et al* [50] combinan un red neuronal convolucional (CNN) preentrenada con una máquina de vectores de soporte (SVM) para aprender la apariencia de los objetivos mediante el cálculo de algún discriminante en los mapas de características. Zhu *et al* [51] entrenaron una CNN con atención espacial capaz de relacionar patrones de determinadas secciones en una imagen de entrada.

Yoon y Bae [52] de manera similar que Kieritz *et al* [53] proponen un método de aprendizaje de apariencia profundo para generar un modelo de apariencia discriminativo de cada objeto de manera secuencial. Kim *et al* [54] proponen un nuevo tipo de red *Bilinear Long Short-Term Memory* (LSTM) para el aprendizaje de modelos de apariencia de forma secuencial.

Sadeghian *et al* [55] utilizan tres redes neuronales recurrentes (RNN) para calcular apariencia, movimiento e interacción social con base en similitud. Fang *et al* [22] utilizan técnicas de reidentificación de personas e información de localización en una red neuronal recurrente autorregresiva (RAN).

1.5.2 Métodos *Offline*

En los métodos offline es muy común el uso de estrategias empleadas en los métodos online, pero el hecho de poder utilizar información de cuadros futuros les brinda la ventaja de poder convertir el seguimiento de objetos en un problema de optimización global en donde los enfoques más populares son: *network flow optimization* [11], *graph based clustering* [12] y *multiple hypothesis tracking* [56].

Wang *et al* [57] proponen combinar información temporal e información de apariencia de manera conjunta, definiendo un modelo de grafo en donde cada fragmento de trayectoria es considerado como un vértice del grafo, mientras que las trayectorias son generadas por la asociación de detecciones en cuadros consecutivos con ayuda de mediciones de similitud en la apariencia y la consistencia espacial.

Sheng *et al* [12] combinan un detector con características de bajo nivel (flujo óptico y color) para la construcción de árboles de trayectorias que pueden ser resueltos bajo el enfoque de MHT. De manera similar Keuper *et al* [58] proponen un enfoque unificado, solo que esta vez combinando el agrupamiento de píxeles (bajo nivel) con la detección y seguimiento de objetos semántico (alto nivel). En donde obtienen que las trayectorias de puntos ayudan a la agrupación de cuadros delimitadores en la detección del mismo objeto a través de oclusiones parciales, mientras que los cuadros delimitadores permiten el agrupamiento de las trayectorias de puntos en la presencia de movimientos articulados. Así mismo, demuestran que la combinación de métodos de bajo y alto nivel es altamente deseable y que sus ventajas se complementan en la práctica. Por su parte Chen *et al* [59] proponen lidiar con el problema de asociación de detecciones centrándose en el diseño de un modelo de apariencia de información histórica, utilizando una CNN multitarea capaz de discriminar características y relación espacio-temporal de manera conjunta.

Chu *et al* [60] proponen un modelo *end-to-end*, en donde la extracción de características, la medición de afinidad y la asignación multidimensional son realizadas en una sola red profunda nombrada “FAMNet”, la cual es optimizada en conjunto para generar las trayectorias de los objetivos. Leal *et al* [61] proponen una red CNN siamesa para relacionar detecciones de objetos

entre cuadros adyacentes, utilizando la salida de la red siamesa como medida de similitud para la asociación de datos.

1.6 Métricas de evaluación

En la literatura se han propuesto una gran cantidad de métricas para evaluar de manera cuantitativa el desempeño de un seguidor de objetos y facilitar su comparación con diferentes métodos [62]-[63]. En la Tabla 1.1 se enlistan algunas de las principales métricas agrupándolas en cuatro categorías: exactitud, precisión, robustez y complementarias, en donde el símbolo “↑” indica mejor desempeño entre más alto es el valor, mientras que el símbolo “↓” indica un mejor desempeño ante valores más bajos.

Tabla 1.1. Principales métricas dentro del seguimiento de múltiples objetos.

Métrica		Descripción	
Exactitud	MOTA <i>Multi Object Tracking Accuracy</i>	Combinación de falsos positivos, falsos negativos y cambios de identidad	↑
	IDsw <i>Identity switches</i>	Número de ocasiones en que un mismo objetivo es asociado a un identificador diferente	↓
Precisión	MOTP <i>Multi Object Tracking Precision</i>	Traslape entre la posición estimada y el <i>ground-truth</i>	↑
	TDE <i>Tracking Distance Error</i>	Distancia entre el <i>ground-truth</i> y el resultado del seguimiento	↓
	OSPA <i>Optimal Subpattern assignment</i>	Error de cardinalidad y distancia espacial entre <i>ground-truth</i> y el resultado del seguimiento	↓
Robustez	RS <i>Recover Short-term Occlusion</i>	Porcentaje de trayectorias correctas ante oclusiones moderadas	↑
	RL <i>Recover Long-term Occlusion</i>	Porcentaje de trayectorias correctas ante oclusiones prolongadas	↑
Complementarias	MT <i>Mostly Tracked</i>	Número de trayectorias que concuerdan con el <i>ground-truth</i> en al menos el 80% de su duración	↑
	ML <i>Mostly Lost</i>	Número de trayectorias que concuerdan con el <i>ground-truth</i> en menos del 20% de su duración	↓
	FM <i>Fragmentation</i>	Número de ocasiones en que el resultado del seguimiento es interrumpido	↓

Seleccionar las métricas adecuadas para evaluar un seguidor de objetos depende en gran medida de la aplicación para la cual fue diseñado. Sin embargo, con el fin de realizar una comparación más directa entre seguidores de objetos, es que se han propuesto una serie de estándares que resumen el desempeño de un seguidor de objetos mediante la selección estratégica de algunas métricas. El estándar CLEAR MOT Metrics [64], es uno de los estándares más aceptados en la literatura y se basa en las métricas MOTP y MOTA.

MOTP: Es el promedio de disimilitud entre verdaderos positivos y su correspondiente *ground-truth*. Esta métrica muestra la habilidad del seguidor para estimar precisamente la posición de los objetos sin tomar en cuenta su habilidad para reconocer dichos objetos, únicamente se centra en mantener la consistencia de las trayectorias, Ec (1.1).

$$MOTP = \frac{\sum_{t,i} AT_t^i}{\sum_t nR_t} \quad (1.1)$$

En donde nR_t es el número total de relaciones en el cuadro t , mientras que AT_t^i se refiere al área traslapada entre el cuadro delimitador del objeto i con su respectivo *ground-truth*.

MOTA: Es la métrica más utilizada en la literatura, debido a que combina tres tipos de errores como se muestra en la Ec. (1.2).

$$MOTA = 1 - \frac{\sum_t (Fn_t + Fp_t + IDsw_t)}{\sum_t GT_t} \quad (1.2)$$

En donde GT_t es el número de *ground-truth* en el cuadro t y Fn , Fp e $IDsw$ denotan el número de falsos negativos, falsos positivos y cambios de identidad respectivamente.

En la literatura reciente, el estándar CLEAR MOT Metrics se suele complementar con la métrica IDF1, Ec. (1.3). La cual, tiene como principal objetivo evaluar qué tan bien las identidades calculadas se ajustan a las identidades verdaderas de los objetos.

$$IDF1 = \frac{2IDTP}{2IDTP + IDFP + IDFN} \quad (1.3)$$

En donde IDTP representa las identidades correctas, mientras que IDFP e IDFN representan los falsos positivos y falsos negativos en las identidades de los objetos respectivamente.

1.7 Aplicaciones

El seguimiento de múltiples objetos recibe especial interés, debido a que es un elemento fundamental en el desarrollo de diversas aplicaciones en áreas como: videojuegos, monitoreo de tráfico, manejo autónomo, video vigilancia, diagnósticos médicos, reconocimiento de actividades, análisis de movimiento entre otras. En la Figura 1.5 se muestra una representación gráfica de las áreas en donde se utiliza el seguimiento de múltiples objetos y posteriormente se mencionan algunas aplicaciones dentro de las diferentes áreas.

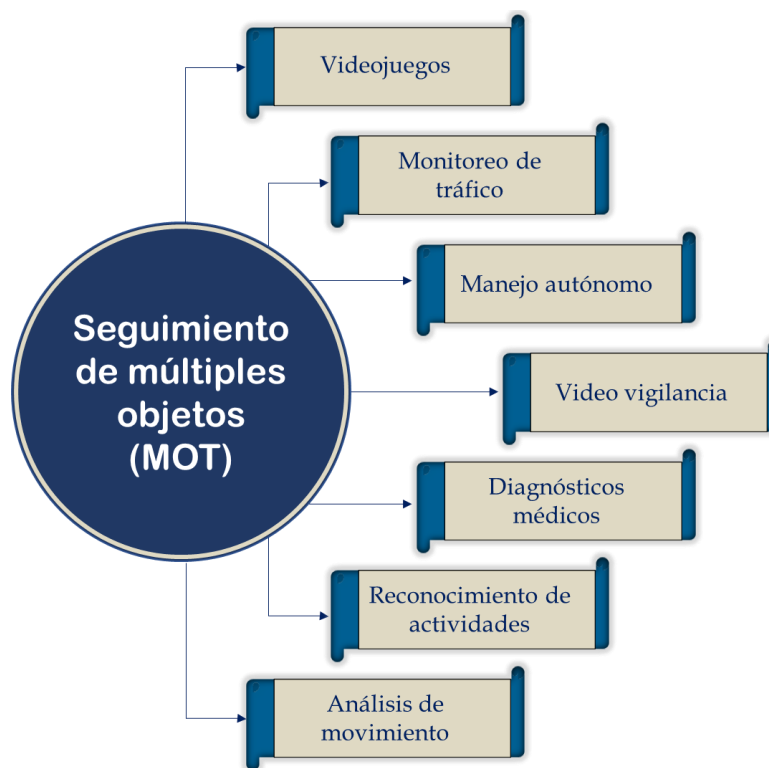


Figura 1.5. Áreas en donde se utiliza el seguimiento de múltiples objetos.

Videojuegos: MOT se utiliza como una alternativa en muchos videojuegos para que los usuarios tengan control de sus personajes mediante el movimiento de partes de su cuerpo, brindando una mejor interacción entre usuarios y videojuegos.

Monitoreo de tráfico: MOT es utilizado para el monitoreo y la administración de tráfico en las carreteras, mediante aplicaciones como detecciones de accidentes, embotellamientos y conteo de vehículos.

Manejo autónomo: MOT ayuda a los vehículos autónomos a rastrear elementos importantes como la posición de peatones, autos, animales o señalamientos de tránsito.

Video vigilancia: MOT es un elemento crucial en sistemas inteligentes de video vigilancia, en donde es necesario mantener la seguridad y defensa de lugares de interés, mediante la detección de intrusos o actividades sospechosas.

Diagnósticos médicos: MOT ha demostrado ser una herramienta de utilidad en el desarrollo de sistemas para diagnósticos médicos en aplicaciones como el seguimiento de la pared ventricular, reconstrucción de forma del tracto vocal y seguimiento de microrobots en operaciones de cateterización cardiaca.

Reconocimiento de actividades: MOT brinda la información suficiente para detectar que determinadas acciones se están llevando a cabo, permitiendo el desarrollo de aplicaciones como el reconocimiento de distracciones en los conductores y la estimación de la atención visual en interacciones sociales.

Análisis de movimiento: MOT es utilizado para presentar información de manera más compacta y simbólica de mayor utilidad para el usuario final, con aplicaciones como el análisis de eventos deportivos y análisis de mortalidad de organismos vivos.

En este capítulo se realizó una revisión de los trabajos relacionados con el tema de investigación de esta tesis. En donde se abordan conceptos básicos relacionados con el seguimiento de múltiples objetos, la metodología que se suele utilizar, los distintos enfoques empleados en la literatura recientemente, así como su impacto en el desarrollo de aplicaciones y solución de problemas. De esta forma se brinda una visión general del seguimiento de múltiples objetos y se proporcionan los elementos básicos para su comprensión.

CAPÍTULO 2

MODELO DE APARIENCIA (MA)

Los métodos de seguimiento de múltiples objetos que operan bajo el enfoque del *tracking-by-detection* generalmente tienen un desempeño que se ve influenciado principalmente por dos factores, un algoritmo de optimización para la asociación de datos y una función de similitud. La función de similitud es la encargada de medir la similitud entre los objetivos, mediante la comparación entre las predicciones realizadas por un modelo de movimiento y/o un modelo de apariencia (MA). Este último es de especial interés debido a que varios autores concuerdan en que es un paso crucial en el desarrollo de un algoritmo de seguimiento de objetos y que gran parte del desempeño del algoritmo depende del MA.

En este capítulo se da a conocer la metodología empleada para el desarrollo del MA utilizado en el algoritmo propuesto para el seguimiento de múltiples objetos.

2.1 Revisión de las principales características utilizadas en los modelos de apariencia

El desarrollo de un MA robusto que permita describir y adaptarse a los cambios de apariencia de los objetivos, ha sido de gran interés y ampliamente investigado durante los últimos años. Dentro de los modelos propuestos en la literatura la mayoría de éstos se basan en características como: forma del objetivo, histogramas de color, histogramas de gradientes orientados (HOG), características *Haar-like* y características profundas (DF), las cuales llevan ese nombre debido a que son extraídas de una red neuronal profunda (DNN), siendo las más utilizadas las redes neuronales convolucionales (CNN). Sin embargo, con el objetivo de conocer cuál de las características mencionadas anteriormente se utiliza con mayor frecuencia en la literatura, en esta sección se presenta una revisión que consta de 30 trabajos publicados recientemente, en donde se ha utilizado un MA. Dichos trabajos, el año en que se publicaron y el tipo de características que utilizan en el MA se muestran en la Tabla 2.1. A su vez, en la Figura 2.1 se muestra de manera resumida el porcentaje en que se utilizan los diferentes tipos de características en los trabajos consultados. En donde es posible apreciar una notoria diferencia en el uso de DF respecto a las demás características, siendo utilizada por el 77% de los trabajos

consultados. Esta preferencia se debe principalmente a su extraordinaria capacidad de representación de los objetivos. Por tal motivo y debido a la gran aceptación que reciben por parte de la comunidad científica, es que en esta tesis se decide desarrollar un MA utilizando DF. En las siguientes secciones se proporciona una descripción más detallada sobre las CNN y el procedimiento para extraer este tipo de características.

Tabla 2.1. Revisión de las características más utilizadas para el desarrollo de modelos de apariencia.

Nombre del artículo	características	Año	Ref.
Multiple Objects Tracking with Improved Sparse Representation and Rank Based Dynamic Estimation	Color (HSV)	2018	[65]
Interacting Tracklets for Multi-Object Tracking	Color (HSV)	2018	[19]
Multiple Object Tracking with Attention to Appearance, Structure, Motion and Size	HOG y color (HSV)	2019	[66]
Particle PHD Filter Based Multiple Human Tracking Using Online Group-Structured Dictionary Learning	HOG y color (RGB)	2018	[37]
Learning Support Correlation Filters for Visual Tracking	HOG y color	2019	[15]
On Detection, Data Association and Segmentation for Multi target Tracking	HOG y color	2019	[67]
Robust Structural Sparse Tracking	HOG y DF (VGG-Net)	2019	[8]
A Scale-Adaptive Particle Filter Tracking Algorithm Based on Offline Trained Multi-Domain Deep Network	DF (VGG-Net)	2020	[31]
Tracking the Untrackable: Learning to Track Multiple Cues with Long-Term Dependencies	DF (VGG-Net)	2017	[55]
Visual Tracking via Dynamic Memory Networks	DF (VGG-Net)	2019	[68]
Multi-Target Tracking Based on High-Order Appearance Feature Fusion	DF (VGG-Net)	2019	[69]
Multi-Agent Deep Reinforcement Learning for Multi Object Tracker	DF (VGG-Net)	2019	[70]
Deep Affinity Network for Multiple Object Tracking	DF (VGG-Net)	2019	[71]
Hedging Deep Features for Visual Tracking	DF (VGG-Net)	2019	[17]
Robust Visual Tracking via Hierarchical Convolutional Features	DF (VGG-Net)	2019	[14]
Fast Online Multi-Pedestrian Tracking via Integrating Motion Model and Deep Appearance Model	DF (VGG-Net)	2019	[72]
Learning Multi-task Correlation Particle Filters for Visual Tracking	DF (VGG-Net)	2019	[16]
Online Multi-Object Tracking with Dual Matching Attention Networks	DF (ResNet-50)	2018	[51]
Multi-target Tracking in Multiple Non-overlapping Cameras Using Fast-Constrained Dominant Sets	DF (ResNet-50)	2019	[26]

Tabla 2.1 Continuación.

Nombre del artículo	características	Año	Ref.
Multi-object Tracking with Neural Gating Using Bilinear LSTM	DF (ResNet-50)	2018	[54]
Spatial-Temporal Relation Networks for Multi-Object Tracking	DF (ResNet-50)	2019	[73]
Recurrent Autoregressive Networks for Online Multi-Object Tracking	DF (Inception Network)	2018	[22]
Heterogeneous Association Graph Fusion for Target Association in Multiple Object Tracking	DF (GoogLeNet)	2019	[12]
Exploit the Connectivity: Multi-Object Tracking with TrackletNet	DF (TrackletNet)	2019	[57]
Aggregate Tracklet Appearance Features for Multi-Object Tracking	DF	2019	[59]
Confidence-Based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-Object Tracking	DF	2018	[52]
FAMNet: Joint Learning of Feature, Affinity and Multi-dimensional Assignment for Online Multiple Object Tracking	DF	2019	[60]
Learning by tracking: Siamese CNN for robust target association	DF	2016	[61]
Online Tracking by Learning Discriminative Saliency Map with Convolutional Neural Network	DF	2015	[50]
Real-time Multiple People Tracking with Deeply Learned Candidate Selection and Person Re-identification	DF	2018	[29]

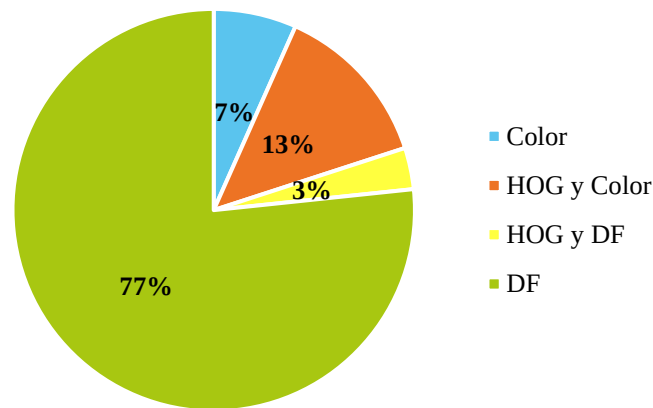


Figura 2.1. Porcentaje de trabajos en el que son utilizadas los diferentes tipos de características.

2.2 Red neuronal convolucional

Una CNN es un modelo del aprendizaje profundo, en donde es posible distinguir dos grandes etapas: una etapa de extracción de características y otra etapa de clasificación [74]. En la Figura 2.2 se muestra un esquema general de una CNN. Nótese que la señal de entrada corresponde a

una imagen y aunque las CNN no están restringidas a trabajar solo con imágenes este es el caso particular que se aborda en esta tesis.

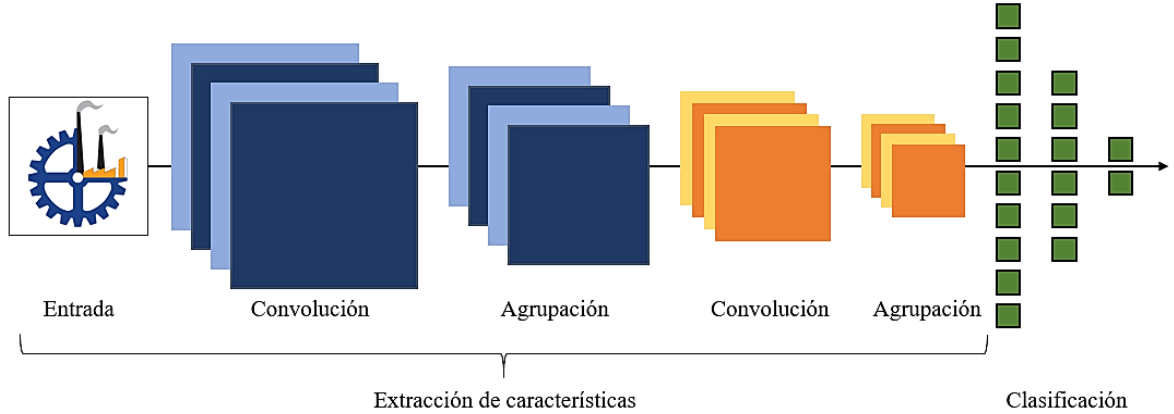


Figura 2.2. Ejemplo de una arquitectura de CNN.

La principal diferencia entre las dos etapas en una CNN reside en el tipo de capas que las conforman. Esto debido a que la etapa de extracción de características suele estar compuesta principalmente por capas convolucionales y de agrupamiento, mientras que la etapa de clasificación se caracteriza por tener al menos una capa totalmente conectada. En las siguientes secciones se describirá el funcionamiento y el objetivo de cada una de estas capas.

2.2.1 Capa de convolución

La capa de convolución es el núcleo y la principal distinción en una red neuronal convolucional, debido a que utiliza la operación de convolución en lugar de una matriz de multiplicación como se hace en las redes neuronales tradicionales. Con dicha modificación se reduce considerablemente el número de parámetros de la red y permite incrementar su profundidad [75].

En esta capa se tiene como objetivo detectar las características distintivas, que posteriormente serán mapeadas en mapas de características, FM . Considere FM_i^l como el i -ésimo mapa de características de tamaño $h^l \times \omega^l$ en la capa l , donde h^l y ω^l son la altura y el ancho del mapa de características respectivamente.

De forma analítica la salida de una capa convolucional se obtiene de la siguiente manera. Primero FM_i^{l-1} se convoluciona con diferentes filtros w_{ij}^l de tamaño $m^l \times m^l$ en la capa l , después

el resultado se evalúa por una función de activación no lineal $f(\cdot)$, siendo la función de unidad lineal rectificada (ReLU) la más utilizada. De esta manera se obtiene un j -ésimo mapa de características FM_j^l de tamaño $(h^{l-1} - m^l + 1) \times (\omega^{l-1} - m^l + 1)$ dado por Ec. 2.1.

$$FM_j^l = f\left(\sum_{i=1}^{k^l} FM_i^{l-1} * w_{ij}^l + b_j^l\right) \quad (2.1)$$

En donde b_j^l es el factor del *bias* y k^l es el número de filtros utilizados en la capa l , FM_i^{l-1} corresponde a la imagen de entrada cuando $l=1$. La cantidad de FM que se generen dependerá del número de filtros que contenga la capa de convolución, Figura 2.3.

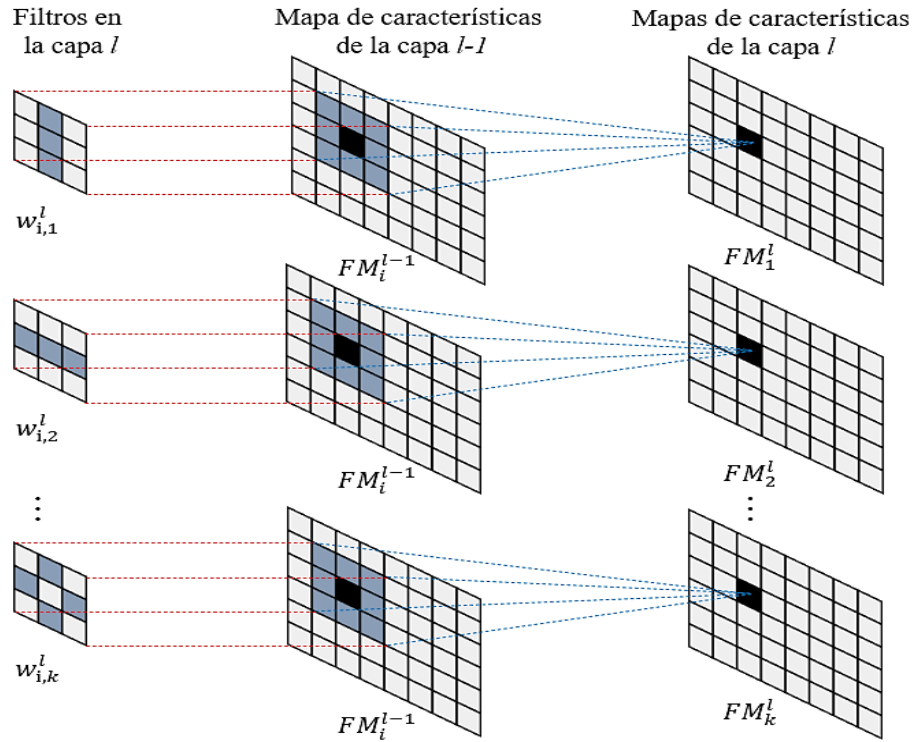


Figura 2.3. Ejemplo de la generación de mapas de características en la l -ésima capa.

En cuanto a la función de activación, esta puede ser cualquier función no lineal, diferenciable y monótonicamente incremental. Entre las funciones de activación más populares se encuentran: sigmoideal, tangente hiperbólica ($\tanh$) y *softmax*, pero en los últimos años la función ReLU mostrada en la Ec. (2.2) es la más utilizada en DNN. Esto debido a que las redes neuronales con ReLU tienen tiempos de entrenamiento con menor duración en comparación

con otras en las que se utilizan funciones de activación como la sigmoidal o tanh [74].

$$\text{ReLU}(x) = \max(0, x) = \begin{cases} 0 & \text{para } x < 0 \\ x & \text{para } x \geq 0 \end{cases} \quad (2.2)$$

2.2.2 Capa de agrupación

En una CNN la secuencia de una capa de convolución y una función de activación frecuentemente va seguido de una capa opcional de agrupamiento o también conocida como submuestreo. Esta capa tiene como objetivos reducir la carga computacional, el uso de memoria y el número de parámetros en la red. Para llevar a cabo este submuestreo las técnicas más populares son el *max-pooling* y el *average pooling*, en donde la primera consiste en tomar el valor máximo dentro de una subregión de la entrada de tamaño $r \times r$, mientras que en la segunda se obtiene un promedio de los elementos de dicha subregión como se muestra en la Figura 2.4.

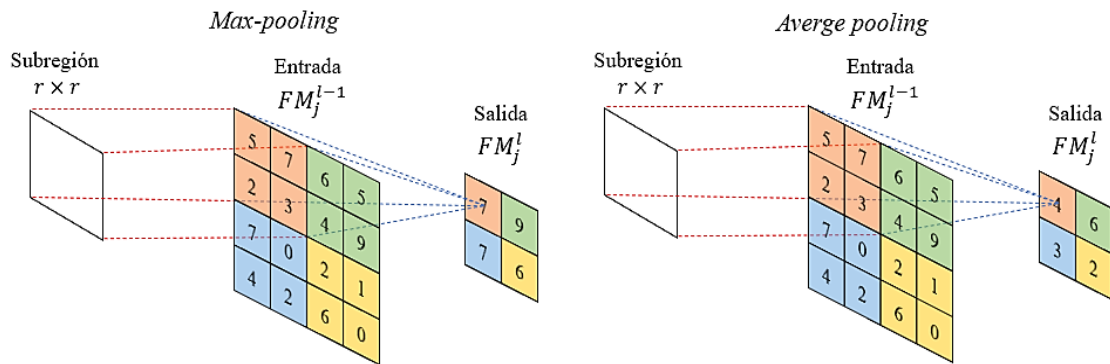


Figura 2.4. Ejemplo de la operación de *max-pooling* y *average pooling*.

Algunos autores consideran que la capa de agrupación utilizando *max-pooling* hace a las CNN invariantes a la traslación [75] y mediante la reducción de parámetros se logra disminuir el riesgo de sobre entrenamiento [76]. Sin embargo, al descartar la información menos significativa también se pierde información de la localización de las características, por tal motivo esta estrategia funciona bien para tareas simples en donde solo interesa conocer si el objeto existe en la imagen o no, pero puede no ser adecuada para otro tipo de tareas más complejas. De manera matemática el *max-pooling* se define en la Ec. (2.3).

$$FM_j^l = \max(FM_j^{l-1}) \quad (2.3)$$

En donde $\max(\cdot)$ consiste en tomar el valor máximo en cada subregión $r \times r$ de FM_j^{l-1} .

2.2.3 Capa totalmente conectada (FC)

Al final de una red neuronal convolucional, hay al menos una capa totalmente conectada (FC). Una capa FC se diferencia de las demás capas debido a que cada una de sus neuronas está conectada a cada una de las neuronas de la capa anterior. Esta capa tiene como objetivo realizar una clasificación basada en los atributos extraídos mediante las capas previas.

De manera analítica la salida de una capa FC se obtiene de la siguiente manera. En el caso de ser la primera FC, es necesario concatenar todos los mapas de características de la capa previa es decir $DF^{l-1} = [FM_{j=1}^{l-1}, \dots, FM_{k^{l-1}}^{l-1}]$ para obtener una representación en forma vectorial. Una vez creado el vector DF^{l-1} este es multiplicado por la matriz de pesos W^l de las neuronas en la capa totalmente conectada y se evalúa por una función de activación no lineal $f(\cdot)$, siendo la función *softmax* la más utilizada. Finalmente, se obtiene a la salida un nuevo vector DF^l con una longitud igual al número de neuronas que contenga la capa l , utilizando Ec (2.4). En el caso de contar con más de una capa totalmente conectada, su salida seguirá siendo determinada por Ec (2.4), pero ya no es necesario realizar ninguna concatenación, puesto que la capa previa $l-1$ es otra capa totalmente conectada cuya salida es un vector.

$$DF^l = f(W^l DF^{l-1} + b^l) \quad (2.4)$$

2.2.4 Entrenamiento e hiperparámetros de una CNN

El entrenamiento de una CNN se formula en términos de minimizar una función de pérdida. En este contexto, entrenar una CNN significa buscar un conjunto de parámetros (o pesos) de la red en los que la función de pérdida tiene un valor mínimo. El descenso del gradiente es un algoritmo de optimización utilizado para minimizar el error mediante el cálculo de los gradientes necesarios para actualizar los valores de los parámetros de la red. El algoritmo de aprendizaje más común y exitoso para los modelos de aprendizaje profundo es el *backpropagation* basado en el descenso del gradiente, en donde el error se propaga hacia atrás desde la última hasta la primera capa de la red [74].

