Universidad Autónoma de Querétaro

Facultad de Informática

Maestría en Sistemas Computacionales

# Metodología para la identificación y conteo de objetos en movimiento en escenarios exteriores en tiempo real.

Tesis

Que como parte de los requisitos para obtener el Grado de
Maestro en Sistemas Computacionales

Presenta
Gerardo Gudiño García

Dirigido por:
Dra. Ana Marcela Herrera Navarro

Dra. Ana Marcela Herrera Navarro
Presidente

M.S.I. José Alejandro Vargas Díaz
Secretario

M.S.I. Gabriela Xicoténcatl Ramírez
Vocal

M.I.S.D. Carlos Alberto Olmos Trejo
Suplente

Dra. Ma. Teresa García Ramírez
Suplente

Centro Universitario, Querétaro, Qro.
Octubre, 2019
México

## DEDICATORIA

A mi familia

Por su cariño y apoyo durante este proceso de la maestría.

A mi abuelo

Por todo el apoyo, cariño y recursos para poder concluir con mis estudios.

Por esto y por muchas cosas más

Muchas gracias.

## AGRADECIMIENTOS

Principalmente agradezco a mi abuelo, Gerardo Gudiño Vega, por todo el apoyo que me ha dado para poder realizar mis sueños y por brindarme los recursos para concluir con mis estudios de la maestría.

A mi familia por el apoyo durante estos años que invertí en la maestría, a mi papá Alejandro Gudiño Reséndiz, a mi mamá Flor García García, a mi hermana María Guadalupe Gudiño García, a mi abuela Rosa Reséndiz Hernández y a los demás miembros de mi familia cercana.

Al Mtro. José Alejandro Vargas Díaz por la oportunidad brindada para poder estudiar esta maestría y el apoyo otorgado durante la misma.

A la Dra. Ana Marcela Herrera Navarro por la idea inicial y por el apoyo para la realización de este trabajo.

Al Centro de Desarrollo de la Facultad de Informática de la Universidad Autónoma de Querétaro, a todos sus miembros y amigos que han estado y a los que siguen ahí, por su apoyo y preocupación para la finalización de esta tesis, así como el material, espacio y el conocimiento prestado para la realización de la misma.

A Pablo Samano Elton y Ana Laura Peña Hernández por su apoyo y compromiso durante el desarrollo de este trabajo. Esta tesis también es de ustedes.

A Juan Pablo Gutiérrez Oliva, por todo su apoyo, consejos, sugerencias, inspiración y guía al inicio de la maestría. Gracias JP.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el apoyo financiero recibido y a la Universidad Autónoma de Querétaro por las facilidades otorgadas para la realización de este proyecto.

## ÍNDICE

## ÍNDICE DE FIGURAS

## ÍNDICE DE TABLAS

## RESUMEN

La visión por computadora es una herramienta que en los últimos años ha tenido participación en muchas áreas de investigación, esto implica que sea utilizada para resolver problemáticas variadas de acuerdo al propósito para el cual se emplee. Para atender a estos retos, es necesario el poder contar con una guía que haga uso de dicha herramienta y de técnicas para así poder cumplir la tarea que se requiere. En este trabajo se propone una metodología la cual permita identificar y contar ciertos objetos en movimiento en espacios los cuales estén expuestos al aire libre y que sea una detección lo más cercana al tiempo real. Para lograr este objetivo se hacen uso de herramientas que ofrecen las librerías de *OpenCV*, para el pre procesamiento de la imagen, y la librería de detección de objetos llamada *Darknet YOLO.* Ambas partes, con la ayuda del lenguaje de programación Python, trabajan en conjunto en esta metodología, donde tras la realización de pruebas, haciendo uso de una computadora portátil con características normales y a través de la obtención de imagen por medio de una cámara web, donde se logró detectar las regiones que se encuentren en movimiento, reconocer y etiquetar los objetos en tiempo real dentro de un área de prueba en las instalaciones de la Facultad de Informática de la Universidad Autónoma de Querétaro. Dichas pruebas fueron analizadas para determinar cuáles son las mejores condiciones para que la metodología funcione de manera óptima.



(**Palabras clave:** Procesamiento digital de imágenes, visión por computadora, detección de objetos, conteo de objetos).

## SUMMARY

Computer vision is a tool that in recent years has had a significant involvement in many research areas, this implies that it may be used for solving various problematics according to the purpose for which its employed. To attend these challenges, a guide for how to use this technology with its different technics is needed. The present research proposes a methodology for the identification and counting of certain moving objects on free space environments and as close to real-time as possible. To achieve this objective, OpenCV libraries were used for image pre-processing and the Darknet YOLO library was used for object detection. Both of the aforementioned libraries in conjunction with the Python programming language, work together in the proposed methodology, were after the testing, through the use of a normal laptop and the image gathering from a webcam, region moving detection, recognition and labeling of objects in real-time was achieved on a testing area in the Faculty of Informatics of the Autonomous University of Queretaro installations. Such tests were analyzed to determine which are the best conditions to ensure the optimum results of the methodology.



(**Keywords**: Computer Vision, Image Digital Processing, Object Counting, Object Detection.)

# 1. INTRODUCCIÓN

Actualmente, la visión por computadora se ha vuelto una herramienta muy poderosa en muchas áreas de investigación y aplicación, de las cuales se pueden destacar la medicina, la robótica, la agricultura, la biometría, la vigilancia y la seguridad. De éstas últimas, se desglosa un problema y al mismo tiempo un área de oportunidad como lo es el monitoreo, identificación, conteo y seguimiento de objetos en circunstancias varias, como puede ser un cuarto, un almacén, una calle o una esquina, dependiendo el ambiente en el cual se ponga en operación el sistema encargado de dichas tareas.

A este tipo de encomiendas, las acompañan ciertos requerimientos y exigencias, como puede ser la exposición a ruido, a la iluminación, a la concentración de objetos o personas, u otras trabas que impidan la buena ejecución de los sistemas. Para prevenir este tipo de interferencias y obtener datos y resultados más confiables, existen ciertas herramientas, llámense *frameworks* o librerías, las cuales ayudan al tratamiento de las imágenes y éstas puedan ser utilizadas de una manera más limpia para obtener los verdaderos resultados esperados.

Al mismo tiempo, existen dentro de estos *frameworks* o librerías algoritmos básicos que comprenden las necesidades de las imágenes, para así aplicar ciertos filtros para obtener características necesarias como podrían ser el reconocimiento de bordes, eliminación de ruido, ampliar o reducir una imagen o un área de la misma, histograma, convertir el modo de color de las imágenes, detectar objetos, rostros, movimiento, segmentar la imagen, entre otras aplicaciones más.

Con todo este tipo de herramientas y facilidades, se puede resolver el problema el cual se plantea en este trabajo, el cual es el identificar y contabilizar los diferentes objetos que se encuentren en movimiento en una escena en exteriores esperando que se resuelva de una forma rápida para así decir que es

una metodología en tiempo real, ya que las exigencias de los tiempos presentes esperan una respuesta inmediata que pueda ayudar a otros sistemas más expertos u otras circunstancias en la toma de decisiones.

En este trabajo se presenta una metodología para la detección de objetos en movimiento la cual se construye a través del análisis de ciertas herramientas, librerías y algoritmos que se consideren puedan ayudar a resolver la problemática y con ello, por medio de pruebas, se pueda elegir la mejor manera para proponer una metodología en tiempo real.

## 2. DESCRIPCIÓN DEL PROBLEMA

Actualmente el interés por los procesos de reconocimiento de imágenes ha ganado mucha importancia, a medida que la tecnología ha evolucionado y aunque existen aplicaciones robustas de visión, se mantienen problemas a la hora de realizar análisis e interpretar los datos percibidos por maquinas dedicadas a extraer información de imágenes. Dichos problemas se deben a las condiciones de iluminación, contraste, ruido, perturbaciones ambientales y cambio de posición de objetos, por lo que la comprensión de escenas ha ganado mucha importancia ya que muchos métodos no solo se centran en la segmentación, están encaminados al análisis de forma, características, geometría y cuestiones físicas que proporcionen más información sobre las condiciones del objeto, por ejemplo, algunos autores detectan personas como objetos aislados, pero en escenarios reales existen grupos de personas o parejas, siendo este el enfoque de esta propuesta se presenta una metodología que permita identificar y contar mediante ciertas estrategias el número de objetos en un escenario exterior y en tiempo real para así ayudar mediante estadísticos al desarrollo de ciertos sistemas autónomos.

Por otro lado, es necesario considerar y determinar el tránsito peatonal de un determinado lugar donde diferentes personas pueden solaparse lo que dificulta el conteo. En este sentido es conveniente el conocimiento aproximado del número de personas para poder establecer la estrategia para predecir los días con mayor afluencia.

## 3. JUSTIFICACIÓN

Durante los últimos años, la visión por computador ha jugado un papel importante en la percepción visual humana, que en general, se refiere a una capacidad básica del sistema visual para derivar agrupaciones y estructuras correspondientes de una imagen sin conocimiento previo de su contenido, actualmente los desarrollos realizados para reconocimiento de escenas se han enfocado principalmente en detección y seguimiento de personas Ciric, Cojbasic, Nikolic, y Antic (2013), y detección de obstáculos Chen, Guo, y Sun (2010), por mencionar algunos.

La identificación y conteo de objetos en un ambiente externo y en movimiento tiene un amplio campo para su uso, tanto para la detección de ciertos elementos como podría ser una persona, así como un conjunto de los mismos elementos, que podrían ser una multitud. En otro panorama, una metodología de este tipo y en tiempo real, podría apoyar en el desarrollo de sistemas autónomos implementados en ciertos lugares para generar una estadística e información de concurrencia y poder controlar o ayudar en la toma de decisiones basados en el flujo de personas. Esta metodología pretende trabajar con las técnicas y filtros necesarios para llegar al objetivo, superando los principales problemas que hay en el análisis de imágenes que pueden ser la iluminación, color, relación de superficies e información semántica de los objetos, con clasificador que ofrezca

una mejora en la identificación de los objetos en un escenario exterior, en movimiento y en tiempo real.

## 4. ANTECEDENTES

Entre los objetivos que tiene la visión por computadora y el procesamiento de imágenes es la detección, diferenciación y conteo de ciertos elementos que son de interés mediante el uso de algunas técnicas de procesamiento de imágenes basados en algoritmos y técnicas ya establecidas. Estas técnicas son muy variadas y sirven para realizar tareas, ya sea específicas o generales, que juntas ayudan a resolver problemáticas que son objeto de estudio. Entre las tareas más básicas a las cuales se somete una imagen para su procesamiento es el de segmentar y separa los elementos de interés del fondo, en donde el fondo puede ser tanto fijo como en movimiento de acuerdo a las condiciones en las cuales se haya capturado la imagen.

La segmentación en sistemas de visión por computador tiene el fin de subdividir una imagen para separar las partes de interés del resto de la imagen, esta subdivisión depende del problema a resolver, por lo que existen distintos métodos de segmentación de los cuales algunos se basan en características tales como colores, texturas y bordes para descomponer una imagen en regiones uniformes, dentro de estos métodos, expertos han realizado investigaciones con el fin de realizar reconocimiento de objetos, a continuación se describen algunos de los métodos desarrollados.

Wang, Bo y Wang (2010) diseñaron un algoritmo no supervisado para adaptarse automáticamente a las condiciones del suelo en diferentes habitaciones donde supone que el suelo es la mayor superficie y tiene las mismas características de color y textura. Por su parte, Husain, Schulz, Dellen, Torras, y Behnke (2016) propusieron un método basado en el aprendizaje, fundamentado

en una arquitectura *multi-pooling* que toma el color de la imagen, la codifica de acuerdo a su disparidad horizontal, altura sobre el suelo, y el ángulo de los píxeles y añade la distancia a la pared lo que permite que el modelo aprenda una representación más detallada de una escena. El método de Hermans, Floros y Leibe (2014) está fundamentado en un número de árboles binarios donde un píxel atraviesa el árbol y el camino decidido se basa en un rasgo y un umbral almacenado en cada nodo, una vez que el pixel alcanza un nodo, es asignado a la distribución de clase correspondiente, el resultado final es obtenido por hacer un promedio de las distribuciones de los nodos diferentes. Mientras que el método de Bhanu y Peng (2000) se basa en adaptar los parámetros del algoritmo de segmentación a las condiciones ambientales, en combinación con la medida de borde de frontera para la evaluación de la segmentación, con el fin de reducir los gastos de cálculo asociados con el modelo durante la etapa de adaptación.

La segmentación mediante la fusión de color y profundidad de los objetos presentada por He y Upcroft (2014), modela el problema de manera similar a las rutinas estándar de segmentación como un campo aleatorio de Markov y la llevan a cabo mediante optimización de cortes gráficos, determinan semillas de píxeles en el primer fotograma que luego son re proyectados automáticamente en las tramas restantes y de esta forma segmentan los objetos en un escenario. Mientras que, el algoritmo propuesto por Siricharoen, Aramvith, Chalidabhongse y Siddhichai (2010) está basado en el acercamiento estadístico para la substracción de fondo en tiempo real y la detección de la sombra (SBGS) en el cual, la base de borde y la intensidad de fondo son modelados por SBGS. El rasgo en color, el rasgo de borde y la substracción de intensidad de fondo son integrados para alcanzar el mejor funcionamiento de segmentación. Dentro de esta perspectiva, Blas, Agrawal, Sundaresan y Konolige (2008) presentan un descriptor de color y textura el cual se utiliza en un marco de agrupamiento rápido de dos etapas utilizando *K-means* para realizar la segmentación de imágenes naturales en línea.

