UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

Aplicación de video vigilancia inteligente para el control de aforos permitidos

Adrián Humberto Tello Cedeño

Ingeniería en Ciencias de la Computación

Trabajo de fin de carrera presentado como requisito
para la obtención del título de
Ingeniero en Ciencias de la Computación

Quito, 1 de octubre de 2021

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

Aplicación de video vigilancia inteligente para el control de aforos permitidos

Adrián Humberto Tello Cedeño

Nombre del Profesor, Título académico: Ricardo Flores Moyano, Ph.D

Quito, 1 de octubre de 2021

© DERECHOS DE AUTOR

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en la Ley Orgánica de Educación Superior del Ecuador.

Nombres y apellidos: Adrián Humberto Tello Cedeño

Código: 00124046

Cédula de identidad: 1717839367

Lugar y fecha: Quito, 1 de octubre de 2021

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

RESUMEN

El control ineficiente de aforos es una de las causas principales de la propagación acelerada de la pandemia COVID-19 en los años 2020-2021. Sin embargo, se trata de un problema con una importancia mucho más trascendental. Pues, en años anteriores esta misma problemática ha sido la causante de que se pierdan numerosas vidas humanas en distintas situaciones. Es común que en eventos de caracter deportivo, artístico o social se llegue a superar el aforo máximo o que a su vez no se lleve un control acertado del mismo. De igual manera áreas como el transporte público o zonas de comercio concurrido rara vez respetan el límite de aforo permitido. Esta violación hacia el aforo de un espacio puede desencadenar situaciones lamentables al momento de existir una emergencia que requiera de una evacuación, pues esta se ve interrumpida pudiendo causar muerte o daños irreversibles en las vidas de numerosas personas.

El presente proyecto de titulación propone el uso de análisis de imagen junto con la implementación de inteligencia artificial para el desarrollo de un sistema de video vigilancia. Dicho sistema permite el monitoreo y conteo de personas en un espacio a través de las cámaras de seguridad instaladas en el mismo. La imagen recibida pasa por un proceso de análisis que permite identificar siluetas humanas mediante redes neuronales. Una vez reconocida la silueta se realiza el seguimiento de la misma para aislar la imagen de cada persona y su respectivo movimiento. El sistema actualiza la cantidad de personas encontradas y guarda un registro de este conteo. Cuando la cantidad supera el aforo máximo permitido el sistema envía una alerta de sobrepoblación en el lugar.

En este trabajo se explica con detalle la arquitectura propuesta y la implementación de bajo costo correspondiente que se ejectua en un computador personal con la ayuda de dos cámaras web. De igual manera se presentan distintas pruebas de la ejecución del prototipo y

conclusiones finales del uso de la arquitectura propuesta como posible solución al manejo y conteo automatizado de aforos.

Palabras clave: video vigilancia, inteligencia artificial, red neuronal, aforo, prototipo, OpenCV, MobileNetSSD.

ABSTRACT

Poor control of shared spaces is one of the main causes of the accelerated spread of the COVID-19 in 2020-2021. However, it is a problem of far more fundamental importance. Well, in previous years this same problem has been the cause of the loss of numerous human lives in different situations. It is common that in sporting, artistic or social events the maximum capacity is exceeded or that in turn it is not properly controlled. In the same way, areas such as public transport or busy commercial areas rarely respect the allowed capacity limit. This violation of the capacity of a space can trigger unfortunate situations at the time of an emergency that requires an evacuation, since this is interrupted and can cause death or irreversible damage to the lives of many people.

This degree project proposes the use of image analysis together with the implementation of artificial intelligence for the development of a video surveillance system. This system allows the monitoring and counting of people in a space through the security cameras installed in it. The image received goes through an analysis process that allows identifying human silhouettes through neural networks. Once the silhouette is recognized, it is tracked to isolate the image of each person and their respective movement. The system updates the number of people found and keeps a record of this count. When the amount exceeds the maximum capacity allowed, the system sends an alert in the place.

This paper explains in detail the proposed architecture and the low-cost implementation that runs on a personal computer with the help of two webcams. Also, different tests performed by the prototype are presented. Final conclusions of the use of the proposed architecture as a possible solution to the management and automated counting are shown.

Key words: video surveillance, artificial intelligence, neural network, capacity, prototype, OpenCV, MoibleNetSSD

TABLA DE CONTENIDO

Introducción **10**

Propuesta de arquitectura para conteo de aforos permitidos **12**
1.1 Recepción de imagen: **14**
1.2 Optimización de Imagen: **14**
1.3 Suavizar la imagen: **15**
1.4 Análisis de Superposición **17**
1.5 Segmentación Binaria **18**
1.6 Análisis y categorización de siluetas **19**
1.6.1 Análisis de siluetas. 19
1.6.2 Categorización de siluetas: 20
1.7 Conteo y rastreo de figuras humanas **21**
1.8 Alerta de incumplimiento de aforo **22**
1.8.1 Rango de entrada/salida: 22
1.8.2 Dirección de traslado: 23

Desarrollo del prototipo **24**
2.1 Características técnicas del prototipo: **24**
2.2 Aplicación de la arquitectura propuesta: **24**
2.2.1 Recepción de imagen y detección del fondo 25
2.2.2 Optimización a escala de grises 26
2.2.3 Suavizar la imagen: 27
2.2.4 Análisis de superposición de objetos: 28
2.2.5 Representación de una segmentación binaria: 29
2.2.6 Análisis de siluetas: 29
2.3 Desarrollo para el rastreo de objetivos: **33**
2.3.1 Médodo Update de CentroidTracker: 33
2.3.2 Método Nuevo de CentroidTracker: 34
2.3.3 Método Eliminar de CentroidTracker: 34
2.4 Conteo de objetivos: **34**

Validación del prototipo y resultados **36**
3.1 Versión 1.0: **36**
3.2 Versión 2.0: **37**
3.3 Versión 3.0 ..... **38**

Conclusiones y trabajos futuros **42**

Referencias bibliográficas **44**

ÍNDICE DE FIGURAS

Figura 1: Arquitectura Propuesta 13
Figura 2: Funcionamiento de un filtro de tipo dominio del espacio 16
Figura 3: Observando las imágenes de izquierda a derecha y de arriba hacia abajo se presenta la aplicación de un filtro Gaussiano para valores σ=1 σ=2 σ=3 y σ=4 respectivamente. . 17
Figura 4: Funcionamiento de un análisis de superposición 18
Figura 5: Aplicación de Segmentación Binaria 19
Figura 6: Aplicación de un filtro de dilatación 20
Figura 7: Funcionamiento de Centroid Tracking Algorithm 22
Figura 8: Escenario de pruebas vacío. 25
Figura 9: Frame a analizar 26
Figura 10: Cambio de imagen a escala de grises mediante OpenCV 27
Figura 11: Aplicación de un filtro Gaussiano 28
Figura 12: Análisis de objetos superpuestos 28
Figura 13: Aplicación de una segmentación binaria no inversa 29
Figura 14: Detección persona de frente 31
Figura 15: Detección persona de perfil 31
Figura 16: Detección persona de espaldas 32
Figura 17: Detección busto de persona 32
Figura 18: Persona 1 en cámara 1 39
Figura 19: Persona 1 y 2 en cámara 1 39
Figura 20: Persona 1 en cámara 1 y persona 2 saliendo del cuadro 39
Figura 21: Persona 1 en cámara 1, persona 2 en cámara 2 40
Figura 22: Aforo superado 40
Figura 23: Aforo correcto 40
Figura 24: Aforo vuelve a cero 41

INTRODUCCIÓN

A medida que la pandemia de COVID-19 progresó aceleradamente en el mundo varios problemas ya existentes en la sociedad provocaron que la prevención y manejo de esta se dificulte innecesariamente. Uno de los principales problemas a franquear ha sido el manejo de aforos y control de grandes aglomeraciones, pues ningún país ha logrado dominar un correcto flujo de sus habitantes (Bernal, 2021). Los puntos de encuentro de grandes cantidades de personas como supermercados, medios de transporte y centros de salud son lugares clave para la propagación de virus respiratorios (Bernal, 2021).

En Ecuador, la pandemia solo ha resaltado el problema de aforos; pues incluso antes de que esta ocurra no se contaba con un correcto manejo del flujo de personas en lugares como: bares, discotecas, restaurantes, salones de eventos, espacios deportivos, medios de transporte, entre otros. Esto obliga a los gobiernos zonales a disponer de recursos humanos y económicos para el control y el manejo de aforos manualmente (El Universo, 2021). Estos controles se pueden automatizar con sistemas de video vigilancia inteligente más precisos, ágiles y económicos. De esta manera, se podría redirigir los recursos mencionados hacia otras áreas más importantes como el abastecimiento de personal e insumos médicos.

Cabe mencionar que el manejo correcto de aforos y flujos tiene utilidad más allá de la situación sanitaria mundial y de la realidad actual del país, pues, permite que cualquier espacio esté mejor preparado para todo tipo de evacuación como las provocadas por: incendios, sismos, emergencias médicas, políticas y/o de seguridad nacional.

Dicho lo anterior, el presente proyecto consiste en proponer una arquitectura y un prototipo de video vigilancia inteligente para ubicar y contar la cantidad de personas en un espacio determinado. La arquitectura se basa en herramientas de análisis de imagen, inteligencia artificial, detección, rastreo y conteo de objetos. El prototipo es lo suficientemente

robusto como para resolver el problema de salida de cuadro sin realizar un doble conteo de una persona. El problema de salida de cuadro se refiere a dos problemáticas principales: la primera se produce cuando una persona sale del espacio determinado de alcance de una cámara para ingresar a otro espacio captado por otra cámara. En este caso el sistema reconoce que solamente existe un desplazamiento de una misma persona de un espacio a otro y por tanto no hace un conteo repetido de la persona. La segunda problemática de salida de cuadro está ligada a cuando una persona sale del espacio captado por una cámara y poco tiempo después vuelve a ingresar. En ese caso el sistema debe reconocer que la persona estuvo en algún punto ciego o espacio límite del alcance de cámara y que por lo tanto se trata de un reingreso a la escena, de esta manera tampoco se hace un doble conteo. El sistema tiene previamente definido el aforo máximo para el espacio que está analizando. De esta manera, es capaz de detectar cuando el espacio ha superado su aforo máximo y envía una alerta inmediata.