A su vez, las CNN cuentan con una gran cantidad de hiperparámetros que son utilizados para controlar el comportamiento del modelo, estos no se utilizan para modelar los datos directamente, pero influyen en la capacidad y características de aprendizaje del modelo [74]. Entre los hiperparámetros más importantes en una CNN se encuentran:

1. Tasa de aprendizaje: La tasa de aprendizaje (η) es la encargada de regular el tamaño de paso en algoritmos de optimización como el descenso del gradiente, durante el entrenamiento de la CNN. Su valor debe ser un número real positivo tal que $0 < \eta < 1$. Para su ajuste se recomienda iniciar con un valor alto (0.8 aproximadamente) y reducirlo de manera gradual a medida que avanza el aprendizaje.
2. Tamaño del filtro: Los filtros deben de ser de un tamaño mayor a 2×2 y menor al tamaño de la entrada, el tamaño de los filtros convencionales varía en el rango de 3×3 a 11×11 . Las señales con más de un canal como el caso de las imágenes a color en el espacio RGB que cuentan con 3 canales, implica que los filtros aplicados deben contar con el mismo número de canales.
3. Número de filtros por capa: Este valor puede ser cualquier número razonable, para el cual no existe ninguna convención e incluso puede variar entre distintas capas convolucionales dentro de la misma red. En algunas CNN como Alexnet y VGG se utilizan 96 y 64 filtros respectivamente.
4. Stride: Es el número de píxeles que el filtro se desplaza de manera horizontal y vertical sobre la imagen de entrada, dicho valor por defecto es de [1 1], en donde el filtro se desplaza por cada fila y columna pixel por pixel.
5. Zero Padding: Este hiperparámetro describe el número de filas y columnas de la señal de entrada que son rellenas con ceros, con el fin de tener un control sobre el tamaño de la señal de salida.

2.3 Análisis de CNN propuestas en la literatura

En esta sección se presenta un análisis de algunos modelos de redes neuronales convolucionales, que pueden ser utilizadas en el desarrollo de un MA. Para este análisis, se

crearon dos experimentos que permiten evaluar la robustez de las CNN, así como su capacidad para discriminar objetos. Con el objetivo de identificar cuáles son las CNN que mejor se adaptan a los distintos desafíos que se presentan durante el seguimiento de múltiples objetos.

A continuación, se describe la metodología implementada para evaluar la robustez y el grado de discriminación de los distintos modelos de CNN considerados en este estudio.

2.3.1 Selección de los modelos de CNN

Las CNN seleccionadas para este análisis son: Alexnet [77], ResNet-18 [78], VGG-16 [79], Inception-ResNet-v2 [80], DenseNet-201 [81] y NASNet-Large [82], las cuales fueron seleccionadas para representar de manera estratégica el amplio espectro de CNN que han sido propuestas en la literatura para la clasificación de imágenes. En la Figura 2.5 se muestran algunas de las CNN más populares en la clasificación de imágenes a gran escala, ordenadas mediante una relación entre su exactitud y el tiempo que tardan en entregar un resultado. El área de los círculos es proporcional al espacio en megabytes necesarios para almacenar cada red. Las CNN seleccionadas para este análisis aparecen resaltadas en color naranja.

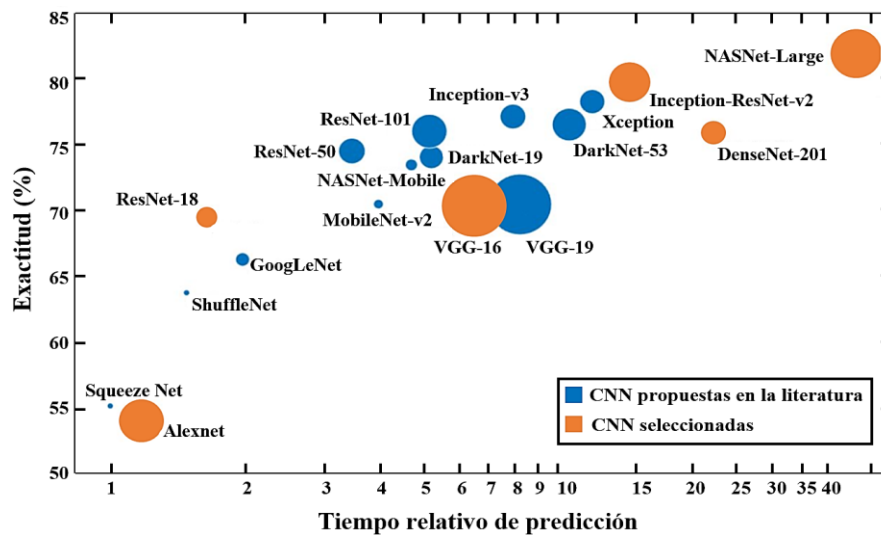


Figura 2.5 Relación entre la exactitud y el tiempo de predicción de distintas CNN propuestas en la literatura.

La razón de utilizar CNN entrenadas para la clasificación de imágenes es debido a que en el seguimiento de objetos uno de los principales problemas es que no se cuenta con la cantidad de datos necesarios para entrenar un modelo de CNN, ya que la poca información que se tiene

sobre los objetivos no es suficiente para dicho propósito. Para solucionar este problema se utiliza la transferencia de aprendizaje, en donde se emplea un modelo de CNN preentrenado, cuya etapa de entrenamiento fue llevada a cabo en una base de datos extensa para dar solución a un problema similar al que se quiere resolver en el seguimiento de objetos.

2.3.2 Escenarios de prueba

Para evaluar el desempeño de las CNN seleccionadas, se crearon 6 escenarios de prueba en donde se obtiene el desempeño de las CNN ante distintos desafíos que comúnmente se presentan en el seguimiento de objetos, tales como: cambios de apariencia, cambios de tamaño, cambios de iluminación, oclusiones, apariencias similares y ruido en las imágenes. A continuación, se describe la manera en que se conformó cada uno de los escenarios de prueba.

1. Cambios de apariencia (CA): Este conjunto está compuesto por objetivos captados en cuadros donde se aprecia un cambio en su aspecto físico.
2. Cambios de tamaño (CT): Este conjunto consta de objetivos captados en cuadros donde se aprecia un acercamiento o alejamiento del objetivo con respecto a la cámara, provocando que el tamaño del objetivo aumente o disminuya respectivamente.
3. Cambios de iluminación (CI): Esta condición se simuló realizando ajustes en los objetivos para aclarar y oscurecer su aspecto de manera gradual. Esta condición al ser simulada permite un mejor control sobre el desafío que se desea evaluar, debido a que evita posibles combinaciones entre desafíos como CI y CA.
4. Apariencias similares (AS): Este conjunto está compuesto por objetivos cuyo aspecto físico es muy parecido.
5. Ruido en la imagen (RI): Esta condición se simuló mediante la agregación de ruido sobre los objetivos de manera gradual. En esta simulación se utilizaron dos tipos de ruido, el ruido Gaussiano y el ruido sal y pimienta, seleccionados por ser los que comúnmente se presentan en imágenes digitales.

6. Oclusiones (OCL): Este conjunto está compuesto por objetivos captados en cuadros consecutivos, en donde en un principio el objetivo se encuentra visible y gradualmente va siendo ocluido por algún otro objeto hasta que finalmente deja de ser visible.

Cada uno de los 6 escenarios de prueba contiene 5 objetivos captados durante 15 cuadros, extraídos de la base de datos MOT17 [83], en la Figura 2.6 se muestra un ejemplo de cada categoría.



Figura 2.6. Ejemplos de los objetivos que conforman los 6 escenarios de pruebas.

2.3.3 Extracción de características

La extracción de características de los objetivos que conforman los escenarios de pruebas se realiza de la siguiente forma. El vector de características al que llamaremos descriptor se obtiene mediante la concatenación de todos los mapas de características de la capa $l=L-1$ utilizando la Ec. (2.5).

$$DF^{L-1} = [FM_{j=1}^{L-1}, ..., FM_{k^{L-1}}^{L-1}] \quad (2.5)$$

En donde L se refiere a la primera capa totalmente conectada de cada CNN seleccionada, esta capa varía dependiendo de la profundidad de la red, Figura 2.7. Por simplicidad y tomando en cuenta que en este experimento todos los descriptores se obtendrán de la capa $l=L-1$, esta

notación se omitirá y DF_i^j hará referencia al descriptor del i -ésimo objetivo en el j -ésimo cuadro.

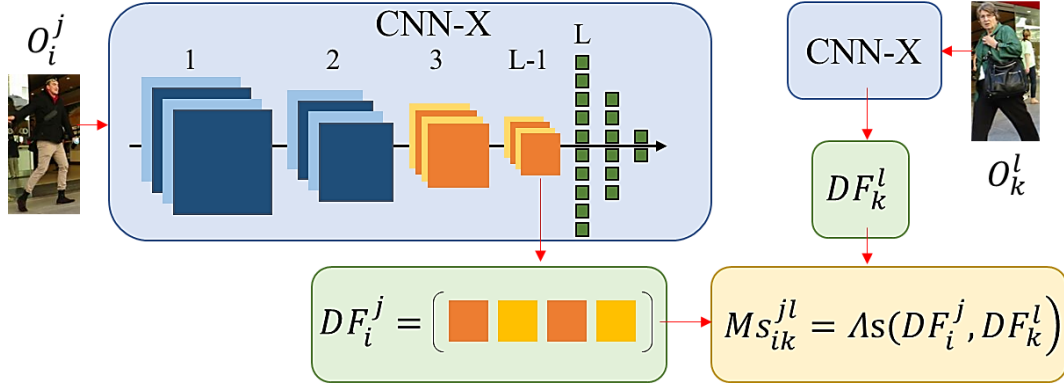


Figura 2.7. Procedimiento para obtener la medida de similitud entre los objetivos O_i^j y O_k^l .

2.3.4 Función de similitud

Una vez obtenido el descriptor de cada objetivo, estos se evalúan por medio de una función de similitud $\Lambda s(\cdot)$, para obtener la medida de similitud Ms , que permite definir si se trata del mismo objetivo o no. En este trabajo se utilizan dos $\Lambda s(\cdot)$. El coeficiente de correlación cruzada ρ , Ec. (2.6), y la distancia euclidiana dE , Ec. (2.7). Antes del cálculo de dE los descriptores DF son normalizados para convertirlos en vectores unitarios $\overline{DF}$.

$$\rho = \frac{\sum_{n=0}^{N-1} (DF_i^j[n]) \cdot (DF_k^l[n])}{\sqrt{\sum_{n=0}^{N-1} (DF_i^j[n])^2 \cdot \sum_{n=0}^{N-1} (DF_k^l[n])^2}} \quad (2.6)$$

$$dE = \sqrt{\sum_{n=0}^{N-1} (\overline{DF_i^j[n]} - \overline{DF_k^l[n]})^2} \quad (2.7)$$

En donde DF_k^l se refiere al descriptor del k -ésimo objetivo en el l -ésimo cuadro. De tal manera que la medida de similitud Ms_{ik}^{jl} representa la similitud entre el objetivo O_i^j con respecto al objetivo O_k^l , como se muestra en la Figura 2.7. Cuando Ms_{ik}^{jl} se obtiene utilizando ρ , los objetivos serán más similares conforme el valor de Ms_{ik}^{jl} se aproxime a 1. Por otro lado, si Ms_{ik}^{jl} se obtiene utilizando dE , entonces la similitud entre objetivos es mayor conforme su valor se aproxime a 0.

En la Figura 2.8 se muestra un ejemplo de la medida de similitud en el escenario de prueba de cambios de apariencia. En este ejemplo mediante el uso de ρ y dE se obtiene la similitud entre el objetivo 5 en el cuadro j con respecto a los otros 4 objetivos y consigo mismo en el cuadro $j+1$ durante 15 cuadros. En este ejemplo solo se muestra la medida de similitud cuando el descriptor se obtiene utilizando la CNN Alexnet, Figura 2.8 (a) y Figura 2.8 (b) y de la CNN Inception-ResNet-v2 Figura 2.8 (c) y Figura 2.8 (d).

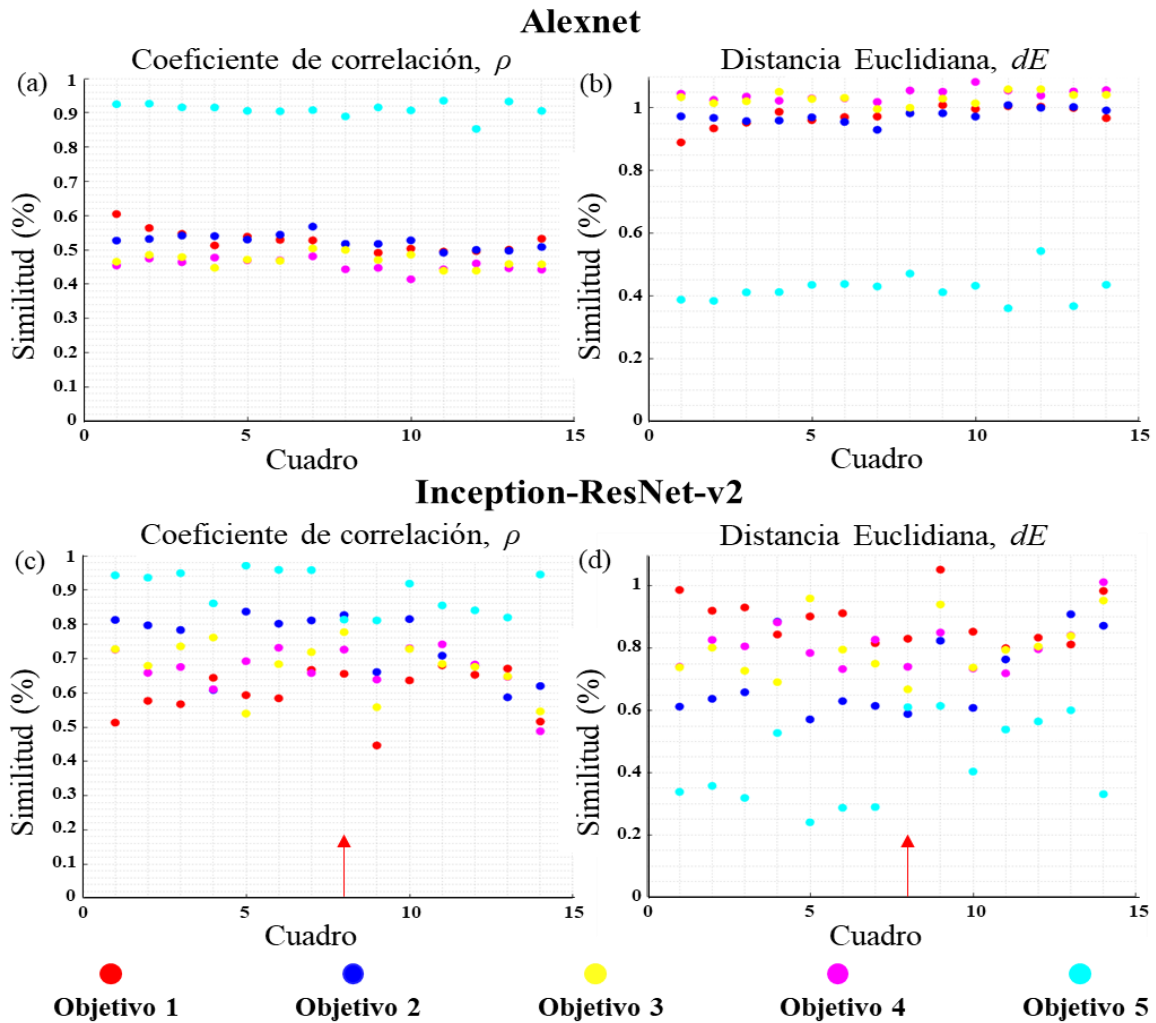


Figura 2.8: Medida de similitud utilizando ρ , dE y los descriptores extraídos de la CNN Alexnet (a), (b) e Inception-ResNet-v2 (c), (d).

En el ejemplo de la Figura 2.8 es evidente que la discriminación del objetivo es mejor al utilizar descriptores extraídos de la CNN Alexnet, ya que esta red mantiene un margen de

discriminación consistente entre el objetivo de interés, y los demás objetivos en la prueba durante el transcurso de cada cuadro. Dicho comportamiento no se presenta al utilizar descriptores extraídos de la CNN Inception-ResNet-v2, ya que el margen de discriminación es más pequeño en comparación con la Alexnet, e incluso inexistente como en el caso del cuadro 8 denotado por la flecha roja, en donde un objetivo distinto al objetivo de interés presenta mayor grado de similitud que el propio objetivo de interés. Con base en estas observaciones se creó el grado de discriminación Gd , una medición que permite resumir la información mostrada en las gráficas de la Figura 2.8 y que sirve para evaluar el desempeño de las CNN en los distintos escenarios de prueba.

2.3.5 Grado de discriminación Gd , MA actualizado

A continuación, se describe el procedimiento para obtener el grado de discriminación Gd . Considere la medida de similitud del objetivo de interés consigo mismo como $MsOI_i^j = As(DF_i^j, DF_i^{j+1})$, y la medida de similitud del objetivo de interés con los demás objetivos presentes en el escenario de prueba como $MsOP_i^j = As(DF_i^j, DF_{k \neq i}^{j+1})$. Nótese que únicamente se evalúa la similitud de los objetivos en el cuadro actual y el siguiente (j y $j+1$), lo que corresponde a un modelo de apariencia actualizado, donde dicho modelo se actualiza después de procesar cada cuadro. De tal forma que el margen de discriminación en el cuadro j para el objetivo i , Se_i^j , está dado por la Ec. (2.8) cuando se utiliza ρ y por la Ec. (2.9) cuando se utiliza dE como función de similitud.

$$Se_i^j = MsOI_i^j - \max(MsOP_i^j) \quad (2.8)$$

$$Se_i^j = \min(MsOP_i^j) - MsOI_i^j \quad (2.9)$$

Tomando en cuenta que cada escenario de prueba contiene 5 objetivos captados durante 15 cuadros, el Gd de cada CNN para un determinado desafío se obtiene mediante el promedio del margen de discriminación durante los 15 cuadros para los 5 objetivos, utilizando la Ec. (2.10).

$$Gd = \frac{1}{5} \sum_{i=1}^5 \frac{1}{14} \sum_{j=1}^{14} Se_i^j \quad (2.10)$$

Donde Gd se puede interpretar como la facilidad de distinguir un objeto de interés de los demás objetivos presentes, ante un determinado desafío en el seguimiento de objetos. De esta manera, entre más alto es el valor de Gd dicho desafío es superado con mayor facilidad. Cabe mencionar que no existe una comparación directa entre los resultados al utilizar ρ y dE como función de similitud. En la Figura 2.9 se ejemplifica el procedimiento para obtener el Gd ante un determinado desafío utilizando ρ .

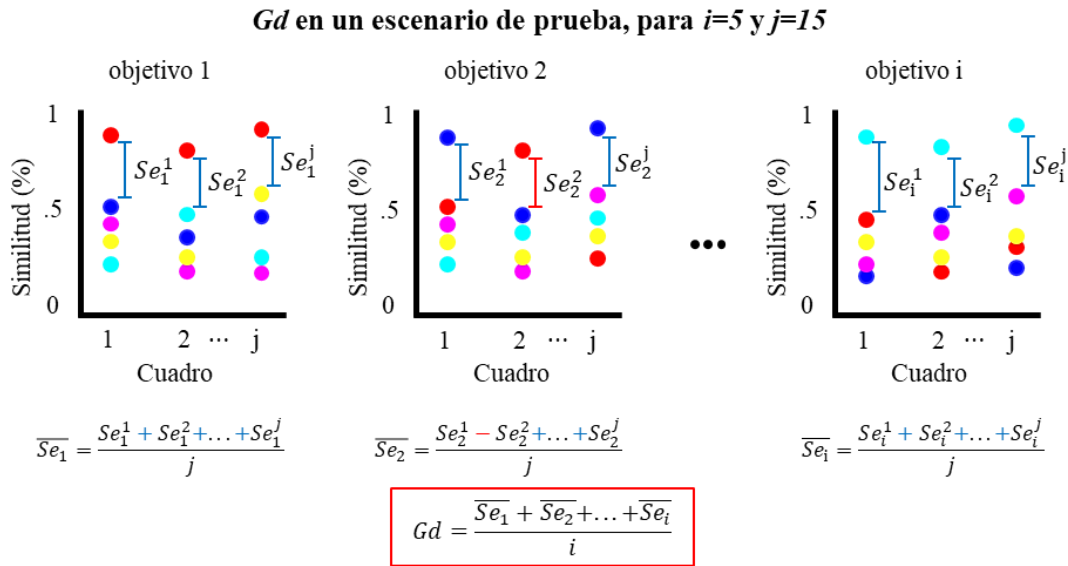


Figura 2.9. Procedimiento para obtener el grado de discriminación de cada CNN ante un determinado desafío.

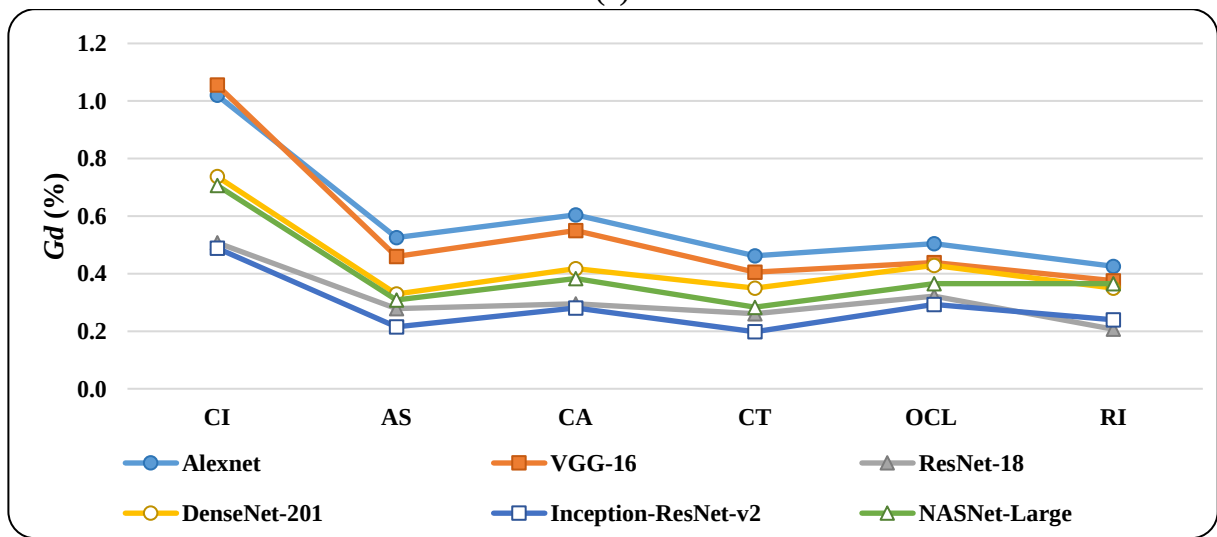
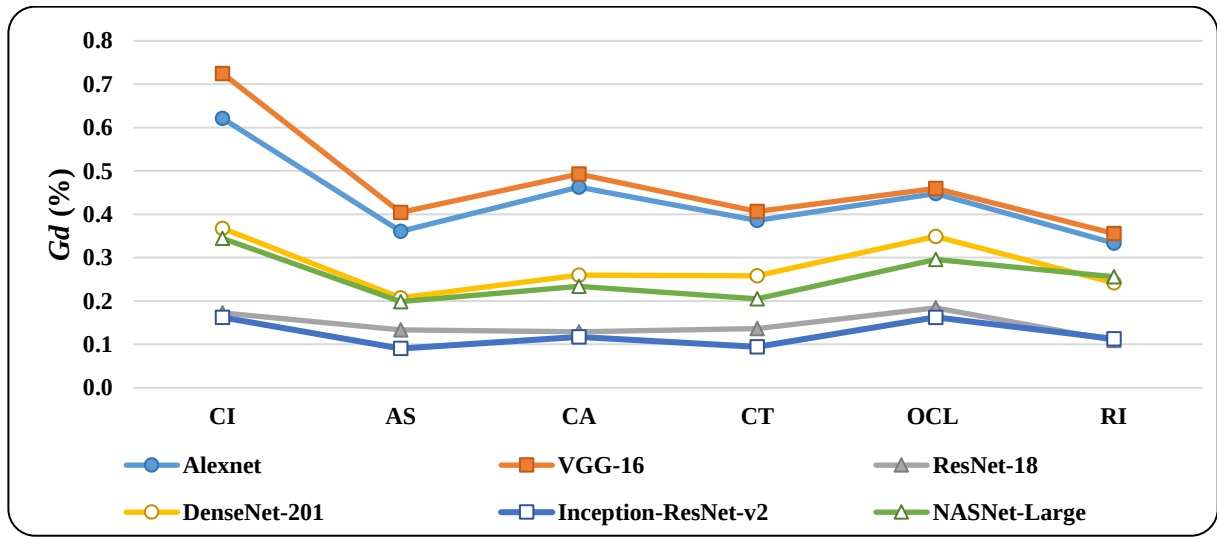
Los resultados de esta primera versión del experimento se muestran en la Tabla 2.2 al utilizar ρ , y en la Tabla 2.3 al utilizar dE . Esta misma información se muestra de forma gráfica en la Figura 2.10 (a) y la Figura 2.10 (b) respectivamente.

Tabla 2.2. Gd para las distintas CNN utilizando ρ en un MA actualizado.

CNN	CI	AS	CA	CT	OCL	RI
Alexnet	0.62	0.36	0.46	0.39	0.45	0.33
VGG-16	0.72	0.40	0.49	0.41	0.46	0.36
ResNet-18	0.17	0.13	0.13	0.14	0.18	0.11
DenseNet-201	0.37	0.21	0.26	0.26	0.35	0.24
Inception-ResNet-v2	0.16	0.09	0.12	0.09	0.16	0.11
NASNet-Large	0.34	0.20	0.23	0.20	0.30	0.26

Tabla 2.3. G_d para las distintas CNN utilizando dE en un MA actualizado.

CNN	CI	AS	CA	CT	OCL	RI
Alexnet	1.02	0.53	0.60	0.46	0.50	0.43
VGG-16	1.06	0.46	0.55	0.41	0.44	0.38
ResNet-18	0.51	0.28	0.30	0.26	0.32	0.21
DenseNet-201	0.74	0.33	0.42	0.35	0.43	0.35
Inception-ResNet-v2	0.49	0.22	0.28	0.20	0.29	0.24
NASNet-Large	0.71	0.31	0.38	0.28	0.37	0.37

Figura 2.10. G_d para las distintas CNN utilizando ρ (a) y dE (b), en un MA actualizado.

Los resultados observados en esta primera versión del experimento con ambas funciones de similitud muestran que las CNN Alexnet y VGG-16 alternan en el primer lugar. Esto quiere decir que para el seguimiento de objetos bajo las condiciones en que se desarrolló el experimento, el uso de estos modelos es más recomendable para superar con éxito los distintos desafíos evaluados. Cabe mencionar que la Alexnet y la VGG-16 son las CNN menos profundas que se consideraron en este análisis, siendo la Alexnet con 8 capas la red con menor número de capas profundas, seguido de la VGG-16 con 16 capas. Esto sugiere que conforme la profundidad de la red se reduce, su capacidad para diferenciar objetivos incrementa. Lo cual difiere respecto a otros problemas como la clasificación de imágenes, donde es bastante aceptada la idea de que el desempeño general de la clasificación mejora conforme la profundidad de la red se incrementa (ver sección 2.3.7).

2.3.6 Grado de discriminación Gd , MA no actualizado

En la primera versión del experimento únicamente se evalúa la similitud de los objetivos en cuadros subsecuentes. En esta segunda versión se plantean algunas modificaciones al primer experimento con el objetivo de evaluar la robustez de los descriptores extraídos de las distintas CNN. Dichas modificaciones consisten en que la medida de similitud del objetivo de interés consigo mismo este dada por $MsOI_l^j = As(DF_i^1, DF_i^{j+1})$, y la medida de similitud del objetivo de interés con los demás objetivos en la prueba este dada por $MsOP_l^j = As(DF_i^1, DF_{k \neq i}^{j+1})$. Esto equivale a un modelo de apariencia en donde el modelo no se actualiza, quedando fijo desde el primer cuadro para cada uno de los objetivos. En esta versión, la similitud entre los objetivos se evalúa en el cuadro 1 y el cuadro $j+1$: $j=1, 2, \dots, 14$ obteniendo la similitud de los objetivos en cuadros temporalmente más separados, lo que permite evaluar la robustez de las CNN para modelar cambios más drásticos en los objetivos.

El proceso para obtener el Gd en esta versión del experimento se realiza de la misma forma que en la versión anterior utilizando las ecuaciones (2.8), (2.9) y (2.10).

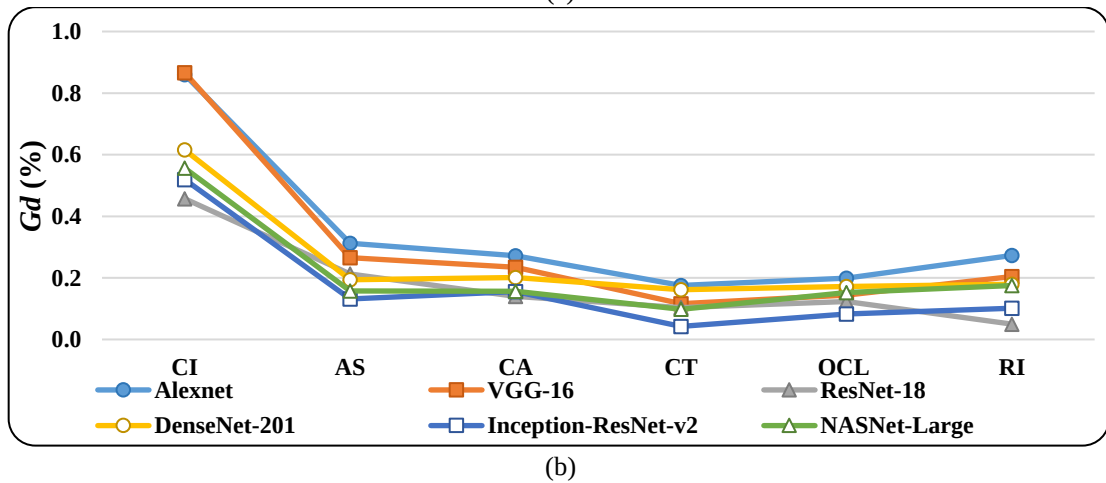
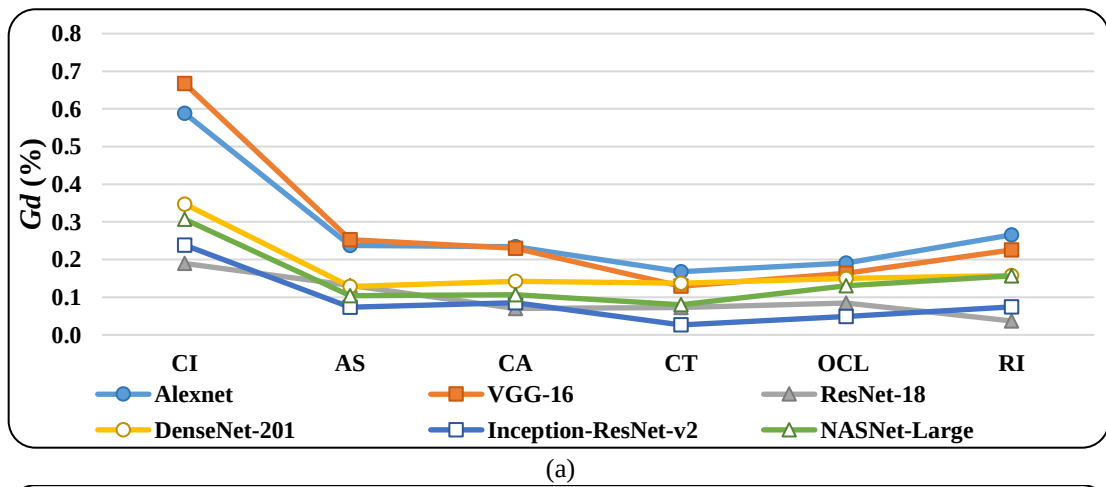
Los resultados para esta versión del experimento se muestran en la Tabla 2.4 al utilizar ρ , y en la Tabla 2.5 al utilizar dE . Esta misma información se muestra de forma gráfica en la Figura 2.11 (a) y la Figura 2.11 (b) respectivamente.

