

# Catálogo de Algoritmos

## Integrantes:

- Daniel Castro Campos - Carnet 2016165624
- Gustavo Segura Umaña - Carnet 2016089824
- Joseph Vargas Blanco - Carnet 2016200170

## 1 Semana 3: SVD e imágenes

### 1.1 Compresión de imágenes

---

```
1 clear;
2 clc;
3 close all;
4 pkg load image;
5
6 I_color = imread("WingedFigure.jpg");
7 A = I_color(:,:,1);
8
9 A =im2double(A);
10 val_sing=diag(A);
11 [m,n]=size(A);
12
13 [U,S,V] =svd(A);
14
15 r=50;
16 Ur= U(:,1:r);
17 Vr= V(:,1:r);
18 Sr= S(1:r,1:r);
19
20 B=Ur*Sr;
21 C= Vr';
22 Ar=B*C;
23 Ar = im2uint8(Ar);
24
25
26 subplot(1,2,1);
27 imshow(A)
28 title('Imagen original');
29 subplot(1,2,2);
30 imshow(Ar)
31 title('Imagen reducida');
32
33 error=norm(A-im2double(Ar),'fro');
```

---

## 1.2 Compresión de imágenes con Proyección Bilateral

---

```
1 import numpy as np
2 from PIL import Image, ImageOps
3 import matplotlib.pyplot as plt
4
5 def imageToArray(filename:str, rgb:bool=False):
6     img = Image.open(filename)
7     if not rgb:
8         img = ImageOps.grayscale(img)
9     img_array = np.array(img)
10    return img_array
11
12 def bilateral(A, r, p):
13     if(type(p)==int and p>=2):
14         Y2=np.array(np.random.randint(255, size=A.shape), dtype=float)
15         for k in range(1,p+1):
16             Y1=np.matmul(A,Y2)
17             Y2=np.matmul(A.T,Y1)
18         Q, R = np.linalg.qr(Y2)
19         Qr=Q[ : , :r]
20         B=np.matmul(A,Qr)
21         C=Qr.T
22         return (B,C)
23
24
25 A = imageToArray("WingedFigure.jpg")
26 B,C=bilateral(A,30,p=5)
27 Ar=np.matmul(B,C)
28
29 plt.subplot(1, 2, 1)
30 plt.title('Imagen original')
31 plt.imshow(A, cmap='gray', vmin=0, vmax=255)
32
33 plt.subplot(1, 2, 2)
34 plt.title('Imagen resultante')
35 plt.imshow(Ar, cmap='gray', vmin=0, vmax=255)
36 plt.show()
```

---

## 2 Semana 4: Operaciones geométricas e interpolación

### 2.1 Filtro de la mediana

---

```
1 clc; clear;
2 pkg load image;
3
4 A = imread('WingedFigureNoise.jpg');
5 A = A(:,:,1);
6
7 [m,n] = size(A);
```

```

8  B = zeros(m,n);
9
10
11 B(1, 1) = median(A(1:2, 1:2)(:));
12 B(m, 1) = median(A(m-1:m, 1:2)(:));
13 B(1, n) = median(A(1:2, n-1:n)(:));
14 B(m, n) = median(A(m-1:m, n-1:n)(:));
15
16 for x = 2:m-1
17     B(x, 1) = median(A(x-1:x+1, 1:2)(:));
18     B(x, n) = median(A(x-1:x+1, n-1:n)(:));
19 endfor
20
21 for y = 2:n-1
22     B(1, y) = median(A(1:2,y-1:y+1)(:));
23     B(m, y) = median(A(m-1:m,y-1:y+1)(:));
24 endfor
25
26 for x = 2:m-1
27     for y = 2:n-1
28         neighborhood = A(x-1:x+1, y-1:y+1);
29         temp = neighborhood(:);
30         B(x, y) = median(temp);
31     end
32 end
33 B = uint8(B);
34
35
36 subplot(1, 2, 1);
37 imshow(A);
38 title('Imagen con ruido');
39
40 subplot(1, 2, 2);
41 imshow(B);
42 title('Imagen resultante');

