

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO
NACIONAL
ROBÓTICA Y MANUFACTURA AVANZADA



Cinvestav
Unidad Saltillo

Visión por Computadora

PROYECTO

Hand Gesture Detection and Recognition

Integrantes:
Hilario Acuapan Gabriela
Pineda Gómez Luis Alberto

Contents

1	Introducción	3
2	Fundamentos teóricos	3
2.1	Análisis de componentes principales	3
3	Desarrollo de la metodología	6
3.1	Adquisición de datos	7
3.2	Preprocesamiento de las imágenes de entrada para la formación de la base de datos	7
3.3	Training set	9
3.4	Eigenhands con PCA	10
3.5	Clasificación	12
4	Resultados	13
4.1	K-ésima clase	13
4.2	Nueva imagen de entrada	14
5	Conclusiones	18

1 Introducción

El objetivo principal de este trabajo es detectar y reconocer los gestos de mano en 2D (en imágenes) a partir de su forma, comparándolos con gestos existentes en una base de datos. Para cumplir este objetivo se utiliza un algoritmo para la detección y reconocimiento de gestos basado en los componentes principales (PCA). El algoritmo a implementar se describe en el artículo de Turk y Pentland [3] (*Face Recognition Using Eigenfaces*), pero en lugar de aplicarlo a la detección de rostros humanos, se utilizará para la detección de gestos de mano.

El algoritmo de *Eigenfaces*, fue presentado por primera vez por Sirovich y Kirby en su artículo de 1987 [2], y posteriormente formalizado por Turk y Pentland en su artículo de 1991 [3]. El procedimiento general indica que cada cara se almacena en un vector de dimensión N^2 . Y el Análisis de los Componentes Principales (PCA) se utiliza para encontrar un subespacio de dimensión M cuyos vectores de la base corresponden a las direcciones de máxima varianza en el espacio original de la imagen (N^2). Este nuevo subespacio es normalmente de dimensión más baja ($M \ll N^2$).

EL enfoque del algoritmo es descomponer las imágenes de los rostros en un conjunto más pequeño de rasgos característicos llamados "eigenfaces", que pueden considerarse como los componentes principales de la base de datos original. Para la parte del reconocimiento se proyecta una nueva imagen en el subespacio generado por las eigenfaces (llamado "face space"), para luego clasificar el rostro comparando su posición en el *face space* de las personas conocidas (o clases conocidas).

El trabajo aquí presentado aplica este mismo procedimiento, pero en lugar de reconocer rostros, se utilizará para reconocer y clasificar gestos de la mano. Estos gestos son estáticos, es decir, son posiciones bien definidas de la mano en un instante de tiempo específico. El reconocimiento de gestos de la mano no es un problema trivial, hay una gran variedad de modelos y algoritmos dedicados a intentar resolver este problema, sin embargo, la finalidad de implementar un algoritmo como el de *eigenfaces* es conocer las bases de un problema de reconocimiento de patrones.

2 Fundamentos teóricos

2.1 Análisis de componentes principales

El objetivo de implementar un análisis por componentes principales es realizar una **reducción de dimensionalidad** ó del número de variables en una base datos. Sin embargo, el realizar este análisis viene acompañado de un precio a pagar y esto es la relación **precisión - simplicidad**. Por un lado, al reducir la dimensión de nuestra base de datos, el realizar cómputos con ella será mucho más rápido y eficiente, además de que la interpretación de los datos es más sencilla a expensas de perder precisión y exactitud de la base original.

El análisis por componentes principales puede ser implementado en 5 pasos:

Paso 1.- Estandarización

En este primer paso realizamos una estandarización de los datos, esto con el objetivo de evitar que valores grandes, por ejemplo, que se encuentren en un rango de (100 a 1000) no dominen entre valores menores, por ejemplo (0 a 10). Esto es para evitar que nuestra base de datos presente lo que conocemos como **sesgo o bias**.

La estandarización de los datos se logra aplicando la siguiente expresión 1:

$$Z = \frac{\text{valor} - \text{media}}{\text{desviacion estandar}} \quad (1)$$

Paso 2.- Obtención de la matriz de covarianza

El propósito por el cual se calcula la matriz de covarianza es para obtener la relación existente entre las variables de la base de datos. Se busca encontrar cómo es que varían entre ellas. En algunas ocasiones, las variables se encontrarán relacionadas de tal forma, que tendrán información **redundante**, esto es, que varias variables contengan información similar. Este tipo de relaciones son las que se encuentran en la matriz de covarianza.

La matriz de covarianza es una matriz **simétrica** de dimensiones $p \times p$. Por ejemplo, supongamos que tenemos una matriz de covarianza de 3×3 :

$$\begin{bmatrix} Cov(x, x) & Cov(x, y) & Cov(x, z) \\ Cov(y, x) & Cov(y, y) & Cov(y, z) \\ Cov(z, x) & Cov(z, y) & Cov(z, z) \end{bmatrix}$$

Donde los elementos de la diagonal de la matriz $Cov(x, x) = Var(x)$ y también es importante mencionar que $Cov(x, y) = Cov(y, x)$.