Tabla 2.4. Gd para las distintas CNN utilizando ρ en un MA no actualizado.

CNN	CI	AS	CA	CT	OCL	RI
Alexnet	0.59	0.24	0.23	0.17	0.19	0.27
VGG-16	0.67	0.25	0.23	0.13	0.16	0.23
ResNet-18	0.19	0.13	0.07	0.07	0.08	0.04
DenseNet-201	0.35	0.13	0.14	0.14	0.15	0.16
Inception-ResNet-v2	0.24	0.07	0.09	0.03	0.05	0.07
NASNet-Large	0.31	0.10	0.11	0.08	0.13	0.16

Tabla 2.5. Gd para las distintas CNN utilizando dE en un MA no actualizado.

CNN	CI	AS	CA	CT	OCL	RI
Alexnet	0.86	0.31	0.27	0.18	0.20	0.27
VGG-16	0.87	0.27	0.23	0.12	0.14	0.20
ResNet-18	0.46	0.21	0.14	0.10	0.12	0.05
DenseNet-201	0.62	0.19	0.20	0.16	0.17	0.18
Inception-ResNet-v2	0.52	0.13	0.16	0.04	0.08	0.10
NASNet-Large	0.56	0.16	0.16	0.10	0.15	0.18

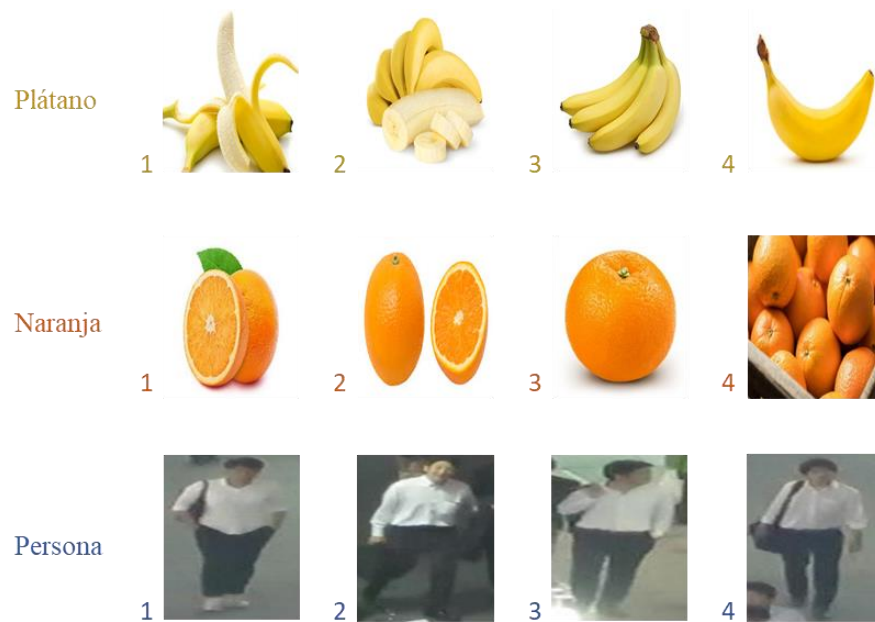
Figura 2.11. Gd para las distintas CNN utilizando ρ (a) y dE (b), en un MA no actualizado.

En esta versión del experimento los resultados muestran que las CNN Alexnet y VGG-16 se mantienen en los primeros lugares. Sin embargo, el G_d se redujo considerablemente en la mayoría de los desafíos. Lo cual indica que ninguno de los modelos de CNN considerados fue capaz de modelar satisfactoriamente la apariencia de los objetivos cuando estos no se encuentran en cuadros subsecuentes, careciendo de robustez para modelar cambios más drásticos en la apariencia de los objetivos.

La robustez para modelar cambios drásticos en la apariencia de los objetivos es una característica muy importante en un MA, debido a que permite que un objetivo que ha sido perdido de vista a causa de una oclusión prolongada pueda volver a ser identificado correctamente. En este análisis se observa que ninguno de los modelos de CNN tradicionales fue capaz de lograr dicho objetivo. Por tal motivo en esta tesis se decidió desarrollar un MA basado en una CNN en configuración siamesa, debido a que este tipo de arquitectura permite un mejor aprendizaje sobre la similitud de los objetivos, permitiendo que adquieran una mayor robustez para modelar cambios drásticos en la apariencia de los objetivos con respecto a las CNN tradicionales. Los detalles sobre la red siamesa se proporcionan en la sección 2.4.

2.3.7 Comparación entre modelos de CNN seleccionados

En esta sección se presenta una explicación de los resultados obtenidos en el experimento del grado de discriminación en sus 2 versiones. Para esto se diseñó un nuevo experimento, en donde se seleccionaron 12 imágenes compuestas de 3 clases diferentes (*persona*, *plátano* y *naranja*), como se muestra en la Figura 2.12. Dichas imágenes se propagaron por los mismos modelos de CNN seleccionados anteriormente para obtener sus respectivos descriptores. La extracción de características para este experimento se realiza de la misma manera que se describe en la sección 2.3.3 mediante la concatenación de todos los mapas de características de la capa $l=L-1$. Posteriormente el descriptor de cada determinada imagen se compara con los descriptores de las imágenes restantes utilizando como función de similitud ρ y dE , con el objetivo de medir la similitud entre los descriptores de las distintas imágenes de prueba. En la Tabla 2.6 se muestra un ejemplo de la similitud entre los descriptores de las imágenes de prueba utilizando ρ y la CNN Alexnet.

Figura 2.12. Conjunto de imágenes de la clase *plátano*, *naranja* y *persona*.Tabla 2.6. Similitud entre los descriptores del conjunto de imágenes de prueba utilizando ρ y la CNN Alexnet.

		Plátano				Naranja				Persona			
		1	2	3	4	1	2	3	4	1	2	3	4
Plátano	1	1.00	0.37	0.35	0.31	0.30	0.23	0.22	0.26	0.15	0.17	0.15	0.14
	2	0.37	1.00	0.38	0.26	0.27	0.35	0.33	0.27	0.15	0.14	0.16	0.16
	3	0.35	0.38	1.00	0.39	0.26	0.26	0.29	0.21	0.14	0.15	0.16	0.13
	4	0.31	0.26	0.39	1.00	0.32	0.33	0.40	0.22	0.11	0.13	0.17	0.12
Naranja	1	0.30	0.27	0.26	0.32	1.00	0.47	0.59	0.33	0.13	0.13	0.13	0.14
	2	0.23	0.35	0.26	0.33	0.47	1.00	0.58	0.28	0.10	0.13	0.15	0.12
	3	0.22	0.33	0.29	0.40	0.59	0.58	1.00	0.30	0.09	0.11	0.17	0.11
	4	0.26	0.27	0.21	0.22	0.33	0.28	0.30	1.00	0.13	0.16	0.14	0.16
Persona	1	0.15	0.15	0.14	0.11	0.13	0.10	0.09	0.13	1.00	0.34	0.32	0.34
	2	0.17	0.14	0.15	0.13	0.13	0.13	0.11	0.16	0.34	1.00	0.44	0.41
	3	0.15	0.16	0.16	0.17	0.13	0.15	0.17	0.14	0.32	0.44	1.00	0.36
	4	0.14	0.16	0.13	0.12	0.14	0.12	0.11	0.16	0.34	0.41	0.36	1.00

En el ejemplo de la Tabla 2.6 es posible observar que el resultado de la evaluación de similitud es una matriz simétrica cuya diagonal principal es el valor de similitud más alto, que se produce cuando se compara un descriptor de determinada imagen de prueba consigo mismo. Los elementos de la matriz resaltados en color amarillo, naranja y azul representan la similitud entre descriptores de la misma clase *plátano-plátano*, *naranja-naranja* y *persona-persona*.

respectivamente. Mientras que los datos resaltados en color verde representan la similitud entre descriptores de clases diferentes, ya sea *plátano-naranja*, *plátano-persona* o *naranja-persona*. Por lo tanto, al promediar dichos valores se puede obtener una representación de que tan similares son los descriptores entre objetivos de la misma clase y objetivos de clases diferentes. Por ser una matriz simétrica es suficiente con considerar solo los elementos por encima de la diagonal principal.

El valor promedio de similitud entre descriptores de las clases de prueba se muestra en la Tabla 2.7 al utilizar ρ , y en la Tabla 2.8 al utilizar dE para cada una de la CNN analizadas. Esta misma información se muestra de forma gráfica en la Figura 2.13 (a) y la Figura 2.13 (b) respectivamente.

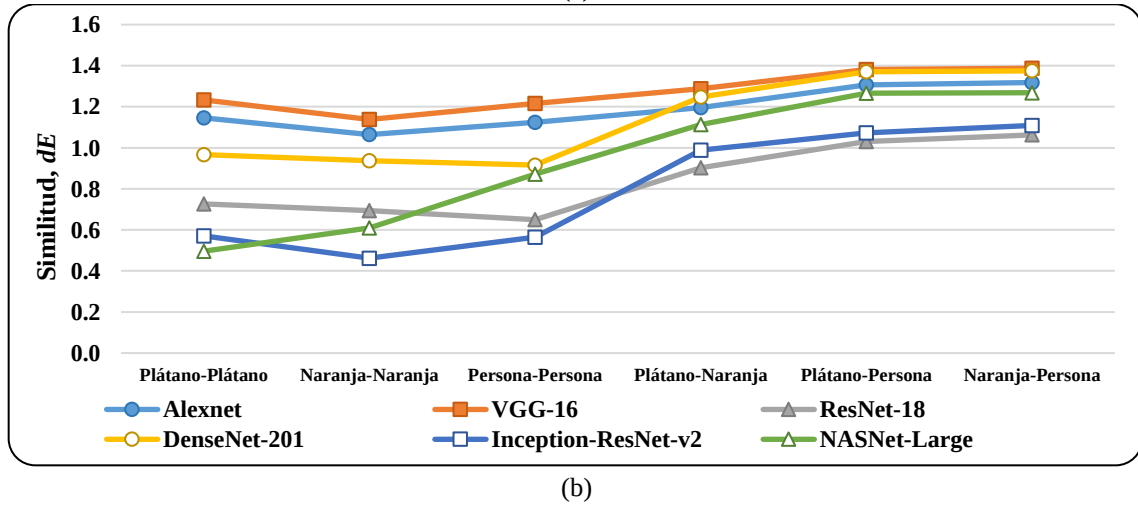
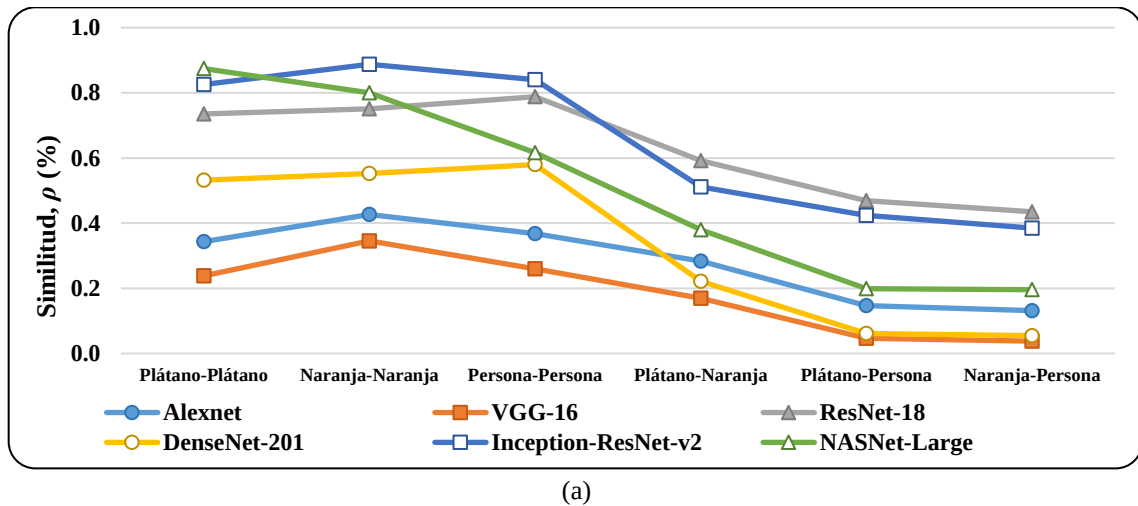


Figura 2.13. Promedio de similitud entre descriptores de las clases de prueba utilizando ρ (a) y dE (b).

Tabla 2.7. Promedio de similitud entre descriptores de las clases de prueba utilizando ρ .

CNN	Plátano Plátano	Naranja Naranja	Persona Persona	Plátano Naranja	Plátano Persona	Naranja Persona
Alexnet	0.34	0.43	0.37	0.28	0.15	0.13
VGG-16	0.24	0.35	0.26	0.17	0.05	0.04
ResNet-18	0.73	0.75	0.79	0.59	0.47	0.43
DenseNet-201	0.53	0.55	0.58	0.22	0.06	0.05
Inception-ResNet-v2	0.83	0.89	0.84	0.51	0.42	0.38
NASNet-Large	0.87	0.80	0.62	0.38	0.20	0.20

Tabla 2.8. Promedio de similitud entre descriptores de las clases de prueba utilizando dE .

CNN	Plátano Plátano	Naranja Naranja	Persona Persona	Plátano Naranja	Plátano Persona	Naranja Persona
Alexnet	1.15	1.06	1.12	1.20	1.31	1.32
VGG-16	1.23	1.14	1.22	1.29	1.38	1.39
ResNet-18	0.73	0.69	0.65	0.90	1.03	1.06
DenseNet-201	0.97	0.94	0.92	1.25	1.37	1.37
Inception-ResNet-v2	0.57	0.46	0.56	0.99	1.07	1.11
NASNet-Large	0.50	0.61	0.87	1.11	1.27	1.27

Los resultados de este experimento al utilizar ρ muestran que modelos como Inception-ResNet-v2 y NasNet-Large (los más profundos) obtienen alrededor de un 80% de similitud al evaluar descriptores de objetos de la misma clase. Mientras que dicha similitud disminuye cerca de un 40% cuando se evalúan descriptores de objetos de clases diferentes. Por otra parte, cuando se utilizan modelos como Alexnet y VGG-16 (los menos profundos) se obtiene una similitud alrededor del 30% para objetos de la misma clase y alrededor del 15% para objetos de clases diferentes. Lo cual es un valor de similitud muy bajo tanto para objetos de la misma clase como para objetos de clases diferentes. En lo referente a los resultados del experimento al utilizar dE estos muestran el mismo comportamiento que al utilizar ρ , teniendo en cuenta que en este caso la similitud entre objetos es mayor cuando su valor es más cercano a cero.

En las secciones 2.3.5 - 2.3.6 se observó que las CNN menos profundas obtienen un mejor desempeño para discriminar objetos en el seguimiento de múltiples objetos. Esto se debe a que en el seguimiento de múltiples objetos los objetos de interés comúnmente corresponden a objetos de la misma clase. Por tal motivo las CNN menos profundas cuya similitud es muy pobre entre objetos de la misma clase, permiten una mejor discriminación en el seguimiento de múltiples objetos en comparación con las redes más profundas.

En otras palabras, estos resultados se pueden explicar teniendo en cuenta la información que codifican las capas más profundas de una CNN, como menciona Chao Ma en [14] esta información se asocia con el contenido semántico de los objetos y su calidad mejora conforme se incrementa la profundidad de la red, el contenido semántico es de gran utilidad para discriminar objetos de distintas clases pero no cuando los objetos pertenecen a la misma clase. Por tal motivo las redes con menor capacidad para describir el contenido semántico (las menos profundas) obtienen mejores desempeños para discriminar objetos de la misma clase, como se observó en el experimento del *Gd* en las secciones 2.3.5 - 2.3.6.

2.4 Propuesta del MA basado en una CNN en configuración siamesa

En esta sección se propone un MA basado en una CNN en configuración siamesa. Este tipo de arquitectura fue seleccionada porque permite un mejor aprendizaje sobre la similitud de los objetivos, debido a que en esta arquitectura el concepto de similitud y disimilitud se ve directamente reflejado en el algoritmo de aprendizaje de la red.

Una red siamesa consiste en utilizar dos CNN idénticas que comparten los mismos parámetros. La principal diferencia entre una red siamesa y una CNN tradicional, es que la red siamesa recibe dos imágenes como entrada, que puede ser un par positivo (imágenes del mismo objetivo) o un par negativo (imágenes de objetivos diferentes). Dichas imágenes se propagan por la red para obtener sus respectivos descriptores, para posteriormente evaluar la similitud entre ellos, siendo esta similitud la salida de la red.

Para el diseño de la red siamesa se tomaron en cuenta los resultados obtenidos en los distintos experimentos realizados en las secciones 2.3.5 - 2.3.7, así como la profundidad de las CNN analizadas.

En una CNN una capa convolucional únicamente puede extraer características simples del objetivo como brillo o bordes. Sin embargo, la sucesión de varias capas convolucionales incrementa la complejidad de las características, de manera que las capas finales de la red pueden extraer características que definen la forma única del objetivo. Como se observó en la sección 2.3.7 este tipo de características no son útiles en el seguimiento de múltiples objetos.

Por tal motivo se decidió diseñar una red con una reducción del 50% en las capas convolucionales con respecto a la CNN Alexnet, por ser la CNN menos profunda que se analizó en los experimentos de las secciones 2.3.5 -2.3.7. Esta reducción se hace con el propósito de que al disminuir el número de capas convolucionales se reduce la capacidad de la red para describir el contenido semántico, permitiendo que las características que se extraen de la red sean de mayor utilidad para discriminar objetivos en el seguimiento de múltiples objetos.

El modelo propuesto consta de 3 capas convolucionales (C), 3 capas de agrupación (P) y una capa totalmente conectada (FC), dadas por las ecuaciones 2.2, 2.3 y 2.4 respectivamente. En la Figura 2.14 se muestra una representación del modelo propuesto, donde el bloque final corresponde a la evaluación de los descriptores mediante la función de similitud ρ , definida en Ec. 2.6. En la sección 2.4.1 se describe con mayor detalle el proceso realizado para el desarrollo del MA propuesto.

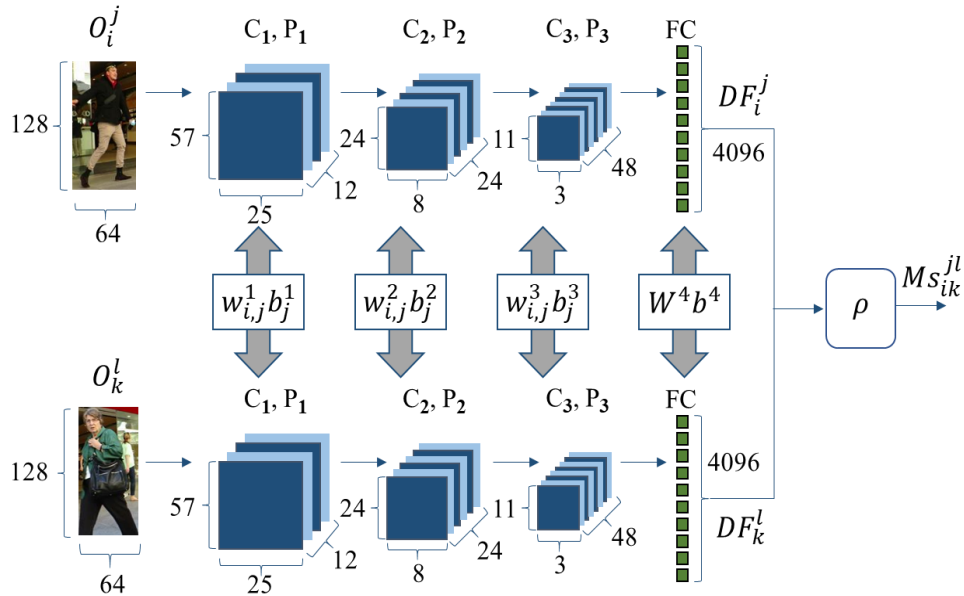


Figura 2.14. MA propuesto basado en una CNN en configuración siamesa.

2.4.1 Diseño de la arquitectura del MA propuesto

En el MA propuesto algunos de sus hiperparámetros fueron ajustados de manera experimental, utilizando el Gd de la sección 2.3.5 para evaluar el desempeño del modelo ante

la modificación de alguno de sus hiperparámetros. Posteriormente se selecciona la configuración del modelo que presenta mejores resultados.

Dentro de los hiperparámetros ajustados se encuentran: la normalización y el espacio de color de las imágenes de entrada, el número y tamaño de los filtros de las capas convolucionales y el número de neuronas de la capa totalmente conectada.

Durante el ajuste solo se varía un hiperparámetro a la vez y la configuración del modelo que obtenga el mejor desempeño se utiliza en las siguientes configuraciones. En la Tabla 2.9 se muestra un ejemplo de la manera en que se ajustó el tipo de normalización que se utiliza en las imágenes de entrada.

Tabla 2.9. Resultados del *Gd* para determinar el tipo de normalización que se utiliza en el MA propuesto.

	Iteraciones	CI	CA	CT	RI		Iteraciones	CI	CA	CT	RI
Modelos con normalización Max	500	0.552	0.523	0.348	0.377	Modelos con normalización Z-score	500	0.733	0.587	0.351	0.444
	1000	0.584	0.551	0.344	0.254		1000	0.700	0.500	0.459	0.458
	1500	0.622	0.471	0.402	0.274		1500	0.762	0.542	0.489	0.404
	2000	0.695	0.557	0.328	0.214		2000	0.665	0.446	0.557	0.369
	2500	0.646	0.543	0.368	0.214		2500	0.558	0.408	0.534	0.425
	3000	0.685	0.579	0.291	0.134		3000	0.642	0.440	0.516	0.431
	Promedio	0.631	0.537	0.347	0.244		Promedio	0.677	0.487	0.485	0.422

La Tabla 2.9 muestra dos normalizaciones candidatas *Max* y *Z-score*, la primera consiste en dividir la imagen de entrada sobre el valor máximo que puede tomar un pixel de la imagen, mientras que la segunda consiste en generar una imagen con media cero y desviación estándar unitaria, mediante la substracción de la media del valor de los pixeles de la imagen dividido sobre su desviación estándar. Para determinar cuál normalización es mejor opción se realizaron 6 entrenamientos de la red siamesa para ambas configuraciones bajo las mismas condiciones, incrementando el número de iteraciones en el entrenamiento desde 500 hasta 3000, para identificar posibles casos de sobreentrenamiento o subentrenamiento de la red siamesa.

El entrenamiento de la red siamesa se realizó por lotes, con un tamaño de lote de 180, con una constante de aprendizaje de 0.00001 y la función de costo entropía cruzada binaria. Las imágenes para el entrenamiento se obtuvieron de la base de datos CUHK03 [84], la cual está

diseñada para la tarea de reidentificación de personas vistas desde 10 ángulos distintos. Durante el entrenamiento se proporcionaron pares de imágenes positivos y pares de imágenes negativos en la misma proporción.

Una vez realizados los 6 entrenamientos para cada configuración, se promedió el Gd obtenido por los distintos modelos como se muestra en la Tabla 2.10. En donde se observa que la normalización Z -score obtiene mejores desempeños en los desafíos de CI, CT y RI, siendo superada solo en el desafío de CA. Por tal motivo la normalización Z -score es considerada como mejor opción y es la normalización que se utilizará en el modelo propuesto y durante el ajuste de los siguientes hiperparámetros. En la Tabla 2.11 se muestra un resumen de los demás hiperparámetros que fueron ajustados de la misma forma que se describió para el tipo de normalización. Los elementos en la Tabla 2.11 que aparecen resaltados en color amarillo indican la capa de la red siamesa en donde se realiza la modificación de cada hiperparámetro.

Tabla 2.10. Resultados del Gd promedio para determinar el tipo de normalización que se aplicará a la imagen de entrada en el MA propuesto.

Imagen de entrada	Normalización			
	CI	CA	CT	RI
Max	0.6305	0.5375	0.3467	0.2442
Z-score	0.6767	0.4872	0.4845	0.4218

Tabla 2.11. Resultados del Gd promedio para determinar los hiperparámetros del MA propuesto.

				Espacio de color			
				CI	CA	CT	RI
Escala de grises				0.2039	-0.0270	0.0000	0.0000
HSI				0.5972	0.4932	0.4844	0.4427
RGB				0.6767	0.4872	0.4845	0.4218

				Tamaño de filtros			
C1	C2	C3		CI	CA	CT	RI
19	13	3		0.6742	0.4977	0.4266	0.4530
15	9	3		0.6767	0.4872	0.4845	0.4218
11	5	3		0.5989	0.4669	0.4998	0.4215

				Número de filtros			
C1	C2	C3		CI	CA	CT	RI
24	48	96		0.6767	0.4872	0.4845	0.4218
12	24	48		0.7112	0.5358	0.4458	0.4280
6	12	24		0.5297	0.4385	0.2801	0.4060

Tabla 2.11. Continuación.

FC	Número de neuronas			
	CI	CA	CT	RI
4096	0.7112	0.5358	0.4458	0.4280
2048	0.6584	0.4846	0.3756	0.4248
1024	0.7302	0.5329	0.3668	0.4115

En la Tabla 2.11 las letras rojas indican la configuración ganadora para cada hiperparámetro. Cabe mencionar que en algunos hiperparámetros no es muy evidente que una configuración sea mejor que otra, puesto que se obtienen resultados muy similares en la mayoría de los desafíos. Para seleccionar una configuración ganadora en esta situación se obtiene el promedio de todos los desafíos evaluados y la configuración que obtenga el valor promedio más alto es la que se considera como mejor opción. Los desafíos de apariencias similares y oclusiones no se consideran durante el ajuste de los hiperparámetros debido a que la red siamesa no es capaz de superar con éxito estos desafíos, lo cual se abordará con más detalle en la sección 2.4.3.

Las características de la arquitectura final del MA propuesto se muestran en Tabla 2.12. Esta arquitectura es el resultado de haber seleccionado la mejor opción en los diferentes hiperparametros que se ajustaron. Por tal motivo esta configuración es la que mayor *Gd* obtiene en la mayoría de los desafíos evaluados, lo que la convierte en la mejor opción para el seguimiento de múltiples objetos en comparación con las demás configuraciones utilizadas.

Tabla 2.12 Arquitectura del MA propuesto basado en una CNN en configuración siamesa.

Tipo de capa	Propiedades de la capa
Imagen de entrada	128x64x3 RGB con normalización z-score
Capa Convolutiva 1, C1	12 filtros 15x15x3 con <i>stride</i> [1 1], <i>padding</i> [0 0 0 0] y ReLU
Capa <i>Max Pooling</i> 1, P1	2x2 <i>max pooling</i> con <i>stride</i> [2 2] y <i>padding</i> [0 0 0 0]
Capa Convolutiva 2, C2	24 filtros 9x9x12 con <i>stride</i> [1 1], <i>padding</i> [0 0 0 0] y ReLU
Capa <i>Max Pooling</i> 2, P2	2x2 <i>max pooling</i> con <i>stride</i> [2 2] y <i>padding</i> [0 0 0 0]
Capa Convolutiva 3, C3	48 filtros 3x3x24 con <i>stride</i> [1 1], <i>padding</i> [0 0 0 0] y ReLU
Capa <i>Max Pooling</i> 3, P3	2x2 <i>max pooling</i> con <i>stride</i> [2 2] y <i>padding</i> [0 0 0 0]
Capa totalmente conectada, FC	4096 neuronas con sigmoideal

2.4.2 Determinación del número de épocas de entrenamiento del MA propuesto

En la sección 2.4.1 se determinó la arquitectura del MA propuesto mediante el ajuste de sus hiperparámetros. En esta sección se evalúa el Gd que obtiene dicha arquitectura al incrementar el número de épocas en su entrenamiento, con el objetivo de evaluar la evolución del Gd de la red siamesa y evitar posibles problemas de sobreentrenamiento o subentrenamiento de la red.

Para realizar esta evaluación se entrenó la arquitectura definida en la sección 2.4.1 variando el número de épocas de entrenamiento desde 500 hasta 5000, con incrementos de 500 épocas. Este entrenamiento se repitió 10 veces, obteniendo un total de 100 modelos entrenados 10 por cada determinado número de épocas. La única diferencia entre los modelos entrenados con el mismo número de épocas es que las imágenes de entrenamiento son proporcionadas de manera aleatoria durante el entrenamiento.

Una vez entrenados los 100 modelos se evaluó el desempeño de cada uno de ellos mediante el experimento del Gd de la sección 2.3.5. El Gd de los modelos entrenados con el mismo número de épocas se promedia, permitiendo obtener una representación del desempeño de los modelos al modificar el número de épocas de entrenamiento como se muestra en la Figura 2.15.

