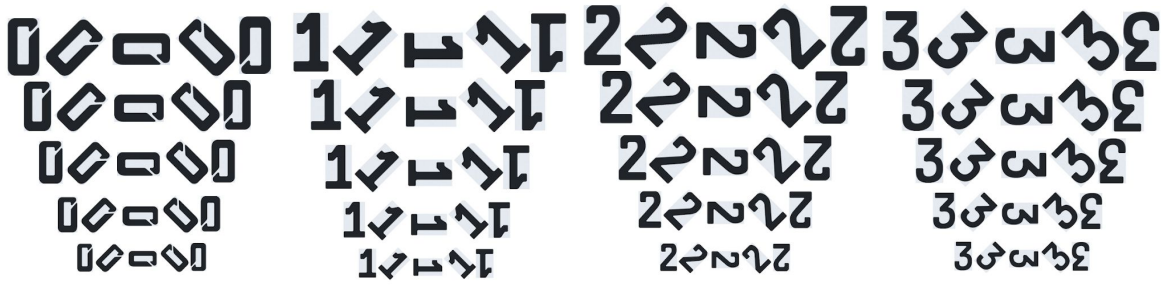
Sistemas de percepción. Ejercicio voluntario 3. Memoria

Se ha realizado la versión básica del ejercicio. Se implementa un discriminador por mínima distancia, usando vectores de características de 4 dimensiones (Momentos de Hu del 1 al 4).

1. **Creación de imágenes artificiales para entrenamiento.**

Utilizando la web https://www.customeuropeanplates.com/, se han creado las siguientes imágenes:



Las utilizamos para obtener los momentos de Hu de cada uno de los dígitos, en distintas posiciones, orientaciones y tamaños. Así nos aseguramos de poder identificar los números independientemente de cómo se encuentren en la imagen.

2. **Cálculo de momentos.**

Los momentos de Hu se calculan de la siguiente manera.
Primero se carga la imagen y se procesa, para quedarnos solamente con el contorno de los dígitos.

```
function [Hu, centroids] = moments2(file)
    img = imread(file);
    [M,N,C] = size(img);
    img_bin = zeros(M,N);

%Depuramos la imagen por colores para dejar solo numeros y bordes
%negros
    for i = 1:M-1
```

```
        for j = 1:N-1
            if(img(i,j,1)>=25 && img(i,j,1)<=50 &&...
            img(i,j,2)>=30 && img(i,j,2)<=40 &&...
            img(i,j,3)>=30 && img(i,j,3)<=55)
                img_bin(i,j) = 1;
            else
                img_bin(i,j) = 0;
            end
        end
    end

    %Abrimos y cerramos para eliminar el ruido
    se = strel('disk',1);
    img_bin = imopen(img_bin,se);
    se = strel('disk',2);
    img_bin = imclose(img_bin,se);
```

Etiquetamos la imagen y nos quedamos con los centroides de cada región etiquetada. Para cada etiqueta, creamos una subimagen que solo la contenga a ella y calculamos su momento de orden 0 y 1, los momentos centrales, momentos centrales normalizados, y finalmente momentos de Hu.

```
    %Imagen binaria etiquetada
    img_etiq = bwlabel(img_bin);

    centroids = regionprops(img_etiq, 'Centroid');

    %Numero de regiones en la imagen
    n = max(max(img_etiq));

    phi1 = [];
    phi2 = [];
    phi3 = [];
    phi4 = [];
    phi5 = [];
    phi6 = [];
    phi7 = [];

    for i = 1:n
        img_aux = (img_etiq==i);
```

```
        %Momentos de orden 0
        m00 = sum(sum(img_aux));

        %Momentos de orden 1 (Para obtencion de centro de masas)
        m01 = 0; m10 = 0;
        for x = 1:N
            for y = 1:M
                m10 = m10 + x*double(img_aux(y,x));
                m01 = m01 + y*double(img_aux(y,x));
            end
        end
        xCM = m10/m00;
        yCM = m01/m00;

%Momentos centrales (Para los momentos centrales normalizados)
        mu00 = m00;
        mu11 = 0;
        mu20 = 0; mu02 = 0;
        mu30 = 0; mu03 = 0;
        mu12 = 0; mu21 = 0;
        for x = 1:N
            for y = 1:M
                mu11 = mu11 +
(x-xCM)*(y-yCM)*double(img_aux(y,x));
                mu20 = mu20 + ((x-xCM)^2)*double(img_aux(y,x));
                mu02 = mu02 + ((y-yCM)^2)*double(img_aux(y,x));
                mu30 = mu30 + ((x-xCM)^3)*double(img_aux(y,x));
                mu03 = mu03 + ((y-yCM)^3)*double(img_aux(y,x));
                mu12 = mu12 +
(x-xCM)*((y-yCM)^2)*double(img_aux(y,x));
                mu21 = mu21 +
((x-xCM)^2)*(y-yCM)*double(img_aux(y,x));
            end
        end

        %Momentos centrales normalizados (Para los momentos de Hu)
        eta11 = mu11/(mu00^2);
        eta20 = mu20/(mu00^2);
        eta02 = mu02/(mu00^2);
        eta30 = mu30/(mu00^(2.5));
        eta03 = mu03/(mu00^(2.5));
```

