MEMORIA PROYECTO IMAGEN

Índice

Trabajo hecho por Carlos Fernández Deus, Laura González Lemos y Diego Díaz Rodríguez

·Introducción del proyecto

Lo que hemos hecho en este proyecto es trabajar con las imágenes principales dadas y también, a parte de estas, tenemos fotos de una fresa, de un plátano y de una pera. A partir de estas hemos hecho los cambios pedidos en las tareas.

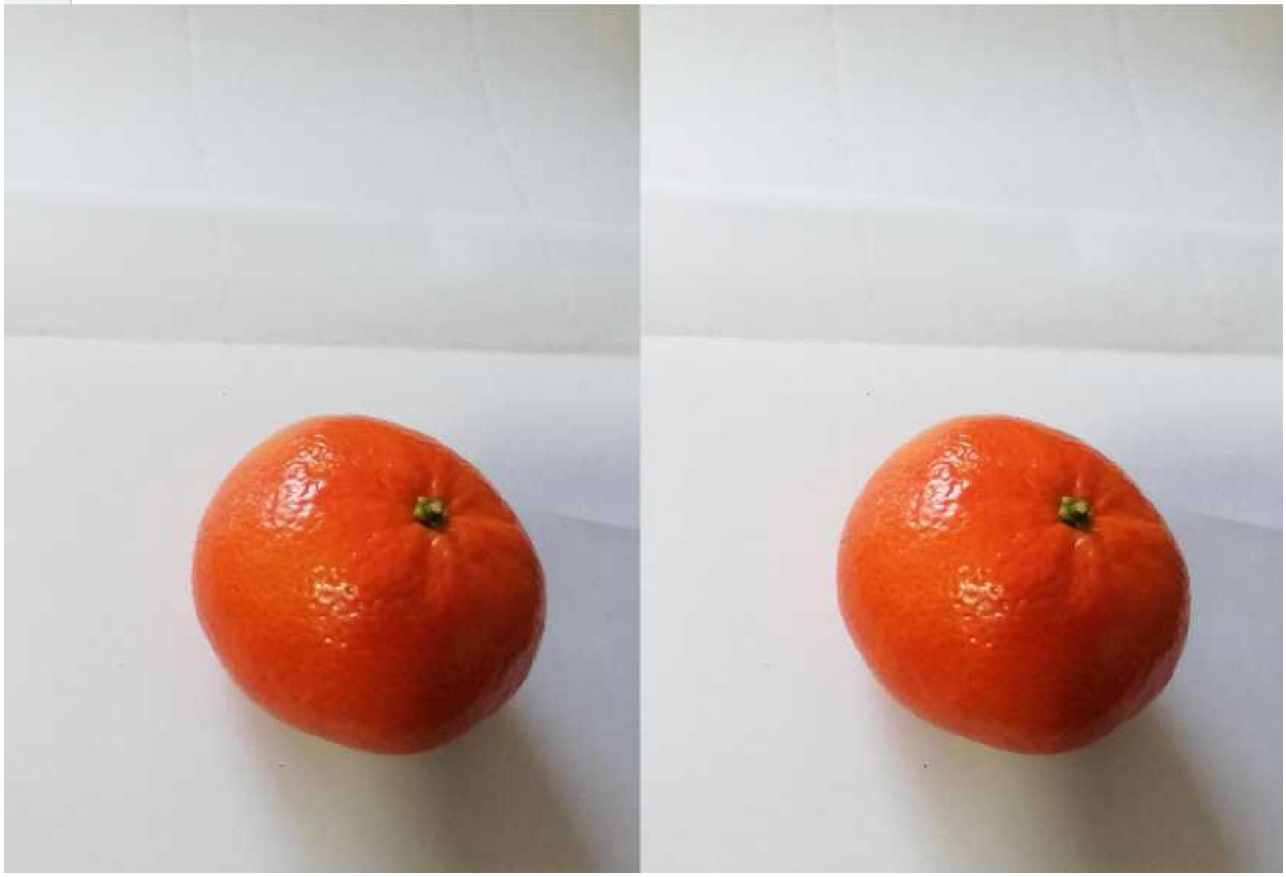
·Explicación de cada tarea

·Tarea 1

En esta tarea se pide que implementemos una mejora en el color.