La información obtenida por la matriz de covarianza viene dada principalmente por el **signo**. Esto es:

- Si es **Positivo**: Las dos variables incrementan o decrementan de la misma forma (Correlacionadas).
- Si es **Negativo**: Una variable incrementa cuando la otra decrementa (Inversamente correlacionadas).

Paso 3.- Obtención de los eigenvalores y eigenvectores de la matriz de covarianza

Esto se realiza con el objetivo de obtener los **componentes principales** de los datos. En los componentes principales, se encuentra contenida la información principal o más significativa.

Por ejemplo, en la figura 1, se muestra una gráfica de la varianza dado el número de componentes principales. En este caso, en el componente principal de mayor importancia se encuentra codificada el 40% de la covarianza. Esto es que dicho componente contiene información **única**.

Con este procedimiento, reducimos la dimensionalidad de la base de datos original sin tener un impacto sustancial en la pérdida de información. La parte sustancial del PCA, es que los componentes principales **caren** de interpretación. Esto quiere decir que no se le puede atribuir cierto atributo o explicación a un componente principal.

Debido a que existe tantos componentes principales como número de variables en la base de datos, el PCA se construye de tal forma que el componente primario contenga la mayor cantidad de varianza posible de la base de datos. El componente secundario contendrá la mayor cantidad de varianza posible, pero que no se encuentra en la primer componente y así sucesivamente con el resto de componentes.

Los **eigenvectores** juegan el papel de indicar los ejes en **dónde** se encuentra la mayor cantidad de varianza, esto es, la mayor cantidad de información de la base de datos. Mientras que los **eigenvalores**, dan la **cantidad** de varianza que contiene cada componente principal.

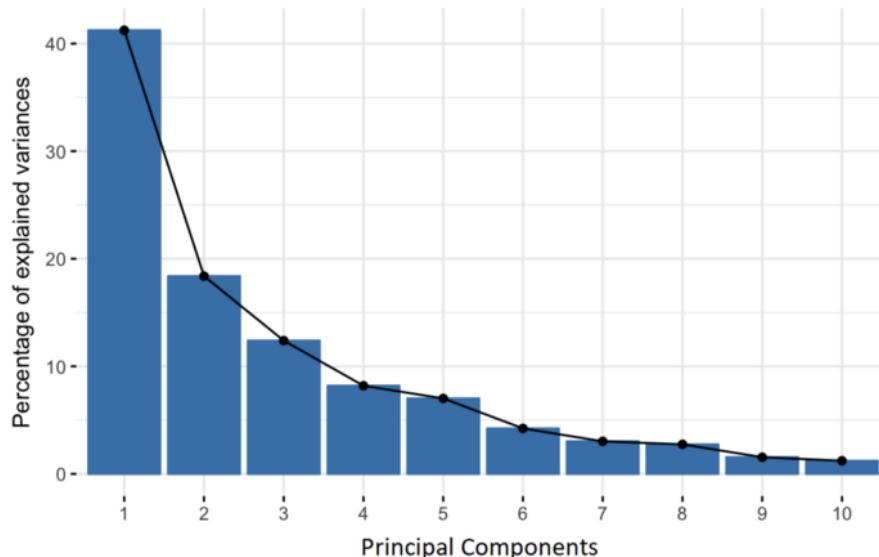


Figure 1: Comparación del Porcentaje de la varianza vs el número de componentes principales

Paso 4.- Construcción del Vector característico

En este paso, es donde se toma la decisión de conservar o descartar aquellos **eigenvectores** que no aporten información relevante de la base de datos. Con los vectores que deseemos conservar, se construirá una nueva matriz, conformada por los vectores columna de los eigenvectores más representativos.

Paso 5.- Reconformar la base de datos a lo largo de los ejes de los componentes principales

Para este último paso, tenemos que **reorientar** los datos a lo largo de los nuevos eigenvectores en donde se encuentra concentrada la mayor cantidad de información. Esto se obtiene de la siguiente forma:

$$\text{Final data set} = \text{Matriz caracteristica}^T * \text{Data set estandarizado}$$

3 Desarrollo de la metodología

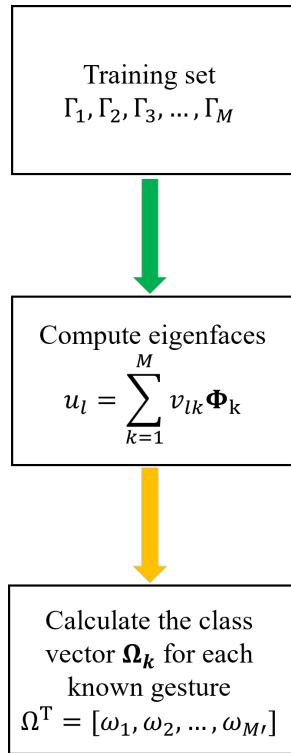


Figure 2: Operaciones de inicialización.

