Diseño e Implementación de un sistema de identificación de personas para la seguridad de los accesos a condominios, basado en el algoritmo de reconocimiento facial LBPH Faces

Verdeguer-Valderrama Diego, Campos-Vasquez Neicer, Maestro en Ciencias Económicas
Universidad Privada del Norte, Perú, N00159324@upn.pe, neicer.campos@upn.edu.pe

Resumen– El trabajo que a continuación se presenta tiene como propósito el diseño e Implementación de un sistema de identificación de personas para la seguridad de los accesos a condominios, basado en el algoritmo de reconocimiento facial LBPHFaces. Se identificaron distintos trabajos de investigación con respecto al reconocimiento facial para la seguridad de las personas y de esa manera analizar los métodos algorítmicos que utilizaron. Se realizó la captura de 300 imágenes de los rostros de las personas que el sistema identificará y se almacenaron en un repositorio con el nombre del rostro de la persona. El algoritmo identifica los rostros, transforma las imágenes a escala de grises y los entrena para identificar si el rostro en tiempo real que muestra la cámara de vigilancia tiene similitud a las imágenes entrenadas dentro del repositorio. Se analizaron los 3 métodos más utilizados, que son el FisherFaces, EigenFaces y el LBPHFaces. Al final los resultados arrojaron valores muy positivos con respecto al algoritmos escogidos y se concluyó que el LBPHFaces fue superior a los otros dos mencionados en tiempo de respuesta, de entramiento y porcentaje de aciertos, lo cual lo hace el más confiable para la seguridad de las personas.

Palabras Clave: Reconocimiento Facial, Nivel de Confianza, algoritmo LBPH Faces.

Abstract– The purpose of the work presented below is the design and implementation of a person identification system for the security of access to condominiums, based on the LBPHFaces facial recognition algorithm. Different research works were identified with respect to facial recognition for the safety of people and thus analyze the algorithmic methods they used. 300 images of the faces of the people that the system will identify were captured and stored in a repository with the name of the person's face. The algorithm identifies the faces, transforms the images to grayscale and trains them to identify if the face in real time displayed by the surveillance camera has similarity to the images trained within the repository. The 3 most used methods were analyzed, which are the FisherFaces, EigenFaces and the LBPHFaces. In the end, the results yielded very positive values with respect to the chosen algorithms, and it was concluded that the LBPHFaces was superior to the other two mentioned in response time, training and percentage of hits, which makes it the most reliable for the safety of the people.

Key Words: Facial Recognition, Confidence Level, LBPH Faces algorithm.

Digital Object Identifier: http://dx.doi.org/10.18687/LACCEI2021.1.1.213
ISBN: 978-958-52071-8-9 **ISSN**: 2414-6390
DO NOT REMOVE

I. INTRODUCCION

La tecnología avanza día a día, ahora se habla de ambientes inteligentes, computación ubicua, dispositivos inteligentes, lo cual está influenciando y cambiando la forma de vida de las personas. (Pentland & Choudhury, 2000). Como menciona Russo C. (Informática y Tecnologías Emergentes, ET, 2019): "Las tecnologías emergentes son innovaciones en desarrollo que como su nombre lo dice en un futuro cambiarán la forma de vivir del ser humano brindándole mayor facilidad a la hora de realizar sus actividades." Es por ello por lo que no es novedad que utilicemos muchas de estas tecnologías emergentes para reducir el índice de robos en todos los establecimientos ya mencionados y es aquí donde nos hacemos la siguiente pregunta, ¿Qué tan seguro es la implementación de un sistema de detección de rostros utilizando el reconocimiento facial para los condominios?

Por esa razón se han desarrollado sistemas inteligentes capaces de detectar los rostros de aquellas personas que han sido capturadas por una cámara de videovigilancia o que simplemente su rostro aparezca en los diarios o televisión como "persona de cuidado". Se han implementado varios sistemas de reconocimiento facial que pueden reconocer el rostro de aquellas personas e informar cuando el sistema inteligente los detecte. Ningún sistema de seguridad es confiable al cien por ciento, muchos tienen distintos algoritmos que pueden cambiar la precisión con la que se pueden detectar los rostros. Pero con esta investigación se desea cumplir con esta necesidad que tienen muchas personas.

La inseguridad ciudadana en el Perú es cosa de todos los días, los constantes robos a los locales comerciales, transeúntes y a las casas de las personas nos hace pensar que no estamos seguros en ningún lado. Estos actos son a menudo por los mismos delincuentes, estova de la mano con que se logra capturar a la persona, pero muchas veces este queda libre y esto es lo que ha sucedido con el condominio de San Antonio de Carabayllo, ha sido blanco de los delincuentes en varias ocasiones, esto debido a la falta de seguridad de la urbanización.

Actualmente el condominio no cuenta con un sistema de videovigilancia con detección de rostros, solo cuentan con una cámara de seguridad en la entrada que solo verifican el ingreso

y salida de los propietarios de los departamentos. Es por ello por lo que se planteó diseñar e implementar un sistema que pueda reconocer el rostro de las personas al momento que una persona desee ingresar al condominio. Se le identificará el rostro mediante la cámara de seguridad de la entrada, si el rostro está en la base de datos de los usuarios permitidos, se habilitará para que pueda ingresar, caso contrario lanzará una alerta, se capturará la foto y se guardará. La puerta no se abrirá y el guardia procederá a intervenir a la persona.