T1: Mejora de color

```
% Leo imagen y convierto a double
imagen = imread('F11.jpg');
imagen_contrastada = MejoraDeColor(imagen);
```



Para ello cargamos la imagen en la que queramos hacer la mejora y usamos la función '**imagen_contrastada = MejoraDeColor(imagen)**' y dentro de esta llamamos a estas otras dos: '**imbn = ImagenGrises(imc)**' y'**imagen_contrastada = MaxContraste(imagen, imagen_gris)**'.

- funcion imagen_contrastada = MejoraDeColor(imagen)

```
function imagen_contrastada = MejoraDeColor(imagen)
    imagen = im2double(imagen);

    %Convierto a escala de grises
    imagen_grises = ImagenGrises(imagen); %también se podría hacer con  rgb2gray(imagen)

    % Maximización del contraste
    imagen_contrastada = MaxContraste(imagen, imagen_grises);

    % Visualización de la mejora de color
    comparacion = [imagen imagen_contrastada];
    imshow(comparacion);
end
```

- function imbn = ImagenGrises(imc)

```
function imbn = ImagenGrises(imc)
    R = imc(:,:,1);
    G = imc(:,:,2);
    B = imc(:,:,3);

    Y = 0.3*R+0.59*G+0.11*B;    % Calculo Y
    % Para que esté en escala de grises --> Y = R = G = B
    imbn(:,:,1) = Y;    % R = Y
    imbn(:,:,2) = Y;    % G = Y
    imbn(:,:,3) = Y;    % B = Y
end
```

- function imagen_contrastada = MaxContraste(imagen, imagen_gris)

```
function imagen_contrastada = MaxContraste(imagen, imagen_gris)
    imagen = im2double(imagen);
    imagen_gris = im2double(imagen_gris);

    min_luminancia = min(min(imagen_gris));            % Mínimo de luminancia
    max_luminancia = max(max(imagen_gris));            % Máximo de luminancia

    imagen_contrastada = ((imagen - min_luminancia)./(max_luminancia-min_luminancia))*1 + 0;  % Operación máximo contraste
end
```

- **'imagen_contrastada = MejoraDeColor(imagen)'**: Lo que esta función hace es primero convertir la imagen en un double para poder manejarla con facilidad. A continuación, llama a '**imbn = ImagenGrises(imc)**' que hace su función convirtiéndolo en una escala de grises. Por último, llama a '**imagen_contrastada = MaxContraste(imagen, imagen_gris)**', que maximiza el contraste de la imagen. Posteriormente mostramos por pantalla la imagen original y la contrastada creada tras maximizar el contraste.

- '**imbn = ImagenGrises(imc)**': Esta función primeramente separa unaimagen en sus tres matrices componentes, a continuación, calcula la luminosidad de cada pixel y se iguala cada componente a esta luminosidad.