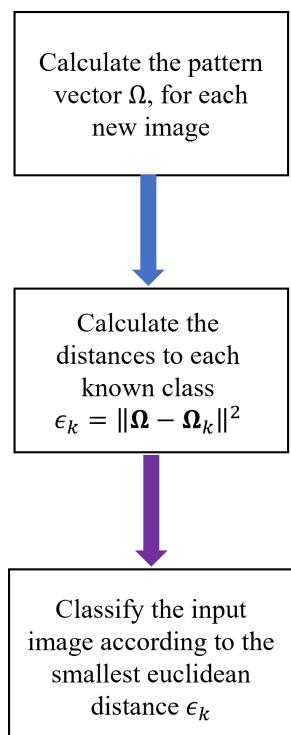


Figure 3: Operaciones para clasificación.

3.1 Adquisición de datos

La base de datos utilizada para este proyecto se creó a partir de la captura de imágenes en infrarrojos, utilizando el Sensor mostrado en la figura 4.



Figure 4: Structure sensor 3D [1].

3.2 Preprocesamiento de las imágenes de entrada para la formación de la base de datos

Antes de poder ingresar las imágenes a nuestro sistema, es necesario el "estandarizarlas". Para ello, se hace uso del la figura 5, en la cual se muestra de forma sencilla las etapas que conforman el preprocesamiento. El propósito por el cual se estandarizan las imágenes es para trabajar con frames cuadrados, esto tiene ventajas al momento de realizar operaciones y simplificar considerablemente el código.

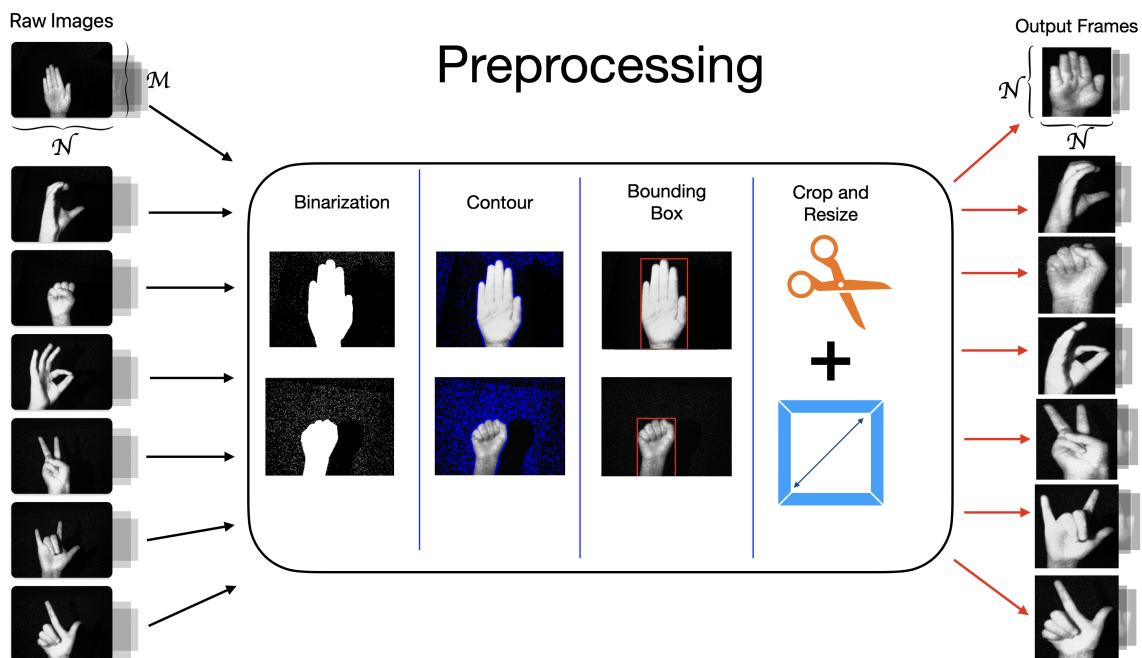


Figure 5: Preprocesamiento de las imágenes para conformar la base de datos del sistema.

Comenzamos con un conjunto de imágenes sin procesar, las cuales denotamos como **Raw Images**. En dicho conjunto, las imágenes tienen dimensiones $M \times N$. Como primer paso, se realiza una **binarización**, esto con el objetivo de encontrar los **contornos de la imagen**. Posteriormente, ya con los contornos detectados, procedemos a delimitar el contorno más importante mediante un **bounding box**. Para finalmente recortar la

imagen que se encuentra contenida dentro del bounding box y realizar un **escalamiento**. Esto con el propósito de obtener imágenes cuadradas y enfocarnos únicamente en el gesto.