Se utilizará el lenguaje de programación Python ya que es mucho más accesible al querer implementar sistemas con inteligencia artificial, porque se pueden hacer uso de múltiples librerías al mismo tiempo, tales como numpy, Scipy, OpenCV y muchas más. Se utilizarán clasificadores de cascada, ya que tienen predefinidas para la detección de rostros y perfiles. Se empleará el algoritmo de LBPHFaces para la precisión con la que se identificarán los rostros. El sistema tendrá una interfaz gráfica en la que el usuario encargado podrá visualizar la lista de las personas autorizadas con sus datos personales (Nombres, teléfonos, pisos en el cual viven). De esta manera se tendrá un mejor control de los propietarios.

Local Binary Pattern (LBP) es un operador de textura simple pero muy eficiente que etiqueta los píxeles de una imagen mediante el umbral de la vecindad de cada píxel y considera el resultado como un numero binario. Debido a su poder discriminativo y simplicidad computacional, el operador de textura LBP se ha convertido en un enfoque popular en diversas aplicaciones. Quizás la propiedad más importante del operador de LBP en aplicaciones del mundo real es su robustez frente a los cambios monótonos de la escala de grises causados, por ejemplo, por las variaciones de iluminación. Otra propiedad importante es su simplicidad computacional, que hace posible analizar imágenes en entornos desafiantes en tiempo real. Usando el LBP combinado con **histogramas** crea el **LBPH** con el cual podemos representar las imágenes de la cara con un vector de datos simple. Como LBP es un descriptor visual, también se puede usar para tareas de reconocimiento facial, como se puede ver en la siguiente explicación paso a paso. (Ahonen, Hadid & Pietikainen, 2006).

Se utilizará librerías de OpenCV, que es una API de aproximadamente de 300 funciones escritas en lenguaje C, que se caracteriza por ser de uso libre. Estas librerías proporcionarían un marco de trabajo de nivel medio-alto que ayudaría al personal docente e investigador a desarrollar nuevas formas de interactuar con los ordenadores. V. M. Arévalo, J. González, G. Ambrosio (2004). OpenCV nos ofrece clasificadores pre entrenados no solo de rostros de personas, sino de ojos, sonrisas, perfiles, animales, entre muchas otras.
Si algún propietario desea agregar el rostro de algún familiar, también se podrá realizar.

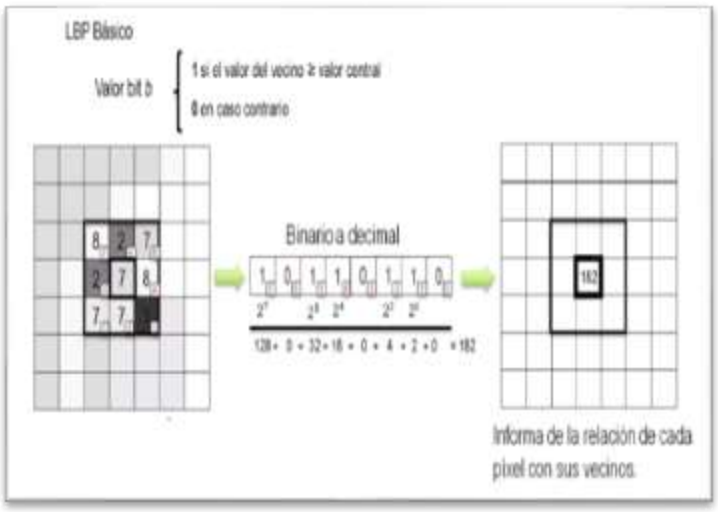

Figura 1. Proceso de **LBP**

Y por último el sistema estará en constante funcionamiento ya que está integrado con la cámara de videovigilancia del edificio. Al final el sistema se implementará en el condominio y se realizarán pruebas con el rostro de las personas autorizadas y con desconocidos.

Mediante nuestro software para este condominio se mantendrá a los propietarios correctamente informados respecto a la seguridad de sus departamentos y volverán a tener la confianza de dejar sus casas al realizar algún viaje que los demande varias horas. A la vez, nuestro software como ya se mencionó, identificará a todas las personas que pasen por la entrada del condominio.

II. PROCESO

¿Como se calcula el código LBP para cada una de las imágenes? El primer paso computacional del **LBPH** es crear una imagen intermedia que describa la imagen original de una mejor manera, resaltando las características faciales. Primero debemos localizar un punto en la imagen, en este caso la imagen tendrá una dimensión de 7x7 en donde cada casilla representa un píxel. Luego se va a calcular el código LBP para el píxel central

Ese cálculo se va a basar en primero definir una vecindad del píxel y luego ir comparando los niveles de gris del pixel central con los vecinos. Los vecinos en este caso son todos los que rodean al pixel central. Luego debe de aplicar una regla para los pixeles vecinos. Para cada vecino del valor central (umbral), establecemos un nuevo valor binario. Establecemos 1 para valores iguales o superiores al umbral y 0 para valores inferiores al umbral. Ahora, la matriz contendrá solo valores binarios (ignorando el valor central). Como hay 8 vecinos, en realidad se forma un byte donde cada bit tendrá un valor de 0 y 1.

Necesitamos concatenar cada valor binario de cada posición de la matriz línea por línea en un nuevo valor binario (por ejemplo,

10110110). Nota: algunos autores utilizan otros enfoques para concatenar los valores binarios pero el resultado final será el mismo, en nuestro caso utilizamos el sentido horario. Luego, convertimos este valor binario en un valor decimal y lo establecemos en el valor central de la matriz, que en realidad es un píxel de la imagen original. En este caso obtenemos el valor de 182. Por tanto, el código de LBP de ese pixel central es 182. Aquí hemos pasado un valor de gris que va desde 0 a 255 a un valor de imagen LBP que igual va de 0 a 255. Al final de este procedimiento (procedimiento LBP), tenemos una nueva imagen que representa mejor las características de la imagen original.