Todo el sistema trabaja en línea junto con la localización, detección de obstáculos en 3D, y la planificación.

El método propuesto por Kim, Trinh y Jo (2007) se sustenta en múltiples señales las cuales son color, una línea directa, bordes, puntos límites, *PCs* y *HCM* (*Hue Co-occurrence Matrix*). Combinando dichos rasgos, la imagen se segmenta en varias regiones, como la mezcla usa rasgos y métodos *HCM*, los objetos pueden ser descubiertos cuando combinan múltiples señales predefinidas. Por otra parte, Gupta, Girshick, Arbeláez y Malik (2014) proponen una nueva profundidad geocéntrica de imágenes que codifica la altura sobre el suelo y el ángulo con la gravedad para cada píxel, además de la disparidad horizontal. Estas características se usan para el aprendizaje de redes neuronales convolucionales.

Wan, Sun y Model (2015) proponen un método de segmentación de imágenes en escala de grises para la detección del contorno de objetos separando el fondo de la imagen, este método supera el ruido y la intensidad gris no homogénea. Por su parte, Delakis y Garcia (2003) proponen una segmentación basada en el reconocimiento facial mediante una arquitectura neuronal convolucional, en la cual se tiene un sistema entrenado con un total de 507 caras frontales, este método demuestra que el uso de una arquitectura de este tipo, mejora significativamente la detección de rostros, con una aplicación de bajo costo y en tiempo real y con la capacidad de detectar rostros con un giro de hasta 60 grados.

Para Rahaman, Hasan, Ahmed, Maswood y Rahman (2014), la característica principal de este método se basa en obtener un paquete de imágenes de fondo pasado y fondo futuro, los cuales son comparados y generan un historial de mediana para los pixeles de los fotogramas, detectan cambios entre fotogramas pasados y actuales para detectar movimiento. Mientras que, Xin, Tian, Wang y Gao (2015) proponen un algoritmo mediante la ampliación del *ALM* (*Augmented La- grange Multiplier*) para explotar la información de la estructura de los primeros planos y ayudar a modelar el fondo, además de ciertos supuestos,

como el bajo rango o la suposición de escasa-composición (en función de si los marcos de fondo puro se proporcionan) y formulan la extracción de fondo como un problema de descomposición de la matriz usando términos de regularización. Por otra parte, Yingying Chen y National (2015) proponen un sistema de sustracción de fondo basado en (*Gaussian mixture modeling*), en el cual cada píxel busca de forma dinámica el mejor modelo, los modelos compartidos se construyen para la imagen de fondo y la de primer plano, se descartan pequeños los movimientos locales (provocados por ruido) y se añade un mecanismo flexible para la conmutación entre el fondo y los modelos de primer plano.

El método de Liu, Lin, Yang y Huang (2013) está basado en construir modelos de antecedentes que pueden resistir los cambios de escena, proponen un nuevo descriptor binario que presenta eficacia y eficiencia con respecto a los cambios de iluminación y comparan los modelos de fondo con imágenes actuales para extraer los objetos de primer plano. Finalmente, el modelo de fondo propuesto por Subudhi, Ghosh y Ghosh (2013) se construye mediante el análisis de una secuencia de cuadros de imagen linealmente dependientes en un marco Wronskiano, que se utiliza para detectar los cambios entre la escena del fondo y la escena de destino. El modelo de fondo construido se compara con diferentes cuadros de imagen de la misma secuencia para detectar objetos en movimiento.

Danciu (2017) propone un algoritmo basado en la segmentación mediante blobs, para separar objetos de un fondo, también hace uso de métodos de esqueletización, para ello se trabaja con imágenes binarias y la forma en la que funciona el algoritmo es calculando la distancia entre los objetos para después calcular y trabajar con los contornos de los esqueletos, extraer valores máximos para conservar su ubicación y mediante el algoritmo de *K-means* encontrar el centro de cada uno de los objetos para su identificación, separación en regiones o como medida estadística. Por su parte Kong, Akakin, y Sarma (2013) propone un algoritmo generalizado en el cual hace uso de un filtro Laplaciano y Gaussiano (*LoG*) para la detección de blobs en una imagen, el filtro se aplica antes del

proceso de detección para suavizar la imagen y atenuar el ruido mediante el paso gaussiano, mientras que el paso laplaciano destaca las regiones para el reconocimiento de bordes, con este algoritmo se puede mejorar el rendimiento para la detección de blobs de objetos en una imagen con dos vías de aplicación, para biomédica y para imágenes naturales para la detección de puntos de interés.

La tarea de segmentación de escena requiere conocimiento necesario para la extracción de las zonas de imagen y objetos, toda la información adquirida al momento de capturar la imagen y procesarla debe ser fiable para garantizar la calidad de la información, por lo que se requieren de distintos filtros y pre procesamiento para remover defectos, problemas por movimiento o desenfoque, mejorar ciertas propiedades como color, contraste, estructura, etc. Por tal, se requieren de los conceptos que a continuación se describen para procesar y extraer características de una imagen para su descripción e interpretación por computador.

# 5. FUNDAMENTACIÓN TEÓRICA

## 5.1 Imagen digital

Una imagen digital es aquella que puede ser manipulada a través de un equipo informático, para obtener o realizar ciertos propósitos. Existen diferentes formatos y modos de color, los cuales interpretan el valor de cada pixel utilizando valores numéricos (Bits y Vectorial, n.d.).

Una imagen numérica $I$ es una aplicación de un dominio $D_I \subset Z^2$ con valores en $Z$.

$$I = \begin{cases} D_I \to z \\ x \to I(x) \end{cases} \quad (1)$$

Una imagen (ver Figura 1) se puede representar como una matriz donde el valor de cada elemento representa la intensidad luminosa representada por valores comprendidos en el rango de $(0{,}255)$.

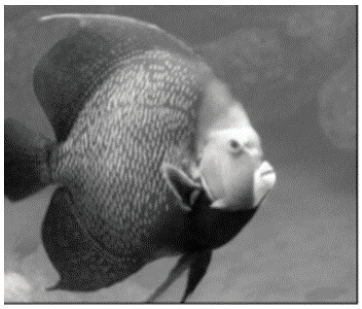


**Figura 1.** Representación de una imagen numérica. Fuente: Elaboración propia (2018).

Una imagen numérica también puede ser representada como un relieve topográfico (ver Figura 2).

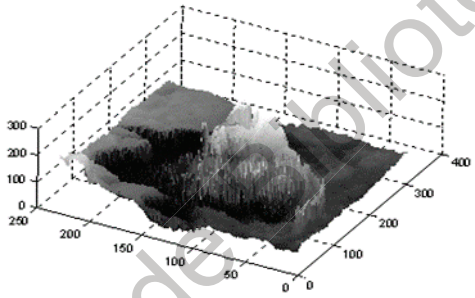



**Figura 2.** Representación de una imagen numérica como un relieve topográfico. Fuente: Elaboración propia (2018).

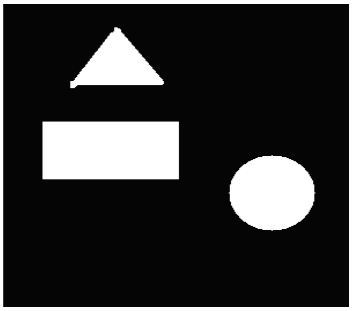
Cuando el rango de la imagen está limitado a solo dos valores $\{0,1\}$ se obtienen imágenes binarias. Una definición formal de una imagen binaria está dada a continuación.

5.1.1 Imagen Binaria

Es una aplicación de un dominio $D_I \subset Z^2$ con valores en $\{0,1\}$

$$I = \begin{cases} D_I \to z \\ x \to I(x) \end{cases} \quad (2)$$

En la práctica una imagen binaria puede ser representada como una matriz donde el valor de cada elemento (píxel) es representado por 2 tonalidades {0,1} como se muestra en Figura 3.



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

**Figura 3.** Representación de una imagen binaria. Fuente: Elaboración propia (2018).

5.1.2 Escala de Grises

Se toma la imagen y se maneja en sólo un canal, el negro. Esto permite una variación de entre 0 y 255, aplicando un tono gris entre un blanco y un negro puro.

5.1.3 CMYK

Utiliza 4 canales, cada uno de ellos corresponde a un color primario de impresión: *cyan* (cian), *magent* (magenta), *yellow* (amarillo) y *black* (negro). Los valores se ubican entre 0 y 255.

5.1.4 RGB

Es el modelo básico que utiliza las componentes primarias rojo, verde y azul, normalizadas, orientado a equipos como cámaras y receptores de televisión (Antonio, Pérez, Javier y Rojas, n.d.). Se puede representar como un “cubo” con un primario en cada eje (Figura 4).

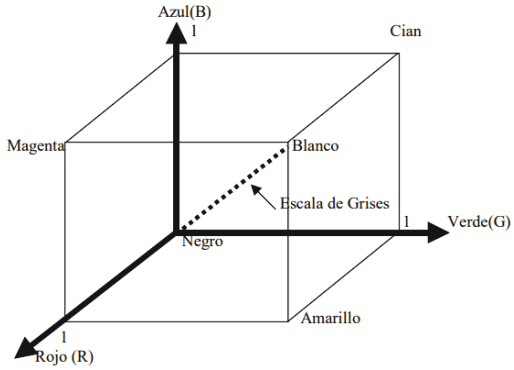



**Figura 4.** Representación del modelo RGB. Fuente: (Sucar, n.d.)

## 5.2 Filtrado

El filtrar una imagen (f) consiste en aplicar una transformación $T$ para obtener una nueva imagen $g$ de forma que ciertas características son acentuadas o disminuidas:

$$g(x,y) = T[f(x,y)] \tag{3}$$

De acuerdo a la teoría de sistemas, al pasar una señal por un sistema lineal, la salida *g (x, y)* es la convolución de la transformación del sistema *h (x, y)* (función de transferencia) con la señal de entrada *f (x, y)*.

$$g(x,y) = h(x,y) * f(x,y) \tag{4}$$

Por el teorema de la convolución, esto corresponde a la multiplicación en el dominio de la frecuencia:

$$G(u,v) = H(u,v)F(u,v) \tag{5}$$

Por esto, podemos pensar en dos formas básicas de filtrar una imagen, realizarlo en el dominio espacial o en el dominio de la frecuencia. Una vez realizado este proceso es importante analizar solo las regiones de interés, por lo que la segmentación es un método que ayuda a dividirla en regiones.

### 5.2.1 Filtros de suavizado

El objetivo de los filtros de suavizado es eliminar ruido o detalles pequeños que no sean de interés. En la Figura 5, se muestra la respuesta de un filtro pasa-bajo en frecuencia que elimina o reduce las altas frecuencias. Los tipos más comunes son:

- Promedio o media aritmética: Obtiene el promedio de los pixeles vecinos (w = 1); es decir, todos los valores de la máscara son 1.
- Mediana: Substituye el valor del pixel central por el de la mediana de los valores contenidos en el vecindario.
- Gaussiano: Aproximación a una distribución *Gaussiana* en dos dimensiones. Considerando una media igual a cero, la función de transformación de un filtro tipo gaussiano

$$T(x, y) = e^{-\left[\frac{(x^2+y^2)}{2\pi\sigma^2}\right]} \quad (6)$$

Donde σ es la desviación estándar.

**Figura 5.** Imagen con ruido (izquierda), aplicación de filtro gaussiano (derecha). Fuente: Elaboración propia (2018).

### 5.2.2 Contraste de una imagen

En esta etapa se aplicará la ecualización del histograma, la cual es un método que consiste en una transformación no lineal que considera la distribución

de los píxeles de la imagen original, actuando en áreas muy claras o muy oscuras de una imagen para generar un histograma de forma más uniforme correspondiente a la imagen que se le ha aplicado el método (Figura 6). Esta transformación se presenta como:

$$S_K = \sum_{j=1}^{k} \frac{n_j}{n} \tag{7}$$

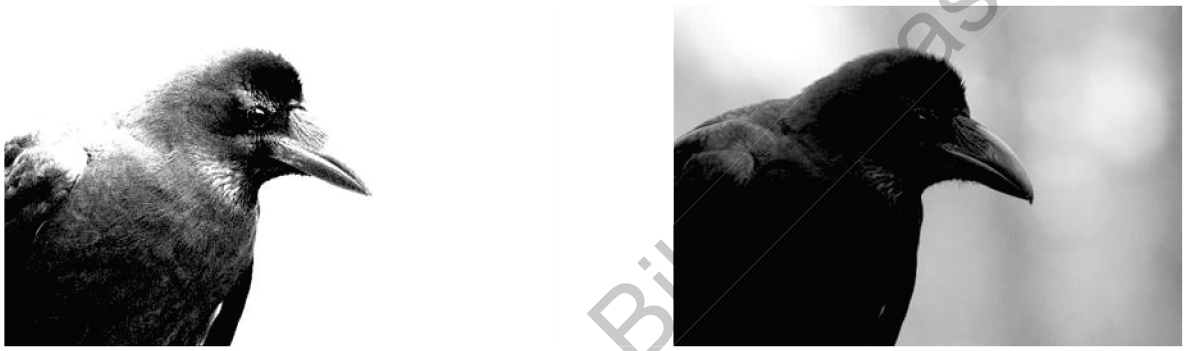
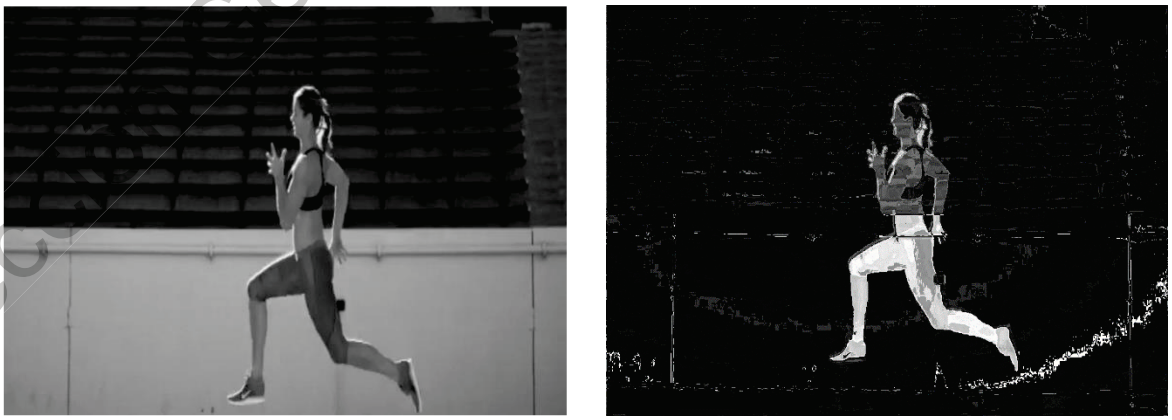


**Figura 6.** Imagen con exceso de brillo (izquierda), aplicación de ecualización de contraste y un filtro gaussiano (derecha). Fuente: Elaboración propia (2018).