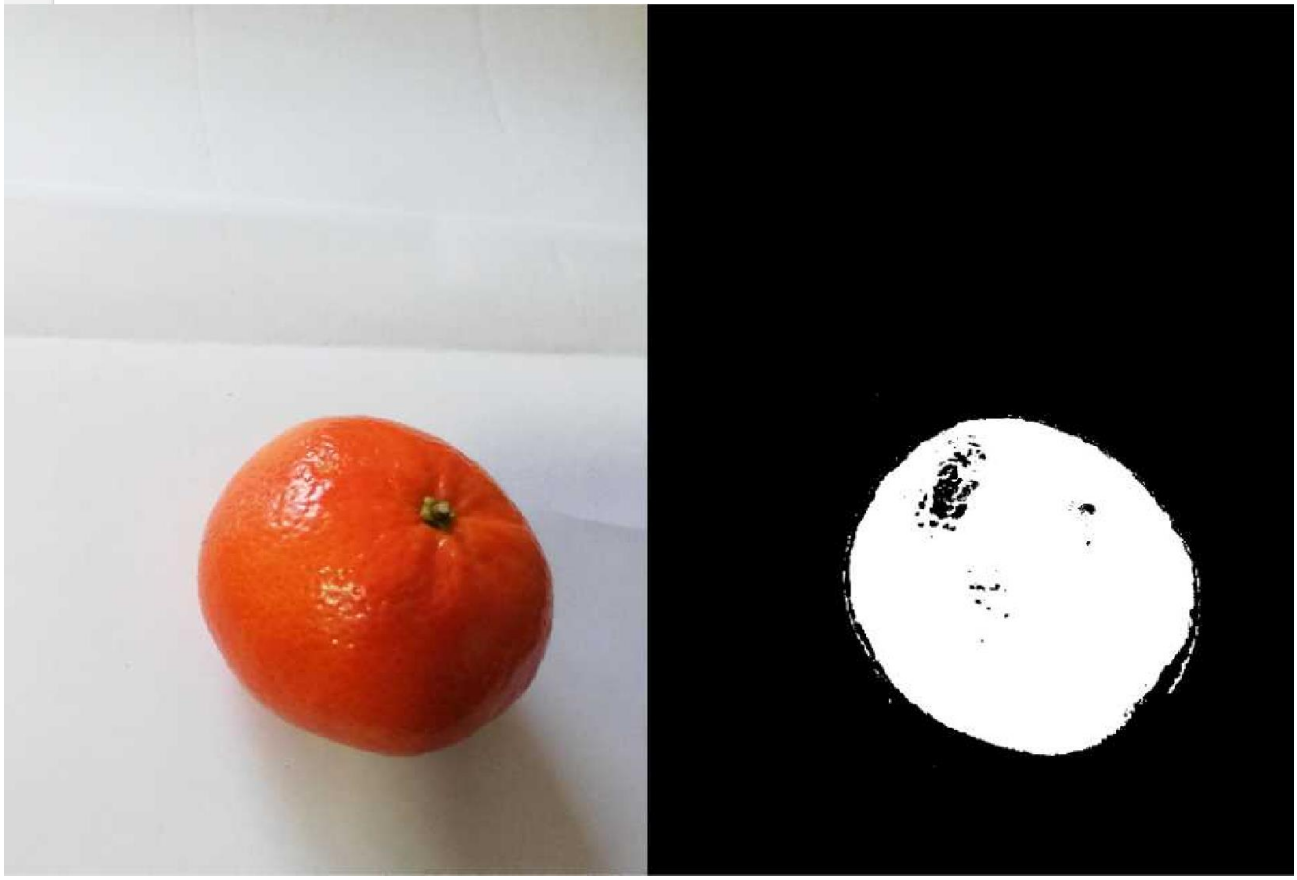
- **'imagen_contrastada = MaxContraste(imagen, imagen_gris)'**: En esta se convierten a double ambas imágenes tanto la original como la de escala degrises y se calcula la máxima y la mínima luminosidad de la imagen en escala de grises, para así usarlos en la operación para sacar el máximo contraste y poder obtener la imagen contrastada al máximo.

·Tarea 2

Esta tarea pretende que a partir de la imagen original cambiemos a negro todo el fondo que no pertenece a la fruta y dejemos marcada que parte dela imagen en la que se sitúa la fruta dejando esa parte en blanco, contrastando con el resto de la imagen en negro.

T2: Segmentación

```
4  imagen_segmentada = Segmentacion(imagen_contrastada);
```



En esta función lo que hacemos es llamar a la función '**imagen_segmentada = Segmentacion(imagen_contrastada)**'.

Tarea 2

- **imagen_segmentada = Segmentacion(imagen_contrastada)**

```
function imagen_segmentada = Segmentacion(imagen_contrastada)
    imagen = imagen_contrastada;
    imagen_hsv = rgb2hsv(imagen);  %% convertir a color
    %imshow([imagen imagen_hsv]);
    imagen_hsv_gris = ImagenGrises(imagen_hsv);
    %imshow([imagen_hsv imagen_hsv_gris]);
    S = imagen_hsv(:,:,2); % esto es para pillar S
    level = graythresh(S); % aplicar umbral con esta función de matlab
    BW = im2double(imbinarize(imagen_hsv_gris,level)); % crae una imagen binaria a partir de la imagen
    imshow([imagen BW]);
    imagen_segmentada = BW;
end
```