Una cuestión técnica va cuando se desea calcular el valor de los bordes de la imagen, se encuentran ciertos problemas ya que no tenemos los niveles de gris de sus vecinos, entonces el procedimiento para estos casos va a depender del método. En nuestro caso simplemente se ignoran los bordes en el sentido que no se calcula el LBP para los bordes. Los local Binary patterns se definieron en un principio para distinguir texturas. El LBP se define para imágenes en escala de grises es por eso que se transforma. Por lo que al conseguir una imagen LBP tiene un código que forman la imagen. Para describir las imágenes no se usa todo el array de códigos que conforman la imagen LBP, lo que se hace es un **histograma** normalizado de las imágenes.

Podemos observar en el histograma normalizado de la imagen LBP, donde se observa que hay varios pixeles con código alto y también bajo, pero no llegan al 20%. El histograma es un descriptor de 256 dimensiones. Haciendo uso del histograma se puede describir cómo será la imagen en LBP.

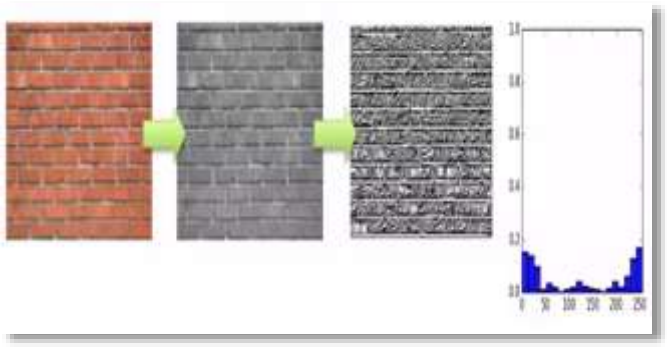

Figura 3. Transformación a imagen LBP e Histograma (**LBPH**)

En este otro ejemplo tenemos una imagen en escala de grises, cada histograma (de cada cuadrícula) contendrá solo 256 posiciones (0 - 255) que representan las ocurrencias de cada intensidad de píxel. Luego, debemos concatenar cada histograma para crear un histograma nuevo y más grande. Supongamos que tenemos rejillas de 8x8, tendremos 8x8x256 = 16.384 posiciones en el histograma final. El histograma final representa las características de la imagen original de la imagen.

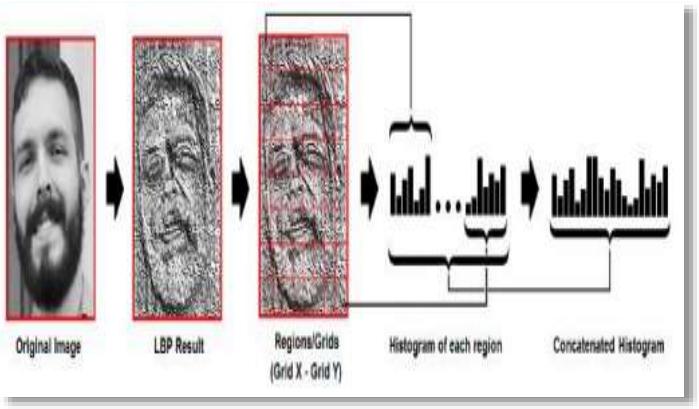

Figura 2. Creación de Histograma basado en imagen LBP (**LBPH**)

Realización del reconocimiento facial:
En este paso, el algoritmo ya está entrenado. Cada histograma creado se usa para representar cada imagen del conjunto de datos de entrenamiento. Entonces, dada una imagen de entrada, realizamos los pasos nuevamente para esta nueva imagen y creamos un histograma que representa la imagen. Entonces, para encontrar la imagen que coincida con la imagen de entrada, solo tenemos que comparar dos histogramas y devolver la imagen con el histograma más cercano.
Podemos usar varios enfoques para comparar los histogramas (calcular la distancia entre dos histogramas), por ejemplo: distancia euclídea, chi-cuadrado, valor absoluto, etc. En este ejemplo, podemos usar la distancia euclidiana (que es bastante conocida) basada en la siguiente formula:

$$D = \sqrt{\sum_{i=1}^{n} (hist1_i - hist2_i)}$$

Figura 4. Fórmula para la creación del LBP

Entonces la salida del algoritmo es la ID de la imagen con el histograma más cercano. El algoritmo también debe devolver la distancia calculada, que puede usarse como una medida de confianza".

III. DESARROLLO

Primero se crearán los scripts necesarios para entrenar nuestro reconocedor de rostros. Hay que mencionar que la aplicación se está diseñando en Python. Comenzaremos importando las librerías dentro de nuestro entorno, luego definimos donde estarán alojadas nuestras imágenes a analizar y el nombre de la carpeta que será el nombre del usuario dueño de los rostros. Si en caso no existe la carpeta, se creará.

Una vez que se crea la carpeta donde se alojaran los rostros, iniciaremos la cámara web o de videovigilancia para capturar las imágenes. Como estamos simulando nuestra aplicación, todo se está realizando en consola, luego se le implementará la

interfaz gráfica. El algoritmo que se utiliza para la detección de rostros es el de Viola-Jones utilizando Haar Cascade. Este algoritmo emplea la extracción de características utilizando imágenes positivas e imágenes negativas.