### 5.2.3 Aplicación de un modelo de fondo

En esta etapa se aplican diferentes modelos de fondo (diferencia de gaussianas, diferencia temporal), como se aprecia en la Figura 7.

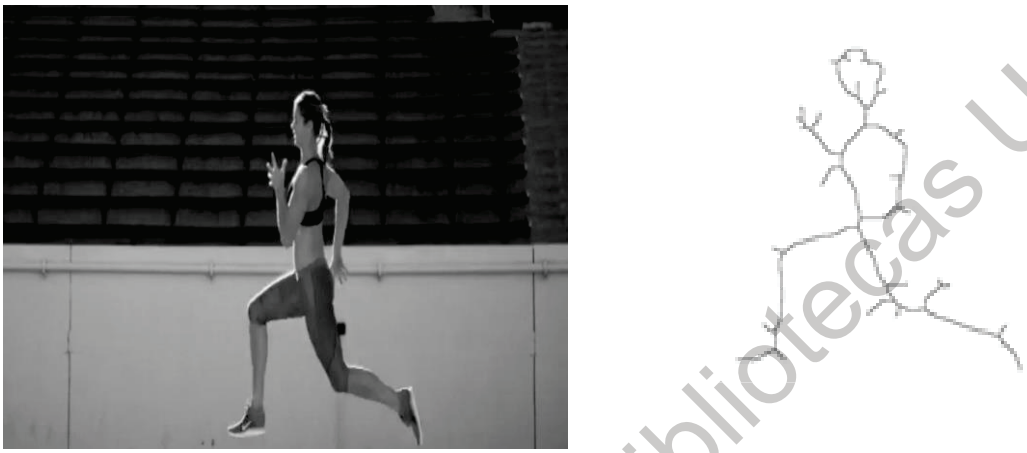


(a) (b)

**Figura 7.** (a) Imagen original, (b) aplicación de modelo de fondo para extraer características de personas. Fuente: Elaboración propia (2018).

5.2.4 Detección de esqueletos

En esta etapa se obtienen las estructuras de objetos mediante los esqueletos. En la Figura 8 se muestra el esqueleto usando Morfología Matemática.



(a) (b)

**Figura 8.** (a) Imagen original, (b) esqueleto obtenido a partir de la imagen (a). Fuente: Elaboración propia (2018).

5.3 Segmentación

La segmentación de la imagen se considera que es uno de los problemas fundamentales para la visión por computador. Un objetivo principal de la segmentación es dividir la imagen en regiones con propiedades coherentes, de modo que cada región corresponda a un objeto o área de interés, existen varias técnicas para la segmentación que se pueden clasificar de forma siguiente:

5.3.1 Segmentación por histograma

La segmentación por histograma (*Thresholding*) es una técnica global que se basa, inicialmente, en asumir que hay un sólo objeto sobre un fondo uniforme. Por esto se consideran dos regiones en la imagen y para dividirlas se toma como base el histograma de intensidades Raju y Neelima (2012).

Podemos asumir que si hay dos regiones se tiene dos picos en el histograma. Entonces se toma el valle (mínimo) entre los dos y este se considera

la división entre las dos regiones. De esta forma todos los pixeles que correspondan a un lado del histograma se toman como una región y el resto como otra, como se ilustra en la Figura 9.

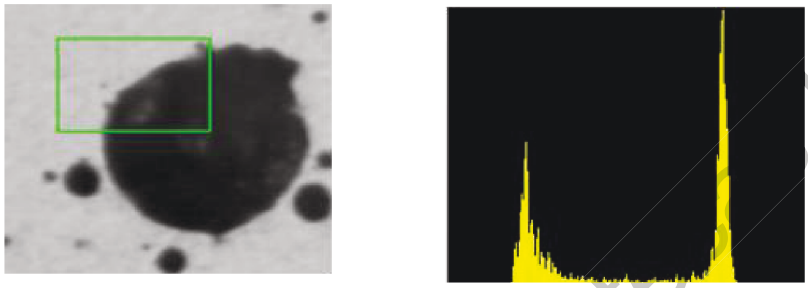


**Figura 9.** Sección de imagen (izquierda), histograma (derecha). Fuente: Elaboración propia (2018).

### 5.3.2 Segmentación por Umbralización OTSU

Se emplea cuando hay una clara diferencia entre los objetos a extraer respecto del fondo de la escena Liao, Chen, y Chung (2001). Los principios que rigen son la similitud entre los píxeles pertenecientes a un objeto y sus diferencias respecto al resto. Por tanto, la escena debe caracterizarse por un fondo uniforme y por objetos parecidos Otsu (1979).

Al aplicar un umbral, T, la imagen en escala de grises, *f (x, y),* quedará binarizada; etiquetando con '1' los píxeles correspondientes al objeto y con '0' aquellos que son del fondo. Por ejemplo, si los objetos son claros respecto del fondo, se aplicará:

$$g(x,y) = \begin{cases} 1 \; si \; f(x,t) > T \\ 0 \; si \; f(x,t) \leq T \end{cases} \quad (8)$$

En el caso de que los objetos sean oscuros respecto del fondo, la asignación sería a la inversa:

$$g(x,y) = \begin{cases} 1 \; si \; f(x,t) < T \\ 0 \; si \; f(x,t) \geq T \end{cases} \quad (9)$$

El umbral puede depender de f (x, y), de alguna propiedad local del píxel, p (x, y), y hasta de su propia posición:

$$T = T(f(x,y), p(x,y), x, y) \quad (10)$$

5.3.3 *Blobs*

Los *blobs* son partes de la imagen representados por pixeles los cuales resaltan por ser superiores a un umbral dado y que tienen cierta relación entre sí. Con ellos es más sencillo detectar objetos dentro de una imagen o región de interés.

Los blobs se pueden extraer por medio de ciertas librerías que tienen relación con otras especializadas en el procesamiento de imágenes. En este caso, con *OpenCV*, la librería a utilizar es *cvBlobslib*, con ella se puede extraer, manipular, filtrar y obtener diferentes características de los Blobs dentro de imágenes binarias y con los *Blobs* obtenidos se conservan únicamente los objetos que interesan de la imagen (De Detección Implementación Y Seguimiento De Objetos En Redes, n.d.).

5.4 Dilatación y erosión

La dilatación aplica las transformaciones morfológicas para comprobar que el elemento está al menos contenido en un pixel de un conjunto de pixeles. Si es así, el pixel se pone en uno. Con ello se logra que el área del objeto se expanda y se rellenen las discontinuidades o vacíos menores. De forma contraria, la aplicación continua ocasiona que los elementos se expandan, se deformen y ocupen toda la imagen.

La dilatación de $X$ por $B$ es el conjunto de los puntos $x$ en $Z^2$ tal que la intersección entre $X$ y $B_x$ es no vacía. El conjunto se denota por:

$$X \oplus \tilde{B} = \{x \in Z^2, X \cap B_X \neq \emptyset\} \quad (11)$$

Por su parte, la erosión es aplicar en una imagen cierta transformación morfológica para ver si un elemento está completamente contenido en un conjunto de pixeles. De esta manera, si un pixel no cumple con la limitación impuesta, se pone en cero. Esto hace que el área del objeto disminuya y desaparezcan los elementos más pequeños que no pertenezcan al elemento principal o de interés. Si se aplica repetitivamente, se corre el riesgo de que en la imagen desaparezcan todos los objetos de la imagen.

La erosión se define como se expresa a continuación:

$$X\Theta\tilde{B} = \{x \in Z^2, B_x \in X\} \quad (12)$$

Si se aplica en el orden de erosionar y luego dilatar, se realiza una apertura, que es cuando el elemento se limpia de los pequeños objetos que interfieren en la imagen y después se dilata para regresar a su tamaño real, pero ya sin las imperfecciones. Cuando la combinación se realiza de forma inversa, dilatar y luego erosionar, se logra rellenar los huecos en la imagen o el elemento haciendo más grande el elemento, entonces se aplica la erosión para disminuir el tamaño y regresarlo a su estado normal, pero ya rellenado, esto se conoce como cierre (Rossius y Bosch Roig, n.d.).

## 5.5 Rectángulo delimitador mínimo

Se trata de un rectángulo el cual se define por el área de interés en la imagen, en donde están los puntos que no son negros y por lo cual delimita el área que se irá a procesar. El tamaño del rectángulo puede variar de acuerdo al tamaño del elemento que contenga. Esta práctica puede detectar rectángulos pequeños que podrían representar algo, pero por lo pequeños es complicado analizar y procesar, por lo cual se debe de determinar un límite para el tamaño mínimo de los rectángulos, el cual se debe de analizar ( Nicolás, 2013).

## 5.6 Técnicas de extracción de fondo

La sustracción de fondo aplicado a secuencias de imágenes sirve para la detección del primer plano; esto quiere decir intentar separar todos los cambios que ocurren en primer plano con el fin de detectar movimiento, para ello se utilizan distintas técnicas de análisis de sustracción de fondo de las cuales solo se explicarán algunas de ellas.

### 5.6.1 Diferencia de imágenes

Este método consiste en comprobar la diferencia entre una imagen de referencia $I_k$ y una nueva imagen $I_{K+1}$ la cual será la imagen actual, el método de diferencia es el método común de detección de movimiento. Este método adopta la diferencia basada en píxeles para encontrar el objeto en movimiento Singla, (2014). Este método es muy sensible al umbral Th.

$$Th < |I_k - I_{k+1}| \tag{13}$$

### 5.6.2 Fondo como promedio o mediana

Este método consiste en estimar el fondo acumulando $n$ imágenes y calcular el promedio o mediana para cada pixel Chowdhury, Cho, y Chong (2011). Este método es bastante rápido, pero consume mucha memoria la cual se determina en base a la siguiente ecuación:

$$memoria = n * tamaño(imagen) \tag{14}$$

Para determinar el promediado del fondo:

$$B_{(i+1)} = \alpha * F_i + (1 - \alpha) * B_i \tag{15}$$

Donde $B$ es la imagen de fondo, $F$ la imagen de primer plano, α la velocidad de aprendizaje (0.05) e i denota la iteración anterior.

5.6.3 Mezcla de Gaussianas

Es un algoritmo iterativo que se inicia desde una estimación inicial de θ y, a continuación, procede a actualizar iterativamente θ hasta que se detecta la convergencia. Cada iteración consiste en un paso E (denota el parámetro actual de θ) y un paso M (ciertos pesos y los datos para calcular nuevos valores de parámetros) y a probabilidad de observar el valor actual pixel se considera dado por la siguiente fórmula en el caso multidimensional.

$$P(X_t) = \sum_{i=i}^{K} \omega_{i,t} \cdot \eta(X_t, \mu_{i,t}, \sum_{i,t}) \qquad (16)$$

Cada píxel del modelo de mezcla gaussiana necesita ser actualizada continuamente y entrenado, al estar actualizado deberán ser normalizados los componentes gaussianos Bouwmans, El Baf, y Vachon (2008).

5.7 Esqueletos

Un esqueleto, conocido también como eje centro, es una estructura que describe la topología y la geometría de la forma. La cual representa una alternativa para manipular la estructura de una forma (Figura 10).

(a) (b)

**Figura 10.** Representación de esqueletos. Fuente: Elaboración propia (2018).

Los esqueletos presentan algunas de las siguientes propiedades:

- Un esqueleto es invariante bajo las transformaciones lineales (traslación, rotación y cambio de escala).

- Un esqueleto es una transformación homotópica: conserva las propiedades topológicas de la forma.
- Un esqueleto es una transformación semi-continua. Esto significa que la más mínima alteración en el contorno o en el interior de la forma puede producir la creación de una rama importante en el esqueleto.
- Para cada punto del esqueleto debería haber una distancia igual de al menos dos diferentes puntos de la forma a una superficie *S*.

