



Universidad Veracruzana

Maestría en Inteligencia Artificial

Visión por Computadora

**Tarea 11. Obtención de un modelo estadístico de
forma con PCA sobre imágenes registradas..**

Ángel García Báez y Ulises Jiménez Guerrero

Profesor: Dr. Héctor Acosta Mesa

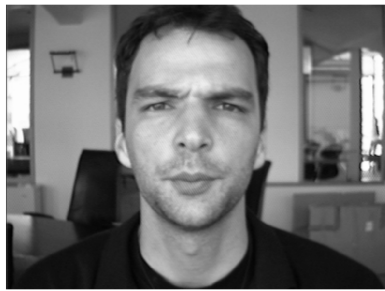
May 27, 2025

Contents

1	Objetivo de la práctica	2
2	Metodología	3
2.1	Modelo estadístico de forma usando PCA	3
3	Resultados	5
3.1	Selección de las componentes del modelo	5
3.2	Distribución de los parámetros para las 12 componentes del modelo	6
3.3	Rostro generado para la cara 1.	7
3.4	Muestra de la interfaz	8
3.5	Manipulación de la interfaz	9
4	Conclusiones	10
5	Referencias	11
6	Anexos	12
6.1	Implementación en MATLAB	13
6.2	Código fuente de la interfaz	14

1 Objetivo de la práctica

Para la presente práctica, se utilizaron los resultados de la tarea 8. En dicha tarea, se empleó la imagen 1 como referencia, la cual contenía marcas en puntos clave del rostro. A partir de esta, se aplicó un proceso de registro a un total de 1519 imágenes, que también estaban previamente marcadas. Posteriormente, se seleccionaron las 100 imágenes cuyas marcas mostraron mayor similitud con la imagen de referencia:



(a) Imagen 1 del conjunto.



(b) Imagen 1 con sus puntos de referencia.

Figure 1: Imagen y puntos de referencia.

Una vez entendido esto, lo que se pide hacer es lo siguiente:

Tomando de referencia la imagen base y el conjunto de las 100 caras que ya fueron registradas y que mostraron una mayor similitud a la imagen de referencia:

- Formar una base de datos (matriz) con los puntos 20 puntos de las 100 caras que más se parecen a la de referencia.
- Aplicar PCA para crear un modelo estadístico de forma para el rostro con los puntos de referencia.
- Crear una interfaz con botones deslizables que permita la manipulación de los parámetros del modelo estadístico de forma.
- Mostrar un histograma de cada uno de los parámetros que refleje el comportamiento de los mismos, así como que muestre los valores máximos y mínimos que pueden tomar.

2 Metodología

2.1 Modelo estadístico de forma usando PCA

A continuación se describe el proceso de modelado estadístico de forma conforme a lo explicado en Prof. Tim Cootes and Dr. Yipeng Hu (2019), University of Basel, Department of Mathematics and Computer Science (2022), Cootes (2025) y Sarkalkan et al. (2014) que retoman la idea del PCA vista en Johnson and Wichern (2007) pero con los resultados de haber hecho el proceso de registro como se explica en Coste (2012).

Para cada rostro, se tiene un conjunto de puntos (20 observaciones) que representan los puntos claves utilizados para el proceso de registro, esto equivale a tener una matriz de datos de tamaño 20×2 por cada una de las imágenes que se ve así:

$$\mathbf{x}^{(i)} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_{20} & y_{20} \end{bmatrix}$$

Ahora, cada uno de esa pareja de puntos va a ser concatenada como un solo vector fila tal y como se muestra a continuación:

$$\mathbf{x}^{(i)} = [x_1 \ y_1 \ x_2 \ y_2 \ \cdots \ x_n \ y_n]$$

Posteriormente, se construye una nueva matriz \mathbf{X} de forma que cada coordenada de cada punto pasara a ser una variable de la nueva matriz con las 100 observaciones dispuestas como filas, conformando así una nueva matriz \mathbf{X} de tamaño 100×40

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)T} \\ \mathbf{x}^{(2)T} \\ \vdots \\ \mathbf{x}^{(m)T} \end{bmatrix}$$

Una vez conformada la matriz \mathbf{X} que contiene los datos de los puntos obtenidos después del proceso de registro, comienza el proceso de modelado estadístico mediante PCA, iniciando por obtener el vector de medias de la matriz que viene a representa la forma promedio de las imágenes.

$$\bar{\mathbf{x}} = \frac{1}{m} \sum_{i=1}^m \mathbf{x}^{(i)}$$

Teniendo el vector de medias listo, se procede a centrar la matriz respecto al vector como sigue:

$$\mathbf{X}_{\text{centrada}} = \mathbf{X} - \bar{\mathbf{x}}$$

donde $\mathbf{1}_m$ es un vector columna de unos.