Este clasificador (Haar Cascade) lo obtenemos de OpenCv, ya nos brinda de manera gratuita en su repositorio. Dicho esto, el archivo clasificador lo guardamos en la misma carpeta donde tenemos nuestros scripts. Una vez que definimos la carpeta donde se almacenarán los rostros, definimos que tengan todo un mismo tamaño (150x150px). Hacemos un ciclo hasta que capturen 300 imágenes del rostro de la persona. Se debe tener en cuenta que los rostros almacenados sean del ambiente donde se encontrará el sistema ya que de esta manera se tendrá una mayor precisión al analizarlos.

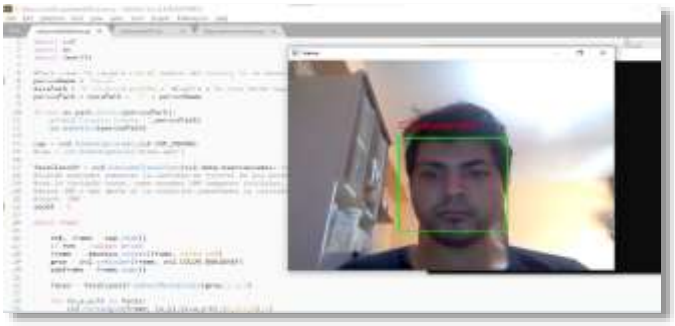

Figura 5. Capturando rostro

Luego de realizar las capturas de los rostros, ya podemos entrenar nuestro sistema para que pueda reconocer a los usuarios.

Primero definimos la ruta de la carpeta creada para que busque las imágenes. Luego utilizaremos una etiqueta para que el sistema pueda identificar las carpetas de las personas con sus rostros capturados.

Luego de ello almacenaremos las etiquetas con sus rostros que le corresponden a cada uno, añadiendo cada una de las imágenes en escala de grises (esto porque el clasificador trabaja con imágenes en escala de grises). Ahora para el entrenamiento utilizaremos el método LPBHFaces. Este método analizará imagen por imagen y píxel por pixel, mientras más imágenes se tenga mejor resultado se obtendrá.

Una vez entrenado, el método se creará y lo podemos guardar en nuestra ruta donde se encuentran todos nuestros scripts, esto para no volver a ejecutar el entrenador y tenerlo siempre al alcance.

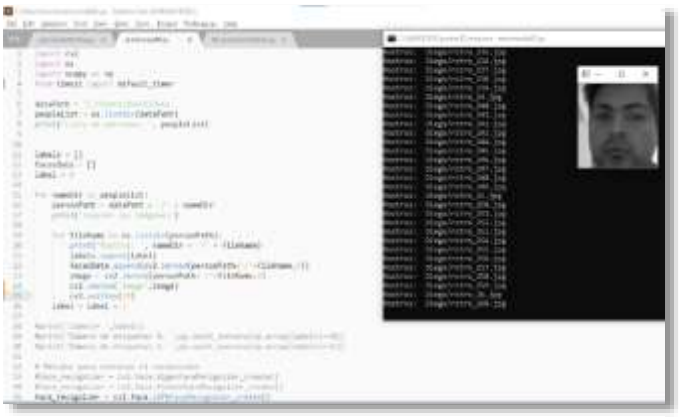

Figura 6. Entrenamiento de los rostros capturados

Ahora, para el reconocedor debemos especificar la ruta donde se encuentran las imágenes.

Luego se leerá el modelo que se ha almacenado cuando ejecutamos el entrenador. Después definimos porque medio se realizará la captura del video, si por una cámara web o una de videovigilancia. Luego de ello el procedimiento es muy parecido al del capturador. Se realiza un bucle que dibuja un rectángulo alrededor del rostro y se mostrará una etiqueta de la precisión que y/o el nombre del usuario registrado.

Se debe tomar en cuenta que debemos redimensionar el rostro que se obtiene a través de la cámara a 150x150px en escala de grises, ya que de esa manera se realizó en el capturador y entrenar al reconocedor de rostros. Se utilizará la función predict("imagenCapturada") para predecir la etiqueta y la confianza asociada para una imagen de entrada especificada en este caso se obtendrán los valores necesarios para verificar cual es la precisión del método utilizado.
Ahora se realizará la comparación de estos valores de precisión, el cual es importante para determinar si la persona identificada en la cámara es auténtica. Nuestro algoritmo LPBHFaces tiene un valor de precisión no mayor a 70, si en caso la imagen que es capturada fuese mayor, nos arrojará el mensaje de rostro desconocido por ende la alerta.

Se realizó una simulación del sistema y nos arrojó resultados positivos, donde se pudo identificar los rostros.

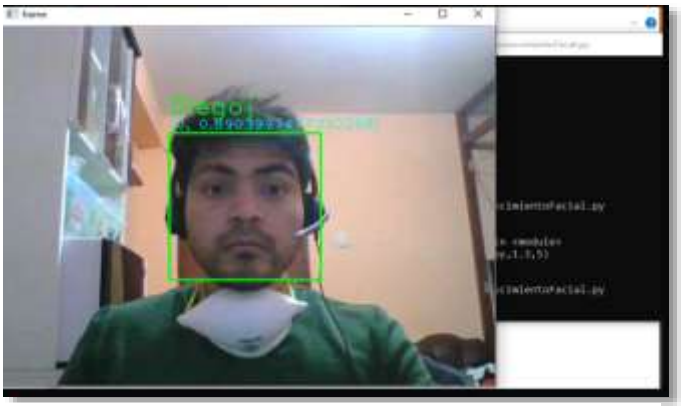

Figura 7. Simulación de reconocimiento facial

IV. RESULTADOS

A continuación, presentamos el análisis de la detección del rostro que nos brinda el reconocedor facial, utilizando el método establecido que es el algoritmo LBPHFaces.
Este análisis trata de evaluar los resultados obtenidos al detectar los rostros y que tan confiable es la aplicación al momento de reconocerlos.