```

---

## 2.2 Transformadas Afines Lineales: Rotación

---

```

1  import numpy as np
2  from PIL import Image, ImageOps
3  import matplotlib.pyplot as plt
4
5  def imageToArray(filename:str, rgb:bool=False):
6      img = Image.open(filename)
7      if not rgb:
8          img = ImageOps.grayscale(img)
9      img_array = np.array(img)
10     return img_array
11
12 def rotation(A:np.array, angle:int):
13     angle = np.radians(angle)
14     result = np.zeros(A.shape, dtype=np.uint8)

```

```

15     M, N = A.shape[0], A.shape[1]
16     xc= M//2
17     yc= N//2
18     a0 = np.cos(angle)
19     a1 = np.sin(angle)
20     b0 = -np.sin(angle)
21     b1 = np.cos(angle)
22     for i in range(M):
23         for j in range(N):
24             new_i=round(b0*(j - xc) + b1 * (i - yc) + yc)
25             new_j=round(a0*(j - xc) + a1 * (i - yc) + xc)
26             if (new_i>=0 and new_i<M and new_j>=0 and new_j<N ):
27                 result[new_i][new_j]=A[i][j]
28     return result
29 def medianFilterBN(A:np.array):
30     result = np.zeros(A.shape, dtype=np.uint8)
31     M, N = A.shape[0], A.shape[1]
32     # esquinas
33     result[0][N-1] = np.median([A[0][N-1],
34                                 A[1][N-1],
35                                 A[0][N-2]])
36     result[M-1][0] = np.median([A[M-2][0],
37                                 A[M-1][0],
38                                 A[M-1][1]])
39     result[M-1][N-1] = np.median([A[M-1][N-1],
40                                 A[M-2][N-1],
41                                 A[M-1][N-2]])
42     result[0][0] = np.median([A[0][0],
43                               A[0][1],
44                               A[1][0]])
45     for y in range(1,N-1):
46         result[0][y] = np.median(A[0:2,y-1:y+1])
47         result[M-1][y] = np.median(A[M-2:M,y-1:y+1])
48     for x in range(1,M-1):
49         result[x][0] = np.median(A[x-1:x+2,:2])
50         result[x][N-1] = np.median(A[x-1:x+2,-2:])
51         for y in range(1,N-1):
52             result[x][y] = np.median(A[x-1:x+2,y-1:y+2])
53     return result
54
55
56 A = imageToArray("WingedFigure.jpg")
57 B = rotation(A, 35)
58 C = medianFilterBN(B)
59
60
61 plt.subplot(1, 3, 1)
62 plt.title('Imagen original')
63 plt.imshow(A, cmap='gray', vmin=0, vmax=255)
64
65 plt.subplot(1, 3, 2)
66 plt.title('Imagen rotada')
67 plt.imshow(B, cmap='gray', vmin=0, vmax=255)
68

```

```

69 plt.subplot(1, 3, 3)
70 plt.title('Imagen rotada y filtrada')
71 plt.imshow(C, cmap='gray', vmin=0, vmax=255)
72 plt.show()

```

---

## 2.3 Transformadas Afines Lineales: Traslación

---

```

1  import numpy as np
2  from PIL import Image, ImageOps
3  import matplotlib.pyplot as plt
4
5  def imageToArray(filename:str, rgb:bool=False):
6      img = Image.open(filename)
7      if not rgb:
8          img = ImageOps.grayscale(img)
9      img_array = np.array(img)
10     return img_array
11
12 def translation(A:np.array, x:int, y:int):
13     result = np.zeros(A.shape, dtype=np.uint8)
14     M, N = A.shape[0], A.shape[1]
15     for i in range(M):
16         for j in range(N):
17             new_i=i+y
18             new_j=j+x
19             if (new_i>=0 and new_i<M and new_j>=0 and new_j<N ):
20                 result[new_i][new_j]=A[i][j]
21     return result
22
23 A = imageToArray("WingedFigure.jpg")
24 B = translation(A, 50,200)
25
26
27 plt.subplot(1, 2, 1)
28 plt.title('Imagen original')
29 plt.imshow(A, cmap='gray', vmin=0, vmax=255)
30
31 plt.subplot(1, 2, 2)
32 plt.title('Imagen trasladada')
33 plt.imshow(B, cmap='gray', vmin=0, vmax=255)
34 plt.show()

```

---

## 2.4 Transformadas Afines No Lineales: Efecto Rippling

---

```

1  import numpy as np
2  from PIL import Image, ImageOps
3  import matplotlib.pyplot as plt
4
5  def imageToArray(filename:str, rgb:bool=False):
6      img = Image.open(filename)