Acto seguido, se procede con el calculo de la matriz de varianzas y co-varianzas como normalmente se haría en la metodología de PCA:

$$\mathbf{C} = \frac{1}{m-1} \mathbf{X}_{\text{centrada}}^T \mathbf{X}_{\text{centrada}}$$

El siguiente paso es la obtención de los valores y vectores propios, haciendo la resolución de la ecuación característica de la matriz:

$$\mathbf{C}\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

Se hace la selección de los primeros k valores propios que más expliquen la variabilidad de los datos. Un criterio a tomar en cuenta es si k valores propios explican más del 70% de la variabilidad o siendo aun más optimistas, si explican el 90% o más.

$$\mathbf{P} = [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_k]$$

Finalmente, el modelo estadístico de forma permite representar o generar una forma aproximada mediante la siguiente expresión:

$$\mathbf{x} \approx \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}$$

donde $\mathbf{b} \in \mathbb{R}^k$ son los parámetros del modelo (coeficientes de forma), calculados mediante:

$$\mathbf{b} = \mathbf{P}^T(\mathbf{x} - \bar{\mathbf{x}})$$

3 Resultados

3.1 Selección de las componentes del modelo

A continuación se muestra la varianza explicada de las componentes del PCA para los puntos de las imágenes.

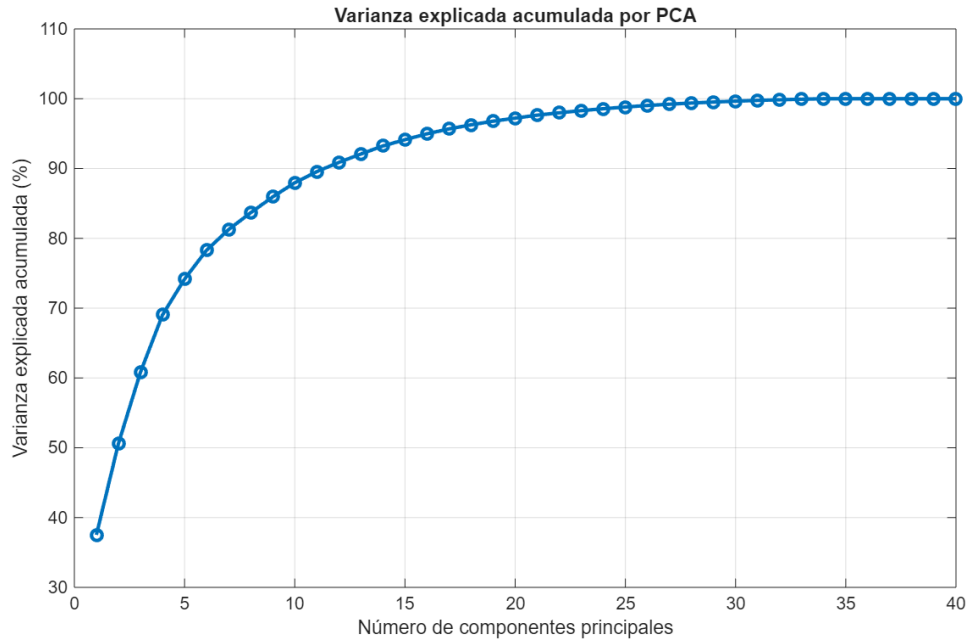


Figure 2: Varianza explicada por componente

El gráfico de los valores propios normalizados, muestra que para el modelo de forma generado a partir de las 100 caras más parecidas a la de referencia, es necesario tomar en cuenta los primeros 12 componentes principales para poder explicar el 90.88% de la varianza.

3.2 Distribución de los parámetros para las 12 componentes del modelo

A continuación se muestran los histogramas que reflejan la variabilidad que hay entre los valores por cada una de las componentes.