5.8 Canny

El filtro *Canny* es utilizado para detectar todos los bordes existentes en una imagen y es considerado como uno de los mejores métodos para la detección de bordes empleando la primera derivada y el uso de máscaras de convolución que representan diferencias finitas de los puntos de contorno (pixeles) cuando existe un cambio repentino a nivel de grises.

En la localización de bordes existe una distancia mínima de los bordes reales con los bordes detectados, esta detección se basa en un gradiente (generalmente Sobel) y en un cierto nivel de detección llamado *Threshold*.

Está basado en 3 criterios:

1. Detección: Suprime los bordes que no se encuentren en el rango de los máximos.
2. Localización: Con la máscara de convolución detecta cada punto de la imagen dependiendo de cada pixel vecino.
3. Respuesta: Conecta los bordes que se suprimieron a los pasaron el paso de detección.

Tiene como principal ventaja su gran adaptabilidad para poder ser aplicado a diversos tipos de imágenes, además de no disminuir su rendimiento ante la presencia de ruido en la imagen original. Tiene como desventaja el hecho

de que al realizar el suavizado de la imagen se pueden difuminar ciertos bordes, aunque con eso se consiga reducir el ruido.

El algoritmo de Canny implementa una técnica de análisis de componentes conectados basada en una heurística de umbral de histéresis.

Este paso utiliza dos umbrales, $t1, t2$

a. dónde $t1 > t2$, para dividir los píxeles de la cresta en bordes/no-bordes. Pixeles con magnitudes de gradiente arriba.
b. $t1$ Se clasifican como aristas definidas.
c. Píxeles entre $t1$ $y$ $t2$, se clasifican como posibles aristas.
d. Pixeles debajo $t2$ se clasifican como no-bordes.

A continuación, todos los bordes potenciales que se pueden remontar a un borde definido a través de los bordes potenciales adyacentes también se marcan como bordes definidos.

El proceso resuelve algunos de los problemas asociados con el rayado de bordes y la discontinuidad en los resultados logrados por los detectores simples al identificar bordes fuertes y al mismo tiempo por los comparativamente más débiles.

## 5.9 Clasificadores

La clasificación es un procedimiento que permite asignar un conjunto de píxeles a categorías específicas; de acuerdo a su valor de gris Weis, Rumpf, Gerhards, y Plümer, (2009). Existen distintos métodos para discriminar objetos y clasificarlos, a continuación, se mencionarán algunos.

### 5.9.1 Clasificación empleando *K-Means*.

Este método se basa determinar las medias de las clases y de forma iterativa los píxeles son insertados en las clases más cercanas utilizando la

técnica de mínima distancia. En cada iteración se recalcula la media de la clase y se vuelven a reclasificar todos los píxeles. Todos los píxeles serán clasificados si se limita la desviación estándar o la distancia máxima de búsqueda (Rocha, Anderson y Goldenstein, 2009).

5.9.2 Clasificación por distancia mínima

Consiste en la determinación de las medias de cada clase y la asignación se realiza hacia la clase con menor distancia. Algunos de los píxeles quedarán sin clasificar si se introduce una distancia máxima o una desviación estándar máxima. Utilizar varias distancias de limitación y otras sin limitación (Weis, Rumpf, Gerhards, y Plümer, 2009).

También es necesario conocer ciertos tipos algoritmos que ayuden a la detección del movimiento como tal, éstos hacen uso de otras funciones o métodos para su funcionamiento e implementación.

5.10 Algoritmos de detección de movimiento basados en modelos y no basados en modelos

Los algoritmos basados en modelos son aquellos que utilizan la recopilación predeterminada de seguimiento de un objeto o una persona para realizar aproximaciones de predicciones de movimiento. Sigue una serie de pasos entre ellos la postura del modelo contando la información previa. Después de determinar la postura, el modelo se sintetiza y se proyecta en una imagen para realizar la comparación. Mediante esto y una función que funja como evaluador, se realizan diferentes estrategias de búsqueda y con ello se va actualizando el modelo.

- Construcción del modelo.
- Representación del modelo.
- Predicción y estrategia de búsqueda.

Por su parte, los algoritmos no basados en modelos, no necesitan de conocer la información estructural del objeto, por ello son menos robustos y tienen un costo computacional menor. Entre los algoritmos no basados en modelos, resaltan los siguientes.

- Detección basada en regiones.
- Detección basada en contornos activos.
- Detección basada en rasgos.

De entre estos, el tipo de algoritmos de interés y estudiados para esta metodología son los basados en regiones y los basados en contornos activos.

Los basados en regiones, hacen uso de los *Blobs* dentro de una imagen, los blobs representan al objeto o los objetos de interés. La parte del fondo de la imagen debe de ser tratado previamente para que éste no interfiera con la región mencionada. El uso de redes neuronales en la localización e identificación del objeto de interés es muy eficiente, sobre todo en las imágenes que son de baja resolución o que han salido borrosas. Un objeto detectado en un *frame*, es buscado en el *frame* siguiente, al detectar cierta similitud, se dibuja un delimitador, en este caso podría ser un rectángulo y con ello se obtienen las coordenadas dentro de la imagen, esto es sólo para regiones y es muy confiable a la hora de detectar movimiento.

Por su parte, por medio de contornos activos, extrae el contorno de los objetos delimitando y actualizando cada uno de los *frames* sucesivos. De esta forma se reconoce al objeto y al movimiento del mismo, así como extraer de una forma más efectiva la forma y descripción del objeto. Este tipo de algoritmos son computacionalmente menos complejos que los algoritmos anteriores basados en regiones (Ecnica, 2010).

## 6. HIPÓTESIS

### 6.1 Hipótesis

Mediante una metodología que aplique técnicas y algoritmos ya establecidos es posible identificar y contar objetos en movimiento como personas, vehículos y animales en espacios exteriores y en tiempo real de acuerdo a ciertas características (longitud, ángulo, forma, etc.).

## 7. OBJETIVOS

### 7.1 Objetivo General

Diseñar una metodología que permita identificar y contar objetos en exteriores en tiempo real mediante modelos establecidos.

### 7.2 Objetivos parciales o específicos

- Pre procesar la imagen mediante filtros para su análisis y métodos de mejoramiento de contraste.
- Determinar un modelo de fondo para localizar regiones de interés en la imagen.
- Determinar un modelo para identificar objetos de interés en la imagen.
- Contar los objetos mediante los parámetros establecidos.

# 8. TECNOLOGÍAS PARA EL DESARROLLO Y LA IMPLEMENTACIÓN DE LOS ALGORITMOS PLANTEADOS

## 8.1 *OpenCV*

Es una librería que ofrece funciones para el tratamiento de imágenes y visión por computadora. Funciona en diferentes sistemas operativos como Mac OSX, Windows y Linux. Entre sus principales funciones están: captura de imágenes y video en tiempo real, importación de imágenes y video, tratamiento básico de imágenes, detección de objetos, aplicación de filtros o máscaras, optimización de algoritmos para el procesamiento de imágenes, etc. (Kaehler y Bradski, n.d.).

## 8.2 *Darknet*

*Darknet* es un marco de red neuronal de código abierto escrito en C y CUDA. Desarrollado por Joseph Redmond en 2013. Su objetivo es realizar detección de objetos a tiempo real a través de YOLO (*You Only Look Once*) (Darknet, 2018).

El uso de *Darknet* se ha especificado últimamente en la utilización de YOLO (*You Only Look Once*), por ejemplo, Redmon y Farhadi (2017) en YOLO9000, en la cual ejemplifican la implementación del framework para detección de objetos en general en tiempo real (Figura 11).

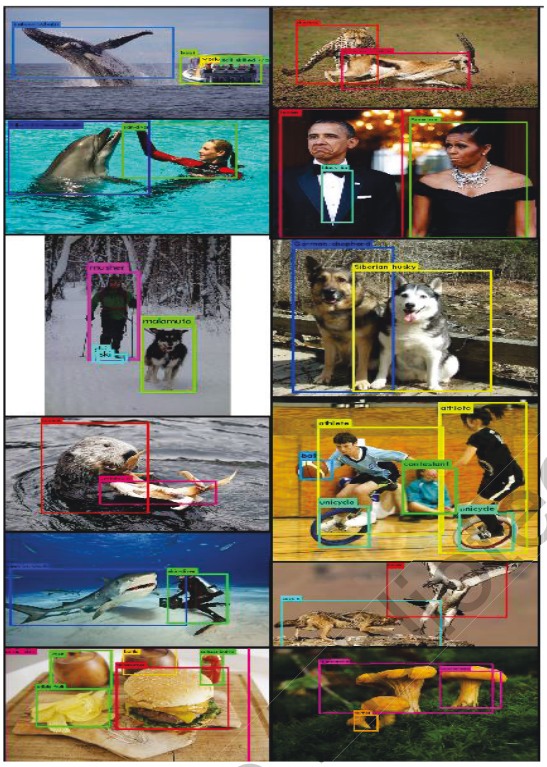


**Figura 11.** YOLO9000 puede detectar una gran variedad de objetos en tiempo real. Fuente: Redmon y Farhadi (2017).

*Darknet* utiliza como base principal el modelo de colores RGB (Red, Green, Blue) para el procesamiento digital de imágenes, además de que permiten la transformación de otros modelos de colores como *CMYK*, *HSB*, *Lab* y *RYB* a *RGB*, esto para que la base de colores no sea una limitante al momento de trabajar con imágenes de diferentes modelos de colores.

## 8.3 *TensorFlow*

*TensorFlow* es una biblioteca de software libre que se utiliza para realizar cálculos numéricos mediante diagramas de flujo de datos. En su origen, *TensorFlow* fue fruto del trabajo de investigadores e ingenieros de *Google Brain Team* que formaban parte de la organización de investigación del aprendizaje automático de *Google*. Su objetivo era realizar investigaciones en el campo del aprendizaje automático y las redes neuronales profundas (TensorFlow, 2014).

Un ejemplo del uso de este *Framework* es el Reconocimiento de Texto Manuscrito con *Deep Learning* (Figura 12), el cual reconoce números, palabras y líneas, esto permite un resultado eficaz al realizar transcripciones de libros físicos a su formato digital (Aradillas, 2015).

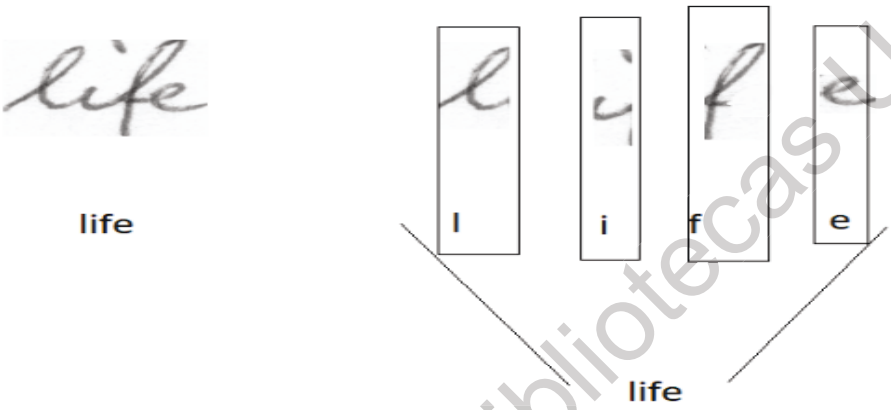



**Figura 12.** Diferencia entre clasificación temporal y clasificación de caracteres individuales. Fuente: Aradillas (2015).

Dichos *frameworks* y librerías trabajan en conjunto con el lenguaje de programación multiparadigma Python, el cual cuenta con múltiple soporte y una comunidad muy amplia. Como se puede mencionar a continuación.

## 8.4 Python

Python fue creado por Guido Van Rossum en los años 90 y se considera como un lenguaje de programación interpretativo, con muchas características, como, por ejemplo, multiplataforma, de alto nivel, tipado dinámico y capaz de soportar multiparadigmas de programación, además de ser muy flexible y portable. Por su comunidad, es un lenguaje el cual tiene mucho soporte en todo el mundo y en múltiples idiomas, por lo cual también se encuentra muy bien documentado y con una amplia variedad de librerías y *frameworks* los cuales cumplen con algunas actividades específicas de desarrollo. Algunas de esas librerías y *frameworks* son *Django* y *OpenCV*, las cuales se utilizarán en el área experimental de este artículo, ya que ambas cumplen con todas las características necesarias para el objetivo de este trabajo (Magaña, 2017).

Con todo esto dicho, para la parte de la experimentación se hará uso, como lenguaje base, a Python en su versión 3.6, *OpenCV* para el manejo y preprocesamiento de las imágenes, así como medio para la implementación de ciertos algoritmos, y se apoyará en la integración de los diferentes *frameworks* para comparar resultados y poder ver de qué forma se pude llegar a lograr los diferentes objetivos establecidos en esta investigación.

## 9. CARACTERÍSTICAS DEL MEDIO O ENTORNO DE EJECUCIÓN

Para la elaboración de la fase de experimentación, se planteó un lugar en el cual haya un cierto flujo de personas y en el cual el fondo o paisaje no tenga cambios tan bruscos y mantenga su integridad lo mejor posible.