Las imágenes consideradas para ser analizadas y posteriormente entrenar al sistema, forman parte de una base de datos construida para llevar a cabo nuestro objetivo, y está conformada por los rostros de varias personas que forman parte de los rostros de "Confianza" el rostro que no se encuentre dentro de la base de datos será reconocido como una persona "Desconocida". Dependiendo de la imagen tomada, se identificará al sujeto. A lo largo de esta investigación se pudo observar que los cambios del entorno influyen mucho al momento de detectar el rostro, si el rostro contiene lentes o accesorios también influyen al momento de la detección, en caso cuente con barba, bigote, corte de cabello también es un factor que se debe de tener en cuenta.

Al detectar estos escenarios, se decidió que el usuario pueda agregar más rostros a la base de datos del sujeto en cuestión, con el fin de aumentar la precisión y el nivel de confianza que arrojará el sistema.

Hay que mencionar que en un principio el algoritmo trabaja con solo 300 imágenes, luego pueden agregarse más imágenes si se identifica que el sujeto cambia un poco su rostro.

Iniciamos evaluando la Figura 8 que muestra un rostro con un nivel de confianza de 41.43, el nivel de confianza del algoritmo se establece para los valores menores del 70, esto significa que todos los rostros que detecte y que superen este valor, arrojará un mensaje de "Desconocido".

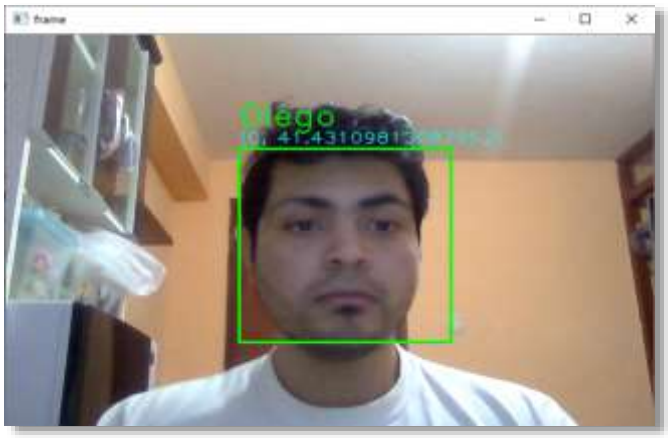

Figura 8. Detección de rostro entrenado con valores de confianza dentro de lo permitido.

En la Figura 9 se observa un rostro desconocido, es por ello por lo que el valor que arroja es superior al 70.

También se realizaron las pruebas midiendo la distancia en que la cámara detectaba el rostro del sujeto, por lo que, al situar al sujeto a una distancia mayor a 1 metro, el sistema no lograba reconocer eficientemente el rostro. Por lo que se procedió a ejecutar el capturador de rostros nuevamente, pero con el sujeto a una distancia mayor de la cámara, para que de esta manera pueda analizar mejor su rostro.

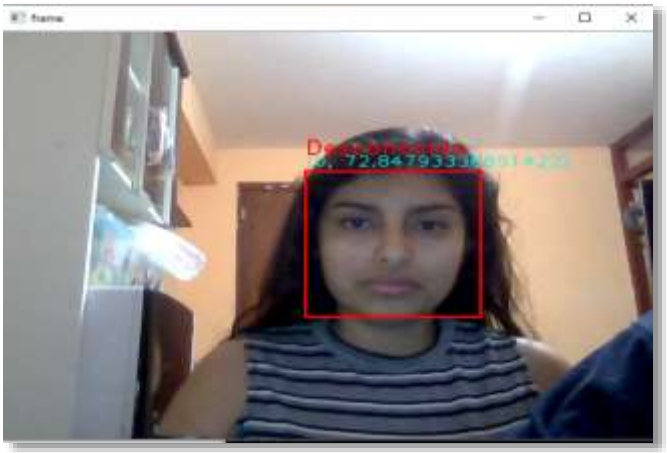

Figura 9. Detección de rostro sin entrenar con valores de confianza fuera de lo permitido

En la Figura 10 se observa como el sistema no logra detectar el rostro a una distancia mayor a un metro de la cámara a pesar de que el rostro ya ha sido entrenado, pero a una distancia corta.

Figura 10. Rostro sin entrenar a distancia

En la Figura 11 ya se ha capturado el rostro a una distancia mayor y se entrenó la aplicación para que detecte el rostro a una mayor distancia y se puede observar que el rostro ahora si es reconocible por el sistema. Al observar los niveles de confianza se identifica que se redujo considerablemente, al punto de arrojar valores similares de una imagen a corta distancia (52.17). Por esta razón se recomienda capturar los rostros a una distancia corta y también a una distancia un poco más larga, esto porque la cámara no debe estar al alcance de las personas y deberían ser identificadas a una distancia prudente.

Figura 11. Rostro con lentes entrenado a distancia

Se realizaron las pruebas con accesorios como son los lentes, ya que la aplicación no debe de cambiar la forma de vestir de las personas, por lo que si alguien usa lentes debería de reconocerlo sin problemas. Los resultados como se observan en la Figura 12 logran reconocer el rostro con los lentes puestos, sin embargo, se observa que el valor de confianza que nos devuelve el algoritmo aumenta en comparación a la detección con el rostro descubierto, esto es porque la similitud con las imágenes entrenadas no tiene puesto los lentes, por ello
la imagen captada por la cámara en tiempo real se aleja de las imágenes guardadas, los valores irán aumentando y se irá perdiendo el nivel de confianza.