```

```

7     if not rgb:
8         img = ImageOps.grayscale(img)
9     img_array = np.array(img)
10    return img_array
11
12    def rippling(A:np.array,Ax=15,Ay=15,Lx=75,Ly=75):
13        result = np.zeros(A.shape,dtype=np.uint8)
14        M, N = A.shape[0],A.shape[1]
15        for x in range(M):
16            for y in range(N):
17                x_new=round(x+Ax*np.sin(2*np.pi*y/Lx))
18                y_new=round(y+Ay*np.sin(2*np.pi*x/Ly))
19                if (x_new>=0 and x_new<M and y_new>=0 and y_new<N ):
20                    result[x_new][y_new]=A[x][y]
21        return result
22
23    A = imageToArray("WingedFigure.jpg")
24    B=rippling(A,Ax=15,Ay=15,Lx=200,Ly=200)
25
26
27    plt.subplot(1, 2, 1)
28    plt.title('Imagen original')
29    plt.imshow(A, cmap='gray', vmin=0, vmax=255)
30
31    plt.subplot(1, 2, 2)
32    plt.title('Imagen transformada')
33    plt.imshow(B, cmap='gray', vmin=0, vmax=255)
34    plt.show()

```

---

```

1    import numpy as np
2    from PIL import Image, ImageOps
3    import matplotlib.pyplot as plt
4
5    def imageToArray(filename:str, rgb:bool=False):
6        img = Image.open(filename)
7        if not rgb:
8            img = ImageOps.grayscale(img)
9        img_array = np.array(img)
10       return img_array
11
12    def rippling(A:np.array,Ax=15,Ay=15,Lx=75,Ly=75):
13        result = np.zeros(A.shape,dtype=np.uint8)
14        M, N = A.shape[0],A.shape[1]
15        for x in range(M):
16            for y in range(N):
17                x_new=round(x+Ax*np.sin(2*np.pi*y/Lx))
18                y_new=round(y+Ay*np.sin(2*np.pi*x/Ly))
19                if (x_new>=0 and x_new<M and y_new>=0 and y_new<N ):
20                    result[x_new][y_new]=A[x][y]
21        return result
22
23    def ripplingAnimation(A, filename="out.gif", minA=0, maxA=50, step=5,Lx=200,Ly=200):
24        images=[]

```

---

```

25     for a in range(minA, maxA, step):
26         images.append(rippling(A,Ax=a,Ay=a,Lx=Lx,Ly=Ly))
27     B_img=list(map(Image.fromarray, images))
28     B_img[0].save(filename,save_all=True, append_images=B_img[1:], loop=0)
29
30 A_color=imageToArray("WingedFigure.jpg", rgb=True)
31 ripplingAnimation(A_color, filename="rippling.gif", minA=0, maxA=100, step=5,Lx=200,Ly=200)

```

---

## 3 Semana 5: Transformaciones a escala de grises

### 3.1 Transformacion Lineal

---

```

1  import cv2
2  import matplotlib.pyplot as plt
3  import numpy as np
4
5
6  image0 = cv2.imread("boat.jpg",0)
7  plt.subplot(1,2,1)
8  plt.imshow(image0, cmap='gray')
9
10 new_image = np.zeros(image0.shape, image0.dtype)
11 alpha = 1.8
12 beta = 0
13
14 for y in range(image0.shape[0]):
15     for x in range(image0.shape[1]):
16         new_image[y,x] = np.clip(alpha*image0[y,x] + beta, 0, 255)
17
18 plt.subplot(1,2,2)
19 plt.imshow(new_image, cmap='gray')
20 plt.show()

```

---

### 3.2 Transformacion Autocontraste

---

```

1  clc;
2  clear;
3  close all;
4
5  A = imread('boat_new.jpg');
6  subplot(1,2,1);
7  imshow(A)
8  title('Imagen original');
9
10 A = double(A);
11 [m,n] = size(A);
12 B = zeros(m,n);
13
14 % Valores del nuevo contraste

```

```

15 rmin = min(min(A)); % Extrae valor de las columnas
16 rmax = max(max(A));
17
18 alpha = 255 / (rmax - rmin);
19 beta = rmin;
20
21 B = alpha * (A - beta);
22
23
24 B = uint8(B);
25 subplot(1,2,2);
26 imshow(B)
27 title('Imagen Modificada');