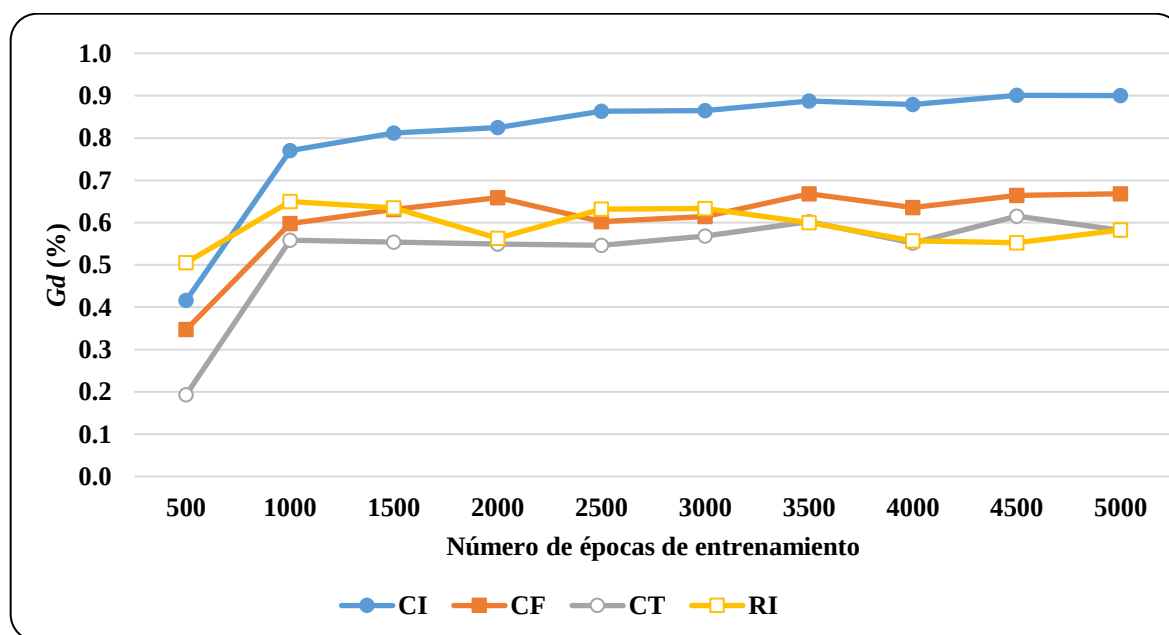


Figura 2.15. Evolución del Gd de los modelos ante diferente número de épocas de entrenamiento.

En la Figura 2.15 se observa que los modelos entrenados con 500 épocas presentan posibles problemas de subentrenamiento, debido a que presentan los desempeños más bajos en todos los desafíos evaluados. Dicho desempeño mejora considerablemente en los modelos entrenados con 1000 épocas. Mientras que al seguir incrementando el número de épocas no se observa que el desempeño mejore significativamente. Por tal motivo y para evitar posibles problemas de sobreentrenamiento se decidió utilizar los modelos que fueron entrenados con 1000 épocas para seleccionar el modelo ganador.

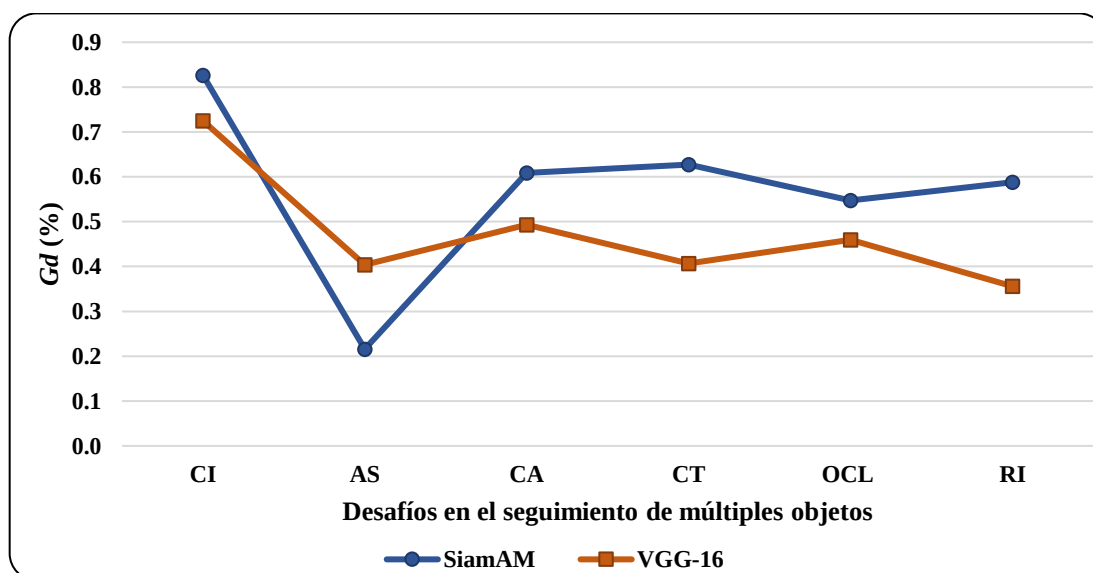
En Tabla 2.13 se muestra el Gd de los 10 modelos entrenados con 1000 épocas, en donde aparece resaltado en color rojo el modelo ganador, el cual fue seleccionado por haber obtenido el mejor desempeño en 2 de los 4 desafíos evaluados. A dicho modelo nos referiremos con el nombre de SiamAM (*Siamese Appearance Model*) y será el modelo de apariencia que se utilizará en el algoritmo propuesto para el seguimiento de múltiples objetos.

Tabla 2.13. Gd de los 10 modelos entrenados con 1000 épocas.

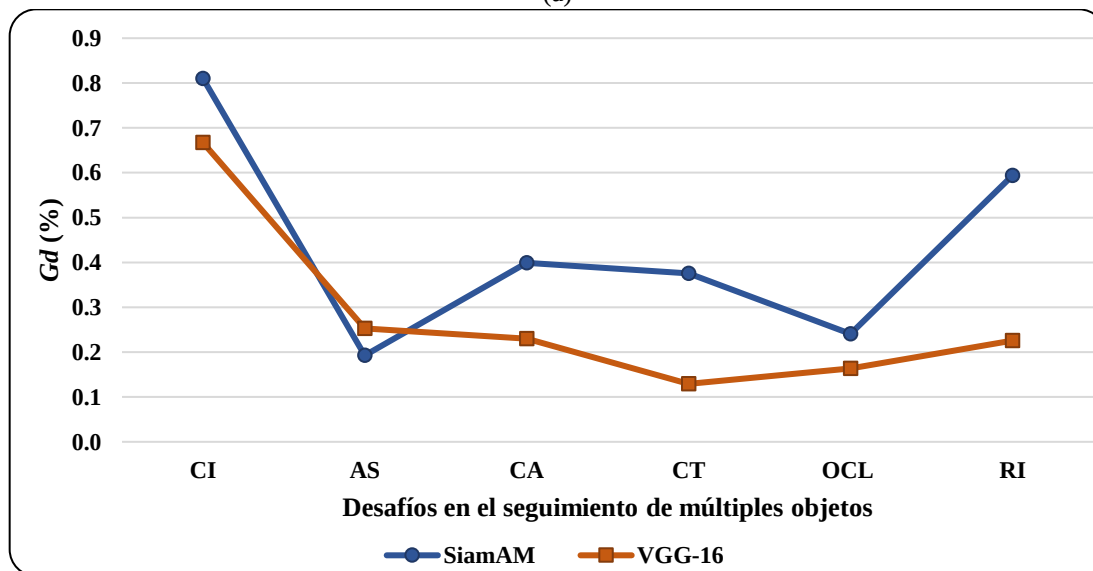
		Desafíos			
		CI	CF	CT	RI
Modelos entrenados con 1000 épocas	1	0.8219	0.6604	0.5617	0.6096
	2	0.7034	0.5651	0.5532	0.5211
	3	0.8062	0.5683	0.4864	0.6591
	4	0.8082	0.6732	0.5806	0.6217
	5	0.8265	0.6084	0.6269	0.5876
	6	0.8165	0.6995	0.4650	0.5679
	7	0.6892	0.5548	0.5986	0.8130
	8	0.8060	0.5662	0.5194	0.6997
	9	0.6282	0.4737	0.6043	0.8108
	10	0.7950	0.6043	0.5911	0.6073

2.4.3 Comparación de SiamAM con VGG-16

Para validar el desempeño del modelo propuesto SiamAM, este se comparó con la CNN VGG-16 por ser la que mejor desempeño obtuvo al utilizar la función de similitud ρ en el experimento del Gd de la sección 2.3.5. En la Figura 2.16 (a) se muestra el Gd utilizando ρ en un MA actualizado, mientras que en la Figura 2.16 (b) se muestra Gd en un MA no actualizado.



(a)



(b)

Figura 2.16. Comparación entre el Gd de SiamAM y la CNN VGG-16 utilizando ρ , en un MA actualizado (a) y un MA no actualizado (b).

En los resultados mostrados en la Figura 2.16 (a) se observa que SiamAM supera los resultados obtenidos por VGG-16 en los desafíos de CI, CA, CT, OCL y RI, pero se ve superado en el desafío de AS. Por su parte, en la Figura 2.16 (b) se observa que SiamAM supera los resultados obtenidos por VGG-16 con una amplia ventaja en los desafíos de CA, CT y RI y es ligeramente superior en los desafíos de CI y OCL, siendo superado únicamente en el desafío de AS.

Un aspecto para destacar es que el Gd de SiamAM se mantiene alto en las 2 versiones del experimento para la mayoría de los desafíos evaluados, con excepción de los desafíos AS y OCL en los cuales se presenta una disminución en el Gd cuando se utiliza un MA no actualizado. Sin embargo, esta disminución es aún más grave en el caso de la VGG-16, en donde se observa que el Gd se reduce drásticamente en la mayoría de los desafíos cuando se utiliza un MA no actualizado. Lo cual indica que a diferencia de la VGG-16, SiamAM es capaz de modelar la apariencia de los objetivos aun y cuando estos no se encuentran en cuadros subsecuentes, presentando una mayor robustez para modelar cambios más drásticos en los objetivos. No obstante, dicha robustez limita su capacidad para distinguir entre objetivos diferentes con apariencias muy similares.

En resumen, en este capítulo se realizó un análisis de distintos modelos de CNN propuestos en la literatura, en donde se evaluó su capacidad para reidentificar objetivos de interés ante los principales desafíos que se presentan en el seguimiento de objetos. Los resultados obtenidos en dicha evaluación se tomaron como referencia para el diseño de un nuevo MA basado en una CNN en configuración siamesa, nombrado SiamAM. Dicho MA será utilizado en el algoritmo propuesto para el seguimiento de múltiples objetos, del cual se brinda una descripción más detallada en el capítulo 5.

CAPÍTULO 3

MODELO DE MOVIMIENTO (MM)

Un modelo de movimiento (MM) es aquel que con base en el comportamiento dinámico del objetivo es capaz de predecir su posición en instantes futuros. Esta característica es fundamental para el seguimiento de objetos, ya que conocer la posición de los objetivos en instantes futuros ayuda a reducir el espacio de búsqueda y aumenta la exactitud del modelo de apariencia.

En la literatura, la combinación de un MM y un MA es una práctica muy frecuente que a menudo produce una sinergia, obteniendo el mejor desempeño cuando ambos modelos trabajan en conjunto. Por tal motivo, en este capítulo se da a conocer la metodología empleada para el desarrollo del MM utilizado en el algoritmo propuesto para el seguimiento de múltiples objetos.

3.1 Revisión de los principales métodos utilizados en los modelos de movimiento

Los modelos de movimiento utilizados para el seguimiento de múltiples objetos se pueden clasificar en modelos lineales y no lineales.

1. Modelo de movimiento lineal: Es aquel que siguen un movimiento lineal con velocidad constante a través de cada cuadro de la secuencia de video, este enfoque se caracteriza por ser el más simple, en el cual comúnmente se utilizan modelos Bayesianos como lo son los filtros de partículas y los filtros de Kalman [37]. Este tipo de modelos tienen la limitante de necesitar de un sistema de captura de al menos 25 FPS, ya que de esta manera se garantiza que la diferencia en la posición del objetivo en dos cuadros de video consecutivos no sea tan grande y sea posible asumir que el objeto se desplaza de manera lineal con una velocidad constante.
2. Modelo de movimiento no lineal: Es aquel capaz de modelar el movimiento de objetivos que se desplazan de una manera no lineal. Este tipo de modelos proporcionan una predicción más precisa que los modelos lineales ante la presencia de desafíos como oclusiones o cámaras en movimiento. Sin embargo, presentan la desventaja de ser computacionalmente más costosos y con un desarrollo más complejo y elaborado, ya que recientemente este tipo de métodos suelen estar

basados en redes neuronales recurrentes (RNN), específicamente en redes *Long Short-Term Memory* (LSTM).

En la Tabla 3.1 se presenta una revisión que consta de 30 trabajos publicados recientemente, en donde se ha utilizado un MM enfocado al seguimiento de múltiples objetos. A su vez en la Figura 3.1 se ilustra de manera gráfica el porcentaje en que ha sido utilizado cada método con base en los trabajos consultados.

Tabla 3.1. Revisión de los métodos más utilizados para el desarrollo de modelos de movimiento.

Nombre del artículo	Método	Año	Ref.
Automated Individual Pig Localization, Tracking and Behavior Metric Extraction Using Deep Learning	Filtro de Kalman	2019	[85]
Online Multi-Object Tracking with Historical Appearance Matching and Scene Adaptive Detection Filtering	Filtro de Kalman	2019	[86]
Deep Alignment Network Based Multi-Person Tracking with Occlusion and Motion Reasoning	Filtro de Kalman	2019	[87]
Real-time visual tracking and identification for a team of homogeneous humanoid robots	Filtro de Kalman	2017	[88]
Enhanced Association with Supervoxels in Multiple Hypothesis Tracking	Filtro de Kalman	2019	[89]
Fast Online Multi-Pedestrian Tracking via Integrating Motion Model and Deep Appearance Model	Filtro de Kalman	2019	[72]
A New Adaptive Robust Unscented Kalman Filter for Improving the Accuracy of Target Tracking	Filtro de Kalman	2019	[90]
Motion Guided Siamese Trackers for Visual Tracking	Filtro de Kalman	2019	[91]
Multi-Frame Track-Before-Detect Algorithm for Maneuvering Target Tracking	Filtro de Kalman	2020	[92]
Multi-Target State Estimation Using Interactive Kalman Filter for Multi-Vehicle Tracking	Filtro de Kalman	2020	[93]
Real-Time Multiple People Tracking with Deeply Learned Candidate Selection and Person Re-Identification	Filtro de Kalman	2018	[29]
Online Multi-Object Tracking with Visual and Radar Features	Filtro de Kalman	2020	[94]
Online Multi-Object Tracking with GMPHD Filter and Occlusion Group Management	Filtro de Kalman	2019	[95]
Long-Term Tracking with Deep Tracklet Association	RNN (LSTM)	2020	[96]
On human motion prediction using recurrent neural networks	RNN (LSTM)	2017	[97]
Multi-object tracking with neural gating using bilinear LSTM	RNN (LSTM)	2018	[54]
Online Multi-Target Tracking Using Recurrent Neural Networks	RNN (LSTM)	2017	[98]
Structural-RNN: Deep Learning on Spatio-Temporal Graphs	RNN (LSTM)	2016	[99]

Tabla 3.1. Continuación.

Nombre del artículo	Método	Año	Ref.
Social LSTM: Human trajectory prediction in crowded spaces	RNN (LSTM)	2016	[100]
Deep Recurrent Neural Network for Multi-target Filtering	RNN (LSTM)	2019	[101]
Online Multi-Object Tracking Based on Feature Representation and Bayesian Filtering Within a Deep Learning Architecture	RNN (LSTM)	2019	[102]
Online Visual Robot Tracking and Identification using Deep LSTM Networks	RNN (LSTM)	2017	[103]
Multi-Target Tracking with Trajectory Prediction and Re-Identification	RNN (LSTM)	2019	[104]
Deep Object Tracking on Dynamic Occupancy Grid Maps Using RNNs	RNN (LSTM)	2018	[105]
Convolutional Recurrent Predictor: Implicit Representation for Multi-Target Filtering and Tracking	RNN (LSTM)	2019	[106]
Tracking the Untrackable: Learning to Track Multiple Cues with Long-Term Dependencies	RNN (LSTM)	2017	[55]
Long-Term Action Dependence-Based Hierarchical Deep Association for Multi-Athlete Tracking in Sports Videos	RNN (LSTM)	2020	[107]
Target Tracking System Using Multiple Cameras and Bayesian Estimation	Filtro de Particulas	2017	[40]
Particle PHD Filter Based Multiple Human Tracking Using Online Group-Structured Dictionary Learning	Filtro de Particulas	2018	[37]
Research on Robust Visual Tracker Based on Multi-Cue Correlation Particle Filters	Filtro de Particulas	2020	[108]

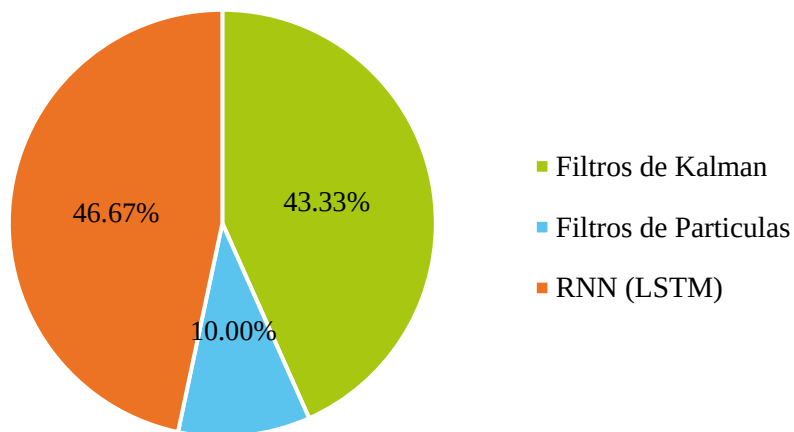


Figura 3.1. Porcentaje en que sea a utilizado cada método con base en los trabajos consultados.

En esta revisión se observa que la mayoría de los MM están basados en redes LSTM y filtros de Kalman, con una ligera inclinación en el uso de redes LSTM. Sin embargo, dicha diferencia

es de apenas el 3% y no es tan evidente que exista una preferencia entre un método u otro. Por tal motivo, en Tabla 3.2 se presenta una comparación con mayor detalle entre los filtros de Kalman y las redes LSTM [103], para seleccionar el método que mejor se adapte a las necesidades en esta tesis.

Tabla 3.2. Comparativa de los MM basados en filtros de Kalman y RNN (LSTM).

	Ventajas	Desventajas
RNN (LSTM)	• Es capaz de modelar movimientos no lineales. • Permite lidiar adecuadamente con problemas de oclusiones prolongadas. • Proporciona una predicción con un alto grado de precisión. • Es capaz de extraer patrones de movimiento robustos a largo plazo.	• No cuenta con un proceso de diseño bien definido. • Se necesita una gran cantidad de datos de entrenamiento para obtener una predicción confiable. • Presenta un entrenamiento muy tardío. • Tiene un alto costo computacional y baja velocidad de respuesta.
Filtros de Kalman	• Cuenta con un proceso de diseño e implementación muy simple. • No necesita de una etapa de entrenamiento. • Tiene un bajo costo computacional y alta velocidad de respuesta.	• La precisión disminuye, si el filtro no se actualiza constantemente. • Presenta predicciones erróneas ante movimientos no lineales. • No es capaz de lidiar adecuadamente con oclusiones prolongadas.

Con base en la información presentada en la Tabla 3.2, en esta tesis se decide desarrollar un MM basado en filtros de Kalman, esto debido a la facilidad en su desarrollo e implementación, así como su bajo costo computacional y alta velocidad de respuesta. Respecto a las redes LSTM estas pueden ser exploradas en trabajos futuros ya que son una mejor opción en términos de precisión y robustez. Sin embargo, su proceso de desarrollo e implementación puede llegar a ser muy extenso y tardío.

3.2 Filtro de Kalman

En el filtro de Kalman (FK) existen diversas variantes, el uso de una u otra variante depende de la naturaleza del proceso que se quiere estimar. En esta tesis se utiliza el FK sin modificaciones, también conocido como *Linear Quadratic Estimation*. Ya que como menciona

García [43] no es necesario utilizar versiones como EKF (*Extended Kalman Filter*) o UKF (*Unscented Kalman Filter*) debido a que todas las secuencias de video que serán analizadas provienen de sistemas de captura con una velocidad superior a los 25 FPS, lo cual permite asumir que los objetos se desplazan de manera lineal con una velocidad constante.

El FK consiste en un conjunto de ecuaciones matemáticas que proveen una solución recursiva óptima, por el método de mínimos cuadrados. La meta de dicha solución consiste en calcular un estimador lineal del estado de un sistema en el instante t con base en la información disponible en el instante $t-1$, y actualizar dichas estimaciones con la información adicional disponible en el instante t .

El FK es el principal algoritmo para estimar sistemas dinámicos representados en espacio de estados, en esta representación el sistema es descrito por un conjunto de variables denominadas variables de estado. En donde el estado hace referencia a la información del sistema en un determinado instante de tiempo, esta información debe permitir la inferencia del comportamiento pasado del sistema con el objetivo de predecir su comportamiento futuro.

3.2.1 Algoritmo del Filtro de Kalman

El algoritmo del FK puede definirse como un algoritmo de pronóstico-corrección, en donde el FK funciona por medio de un mecanismo de proyección y corrección, al pronosticar el estado del sistema y su incertidumbre para posteriormente corregir dicha proyección utilizando la nueva información disponible (mediciones en el instante t). Por lo anterior el FK se suele dividir en dos grandes etapas: predicción y corrección, las cuales interactúan de manera recursiva como se muestra en la Figura 3.2.

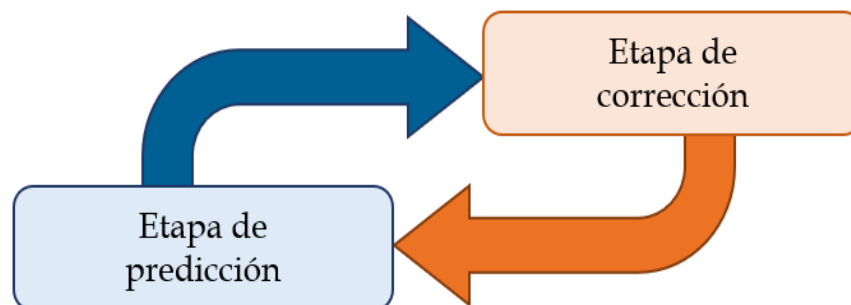


Figura 3.2. El ciclo del filtro de Kalman.

La etapa de predicción consiste en realizar una proyección del estado del sistema X_t^* y de la covarianza del error P_t^* , utilizando las ecuaciones (3.1) y (3.2) respectivamente.

$$X_t^* = AX_{t-1} \quad (3.1)$$

$$P_t^* = AP_{t-1}A^T + Q \quad (3.2)$$

En donde X_{t-1} es el vector de estado del sistema en un instante previo tal que $X \in \mathbb{R}^n$, la matriz de transición de estado, A , es una matriz de dimensiones $n \times n$ que relaciona el estado en el instante previo con el estado en el momento t . Mientras que Q representa la matriz de covarianza del ruido del proceso que trata de estimar el estado. Estos elementos se definirán con mayor detalle en la sección 3.4.

La etapa de corrección consiste en generar un pronóstico mejorado del estado, de tal manera que el error sea minimizado estadísticamente. La primera tarea durante la etapa de corrección es el cálculo de la ganancia de Kalman, K_t , ganancia que se obtiene de tal forma que minimice la covarianza del error de la nueva estimación del estado, utilizando la Ec (3.3). El siguiente paso es obtener la medición del proceso Z_t tal que $Z \in \mathbb{R}^m$, en donde Z contiene todas las variables de estado que son medibles. Después de obtener dicha medición se genera una nueva estimación del estado que incorpora la nueva información medida, utilizando la Ec (3.4). El paso final es obtener una nueva estimación de la covarianza del error mediante la Ec (3.5).

$$K_t = P_t^* H^T (HP_t^* H^T + R)^{-1} \quad (3.3)$$

$$X_t = X_t^* + K_t (Z_t - HX_t^*) \quad (3.4)$$

$$P_t = (I - K_t H) P_t^* \quad (3.5)$$

En donde la matriz de medición, H , es una matriz de dimensiones $m \times n$ que relaciona el estado con la medición Z_t , mientras que R representa la matriz de covarianza del ruido de la medición e I hace referencia a la matriz identidad. Estos elementos se definirán con mayor detalle en la sección 3.4.

El algoritmo se repite tomando como punto de partida las nuevas estimaciones del estado y

de la covarianza del error. Esta naturaleza recursiva es una de las características más llamativas del FK y una de las razones por las que es muy utilizado en una gran variedad de aplicaciones.

A continuación, en la Figura 3.3 se presenta un diagrama general del funcionamiento del FK, combinando la Figura 3.2 y sus respectivas ecuaciones. En donde se observa que el FK parte de las estimaciones iniciales X_{t-1} y P_{t-1} . En la práctica la estimación inicial de X_{t-1} se obtiene mediante la medición del proceso que se quiere estimar, siendo X_{t-1} la primera medición del proceso. Por su parte P_{t-1} se fija de acuerdo con la certeza que se tiene sobre la medición de X_{t-1} , en donde valores bajos de P_{t-1} representan una incertidumbre baja y valores altos representan una incertidumbre alta en la medición de X_{t-1} . En la sección 2.4 se muestra un ejemplo de P_{t-1} indicando una alta y baja incertidumbre.

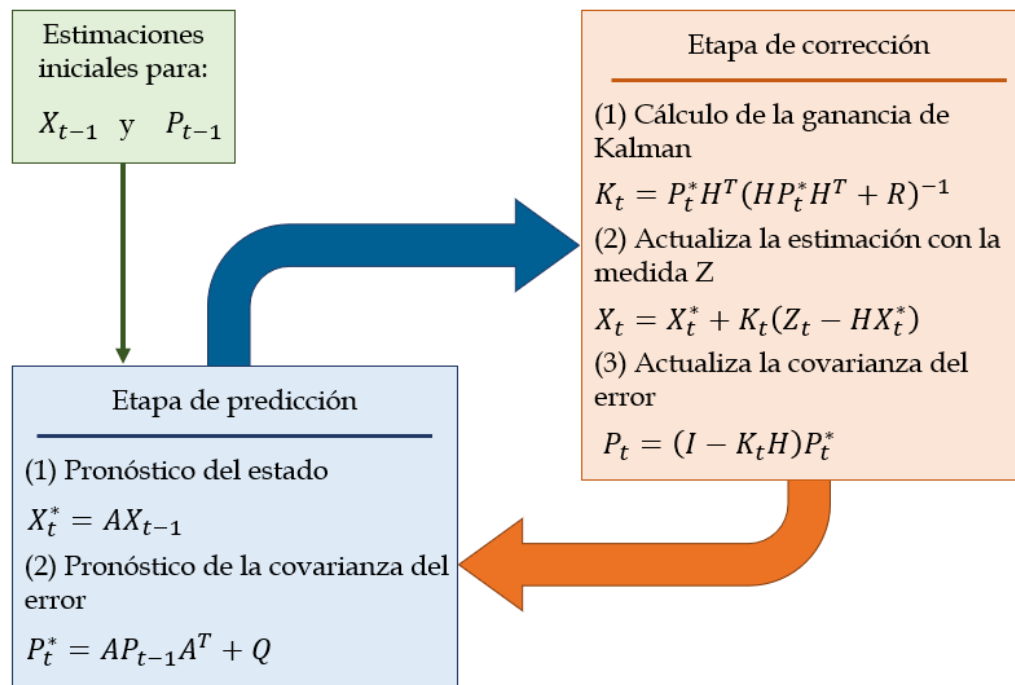


Figura 3.3. Visión completa del FK.

3.3 El Filtro de Kalman enfocado al seguimiento de múltiples objetos

En el seguimiento de múltiples objetos el FK se utiliza para aproximar la posición de un objetivo, dicha posición generalmente suele estar representada por un cuadro delimitador que encierra al objetivo. La función del FK consiste predecir el cuadro delimitador del objetivo en

el siguiente cuadro de la secuencia de video, considerando la información del objetivo en instantes anteriores. En la Figura 3.4 se muestra una representación de la predicción de la posición de un objetivo, mientras que en la Figura 3.5 se muestra una comparación real entre las predicciones del FK y la trayectoria de un objetivo.

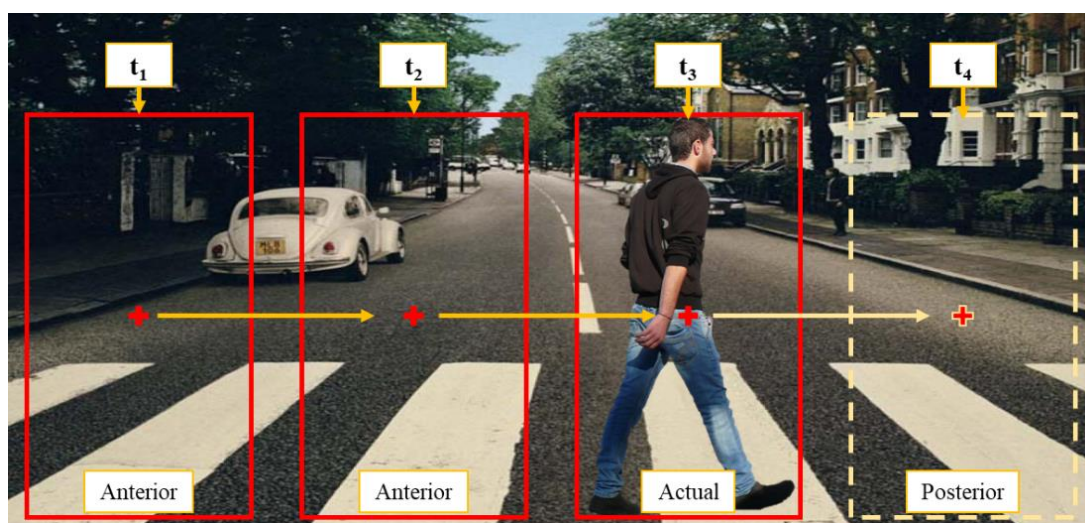


Figura 3.4. Predicción de la posición de un objetivo en un instante futuro.