Paso 1.- Binarización

La operación de binarización juega un papel fundamental en el preprocesamiento, debido a que se tiene que elegir un valor de umbral o **threshold** para permitir conservar la mayor cantidad de información de cada frame pero filtrando el mayor ruido posible. En la figura 6, se muestra la binarización de un frame. Es importante mencionar que el valor del threshold o umbral fue elegido de manera experimental.

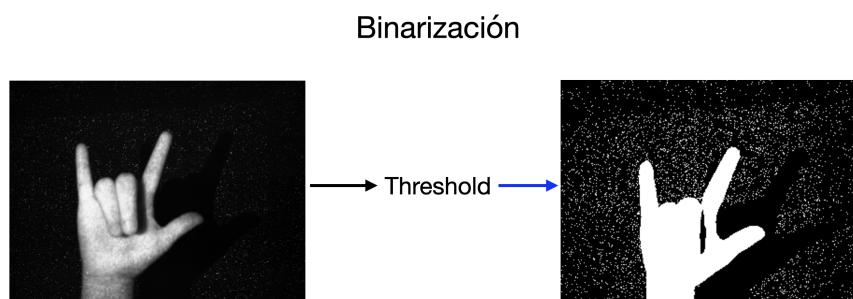


Figure 6: Binarización de las imágenes de entrada

Paso 2.- Contornos

En la siguiente etapa, encontramos los contornos tomando como base la imagen binarizada. Es importante mencionar que el contorno más relevante será aquel que contenga la mayor cantidad de elementos. En la figura 7, se muestran los contornos de la imagen binarizada.

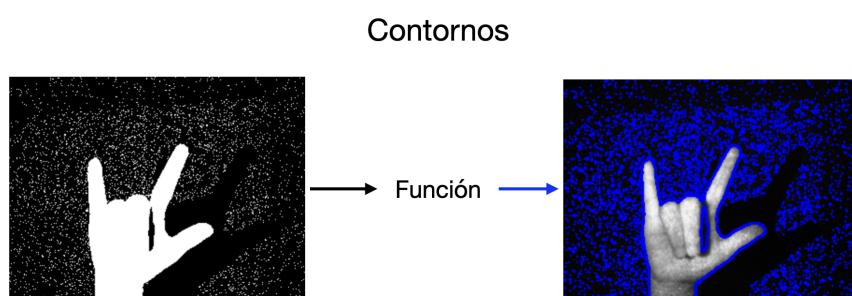


Figure 7: Contornos de la imagen binarizada

Paso 3.- Bounding Box

En la penúltima etapa, se implementa una función para generar un bounding box el cual contiene la información del contorno más relevante. Esto se muestra en la figura 8.

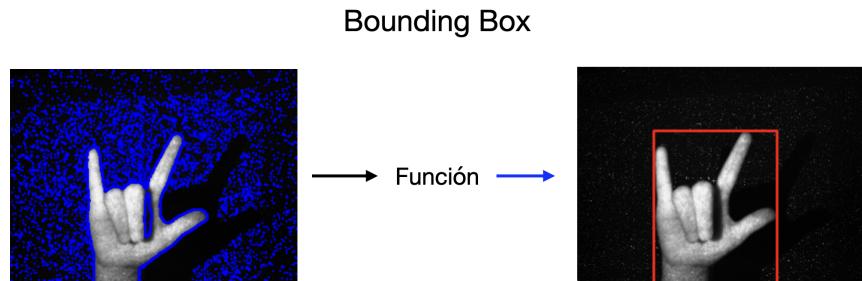


Figure 8: Contornos de la imagen binarizada

3.3 Training set

Denotando al conjunto de entrenamiento como:

$$\Gamma_1, \Gamma_2, \dots, \Gamma_M \quad (2)$$

Se utilizaron $M = 70$ imágenes para el conjunto de entrenamiento, donde se incluyeron los 7 gestos de mostrados en la figura 9.



Figure 9: Gestos para el set de entrenamiento.

Cada imagen pasa por el preprocessamiento mostrado en la sección 3.2, así, obtenemos imágenes de dimensiones cuadradas. Antes de seguir con el proceso, cada imagen debe ser normalizada dividiendo el valor de sus píxeles en 255, para poder trabajar con ella.