PROPUESTA DE ARQUITECTURA PARA CONTEO DE AFOROS PERMITIDOS

El manejo de aforos es un problema global que empeoró la situación de pandemia de COVID-19 pues aún con medidas de control de flujo extremas ningún país pudo dar una correcta respuesta al control de grandes aglomeraciones en lugares como supermercados, centros financieros, centros comerciales, medios de transporte o centros de salud (BBC, 2020). Esto desembocó en una pandemia activa con más de un año de duración pues es complicado detener los contagios mientras existan lugares con gran aglomeración de personas que sobrepasan el aforo máximo permitido.

El presente proyecto propone el uso de video vigilancia inteligente como solución a esta problemática. De igual manera se presenta un prototipo simple a través de un computador, dos cámaras web, y un programa desarrollado en Python que permite evaluar una habitación o espacio cerrado para encontrar formas humanas, verificarlas y cuantificarlas. De esta manera el sistema envía una alarma inmediata en el caso que el aforo máximo del espacio analizado haya sido superado. En la Fig. 1 se presenta y explica mediante un diagrama de bloques la división de los distintos módulos que cumplen con la arquitectura propuesta para el proyecto. En las secciones siguientes se detalla el funcionamiento de cada uno de los módulos presentados en la Fig.1

Figura 1: Arquitectura Propuesta

1.1 Recepción de imagen:

Para la arquitectura propuesta para el proyecto se parte desde el principio de funcionamiento de un video. Se entiende por video como el conjunto de imágenes o fotogramas recurrentes uno después del otro que genera la ilusión de movimiento (Pérez y Gardey, 2009) Es necesario partir de esta definición para entonces comprender el análisis a proponer.

De ahora en adelante al referirse a la recepción de la imagen significa que se toma cada imagen o frame como individual. Es decir, a partir de este punto se analizan frames individuales que serán comparados con otros para un fin específico. Dentro de este módulo es importante entender que el frame más importante es el del lugar a analizar pero sin personas. Como se mostró brevemente en el diagrama de bloques de la Fig. 1 se realizará un análisis de siluetas, sin embargo, para la arquitectura propuesta una silueta será identificada como un objeto superpuesto a un fondo ya conocido.

Es por esto que es primordial primero reconocer cuál es ese fondo. Una vez que tengamos identificado este primer frame el resto de imágenes serán comparadas con este para poder reconocer que ha cambiado y por lo tanto empezar a detectar qué objetos nuevos han aparecido y deben ser analizados.

1.2 Optimización de Imagen:

Una imagen digital es un conjunto de pixeles. A su vez, un pixel es visto como una variable matricial que guarda argumentos de posición y color (Pérez y Merino, 2009). Es este segundo argumento el de importancia en este módulo. Al hablar de imágenes digitales a color se establece que este segundo argumento esté compuesto por tres o más valores dependiendo de con qué método sea representado: RGB, HSV, HSL, etc. Por ejemplo, el modelo RGB parte de que cualquier color puede ser formado a partir de una combinación de los colores rojo, verde y azul, por lo tanto el argumento que guarda el color de cada pixel está formado por una matriz

de tres valores que contiene la intensidad de rojo, verde y azul necesarias para que combinados generen el color resultante. Algo parecido ocurre con los métodos HSV y HSL. Sin embargo, el método de representación del pixel no es lo verdaderamente relevante a comprender, por el momento bastará con entender que un pixel es una variable matricial compleja.

Aún cuando los ordenadores de última generación pueden manejar correctamente valores de este tipo si se propone trabajar con análisis de imágenes esto es algo que no se puede tomar a la ligera, pues representará un porcentaje considerable de los recursos del computador. Es por esto que en este módulo se busca la optimización de la imagen a través de la transformación de la misma a una escala de grises. De esta manera el segundo argumento del pixel ya no será una matriz de tamaño tres sino que será reducida a un único valor entre 0 y 255 que simplemente especifica el nivel de intensidad luminosa. Al reducir la complejidad de cada pixel se termina con una imagen mucho mas liviana que precisará de menos recursos para el resto del proceso.

1.3 Suavizar la imagen:

La técnica para suavizar una imagen corresponde a una de las muchas herramientas que existen para el procesamiento de imágenes digitales. Este módulo se realiza con el objetivo de mejorar y facilitar la extracción de datos para el trabajo que realizarán los módulos posteriores. En términos prácticos consiste en la aplicación de un filtro para obtener un frame distinto pero más práctico a la hora de procesar. Lo que se busca es mejorar las características de interés del frame y eliminar las de poco uso o necesidad.

En este caso lo que se busca a futuro es determinar siluetas, para esto lo que nos interesa es corregir ruido, marcar bordes, y eliminar detalles finos. De esta manera el resultado será un frame de menor resolución pero que maximiza los bordes y siluetas de los objetos encontrados.

Para esta arquitectura se propone utilizar un filtro del tipo dominio del espacio, más específicamente un filtro Gaussiano. En la Fig.2 se hace una referencia al funcionamiento de un filtro de tipo dominio del espacio. Estos filtros se caracterizan por cambiar netamente el valor de cada pixel, tomando a este como núcleo y a un grupo de los que le rodea como argumento que determinar cuál debería ser el cambio.

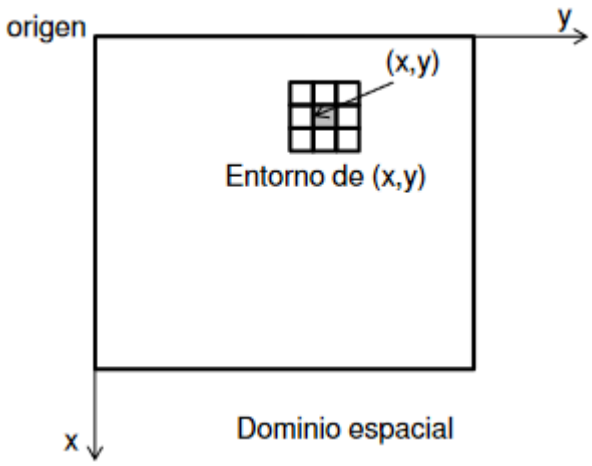

Figura 2: Funcionamiento de un filtro de tipo dominio del espacio

De la Fig.2 se comprende que se aplicará una transformación del pixel (x,y) además se comprende que el entorno de (x,y) influye en esta transformación. Sin embargo no se conoce qué función es la que guía esta transformación. Es aquí donde se introduce al filtro Gaussiano, Este filtro lleva dicho nombre dado que transforma al pixel (x,y) a través de la función gaussiana presentada en la ecuación 1.

$$G\,(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Ecuación 1: Transformación Gaussiana para el par de coordenadas (x,y)

Se propone este filtro por sobre otros filtros de media sobretodo porque produce un suavizado mucho más uniforme dado que al tratar la matriz no hace una convolución

bidimensional sino que realiza dos convoluciones unidimensionales (Olveres y Escalante, 2011). En la Fig.3 se presenta un clásico ejemplo de la aplicación de un filtro Gaussiano para distintos valores de σ.

Figura 3: Observando las imágenes de izquierda a derecha y de arriba hacia abajo se presenta la aplicación de un filtro Gaussiano para valores σ=1 σ=2 σ=3 y σ=4 respectivamente.

Como se puede observar en al Fig.3 se sacrifica nitidez de la imagen a cambio de resaltar los bordes para una posterior facilidad de reconocer situeltas y objetos.

1.4 Análisis de Superposición

En este módulo es donde se procede a realizar una comparación de dos imágenes que ya han recibido el proceso previo. La primera imagen como se mencionó en la sección 1.1, corresponde al fondo del lugar a analizar y siempre se mantendrá igual. La segunda imagen

será la que realmente cambiará siguiendo la secuencia de cada frame que obtenga el video de vigilancia. Lo que propone esta arquitectura es hacer un análisis pixel a pixel de ambas imágenes. De esta manera si ambos pixeles son iguales se deduce que esa sección del fondo siguen exactamente igual, es decir está vacío. Cuando la comparación de píxeles proporciona una desigualdad se deduce que esa sección del fondo ha cambiado y que por tanto un objeto está por encima del fondo (Authentise, 2016). El objetivo de realizar esta comparación es generar una tercera imagen donde solo se guardan los pixeles desiguales, de esta manera se obtiene una representación de los objetos que han invadido el fondo estudiado. En la Fig.4 se puede observar el proceso descrito, donde el círculo azul representa al fondo que ha sido invadido por dos nuevos objetos que se superponen a dicho fondo.