- **‘imagen_segmentada = Segmentacion(imagen_contrastada)’**: Esta función coge la imagen calculada en la tarea 1, es decir, la imagen contrastada. Posteriormente convierte a color tanto la imagen original como la de escala de grises. Y con ‘S’ que es igualado al vector saturación, se calculael umbral con la función *graythresh.* Este umbral es usado como media para el *imbinarize* de la imagen en escala de grises convertida a HSV y convirtiendo así la imagen en una imagen con dos colores únicos tanto blanco como negro, siendo este último el fondo de la imagen y el blanco,la fruta.

·Tarea 3

El objetivo de esta tarea es crear un vector con las 3 componentes de cada imagen que tenemos pero solo en la zona de la máscara, es decir, únicamente los vectores de componentes de las frutas, sin tener en cuenta el fondo.

T3: Extracción de características

```
prototipo = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada)
```

```
prototipo = 1×3
    0.7047    0.2139    0.0800
```

En este caso también llamaremos únicamente a una función, esta es‘**prototipo = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada)**’.

Tarea 3

- **prototipo = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada)**

```
function prototipo = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada)
    M = imagen_segmentada(:,:,2);

    R = imagen_contrastada(:,:,1);
    G = imagen_contrastada(:,:,2);
    B = imagen_contrastada(:,:,3);

    frutaR = mean(R(M == 1)); %los valores medios
    frutaG = mean(G(M == 1));
    frutaB= mean(B(M == 1));

    prototipo = [frutaR frutaG frutaB];
end
```

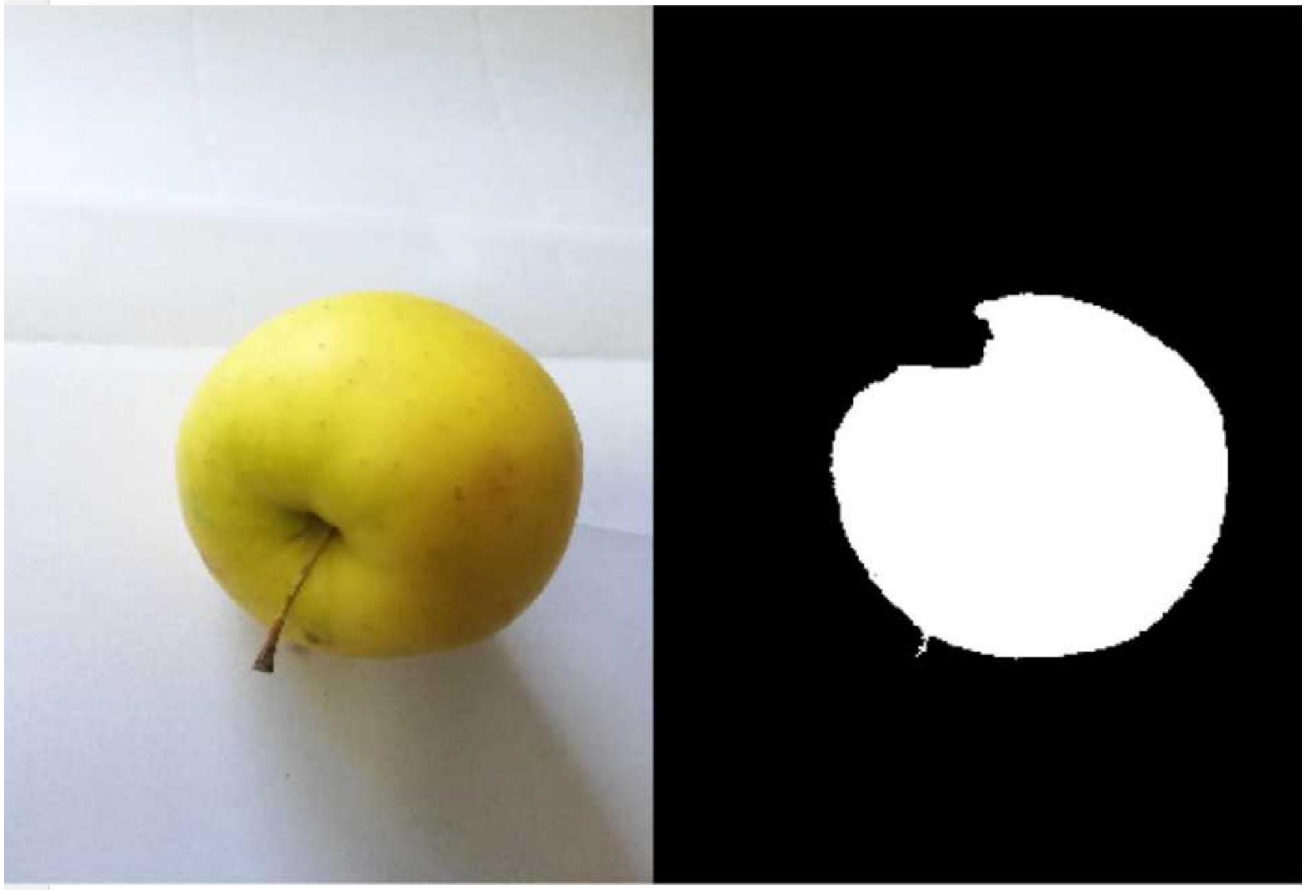
- ‘**prototipo = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada)**’: Se usa la imagen del ejercicio anterior, para así poder tomar los valores de solo la fruta y luego se calculan los valores medios de cada componente para posteriormente introducirlos en el vectorcomponentes.