Con tales requisitos también se buscó un lugar iluminado por luz natural y que las condiciones del clima fueran las óptimas. Día soleado o poco nublado, sin ráfagas de vientos violentas y sin animales, pájaros, que pudieran estorbar, también sin un gran número de árboles alrededor, ya que el movimiento de las hojas podría entorpecer el funcionamiento principal de los algoritmos, el experimento y en sí de la metodología completa.

Con todos estos parámetros se optó por realizar las pruebas, como primera etapa, con un video de la plataforma de YouTube (Cuevas, 2014), y como segunda etapa, en la facultad de Informática de la Universidad Autónoma de Querétaro.

El espacio requerido a la facultad para la realización del experimento, fue un espacio abierto entre los diferentes edificios de salones, laboratorios y administrativos.

La hora adecuada para la realización del experimento en la facultad fue durante los cambios de materias de los alumnos durante la mañana o a medio día,

así se obtenía un flujo moderado de los alumnos y las condiciones climáticas eran las requeridas.

El dispositivo con el cual se capturó la imagen de video fue una cámara *GoPro Hero 7 Black* de 12MP. Los videos de muestra se grabaron a una buena calidad HD de 720p a 60fps y con una duración menor a un minuto.

Todas estas características de entorno y videos, fueron para la realización de pruebas de laboratorio. Para las pruebas en tiempo real y la implementación final, se requirieron de otras características de dispositivos y calidad de video. El escenario de pruebas se buscó prevaleciera el mismo, sólo se consideró el cambio de ángulo por el tipo de dispositivos empleados y la calidad de video que los dispositivos disponibles ofrecían.

## 10. PROCESO DE APLICACIÓN

### 10.1 Adquisición de las imágenes

De acuerdo a las características y el entorno en el cual se desarrollaron, el trabajo de investigación se puede dividir en tres etapas: 1) pruebas de adaptación y selección de configuraciones, 2) selección y aplicación de algoritmos y la previa definición de la metodología a proponer, 3) ya con los algoritmos configurados con los correspondientes parámetros, se puede implementar con la idea final de este trabajo el cual es aplicarlos en un ambiente aleatorio como lo sería un escenario en tiempo real.

Para cada una de las etapas se requiere de imágenes o fotogramas con los cuales trabajar, a cada uno de los *frames* se les aplican ciertas técnicas para poder trabajar con ellas, las cuales se describen a continuación.

## 10.2 Primera etapa

Para la primera etapa, el fotograma de prueba se recortó del video especificado, el cual, al tener un fondo fijo, como lo es una calle, es posible extraer y eliminar dicho fondo de la función o el propósito requerido y así sólo utilizar los elementos de interés que serían los objetos en movimiento que no corresponden al fondo, en este caso personas que transitan la calle o el lugar designado.

Para la substracción del fondo y poder distinguir a los elementos en movimiento, se optó por utilizar la diferencia entre dos imágenes y así obtener las coordenadas de los objetos no pertenecientes al fondo. Para ello cada *frame* se pasa de una escala RGB a una escala de grises, de esta forma se facilita hacer la diferencia entre ambos fotogramas.

El resultado de esta diferenciación entre un fotograma pasado y uno actual, muestra una imagen en escalas de grises en la cual el fondo, al permanecer estático, queda de color negro, mientras que las siluetas de los diferentes objetos que se encuentran en movimiento quedan en una tonalidad entre grisácea y blanca. A este *frame* resultado se le conoce como "*flow*" y es en el cual se aplican las técnicas de preprocesamiento.

### 10.2.1 Aplicación de filtros

Al tener el *frame* "*flow*", se le aplica un filtro para definir los contornos de los elementos que resaltan, el filtro se denomina como *Canny*, con él los bordes quedan marcados y es sencillo trabajar y distinguirlos del fondo. La función *Canny* pertenece a la librería de *OpenCV*.

*Canny* queda definido por:

$$G = \sqrt{G_x^2 + G_y^2} \tag{17}$$

$$\Theta = atan2(G_y, G_x) \tag{18}$$

Donde $G$ es el valor devuelto de la primera derivada en la dirección horizontal ($G_x$) y la dirección vertical ($G_y$), con ello se determina el gradiente del borde y la dirección. Mientras que el ángulo ($\Theta$) se obtiene usando la función de arco tangente con los dos argumentos de dirección horizontal ($G_x$) y la dirección vertical ($G_y$).

Ahora con el filtro *Canny* aplicado al *frame* "*flow*" se realiza una binarización de la imagen para poder definir de una mejor manera los bordes de las regiones de interés de la imagen, así queda completamente de color negro el fondo y de color blanco los contornos. Los rangos para aplicar el umbral pueden variar, es decir, el pixel que cumpla o esté dentro de dicho rango puede quedar de color blanco o negro, para la metodología el umbral empleado es de 127. La técnica de binarización, igualmente, pertenece a la librería de *OpenCV*.

La binarización en este punto queda definida por:

$$g(x,y) = \begin{cases} 1 \; si \; f(x,t) > 127 \\ 0 \; si \; f(x,t) \leq 127 \end{cases} \tag{19}$$

Como siguiente paso, se aplican las operaciones morfológicas básicas de dilatación y erosión mediante la función de "*closing*" de *OpenCV*, dicho proceso se conoce como transformación morfológica de la imagen, con esta técnica se obtiene una imagen en la cual las zonas de interés quedan más definidas sin atentar con su tamaño o forma, ya que la alteración de dichos aspectos podría entorpecer los resultados esperados.

La función de "*closing*" se define como:

$$\varphi_B(X) = X \bullet B = \varepsilon_B\big(\delta_B(X)\big) \tag{20}$$

Al tener las regiones de interés definidas correctamente, lo siguiente es conocer los contornos, es decir, las posiciones de los mismos en la imagen, este punto es muy importante ya que es parte esencial de la metodología. Las coordenadas se obtienen mediante una función que encuentra los contornos de

las figuras resaltadas del fondo, al ser el fondo de color negro, sólo arrojará las coordenadas de las regiones de interés resaltadas en blanco y se guardarán en un arreglo para su posterior tratamiento.

### 10.2.2 Redibujado en la imagen

Como parte de la primera etapa, es importante evaluar en este punto cuales regiones está marcando como zonas de interés, para ello se hace uso de la función para dibujar rectángulos, por lo cual es necesario usar dos funciones de *OpenCV*. La primera función recibe como parámetro los contornos encontrados en el paso anterior, con ellos se obtienen los máximos para definir los puntos (x, y), así como los valores de anchura (*weight*) y altura (*height*). Al tener estos cuatro valores, pasan ahora sí por la función que dibujará el respectivo rectángulo en la imagen a mostrar. La función recibe como parámetros principalmente la imagen en donde se va a dibujar, en este caso la imagen "*flow*" y las coordenadas dadas por los puntos, (x, y, w, h), dados por la función anterior.

También es importante que la metodología distinga entre regiones muy pequeñas, por lo cual es conveniente aplicar una validación en la que se evalué el tamaño de cada una, esto se realiza mediante una función que toma como parámetros a los contornos dados y dependiendo de la diferencia entre ellos define si la región es muy pequeña para ser considerada. Si cumple con las expectativas, prosigue con el algoritmo, de no ser así se pasa al siguiente conjunto de contornos.

### 10.2.3 Recorte de la imagen

Con los puntos que se utilizan para el dibujado de los rectángulos, se utilizan para recortar las regiones de interés, esto se realiza mediante la selección de los puntos dados y tratando la imagen como una matriz, con esto se obtienen imágenes de cada una de las zonas. El número de nuevas imágenes depende del número de regiones que se detectaron y que cumplen las condiciones propuestas.

Además, las imágenes recortadas se obtienen de la imagen de entrada, es decir, de la primera imagen antes de ser convertida a escala de grises.

### 10.2.4 Aplicación de *Darknet YOLO*

Cada una de las imágenes recortadas, se utilizan como parámetro de entrada para poder ejecutar la herramienta de *Darknet YOLO*.

Para poder utilizar la librería de *Darknet YOLO*, es necesario referenciar una versión compatible que permita el uso de la GPU de la computadora y, como se menciona en el trabajo, que sea compatible con el lenguaje de programación Python.

Con la librería bien referenciada, se utilizan simplemente dos funciones para cumplir con el objetivo. Pero antes de eso, es necesario resaltar ciertos puntos como son los parámetros de configuración inicial. Para ello se requiere descargar el modelo por defecto que ofrece la misma librería de *Darknet YOLO* denominado como "*weights*" y su respectivo archivo de configuración, así como el archivo con los nombres de cada una de las clases que el modelo puede detectar. Estos archivos son utilizados como configuración inicial al momento de referenciar e instanciar la librería, es decir, son sus parámetros de entrada para que pueda funcionar correctamente.

La primera función simplemente se utiliza para convertir la imagen de entrada al formato con la cual puede trabajar la librería de *Darknet YOLO*. La imagen de entrada de dicha función debe de estar en formato RGB, ya que ocupa que la imagen sea de tres canales y no de uno como lo sería una imagen en escala de grises.

Al tener la imagen con las características que necesita *Darknet YOLO*, se hace uso de la segunda función, la cual detectará todos los elementos que contenga ese *frame* recortado en específico. Como entrada se manda la imagen a analizar en el formato correspondiente y como salida se obtiene un arreglo el cual

contiene las coordenadas o puntos, el porcentaje de certeza y la etiqueta para cada uno de los elementos que detecte en el fotograma.

Ahora con los resultados obtenidos es posible ubicar, dentro de la imagen principal de entrada, cada uno de los objetos detectados.

### 10.2.5 Ubicar y etiquetar objetos

Para cada uno de los objetos obtenidos de la detección, se organizan las coordenadas junto con la etiqueta correspondiente y, mediante un ciclo, se van recorriendo cada una de las posiciones mientras que al mismo tiempo se hace uso de la función para dibujar el rectángulo en la imagen. Como complemento se hace uso de otra función para poder poner texto en la imagen y así etiquetar el objeto en específico para una mayor representación. Dicha función para colocar texto en la imagen corresponde a *OpenCV*.

Al tener los resultados de esta etapa, se concluye como buena. Ahora, con los algoritmos ya definidos y con los parámetros configurados para una imagen, se comienza con la segunda etapa.

## 10.3 Segunda etapa

En la segunda etapa, los algoritmos mencionados en la etapa anterior se someten a la prueba con un video, es decir, de entrada, se tiene un video el cual se va a leer dentro de un bucle y para cada uno de sus fotogramas se aplican los pasos anteriores.

El video es una grabación de la facultad, en la cual se ubican unas escaleras y de frente se cuenta con un edificio de tres plantas, en donde la cámara o toma del video se ubica en una ventana de la segunda planta. El flujo de las personas es constante, ya que fue tomado en el cambio de hora, los alumnos transitan en diferentes direcciones y suben y bajan escaleras a un paso tranquilo, lo cual facilita el desarrollo de esta prueba. Otro punto importante es que en la toma del video no hay otros objetos que puedan entorpecer los algoritmos como

serían unos árboles o cambios de escenario bruscos. Las condiciones del clima también son importantes, por lo cual el video fue tomado durante un clima soleado a medio día.

### 10.3.1 Aplicación de filtros

Cada uno de los fotogramas se considera como una imagen de entrada a la cual se le aplican los filtros descritos en la primera etapa. Los *frame* se convierten a escala de grises y se someten a una distinción entre ellos para tener como resultado un *frame* "*flow*" (Figura 13). A dicho fotograma resultante, se le aplica un filtro *Canny* para resaltar los bordes, se binariza la imagen resultante y se aplican los filtros restantes como son dilatación y erosión. Como resultado se tienen las zonas de interés y se obtienen sus contornos.

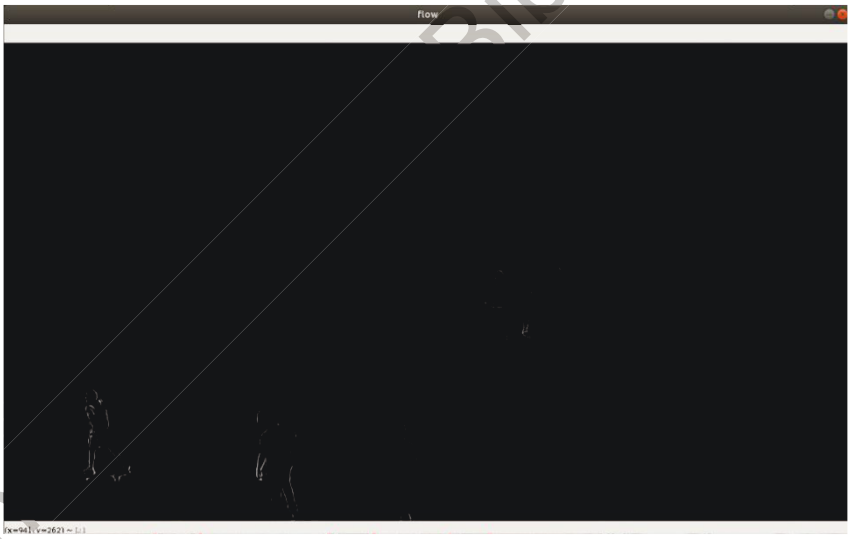



**Figura 13.** Imagen "*flow*" resultante de la diferenciación entre fotogramas. Fuente: Elaboración propia (2019).

### 10.3.2 Delimitación y aplicación de *Darknet YOLO*

Igual que en la etapa anterior, se muestran las regiones de interés como guía mediante el dibujado de los rectángulos delimitando dichas zonas (Figura 14).