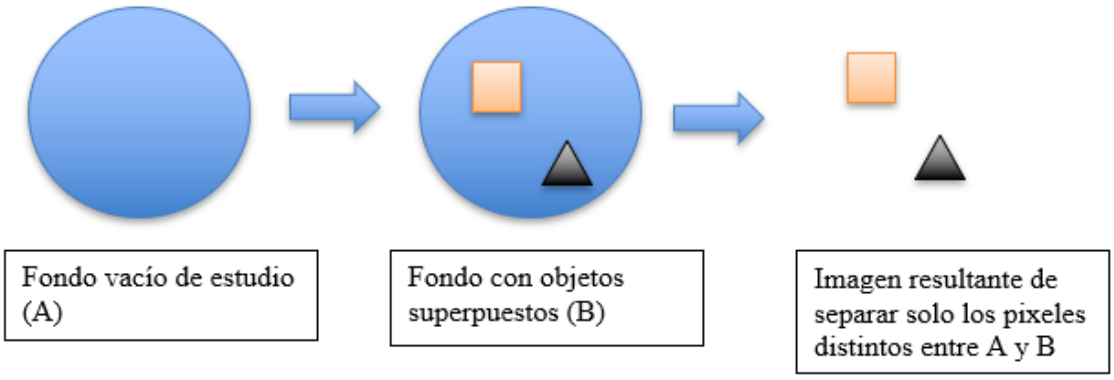

Figura 4: Funcionamiento de un análisis de superposición

1.5 Segmentación Binaria

En este módulo se trabaja con la imagen resultante de los pasos previos. Dado que para el problema de estudio lo realmente importante es poder reconocer una silueta, en este módulo se busca segmentar o separar claramente los objetos del fondo. Es por ello que se propone utilizar una segmentación binaria ya que al ser la más extrema de las técnicas de segmentación permite amplificar al máximo la separación requerida (Zhang et al. 2018).

Como su nombre lo indica, lo que se quiere realizar es un barrido pixel por pixel y agrupar a cada uno dentro de dos posibles segmentos. Es decir, si el pixel tiene una intensidad menor a un número dado, su intensidad será automáticamente editada a uno de los posibles segmentos. Y si por el contrario la intensidad del pixel es mayor a este número dado, su intensidad será editada al otro segmento posible. En la Fig.5 se muestra la aplicación de una segmentación binaria inversa. Se trata de una segmentación binaria ya que como se puede observar transforma a la imagen a solo dos posbiles estados: negro y blanco. Sin embargo, es inversa ya que los pixeles más oscuros son transformados a blanco mientras que los pixeles claros son transformados a negro, esto obedece a una situación de convención ya que las redes neuronales de búsqueda de objetos normalmente están diseñadas para buscar a un objeto conformado por los pixeles pertenecientes a las regiones blancas.

Figura 5: Aplicación de Segmentación Binaria

1.6 Análisis y categorización de siluetas

Debido a las tareas que se proponen para la arquitectura, es necesario revisar por separado el análisis de la categorización.

1.6.1 Análisis de siluetas.

Para este punto se entiende que ya se trabaja sobre imágenes de intensidad binaria con objetos bien definidos. Sin embargo, es necesario todavía realizar algunos procesos de

preparación antes de categorizar el tipo de silueta. Por ejemplo, es útil que cada objeto pueda estar lo más aislado posible. De esta manera se evita que posteriormente se confunda dos objetos muy cercanos como si fueran uno solo. Para este problema se propone utilizar operaciones morfológicas, más específicamente una operación de dilatación. Este proceso es propio de imágenes binarias con formas. Consiste en agrandar el espacio dentro de los objetos y agrandar los espacios entre objetos. Esto permite tener un mayor control al tener cada silueta mejor marcada y más separada de las otras (Fisher et al. 2003). La forma lógica de realizarlo es similar a aplicar un filtro, es decir, hacer un barrido de la imagen pasando cada pixel por una función de transformación que aumente la distancia entre los pixeles.

En la Fig.6 se presenta un ejemplo de realizar una operación morfológica de dilatación

Figura 6: Aplicación de un filtro de dilatación

1.6.2 Categorización de siluetas:

A este punto ya es efectivo hacer la comparación de siluetas y así categorizar o elegir a aquellas que efectivamente se traten de siluetas humanas. Para este módulo se podrían realizar algoritmos propios de detección de contornos o crear y entrenar una red neuronal propia de detección de formas humanas. Sin embargo, para la arquitectura propuesta se utilizará el algoritmo de detección de contornos de Satoshi Susuki (Susuki y be, 1985). Y la red neuronal

de MobileNet (Howard et al. 2017), ambos ampliamente aceptados por su efectividad en sus respectivas tareas.

1.7 Conteo y rastreo de figuras humanas

Hasta el momento se han especificado las tareas a realizar para la detección humana. Sin embargo, también es necesario realizar una tarea de rastreo o tracking. El proceso de rastreo obedece a actualizar la posición de los objetos previamente encontrados a medida que estos se mueven de un frame a otro.

Para el proceso de rastreo se propone utilizar un algoritmo de rastreo por centroide, más conocido por su definición en inglés centroid tracking algorithm. Este es un algoritmo donde para cada objeto encontrado se busca las coordenadas del rectángulo que lo contiene y se calcula y almacena su centroide. Cuando se obtiene nueva información perteneciente al siguiente frame donde las coordenadas de los rectángulos contenedores han cambiado se procede a calcular estos nuevos centroides. Luego, se mide la distancia euclidiana entre todos los centroides. De esta manera se puede deducir que los centroides que más juntos se ecuentren probablemente se traten del mismo objeto que ha sufrido un mínimo desplazamiento (Rosebrock, 2018).

Finalmente, se actualiza el nuevo centroide de cada objeto y se continua con el proceso de forma iterativa. En la Fig.7 se detalla en una ilustración el funcionamiento del algoritmo. El proceso empieza una vez que dos objetos han sido encontrados. Luego un nuevo frame con dos objetos aparece, por lo tanto se hace un cálculo de distancia euclídea entre los centroides nuevos y antiguos. A partir de aquí se reconoce qué objeto antiguo está más cercano a cada objeto nuevo. Finalmente el sistema deduce que los objetos nuevos son realmente los objetos antiguos solo que desplazados una pequeña distancia.

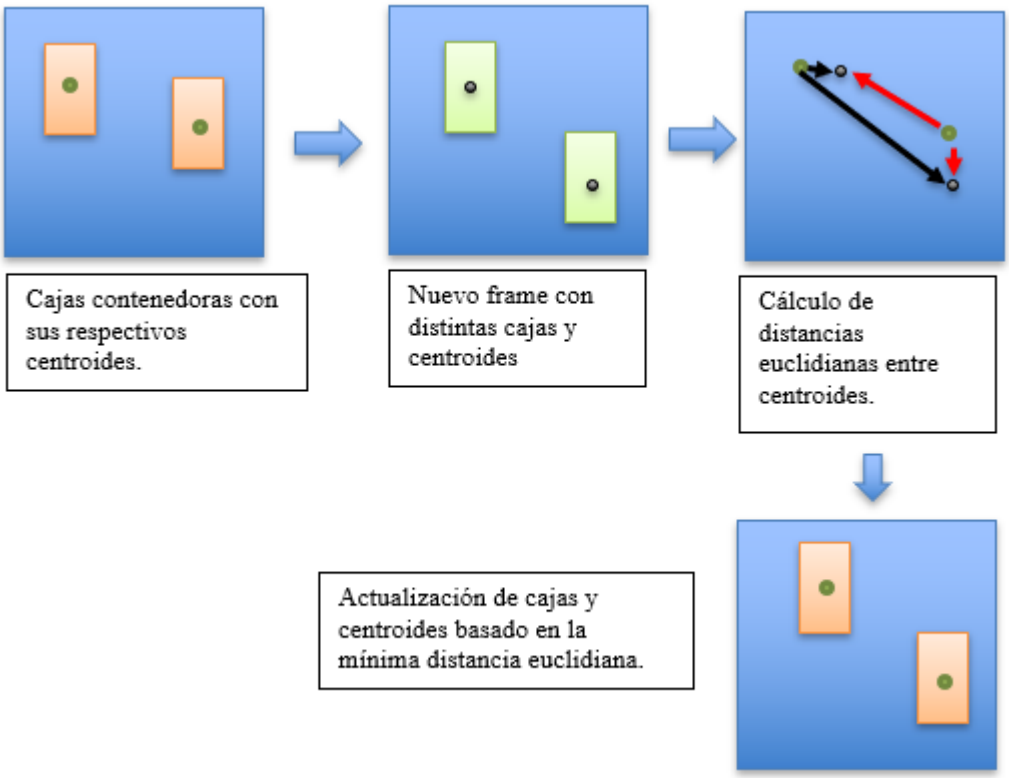

Figura 7: Funcionamiento de Centroid Tracking Algorithm

Cuando un centroide nuevo no puede ser asignado a ningún centroide antiguo se deduce que se trata de un nuevo objeto. De esta manera se puede reconocer cuando una nueva persona ingresa. Al mismo tiempo se hace una continua actualización de la localización del resto de personas previamente encontradas.

1.8 Alerta de incumplimiento de aforo

Hasta ahora se conoce si una persona se encuentra desplazándose, sin embargo para poder detectar si sale o entra del lugar se necesitan de dos argumentos nuevos a calcular.

1.8.1 Rango de entrada/salida:

Previamente se necesita conocer qué cámaras se encuentran enfocando una puerta de entrada o salida. Así podemos establecer un rango para que si alguna de las coordenadas del

centroide son mayores o menores a este se pueda deducir que dicho objeto a traspasado la frontera de entrada/salida.

1.8.2 Dirección de traslado:

No basta con conocer si un objeto ha traspasado la frontera, también se debe conocer en que dirección se estaba desplazando el objeto para finalmente poder decidir si traspasó la frontera al entrar o al salir. La dirección puede obtenerse fácilmente de restar las coordenadas del centroide nuevo de las del centroide antiguo. Para esta arquitectura se propone almacenar un vector con los valores de los últimos 100 centroides de un objeto, ya que al obtener la media de estos se tiene una mejor noción de si realmente el objeto está sufriendo un desplazamiento con dirección.

Si esta resta es menor que cero significa que el nuevo centroide tiene una coordenada menor que los anteriores, es decir el objeto está sufriendo un desplazamiento de derecha a izquierda o de abajo a arriba según sea el caso. Por el contrario un valor mayor que cero indica un nuevo centroide mayor que los anteriores, es decir un desplazamiento de izquierda a derecha o de arriba hacia abajo.