·Tarea 4

En esta tarea lo que se hace es calcular la media de cada componente de las tresfotos de cada fruta y a continuación se hace la media de estas componentes, para así crear el alfabeto con las distintas clases de frutas.

T4: Creación de un alfabeto

```
% Imagenes de manzanas:
imagen_1_manzana = imread('F01.jpg');
imagen_2_manzana = imread('F02.jpg');
imagen_3_manzana = imread('F03.jpg');
% Vector prototipo de las manzanas:
prototipo_manzanas = PromedioTotal(imagen_1_manzana,imagen_2_manzana,imagen_3_manzana)
```



```
prototipo_manzanas = 1×3
    0.6949    0.5848    0.1269
```

Lo que nosotros hacemos es repetir el mismo proceso con cada fruta (de lacual cogemos 3 fotos), primero asignamos un nombre, en este caso

imagen_1, imagen_2, imagen_3, a cada una de las fotos de cada fruta yluego llamamos a la función '**promedio = PromedioTotal(imagen_1, imagen_2, imagen_3)**'.

Tarea 4

- promedio = PromedioTotal(imagen_1,imagen_2,imagen_3)

```
function promedio = PromedioTotal(imagen_1,imagen_2,imagen_3)
    % Cojo la imagen_1
    imagen_contrastada = MejoraDeColor(imagen_1);
    imagen_segmentada = Segmentacion(imagen_contrastada);
    prototipo_1 = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada);

    imagen_contrastada = MejoraDeColor(imagen_2);
    imagen_segmentada = Segmentacion(imagen_contrastada);
    prototipo_2 = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada);

    imagen_contrastada = MejoraDeColor(imagen_3);
    imagen_segmentada = Segmentacion(imagen_contrastada);
    prototipo_3 = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada);

    matriz = [prototipo_1;prototipo_2;prototipo_3];

    promedio = mean(matriz);

end
```