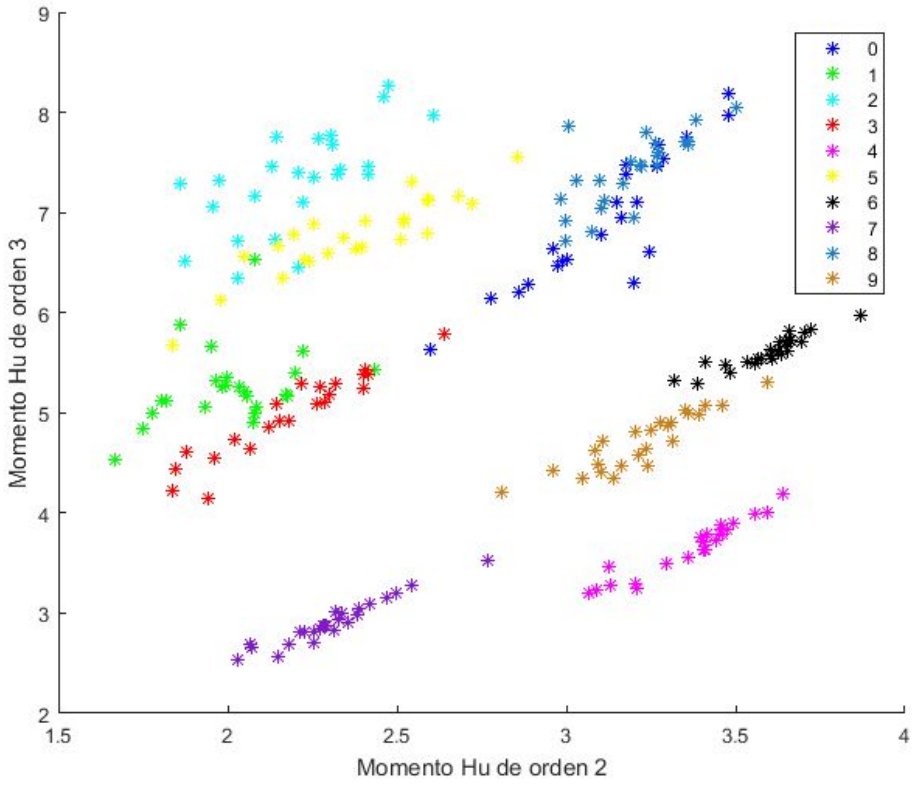
```
        eta12 = mu12/(mu00^(2.5));
        eta21 = mu21/(mu00^(2.5));

        %Momentos de Hu
        phi1 = [phi1 (eta20+eta02)];
        phi2 = [phi2 ((eta20-eta02)^2 + 4*eta11^2)];
        phi3 = [phi3 ((eta30-3*eta12)^2 + (3*eta21-eta03)^2)];
        phi4 = [phi4 ((eta30+eta12)^2 + (eta21+eta03)^2)];
        phi5 = [phi5
((eta30-3*eta12)*(eta30+eta12)*((eta30+eta12)^2-3*(eta21+eta03)^2)
+(3*eta21-eta03)*(eta21+eta03)*(3*(eta30+eta12)^2-(eta21+eta03)^2)
)];
        phi6 = [phi6
((eta20-eta02)*((eta30+eta12)^2-(eta21+eta03)^2)+4*eta11*(eta30+et
a12)*(eta21+eta03))];
        phi7 = [phi7
((3*eta21-eta03)*(eta30+eta12)*((eta30+eta12)^2-3*(eta21+eta03)^2)
-(eta30-3*eta12)*(eta21+eta03)*(3*(eta30+eta12)^2-(eta21+eta03)^2)
)];
```