A continuación, cada imagen de $N \times N$ píxeles se convierte en un vector \mathbf{F} de dimensión N^2

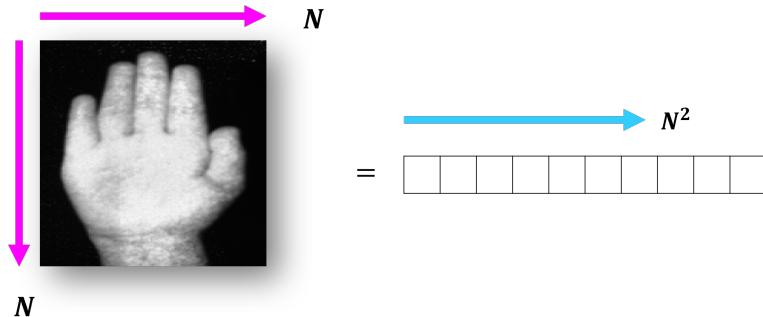


Figure 10: Vector imagen.

Estos vectores forman el conjunto de entrenamiento (2),

$$\begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \vdots \\ \Gamma_M \end{bmatrix} = \left(\begin{array}{ccccccc} \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} \\ \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} \\ \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} \\ \vdots & & & & & & \\ \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} & \boxed{} \end{array} \right)_{M \times N^2}$$

Figure 11: Conjunto de entrenamiento.

3.4 Eigenhands con PCA

Teniendo los vectores del conjunto (2), se obtiene la mano promedio (figura 12) mediante la siguiente operación

$$\Psi = \frac{1}{M} \sum_{n=1}^M \Gamma_n \quad (3)$$

Posteriormente se calcula la desviación Φ que tiene cada Γ con respecto al promedio Ψ

$$\Phi = \Gamma_i - \Psi \quad (4)$$

Este conjunto de vectores Φ se somete a un análisis de componentes principales (PCA), que busca un conjunto de M vectores ortonormales, u_n , que describen mejor la distribución de los datos.

Para esto definimos una matriz A que contiene a los vectores de las desviaciones Φ

$$A = [\Phi_1, \Phi_2, \dots, \Phi_M] \quad (5)$$

Para construir una matriz L de dimensiones $M \times M$

$$L = A^T A \quad (6)$$

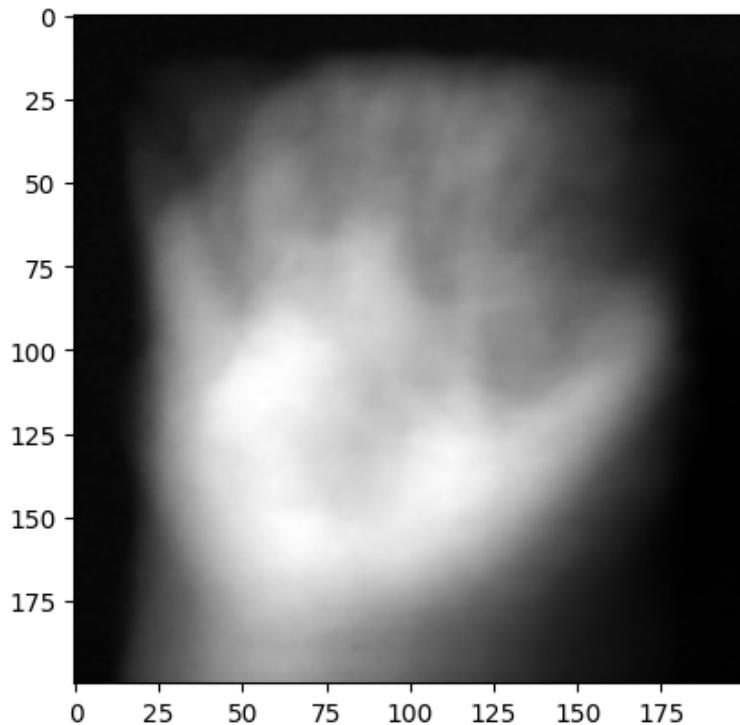


Figure 12: Mano promedio.

Y encontrar los eigenvalores μ_i y eigenvectores v_i de la matriz L, tal que

$$Av_i = \mu_i v_i \quad (7)$$

Se deben encontrar los M' eigenvectores v_i asociados a los eigenvalores μ_i más representativos. Para esto obtenemos la suma acumulada y observamos que con 30 eigenvectores reconstruimos hasta el 93% de la información (figura 13)

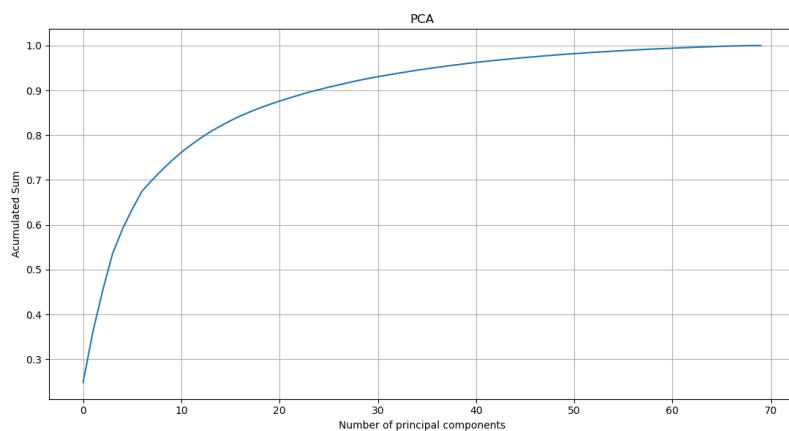


Figure 13: Suma acumulada de los componentes principales de la matriz L.