Una vez que se cuenta con la dirección de desplazamiento y con el rango de coordenadas que representan a la zona de entrada/salida del lugar finalmente se puede llevar un conteo del aforo actual (entradas menos salidas). Cuando este aforo es mayor al máximo permitido, previamente declarado en el sistema, se genera una alerta visual en el video mencionando que el aforo ha sido superado.

DESARROLLO DEL PROTOTIPO

2.1 Características técnicas del prototipo:

Para el desarrollo del presente prototipo se ha utilizado el lenguaje de programación Python, específicamente la versión 3.8.5. Como entorno de desarrollo o IDE por sus siglas en inglés se ha utilizado Sublime Text versión 3.2.2 y el plugin Anaconda versión 1.7.2. Este plugin es una distribución de Python destinado al procesamiento de datos masivos, análisis predictivo y programación científica. De igual manera Anaconda simplifica la gestión de paquetes (Anaconda, 2021). Esto es clave al momento de implementar la biblioteca OpenCV la cual será de gran importancia y utilidad en este proyecto.

OpenCV es una biblioteca multiplataforma programada en C/C++ que es líder en tareas de visión artificial, segmentación de objetos y detección de movimiento (OpenCV, 2021). Se trata de la biblioteca principal para el desarrollo de este prototipo. De igual manera, se utilizan las bibliotecas imutils, numpy, time y dlib para satisfacer todas las necesidades de programación posteriores.

Para el análisis y reconocimiento de personas se utiliza la red neuronal MobileNetSSD. Se trata de una de las redes neuronales mejor entrenadas y más respetadas internacionalmente para aplicaciones de visión artificial (Howard et al. 2017).

2.2 Aplicación de la arquitectura propuesta:

Haciendo uso de la biblioteca OpenCV mencionada en la sección 2.1, se puede acceder a la cámara web del ordenador para capturar hasta 30 frames por segundo y reproducirlos a manera de video en una ventana nueva. Así mismo es posible el manejo y edición de cada frame permitiendo el análisis necesario.

2.2.1 Recepción de imagen y detección del fondo

Para este procedimiento es clave que la primera imagen que obtenga la cámara sea la del espacio a analizar vacío. Si esto no es posible, se debe tener almacenada una fotografía del espacio vacío en cuestión. Esto es necesario para poder comparar los cambios en las imágenes y así detectar nuevas figuras y posteriormente analizar si son humanas.

A continuación se repasa todo el proceso de análisis de imagen mediante el prototipo. En la Fig. 8 se muestra una captura de lo que de ahora en adelante será el fondo del primer escenario de prueba. En el capítulo 3 se presenta otro escenario de estudio.

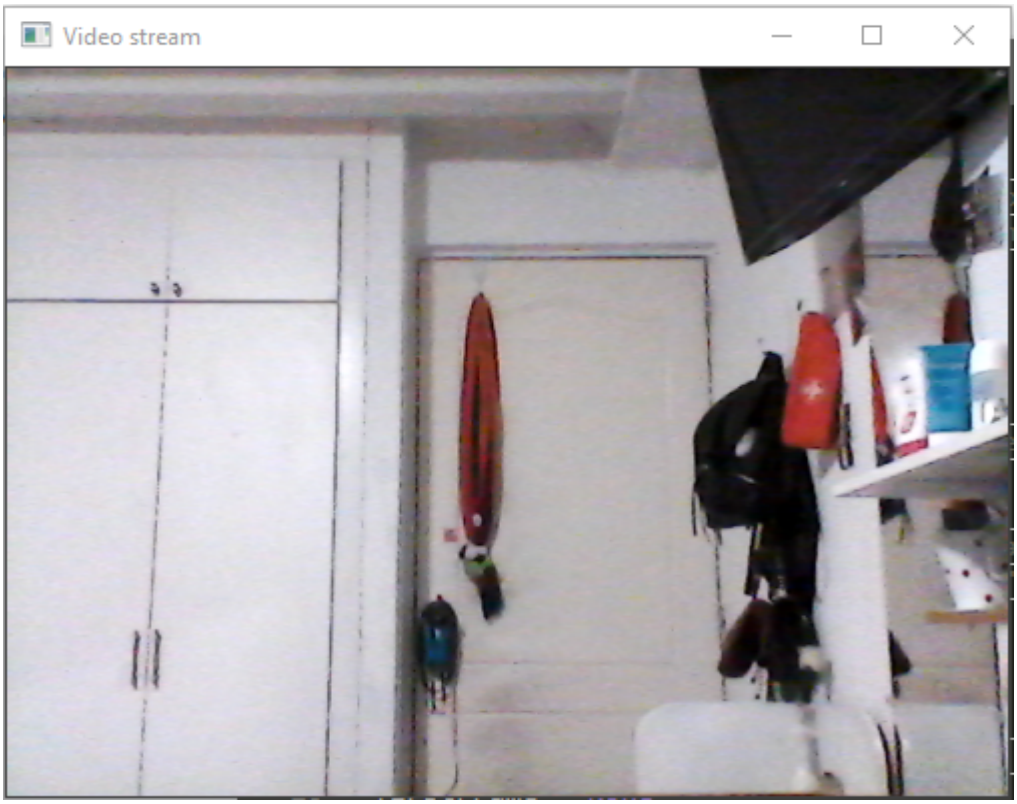

Figura 8: Escenario de pruebas vacío.

Una vez que se tiene listo el escenario ya se puede empezar a analizar cualquier frame futuro. En la Fig. 9 se observa un frame donde una persona ha invadido el fondo de estudio.

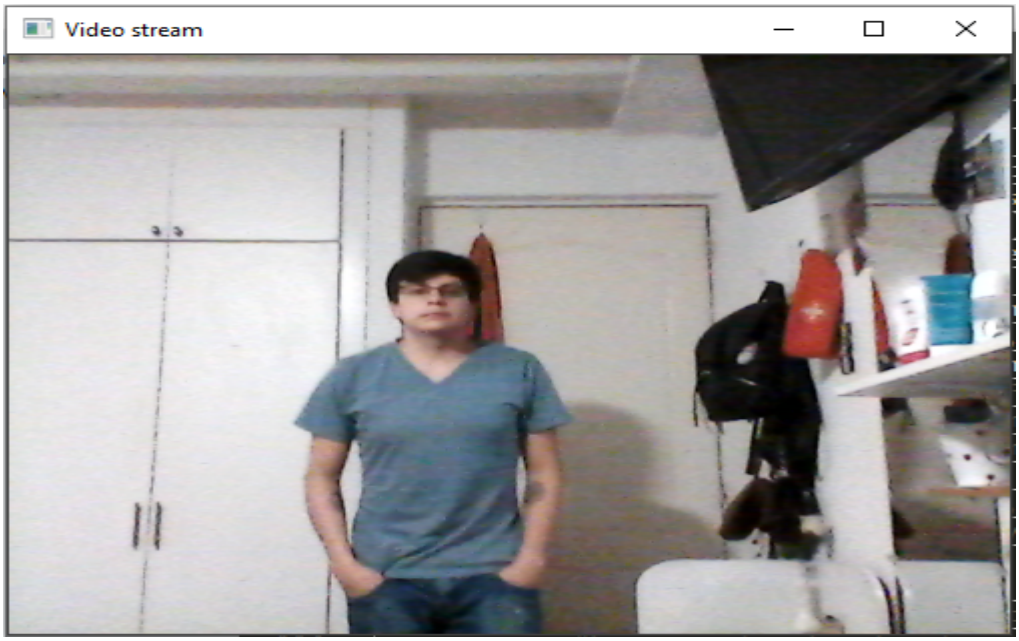

Figura 9: Frame a analizar

2.2.2 Optimización a escala de grises

Como se mencionó en la sección 1.2 del documento el primer proceso por el que debe correr cada frame a analizar es cambiar a una escala de grises. Este proceso puede ser realizado fácilmente con la ayuda de OpenCV. Esta librería dispone de múltiples métodos que nos permiten trabajar con imágenes. Uno de estos métodos es cvtColor(frame, change) el cuál necesita como argumentos, el frame a actualizar y el cambio de colores deseado. Este segundo argumento se debe elegir de entre los posibles valores de cambio soportados por el método, para este caso se elige el valor COLOR_BGR2GRAY. El resultado de aplicar esta transformación se muestra en la Fig. 10