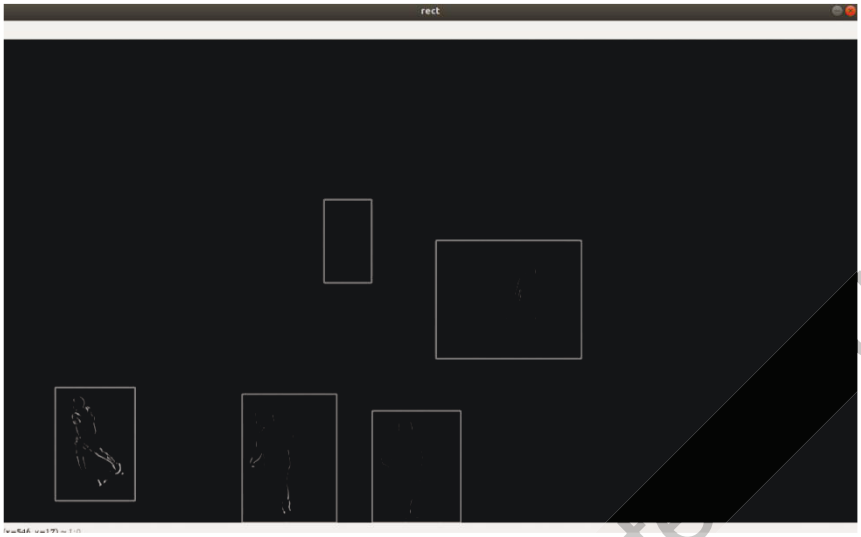



**Figura 14.** Imagen con las regiones de interés marcadas por la función de *OpenCV*. Fuente: Elaboración propia (2019).

Con los contornos también se utilizan para recortar las zonas desde la imagen de entrada y enviarlas a analizar con la ayuda de *Darknet YOLO*. De la misma forma que la prueba pasada, es necesario convertir cada uno de los fotogramas al formato aceptado por la librería, esta función es propia del marco de trabajo. Al tener la imagen con el formato correcto, se manda para detectar a los elementos dentro de dicha parte de la imagen. Se obtienen los resultados y pasa algo interesante.

Al ser una prueba que tiene como muestra un video medianamente controlado y al tener a un cierto número de personas en transición constante y teniendo en cuenta que el modelo utilizado para la detección es un modelo basado en clasificadores, previamente entrenado que cuenta con muchas clases para detectar muchos objetos diferentes y es muy potente, para el objetivo de la prueba, que es detectar y contabilizar a las personas, se aplica una validación para detectar únicamente personas. Esta validación puede ser modificada de acuerdo a lo que se requiera detectar y que esté dentro de las clases del modelo, puede ser una sola clase o un conjunto de ellas. También se aplica una validación de certeza para que el modelo sólo detecte los elementos que esté

completamente seguro o con un índice alto, para la prueba se establece un índice del 98%, el valor de certeza para cada elemento lo define el mismo modelo y *Darknet YOLO*.

### 10.3.3 Etiquetado de objetos

Con los contornos, etiquetas y coordenadas de cada uno de los objetos detectados, se almacenan en un arreglo el cual se recorrer para ubicar y dibujar sus límites dentro de la imagen entrante en el fotograma en cuestión. El recorrido del arreglo con las coordenadas de los objetos se realiza para adquirir la ubicación precisa de cada uno de ellos y así ubicarlos de una forma correcta en la imagen de salida (Figura 15). Se hace uso de las diferentes funciones de *OpenCV* tanto para el dibujado de los rectángulos como de los carteles con la etiqueta del elemento.

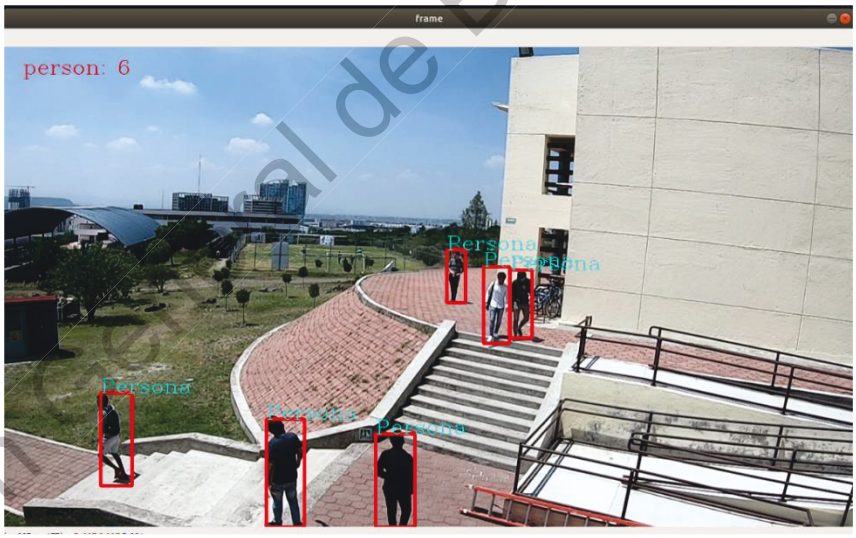



**Figura 15.** Etiquetado de elementos detectados con base a los resultados de Darknet YOLO.
Fuente: Elaboración propia (2019).

Al tener los elementos correctamente etiquetados, se concluye con que la etapa está completa. En sí, dentro de la metodología, sólo se agregó la distinción de cada uno de los objetos detectados y la validación de que correspondan a la clase que se quiere detectar, así como también tengan una certeza correcta de acuerdo a los parámetros esperados

Con la segunda etapa concluida, la metodología tiene una estructura mucho más sólida y lista para entrar a la etapa tres en la cual se realizan pruebas teniendo como muestra un video en tiempo real.

## 10.4 Tercera etapa

En esta etapa, la metodología hasta este punto se prepara para poder ejecutarse en un ambiente variable y un tanto menos controlado, pero con una similitud con la etapa anterior.

Se considera como una etapa con más variables ya que se somete a una transmisión en vivo por lo cual no se tiene un control sobre el flujo de personas que en ese momento transiten por el área o los objetos que podrían acompañarlos, así como las condiciones del clima o luz.

También tiene similitud con la etapa anterior por el hecho de estar en el mismo lugar en el cual se grabó el video de la etapa anterior, por lo que los objetos extra o forma del fondo no tiene cambios bruscos que entorpezcan la aplicación de la metodología y los resultados de la prueba.

### 10.4.1 Aplicación de filtros

De la misma forma que en las etapas anteriores, se hace una diferenciación de los fotogramas para eliminar el fondo y que quede como resultado las regiones de interés, las cuales son recalcadas mediante los diferentes filtros, *Canny*, binarización, erosión y dilatación. Estos filtros son aplicados al *frame* denominado "*flow*" que es el resultado de la diferenciación. También se obtienen los bordes de las regiones y sus coordenadas, con esto se continua con siguiente paso de la metodología.

### 10.4.2 Dibujado en la imagen

Igualmente, como guía, se utilizan las coordenadas obtenidas para dibujar y diferenciar las zonas de interés dentro de la imagen, este paso es posible gracias a las funciones de *OpenCV*.

Con las zonas delimitadas y de buena referencia, se prosigue con el recorte de la imagen y su posterior análisis.

### 10.4.3 Recorte y análisis de la imagen

Las zonas delimitadas indican la parte de la imagen que se recorta para poder ser analizada por *Darknet YOLO*. Igualmente, el proceso de recorta la sección especificada por las coordenadas es tratando como una matriz a la imagen y seleccionando las posiciones que se necesitan para así crear una nueva imagen.

Con la imagen recortada, se manda a analizar con las funciones de *Darknet YOLO*, primero para convertirla al formato que es requerido y después a la detección. Con los resultados se prosigue a el dibujado y marcado en la imagen de entrada. Los objetos detectados están limitados a sólo personas y con cierto grado de certeza como se configuró anteriormente.

### 10.4.4 Dibujado y etiquetado en la imagen

En este paso, como en la etapa anterior, se muestran en la imagen de salida los objetos en movimiento detectados, definidos por un rectángulo y una etiqueta los cuales se colocan haciendo uso de las funciones de *OpenCV*. Las coordenadas y la etiqueta del objeto están definidas por los resultados de la función correspondiente de *Darknet YOLO*.

### 10.4.5 Latencia de respuesta

En este punto se detecta que el análisis que se hace es un tanto tardado y la latencia es un poco alta, en promedio 2 segundos, esto se debe a que cada uno de los fotogramas es analizado y pasa por todos los pasos propuestos, aunque los mismos no tengan grandes cambios significantes. Para poder resolver esta problemática, se establece un filtro más que hace que cada tantos fotogramas, se mande a analizar y pase por cada uno de los pasos anteriores, descartando los demás *frame*. Si el número de fotogramas por segundo es igual a 30, como es el

caso de esta prueba, se considera que una buena opción es someter un *frame* dentro del rango de entre 7 o 15 a los diferentes algoritmos, de esta forma se tiene una prueba más fluida y sin tener que sacrificar muchos detalles o que entorpezca los resultados u objetivos de la metodología.

Para la prueba en tiempo real se fijó que cada 10 fotogramas, el *frame* seleccionado se somete a pasar por cada uno de los pasos de la metodología, dando como resultado una prueba fluida que cumple con las expectativas y objetivo de la metodología, así como resolviendo el problema de la latencia.

### 10.4.6 Conteo de los elementos

Otro punto importante que falta en la metodología es el conteo de los objetos en movimiento y que son parte del caso de estudio. Para poder contar cuántos elementos son, es necesario revisar el tamaño del arreglo en el cual se guardan las coordenadas y etiquetas de cada uno de ellos, es decir, los resultados obtenidos por *Darknet YOLO*. Estos resultados, previamente validados, posibilitan el conteo de los mismos para obtener el número exacto de elementos que se han detectado.

En sí lo que se realiza en este paso es analizar y guardar los diferentes elementos únicos que detecta la función de *Darknet YOLO* dentro de un arreglo, el cual va guardando la etiqueta del elemento y un número que funge como contador, este mismo va aumentando de acuerdo a los elementos que pertenezcan a la misma clase que se detecten en el fotograma analizado. Al realizar el conteo, los resultados son adjuntados y desplegados en la imagen de salida, el dato va cambiando de acuerdo a cada imagen que se analice.

Con todas estas etapas y pasos definidos, se tiene como resultado una metodología la cual está preparada para ser sometida a diferentes pruebas y así comprobar si cumple con los objetivos propuestos. Para las pruebas, se dividen en dos partes, unas con videos previamente grabados con las mismas condiciones en las cuales se grabaron los videos de las etapas pasadas y las demás pruebas

realizando una comprobación en tiempo real, en este punto respetando la ubicación en la cual se ubicó la cámara para la grabación de los videos ocupados en las otras pruebas realizadas, así como tratando de respetar las condiciones que podrían llegar a interferir en el desarrollo de las pruebas.

## 11. PRUEBAS GENERALES

Las pruebas generales se realizan con la ayuda de videos previamente grabados, dichos videos han sido tomados con una cámara *GoPro Hero 7 Black*. La configuración de la cámara es la siguiente:

- Resolución: 720px
- FPS: 60fps
- Estabilización: Auto

Los videos tienen una duración promedio de 17 minutos, de los cuales se eligen las partes en las cuales hay flujo de gente, dando como resultado pequeños cortos de 15 a 20 segundos de duración.

El escenario en el cual se grabaron los videos para las pruebas generales es el mismo con el que se grabaron los videos para las etapas de la metodología. Se buscó un espacio en el cual haya un flujo de gente, la iluminación se la normal sin cambios bruscos, el clima sea óptimo, sin ráfagas de viento, sin paso de objetos que interfieran con el propósito y, sobre todo, sin cambios de fondo, es decir, que el escenario sea lo mismo.

En cuanto al ángulo del cual se colocó la cámara para la grabación, se buscó un lugar en el cual se pudiera tener una buena toma, para ello se hizo uso de la segunda planta del edificio de cubículos de la Facultad de Informática de la Universidad Autónoma de Querétaro. En dicho espacio, desde un cubículo, se posicionó la cámara y se dejó grabando.

Cada uno de los videos fue sometido a la prueba, es decir, para cada cierto número de fotogramas, el *frame* seleccionado pasaba por cada uno de los pasos de la metodología, desde la conversión a escala de grises, pasando por sustraer el fondo haciendo una diferenciación entre el fotograma actual con el fotograma pasado, generando una nueva imagen denominada como "*flow*", la cual se ocupa para los siguientes pasos de la metodología, la aplicación de filtros como *Canny*, la binarización, la respectiva erosión y dilatación para resaltar cada una de las zonas de interés y la obtención de los correspondientes contornos de las áreas mencionadas.

Con los contornos obtenidos, los pasos siguientes son el análisis de cada una de las imágenes que se recortan de la imagen principal. Dichas imágenes recortadas se convierten al tipo de imagen o dato que es compatible con las funciones de *Darknet YOLO*. Así, al tener la imagen en formato compatible, es analizada por la función correspondiente, con los datos que se obtienen, se contabilizan los mismos con base a la clase que pertenecen y se muestran en la imagen resultante. De igual manera, se dibujan los rectángulos de acuerdo a las coordenadas que se detectan del análisis dado por *Darknet YOLO* para así distinguir y delimitar las regiones dentro de la imagen principal. Dentro de cada región, se colocan las etiquetas correspondientes del objeto a contabilizar, en este caso, como objeto o elemento a detectar y contabilizar, son las personas que transitan por la vía en la cual se grabaron las pruebas.