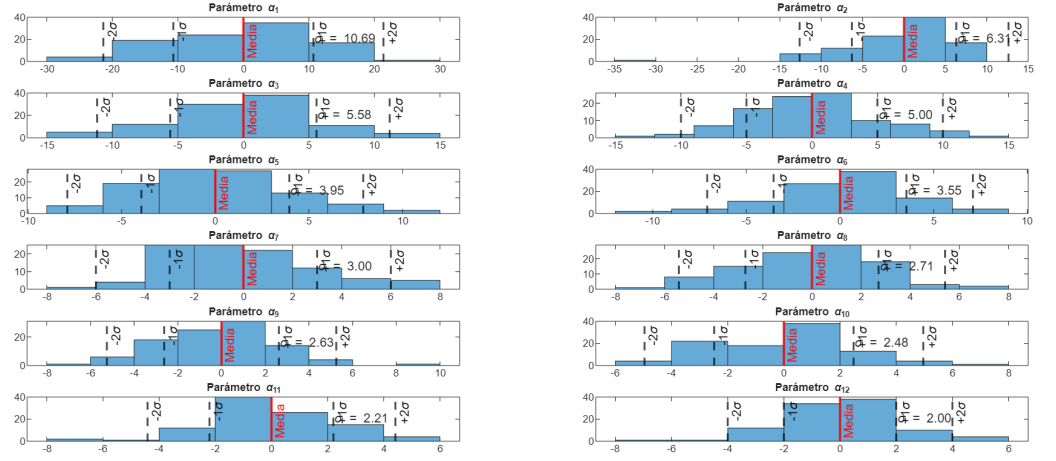


Figure 3: Histograma de los 12 parámetros.

Se observa que todos los componentes tienden a presentar un comportamiento similar a la distribución normal, salvo algunos componentes que presentan valores muy alejados del resto, como se observa en el histograma del parámetro a_2 , a_9 , a_{11} y a_{12} .

3.3 Rostro generado para la cara 1.

Se muestra el resultado para la cara que genera el modelo con los puntos de la primera imagen. Se decidieron unir los puntos por razones estéticas para dejar plasmado de mejor manera la cara que se forma.

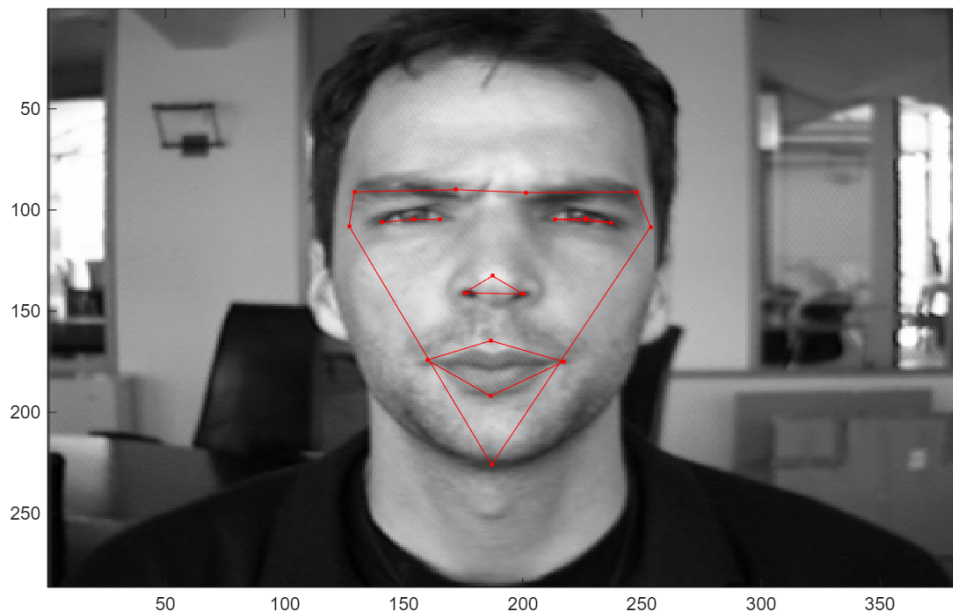


Figure 4: Rostro 1 generado por el modelo

Se observa como el rostro generado se ajusta perfectamente al primer rostro dentro de las zonas de interés, delimitando claramente el rango donde están dichas zonas de interés (la boca, la nariz y los ojos.).

3.4 Muestra de la interfaz

Ahora, se muestra la interfaz que fue desarrollada en MATLAB para manipular de forma más cómoda los valores de los parámetros y que se refleje el impacto de mover dichos valores sobre la cara que se genera.



Figure 5: Interfaz para la manipulación de los parámetros

Se cuentan con los 12 parametros (1 por cada componente principal utilizado), donde los rangos que puede tomar cada uno vienen dados por los valores mínimos y máximos dentro del vector de proyecciones de cada componente.

3.5 Manipulación de la interfaz

Ahora, se muestra un caso donde se fijan los valores de los primeros 2 parámetros en su mínimo permitido y el resto en su máximo permitido según el rango.