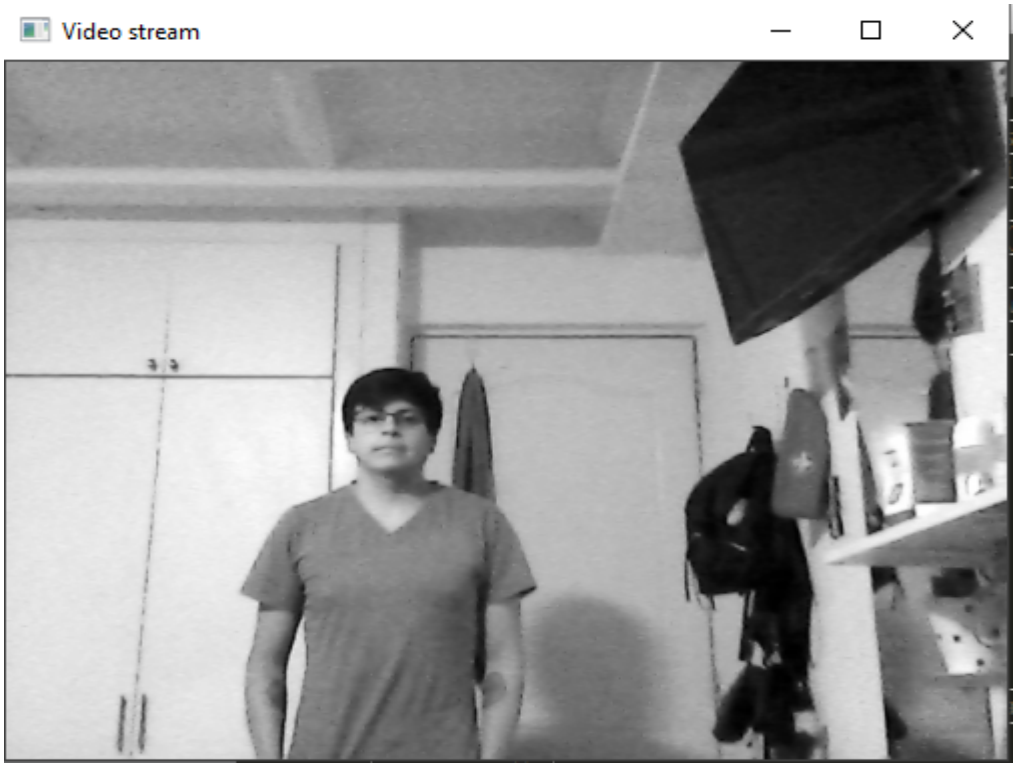

Figura 10: Cambio de imagen a escala de grises mediante OpenCV

2.2.3 Suavizar la imagen:

Siguiendo con el procedimiento que corresponde a la sección 1.3 del documento se procede a quitarle nitidez a la imagen. El distorcionar la imagen permite que el análisis de contornos sea más sencillo. Pues para este proyecto es más útil el análisis de contornos de figuras que la nitidez de las mismas. Para el caso de este prototipo se está utilizando un filtro Gaussiano de distorsión. Cabe recalcar que las secciones 2.2.3, 2.2.4 y 2.2.5 obedecen a un mismo comando, blobFromImage(frame, scale, size, mean). La utilización de este método de OpenCV realiza el proceso de las 3 secciones mencionadas, es decir, no es necesario realizar cada acción explícitamente.

Sin embargo, para un mejor entendimiento en la Fig. 11 se observa la aplicación de un filtro gaussiano a través de otro método de OpenCV, esto con el fin de entender de mejor manera el proceso que realiza el método blobFromImage por detrás.

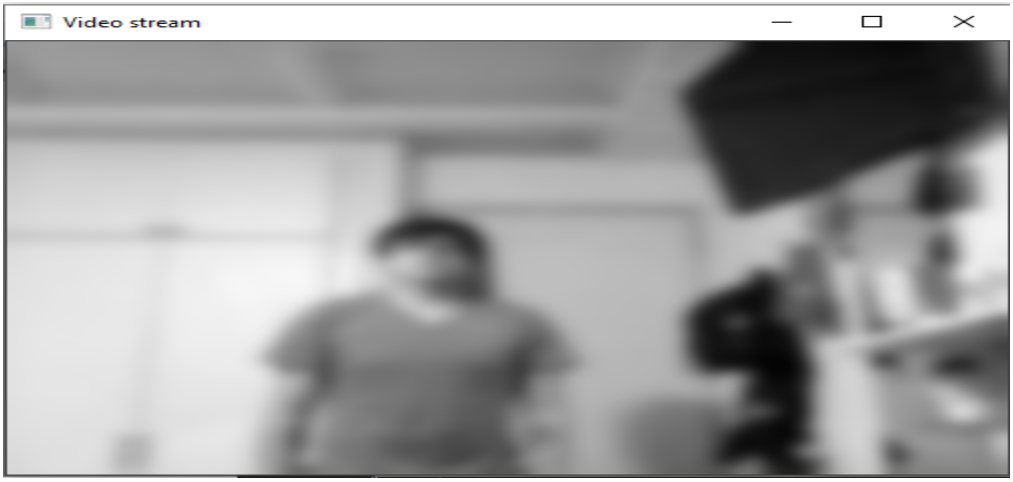

Figura 11: Aplicación de un filtro Gaussiano

2.2.4 Análisis de superposición de objetos:

Siguiendo con el proceso explicado en la sección 1.4 y Fig. 4 del documento, se contrasta el frame actual con el de la imagen del espacio vacío para obtener los pixeles diferentes entre ambos frames. Como ya se mencionó, este proceso es automático mediante blobFromImage. Sin embargo, en la Fig. 12 se muestra una representación del proceso con la ayuda de OpenCV.

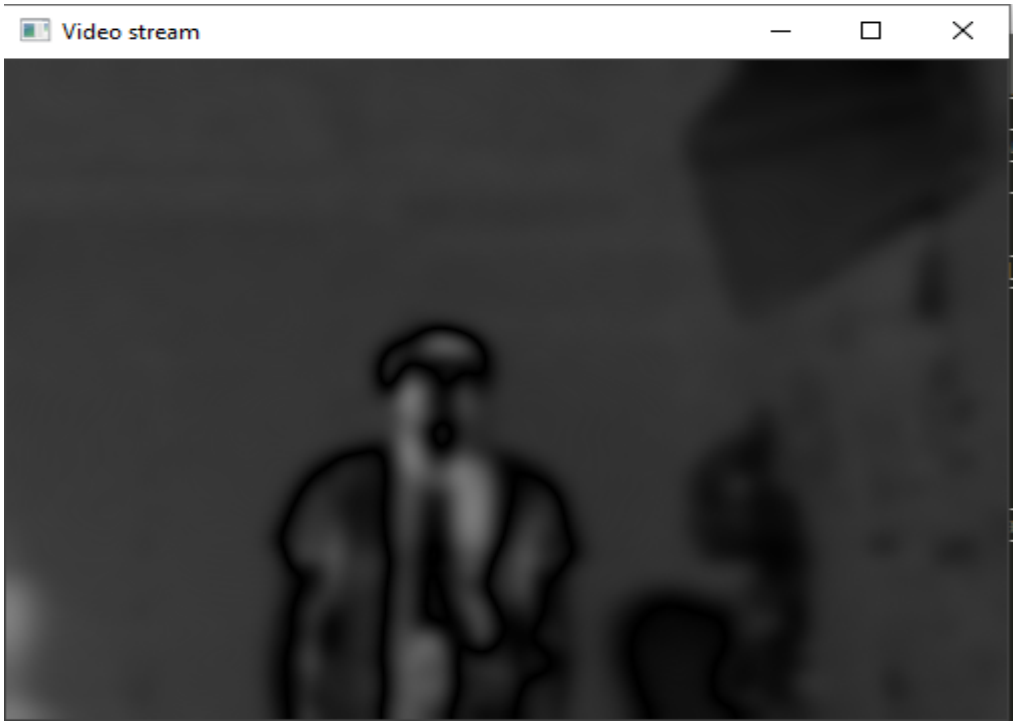

Figura 12: Análisis de objetos superpuestos

2.2.5 Representación de una segmentación binaria:

Una vez que se obtiene esta diferencia la imagen debe pasar por un proceso de threshholding o segmentación binaria tal y como se mencionó en la sección 1.5 del documento. De esta manera se pueden rescatar las siluetas de manera totalmente contrastadas. Este proceso es automático dentro del método blobFromImage pero una representación con la ayuda de OpenCV se muestra en la Fig. 13

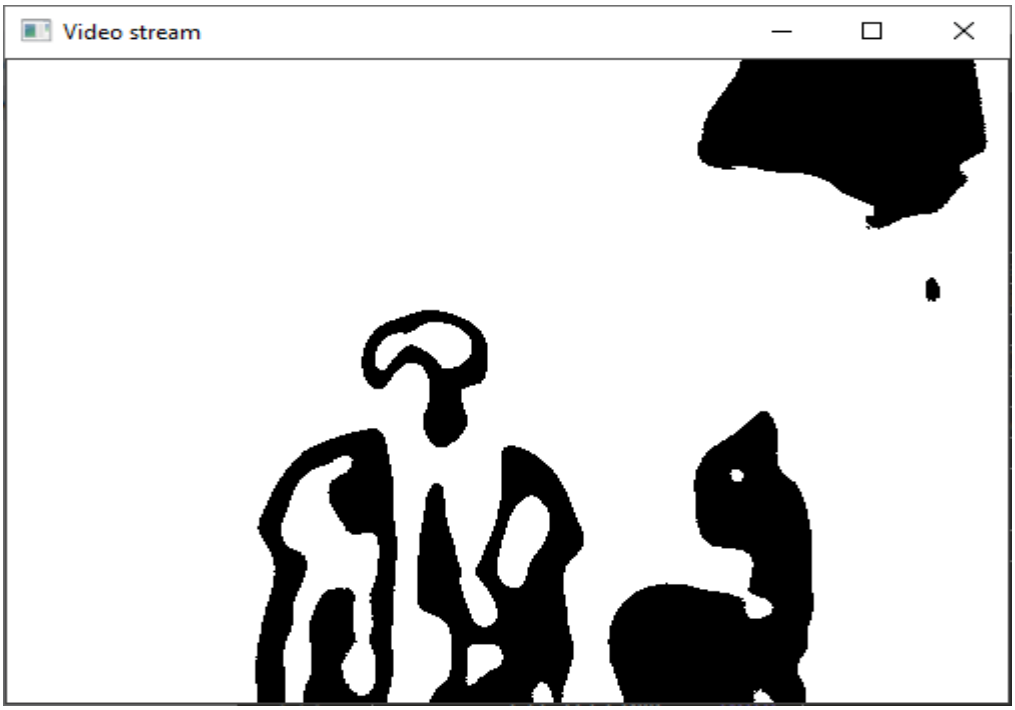

Figura 13: Aplicación de una segmentación binaria no inversa

2.2.6 Análisis de siluetas:

La red neuronal MobileNetSSD tiene dentro de sus utilidades el análisis automático de contornos. Esto nos permite obtener por separado cada figura encontrada hasta este punto. Con la ayuda de un bucle se puede analizar cada silueta para su respectivo análisis de contornos explicado a más detalle en la sección 1.6 del documento. MobileNetSSD, es una de las redes neuronales mejor entrenadas para el análisis de múltiples figuras, entre ellas las figuras humanas.