El desarrollo de las diferentes pruebas generales, se desarrollaron de una muy buena forma, dejando como resultado que la metodología propuesta está preparada para ponerse a prueba en un entorno en tiempo real y así cumplir con uno de los objetivos de la misma.

## 12. PRUEBA ESPECÍFICA EN TIEMPO REAL

Para las pruebas en tiempo real, se hizo uso del material y el lugar con el cual se contaba. Para el espacio, se ocupó un cubículo de la facultad, el mismo que tiene una vista desde la segunda planta hacia el mismo lugar en donde se grabaron los videos para el desarrollo de la metodología y donde se realizaron las pruebas generales (Figura 16). El propósito de ocupar el mismo lugar es para evitar que la metodología tuviera cambios que pudieran ocasionar resultados erróneos en algunas situaciones. Por tal motivo, el escenario es el mismo, sin cambios. Las condiciones climáticas, también se buscó que fuera un día soleado para no afectar con cambios de iluminación y en cuestión del horario, fuera en el horario de cambio de materias para que hubiera un flujo considerable de personas y así poner a prueba a la metodología.

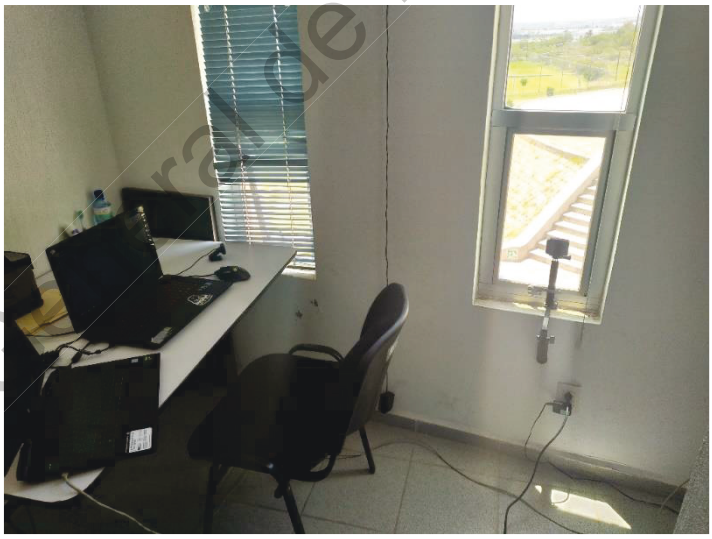


**Figura 16.** Lugar donde se realizaron las pruebas en tiempo real. Fuente: Elaboración propia (2019).

En cuanto al equipo con el cual se realizaron las pruebas en tiempo real, se ocupó una computadora portátil que cuenta con un procesador *Intel Core i7* de cuarta generación a *2.5GHz*, con una memoria *RAM* de *8GB* y una tarjeta de video *Nvidia GTX 860* de *2GB*. Una cámara web *Logitech C170* de *5MP* con una

resolución de video de *480p* y una fluidez de video de *15fps*. También se utilizó una cámara *GoPro* de *12MP*, con una resolución de video de *720p* a *60fps*. Con cada una de las cámaras se realizaron pruebas diferentes.

Con la cámara web, se realizaron pruebas más directas, pero con menor calidad. Para ello se conectó directamente la cámara web a la computadora, se modificó el código de la metodología para indicar que la fuente de entrada era el dispositivo designado y se colocó la cámara sobre una superficie firme y apuntando hacia el punto de interés.

Para llevar a cabo la metodología propuesta, se captura el fotograma desde la fuente de entrada, se convierte a escala de grises, se realiza la diferenciación entre el frame anterior y el actual. A la imagen resultante, imagen "*flow*", se le aplican, en primer lugar, el filtro *Canny* para resaltar los contornos que se detecten, para mejorar estos rasgos, se aplica una binarización para eliminar por completo el fondo y dejar únicamente las regiones de interés de color blanco. Para definir dichas regiones, se aplica una erosión y dilatación, facilitando la adquisición de los contornos, que son el siguiente paso de la metodología.

Se recortan las imágenes a analizar, esto desde la imagen de entrada principal, con los contornos que se obtienen y con las imágenes recortadas se someten al análisis de *Darknet YOLO*.

Cada una de las imágenes recortadas se convierten al tipo de imagen que requiere la función de *Darknet YOLO*, para ello se emplea la otra función del mismo marco de trabajo. Al enviar la imagen ya convertida, la función correspondiente devuelve los resultados, estos resultados son la clase del objeto, la certeza de que el objeto pertenece a la clase indicada y las coordenadas de sus contornos para ubicar dicho objeto en la imagen. Con estos datos, se completa la metodología indicando, con los contornos, la ubicación de los elementos en la imagen principal y con las clases de los objetos, el conteo de los mismos en

movimiento, así mismo, se coloca el dato que se obtiene del conteo en la imagen principal con su respectiva clase para mayor referencia y distinción.

Se realizaron también pruebas conectando la cámara *GoPro*, mediante una red local proporcionada por la misma, a la computadora portátil utilizada en la prueba de la cámara web. La conexión entre ambos dispositivos se logró mediante la implementación de un código de *Python* y el envío de datos mediante protocolos *UDP*. Al ser una conexión por red inalámbrica y no alámbrica como la cámara web, el envío y recepción de los datos tenía cierta latencia que se consideraba en el desarrollo de la prueba, así mismo, la calidad de imagen es mayor por la configuración y calidad de la cámara por lo cual la detección y etiquetado de los elementos en movimiento se ve potenciada.

Al configurar los dispositivos y especificar como fuente de entrada la imagen de la cámara *GoPro*, se realizan los diferentes pasos de la metodología con el fotograma de entrada. Los pasos de la metodología son los mismos ya propuestos y empleados en las diferentes pruebas, con la única diferencia es la calidad con la cual se muestran los resultados.

Con las pruebas específicas de tiempo real finalizadas, tanto con la cámara web y la cámara *GoPro*, se obtienen los resultados los cuales, de acuerdo y limitados por los materiales y lugares proporcionados, se describen a continuación.

## 13. RESULTADOS

Cada una de las pruebas que se realizaron en tiempo real, arrojaron muy buenos resultados. En todas ellas, los resultados se comportaron de manera esperada a la metodología, los fotogramas eran tomados de acuerdo a lo definido en cada método de entrada, se implementaron cada uno de los diferentes filtros para el pre procesamiento de la imagen y así tener una imagen limpia para la detección y etiquetado de los elementos de interés, en este caso personas, con ello poder contabilizar a los mismos y cumplir con los objetivos descritos, también obtener una imagen de salida en la cual se marcan las zonas de interés en donde están cada uno de los elementos detectados con su respectiva clase a la que pertenecen y por último mostrar el total de elementos que pertenecen a esa clase.

La diferencia de resultados entre los dos tipos de pruebas realizados tanto con la cámara web conectada directamente y la cámara *GoPro* conectada mediante una red inalámbrica local, son la latencia que se agrega por el tráfico de datos y la calidad de imagen que se obtiene con ambos dispositivos respectivamente (Tabla 1).

**Tabla 1.** Tabla de comparación de resultados entre los dos tipos de pruebas en tiempo real realizadas.

| | Pruebas con GoPro | Pruebas con cámara web |
|---|---|---|
| **Latencia** | 3 a 4s | 150ms |
| **Calidad de video** | 720p a 60fps | 480p a 15fps |
| **Detección y etiquetado de objetos** | Correcto | Correcto |

En cuanto a la cámara web, la latencia no se ve afectada ya que se establece una conexión directa y alámbrica, por lo que el tráfico de datos es constante y sin interferencias permitiendo que la lectura sea más rápida y sin

afectar a la metodología, la contra parte se encuentra en la calidad de la imagen de entrada, ya que al ser una cámara web con baja resolución, las imágenes tienen una menor calidad por lo que complica un poco la definición de los elementos que se encuentran en ella, también el flujo de fotogramas se ve un poco afectado al tener un ratio de captura de *15fps*. Sin embargo, aún y con los pequeños inconvenientes de calidad, los resultados son los esperados, ya que cada una de las regiones que se detectan con movimiento, las funciones de *Darknet YOLO* las analizan y las detectan sin ningún problema, afectando solamente las regiones que se detectan y que realmente no contienen a ningún elemento en movimiento (Figura 17). Este problema resulta por la variación de la captura de la cámara, por lo cual la diferencia es más brusca. El etiquetado de cada uno de los elementos se dibuja de buena manera sin marcar objetos basura o que no son motivo de estudio, distinguiendo y delimitando únicamente a los que sí pertenecen a la clase y mostrando la cantidad correcta de elementos en movimiento que se detectan en ese momento.

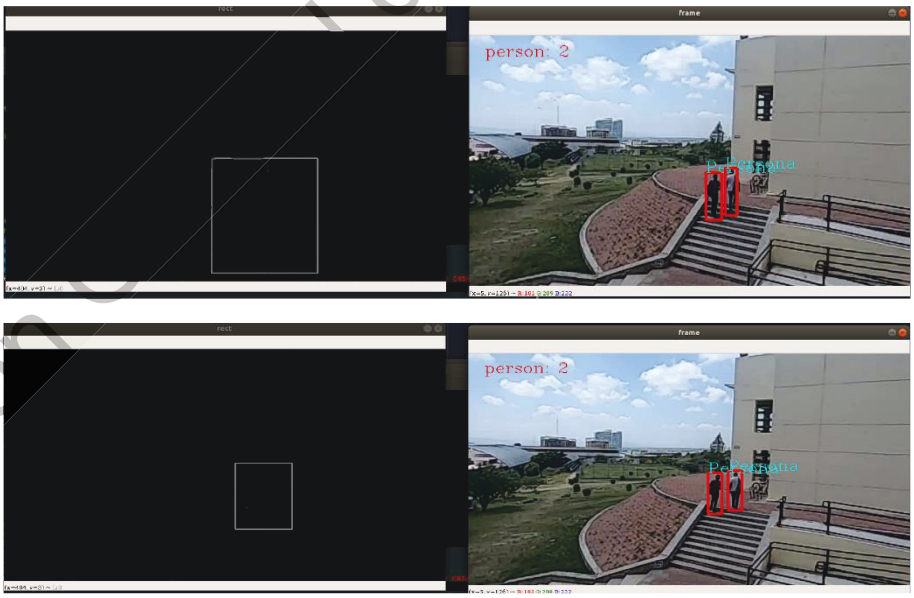



**Figura 17.** Resultados obtenidos con las pruebas en tiempo real realizadas con la cámara web.
Fuente: Elaboración propia (2019).

Por otra parte, la cámara *GoPro* ofrece mejor calidad en la imagen por las configuraciones que ofrece y el hardware que la compone, con inconveniente de un pequeño aumento en la latencia de 3 a 4 segundos, esto por la forma en que se conectan ambos dispositivos, pues la cámara *GoPro* genera una red inalámbrica propia por la cual se transmite la información de imagen que captura por su lente. También es necesario utilizar un código de *Python* el cual configura y habilita la propia transmisión y así poder conectar la cámara correctamente con la computadora portátil. La calidad de las imágenes es mejor y no se ve afectado el flujo o transición de fotograma a fotograma ya que captura video en una resolución superior a la que ofrece la cámara web, sin embargo, la latencia sí se ve afectada pero no pone en riesgo a la metodología propuesta. Aún con eso, los resultados fueron los esperados, de igual manera, cada uno de los fotogramas que se obtienen son pre procesados con los diferentes filtros para limpiar la imagen y recortar cada una de las regiones de interés para convertirlas y analizarlas por medio de las funciones de *Darknet YOLO*, con los resultados se indica en la imagen la ubicación y etiquetado de los elementos, así como el total de los mismos que se detectan.

Los resultados entregados por las diferentes pruebas, confirman que la metodología cumple con los objetivos propuestos y con ello se definen las conclusiones finales del trabajo.

## 14. CONCLUSIONES

Cada una de las diferentes partes que componen la metodología propuesta cumplen por sí solas ciertas tareas, desde la obtención de la imagen hasta el etiquetado de los objetos en la misma.

Para la obtención de la imagen, se considera que *OpenCV* es de gran utilidad para leer archivos de videos guardados en disco hasta archivos en línea enviados por protocolos *UDP*. Además, todas sus demás funciones de filtros y pre procesamiento de la imagen son fáciles de comprender y aplicar en la imagen.

La manipulación del color de la imagen con *OpenCV* es una herramienta muy útil que permite tratar imágenes de tres canales como RGB a manipular y trabajar con imágenes de un sólo canal como lo es la escala de grises, de esta forma se facilitan muchos de los procesos con imágenes, como lo ocupa la metodología al momento de hacer la diferenciación entre el fotograma pasado y el actual, así como la generación de la imagen "*flow*", la cual se necesita para el desarrollo de los siguientes pasos de la metodología.

Los filtros empleados en el pre procesamiento, como el filtro de *Canny* es muy útil al momento de querer seleccionar de la imagen los contornos de las figuras a resaltar, permite, de acuerdo a los parámetros dados, remarcar los contornos sobre el fondo y realizar la tarea.