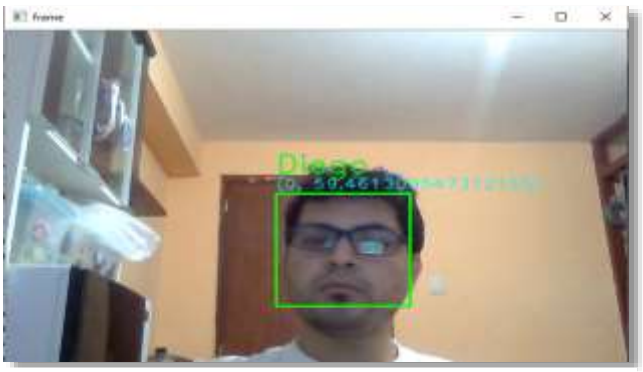

Figura 12. Rostro con lentes sin entrenar

Se probó también utilizando los lentes a una larga distancia y se observa que el reconocedor facial no logra identificar muy bien el rostro con los lentes puestos, si bien los valores no se alejan mucho, en la Figura 13 se observa que nos arroja el mensaje de "Desconocido" caso contrario a la Figura 4.5 que nos detecta el rostro sin problemas ya que no tenemos puestos los lentes. Esto indica que el usuario que lleve lentes también debería de tomarse las fotografías con y sin lentes, para que el entrenamiento del algoritmo sea mucho mejor y el nivel de confianza aumente.

Figura 13. Rostro con lentes a distancia

Luego de entrenar el rostro con lentes a distancia, se observa que el sistema logra detectar sin problemas el rostro, como se muestra en la Figura 14.

Figura 14. Rostro con lentes entrenado a distancia

Por último, se realizaron las pruebas con los dos rostros de lejos para identificar si el sistema logra reconocerlos al mismo tiempo, en la Figura 15 se observa que pudo identificarlos sin problema alguno.

Figura 15. Rostro con lentes entrenado a distancia

A continuación, se muestra un cuadro de las variaciones de los niveles de confianza del rostro normal, sin objetos y un rostro con lentes (rostro sin entrenar)

Tabla 1.Niveles de confianza según entrenamiento

Niveles de confianza		
	Diego	Thalia
Rostro cerca entrenado	41,43	45,07
Rostro cerca con lentes	59,46	58,38
Rostro lejos sin entrenar	72,23	69,84
Rostro lejos con lentes	72,64	70,21
Rostro lejos con rostro entrenado	52,17	43,35

Como se mencionó anteriormente, se puede observar una variación en los niveles de confianza. Ahora se mostrará un cuadro con los niveles de confianza del rostro con lentes y que ya ha sido entrenado por el sistema.

Tabla 2. Variación de los niveles de confianza entrenados

Niveles de confianza		
	Diego	Thalia
Rostro cerca entrenado	**41,43**	**45,07**
Rostro cerca con lentes	59,46	58,38
Rostro cerca con lentes entrenados	**47,83**	**44,13**
Rostro lejos sin entrenar	72,23	69,84
Rostro lejos con rostro entrenado	**52,17**	**43,35**
Rostro lejos con lentes	72,64	70,21
Rostro lejos con lentes entrenados	**54,03**	**43,35**

Como se observa, luego de que el sistema entrenara los rostros con lentes, el nivel de confianza es mucho mayor, por lo que se asemeja al rostro sin lentes.

Luego de realizar varias pruebas se identificó que el nivel de confianza que se debería de establecer para que el algoritmo reconozca los rostros es de 60, ya que los resultados arrojan un valor al mencionado. En un principio se había colocado establecido el valor de 70 pero con los resultados obtenidos se define que el valor óptimo sería 60.

Al cambiar el valor de confianza que debería tener el algoritmo, sería mucho más seguro al momento de identificar a las personas, ya que de esta manera se preverá los falsos positivos con los rostros.

He de mencionar también que no siempre los resultados son positivos, ya que a pesar de que el rostro haya sido entrenado por el reconocedor hay veces en las que nos arroja el mensaje de "Desconocido" ya que el modelo matemático se enfoca en el rostro de manera frontal y cuando se detecta el rostro de perfil la imagen se distorsiona y no lo reconoce al 100 por ciento.

Se realizaron las pruebas también colocando una imagen tomada desde un celular para verificar si esta la identificaba como verdadera y como se observa en Figura 4.14 logra identificarlo, pero el valor de confianza es mayor 60 y muchas veces también que el 70, lo que reafirma nuestra conclusión de colocar el valor de confianza por debajo del 60 para no obtener estos falsos positivos porque alguien con el rostro de la persona puede colocar su celular y podría burlar la seguridad.

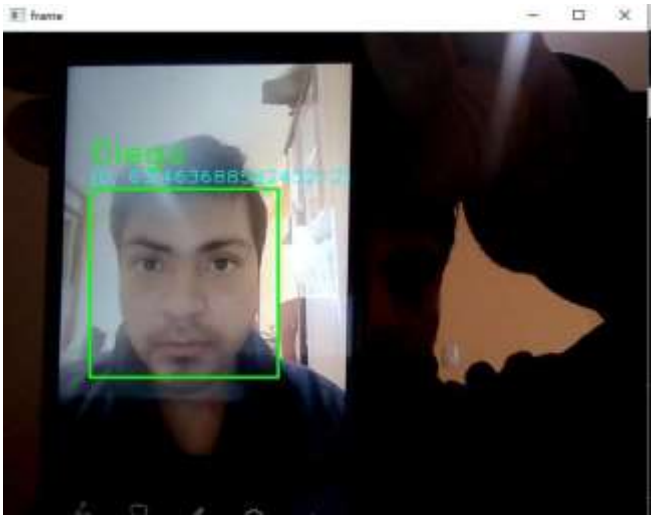

Figura 16. Imagen desde celular

También se realizaron pruebas con baja iluminación y el reconocedor detecta el rostro entrenado y nos devuelve un valor de confianza superior a 60. Se determinó que, si bien es bueno que el sistema detecte el rostro con un cambio de iluminación, el valor de confianza debería de ser superior a 50 pero menor a 60, lo que también permitiría que el sistema detecte como confiable el rostro de una persona registrada mostrando una imagen con buena calidad de este. Se tendría que evaluar el rango de los valores de confianza según los requisitos del

usuario final.