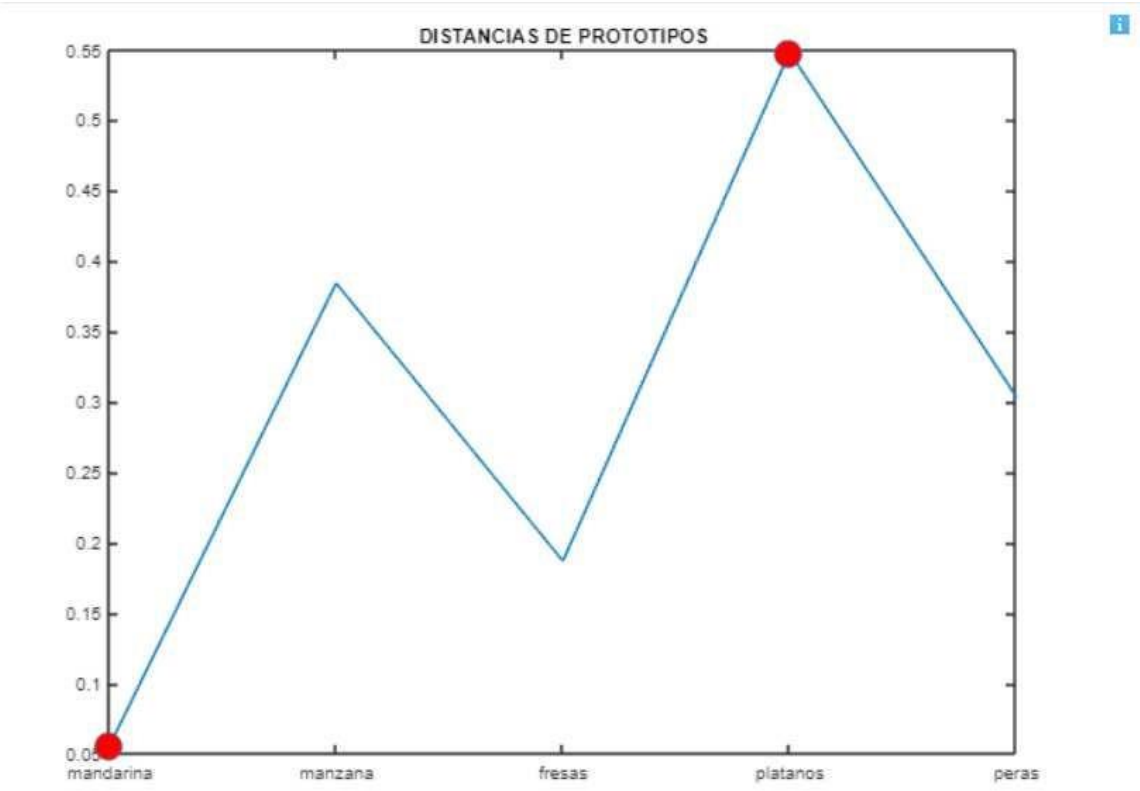
- '**promedio = PromedioTotal(imagen_1, imagen_2, imagen_3)**': Esta función lo que hace es coger tres imágenes de cada fruta y le implementa una mejora de color, ya explicada en la tarea 1, y una segmentación, ya explicada en la tarea 2, y con estas imágenes le extrae las características creando un vector de características, ya explicado en la tarea 3. Con estos vectores de las tres fotos se crea una matriz 3x3 y se le calcula la media.

·Tarea 5

Lo que hay que hacer en esta tarea es crear una función que nos permita, introduciendo una imagen, identificar que fruta es.

T5: Diseño de una función de reconocimiento

```
imagen_entrada = imread('F12.jpg');
tipo_fruta = ReconocerFruta(imagen_entrada,prototipo_mandarinas,prototipo_manzanas,prototipo_fresas,prototipo_platanos,prototipo_peras)
```




```
La fruta a la que más se parece es:Mandarina
La fruta a la que menos se parece es:Platano
ES UNA MANDARINA
tipo_fruta = 2
```

Nuestra solución fue cargar la imagen que queramos identificar y llamar ala función **‘tipo_fruta = ReconocerFruta(imagen_entrada, prototipo_mandarinas, prototipo_manzanas, prototipo_fresas, prototipo_platanos, prototipo_peras)’** donde la imagen_entrada es la imagen que nosotros cargamos.