Al igual, la binarización que ofrece *OpenCV* a través de su función, permite pasar la imagen de una escala de grises a una binaria, en la que el fondo se ignora por completo al ponerlo de color negro y la zona de interés la remarca de color blanco. De la misma forma, otros de los filtros los cuales cumplen una buena tarea, son los filtros de erosión y dilatación.

Erosión y dilatación, permiten que las áreas resaltadas de la imagen conserven su estructura y rellenen sus interiores para obtener como resultado una figura solida la cual resalta del fondo a la región de la figura de interés para su

tratamiento. Con estos filtros, la obtención de los contornos de las figuras se facilita.

La función de *OpenCV* que define los contornos de los objetos en una imagen, permite que de una imagen en la cual se encuentren formas de figuras se obtengan sus contornos, es muy útil para conocer las posiciones de los mismos y con ello realizar ciertas tareas. Para la metodología es necesario, pues con ello se tiene la ubicación de los diferentes objetos y es un proceso relativamente más sencillo porque la imagen de entrada en dicha instancia del proceso se encuentra binarizada.

Otra de las tareas empleadas para el desarrollo de la metodología es el poder recortar las regiones de interés para así reconocer de dichas partes los objetos, en este caso personas, mediante las funciones de *Darknet YOLO*. Las imágenes, al ser matrices, con las metodologías de *Python* es suficiente para obtener un pedazo de la imagen para la detección de objetos.

*Darknet YOLO* es una herramienta muy potente, que como se explicó durante el desarrollo de la metodología, puede detectar diferentes clases de objetos mediante su propio modelo pre entrenado. Es un marco de trabajo muy eficiente que tiene una fácil implementación en el lenguaje de programación de *Python*, además de que puede llegar a complementarse con *OpenCV* y hacer uso de la *GPU* de la computadora para obtener mejores resultados en menor tiempo. La interpretación y detección de la imagen es muy potente y ofrece muchas posibilidades de desarrollo, además que son funciones fáciles de utilizar y que regresan una serie de datos estructurados de una manera fácil para su tratamiento en las diferentes partes de la metodología.

Para una mejor visualización de los resultados, es importante utilizar las funciones que también ofrece *OpenCV*, tanto para dibujar los rectángulos con base a las coordenadas obtenidas, así como colocar el texto del total de objetos detectados que están en movimiento.

Otro tema importante es la latencia y el método con el cual se obtiene la imagen para la implementación de la metodología, hay una gran diferencia entre capturar video con una cámara web de baja calidad, pero conectada directamente a la computadora portátil y capturar el video mediante una cámara más potente, como lo es una cámara *GoPro*, pero con mayor latencia por la limitante de la conexión directa entre ambos dispositivos.

Aún y con estos inconvenientes, la metodología se comporta dentro de lo esperado, pero obteniendo mejores resultados, de acuerdo a los objetivos, utilizando la cámara web, ya que la latencia al conectar un dispositivo como la cámara *GoPro* incrementa los tiempos en el transporte de la información y después de un momento los resultados, es decir la imagen de salida, sufre un retraso de 3 segundos y ya no se considera como un resultado en tiempo real.

La calidad de imagen que ofrece una cámara *GoPro*, facilita la lectura y el tratamiento de la imagen, sin embargo, la metodología se comporta de muy buena forma con imágenes de baja calidad, como las que ofrece la cámara web, esto se debe a que únicamente se mandan las partes de la imagen de interés a detectar y el marco de trabajo de *Darknet YOLO* se ocupa de la detección independientemente de la baja resolución de las imágenes.

En este aspecto, *Darknet YOLO* se concluye que es una herramienta muy potente para la detección de ciertos elementos, de acuerdo al modelo que use como base y también ofrece un abanico amplio de posibilidades de acuerdo a el o los objetos en movimiento que se quieran detectar.

Para concluir, la metodología propuesta cumple con los objetivos propuestos en este trabajo de investigación, cada una de las partes de la misma se encarga de aportar a lo que se busca resolver, tanto el pre procesamiento y la preparación de la imagen, hasta la detección y el etiquetado de los elementos de interés. Por último, un factor muy importante es la potencia y la calidad de los dispositivos que se emplean en la implementación de la metodología, ya que como

se vio durante el desarrollo de las pruebas, es necesario utilizar una cámara que ofrezca una buena resolución de imagen sin sacrificar conexión entre dispositivos y una computadora la cual pueda emplearse su *GPU* para ayudar en la detección de objetos y ofrecer resultados en tiempo real.

## REFERENCIAS


Antonio, M., Pérez, A., Javier, J., y Rojas, B. (n.d.). Espacios de Color RGB, HSI y sus Generalizaciones a n-Dimensiones. Recuperado de: https://inaoe.repositorioinstitucional.mx/jspui/bitstream/1009/362/1/AlonsoPeMA.pdf

Aradillas J. (2015). Reconocimiento de Texto Manuscrito con Deep Learning de ETSIUS Recuperado de: http://bibing.us.es/proyectos/abreproy/70918/fichero/TFM_JC_Aradillas_Jaramillo_v3_0.pdf

Bhanu, B., y Peng, J. (2000). Adaptive integrated image segmentation and object recognition. IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, 30(4), 427–441. Recuperado de: http://doi.org/10.1109/5326.897070

Bits, I. M. D. E., y Vectorial, I. (n.d.). Conceptos básicos de imagen digital 1-.

Blas, M. R., Agrawal, M., Sundaresan, A., y Konolige, K. (2008). Fast color/texture segmentation for outdoor robots. 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 1(d), 4078–4085. Recuperado de: http://doi.org/10.1109/IROS.2008.4651086

Bouwmans, T., El Baf, F., y Vachon, B. (2008). Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey. Recent Patents on Computer Science, 1(3), 219–237. Recuperado de: http://doi.org/10.2174/2213275910801030219

Chen, L., Guo, B., y SUN, W. (2010). Obstacle Detection System for Visually Impaired People Based on Stereo Vision. 2010 Fourth International Conference on Genetic and Evolutionary Computing, 723–726. Recuperado de: http://doi.org/10.1109/ICGEC.2010.183

Chowdhury, A., Cho, S. J., y Chong, U. P. (2011). A background subtraction method using color information in the frame averaging process. Proceedings of the 6th International Forum on Strategic Technology, IFOST 2011, 2, 1275–1279. Recuperado de: http://doi.org/10.1109/IFOST.2011.6021252

Ciric, I., Cojbasic, Z., Nikolic, V., y Antic, D. (2013). Computationally intelligent system for thermal vision people detection and tracking in robotic applications. 2013 11th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services, TELSIKS 2013, 2, 587–590. Recuperado de: http://doi.org/10.1109/TELSKS.2013.6704447

Cuevas, A. [Angel Cuevas]. (2014, marzo 4). Gente paseando por la calle concepción 1 [Archivo de Video]. Recuperado de: https://www.youtube.com/watch?v=8wVKDD8hjys&t=412s

Danciu, G. (2017). Method proposal for blob separation in segmented images, 1108–1113.

Darknet. (2018). Redmon, Joseph and Farhadi, Ali. Recuperado de: https://pjreddie.com/darknet/

De Detección Implementación Y Seguimiento De Objetos En Redes. (n.d.), 24–28.

Delakis, M., y Garcia, C. (2003). Training convolutional filters for robust face detection. Neural Networks for Signal Processing - Proceedings of the IEEE Workshop, 2003–January, 739–748. Recuperado de: https://doi.org/10.1109/NNSP.2003.1318073

Ecnica, O. (2010). A Partir De Inteligencia Artificial.

Gupta, S., Girshick, R., Arbeláez, P., y Malik, J. (2014). Learning rich features from RGB-D images for object detection and segmentation. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence

and Lecture Notes in Bioinformatics), 8695 LNCS (PART 7), 345–360. Recuperado de: http://doi.org/10.1007/978-3-319-10584-0_23

He, H., y Upcroft, B. (2014). Automatic object segmentation of unstructured scenes using colour and depth maps. IET Computer Vision, 8(1), 45–53. Recuperado de: http://doi.org/10.1049/iet-cvi.2013.0018

Hermans, A., Floros, G., y Leibe, B. (2014). Dense 3D semantic mapping of indoor scenes from RGB-D images. Proceedings - IEEE International Conference on Robotics and Automation, 2631–2638. Recuperado de: http://doi.org/10.1109/ICRA.2014.6907236

Husain, F., Schulz, H., Dellen, B., Torras, C., y Behnke, S. (2016). Combining Semantic and Geometric Features for Object Class Segmentation of Indoor Scenes. IEEE Robotics and Automation Letters, 2(1), 49–55. Recuperado de: http://doi.org/10.1109/LRA.2016.2532927

Kaehler, A. y Bradski, G. (n.d.). Learning OpenCV: Computer Vision in C++ with the OpenCV Library.

Kim, D.-N. K. D.-N., Trinh, H.-H. T. H.-H., y Jo, K.-H. J. K.-H. (2007). Regions segmentation using multiple cues for robot navigation on outdoor environment. 2007 International Conference on Control, Automation and Systems, 1768–1772. Recuperado de: http://doi.org/10.1109/ICCAS.2007.4406631

Kong, H., Akakin, H. C., y Sarma, S. E. (2013). A generalized laplacian of gaussian filter for blob detection and its applications. IEEE Transactions on Cybernetics, 43(6), 1719–1733. Recuperado de: https://doi.org/10.1109/TSMCB.2012.2228639

Liao, P. S., Chen, T. S., y Chung, P. C. (2001). A fast algorithm for multilevel thresholding. Journal of Information Science and Engineering, 17(5), 713–727.

Liu, W. C., Lin, S. Z., Yang, M. H., y Huang, C. R. (2013). Real-time binary descriptor based background modeling. Proceedings - 2nd IAPR Asian Conference on Pattern Recognition, ACPR 2013, 722–726. Recuperado de: http://doi.org/10.1109/ACPR.2013.125

Magaña, J. B. (2017). DETECCIÓN DE OBJETOS EN IMÁGENES UTILIZANDO OPENCV.

Nicolás, C. (2013). Método para detección y seguimiento de objetos con aplicaciones en Realidad Aumentada.

Otsu, N. (1979). A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man, and Cybernetics, 9(1), 62–66. Recuperado de: http://doi.org/10.1109/TSMC.1979.4310076

Rahaman, A., Hasan, M. M., Ahmed, R., Maswood, M. M. S., y Rahman, M. M. (2014). A new background updating model for motion detection considering future frame. 16th Int'l Conf. Computer and Information Technology, ICCIT 2013, (March), 378–382. Recuperado de: http://doi.org/10.1109/ICCITechn.2014.6997296

Raju, P. D. R., y Neelima, G. (2012). Image Segmentation by using Histogram Thresholding. Ijcset, 2(1), 776–779.

Redmon, J., y Farhadi, A. (2017). YOLO9000: Better, faster, stronger. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, *2017–January*(April), 6517–6525. Recuperado de: https://doi.org/10.1109/CVPR.2017.690

Rocha, Anderson, y Goldenstein, S. (2009). Classifiers and machine learning techniques for Image Processing and Computer Vision.

Rossius, S., y Bosch Roig, I. (n.d.). Proyecto final de carrera Reconocimiento de objetos mediante WebCam en tiempo real.

Singla, N. (2014). Motion Detection Based on Frame Difference Method. International Journal of Information & Computation Technology, 4(15), 1559–1565.

Siricharoen, P., Aramvith, S., Chalidabhongse, T. H., y Siddhichai, S. (2010). Robust outdoor human segmentation based on color-based statistical approach and edge combination. 1st International Conference on Green Circuits and Systems, ICGCS 2010, 463–468. Recuperado de: http://doi.org/10.1109/ICGCS.2010.5543017

Subudhi, B. N., Ghosh, S., y Ghosh, A. (2013). Moving object detection using Gaussian background model and Wronskian framework. Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2013, 1775–1780. Recuperado de: http://doi.org/10.1109/ICACCI.2013.6637450

Sucar, L. (n.d.). Visión de Alto Nivel. Recuperado de: https://ccc.inaoep.mx/~esucar/Clases-van/van2-bajo-nivel.pdf

TensorFlow™. (2014). Información sobre TensorFlow. de TensorFlow™. Recuperado de: https://www.tensorflow.org/?hl=es

Wan, Z., Sun, K., y Model, A. T. C. V. (2015). Hybrid Active Contour Method Combining Local and Differential Image Information for Image Segmentation, 1(2). Recuperado de: https://doi.org/10.1109/ICNISC.2015.41

Wang, J., Bo, Z., y Wang, S. (2010). Automatic floor segmentation for indoor robot navigation. ICSPS 2010 - Proceedings of the 2010 2nd International Conference on Signal Processing Systems, 1, 684–689. Recuperado de: http://doi.org/10.1109/ICSPS.2010.5555399

Weis, M., Rumpf, T., Gerhards, R., y Plümer, L. (2009). Comparison of different classification algorithms for weed detection from images based on shape parameters. Image Analysis for Agricultural Products and Processes, 69(2005), 53–64.

Xin, B., Tian, Y., Wang, Y., y Gao, W. (2015). Background Subtraction via generalized fused lasso foreground modeling. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June, 4676–4684. Recuperado de: http://doi.org/10.1109/CVPR.2015.7299099

Yingying Chen, J. and H. L., y National. (2015). Learning sharable models for robust background subtraction. 2015 IEEE International Conference on Multimedia and Expo (ICME), 1 – 6. Recuperado de: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7177419