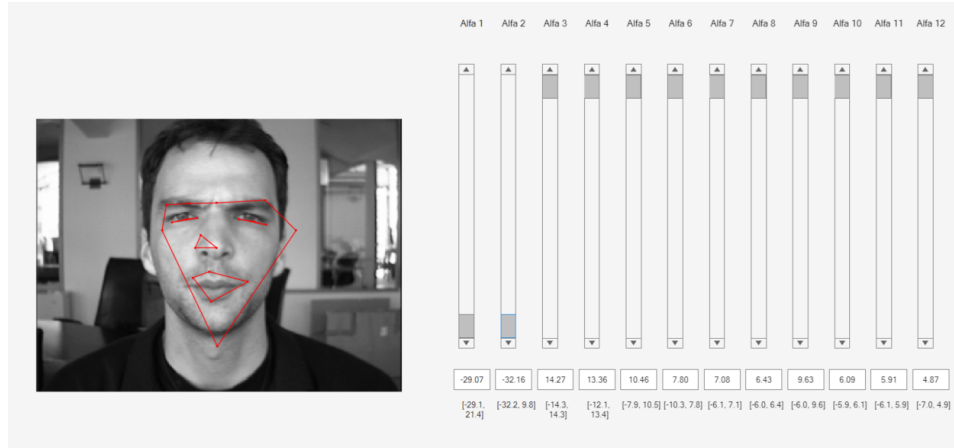


Figure 6: Ejemplo de manipulación de los parámetros

Se observa como al modificar los valores de los parámetros, la cara va deformándose hacia una que no corresponde con la del modelo de esa foto.

4 Conclusiones

Tras la implementación del modelo, es destacable la potencia del modelo para modelar la distribución de los puntos de un conjunto de rostros con marcas mediante el uso de PCA para comprimir dicha información. Permite de una manera sencilla y elegante hacer la reconstrucción de rostros y resume bastante bien la información.

5 Referencias

References

- Cootes, T. F. (2025). Point distribution models. <https://personalpages.manchester.ac.uk/staff/timothy.f.cootes/Models/pdms.html>. University of Manchester.
- Coste, A. (2012). Cs6640: Image processing – project 3: Affine transformation, landmarks registration, non-linear warping. https://www.sci.utah.edu/~acoste/uou/Image/project3/ArthurCOSTE_Project3.pdf. Course project for CS6640, University of Utah.
- Johnson, R. A. and Wichern, D. W. (2007). *Applied Multivariate Statistical Analysis*. Pearson, Upper Saddle River, NJ, 6th edition.
- Prof. Tim Cootes and Dr. Yipeng Hu (2019). 3.2. Statistical Shape Models — MPHY0026 documentation. https://mphy0026.readthedocs.io/en/latest/segmentation/statistical_shape_models.html. Accessed: 27 May 2025.
- Sarkalkan, N., Weinans, H., and Zadpoor, A. A. (2014). Statistical shape and appearance models of bones. *Bone*, 60:129–140.
- University of Basel, Department of Mathematics and Computer Science (2022). Principal component analysis — statistical shape modelling. <https://shapemodelling.cs.unibas.ch/ssm-course/week3/step3-7/>. Parte del curso en línea *Statistical Shape Modelling*.

6 Anexos

6.1 Implementación en MATLAB

Encontrara anexo a este documento un archivo rar con los códigos implementados, incluyendo la interfaz gráfica, esto se debe a que la extensión del proyecto fue tal, que se hizo en varios scripts separados.