- tipo_fruta = ReconocerFruta(imagen_entrada, prototipo_mandarinas, prototipo_manzanas, prototipo_fresas, prototipo_platanos, prototipo_peras)

```
function tipo_fruta = ReconocerFruta(imagen_entrada,prototipo_mandarinas,prototipo_manzanas,prototipo_fresas,prototipo_platanos,prototipo_peras)
    % prototipos = promedios totales
    imagen_contrastada = MejoraDeColor(imagen_entrada);
    imagen_segmentada = Segmentacion(imagen_contrastada);
    caracteristicas_entrada = ExtraccionCaracteristica(imagen_contrastada, imagen_segmentada);

    % Cálculo de la distancia entre dos vectores = módulo(X-Y) --> Distancia de X a Y
    distancia_mandarinas = norm(prototipo_mandarinas-caracteristicas_entrada);  % Calculo el módulo del vector (la norma)
    distancia_manzanas = norm(prototipo_manzanas-caracteristicas_entrada);
    distancia_fresas = norm(prototipo_fresas-caracteristicas_entrada);
    distancia_platanos = norm(prototipo_platanos-caracteristicas_entrada);
    distancia_peras = norm(prototipo_peras-caracteristicas_entrada);

    distancias = [distancia_mandarinas distancia_manzanas distancia_fresas distancia_platanos distancia_peras];
    % La distancia mí
    minimo = min (distancias);
    grafico(distancias);

    if minimo == distancia_manzanas
        disp('ES UNA MANZANA');
        tipo_fruta = 1;

    elseif  minimo == distancia_mandarinas
        disp('ES UNA MANDARINA');
       tipo_fruta = 2;

    elseif  minimo == distancia_fresas
        disp('ES UNA FRESA');
        tipo_fruta = 3;

    elseif  minimo == distancia_platanos
        disp('ES UN PLATANO');
        tipo_fruta = 4;

    else minimo == distancia_peras
        disp('ES UNA PERA');
        tipo_fruta = 5;

    end

end
```

▪ function grafico(distancias)

```
function grafico(distancias)
    X = distancias;
    minimo = min (X);
    maximo = max (X);
    idxmin = find(X == minimo);
    idxmax = find(X == maximo);
    plot(X, '-o', 'MarkerIndices', [idxmin idxmax], 'MarkerFaceColor','red', 'MarkerSize',10)
    xticks([1 2 3 4 5])
    xticklabels({'mandarina', 'manzana', 'fresas', 'platanos', 'peras'})
    title("DISTANCIAS DE PROTOTIPOS");
    nombres{1} = 'Mandarina';
    nombres{2} = 'Manzana';
    nombres{3} = 'Fresa';
    nombres{4} = 'Platano';
    nombres{5} = 'Pera';
    disp(strcat('La fruta a la que más se parece es: ', nombres{idxmin}));
    disp(strcat('La fruta a la que menos se parece es: ', nombres{idxmax}));
end
```