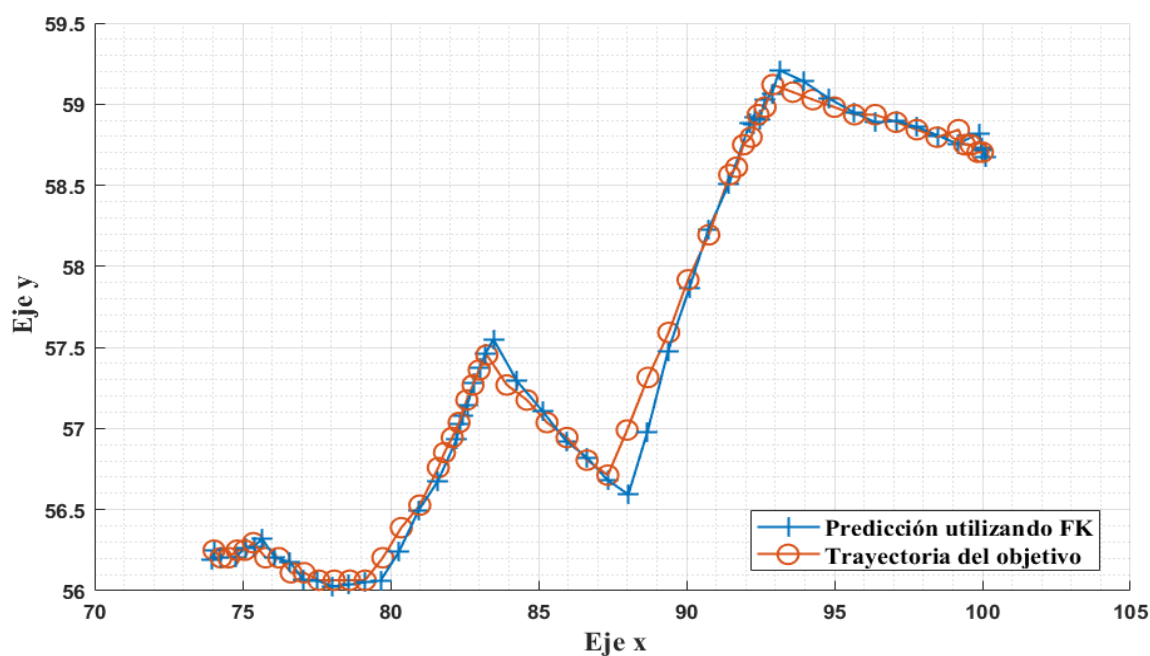


Figura 3.5. Comparación entre las predicciones del FK y la trayectoria real del objetivo, utilizando el centroide del objetivo.

En la Figura 3.5 se observa una buena aproximación de la trayectoria del objetivo al utilizar el FK, aunque existen algunos puntos en la gráfica en donde se presentan cambios drásticos de dirección, que provocan que el FK se desvíe y tarde algunos instantes en volver a predecir de manera acertada la posición del objetivo. De manera que el FK se adapta a la trayectoria del objetivo, aunque este no siga un comportamiento lineal, con la limitante que después de cada cambio de dirección se generan predicciones erróneas durante las primeras instancias, adaptándose a la nueva trayectoria después de recibir actualizaciones de manera constante sobre la posición actual del objetivo.

3.4 Configuración del Filtro de Kalman

La configuración de los parámetros del FK que se utilizan en esta tesis para el desarrollo del MM son los propuestos por Bewley [109]. Dichos parámetros se describen en la Tabla 3.3.

Tabla 3.3. Configuración del FK enfocado al seguimiento de múltiples objetos.

$X = [x, y, a, r, x', y', a']^T$	Vector de estado X: El vector de estado está conformado por las coordenadas del centroide del objetivo (x, y), el área del cuadro delimitador ($a = w * h$), la relación entre el ancho y la altura del cuadro delimitador ($r = w/h$), la velocidad en x e y (x', y') y la variación del área del cuadro delimitador (a').
$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$	Matriz de transición de estado A: Esta matriz define un sistema lineal de velocidad constante, en donde cada columna se asocia con un elemento del vector de estado.
$P = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1E4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1E4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1E4 \end{bmatrix}$	Matriz de covarianza del error inicial P: Esta matriz representa la incertidumbre que se tiene sobre las variables del vector de estado. Indicando una incertidumbre alta en x', y' y a' que corresponden a las variables del vector de estado que no son medibles y una incertidumbre baja en las variables medibles del vector de estado.

Tabla 3.3. Continuación.

$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1E-2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1E-2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1E-4 \end{bmatrix}$	Matriz Q: Esta matriz representa la covarianza del ruido del proceso que se quiere estimar. En este caso el proceso corresponde al movimiento de los objetos, en donde definir esta matriz resulta realmente complicado por lo que en la práctica esta matriz se suele ajustar de manera heurística.
$Z = [x, y, a, r]^T$	Vector de medición Z: El vector de medición está conformado por las variables del vector de estado que son medibles.
$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$	Matriz de medición H: Esta matriz indica las variables del vector de estado que son medibles.
$R = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 & 0 \end{bmatrix}$	Matriz R: Esta matriz representa la covarianza del ruido de la medición. Este ruido es el error que se comete al medir, es decir el error que cometen los sensores. En el seguimiento de objetos se utiliza como sensor un algoritmo de detección de objetos.

Los parámetros mostrados en la Tabla 3.3 fueron seleccionados debido a que permiten modelar la forma completa del cuadro delimitador en instantes futuros. Es decir, aparte de modelar la nueva posición del centroide del objetivo, también se toma en cuenta que su altura y grosor pueden cambiar con el tiempo, por lo que en el modelo se considera la forma en la que el área del cuadro delimitador varía en el tiempo. En algunas otras configuraciones del FK el grosor y la altura no se suelen tomar en cuenta y simplemente se consideran constantes, pero existen diversas situaciones que pueden hacer que el tamaño del objeto aumente o disminuya, por lo que modelar estas variaciones de tamaño incrementa la precisión de las predicciones del FK.

En resumen, en este capítulo se realizó un análisis de los MM basados en FK y redes LSTM para evaluar sus ventajas y desventajas. Con base en este análisis se optó por desarrollar un MM basado en un FK por su bajo costo computacional y alta velocidad de respuesta. Dicho MM será utilizado en el algoritmo propuesto para el seguimiento de múltiples objetos, del cual se brinda una descripción más detallada en el capítulo 5.

CAPÍTULO 4

ASOCIACIÓN DE DATOS

Un algoritmo de optimización para la asociación de datos es de gran importancia para el seguimiento de múltiples objetos, principalmente en aquellos métodos que trabajan bajo el enfoque *tracking-by-detection*, ya que estos métodos generan las trayectorias de los objetivos mediante la asociación de detecciones proporcionadas por un detector de objetos.

En este capítulo se describe el algoritmo de optimización utilizado para la asociación de datos en el algoritmo propuesto para el seguimiento de múltiples objetos.

4.1 Problema de asociación

Un problema de asociación es aquel en donde hay que decidir cómo repartir una serie de destinos entre una serie de orígenes. Un origen hace referencia a individuos, equipos o cualquier otro elemento al que hay que asociarlo con una tarea o un trabajo denominado como destino [110]. En este tipo de problemas la asociación entre un origen y un destino tiene un cierto costo o beneficio, el cual es un valor que puede ser real positivo, negativo o cero como se muestra en la Figura 4.1 (a). Cuando dicho valor es mejor entre mayor sea su valor se considera un beneficio, en caso contrario se trata de un costo. El objetivo en este tipo de problemas es encontrar la solución óptima, que maximiza el beneficio, Figura 4.1 (b), o minimiza el costo.

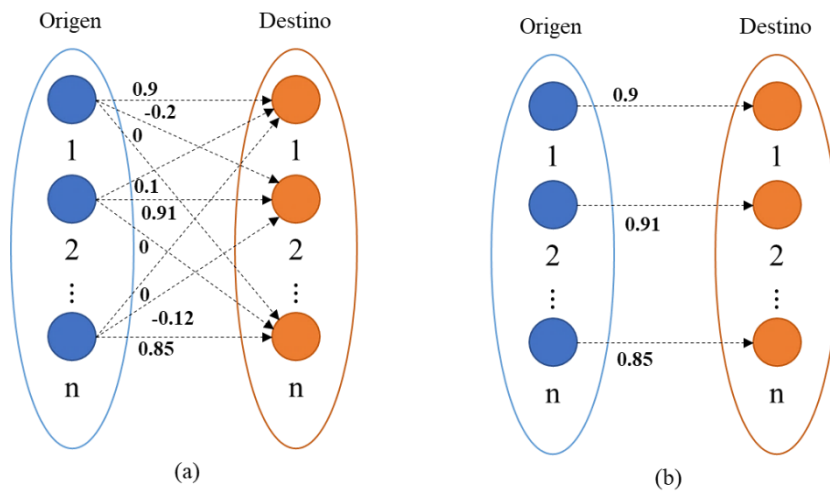


Figura 4.1. Problema de asociación entre orígenes y destinos (a), solución que maximiza el beneficio (b).

4.1.1 Seguimiento de múltiples objetos como un problema de asociación

El seguimiento de múltiples objetos bajo el enfoque *tracking-by-detection* se puede formular como un problema de asociación, en donde un origen hace referencia a las detecciones de los objetivos en un cuadro t , y un destino se refiere a las detecciones de los objetivos en un cuadro $t+1$. Siendo la similitud entre dichas detecciones en los cuadros t y $t+1$ el costo o beneficio. De tal manera que encontrar la solución que maximiza el beneficio o minimiza el costo, permite asociar las detecciones que corresponden al mismo objetivo en los cuadros t y $t+1$, como se muestra en la Figura 4.2.

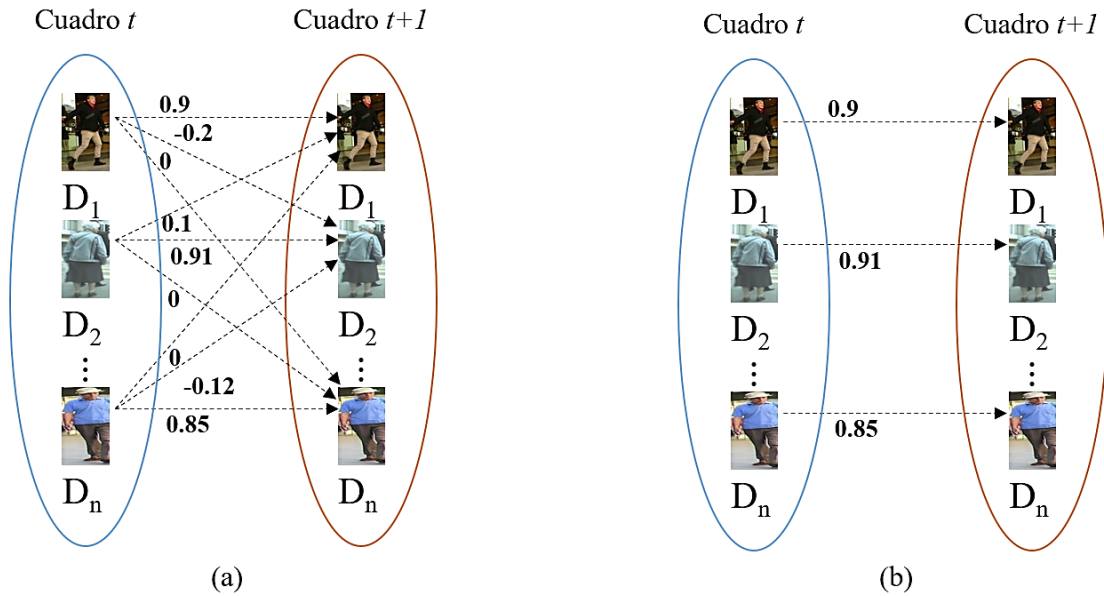


Figura 4.2. Problema de asociación entre detecciones en los cuadros t y $t+1$ (a), solución óptima (b).

La solución al problema de asociación entre los cuadros t y $t+1$ durante el transcurso de toda la secuencia de video, implícitamente permite realizar el seguimiento de múltiples objetos, generando las trayectorias de los objetivos mediante la asociación de las detecciones en los cuadros t y $t+1$. En la literatura uno de los métodos más utilizados para realizar la asociación de datos es el método Húngaro [72], el cual se describe a continuación.

4.2 Método Húngaro

El método Húngaro es utilizado para resolver problemas de asociación de forma eficiente computacionalmente. En la Figura 4.3 se muestra el esquema general del método Húngaro, el

cual consta de dos fases. En la primera fase se realiza un preprocesamiento de los datos y se obtiene una solución inicial, en la segunda fase dicha solución es optimizada de manera iterativa hasta encontrar la solución óptima.

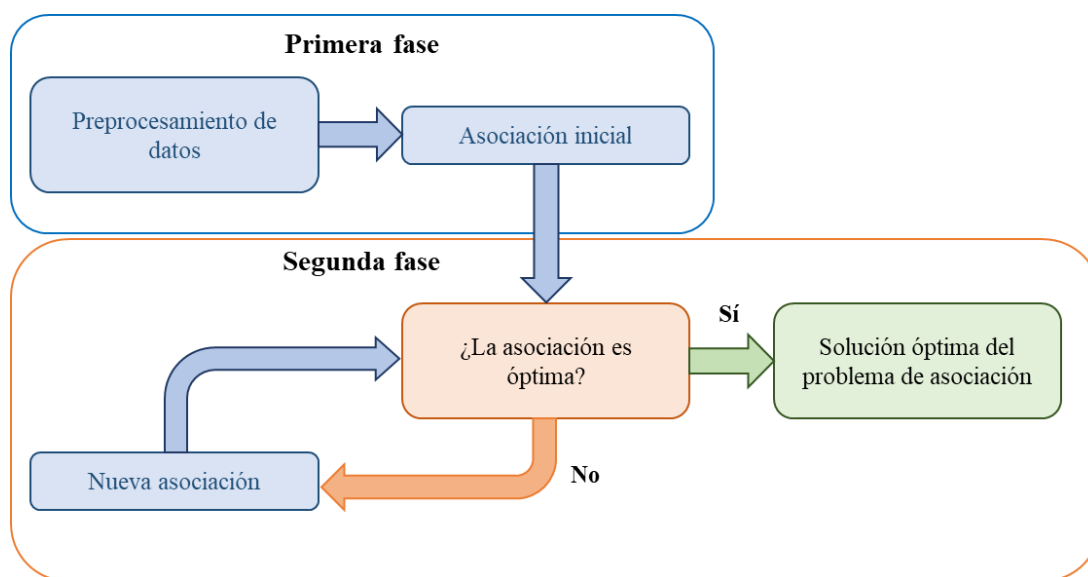


Figura 4.3. Esquema general del funcionamiento del método Húngaro.

El método Húngaro trabaja sobre una matriz de costo Cs_{ij} , donde las filas i corresponden a los orígenes y las columnas j a los destinos. Los elementos de la matriz Cs_{ij} corresponden al costo de asociar el origen i al destino j , como se muestra en la Figura 4.4.

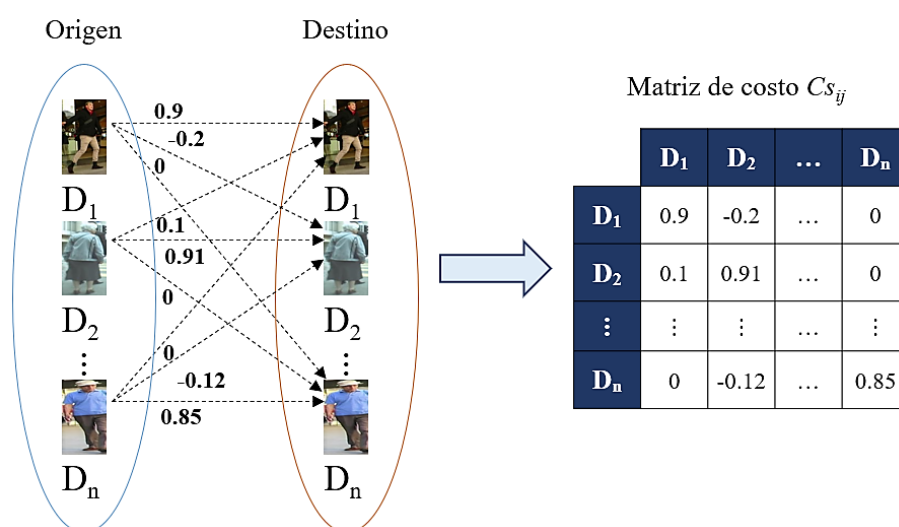


Figura 4.4. Generación de la matriz de costo Cs_{ij} .

La matriz Cs_{ij} , debe cumplir con los siguientes 4 requisitos:

1. Cs_{ij} debe generarse a partir de un problema de minimización, es decir todos los elementos de Cs_{ij} deben corresponder al costo de asociar un origen i con un destino j . En caso de tratarse de un problema de maximización se debe multiplicar por -1 todos los elementos de Cs_{ij} .
2. La asociación se debe realizar uno a uno, es decir que a cada destino le corresponde un único origen y a cada origen le corresponde un único destino.
3. Todos los elementos de Cs_{ij} deben ser positivos o cero, en caso de no cumplirse esta condición se debe tomar el valor más negativo de Cs_{ij} y cambiarle el signo, posteriormente dicho valor se sumará a todos los elementos de Cs_{ij} .
4. Cs_{ij} debe ser una matriz cuadrada, es decir con el mismo número de filas y columnas. Si esto no se cumple se deberá agregar filas o columnas ficticias hasta obtener una matriz cuadrada. Los elementos ficticios que se agreguen deberán tomar un valor aproximadamente 10 veces más grande en comparación con el valor más alto de Cs_{ij} . De esta forma estos elementos ficticios no afectaran la asociación de datos y permitirán cumplir con el requisito de que Cs_{ij} sea una matriz cuadrada.

En la Figura 4.5, se muestra un ejemplo del preprocesamiento que se realiza a Cs_{ij} para que cumpla con los requisitos previos, por simplicidad solo se muestran los valores de Cs_{ij} omitiendo sus encabezados.

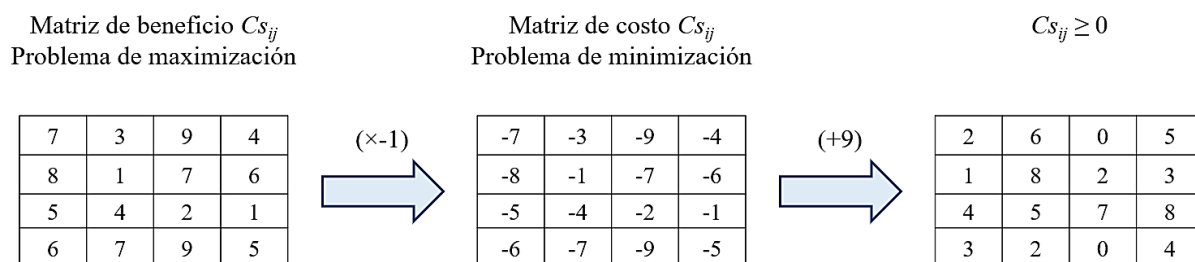


Figura 4.5. Preprocesamiento de la matriz Cs_{ij} .

Una vez que Cs_{ij} cumple con los 4 requisitos se procede a calcular la asociación inicial, para esto se realizan las siguientes 2 operaciones.

1. Restar a todos los elementos de la misma fila el valor mínimo de su respectiva fila.
2. Restar a todos los elementos de la misma columna el valor mínimo de su respectiva columna.

En la Figura 4.6 se muestra el resultado al aplicar las dos operaciones previas a la matriz de costo de la Figura 4.5. A su vez, en la Figura 4.6 se observa que la segunda matriz reducida cuenta con al menos un cero en cada fila y un cero en cada columna. Esto es importante debido a que el método Húngaro utiliza los ceros para realizar la asociación entre orígenes y destinos como se explica a continuación.

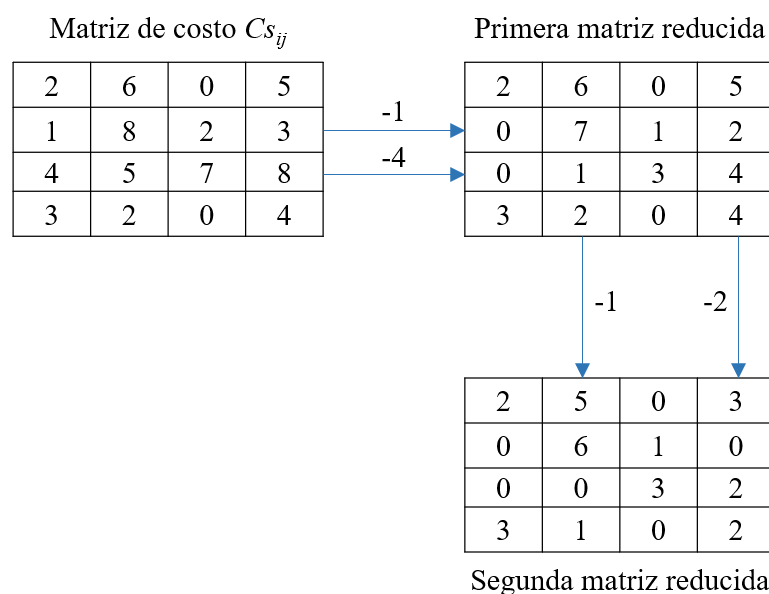


Figura 4.6. Proceso para obtener la matriz reducida.

La asociación inicial se realiza repitiendo el algoritmo de la Figura 4.7, este proceso se repite hasta que todos los ceros de la segunda matriz reducida han sido seleccionados o tachados. Un cero recuadrado [0], hace referencia a un cero seleccionado e indica que el origen de la fila i ha sido asociado al destino de la columna j , mientras que un cero tachado $\emptyset$, indica que el origen o destino fue previamente asociado y no es posible volverlo a asociar, puesto que la asociación debe ser uno a uno. En Figura 4.8 (a) se muestra el resultado de aplicar el algoritmo de asociación para obtener la asociación inicial, mientras en Figura 4.8 (b) se muestra la interpretación de la asociación inicial entre orígenes y destinos.

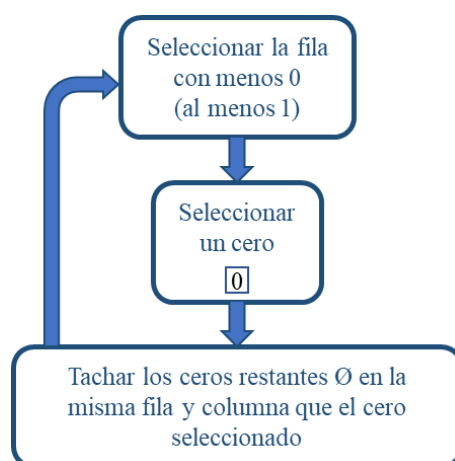


Figura 4.7. Algoritmo de asociación.

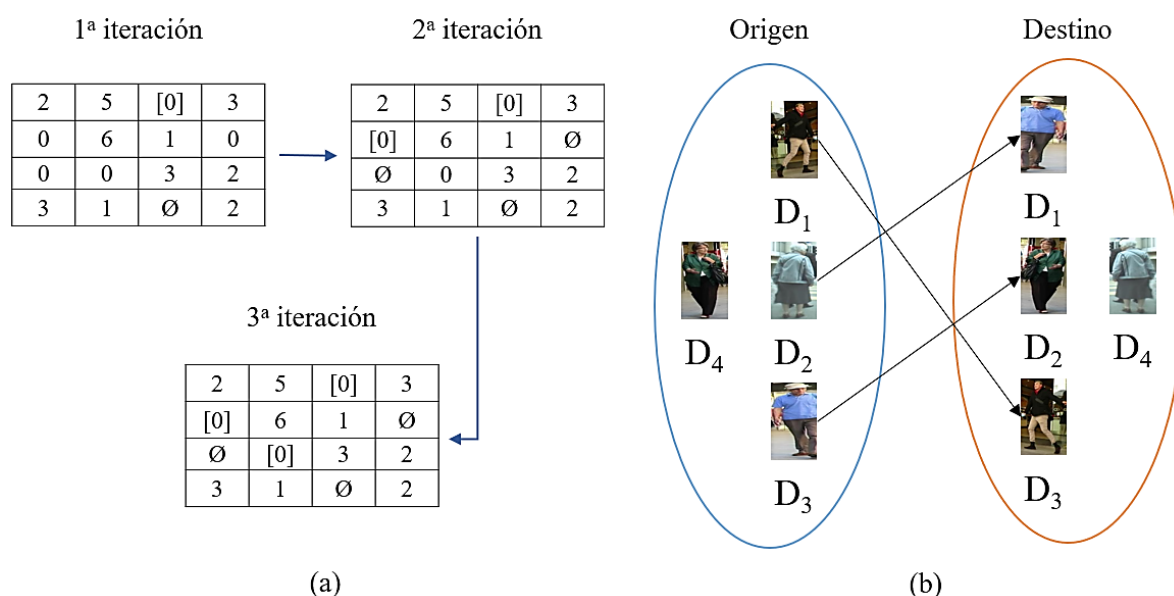


Figura 4.8. Resultado del algoritmo de asociación para obtener la asociación inicial (a), interpretación de la asociación inicial (b).

Una vez generada la asociación entre orígenes y destinos, el siguiente paso es comprobar si la asociación es óptima. Se dice que la asociación es óptima cuando cada origen ha sido asociado a un destino, lo cual se logra cuando existe un cero recuadrado en cada fila y en cada columna. En la Figura 4.8 (a) esta condición no se cumple, puesto que el origen 4 y el destino 4 no fueron asociados, entonces se considera que la asociación inicial no es óptima y se procede a realizar una nueva asociación. Para obtener la nueva matriz de asociación se utilizan los siguientes 2 pasos.

1. Seleccionar el menor número de filas y columnas que cubran todos los ceros de la matriz de asignación previa.
2. Identificar el valor menor dentro de los elementos no seleccionados en el paso 1. Dicho valor se resta a los elementos no seleccionados y se suma a los elementos de las intersecciones de filas y columnas seleccionadas.

En la Figura 4.9 se muestra la nueva matriz de asociación con sus nuevos valores, dicha matriz se utiliza para realizar una nueva asociación entre orígenes y destinos, como se muestra en la Figura 4.10, empleando el mismo algoritmo de asociación descrito en la Figura 4.7.

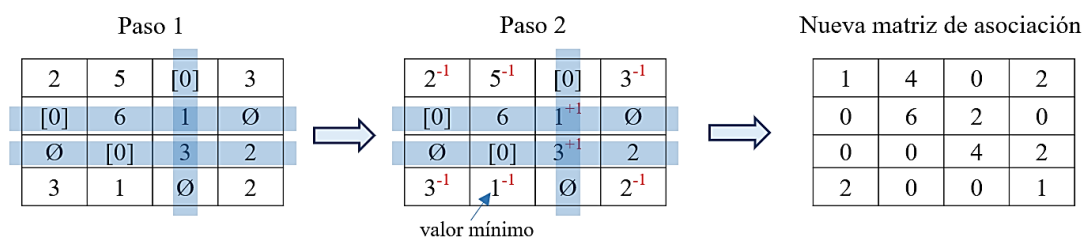


Figura 4.9. Pasos para generar la nueva matriz de asociación.

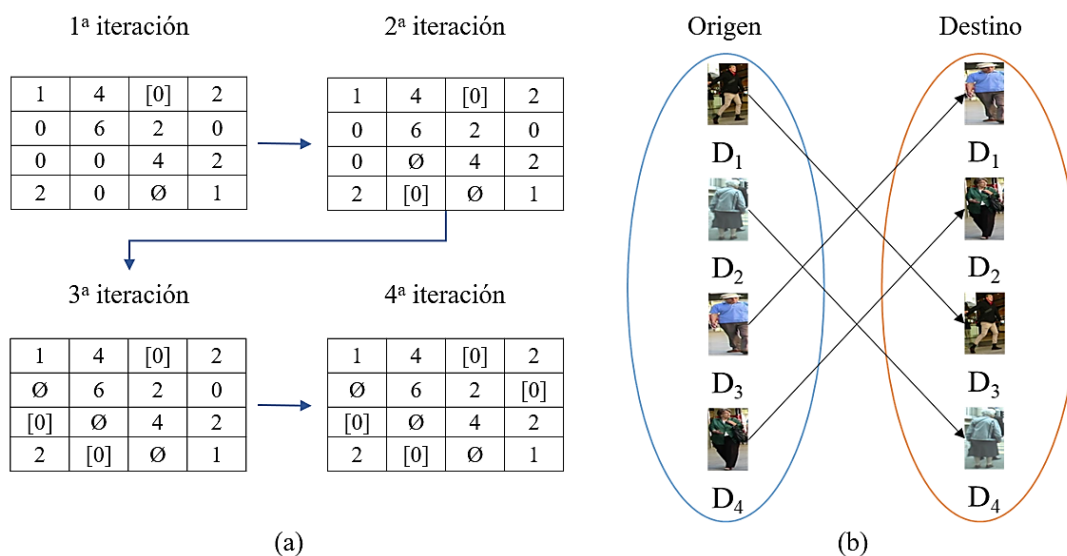


Figura 4.10. Resultado del algoritmo de asociación (a), interpretación de la nueva asociación (b).

En esta nueva asociación, mostrada en la Figura 4.10, se observa que no existe ningún origen ni destino que no haya sido asociado, por lo que en esta ocasión la asociación es óptima y se da por concluido el problema de asociación.

Respecto a la asociación final que entrega el método Húngaro ésta siempre será óptima, pero no hay garantía de que sea correcta. Puesto que el método Húngaro considera que a cada origen le corresponde un destino, aunque existen problemas en los detectores de objetos como falsos positivos y falsos negativos que podrían generar asociaciones incorrectas. Como se ilustra en la Figura 4.11 (a), en donde se observa que la detección D_3 en el cuadro t no existe en el cuadro $t+1$, en su lugar se encuentra un falso positivo. Después de aplicar el método Húngaro para este problema, obtendríamos la asociación que se muestra en la Figura 4.11 (b), donde D_3 se asocia con el falso positivo. Para evitar este tipo de asociaciones erróneas ocasionadas por errores en el detector de objetos, comúnmente se utiliza un umbral de afinidad (Th_{Af}). De manera que si la similitud entre orígenes y destinos no supera dicho umbral la asociación entre ellos no se realiza [109].