Dentro de la documentación de MobileNetSSD se establece que esta red debe implementarse y entrenarse mediante un modelo caffe. Es por esto que para el presente prototipo deben ser descargados los archivos .caffemodel y .prototxt de MobileNetSSD.

La implementación de la red también corre a cargo de OpenCV que mediante su método readNetFromCaffe(prototext, caffemodel) permite cargar cualquier red neuronal que utilice el framework Caffe. Una vez que la red ha sido cargada con éxito se puede utilizar el método de OpenCV setInput(image) que permite que la red analice la imagen de argumento, en este caso la imagen que se pasa como argumento es la resultante del proceso blobFromImage explicado en las secciones 2.2.3, 2.2.4 y 2.2.5.

De esta manera, la red se encarga de detectar y categorizar cada una de las siluetas detectadas en la imagen. De cada silueta se evalúa su categorización y nivel de confianza. Si su categorización es de tipo “person” y su nivel de confianza satisface un porcentaje mínimo entonces se enmarca el contorno de la silueta y se considera que una figura humana ha sido encontrada.

Cabe recalcar que la red MobileNetSSD tiene soporte para reconocer formas humanas desde varias perspectivas. Por ejemplo, en la Fig. 14 se observa la detección de una persona mirando de frente. La Fig.15 muestra la detección de una persona de perfil. En la Fig. 16 se ha localizado a una persona de espaldas. Finalmente, la Fig. 17 muestra solamente al busto de una persona, sin embargo la red neuronal también es capaz de identificarla.