6.2 Código fuente de la interfaz

```
1
2 function generar_rostro_GUI
3 % Cargar datos
4 load("faces_eigenvecs.mat", "pcV")
5 load("faces_mean.mat", "mu")
6 load("alphas_caras.mat", "alphas")
7
8 % Crear figura sobre la que se graficará el rostro
9 f = figure('Name', 'Generar rostros con modelo estadístico de forma', ...
10 'Units', 'normalized', 'OuterPosition', [0 0 1 1], ...
11 'NumberTitle','off');
12
13
14 % Se muestra el primer rostro como base de la interfaz
15 ax = axes('Parent', f, 'Position',[0.05 0.1 0.35 0.8]);
16 cara1 = imread("BioID_0001.pgm");
17
18 % Iniciar los parámetros ajustados a la cara de fondo
19 alpha = [-7.7409,-3.4482,3.420,-3.3884,3.2423,-0.5002,3.7840, ...
20 2.8480,-0.2839,0.2117,-0.285,2.4314];
21
22 % Los límites de cada parámetro están dados por los máximos y mínimos
23 % encontrados en el PCA de las 100 imágenes
24 mins = min(alphas, [], 1);
25 maxs = max(alphas, [], 1);
26
27 % Generar rostro mediante PCA
28 new_face = alpha * pcV' + mu;
29 graph_face(new_face);
30
31 % Número de parámetros
32 N = numel(alpha);
33
34 % Espaciado horizontal entre controles
35 spacing = 0.04;
36 base_x = 0.45;
37
38 % Crear botones deslizables para ajustar parámetros alpha
39 sliders = gobjects(1,N);
```

```

40 % Cajas de texto para ajustar parámetros alfa
41 edit_boxes = gobjects(1,N);
42
43 for i = 1:N
44     x_pos = base_x + (i-1) * spacing;
45
46     % Texto con número de parámetro alpha
47     uicontrol(f, 'Style', 'text', ...
48         'Units', 'normalized', ...
49         'Position', [x_pos 0.92 0.035 0.04], ...
50         'String', sprintf('Alfa %d', i), ...
51         'FontSize', 9);
52
53     % Barras deslizantes para manipular el valor del parámetro
54     sliders(i) = uicontrol(f, 'Style', 'slider', ...
55         'Units', 'normalized', ...
56         'Min', mins(i), 'Max', maxs(i), ...
57         'Value', alpha(i), ...
58         'SliderStep', [0.01 0.1], ...
59         'Position', [x_pos+0.005 0.32 0.015 0.55], ...
60         'Callback', @(src,~) sliderCallback(i));
61
62     % Cajas de texto editable para mostrar el valor actual del parámetro o
63     % para ajustarlo de forma exacta
64     edit_boxes(i) = uicontrol(f, 'Style', 'edit', ...
65         'Units', 'normalized', ...
66         'Position', [x_pos 0.24 0.035 0.04], ...
67         'String', num2str(alpha(i), '%.2f'), ...
68         'Callback', @(src,~) editCallback(i));
69
70     % Texto con rango mínimo y máximo
71     uicontrol(f, 'Style', 'text', ...
72         'Units', 'normalized', ...
73         'Position', [x_pos 0.18 0.04 0.04], ...
74         'String', sprintf('[%.1f, %.1f]', mins(i), maxs(i)), ...
75         'FontSize', 8);
76 end
77
78 % Actualización de la cara cuando se ajustan los parámetros en
79 % la interfaz

```

```

80
81 % Actualización cuando se mueve una barra deslizante
82 function sliderCallback(i)
83 alpha(i) = sliders(i).Value; % Re-evaluar parámetro alfa
84 edit_boxes(i).String = sprintf('%.2f', alpha(i)); % Ajustar cajas de texto
85 actualizar_rostro(); % Encontrar nuevo rostro
86 end
87
88 % Actualización cuando se alteran las cajas de texto
89 function editCallback(i)
90 val = str2double(edit_boxes(i).String); % obtener valor numérico del texto
91 if isnan(val), return; end % Error si no es un número
92 val = max(mins(i), min(maxs(i), val)); % Limitar al rango adecuado
93 alpha(i) = val; % Actualizar alfa
94 sliders(i).Value = val; % Actualizar barra deslizante
95 edit_boxes(i).String = sprintf('%.2f', val); % Actualizar caja de texto
96 actualizar_rostro(); % Graficar nueva cara
97 end
98
99 % Actualizar rostro generado
100 function actualizar_rostro()
101 new_face = alpha * pcV' + mu;
102 graph_face(new_face);
103 end
104
105 % Función para graficar la cara dada lista de coordenadas (1x40)
106 function graph_face(point_list)
107 cla(ax); % Limpiar imagen
108 imshow(caral, 'Parent', ax); % Volver a graficar rostro
109 hold(ax, "on");
110
111 points = reshape(point_list, 2, [])'; % Darle forma de matriz a los
112 % puntos (20, 2)
113
114 % Struct donde se indica a que parte del cuerpo pertenece cada
115 % índice
116 partes = {
117     [3, 18, 4, 19, 3], 'Labios';
118     [10, 1, 11, 10], 'Ojo Izquierdo';
119     [12, 2, 13, 12], 'Ojo Derecho';

```



```

120         [16, 15, 17, 16], 'Nariz';
121         [20, 14, 8, 7, 6, 5, 9, 20], 'Contorno'
122     };
123
124     % Se grafica cada parte del cuerpo, uniendo los puntos
125     for k = 1:size(partes,1)
126         idx = partes{k,1};
127         plot(ax, points(idx,1), points(idx,2), 'r.-', 'DisplayName', partes{k,2});
128     end
129     hold(ax, "off");
130 end
131 end

```