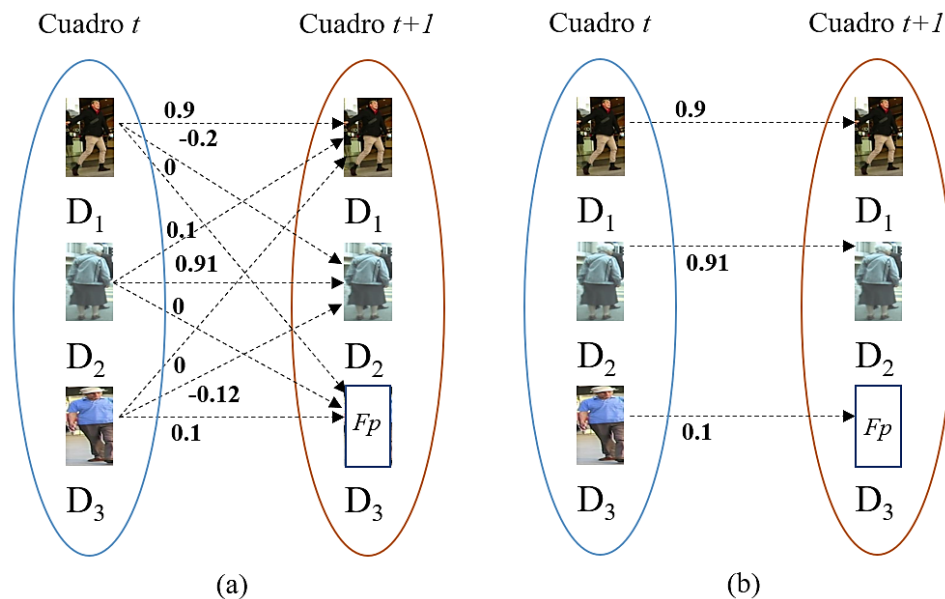


Figura 4.11. Problema de asociación entre detecciones considerando errores en el detector de objetos (a), solución óptima (b).

En resumen, en este capítulo se mostró el procedimiento para formular el seguimiento de múltiples objetos como un problema de asociación. A su vez, se explicó el funcionamiento del método Húngaro, uno de los principales métodos para resolver problemas de asociación. Dicho método será utilizado en el algoritmo propuesto para el seguimiento de múltiples objetos, del cual se brinda una descripción más detallada en el capítulo 5.

CAPÍTULO 5

SEGUIMIENTO DE MÚLTIPLES OBJETOS

En capítulos anteriores se brindó información sobre el diseño y desarrollo de distintos elementos, tales como: el modelo de apariencia, de movimiento y el algoritmo de asociación de datos. En este capítulo se describe la manera en que se integran dichos elementos para conformar cada una de las distintas versiones del algoritmo de seguimiento de múltiples objetos. A su vez se da a conocer el detector de objetos y la base de datos que se utilizaron durante el desarrollo y evaluación de las distintas versiones del algoritmo propuesto.

5.1 Base de datos

Las secuencias de video utilizadas para evaluar el desempeño del algoritmo de seguimiento fueron obtenidas de las bases de datos MOT16 y MOT17, ya que ambas comparten las mismas secuencias de video [83]. Cada base de datos cuenta con un total de 14 secuencias, de las cuales 7 son utilizadas para entrenamiento y 7 para pruebas, en donde los objetos de interés son prioritariamente personas en movimiento. Los escenarios de dichas secuencias son totalmente diferentes entre sí y se pueden clasificar dentro de las siguientes 3 categorías:

1. Cámaras estáticas o en movimiento: Las escenas pueden ser grabadas por una cámara en una posición fija o por una cámara acoplada a una persona o vehículo en movimiento.
2. Punto de vista: Las cámaras pueden grabar una escena desde tres posiciones diferentes, ya sea desde una posición con gran altura, una posición media o una posición baja.
3. Condiciones: Las condiciones en las cuales fueron grabadas las escenas pueden ser a la intemperie con un clima totalmente soleado, parcialmente nublado, de noche o al interior de un establecimiento comercial con iluminación artificial.

En la Tabla 5.1 se muestra una descripción general del contenido de las secuencias de video tanto de entrenamiento como de prueba de la base de datos MOT16, mientras que en la Figura 5.1 y Figura 5.2 se muestra un cuadro de video representativo de cada una de dichas secuencias.

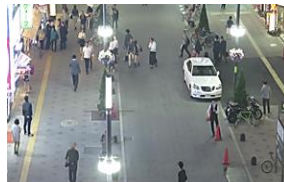
Tabla 5.1. Descripción de las secuencias de video de la base de datos MOT16 y MOT17 para entrenamiento y prueba.

Secuencias de entrenamiento								
Nombre	FPS	Resolución	Duración	Trayectorias	Detecciones	Densidad	Vista	Condiciones
MOT-02	30	1920x1080	600 (0:20)	49	17,833	29.7	media	nublado
MOT-04	30	1920x1080	1,050 (0:35)	80	47,557	45.3	alta	noche
MOT-05	14	640x480	837 (1:00)	124	6,818	8.1	media	soleado
MOT-09	30	1920x1080	525 (0:18)	25	5,257	10	baja	interior
MOT-10	30	1920x1080	654 (0:22)	54	12,318	18.8	media	noche
MOT-11	30	1920x1080	900 (0:30)	67	9,174	10.2	media	interior
MOT-13	25	1920x1080	750 (0:30)	68	11,450	15.3	alta	soleado
Total			5,316 (3:35)	512	110,407	20.8		

Secuencias de prueba								
Nombre	FPS	Resolución	Duración	Trayectorias	Detecciones	Densidad	Vista	Condiciones
MOT-01	30	1920x1080	450 (0:15)	23	6,395	14.2	media	nublado
MOT-03	30	1920x1080	1,500 (0:50)	148	104,556	69.7	alta	noche
MOT-06	14	640x480	1,194 (1:25)	217	11,538	9.7	media	soleado
MOT-07	30	1920x1080	500 (0:17)	55	16,322	32.6	media	nublado
MOT-08	30	1920x1080	625 (0:21)	63	16,737	26.8	media	soleado
MOT-12	30	1920x1080	900 (0:30)	94	8,295	9.2	media	interior
MOT-14	25	1920x1080	750 (0:30)	230	18,483	24.6	alta	soleado
Total			5,919 (4:08)	830	182,326	30.8		



MOT-02



MOT-04



MOT-05



MOT-09



MOT-10



MOT-11



MOT-13

Figura 5.1. Muestras de las secuencias de entrenamiento de la base de datos MOT16 y MOT17.



Figura 5.2. Muestras de las secuencias de prueba de la base de datos MOT16 y MOT17.

5.2 Detector de objetos

El detector de objetos es la principal diferencia entre las dos bases de datos utilizadas, esto debido a que la base de datos MOT16 únicamente proporciona un detector de objetos, mientras que en MOT17 se proporcionan tres diferentes detectores públicos. Esto es importante debido a que el detector de objetos es una parte fundamental en los métodos bajo el enfoque de *tracking-by-detection*, debido a que los errores cometidos por el detector se ven directamente reflejados en el desempeño general del algoritmo de seguimiento. De tal manera que el hecho de proporcionar tres diferentes detectores propicia el desarrollo de algoritmos más robustos para el seguimiento de objetos, ya que es posible evaluar el comportamiento del algoritmo de seguimiento considerando las variaciones entre distintos detectores, evitando caer en problemas de dependencia y sobreentrenamiento hacia un determinado detector.

En la Tabla 5.2 se muestra los tres detectores públicos proporcionados por la base de datos MOT17, incluyendo el detector DPM utilizado en la base de datos MOT16. Además, se muestra la técnica en que están basados y el año en que fueron publicados.

Tabla 5.2. Detectores públicos proporcionados en MOT17.

Detector	Técnica utilizada	Año	Ref.
DPM (<i>Deformable Part Model</i>)	SVM + HOG	2010	[111]
FR-CNN (<i>Faster R-Convolutional neural network</i>)	CNN	2015	[112]
SDP (<i>Scale-Dependent Pooling</i>)	CNN	2016	[113]

A su vez, en la Tabla 5.3 se muestra una comparación entre los tres detectores públicos, proporcionada por MOT17. Para esta comparación se utilizan métricas propias de la detección de objetos como: AP (*Average Precision*), MODA (*Multi Object Detection Accuracy*), MODP (*Multi Object Detection Precision*) y algunas otras de propósito general, cuya descripción se proporcionó en la sección 1.6.

Tabla 5.3. Comparación entre detectores públicos proporcionados por la base de datos MOT17.

Detector	AP ↑	MODA ↑	MODP ↑	FAF ↓	TP ↑	FP ↓	FN ↓	Rcll ↑	Prcn ↑	F1 ↑	FPS ↑
DPM	0.61	31.2	75.8	7.1	78,007	42,308	36,557	68.1	64.8	66.4	19.7
FR-CNN	0.72	68.5	78	1.7	88,601	10,081	25,963	77.3	89.8	83.1	5.1
SDP	0.81	79.9	78	1.3	95,699	7,599	18,865	83.5	92.6	87.9	0.6

Los datos resaltados en negritas en la Tabla 5.3 indican el mejor desempeño en cada una de las distintas métricas. Analizando estos resultados se puede concluir que el mejor desempeño lo obtiene el detector SDP, seguido de FR-CNN y en último lugar se encuentra el detector DPM.

A continuación, en la Figura 5.3 se muestra un ejemplo de las detecciones que proporciona un detector de objetos en dos cuadros de video consecutivos. En donde se observa que cada detección consiste en un cuadro delimitador que encierra lo que el detector considera como un objeto de interés, seguido de una puntuación que corresponde a la confiabilidad que el detector le asigna a cada detección (C_{det}). De tal manera que mientras mayor sea el valor de C_{det} con mayor seguridad la detección se trata de un objeto de interés.



Figura 5.3. Ejemplo de detecciones proporcionadas por un detector en dos cuadros de video consecutivos.

5.2.1 Supresión no máxima modificada

En el ejemplo de la Figura 5.3 se ilustra uno de los principales problemas en el área de la detección de objetos, conocido como detecciones redundantes. Que consiste en que el detector de objetos genera un número de detecciones mayor al número de objetivos presentes en la escena. En el ejemplo de la Figura 5.3 se observa que hay tres objetivos, las dos mujeres y el niño, por lo que idealmente deberían existir solo tres detecciones. Sin embargo, este no es el caso y uno de los objetivos genera detecciones redundantes. Este es un problema muy común que suele resolverse mediante la técnica de postprocesamiento supresión no máxima o NMS por sus siglas en inglés [72].

NMS puede entenderse como una búsqueda máxima local, en donde se pretende localizar las detecciones que tengan un alto grado de traslape, para después eliminar aquellas con C_{det} inferior, conservando únicamente la detección con el máximo valor de C_{det} .

El algoritmo tradicional NMS se basa en el cálculo de la intersección sobre la unión ente detecciones (RU_{D_i,D_j}) como se muestra en la Ec. (5.1).

$$RU_{D_i,D_j} = \frac{area(D_i \cap D_j)}{area(D_i \cup D_j)} \quad (5.1)$$

En donde D_i y D_j son dos detecciones diferentes y $area$ indica el cálculo del área de la detección. El algoritmo NMS consiste en enlistar todas las detecciones D y su correspondiente valor de confiencialidad C_{det} , para seleccionar la detección con el valor C_{det} más alto ($C_{det-max}$). Esta detección se elimina del conjunto D y se agrega al conjunto final D_{NMS} . Posteriormente se calcula RU_{D_i,D_j} , donde D_i corresponde a la detección con $C_{det-max}$ y D_j corresponde a las detecciones restantes en el conjunto D . En los casos en donde RU_{D_i,D_j} supere un umbral T_R se eliminara del conjunto D la detección D_j correspondiente. Este proceso se repite hasta que el conjunto D quede vacío. Los pasos específicos se describen a continuación.

1. Ordene los valores C_{det} de todas las detecciones en el conjunto D , después seleccione la detección con el valor C_{det} más alto ($C_{det-max}$).

2. Escanee las detecciones D_j restantes en el conjunto D , si el área de traslape con la detección $C_{det-max}$ actual es mayor que el umbral T_R , elimine la detección D_j correspondiente del conjunto D .
3. Repita los pasos 1 y 2 con las detecciones restantes hasta que el conjunto D este vacío.

Como resultado del algoritmo NMS se obtiene el conjunto D_{NMS} , que idealmente solo debería contener detecciones que no son redundantes. El verdadero reto en algoritmo NMS es encontrar el valor adecuado de T_R , debido a que un valor de T_R demasiado alto podría no eliminar adecuadamente todas las detecciones redundantes y un valor de T_R demasiado bajo podría eliminar detecciones de objetos diferentes que estén muy cercanos. Por tal motivo, seleccionar este umbral no es tarea fácil y frecuentemente se fija con base en la experiencia.

En esta tesis se desarrolló una versión modificada del algoritmo NMS, que se adapta mejor a las necesidades que se tienen en el seguimiento de múltiples objetos, la intención en esta versión del algoritmo NMS es eliminar solo aquellas detecciones redundantes que están bien alineadas con el objetivo, ya que estas detecciones pueden causar problemas como cambios de identidad a la hora de realizar el seguimiento de objetos, por lo que es de especial interés eliminar este tipo de detecciones. En la Figura 5.4 (a) se muestra un ejemplo de detecciones redundantes alineadas y en la Figura 5.4 (b) detecciones no alineadas.



Figura 5.4. Ejemplo de detecciones redundantes alineadas (a) y no alineadas (b).

Para lograr eliminar este tipo de detecciones se realizó una modificación al algoritmo NMS en donde en lugar de calcular RU_{D_i,D_j} , se calcula la intersección sobre el área mínima RM_{D_i,D_j} como se muestra en la Ec. (5.2), y se fijó el umbral T_R a 0.9 de manera experimental. Al ser un umbral demasiado grande se garantiza que se conserven las detecciones de objetos muy cercanos, lo cual es necesario para escenarios muy concurridos y se eliminan solo aquellas detecciones que pudieran generar confusiones. Salvo por la modificación en el cálculo de RM_{D_i,D_j} , los demás pasos en el algoritmo NMS se realizan de la misma forma.

$$RM_{D_i,D_j} = \frac{\text{area}(D_i \cap D_j)}{\min(\text{area}(D_i), \text{area}(D_j))} \quad (5.2)$$

A continuación, en la Figura 5.5 se muestra un ejemplo de las detecciones que se conservan después de aplicar el algoritmo NMS modificado.

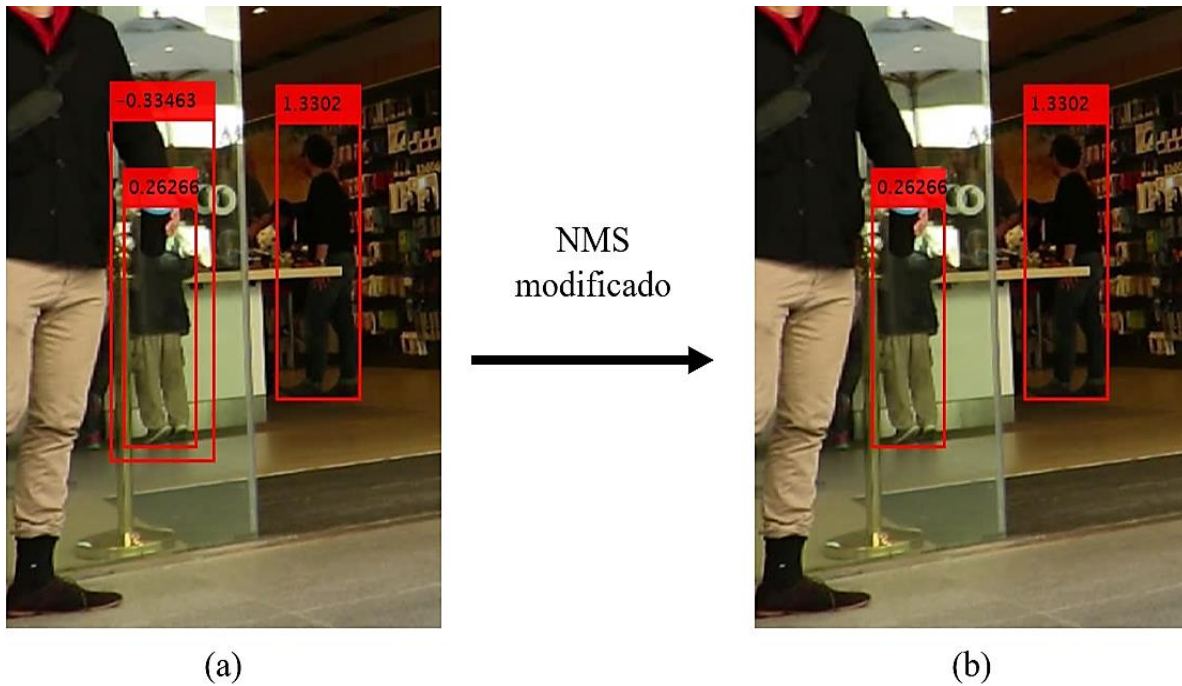


Figura 5.5. Detecciones antes (a) y después (b) de aplicar el algoritmo NMS modificado.

El conjunto de detecciones finales se denota por D_{nms} . Sin embargo, por simplicidad esta notación se omitirá y a partir de este punto al conjunto de detecciones que se conservan después de aplicar el algoritmo NMS modificado se denotará solo como D .

5.3 Algoritmo para el seguimiento de múltiples objetos

En esta sección se describe el algoritmo para el seguimiento de múltiples objetos, en sus tres versiones distintas: T-AM, T-MM y TAMM. Todas las versiones comparten la misma estructura general del algoritmo, la diferencia entre las tres versiones está en la información que se extrae de los objetos, que puede ser patrones de movimiento, apariencia o ambos.

5.3.1 Estructura general del algoritmo

La estructura general del algoritmo para el seguimiento de múltiples objetos se muestra en la Figura 5.6, seguido de una descripción de cada una de las etapas que lo conforman.

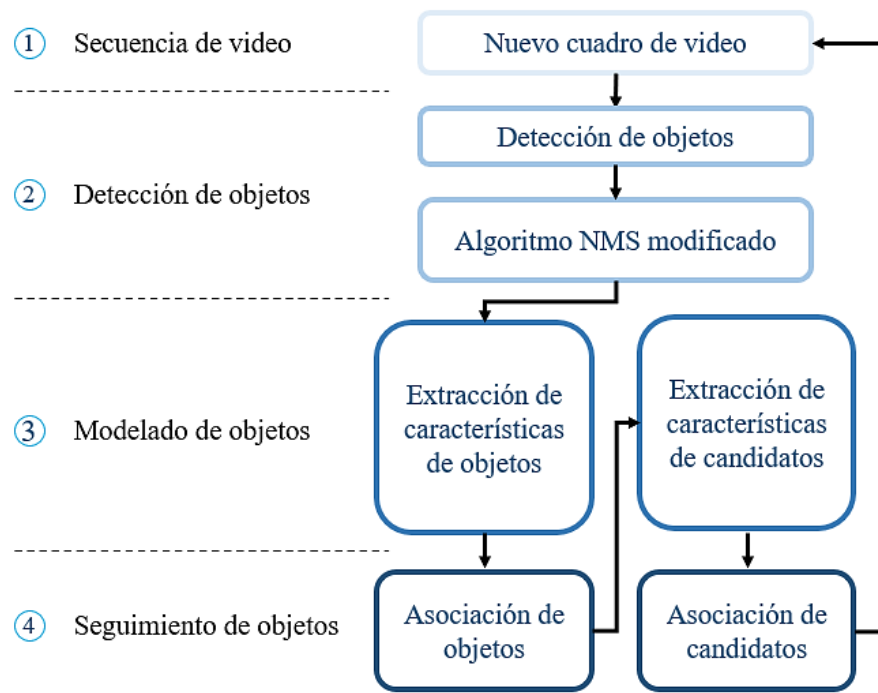


Figura 5.6. Estructura general del algoritmo para el seguimiento de múltiples objetos.

Secuencia de video: Esta etapa se encarga de proveer el cuadro actual de la secuencia de video, el algoritmo termina cuando toda la secuencia de video haya sido procesada.

Detección de objetos: Esta etapa consiste en aplicar un algoritmo de detección de objetos al cuadro actual de la secuencia de video. Posteriormente, las detecciones generadas por el detector de objetos son procesadas por el algoritmo NMS modificado para eliminar detecciones

redundantes y generar el conjunto de detecciones (D).

Modelado de objetos: Esta etapa se encarga de extraer la información relevante de los objetos para conservar su identidad durante su aparición en la escena. Como se mencionó anteriormente las características que se extraen en esta etapa depende de la versión del algoritmo que se utilice. Sin embargo, en las tres versiones T-AM, T-MM y T-AMM lo que se busca en esta etapa es generar la matriz de costo C_{sij} del capítulo 4. La manera exacta en que dicha matriz es generada se explica con mayor detalle en las secciones 5.4, 5.5 y 5.6 según la versión del algoritmo que se utilice.

Seguimiento de objetos: En esta etapa se generan las trayectorias de los objetos mediante la asociación entre las detecciones en el cuadro t (D^t) y los objetos del cuadro $t-1$ (O^{t-1}). Resolviendo la matriz de costo C_{sij} generada en la etapa anterior, utilizando el método Húngaro.

Un aspecto importante en los algoritmos propuestos para el seguimiento de múltiples objetos, Figura 5.6, es la distinción entre objetos y candidatos (C). Esta distinción es un mecanismo para disminuir los falsos positivos generados por los detectores de objetos, ya que toda detección se convierte en un candidato, pero no todo candidato logra convertirse en objeto. Para definir cuales candidatos se convertirán en objetos se creó un valor de confidencialidad (Cnf). El valor Cnf en esencia es un contador que indica en cuantos cuadros de la secuencia de video ha sido visto el candidato. Se dice que el candidato ha sido visto en un determinado cuadro de video cuando existe alguna detección que se asocie con el candidato. El valor Cnf de cada candidato inicia en cero cuando el candidato es visto por primera vez, cada cuadro de video en el que el candidato ha sido visto incrementa en 1 su valor Cnf . En el caso contrario, los cuadros de video en donde el candidato no ha sido visto su valor Cnf disminuye en 1.

Un candidato se convertirá en objeto solo después de haber superado un periodo de prueba, que consiste en que el candidato consiga una confidencialidad tal que $Cnf \geq 5$. Este umbral se fijó de manera experimental e indica que un candidato se convertirá en objeto solo después de haber sido visto durante al menos 5 cuadros de video.

Un candidato desaparece cuando no ha sido visto durante 3 cuadros de video consecutivos o si su valor Cnf es negativo. En el caso de que el candidato logre convertirse en objeto, este

desaparece hasta después de no haber sido visto durante 5 cuadros de video consecutivos. En algunos trabajos propuestos en la literatura, los objetos desaparecen inmediatamente después de no haber sido vistos en el cuadro actual. En esta tesis se incrementó a 3 cuadros en el caso de los candidatos y 5 cuadros en el caso de los objetos, ambos umbrales fijados de manera experimental. La razón es que los objetos pueden no ser vistos en algún cuadro de la secuencia de video a causa de un falso negativo generado por el detector de objetos, pero no necesariamente significa que el objeto desapareció de la escena. En algunos casos el objeto puede no ser visto en un cuadro t y aparecer en el cuadro $t+1$, por lo que extender el periodo de vida de los objetos permite disminuir los falsos negativos generados por el detector de objetos.

En la Figura 5.7 se muestra un ejemplo de la manera en que se generan las trayectorias de los objetos. En la parte superior se muestra el conjunto de detecciones para cada cuadro t de la secuencia de video. Posteriormente se muestra en color azul y rojo los candidatos que logran convertirse en objetos y en color negro aquellos candidatos que desaparecen y no logran convertirse en objetos. Los círculos que aparecen sin rellenar (C_3^3, C_4^8, O_1^7 , etc.) indican que el candidato u objeto no fue visto en dicho cuadro de la secuencia de video.

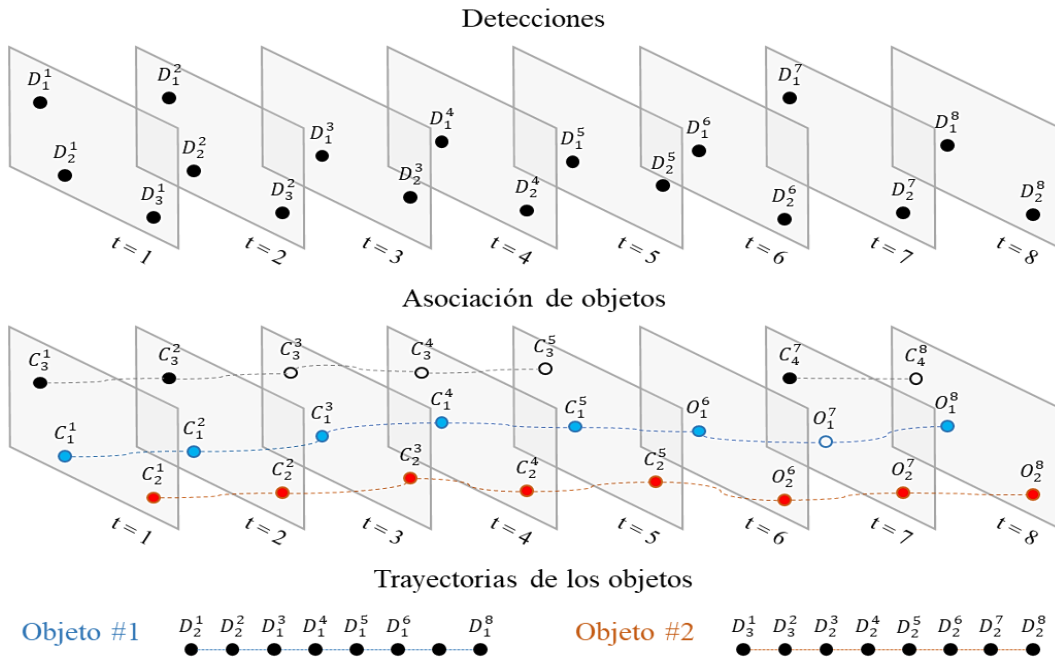


Figura 5.7. Ejemplo de la creación de los candidatos (C) a partir de las detecciones (D) para generar las trayectorias de los objetos (O).

En la Figura 5.7 las detecciones D_1^1 y D_1^7 que generan los candidatos C_3^1 y C_4^7 son un ejemplo de la eliminación correcta de un falso positivo, ya que estos errores no logran convertirse en objetos. Por otra parte, los candidatos C_1^1 y C_2^1 logran alcanzar el valor Cnf solicitado y se convierten en objetos a partir del sexto cuadro de video ($t=6$), después de concluir el periodo de prueba. Respecto a los falsos negativos estos se solucionan utilizando la información previa del objeto para predecir su posición en ausencia de una detección, esta situación se ejemplifica con el objeto O_1^7 en el séptimo cuadro de video ($t=7$).

Finalmente, es importante mencionar que la distinción entre objetos y candidatos también permite realizar una asociación jerárquica, en donde se les da prioridad a los objetos con mayor confiabilidad.

5.3.2 Detalles de la implementación

El algoritmo propuesto para el seguimiento de múltiples objetos en sus tres versiones se evaluó bajo los siguientes lineamientos.

La evaluación cuantitativa del algoritmo se realizó utilizando el estándar propuesto en *MOT Benchmark* [83], utilizando las secuencias de entrenamiento de la base de datos MOT16 y MOT17, así como los detectores públicos proporcionados por la respectiva base de datos.

El estándar propuesto en *MOT Benchmark* consta de una gran variedad de métricas como: MOTA, MOTP, IDF1, MT, PT, ML, FP, FN e IDsw cuya descripción y formalización matemática se abordó en la sección 1.6.

En el estándar *MOT Benchmark* las métricas más relevantes son: MOTA, MOTP e IDF1. Debido a que dichas métricas evalúan los aspectos más importantes en el seguimiento de múltiples objetos, como la cobertura del objeto y la preservación de su identidad.

En lo referente a la velocidad de procesamiento del algoritmo, esta se calcula dependiendo del número de objetos presentes en la escena. La velocidad de procesamiento en las tres versiones del algoritmo se evalúa bajo las mismas condiciones utilizando un CPU Intel Xeon E5620 a 2.4 GHz con una GPU NVIDIA GeForce GTX 1070 y 8 GB de memoria RAM.

5.4 Algoritmo T-AM

En esta sección se describe la primera versión del algoritmo de seguimiento de múltiples objetos, nombrado T-AM el cual está basado en el modelo de apariencia SiamAM descrito en el capítulo 2 y el método Húngaro para la asociación de datos con un umbral $Th_{Af} = 0.5$, descrito en el capítulo 4.

5.4.1 Modelado de objetos en el algoritmo T-AM

La matriz de costo en el algoritmo T-AM se obtiene mediante la propagación de O^{t-1} y D^t por la red siamesa para obtener su similitud, Figura 5.8 (a), las detecciones que no se asignan a ningún objeto (Dc^t), se propagan de nuevo por la red siamesa, pero en esta ocasión con los candidatos C^{t-1} , Figura 5.8 (b). En este ejemplo se aprecia mejor la asignación jerárquica, debido a que primero se busca asociar las detecciones con los objetos y solo en caso de que existan detecciones sobrantes se busca una asociación entre detecciones y candidatos.

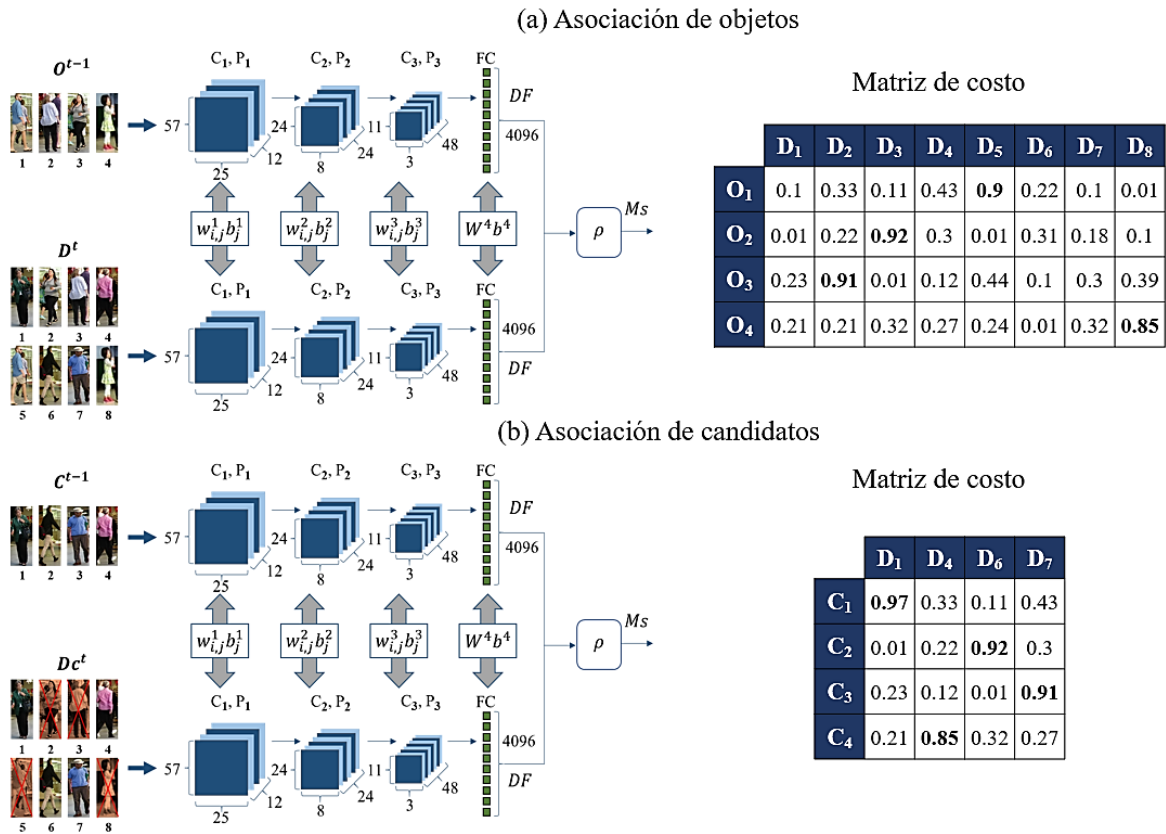


Figura 5.8. Generación de la matriz de costo en el algoritmo T-AM para objetos (a) y candidatos (b).