Estos vectores determinan combinaciones lineales de las imágenes del conjunto de entrenamiento M , y se utilizan para formar a las eigenhands u_i .

$$u_i = \sum_{k=1}^M v_{lk} \Phi_k, \quad l = 1, \dots, M \quad (8)$$

En la figura 16 se muestran 10 de las 30 eigenhands principales derivadas de las imágenes de entrenamiento.

Primeras 10 eigenhands obtenidas

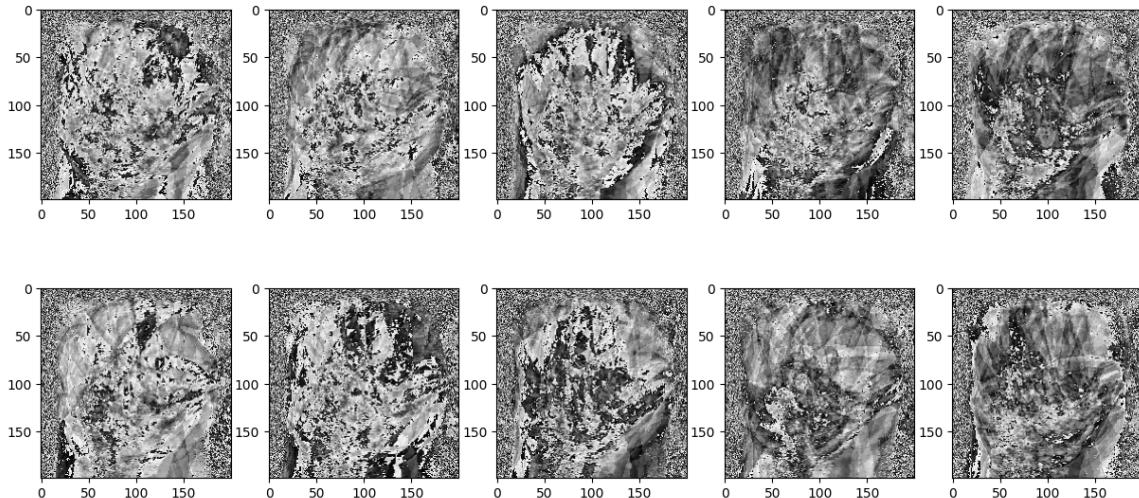


Figure 14: Primeras 10 eigenhands más representativas.

3.5 Clasificación

Las imágenes de eigenhands calculadas a partir de los vectores propios de L abarcan un conjunto básico (de dimensión $M' \times M'$) con el que describir el conjunto de imágenes inicial.

Para realizar la clasificación es necesario encontrar los patrones Ω_K que describen a la k -ésima clase de gestos.

$$\Omega^T = [\omega_1, \omega_2, \dots, \omega_{M'}] \quad (9)$$

Los pesos que forman al vector Ω describen la contribución de cada eigenfaces en la representación de la imagen de la cara de entrada, y se calculan mediante la siguiente operación

$$\omega_k = u_k^T (\Gamma - \Psi) \quad (10)$$

Las clases de los gestos se calculan promediando los patrones de pesos Ω de un pequeño número de imágenes de cada gesto.

Para determinar a qué clase pertenece una nueva imagen de entrada se calcula su patrón de pesos Ω , para luego utilizar la distancia euclídea

$$\epsilon_k = \|\Omega - \Omega_k\|^2 \quad (11)$$

La imagen de entrada se clasifica como perteneciente a la clase que tenga una menor distancia euclídea ϵ_k

4 Resultados

De la figura 13 se observa que con 30 componentes principales se puede reconstruir hasta el 93% de la información. Entonces nos quedaremos solamente con $M' = 30$ eigenvectores, para formar un subespacio de dimensión $M' \times N^2$ al que llamaremos "hand space", cuyos vectores de la base formaran a nuestras "eigenhands" (mostradas en la figura 16).