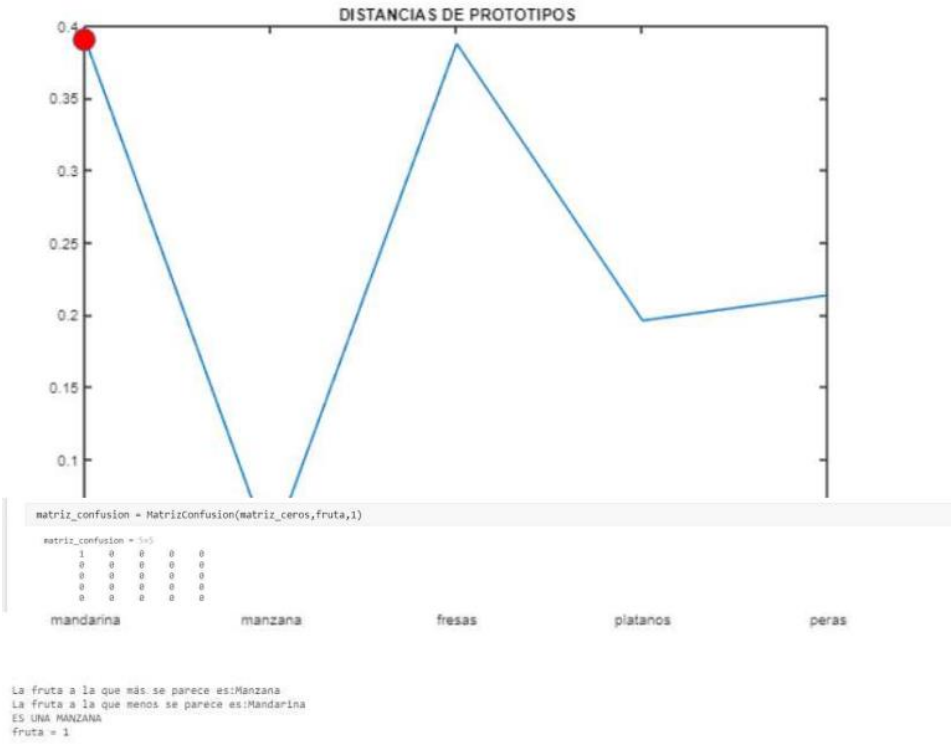
- **‘tipo_fruta = ReconocerFruta(imagen_entrada, prototipo_mandarinas, prototipo_manzanas, prototipo_fresas, prototipo_platanos, prototipo_peras)’**: En esta función se vuelve a sacar el vector de características como lo hicimos en la tarea anterior, y luego calculamos la distancia entre dos vectores como son el de una fruta y el de la imagen queacabamos de calcular, esto se hace mediante el cálculo del módulo de la resta de ambos vectores. Repetimos este proceso con todas las frutas y las incluimos en un vector del cual cogemos el valor ms pequeño, y este es elvalor de la fruta a la que pertenece la imagen.
- En esta función también llamamos a otra función llamada ‘**grafico(distancias)**’, y esta lo único que hace es calcular las distancias máximas y mínimas y representar todas las distancias en una gráfica paraasí decir cuál es la fruta a la que más se parece y a la que menos.

·Tarea 6

En esta tarea se crea una matriz en la que indicamos que fruta reconoce y que fruta es. Es decir, comprueba si la tarea 5 funciona de manera correcta,y para confirmarlo solo puede haber número en la diagonal principal de lamatriz.

T6: Prueba

```
matriz_ceros = zeros(5); %la dimensión es la clase de frutas que hay
imagen_prueba = imread('F04.jpg');
fruta = ReconocerFruta(imagen_prueba,prototipo_mandarinas,prototipo_manzanas,prototipo_fresas,prototipo_platanos,prototipo_peras)
```



Para ello, llamamos a cada imagen, la introducimos en la función de la tarea 5 para reconocer que fruta es y posteriormente llamamos a la función'**matriz_confusion = MatrizConfusion(matriz_ceros,fruta,n)**'.

Tarea 6

- **matriz_confusion = MatrizConfusion(matriz_ceros,fruta,n)**

```
function matriz_confusion = MatrizConfusion(matriz_ceros,fruta,n)

        if fruta == 1
        matriz_ceros(n,1) = matriz_ceros(n,1)+ 1;

        elseif fruta == 2
             matriz_ceros(n,2) = matriz_ceros(n,2) + 1;

        elseif fruta == 3
             matriz_ceros(n,3) = matriz_ceros(n,3) + 1;

        elseif fruta == 4
             matriz_ceros(n,4) = matriz_ceros(n,4) + 1;

        else fruta == 5
             matriz_ceros(n,5) =  matriz_ceros(n,5) + 1;

        end

        matriz_confusion = matriz_ceros;
end
```

- '**matriz_confusion = MatrizConfusion(matriz_ceros,fruta,n)**': La única utilidad de esta es sumarle 1 en la matriz dependiendo de la fruta que reconozca la otra función. El parámetro fruta es la fruta que reconoce el programa y n es la fruta que es en realidad. Las columnas son la fruta que

reconoce y las filas la fruta real. En nuestro proyecto los valores que leasignamos a n son:

- n = 1 (manzanas)
- n = 2 (mandarinas)
- n = 3 (fresas)
- n = 4 (plátanos)
- n = 5 (peras)

·Reflexión final

Con este proyecto hemos aprendido a manejar distintas funciones enmatlab para analizar imágenes y modificarlas de distintas maneras.
Gracias a estas podremos reconocer distintos tipos de objetos, en este casofrutas, gracias a unos parámetros como sus componentes.