5.4.2 Evaluación del algoritmo T-AM

Los resultados obtenidos por el algoritmo T-AM en la base de datos MOT16 y MOT17 se muestran en la Tabla 5.4, mientras que la velocidad relativa de procesamiento se muestra en la Figura 5.9.

Tabla 5.4. Evaluación del algoritmo T-AM en la base de datos MOT16 y MOT17.

Detector	MOTA 0-100↑	MOTP 0-100↑	IDF1 0-100↑	MT ↑	PT ↑	ML ↓	FP ↓	FN ↓	IDsw ↓
MOT16									
DET	60.89	83.22	40.46	175	228	114	4752	36568	1865
MOT17									
DPM	30.46	76.34	23.69	58	222	266	10153	66324	1618
FRCNN	48.87	87.69	46.04	110	252	184	3837	52818	757
SDP	60.03	83.20	40.14	170	232	144	4659	38363	1861
Total	46.45	82.93	36.90	338	706	594	18649	157505	4236

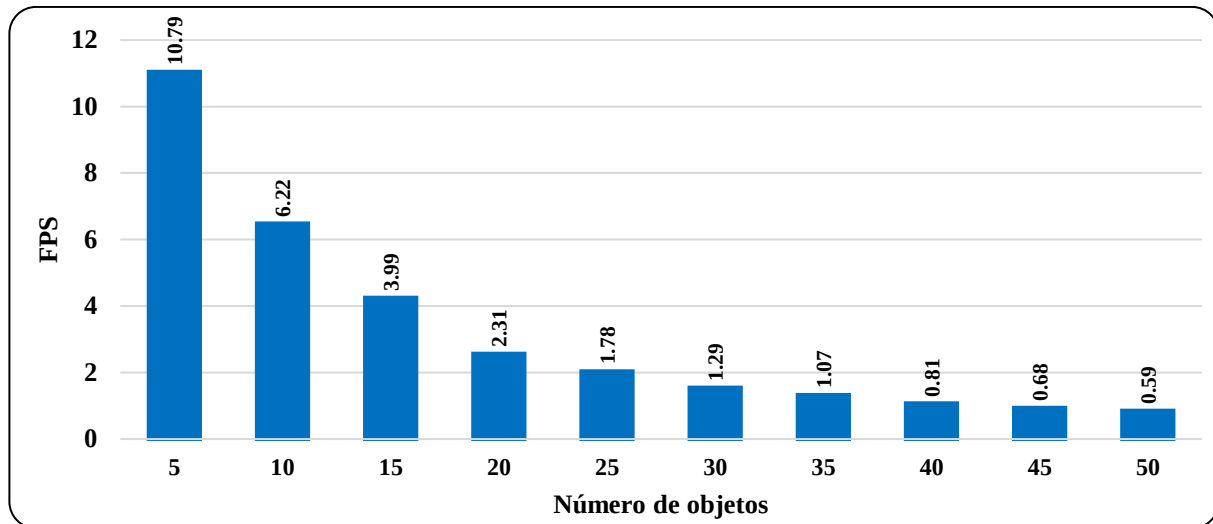


Figura 5.9. Velocidad relativa de procesamiento del algoritmo T-AM.

5.5 Algoritmo T-MM

En esta sección se describe la segunda versión del algoritmo de seguimiento de múltiples objetos, nombrado T-MM el cual está basado en el modelo de movimiento utilizando el filtro de Kalman descrito en el capítulo 3 y el método Húngaro para la asociación de datos con un umbral $Th_{Af} = 0.3$, descrito en el capítulo 4.

5.5.1 Modelado de objetos en el algoritmo T-MM

La matriz de costo en el algoritmo T-MM se obtiene mediante el cálculo de la intersección sobre la unión entre las predicciones realizadas por el FK y las detecciones en el cuadro de video actual. Teniendo en cuenta que la posición de los objetos se representa utilizando un cuadro delimitador que encierra al objetivo. En la Figura 5.10 (a) se muestra un ejemplo para el caso de los objetos y en Figura 5.10 (b) para los candidatos.

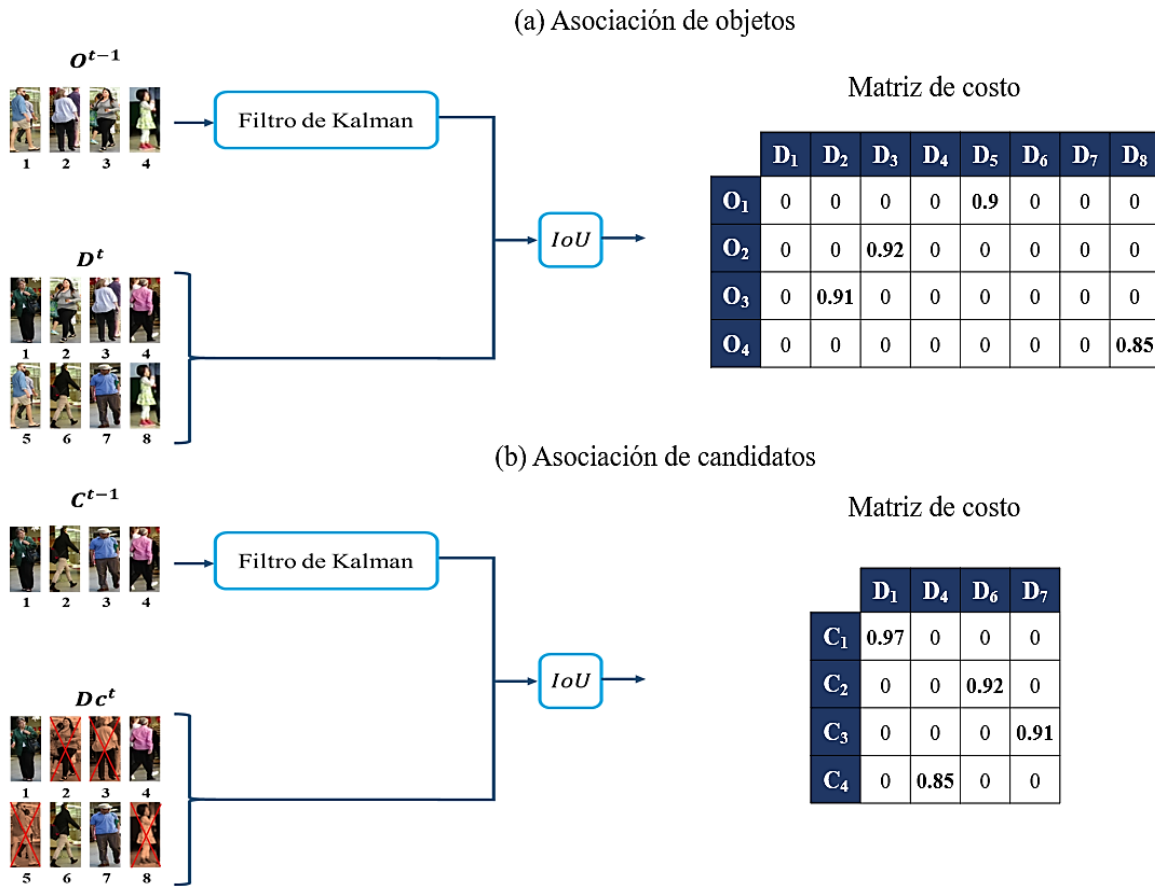


Figura 5.10. Generación de la matriz de costo en el algoritmo T-MM para objetos (a) y candidatos (b).

5.5.2 Evaluación del algoritmo T-MM

Los resultados obtenidos por el algoritmo T-MM en la base de datos MOT16 y MOT17 se muestran en la Tabla 5.5, mientras que la velocidad relativa de procesamiento se muestra en la Figura 5.11.

Tabla 5.5. Evaluación del algoritmo T-MM en la base de datos MOT16 y MOT17.

Detector	MOTA 0-100↑	MOTP 0-100↑	IDF1 0-100↑	MT ↑	PT ↑	ML ↓	FP ↓	FN ↓	IDsw ↓
MOT16									
DET	65.57	85.06	61.09	203	214	100	3590	33760	661
MOT17									
DPM	36.47	77.04	39.97	70	214	262	7582	63305	451
FRCNN	49.40	87.13	50.71	120	254	172	4273	51869	683
SDP	64.68	85.05	60.54	198	220	128	3466	35527	666
Total	50.19	83.61	50.86	388	688	562	15321	150701	1800

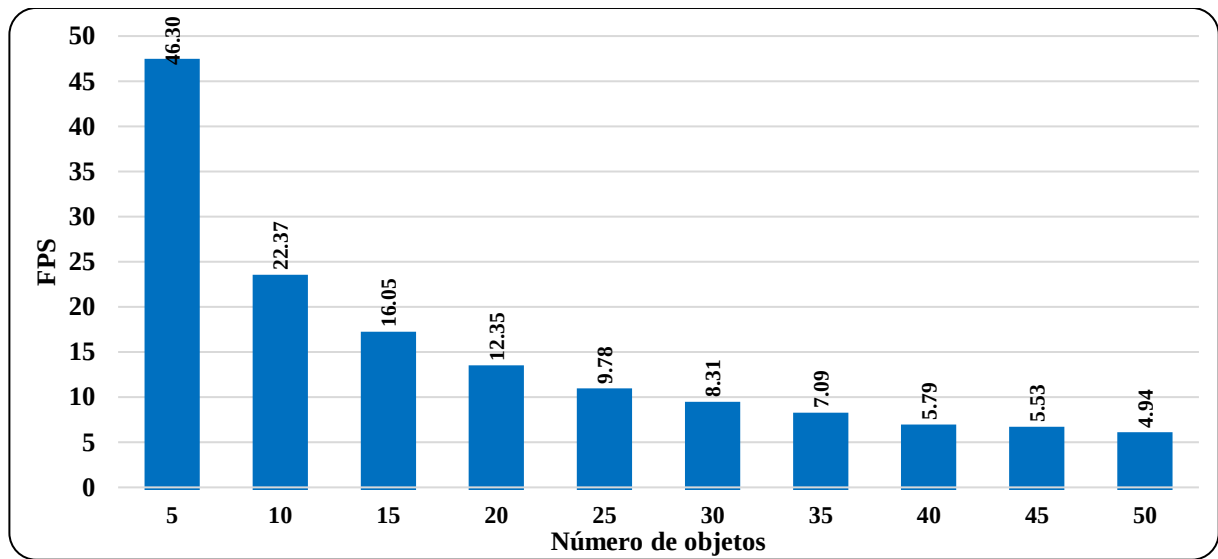


Figura 5.11. Velocidad relativa de procesamiento del algoritmo T-MM.

5.6 Algoritmo T-AMM

En esta sección se describe la tercera versión del algoritmo de seguimiento de múltiples objetos, nombrado T-AMM el cual está basado en la fusión del modelo de apariencia (SiamAM), el modelo de movimiento (filtro de Kalman) y el método Húngaro para la asociación de datos con un umbral $Th_{Af} = 0.4$.

5.6.1 Modelado de objetos en el algoritmo T-AMM

La matriz de costo en el algoritmo T-AMM se obtiene mediante la suma ponderada de la matriz de costo de los algoritmos T-AM y T-MM. La ponderación asignada para cada matriz de costo es de 60% para T-MM y 40% para T-AM, dicha ponderación se ajustó de manera

experimental. La obtención de la matriz de costo en el algoritmo T-AMM se muestra en la Figura 5.12, por cuestiones de espacio solo se muestra el caso de los objetos, aunque para el caso de los candidatos se realiza de forma similar, pero utilizando detecciones Dc^t como se realizó en las versiones T-AM y T-MM.

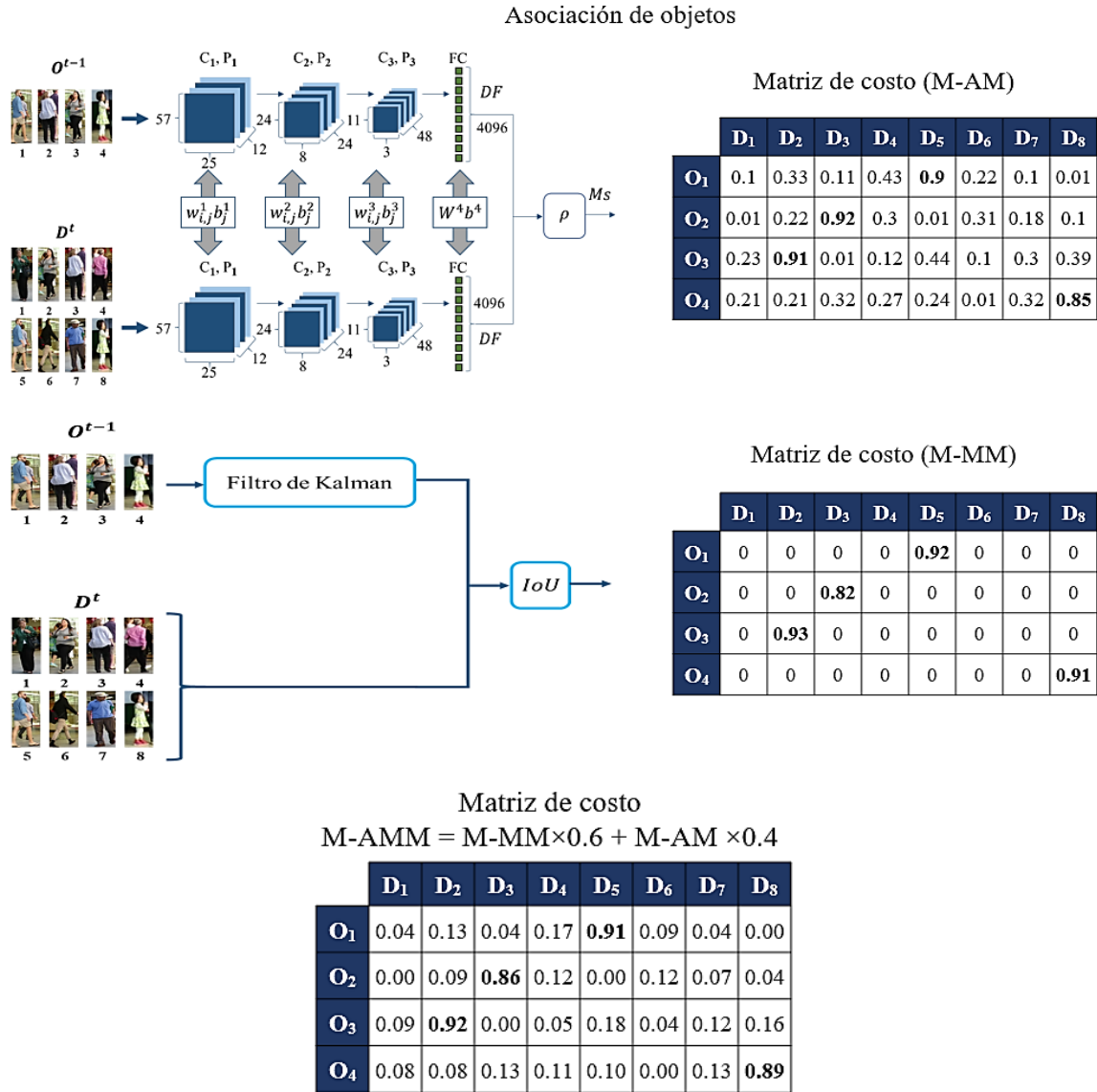


Figura 5.12. Generación de la matriz de costo en el algoritmo T-AMM para objetos.

5.6.2 Evaluación del algoritmo T-AMM

Los resultados obtenidos por el algoritmo T-AMM en la base de datos MOT16 y MOT17 se

muestran en la Tabla 5.6, mientras que la velocidad relativa de procesamiento se muestra en la Figura 5.13.

Tabla 5.6. Evaluación del algoritmo T-AMM en la base de datos MOT16 y MOT17.

Detector	MOTA 0-100↑	MOTP 0-100↑	IDF1 0-100↑	MT ↑	PT ↑	ML ↓	FP ↓	FN ↓	IDsw ↓
MOT16									
DET	66.05	85.16	62.10	211	201	105	3447	33482	558
MOT17									
DPM	36.49	77.27	40.82	76	210	260	7531	63372	421
FRCNN	50.03	87.25	52.24	133	244	169	3869	51735	510
SDP	65.16	85.15	61.55	205	207	134	3319	35243	560
Total	50.56	83.76	51.99	414	661	563	14719	150350	1491

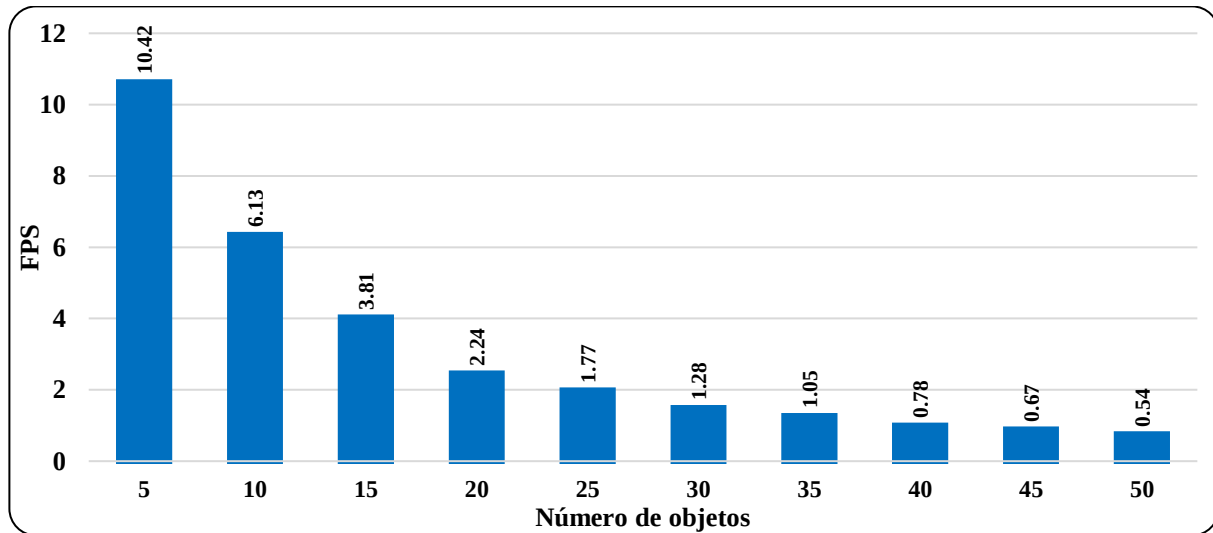


Figura 5.13. Velocidad relativa de procesamiento del algoritmo T-AMM.

5.7 Comparación entre los algoritmos T-AM, T-MM y T-AMM

A continuación, se presenta una comparación de los algoritmos propuesto para el seguimiento de múltiples objetos. En dicha comparación se muestra el desempeño que obtiene cada versión del algoritmo en las métricas más importantes para el seguimiento de múltiples objetos, utilizando la base de datos MOT16 y MOT17. Mostradas en la Figura 5.14 y la Figura 5.15 respectivamente. A su vez, en la Figura 5.16 se muestra la comparación de su desempeño en términos de velocidad relativa de procesamiento.

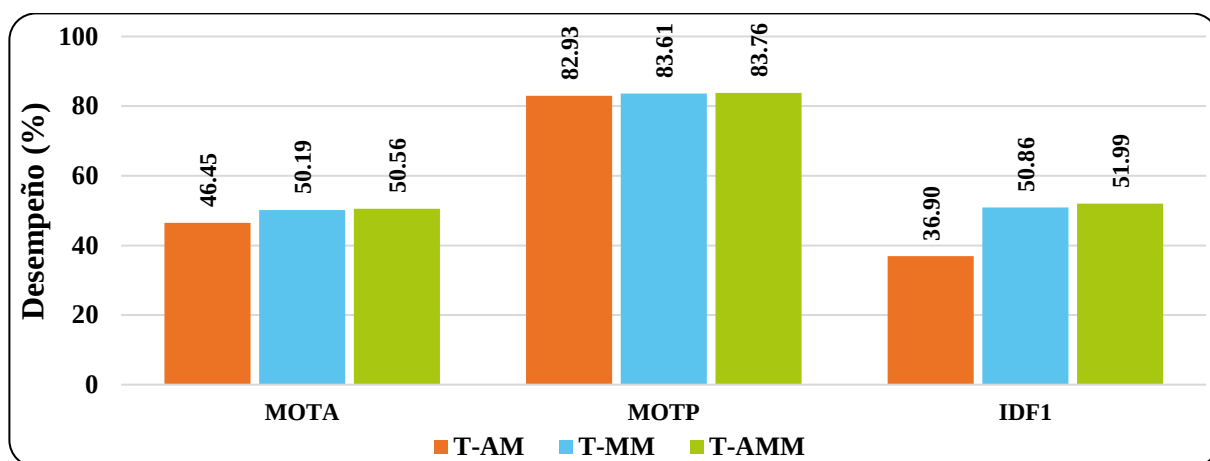


Figura 5.14. Comparación entre algoritmos utilizando las métricas MOTA, MOTP e IDF1 en MOT17.

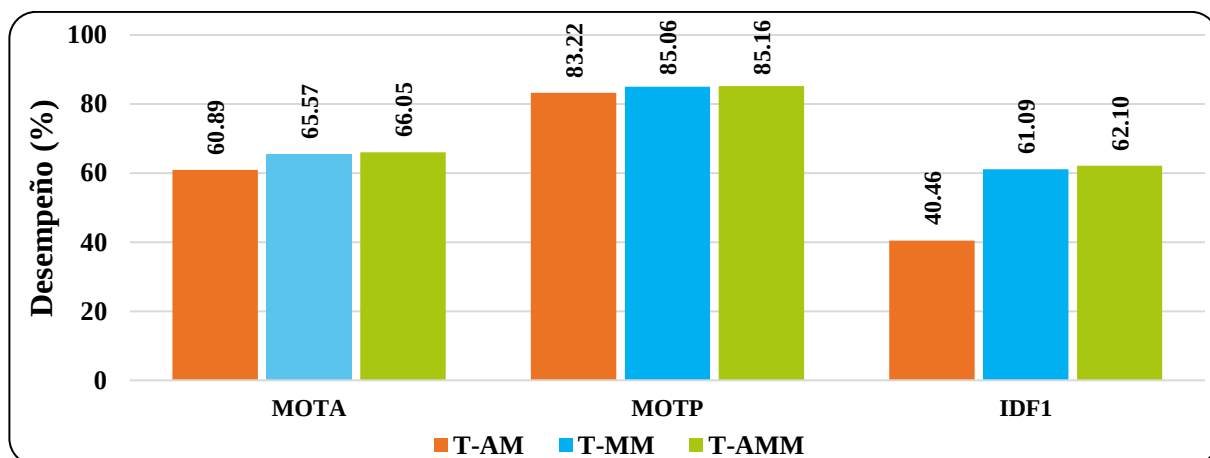


Figura 5.15. Comparación entre algoritmos utilizando las métricas MOTA, MOTP e IDF1 en MOT16.

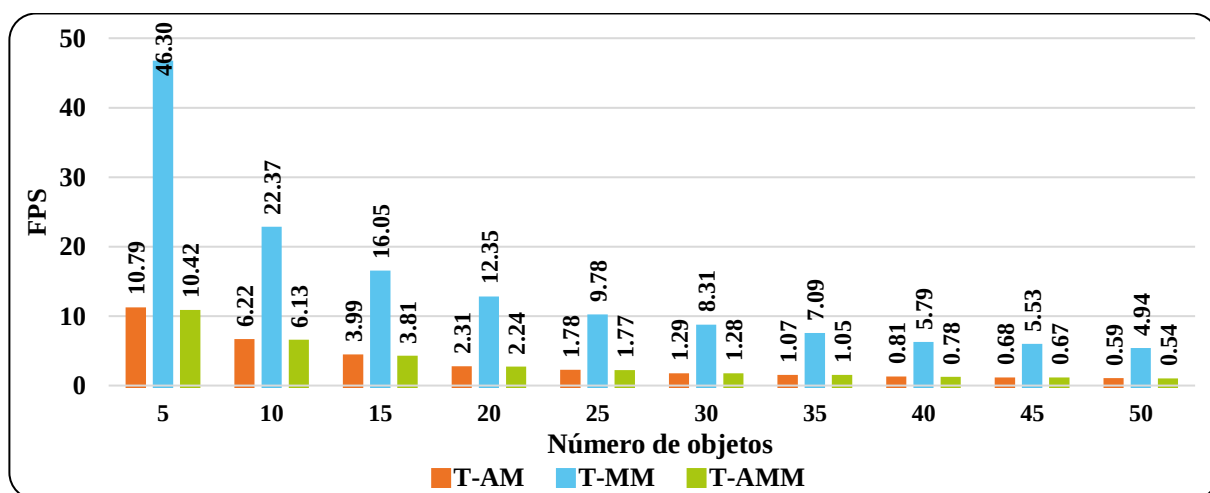


Figura 5.16. Comparación entre algoritmos utilizando la velocidad relativa de procesamiento.

Con base en esta comparación se concluye que el algoritmo T-AMM obtiene el mejor desempeño en las métricas MOTA, MOTP e IDF1 tanto en la base de datos MOT16 como MOT17. Lo que convierte a este algoritmo en el más exacto y preciso, además de ser el que menor cantidad de cambios de identidad genera. Sin embargo, su velocidad de procesamiento es la más baja entre los tres.

Respecto al algoritmo T-MM este es considerado como la versión más práctica. Ya que posee un balance muy apropiado entre su desempeño en las principales métricas de seguimiento y su velocidad de procesamiento. La diferencia en términos de las métricas MOTA, MOTP e IDF1 entre el algoritmo T-MM y T-AMM es alrededor 0.37%, 0.15% y 1.13% respectivamente, lo cual no es muy significativo. En cambio, la diferencia en cuanto a velocidad de procesamiento tomando como base el caso de 20 objetos, coloca al algoritmo T-MM muy por encima de los algoritmos T-AM y T-AMM, con una superioridad de 10.11 FPS en velocidad de procesamiento. Otro aspecto para resaltar del algoritmo T-MM es el hecho de ser la única versión que puede ser extendida para realizar el proceso de seguimiento ante cualquier clase de objetos. Esto debido a que T-MM solo considera la información espacial del cuadro delimitador que encierra al objetivo, por lo que es exactamente lo mismo si este se trata de una persona, un vehículo o cualquier otro objeto. Por otro lado, los algoritmos T-AM y T-AMM están basados en el modelo de apariencia SiamAM, el cual fue diseñado específicamente para la reidentificación de personas. Por tal motivo para utilizarse con otra clase de objetos es necesario entrenar de nuevo el modelo.

5.8 Comparación del algoritmo T-AMM con la literatura

En esta sección se muestra una comparación del algoritmo T-AMM contra métodos propuestos en la literatura. El algoritmo T-AMM se seleccionó por ser el que mejores resultados obtuvo en las métricas de seguimiento de objetos en comparación con T-MM y T-AM. Los métodos en la literatura con los cuales será comparado fueron seleccionados debido a que se evalúan en la base de datos MOT17 bajo el estándar de métricas *MOT Benchmark*, además de haber sido publicados en el periodo 2018 - 2020.

Los resultados obtenidos se muestran en la Tabla 5.7, en donde se realiza una distinción

entre métodos *online* y *offline*. Ya que como se mencionó anteriormente los métodos *offline* suelen obtener un mejor desempeño debido a que en su implementación consideran información de cuadros de video futuros, lo que les da una gran ventaja sobre los métodos *online* ya que estos solo consideran información del cuadro de video actual y/o anteriores.

Tabla 5.7. Comparación del algoritmo T-AMM con la literatura en la base de datos MOT17 (Resultados del 17/noviembre/2020).

Método	Online	MOTA (%) ↑	MOTP (%) ↑	IDF1 (%) ↑	MT (%) ↑	ML (%) ↓	FPS ↑
Lif_T [114]	No	60.5	78.3	65.6	27.0	33.6	0.5
MLT [115]	No	75.3	81.7	75.5	49.3	19.5	4.4
LSST17 [96]	No	54.9	77.2	63.1	24.4	38.1	2.5
TPM [116]	No	54.4	76.7	52.6	22.8	37.5	0.8
MPNTrack [117]	No	58.8	78.6	61.7	28.8	33.5	6.5
FPSN-MOT [118]	Si	44.9	76.6	48.4	16.5	35.8	10.1
YoonKJ17 [119]	Si	51.4	77.0	54.0	21.2	37.3	3.4
STRN [73]	Si	50.9	75.6	56.0	18.9	33.8	1.4
TrctrD17 [120]	Si	53.7	77.2	53.8	19.4	36.6	4.9
SiameseRF [121]	Si	55.8	77.9	46.7	21.6	34.4	11.1
Deep_TAMA [122]	Si	50.3	76.7	53.5	19.2	37.5	1.5
T-AMM	Si	50.6	83.8	52.0	25.2	40.4	2.2
T-AMM (DPM)	Si	36.5	77.3	40.8	13.9	47.6	2.2
T-AMM (FRCNN)	Si	50.0	87.3	52.2	24.4	31.0	2.2
T-AMM (SDP)	Si	65.2	85.2	61.6	37.5	24.5	2.2

En la Tabla 5.7 se observa que el algoritmo T-AMM en comparación con los métodos *online* muestra resultados competitivos en la mayoría de las métricas utilizadas, e incluso obtiene el mejor puntaje en métricas como MOTP y MT.