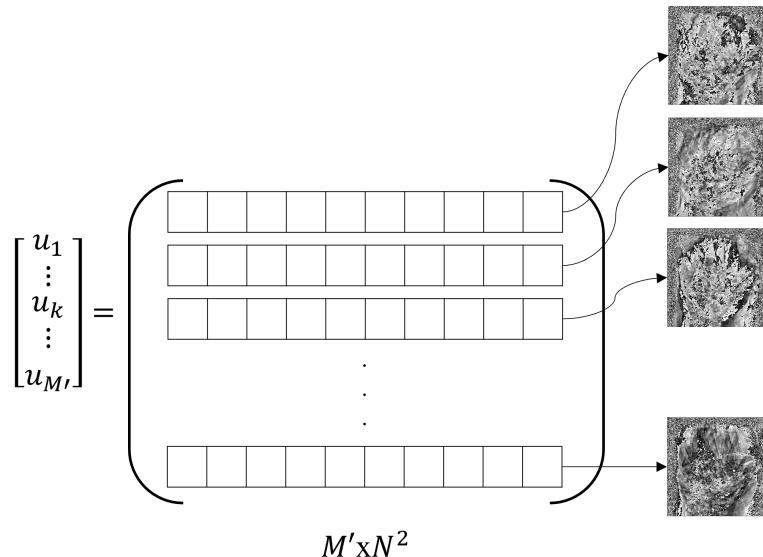


Figure 15: Hand space.

Para reconocer una nueva imagen se proyecta sobre este *hand space*, para luego clasificarla dentro las clases o gestos conocidos.

4.1 K-ésima clase

Para obtener las clases, se necesitan encontrar los vectores de pesos (9), de caja conjunto de imágenes pertenecientes a un solo gesto, es decir

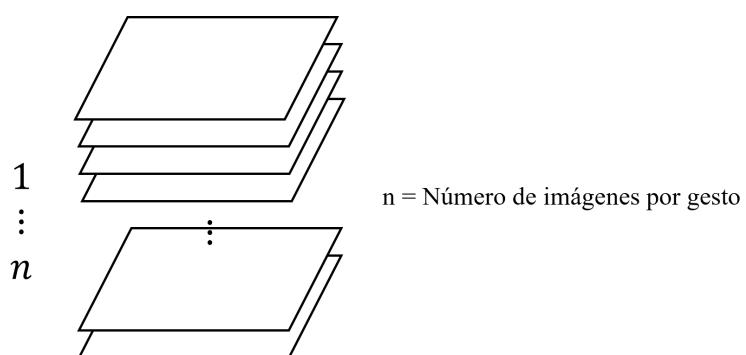


Figure 16: Imágenes por gesto usadas para el training set.

$$\begin{aligned}
 \Omega_1 &= [\omega_1, \omega_2, \dots, \omega_{M'}] \\
 \Omega_2 &= [\omega_1, \omega_2, \dots, \omega_{M'}] \\
 &\vdots \\
 \Omega_n &= [\omega_1, \omega_2, \dots, \omega_{M'}]
 \end{aligned} \tag{12}$$

Promediando los n vectores, se obtiene así el patrón de pesos característico de cada clase

$$\Omega_k = \frac{\Omega_1 + \Omega_2 + \dots + \Omega_n}{n} \tag{13}$$

4.2 Nueva imagen de entrada

Para cada imagen de entrada a clasificar se obtiene primero su patrón de pesos

$$\Omega = [\omega_1, \omega_2, \dots, \omega_{M'}] \tag{14}$$

Posterior a eso, se obtiene la distancia euclíadiana (ecuación 11) entre el vector (14) y el Ω_k de cada clase. Para obtener la minima distancia euclíadiana, esto nos indicará a qué clase (gesto) pertenece la imagen de entrada.

EXPERIMENTOS:

El algoritmo se probó con varias imágenes de entrada (capturadas también con el sensor IR), a continuación se presenta solo algunos ejemplos.

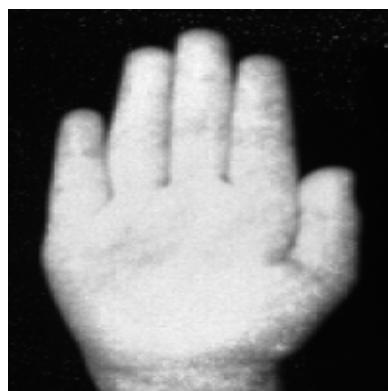


Figure 17: Imagen de entrada 1 (Gesto: *Palm*).

Gesto	ϵ_k
<i>Palm</i>	1,765.23
<i>C</i>	8,965.39
<i>Fist</i>	3,463.38
<i>Ok</i>	8,155.25
<i>Peace</i>	6,431.8
<i>I love you</i>	7,620.04
<i>L</i>	7,851.43

Table 1: Comparación de las distancias ϵ_k obtenidas para cada clase (Para la imagen 17).

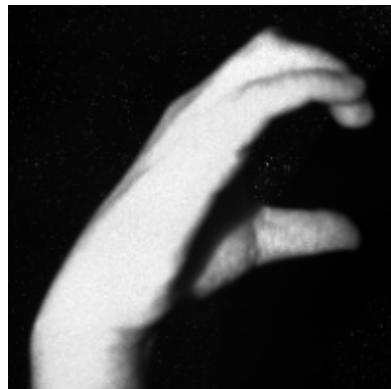


Figure 18: Imagen de entrada 2 (Gesto: *C*).