```

---

### 3.3 Transformacion Negativo

```

1 import cv2
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5
6 image0 = cv2.imread("boat.jpg",0)
7 plt.subplot(1,2,1)
8 plt.imshow(image0, cmap='gray')
9
10 new_image = np.zeros(image0.shape, image0.dtype)
11 a = -1
12 b = 255
13
14 for y in range(image0.shape[0]):
15     for x in range(image0.shape[1]):
16         new_image[y,x] = np.clip(a*image0[y,x] + b, 0, 255)
17
18 plt.subplot(1,2,2)
19 plt.imshow(new_image, cmap='gray')
20 plt.show()

```

---

### 3.4 Transformacion Gamma

```

1 import cv2
2 import matplotlib.pyplot as plt
3 import numpy as np
4
5 image0 = cv2.imread("boat.jpg")
6 plt.subplot(1,2,1)
7 plt.imshow(image0, cmap='gray')
8
9 image0.astype(np.double)
10 new_image = np.zeros(image0.shape, image0.dtype)
11 gamma = 1.9

```



```

12 c = 1
13
14 for y in range(image0.shape[0]):
15     for x in range(image0.shape[1]):
16         new_image[y,x] = np.clip(c * (image0[y,x] ** gamma), 0, 255)
17
18 new_image.astype(np.uint8)
19 plt.subplot(1,2,2)
20 plt.imshow(new_image, cmap='gray')
21 plt.show()

```

---

## 3.5 Transformacion Logaritmica

---

```

1  clc;
2  clear;
3  close all;
4
5  A = imread('log.jpg');
6  subplot(1,2,1);
7  imshow(A)
8  title('Imagen original');
9
10 A = im2double(A);
11 [m,n] = size(A);
12 B = zeros(m,n);
13
14 c = 0
15 for c = 0.1:0.1:5
16     B = c.*log(1+A);
17     subplot(1,2,2);
18     imshow(B);
19     title(['Imagen Modificada: c = ' num2str(c)]);
20     pause(0.05);
21 endfor

```

---

## 4 Semana 6: Histogramas

### 4.1 Mostrar el histograma

---

```

1  from skimage import io
2  import matplotlib.pyplot as plt
3
4  # Cargar imagen
5  image = io.imread('peppers.jpg')
6
7  # Mostrar imagen
8  plt.subplot(2, 1, 1)
9  plt.title('Imagen')
10 plt.imshow(image, cmap='gray', vmin=0, vmax=255)

```

```

11
12 # Mostrar histograma
13 plt.subplot(2, 1, 2)
14 plt.title('Histograma')
15 plt.hist(image.ravel(), bins = 256, range=(0, 255))
16
17 plt.show()

```

---

## 4.2 Ecualización de histograma

---

```

1  clc
2  clear
3  close all
4
5  A = imread('peppers.jpg');
6  subplot(2,2,1);
7  imshow(A)
8  title('Imagen original');
9
10
11 % Calcular el histograma
12 h2 = zeros(256, 1);
13 [m, n] = size(A);
14 for i = 1:255
15     h2(i + 1) = sum(sum(A == i));
16 endfor
17 subplot(2,2,2);
18 bar(0:255, h2)
19 title('Histograma');
20 xlim([0 255])
21
22 % Calcular la distribucion acumulativa
23 [m, n] = size(A);
24 v_ac = zeros(256, 1);
25 for i = 0:255
26     v_ac(i + 1) = sum(h2(1:i+1))/(m*n);
27 endfor
28 subplot(2,2,3)
29 bar(0:255, v_ac)
30 title('Distribucion acumulativa');
31 xlim([0 255])
32
33 % Metodos de ecualizacion
34 B = zeros(m, n);
35 A = double(A);
36 for x = 1:m
37     for y = 1:n
38         B(x, y) = round(v_ac(A(x, y) + 1) * 255);
39     endfor
40 endfor
41
42 B = uint8(B);