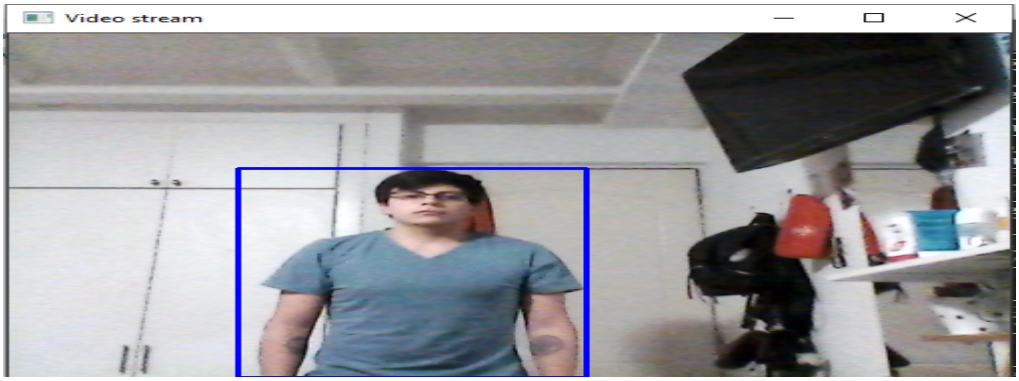

Figura 14: Detección persona de frente

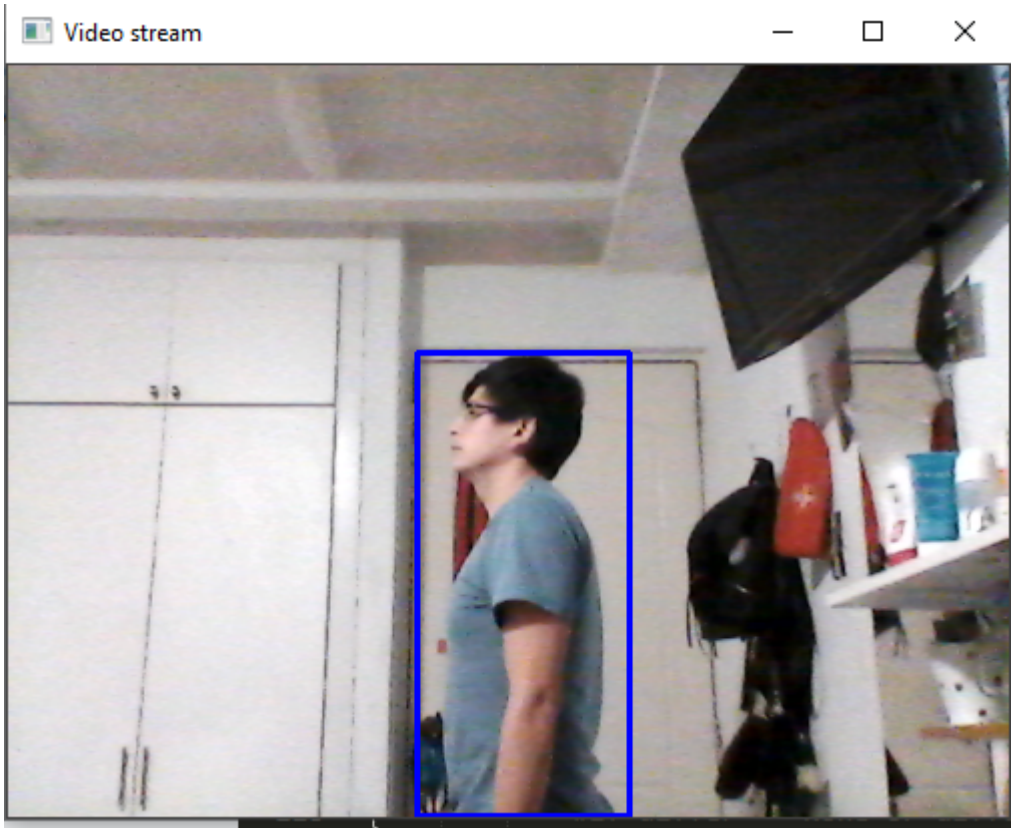

Figura 15: Detección persona de perfil

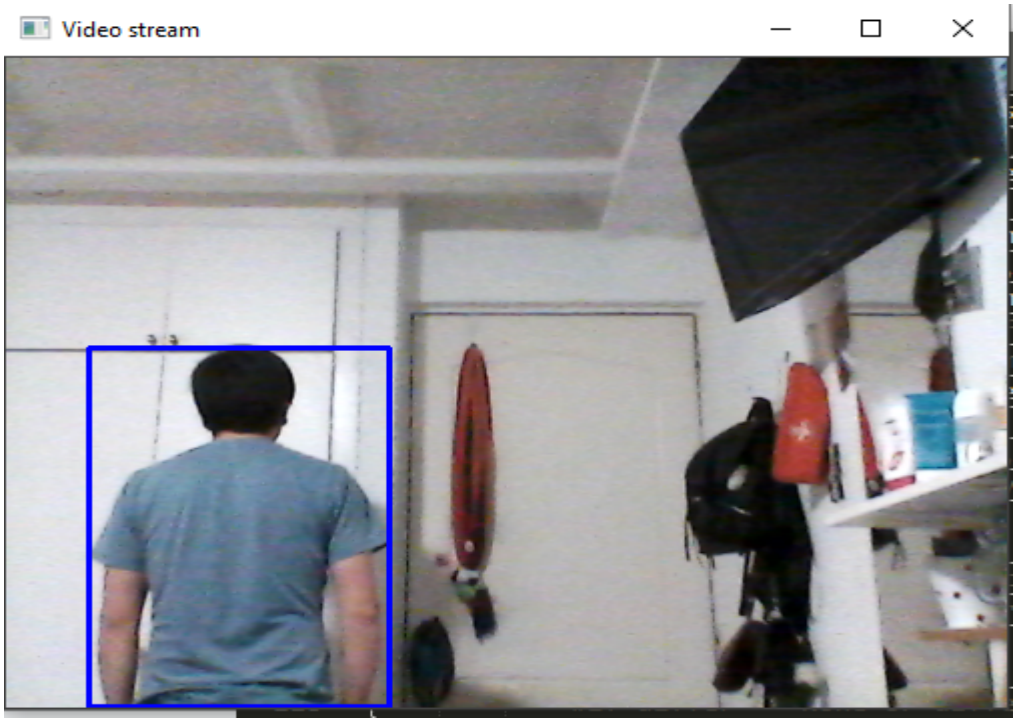

Figura 16: Detección persona de espaldas

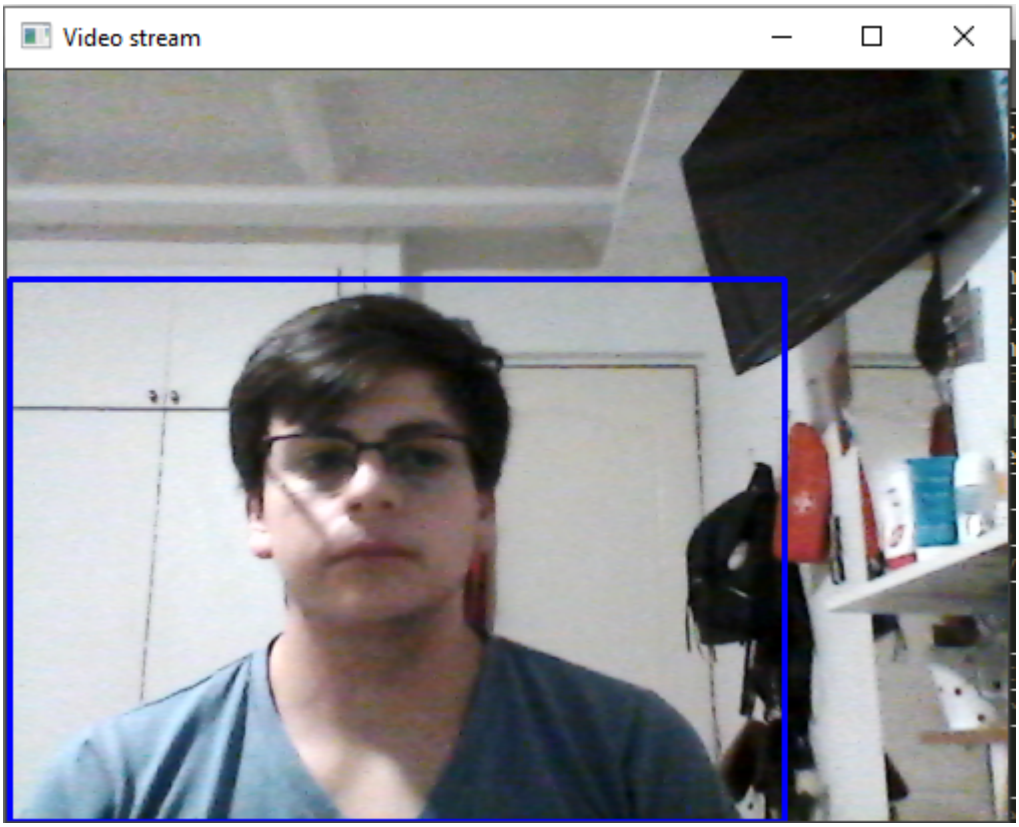

Figura 17: Detección busto de persona

2.3 Desarrollo para el rastreo de objetivos:

Para el rastreo o tracking se ha creado una clase llamada CentroidTracker que consta de tres métodos: nuevo(centroide) para crear un objeto nuevo, elminar(ID) para elminar un objeto que ha salido del cuadro visual de la cámara y update(objetos) la función que realizará la actualización de movimiento de cada objetivo. A continuación, se detalla el funcionamiento de cada método en orden de importancia.

2.3.1 Médodo Update de CentroidTracker:

Como se explicó en la sección 2.2.6 del documento, al aplicar, ejecutar y validar los resultados obtenidos a través de la red neuronal se tiene un conjunto de rectángulos con coordenadas que marcan el espacio donde hay una silueta humana. Este conjunto de rectángulos es el argumento objetos que debe ser utilizado al llamar a update(objetos). Este método es el más importante del rastreo ya que es el encargado de llamar a los métodos nuevo() y eliminar() cuando lo considere necesario.

Para cada uno de estos rectángulos se calcula su centroide y se hace un análisis siguiendo el algoritmo del centroide explicado en la sección 1.7 del documento. Si es la primera vez que este método se ejecuta resulta evidente que todos los objetos serán reconocidos como nuevos ya que no existe un listado previo de centroides con el cual se pueda comparar las distancias. Cuando esto ocurre o cuando uno de los nuevos centroides no puede ser relacionado con algún centroide ya existente se procede a ejecutar el método nuevo(centroide) que será explicado con más detalle en la sección 2.3.2 del documento.

Por el contrario, cuando uno de los centroides antiguos no puede ser relacionado con ninguno de los nuevos centroides se deduce que este objeto ya no se encuentra presente y se procede a ejecutar el método eliminar(ID) explicado más detalladamente en la sección 2.3.3 del documento.

2.3.2 Método Nuevo de CentroidTracker:

Este método recibe como argumento las coordenadas del centroide que no pudo ser relacionado con ningún centroide pre-existente. El funcionamiento es bastante sencillo, simplemente agrega estos valores a un diccionario centroides(ID, centroide) donde ID es creado automáticamente respecto a cuantos centroides se encuentran ya en el diccionario y centroide corresponde a las coordenadas del centroide. Este diccionario actualizado es el que será utilizado para la comparación de centroides la próxima vez que update sea ejecutado.

2.3.3 Método Eliminar de CentroidTracker:

Este método recibe como argumento el identificador del elemento a eliminar del centroide antiguo que no pudo ser asignado a ninguno de los centroides nuevos. Su funcionamiento también afecta al diccionario centroides descrito en la sección 2.3.2, la diferencia es que este método ejecuta una acción de eliminar el elemento centroides(ID, centroide) tal que ID sea igual al identificador dado en el argumento. Nuevamente la actualización de este diccionario será la que sea utilizada la próxima vez que update sea ejecutado.

2.4 Conteo de objetivos:

Para el conteo se utilizan variables globales para que, sin importar el proceso que cambie su valor, este se vea reflejado para el resto del código. Esto permite tener un buen control de la cantidad total y exacta de personas que se encuentran en el espacio. Así, se resuelve el problema de salida de cuadro. Este problema es común cuando una persona está traspasando de un enfoque de visión de una cámara a otra y desaparece momentanemante de las imágenes. Sin embargo, no ha salido del espacio y, por lo tanto, no debería afectar al aforo total.

Los procesos encargados de modificar el conteo se dan una vez terminado el método update explicado en la sección 2.3.1. Para cada objeto cuya ubicación ha sido actualizada se analiza la dirección en la que se ha desplazado el objeto, esto siguiendo la lógica explicada en la sección 1.8.2 del documento. De igual manera, se analiza si las coordenadas del objeto después del desplazamiento están dentro del rango de entrada/salida explicado en la sección 1.8.1.

A partir de esta información y mediante la lógica presentada en la tabla 1, se toma una decisión de si el objetivo ha salido o entrado en el lugar. Dado que las entradas y salidas son propias de cada lugar a analizar y no se puede generalizar que siempre se encontrarán en lugar en específico o que la salida siempre es a la derecha, izquierda, arriba o abajo de un lugar, se ha optado por representar las direcciones y posición de una manera más descriptiva que explique de mejor manera la lógica a utilizar. Los valores sobre qué dirección corresponde a entrar o salir y las coordenadas que marcan los límites de entrada/salida deben ser configuradas en código para cada espacio a analizar distinto.

Dirección	**Posición**	**Resultado**
Entrada	Superior a rango de entrada	Si entra
Entrada	Inferior a rango de entrada	No entra
Salida	Superior a rango de salida	Si sale
Salida	Inferior a rango de salida	No sale

Tabla 1: Tabla de verdad para entrada/salida de objetivos

\

VALIDACIÓN DEL PROTOTIPO Y RESULTADOS

El prototipo presentado es la tercera versión desarrollada. A continuación, se hace un breve recorrido sobre las versiones previas y sus respectivos problemas que fueron solucionados.

3.1 Versión 1.0:

Esta versión corría sobre un video de prueba, reconocía figuras humanas, las marcaba visualmente en un rectáangulo y hacía un rastreo de cada una. Los problemas encontrados en esta versión se explican a continuación.

No se podía realizar un conteo correcto ya que como máximo solo se pudo determinar la cantidad de personas que la red neuronal encontraba en cada iteración. Esto no permitía crear un margen de entrada y salida ya que solo se asumía que toda persona en rango de visión se encontraba dentro del lugar, lo cuál no siempre es cierto. Para esta versión el rastreo no se realizó con el algoritmo del centroide, sino que, se utilizó el método TrackerCSRT_create() de la biblioteca OpenCV. A breves rasgos, este método utiliza un algoritmo que analiza el área cercana a un objeto encontrado en búsqueda de un posible desplazamiento. Si bien es un algoritmo útil, demanda muchos recursos computacionales y como resultado el video se ralentiza y el funcionamiento no es ágil.

Así mismo, se desconocía todavía del potencial de la red MobileNetSSD para encontrar varias siluetas por frame por lo cual no se le pasó como argumento toda la imagen a analizar, sino que, se creaba una nueva imagen para cada una de las siluetas encontradas por frame. Esto también contribuia a un gasto elevado de recursos.

Otro error de esta versión que, si bien no fue visible en su momento, causó problemas después es que se utilizó un valor de confianza muy alto para la validación de si el objeto

encontrado correspondía a una forma humana. Esto con el afán de que el programa sea más exacto, pero como se verá en la sección 3.3 se estaba comentiendo un error.

3.2 Versión 2.0:

Esta versión utilizó videos obtenidos directamente de webcams conectadas al computador. De esta manera se trabajó de forma más rápida que con videos que tenían que cargarse en memoria. Para esta versión también se decidió experimentar el análisis de imagen mediante el método createBackgroundSubtractorMOG() de OpenCV. Este método permite eliminar directamente el background de un frame y separa solamente los objetos que se encuentran en movimiento. Como resultado se eliminaban pasos y se obtenia directamente un imagen con segmentación binaria de los objetos reconocidos como ajenos al fondo. Nuevamente, se trató de un algoritmo útil pero que presentaba errores para los objetivos del prototipo. Por ejemplo, cuando una persona se quedaba quieta dejaba de reconocerla y esta pasaba a ser parte del fondo.

Esto desencadenó constantes errores en el conteo puesto que cuando una persona dejaba de moverse para el programa debaja de existir. Así mismo, el método createBackgroundSubtractorMOG() resultó ser efectivo solo para ciertos escenarios donde la luz del lugar y la distancia de las cámaras era propicia. Esto hacía que la efectividad del programa dependa de las características físicas del entorno.

Un último error de esta versión fue el intentar filtrar las siluetas dependiendo de su área. Esto se hizo con el afán de eliminar desde un principio siluetas muy grandes o muy pequeñas y quitarle así peso a la cantidad de siluetas que tenía que analizar la red neuronal. Esta idea fue desechada en la posterior versión dado que lejos de ayudar contribuía a descartar algunas siluetas que si debían ser analizadas.

3.3 Versión 3.0

Finalmente, esta es la versión con la que se decidió realizar las pruebas obteniendo los mejores resultados. Las principales mejoras se comentan a continuación.