En la tercera sección de la Tabla 5.7 se muestra el desempeño que obtiene el algoritmo T-AMM al utilizar cada uno de los detectores públicos que se proporcionan en la base de datos MOT17 (DPM, FRCNN y SDP). El desempeño promedio del algoritmo T-AMM al utilizar los tres detectores públicos, es el desempeño que se utiliza para comparar el algoritmo T-AMM con la literatura. Sin embargo, un aspecto importante a considerar es que el desempeño de T-AMM utilizando el detector DPM es muy inferior en comparación a cuando se utilizan los detectores FRCNN y SDP. En donde se observa que al utilizar el detector DPM se obtiene un desempeño muy por debajo de la media en la mayoría de las métricas, esto ocasiona que el

desempeño promedio del algoritmo T-AMM disminuya considerablemente. De tal manera que si utilizamos el desempeño que obtiene el algoritmo T-AMM utilizando solo el detector SDP, se observa un desempeño muy superior en comparación con los métodos *online* obteniendo el mejor puntaje en todas las métricas excepto en FPS. Este desempeño es competitivo incluso en comparación con los métodos *offline* obteniendo el mejor puntaje en la métrica MOTP y el segundo lugar en las métricas MOTA, MT y ML.

La razón por la cual el desempeño del algoritmo T-AMM disminuye drásticamente al utilizar el detector DPM, es quizás porque el modelo de apariencia (SiamAM) no es capaz de establecer una similitud entre las detecciones generadas por DPM. Esto debido a que las imágenes que se utilizan durante el entrenamiento de SiamAM son imágenes recortadas con precisión en donde el objeto de interés aparece centrado y bien definido, si el detector DPM no es capaz de generar detecciones igual de precisas SiamAM no puede establecer una similitud entre las detecciones. Además, el detector DPM es el que más falsos negativos genera de los tres detectores utilizados, lo que provoca que también el modelo de movimiento (filtro de Kalman) falle. Esto debido a que durante la fase de corrección en el filtro de Kalman se corrigen las predicciones realizadas por el FK utilizando la información actual de los objetivos. Esta información no es posible obtenerla si se presenta un falso negativo, esto ocasiona que el FK solo realice la etapa de predicción provocando que la incertidumbre en las predicciones crezca y comiencen a ser erróneas debido a que carecen de la etapa de corrección.

Para finalizar en este capítulo se describieron tres algoritmos propuestos para el seguimiento de múltiples objetos. Los cuales se compararon entre sí ante las principales métricas en el seguimiento de múltiples objetos, así como en su velocidad de procesamiento. Esta comparación arrojó que el algoritmo T-AMM obtiene el mejor desempeño en cuanto a métricas de seguimiento y T-MM la mejor velocidad de procesamiento. Por su alto desempeño en las métricas de seguimiento T-AMM fue seleccionado para ser comparado con métodos propuestos en la literatura reciente, en donde se observó un desempeño competitivo contra métodos *online* y *offline*.

CAPÍTULO 6

CONCLUSIONES Y TRABAJO FUTURO

En este trabajo de investigación se realizaron tres algoritmos para el seguimiento de múltiples objetos nombrados T-AM, T-MM y T-AMM. Los tres algoritmos tienen en común que están basados en el enfoque del *tracking-by-detection* utilizando el método Húngaro y un umbral de afinidad Th_{Af} para realizar la asociación entre detecciones en cuadros subsecuentes. La diferencia entre los tres algoritmos está en la información que se extrae de los objetos, que puede ser patrones de movimiento, apariencia o ambos como se explica a continuación.

T-AM utiliza la información de apariencia de los objetos, utilizando características profundas extraídas de un nuevo modelo de apariencia llamado SiamAM, que está basado en una CNN en configuración siamesa diseñado en esta tesis para la tarea de reidentificación de personas. Este modelo obtuvo el mayor grado de discriminación al ser comparado con diversos modelos de CNN propuestos en la literatura.

T-MM se basa en la información de movimiento de los objetos, utilizando el filtro de Kalman propuesto por Bewley [109] para predecir la posición de los objetos. En esta tesis se realizaron algunas modificaciones en la estructura general del algoritmo para el seguimiento de múltiples objetos propuesto en [109], como la asociación jerárquica realizando una distinción entre objetos y candidatos a modo de mecanismo para disminuir los falsos positivos generados por los detectores de objetos. Otra modificación fue la extensión de la duración de las trayectorias de los objetos, en donde se establece que un objeto desaparece hasta después de no haber sido visto durante más de 5 cuadros de video consecutivos, a diferencia de [109] en donde los objetos desaparecen inmediatamente después de no haber sido vistos en el cuadro actual lo cual ocasiona problemas cuando se presentan falsos negativos generados por los detectores. Estas modificaciones permite mejorar el desempeño de T-MM en métricas como MOTA, MOTP e IDF1 en un 7.09%, 5.81% y 11.06% respectivamente en comparación con [109]. T-MM es la versión que se recomienda utilizar en la práctica, ya que posee un buen balance entre su desempeño en las métricas de seguimiento y su velocidad de procesamiento, la cual se estima alrededor de 12.35 FPS, lo que la convierte en la versión más rápida incluso al compararla con

algunos métodos en la literatura reciente. Esta versión también es la única que puede ser fácilmente extendida para realizar el seguimiento de objetos de diferentes clases, ya que únicamente se basa en la información espacial de los objetos y no utiliza información sobre la apariencia de dichos objetos.

T-AMM utiliza una fusión en la información de la apariencia de los objetos y su movimiento, mediante una combinación entre SiamAM y el filtro de Kalman que se utilizan en los algoritmos T-AM y T-MM respectivamente. Esta versión obtiene el mejor desempeño en las principales métricas de seguimiento en comparación con T-AM y T-MM. Además, mostró ser competitiva en comparación con algunos métodos *online* propuestos en la literatura reciente, obteniendo el mejor puntaje en métricas como MOTP y MT. Asimismo, utilizando un detector adecuado como en el caso del detector SDP se observó un incremento notable en el desempeño del algoritmo T-AMM, el cual mostró ser muy superior en comparación con métodos *online* al grado de llegar a ser competitivo incluso con métodos *offline*.

Para finalizar se mencionan algunas áreas de oportunidad para trabajos futuros que podrían contribuir a mejorar el desempeño de los algoritmos propuestos en esta tesis.

Detector de objetos: El detector de objetos es un elemento fundamental en cualquier algoritmo que opere bajo el enfoque del *tracking-by-detection*. De manera que desarrollar un detector de objetos que supere la precisión de los detectores utilizados en esta tesis, repercutirá favorablemente en mejorar el desempeño del algoritmo para el seguimiento de múltiples objetos.

Modelo de apariencia: El modelo de apariencia utilizado en esta tesis, fue diseñado específicamente para la reidentificación de personas. Por lo que tiene la limitante de ser de utilidad solo en aplicaciones en donde los objetos de interés son personas. Para corregir este inconveniente e incrementar el área de aplicaciones, es necesario que el modelo de apariencia sea capaz de trabajar con diferentes clases de objetos.

Modelo de movimiento: El modelo de movimiento utilizado en esta tesis está basado en un filtro de Kalman. El cual tiene la limitante de generar predicciones erróneas durante los primeros instantes cuando los objetos presentan movimientos no lineales. Por lo que utilizar un

modelo capaz de modelar movimientos no lineales, como las redes LSTM podría incrementar la precisión del algoritmo para el seguimiento de múltiples objetos.

REFERENCIAS

- [1] F. Garcia, A. Ponz, D. Martín, J. M. Armingol, and A. De La Escalera, “Fusión de Escáner Laser y Visión por Computador para la Detección de Peatones en Entornos Viarios,” *Rev. Iberoam. Autom. e Inform. Ind.*, vol. 12, no. 2, pp. 218–229, 2015.
- [2] M. I. Chacon, *Procesamiento digital de imagenes*. Trillas, 2007.
- [3] R. Iguernaissi, D. Merad, K. Aziz, and P. Drap, “People tracking in multi-camera systems: a review,” *Multimed. Tools Appl.*, vol. 78, no. 8, pp. 10773–10793, 2018.
- [4] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu, “Online Multi-object Tracking Using CNN-Based Single Object Tracker with Spatial-Temporal Attention Mechanism,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4846–4855.
- [5] E. Maggio and A. Cavallaro, *Video Tracking Theory and Practice*. Willey, 2011.
- [6] S. Zhang, Y. Zhu, and A. Roy-Chowdhury, “Tracking multiple interacting targets in a camera network,” *Comput. Vis. Image Underst.*, vol. 134, pp. 64–73, 2015.
- [7] X. Li *et al.*, “Multi-Task Structure-Aware Context Modeling for Robust Keypoint-Based Object Tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 4, pp. 915–927, 2019.
- [8] T. Zhang, C. Xu, and M. H. Yang, “Robust Structural Sparse Tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 473–486, 2019.
- [9] Z. Shi, S. Zhu, W. Sun, and B. Wang, “Continuous energy minimization based multi-target tracking,” *Commun. Comput. Inf. Sci.*, vol. 483, no. 1, pp. 464–473, 2014.
- [10] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong, “Enhancing Detection Model for Multiple Hypothesis Tracking,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 2143–2152.

- [11] S. Tang, M. Andriluka, B. Andres, and B. Schiele, "Multiple people tracking by lifted multicut and person re-identification," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3701–3710.
- [12] H. Sheng, Y. Zhang, J. Chen, Z. Xiong, and J. Zhang, "Heterogeneous Association Graph Fusion for Target Association in Multiple Object Tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 11, pp. 3269–3280, 2019.
- [13] C. Li, L. Lin, W. Zuo, J. Tang, and M. H. Yang, "Visual Tracking via Dynamic Graph Learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2770–2782, 2019.
- [14] C. Ma, J. Bin Huang, X. Yang, and M. H. Yang, "Robust Visual Tracking via Hierarchical Convolutional Features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2709–2723, 2019.
- [15] W. Zuo, X. Wu, L. Lin, L. Zhang, and M. H. Yang, "Learning support correlation filters for visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 5, pp. 1158–1172, 2019.
- [16] T. Zhang, C. Xu, and M. H. Yang, "Learning Multi-Task Correlation Particle Filters for Visual Tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 365–378, 2019.
- [17] Y. Qi *et al.*, "Hedging deep features for visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 5, pp. 1116–1130, 2019.
- [18] A. Ali *et al.*, "Visual object tracking—classical and contemporary approaches," *Front. Comput. Sci.*, vol. 10, no. 1, pp. 167–188, 2016.
- [19] L. Lan, X. Wang, S. Zhang, D. Tao, W. Gao, and T. S. Huang, "Interacting Tracklets for Multi-Object Tracking," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4585–4597, 2018.
- [20] T. Yang, C. Cappelle, Y. Ruichek, and M. El Bagdouri, "Online multi-object tracking

- combining optical flow and compressive tracking in Markov decision process,” *J. Vis. Commun. Image Represent.*, vol. 58, pp. 178–186, 2019.
- [21] A. Maksai and P. Fua, “Eliminating Exposure Bias and Loss-Evaluation Mismatch in Multiple Object Tracking,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4634–4643.
 - [22] K. Fang, Y. Xiang, X. Li, and S. Savarese, “Recurrent Autoregressive Networks for Online Multi-object Tracking,” in *IEEE Winter Conference on Applications of Computer Vision*, 2018, pp. 466–475.
 - [23] S. R. Balaji and S. Karthikeyan, “A survey on moving object tracking using image processing,” in *11th International Conference on Intelligent Systems and Control*, 2017, pp. 469–474.
 - [24] M. Ramasubramanian, M. A. D. Rangaswamy, and G. N. V. R. Reddy, “A survey study on detecting and tracking objective methods,” in *IEEE National Conference on Emerging Trends In New & Renewable Energy Sources And Energy Management*, 2014, pp. 159–164.
 - [25] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *ACM Comput. Surv.*, vol. 38, no. 4, 2006.
 - [26] Y. T. Tesfaye, E. Zemene, A. Prati, M. Pelillo, and M. Shah, “Multi-target Tracking in Multiple Non-overlapping Cameras Using Fast-Constrained Dominant Sets,” *Int. J. Comput. Vis.*, vol. 127, no. 9, pp. 1303–1320, 2019.
 - [27] R. Henschel, Y. Zou, and B. Rosenhahn, “Multiple People Tracking using Body and Joint Detections,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 770–779.
 - [28] R. Henschel, L. Leal-Taixe, D. Cremers, and B. Rosenhahn, “Fusion of head and full-body detectors for multi-object tracking,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1509–1518.

- [29] L. Chen, H. Ai, Z. Zhuang, and C. Shang, “Real-time Multiple People Tracking With Deeply Learned Candidate Selection and Person Re-identification,” in *IEEE International Conference on Multimedia and Expo*, 2018, pp. 1–6.
- [30] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, “Tracking without bells and whistles,” in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 941–951.
- [31] Y. Tang, Y. Liu, H. Huang, J. Liu, and W. Xie, “A Scale-Adaptive Particle Filter Tracking Algorithm Based on Offline Trained Multi-Domain Deep Network,” *IEEE Access*, vol. 8, pp. 31970–31982, 2020.
- [32] S. A. Daneshyar and M. Nahvi, “Moving objects tracking based on improved particle filter algorithm by elimination of unimportant particles,” *Optik (Stuttg.)*, vol. 138, pp. 455–469, 2017.
- [33] M. M. Naushad Ali, M. Abdullah-Al-Wadud, and S. L. Lee, “Multiple object tracking with partial occlusion handling using salient feature points,” *Inf. Sci. (Ny)*, vol. 278, pp. 448–465, 2014.
- [34] T. Zhang, C. Xu, and M. H. Yang, “Multi-task correlation particle filter for robust object tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4819–4827.
- [35] Z. Zhou, M. Zhou, and J. Li, “Object tracking method based on hybrid particle filter and sparse representation,” *Multimed. Tools Appl.*, vol. 76, no. 2, pp. 2979–2993, 2017.
- [36] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, “Online multiperson tracking-by-detection from a single, uncalibrated camera,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 9, pp. 1820–1833, 2011.
- [37] Z. Fu, P. Feng, F. Angelini, J. Chambers, and S. M. Naqvi, “Particle PHD Filter Based Multiple Human Tracking Using Online Group-Structured Dictionary Learning,” *IEEE Access*, vol. 6, pp. 14764–14778, 2018.
- [38] M. D. Jenkins, P. Barrie, T. Buggy, and G. Morison, “Selective sampling importance

- resampling particle filter tracking with multibag subspace restoration,” *IEEE Trans. Cybern.*, vol. 48, no. 1, pp. 264–276, 2018.
- [39] W. Hu, X. Li, W. Luo, X. Zhang, S. Maybank, and Z. Zhang, “Single and multiple object tracking using log-euclidean riemannian subspace and block-division appearance model,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 12, pp. 2420–2440, 2012.
- [40] D. G. Dondo, J. A. Redolfi, M. Griffo, G. M. Steiner, L. R. Canali, and S. Member, “Target Tracking System Using Multiple Cameras and Bayesian Estimation,” *IEEE Lat. Am. Trans.*, vol. 14, no. 6, pp. 2713–2718, 2016.
- [41] I. Elafi, M. Jedra, and N. Zahid, “Unsupervised detection and tracking of moving objects for video surveillance applications,” *Pattern Recognit. Lett.*, vol. 84, pp. 70–77, 2016.
- [42] G. Choe, T. Wang, F. Liu, C. Choe, and H. So, “Visual tracking based on particle filter with spline resampling,” *Multimed. Tools Appl.*, vol. 74, no. 17, pp. 7195–7220, 2015.
- [43] J. García, A. Gardel, I. Bravo, J. L. Lázaro, M. Martínez, and D. Rodríguez, “Detección y seguimiento de personas basado en estereovisión y filtro de Kalman,” *Rev. Iberoam. Autom. e Inform. Ind.*, vol. 9, no. 4, pp. 453–461, 2012.
- [44] W. C. Hu, C. H. Chen, T. Y. Chen, D. Y. Huang, and Z. C. Wu, “Moving object detection and tracking from video captured by moving camera,” *J. Vis. Commun. Image Represent.*, vol. 30, pp. 164–180, 2015.
- [45] A. Garcés Matilla, F. Guerrero Peña, A. G. Calvelo, E. H. Silva, and M. M. Belett, “Multiple people tracking in video surveillance applications with low-resolution cameras,” *Rev. Cuba. Ciencias Informáticas*, vol. 11, no. 2, pp. 149–161, 2017.
- [46] P. Mirunalini, S. M. Jaisakthi, and R. Sujana, “Tracking of object in occluded and non-occluded environment using SIFT and Kalman filter,” in *IEEE Region 10 Conference*, 2017, pp. 1290–1295.

- [47] N. Yuan and H. Lisheng, “An Extended Kalman Filter Application on Moving Object Tracking,” in *Proceedings of the 5th International Conference on Electrical Engineering and Automatic Control*, 2016, vol. 367, pp. 1261–1268.
- [48] A. Osep, W. Mehner, M. Mathias, and B. Leibe, “Combined image- and world-space tracking in traffic scenes,” in *IEEE International Conference on Robotics and Automation*, 2017, pp. 1988–1995.
- [49] D. Mitzel and B. Leibe, “Real-time multi-person tracking with detector assisted structure propagation,” in *IEEE International Conference on Computer Vision Workshops*, 2011, pp. 974–981.
- [50] S. Hong, T. You, S. Kwak, and B. Han, “Online tracking by learning discriminative saliency map with convolutional neural network,” in *32nd International Conference on Machine Learning*, 2015, vol. 37, pp. 597–606.
- [51] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M. H. Yang, “Online Multi-Object Tracking with Dual Matching Attention Networks,” in *European Conference on Computer Vision*, 2018, pp. 366–382.
- [52] S. H. Bae and K. J. Yoon, “Confidence-Based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-Object Tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 595–610, 2018.
- [53] H. Kieritz, S. Becker, W. Hubner, and M. Arens, “Online Multi-Person Tracking using Integral Channel Features Hilke,” in *13th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2016, pp. 122–130.
- [54] C. Kim, F. Li, and J. M. Rehg, “Multi-object tracking with neural gating using bilinear LSTM,” in *European Conference on Computer Vision*, 2018, pp. 200–215.
- [55] A. Sadeghian, A. Alahi, and S. Savarese, “Tracking the Untrackable: Learning to Track Multiple Cues with Long-Term Dependencies,” in *IEEE International Conference on Computer Vision*, 2017, pp. 300–311.

- [56] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *IEEE International Conference on Computer Vision*, 2015, pp. 4696–4704.
- [57] G. Wang, Y. Wang, H. Zhang, R. Gu, and J. N. Hwang, "Exploit the connectivity: Multi-object tracking with TrackletNet," in *27th ACM International Conference on Multimedia*, 2019, pp. 482–490.
- [58] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele, "Motion Segmentation & Multiple Object Tracking by Correlation Co-Clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 1, pp. 140–153, 2018.
- [59] L. Chen, H. Ai, R. Chen, and Z. Zhuang, "Aggregate Tracklet Appearance Features for Multi-Object Tracking," *IEEE Signal Process. Lett.*, vol. 26, no. 11, pp. 1613–1617, 2019.
- [60] P. Chu and H. Ling, "FAMNet: Joint Learning of Feature, Affinity and Multi-dimensional Assignment for Online Multiple Object Tracking," in *IEEE International Conference on Computer Vision*, 2019, pp. 6171–6180.
- [61] L. Leal-Taixe, C. Canton-Ferrer, and K. Schindler, "Learning by Tracking: Siamese CNN for Robust Target Association," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, pp. 418–425.
- [62] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1442–1468, 2014.
- [63] J. K. Sunkara, M. Santhosh, S. B. Cherukuri, and L. G. Krishna, "Object tracking techniques and performance measures - A conceptual survey," in *IEEE International Conference on Power, Control, Signals and Instrumentation Engineering*, 2017, pp. 2297–2305.
- [64] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The CLEAR MOT metrics," in *J Image Video Processing*, 2008.

- [65] H. Yang, S. Qu, C. Chen, and B. Yang, “Multiple Objects Tracking with Improved Sparse Representation and Rank Based Dynamic Estimation,” *IEEE Access*, vol. 6, pp. 42264–42272, 2018.
- [66] H. Karunasekera, H. Wang, and H. Zhang, “Multiple Object Tracking With Attention to Appearance, Structure, Motion and Size,” *IEEE Access*, vol. 7, pp. 104423–104434, 2019.
- [67] Y. Tian, A. Dehghan, and M. Shah, “On Detection, Data Association and Segmentation for Multi-target Tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2146–2160, 2019.
- [68] T. Yang and A. B. Chan, “Visual Tracking via Dynamic Memory Networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 1, pp. 360–374, 2019.
- [69] G. Han, Y. Gao, and N. Sun, “Multi-Target Tracking Based on High-Order Appearance Feature Fusion,” *IEEE Access*, vol. 7, pp. 173393–173406, 2019.
- [70] M. Jiang, T. Hai, Z. Pan, H. Wang, Y. Jia, and C. Deng, “Multi-Agent Deep Reinforcement Learning for Multi-Object Tracker,” *IEEE Access*, vol. 7, pp. 32400–32407, 2019.
- [71] S. Sun, N. AKHTAR, H. Song, A. S. Mian, and M. Shah, “Deep Affinity Network for Multiple Object Tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 1, pp. 104–119, 2019.
- [72] M. He, H. Luo, B. Hui, and Z. Chang, “Fast Online Multi-Pedestrian Tracking via Integrating Motion Model and Deep Appearance Model,” *IEEE Access*, vol. 7, pp. 89475–89486, 2019.
- [73] J. Xu, Y. Cao, Z. Zhang, and H. Hu, “Spatial-Temporal Relation Networks for Multi-Object Tracking,” in *IEEE International Conference on Computer Vision*, 2019, pp. 3987–3997.
- [74] M. Arif, F. Ahmad, S. Afzal, and A. Iqbal, *Advances in Deep Learning*. Springer,

- 2018.
- [75] A. F. Gad, *Practical Computer Vision Applications Using Deep Learning with CNNs With Detailed Examples in Python Using TensorFlow and Kivy*. Menoufia: Apress, 2018.
 - [76] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2017.
 - [77] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2012.
 - [78] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
 - [79] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *3rd International Conference on Learning Representations*, 2015, pp. 1–14.
 - [80] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-ResNet and the impact of residual connections on learning,” in *31st AAAI Conference on Artificial Intelligence*, 2017, pp. 4278–4284.
 - [81] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2261–2269.
 - [82] B. Zoph, V. Vasudevan, J. Shlens, and Q. V Le, “Learning Transferable Architectures for Scalable Image Recognition,” in *IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.
 - [83] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, “MOT16: A Benchmark for Multi-Object Tracking,” 2016. [Online]. Available: <http://arxiv.org/abs/1603.00831>.

- [84] W. Li and X. Wang, "Locally aligned feature transforms across views," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3594–3601.
- [85] J. Cowton, I. Kyriazakis, and J. Bacardit, "Automated Individual Pig Localisation, Tracking and Behaviour Metric Extraction Using Deep Learning," *IEEE Access*, vol. 7, pp. 108049–108060, 2019.
- [86] Y. C. Yoon, A. Boragule, Y. M. Song, K. Yoon, and M. Jeon, "Online Multi-Object Tracking with Historical Appearance Matching and Scene Adaptive Detection Filtering," in *15th IEEE International Conference on Advanced Video and Signal-Based Surveillance*, 2019, pp. 1–6.
- [87] Q. Zhou, B. Zhong, Y. Zhang, J. Li, and Y. Fu, "Deep alignment network based multi-person tracking with occlusion and motion reasoning," *IEEE Trans. Multimed.*, vol. 21, no. 5, pp. 1183–1194, 2019.
- [88] C. H. B *et al.*, "Real-Time Visual Tracking and Identification for a Team of Homogeneous Humanoid Robots," in *RoboCup 2016: Robot World Cup XX*, 2017, vol. 9776, pp. 613–624.
- [89] H. Sheng, X. Zhang, Y. Zhang, Y. Wu, J. Chen, and Z. Xiong, "Enhanced association with supervoxels in multiple hypothesis tracking," *IEEE Access*, vol. 7, pp. 2107–2117, 2019.
- [90] W. Zhou and J. Hou, "A New Adaptive Robust Unscented Kalman Filter for Improving the Accuracy of Target Tracking," *IEEE Access*, vol. 7, pp. 77476–77489, 2019.
- [91] C. Wu *et al.*, "Motion guided siamese trackers for visual tracking," *IEEE Access*, vol. 8, pp. 7473–7489, 2020.
- [92] W. Yi, Z. Fang, W. Li, R. Hoseinnezhad, and L. Kong, "Multi-Frame Track-Before-Detect Algorithm for Maneuvering Target Tracking," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4104–4118, 2020.

- [93] M. Baradaran Khalkhali, A. Vahedian, and H. Sadoghi Yazdi, “Multi-target state estimation using interactive kalman filter for multi-vehicle tracking,” *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1131–1144, 2020.
- [94] S. H. Bae, “Online Multi-Object Tracking with Visual and Radar Features,” *IEEE Access*, vol. 8, pp. 90324–90339, 2020.
- [95] Y. M. Song, K. Yoon, Y. C. Yoon, K. C. Yow, and M. Jeon, “Online Multi-Object Tracking with GMPHD Filter and Occlusion Group Management,” *IEEE Access*, vol. 7, pp. 165103–165121, 2019.
- [96] Y. Zhang *et al.*, “Long-Term Tracking with Deep Tracklet Association,” *IEEE Trans. Image Process.*, vol. 29, pp. 6694–6706, 2020.
- [97] J. Martinez, M. J. Black, and J. Romero, “On human motion prediction using recurrent neural networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4674–4683.
- [98] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, “Online multi-target tracking using recurrent neural networks,” in *31st AAAI Conference on Artificial Intelligence*, 2017, pp. 4225–4232.
- [99] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, “Structural-RNN: Deep learning on spatio-temporal graphs,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5308–5317.
- [100] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social LSTM: Human Trajectory Prediction in Crowded Spaces Alexandre,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 961–971.
- [101] M. Emambakhsh, A. Bay, and E. Vazquez, “Deep recurrent neural network for multi-target filtering,” in *MultiMedia Modeling*, 2019, pp. 519–531.
- [102] J. Xiang, G. Zhang, and J. Hou, “Online Multi-Object Tracking Based on Feature Representation and Bayesian Filtering Within a Deep Learning Architecture,” *IEEE*

- Access, vol. 7, pp. 27923–27935, 2019.
- [103] H. Farazi and S. Behnke, “Online visual robot tracking and identification using deep LSTM networks,” in *IEEE International Conference on Intelligent Robots and Systems*, 2017, pp. 6118–6125.
 - [104] X. Li, Y. Liu, K. Wang, Y. Yan, and F. Y. Wang, “Multi-Target Tracking with Trajectory Prediction and Re-Identification,” in *Chinese Automation Congress*, 2019, pp. 5028–5033.
 - [105] N. Engel, S. Hoermann, P. Henzler, and K. Dietmayer, “Deep Object Tracking on Dynamic Occupancy Grid Maps Using RNNs,” in *21st International Conference on Intelligent Transportation Systems*, 2018, pp. 3852–3858.
 - [106] M. Emambakhsh, A. Bay, and E. Vazquez, “Convolutional Recurrent Predictor: Implicit Representation for Multi-Target Filtering and Tracking,” *IEEE Trans. Signal Process.*, vol. 67, no. 17, pp. 4545–4555, 2019.
 - [107] L. Kong, D. Huang, and Y. Wang, “Long-Term Action Dependence-Based Hierarchical Deep Association for Multi-Athlete Tracking in Sports Videos,” *IEEE Trans. Image Process.*, vol. 29, pp. 7957–7969, 2020.
 - [108] Y. Xiao and D. Pan, “Research on robust visual tracker based on multi-cue correlation particle filters,” *IEEE Access*, vol. 8, pp. 19555–19563, 2020.
 - [109] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *IEEE International Conference on Image Processing*, 2016, pp. 3464–3468.
 - [110] H. W. Kuhn, “The Hungarian method for the assignment problem,” *Nav. Res. Logist.*, vol. 52, no. 1, pp. 7–21, 1955.
 - [111] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, “Cascade object detection with deformable part models,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2241–2248.

- [112] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017.
- [113] F. Yang, W. Choi, and Y. Lin, “Exploit All the Layers: Fast and Accurate CNN Object Detector with Scale Dependent Pooling and Cascaded Rejection Classifiers,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2129–2137.
- [114] A. Hornakova, R. Henschel, B. Rosenhahn, and P. Swoboda, “Lifted Disjoint Paths with Application in Multiple Object Tracking,” in *37 th International Conference on Machine Learning*, 2020.
- [115] Y. Zhang, H. Sheng, Y. Wu, S. Wang, W. Ke, and Z. Xiong, “Multiplex Labeling Graph for Near-Online Tracking in Crowded Scenes,” *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7892–7902, 2020.
- [116] J. Peng *et al.*, “TPM: Multiple object tracking with tracklet-plane matching,” *Pattern Recognit.*, vol. 107, 2020.
- [117] G. Braso and L. Leal-Taixe, “Learning a Neural Solver for Multiple Object Tracking,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6246–6256.
- [118] S. Lee and E. Kim, “Multiple object tracking via feature pyramid siamese networks,” *IEEE Access*, vol. 7, pp. 8181–8194, 2019.
- [119] K. Yoon, J. Gwak, Y. M. Song, Y. C. Yoon, and M. G. Jeon, “OneShotDA: Online Multi-Object Tracker with One-Shot-Learning-Based Data Association,” *IEEE Access*, vol. 8, pp. 38060–38072, 2020.
- [120] Y. Xu, A. Osep, Y. Ban, R. Horaud, L. Leal-Taixé, and X. Alameda-Pineda, “How To Train Your Deep Multi-Object Tracker,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6787–6796.
- [121] J. Lee, S. Kim, and B. C. Ko, “Online Multiple Object Tracking Using Rule Distillated

- Siamese Random Forest,” *IEEE Access*, vol. 8, pp. 182828–182841, 2020.
- [122] Y. C. Yoon, D. Y. Kim, K. Yoon, Y. M. Song, and M. Jeon, “Online multiple pedestrian tracking using deep temporal appearance matching association,” *Inf. Sci. (Ny)*., 2020.