```

```
43 subplot(2,2,4);
44 imshow(B)
45 title('Imagen ecualizada');
```

---

### 4.3 Estiramiento del histograma

---

```
1 from skimage import io
2 import matplotlib.pyplot as plt
3
4 # Cargar imagen
5 image = io.imread('sydney.jpg', True)
6 # Estirar imagen
7 image_stretched = [(255 - 0) / (image.max() - image.min())] * (image - image.min())
8
9 # Mostrar imagen original
10 plt.subplot(2, 2, 1)
11 plt.title('Imagen original')
12 plt.imshow(image, cmap='gray', vmin=0, vmax=255)
13
14 # Mostrar imagen con histograma estirado
15 plt.subplot(2, 2, 2)
16 plt.title('Imagen con histograma estirado')
17 plt.imshow(image_stretched, cmap='gray', vmin=0, vmax=255)
18
19 # Histograma original
20 plt.subplot(2, 2, 3)
21 plt.title('Histograma original')
22 plt.hist(image.ravel(), bins = 256, range=(0, 255))
23
24 # Histograma estirado
25 plt.subplot(2, 2, 4)
26 plt.title('Histograma estirado')
27 plt.hist(image_stretched.ravel(), bins = 256, range=(0, 255))
28
29 plt.show()
```

---

### 4.4 Reducción del histograma

---

```
1 from skimage import io
2 import matplotlib.pyplot as plt
3
4 # Cargar imagen
5 image = io.imread('peppers.jpg', True)
6 # Reducir imagen
7 max = 175
8 min = 100
9 image_stretched = [(max - min) / (image.max() - image.min())]
10                    * (image - image.min()) + min)
11
12 # Mostrar imagen original
```

```

13 plt.subplot(2, 2, 1)
14 plt.title('Imagen original')
15 plt.imshow(image, cmap='gray', vmin=0, vmax=255)
16
17 # Mostrar imagen con histograma estirado
18 plt.subplot(2, 2, 2)
19 plt.title('Imagen con histograma reducido')
20 plt.imshow(image_stretched, cmap='gray', vmin=0, vmax=255)
21
22 # Histograma original
23 plt.subplot(2, 2, 3)
24 plt.title('Histograma original')
25 plt.hist(image.ravel(), bins=256, range=(0, 255))
26
27 # Histograma estirado
28 plt.subplot(2, 2, 4)
29 plt.title('Histograma reducido')
30 plt.hist(image_stretched.ravel(), bins=256, range=(0, 255))
31
32 plt.show()

```

---

## 4.5 Especificación de histograma

---

```

1  clc
2  clear
3  close all
4
5  A = imread('peppers.jpg');
6  subplot(4,2,1);
7  imshow(A)
8  title('Imagen original');
9
10 B = imread('sydney.jpg');
11 subplot(4,2,2);
12 imshow(B)
13 title('Imagen estandar');
14
15
16 % Calcular el histograma original
17 h1 = zeros(256, 1);
18 [m, n] = size(A);
19 for i = 1:255
20     h1(i + 1) = sum(sum(A == i));
21 endfor
22 subplot(4,2,3);
23 bar(0:255, h1)
24 title('Histograma original');
25 xlim([0 255])
26
27 % Calcular el histograma estandar
28 h2 = zeros(256, 1);
29 [m, n] = size(B);

```

```

30 for i = 1:255
31     h2(i + 1) = sum(sum(B == i));
32 endfor
33 subplot(4,2,4);
34 bar(0:255, h2)
35 title('Histograma estandar');
36 xlim([0 255])
37
38 % Calcular la distribucion acumulativa
39 [m, n] = size(A);
40 v_ac = zeros(256, 1);
41 for i = 0:255
42     v_ac(i + 1) = sum(h1(1:i+1)/(m*n));
43 endfor
44 subplot(4,2,5)
45 bar(0:255, v_ac)
46 title('Distribucion acumulativa original');
47 xlim([0 255])
48
49 % Calcular la distribucion acumulativa
50 [m2, n2] = size(B);
51 v2_ac = zeros(256, 1);
52 for i = 0:255
53     v2_ac(i + 1) = sum(h2(1:i+1)/(m2*n2));
54 endfor
55 subplot(4,2,6)
56 bar(0:255, v2_ac)
57 title('Distribucion acumulativa estandar');
58 xlim([0 255])
59
60 % Metodos de ecualizacion
61 C = zeros(m, n);
62 A = double(A);
63 for x = 1:m
64     for y = 1:n
65         C(x, y) = round(v2_ac(A(x, y) + 1) * 255);
66     endfor
67 endfor
68
69 C = uint8(C);
70 subplot(4,1,4);
71 imshow(C)
72 title('Imagen original con especificacion de histograma estandar');

```

---