La red neuronal es lo primero a cargar en el programa, esto permite que se trabaje de manera más rápida, puesto que en las versiones anteriores la red neuronal era cargada nuevamente en cada iteración haciendo el proceso más lento. Se implementa y trabaja con el algoritmo del centroide para la realización del rastreo. El margen de confianza para el reconocimiento de figuras humanas se ajusta dependiendo del espacio a analizar ya que para sistemas cuyas cámaras enfocan desde muy lejos captan siluetas más pequeñas que si el margen de confianza es muy alto no son reconocidas.

Se decide hacer el análisis de siluetas cada *N* cantidad de frames. Esto agilita la velocidad del programa ya que el proceso de reconocimiento de siluetas es bastante complejo tal como se describrió en el capítulo 1. Esta idea se basa en que los cambios visuales que existen de un frame a su inmediato siguiente son mínimos. Analizar cada *N* frames permite liberar recursos y aliviar las operaciones totales. Este valor *N* debe ser configurado para cada caso ya que depende del flujo medio de personas del lugar y de la capacidad de frames por segundo que pueda analizar el computador donde el programa es ejecutado.

Se introduce el análisis de dirección y posición explicados en la sección 2.4 del documento, permitiendo realizar un control efectivo de objetivos y solucionando el problema de salida de cuadro.

La validación de este prototipo se muestra a continuación para dos cámaras de seguridad y un aforo máximo de 3 personas.

Figura 18: Persona 1 en cámara 1

En la Fig. 18 se aprecia como la persona ingresó solo por el rango de visión de una de las cámaras sin embargo el aforo es actualizado a 1 para ambas cámaras ya que para el sistema es importante el aforo total.

Figura 19: Persona 1 y 2 en cámara 1

Figura 20: Persona 1 en cámara 1 y persona 2 saliendo del cuadro

Figura 21: Persona 1 en cámara 1, persona 2 en cámara 2

En las Figuras 19, 20 y 21 se observa como se maneja un problema de salida de cuadro, donde una de las personas desaparece momentáneamente del campo de visión de una de las cámaras y luego aparece en el otro. Sin embargo, el aforo no se ve afectado ya que se detecta que nadie ha salido o entrado de la escena completa.

Figura 22: Aforo superado

Figura 23: Aforo correcto

Las Figuras 22 y 23 muestran un caso donde el aforo es superado momentáneamente y se genera la etiqueta de advertencia en el video. Sin embargo, momentos después una de las personas abandona el lugar y el aforo vuelve a estar dentro del permitido.

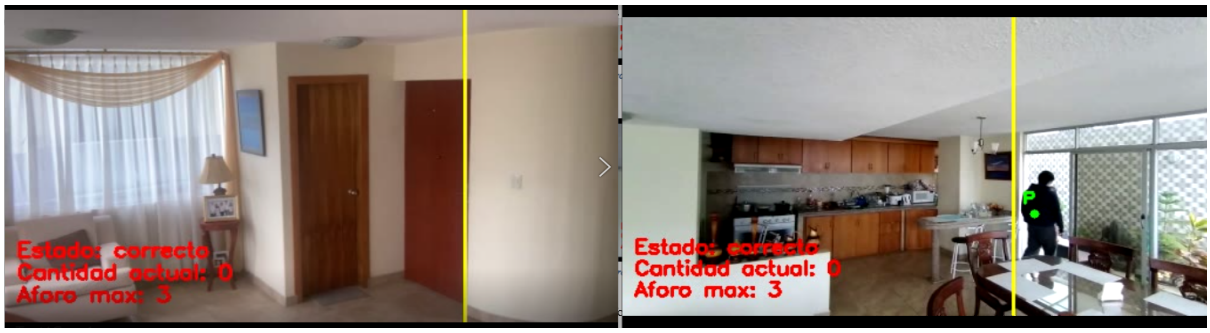

Figura 24: Aforo vuelve a cero

En la Fig.24 la última persona abandona el lugar y el aforo es actualizado a cero.

CONCLUSIONES Y TRABAJOS FUTUROS

Tanto la arquitectura propuesta como su prototipo implementado funcionan para el problema presentado. Se ha conseguido implementar técnicas de análisis de imagen, redes neuronales y rastreo de objetivos para obtener un sistema de video vigilancia inteligente para el control de aforos. Esto podría presentar mejoras en la agilidad y eficacia con la que hoy en día se manejan estos controles.

Actualmente, el uso de análisis inteligente de imágenes tiene su fuerte en el reconocimiento facial y detección de objetos para vehículos autónomos. Con este proyecto se espera ampliar el campo de interés y utilidad del análisis inteligente de imágenes. El proyecto contempla una investigación completa sobre los últimos avances y técnicas de detección humana y procesamiento de imágenes. En la actualidad, estas técnicas aún mantienen un alto consumo de recursos computacionales, pero se espera que con el tiempo estas operaciones puedan ejecutarse con mayor eficacia y facilidad.

A manera de sugerencia, queda la ampliación de funcionalidad del programa. Se podría reconocer sub-aforos dentro de cada escenario. De esta forma no solo se tendría en cuenta que no se incumpla el aforo total, sino que se cumpla el aforo de cada zona en específico. Para futuras implementaciones se contempla también módulos extras como el reconocimiento autónomo del aforo máximo mediante el cálculo del volumen del lugar, análisis del distanciamiento de las personas, y el uso de artefactos como mascarillas o credenciales para poder acceder a un lugar específico.

La mayor dificultad de este proyecto se basa en el reconocimiento humano, más específicamente en su sensibilidad a la luz. Los pasos repasados durante el documento para encontrar una silueta pueden verse ampliamente afectados cuando existen cambios de luz extremos ya que pueden identificarse sombras o reflejos como siluetas humanas. De igual

manera, cuando una persona recibe demasiada luz el brillo hace que sea sumamente difícil poder aislar su silueta y por ende se trasforma en un objetivo difícil de encontrar y posteriormente rastrear.

Sin embargo, para un ambiente ligeramente controlado y manejando una posición favorable de las cámaras se puede conseguir resultados altamente efectivos. Se concluye que el proyecto ha sido útil para enriquecer el aprendizaje y fomentar a la investigación. Se han cumplido los objetivos establecidos dentro del mismo, proponiendo una arquitectura para la creación de un sistema de video vigilancia inteligente para el control de aforos y comprobando su funcionalidad mediante un prototipo de bajo costo.

REFERENCIAS BIBLIOGRÁFICAS

Anaconda. (2021). *Anaconda Documentation.* Anaconda Inc. Obtenido el 18 de septiembre de 2021 de https://docs.anaconda.com/

Authentise. (6 de junio de 2016). *How to track objects with stationary background?* Authentise. Philadelphia. USA. Obtenido el 16 de septiembre de 2021 de https://www.authentise.com/post/how-to-track-objects-with-stationary-background

BBC. (1 de marzo de 2021). *Coronavirus: las extraordinarias medidas que están tomando ciudades de todo el mundo ante el virus (y cuáles se están aplicando en América Latina).* BBC NEWS. Obtenido el 14 de septiembre de 2021 de https://www.bbc.com/mundo/noticias-51687968

Bernal, J. (Apr. Jun. 2021). *Pandemias y Aglomeraciones.* Revista Medicina. Colombia. Obtenido el 14 de septiembre de 2021 de https://revistamedicina.net/ojsanm/index.php/Medicina/article/view/1604/2055

El Universo. (8 de febrero de 2021). *Más de 10 mil personas causaron aglomeraciones en Quito en la jornada electoral del domingo 7 de febrero.* El Universo. Obtenido el 14 de septiembre de 2021 de https://www.eluniverso.com/noticias/2021/02/08/nota/9618423/mas-10-mil-personas-causaron-aglomeraciones-quito-jornada-electoral/

Fisher, R. & Perkins, S. & Walker, A. & Wolfart, E. (2003). *Dilation.* HIPR2. Obtenido el 16 de septiembre de 2021 de https://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm

Howard, A. & Zhu, M. & Chen, B. & Kalenichenko, D. & Wang, W. & Weyand, T. & Andreetto , M. & Hartwig, A. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.* Cornell University. Obtenido el 20 de septiembre de 2021 de https://arxiv.org/abs/1704.04861v1

Olveres, J. & Escalante, B. (2011). *Convolusión y Filtrado.* UNAM. México. Obtenido el 16 de septiembre de 2021 de http://lapi.fi-p.unam.mx/wp-content/uploads/6-Filtros-y-morfologia.pdf

OpenCV. (2021). *Open Source Computer Vision Library.* Doxygen. Obtenido el 18 de septiembre de 2021 de https://docs.opencv.org/master/d1/dfb/intro.html

OpenCV. (2021). *Smoothing Images.* Doxygen. Obtenido el 18 de septiembre de 2021 de https://docs.opencv.org/4.5.2/dd/d6a/tutorial_js_filtering.html

Pérez, J. & Gardey, A. (2009). *Definición de Video.* Definición.de. Obtenido el 16 de septiembre de 2021 de https://definicion.de/video/

Pérez, J. & Merino, M. (2009). *Definición de Pixel.* Definición.de. Obtenido el 16 de septiembre de 2021 de https://definicion.de/pixel/

Rosebrock, A. (23 de julio de 2018). *Simple Object Tracking with OpenCV.* Pyimagesearch. Obtenido el 20 de septiembre de 2021 de https://www.pyimagesearch.com/2018/07/23/simple-object-tracking-with-opencv/

Suzuki, S. & be, K. (1985). *Topological structural analysis of digitized binary images by border following.* (Vol. 30 pp. 32-46). Obtenido el 20 de septiembre de 2021 de https://doi.org/10.1016/0734-189X(85)90016-7.

Zhang, J. & Zhang, L. & Teng, Y. & Zhang, X. & Wang, S. & Ju, L. (2018). *Interactive Binary Image Segmentation with Edge Preservation.* Cornell University. Obtenido el 20 de septiempre de 2021 de https://arxiv.org/abs/1809.03334