En la Figura 6 se observa el tiempo que tarda en entrenar el modelo para la identificación del rostro. Se logra observar que el método EigenFaces tarda mucho más en crear el modelo, seguido del FisherFaces y el que lo realiza en menor tiempo es el método LPBHFaces, dando como vencedor este último método.

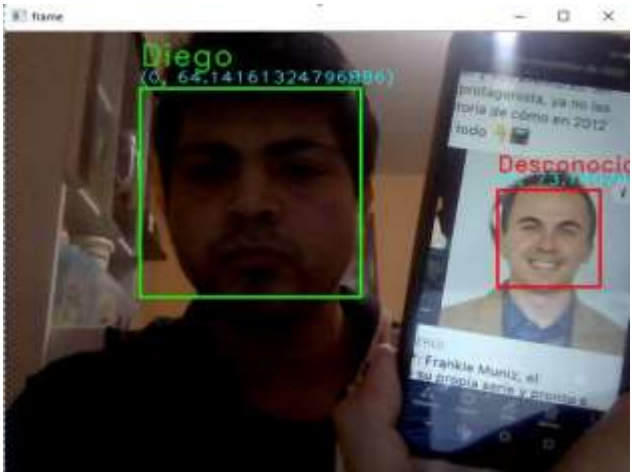

Figura 17. Rostro con baja iluminación

Figura 18. Tiempo de entrenamiento por modelo

Asimismo, en la Figura 7 se observa el tiempo que tarda el reconocer el rostro según el método escogido. Se logra observar que el método FisherFaces tarda mucho menos en reconocer los rostros, seguido del EigenFaces y por último elque tarda más es el LPBHFaces.

Figura 19. Comparación del tiempo de reconocimiento

V. DISCUSIÓN

Mediante el análisis de los distintos métodos de reconocimiento facial y comparándolos con distintas investigaciones realizas por diversos autores, se demostró que el algoritmo LPBHFaces prevalece en comparación de los otros dos (FisherFaces y EigenFaces). Esto se debe a la cantidad de imágenes utilizadas para su entrenamiento, ya que mientras haya una buena cantidad de imágenes para su análisis, el algoritmo responderá de manera mucho más positiva. Por otro lado, el tiempo de entrenamiento y ejecución son superiores. La tasa de aciertos es mucho mejor y se detectaron pocos falsos positivos en comparación al modelo FisherFaces y EigenFaces. Si bien Lopez A. & Torres F. (2018) y Esparza C. & Tarazona C. (2015) determinan que el modelo LPBHFaces sólo es mejor en cuanto a resultados, en este presente trabajo se ha demostrado que el modelo LPBHFaces es el claro vencedor en todo aspecto, por lo que se afirma que es el mejor método y el cual se debe de utilizar para la seguridad de las personas, ya que hay un mayor índice de confiabilidad.

VI. CONCLUSIONES

En el presente trabajo se comprobó, a lo largo de diferentes análisis realizados, que el algoritmo LPBHFaces es el método más recomendado para la seguridad de las personas en sus hogares; que, al momento de entrenar los rostros de las personas, el tiempo de ejecución y de entrenamiento es mucho menor a comparación de los métodos EigenFaces y FisherFaces. Mientras más personas entrenemos el tiempo para su ejecución aumentará y al tardar el 5.8% y el 11.3% del tiempo con respecto a los métodos EigenFaces y FisherFaces respectivamente, lo hace muy superior. También se concluyó que es el que mejor analiza los rostros y tiene menos falsos positivos en comparación a los otros dos métodos.

Con respecto al análisis múltiple de rostros en tiempo real y el monitoreo constante, se verificó que con los 3 métodos el sistema funciona, sin embargo, va a depender del hardware que se utilizará para que no haya problema al momento de su ejecución, ya que se necesita un equipo que contenga mínimo 12GB de Ram y un procesador Core i5 de décima generación, que es donde se realizaron las pruebas del sistema. El sistema soporta el monitoreo constante de los rostros.

Respondiendo a la pregunta de investigación, el sistema proporciona una mejor seguridad para el condominio san Antonio de Carabayllo, ya que el método LPBHFaces identifica mejor los rostros y para una mejor precisión, se debe de tomar la captura de las imágenes de cerca y de lejos para así evitar los falsos positivos.

El sistema puede albergar el rostro de varias personas, pero al aumentar la cantidad de personas de confianza, el tiempo de entrenamiento también aumentará, es por ello por lo que el LPBHFaces es la mejor opción, ya que su tiempo de ejecución y entrenamiento es mucho menor.

Al identificar a las personas que entran y salen del condominio y tener una buena precisión al analizar los rostros, se reduce el índice de robos al condominio, ya que, si se detecta a una persona que no esté registrada, se procederá a intervenirla de inmediato validar su acceso; de esta manera se reforzará la seguridad del condominio.