Gesto	ϵ_k
Palm	8,857.44
<i>C</i>	3,916.78
Fist	8,732.62
Ok	5,174.77
Peace	5,918.08
I love you	4,644.84
L	4,988.04

Table 2: Comparación de las distancias ϵ_k obtenidas para cada clase (Para la imagen 18).



Figure 19: Imagen de entrada 3 (Gesto: *Fist*).

Gesto	ϵ_k
Palm	3,005.77
<i>C</i>	8,465.81
<i>Fist</i>	3,018.81
Ok	7,784.55
Peace	6,935.6
I love you	7,843.9
L	8,295.76

Table 3: Comparación de las distancias ϵ_k obtenidas para cada clase (Para la imagen 19).



Figure 20: Imagen de entrada 4 (Gesto: *Ok*).

Gesto	ϵ_k
Palm	4,931.07
C	3,599.23
Fist	5,952.86
Ok	2,650.07
Peace	5,106.42
I love you	3,643.67
L	3,966.60

Table 4: Comparación de las distancias ϵ_k obtenidas para cada clase (Para la imagen 20).



Figure 21: Imagen de entrada 5 (Gesto: *Peace*).

Gesto	ϵ_k
Palm	4,455.62
C	7,809.5
Fist	4,914.63
Ok	7,040.54
Peace	3,081.97
I love you	5,660.82
L	5,739.24

Table 5: Comparación de las distancias ϵ_k obtenidas para cada clase (Para la imagen 21).



Figure 22: Imagen de entrada 6 (Gesto: *I Love you*).

Gesto	ϵ_k
Palm	5,658.64
C	4,849.28
Fist	6,366.77
Ok	4,161.36
Peace	4,155.54
<i>I love you</i>	1,512.36
L	2,969.92

Table 6: Comparación de las distancias ϵ_k obtenidas para cada clase (Para la imagen 22).



Figure 23: Imagen de entrada 7 (Gesto: *L*).

Gesto	ϵ_k
Palm	5,072.34
C	4,386.48
Fist	6,338.76
Ok	3,252.06
Peace	4,155.84
<i>I love you</i>	2,661.54
<i>L</i>	1,876.78

Table 7: Comparación de las distancias ϵ_k obtenidas para cada clase (Para la imagen 23).

5 Conclusiones

Al comienzo de este proyecto se estuvo utilizando una base de datos de imágenes alojada en la plataforma de Kaggle, sin embargo los resultados que fueron obtenidos en su momento no fueron prometedores, ya que las imágenes presentaban demasiada variación entre ellas, y el algoritmo no fue lo suficientemente eficiente ni robusto al momento de realizar las clasificaciones.

Siguiendo con lo anterior, se considera importante mencionar que el algoritmo propuesto por [3] está muy limitado para los fines de este proyecto. Ya que al ser los gestos de la mano muy parecidos entre sí, le resulta complicado encontrar aquellas características que resultan más relevantes de cada gesto. Especialmente en los gestos de la palma (*Palm*) y el puño (*Fist*) había mucha confusión, como puede observarse en la tabla 3 la imagen de entrada correspondía a un puño, sin embargo este lo clasificó como una palma.

Otro factor que probablemente esté influyendo en una clasificación ineficiente es el procesamiento previo de las imágenes, especialmente la etapa de reescalamiento. Debido a que al momento de recortar y reescalar las imágenes, el gesto se distorsiona perdiendo las dimensiones originales. Siguiendo con el caso de la palma y el puño, la palma es una figura más alargada verticalmente, pero al momento de reescalarla esta se achata y puede confundirse en dimensión con el puño. Ya que el método se basa más en la detección de patrones de la forma general del contorno en lugar de la posición de los dedos, es de esperarse este comportamiento.

En resumen, se concluye que el método de *Eigenfaces* tuvo buenos resultados clasificando correctamente la mayoría de las imágenes de entrada. El algoritmo no requiere una alta capacidad de procesamiento, por lo que puede ser implementado para la detección en tiempo real, tomando en cuenta las limitaciones que se mencionaron anteriormente. Sin embargo, para el trabajo a futuro se debe considerar hacer pruebas con una base de datos más amplia y con mayor variación, para determinar si esto mejora el reconocimiento o los resultados son similares.

References

- [1] Structure - the world's leading healthcare 3d scanning platform. Consultado el 10 de abril de 2023. [Online]. Available: <https://structure.io/>
- [2] L. Sirovich and M. Kirby, "Low-dimensional procedure for the characterization of human faces," *Josa a*, vol. 4, no. 3, pp. 519–524, 1987.
- [3] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.