REFERENCIAS

[1] Bravo, C. J., Ramírez, P. E., & Arenas, J. (2018). Aceptación del reconocimiento facial como medida de vigilancia y seguridad: Un estudio empírico en Chile. Información tecnológica, 29(2), 115-122.

[2] Buchelly, F., Pastore, J., Passoni, I., & Ballarin, V. (2016). Identificación de rasgos faciales mediante técnicas de procesamiento de imágenes. 2016 IEEE Biennial Congress of Argentina (ARGENCON), Biennial Congress of Argentina (ARGENCON), 2016 IEEE, 1–5. Recuperado: https://doi.org/10.1109/ARGENCON.2016.7585362

[3] Caballero Julián, F. G., Vidal Reyes, M., López Sánchez, A., & Jerónimo Ríos, C. A. (2018). Reconocimiento Facial Por El Método De EigenFaces. Pistas Educativas, 39(127).

[4] Cardona López, A., & Pineda Torres, F. (2018). Reconocimiento de Rostros en Tiempo Real sobre Dispositivos Móviles de Bajo Costo. Lámpsakos, 20, 30–39. Recuperado: https://doi.org/10.21501/21454086.2938

[5] Cortes Martínez, F. R., Herrera Candelaria, Á. D., Martínez Peláez, R., Saavedra Benítez, Y. I., & Velarde Alvarado, P. (2018). Identificación Digital E Infraestructura Para Incrementar La Seguridad en El Ciberespacio. Pistas Educativas, 38(120).

[6] Franco, C. E., Ospina, C. T., Cuevas, E. S., & Capacho, D. V. (2017). Reconocimiento Facial basado en eigenfaces, lbhp y fisherfaces en la beagleboard-XM. Recuperado: https://doi.org/10.24054/16927257.v26.n26.2015.2387

[7] Gebhart A. (2019) Reconocimiento Facial: Apple, Amazon, Google y la carrera por captar tu cara. La tecnología de reconocimiento facial es a la vez innovadora y preocupante. Referencia de: https://www.cnet.com/es/noticias/reconocimiento-facial-apple-amazon-google-ai/

[8] Hernández-Durán, M., & Plasencia-Calaña, Y. (2016). Aprendizaje de métrica para el reconocimiento de rostros a partir de imágenes de baja resolución / Metric Learning for Low-Resolution Face Recognition. Revista Cubana de Ciencias Informáticas, 124–133.

[9] Introna, L. D., & Wood, D. (2002). Picturing Algorithmic Surveillance: The Politics of Facial Recognition Systems. Barcelona: Surveillance & Society. Surveillance & Society, 2(2/3), 177-198.

[10] Lorente Giménez, L. (1998). Representación de caras mediante eigenfaces. Buran, (11), 13-20

[11] Martínez-Díaz, Y., Hernández, N., & Méndez-Vázquez, H. (2016). Detectores espaciotemporales para la detección de rostros en video / Spatio-temporal detectors for face detection in video. Revista Cubana de Ciencias Informáticas, 205–214.

[12] Niola Quito, C. X., & Sanango Zhinin, W. A. (2019). Desarrollo de un software de seguridad para detección y Reconocimiento Facial basado en los algoritmos de Viola-Jones y PCA Eigenface (Tesis de bachiller). Departamento de ingeniería electrónica Universidad politécnica salesiana sede cuenca de Ecuador, 2019

[13] Cardenas, F. D. J. N., Mariano, V. T. T., Hernandez, J. D. C., & Hernández, E. A. (2017). Reconocimiento facial mediante redes neuronales “hopfield”, “backpropagation” y algoritmo pca: un enfoque comparativo. Ciencia Huasteca Boletín Científico de la Escuela Superior de Huejutla, 5(9). Recuperado: https://doi.org/10.29057/esh.v5i9.2225

[14] Paul Viola, & Michael Jones. (2001). Rapid Object Detection Using a Boosted Cascade of Simple Features. In Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001 (Vol. 1, pp. I-I). IEEE.

[15] Russo, C., Sarobe, M., Alonso, N., Moretti, N., Beloso, J. P., Ahmad, T., González González, C. S., Serafino, S., Collazos, C. A., & Decoud, C. (2019). Informática y tecnologías emergentes. In XIX Workshop de Investigadores en Ciencias de la Computación (WICC 2017, ITBA, Buenos Aires).

[16] Sierra Zapata, M. (2015). Estudio comparativo de modelos de identificación facial basados en correlación (Tesis de grado). Departamento Ingeniería de Sistemas y Automática. Escuela Técnica Superior de Ingeniería de la Universidad de Sevilla, 2015.

[17] Tesillo Gómez, C. M. (2016). Análisis comparativo de los algoritmos Fisherfaces y LBPH para el Reconocimiento Facial en diferentes condiciones de iluminación y pose, Tacna – 2015.

[18] Viorica, P., Capitan, F., (2016) Aplicación para detección y Reconocimiento Facial en interiores. Trabajo final de grado Ingeniería Electrónica, Robótica y Mecatrónica. Escuela Técnica Superior de Ingeniería. Universidad de Sevilla, 2016.

[19] Xie, Z., Liu, G., & Fang, Z. (2011) Face Recognition Based on Combination of Human Perception and Local Binary Pattern. Intelligent Science and Intelligent Data Engineering: Second Sino-Foreign-Interchange Workshop, IScIDE 2011, Xi’an, China, October 23-25, 2011, Revised Selected Papers, 365. Recuperado: https://doi.org/10.1007/978-3-642-31919-8_47

[20] Zambrano Zambrano, G. V. (2014). Sistema De Vigilancia Mediante Cámaras Ip Con Un Software De Detección De Rostro. Revista Espamciencia, 5(1), 47–51.