Finalmente y con el objetivo de que todos los momentos tengan el mismo orden de magnitud (para que todos influyan de manera similar al calcular distancias), agrupamos y reescalamos los valores.

```
Hu = [phi1; phi2; phi3; phi4];
Hu = -sign(Hu).*log10(abs(Hu));
```

3. Visualización de los momentos.

Una vez tenemos una nube de puntos para cada dígito, podemos plotearlos para verificar que se distinguen bien. Aquí vemos representados los momentos 2 y 3:



4. Determinación de prototipos

Para clasificar una nueva región, calcularemos sus momentos de Hu y los agrupamos en un vector de características. Posteriormente podríamos calcular las distancias de dicho vector de características a cada uno de los vectores que hemos obtenido en el entrenamiento. Para evitar esto, creamos un prototipo (media) para cada clase y solo tendremos que calcular las 10 distancias a los 10 prototipos.

```
%% Entrenamiento
%Calculamos los momentos de Hu de orden 1, 2, 3, 4
%Hu_ord1_, Hu_ord2_, Hu_ord3_, Hu_ord4_
%Calculamos los 4 primeros momentos de Hu para cada dígito
[Hu0,~] = moments2('Imagenes/0.png');
[Hu1,~] = moments2('Imagenes/1.png');
[Hu2,~] = moments2('Imagenes/2.png');
[Hu3,~] = moments2('Imagenes/3.png');
[Hu4,~] = moments2('Imagenes/4.png');
[Hu5,~] = moments2('Imagenes/5.png');
[Hu6,~] = moments2('Imagenes/6.png');
[Hu7,~] = moments2('Imagenes/7.png');
[Hu8,~] = moments2('Imagenes/8.png');
```

```
[Hu9,~] = moments2('Imagenes/9.png');

%% Agrupamos en vectores de caracteristicas
Hu_ord1_ = [Hu0(1,:); Hu1(1,:); Hu2(1,:); Hu3(1,:); Hu4(1,:);...
            Hu5(1,:); Hu6(1,:); Hu7(1,:); Hu8(1,:); Hu9(1,:)];

Hu_ord2_ = [Hu0(2,:); Hu1(2,:); Hu2(2,:); Hu3(2,:); Hu4(2,:);...
            Hu5(2,:); Hu6(2,:); Hu7(2,:); Hu8(2,:); Hu9(2,:)];

Hu_ord3_ = [Hu0(3,:); Hu1(3,:); Hu2(3,:); Hu3(3,:); Hu4(3,:);...
            Hu5(3,:); Hu6(3,:); Hu7(3,:); Hu8(3,:); Hu9(3,:)];

Hu_ord4_ = [Hu0(4,:); Hu1(4,:); Hu2(4,:); Hu3(4,:); Hu4(4,:);...
            Hu5(4,:); Hu6(4,:); Hu7(4,:); Hu8(4,:); Hu9(4,:)];

%% Prototipos
%Calculamos el prototipo de cada clase.
%z0, z1, ..., z8, z9
Hu_ord1_media = mean(Hu_ord1_,2);
Hu_ord2_media = mean(Hu_ord2_,2);
Hu_ord3_media = mean(Hu_ord3_,2);
Hu_ord4_media = mean(Hu_ord4_,2);

z0 = [Hu_ord1_media(1); Hu_ord2_media(1); Hu_ord3_media(1);
Hu_ord4_media(1)];
z1 = [Hu_ord1_media(2); Hu_ord2_media(2); Hu_ord3_media(2);
Hu_ord4_media(2)];
z2 = [Hu_ord1_media(3); Hu_ord2_media(3); Hu_ord3_media(3);
Hu_ord4_media(3)];
z3 = [Hu_ord1_media(4); Hu_ord2_media(4); Hu_ord3_media(4);
Hu_ord4_media(4)];
z4 = [Hu_ord1_media(5); Hu_ord2_media(5); Hu_ord3_media(5);
Hu_ord4_media(5)];
z5 = [Hu_ord1_media(6); Hu_ord2_media(6); Hu_ord3_media(6);
Hu_ord4_media(6)];
z6 = [Hu_ord1_media(7); Hu_ord2_media(7); Hu_ord3_media(7);
Hu_ord4_media(7)];
z7 = [Hu_ord1_media(8); Hu_ord2_media(8); Hu_ord3_media(8);
Hu_ord4_media(8)];
z8 = [Hu_ord1_media(9); Hu_ord2_media(9); Hu_ord3_media(9);
Hu_ord4_media(9)];
```

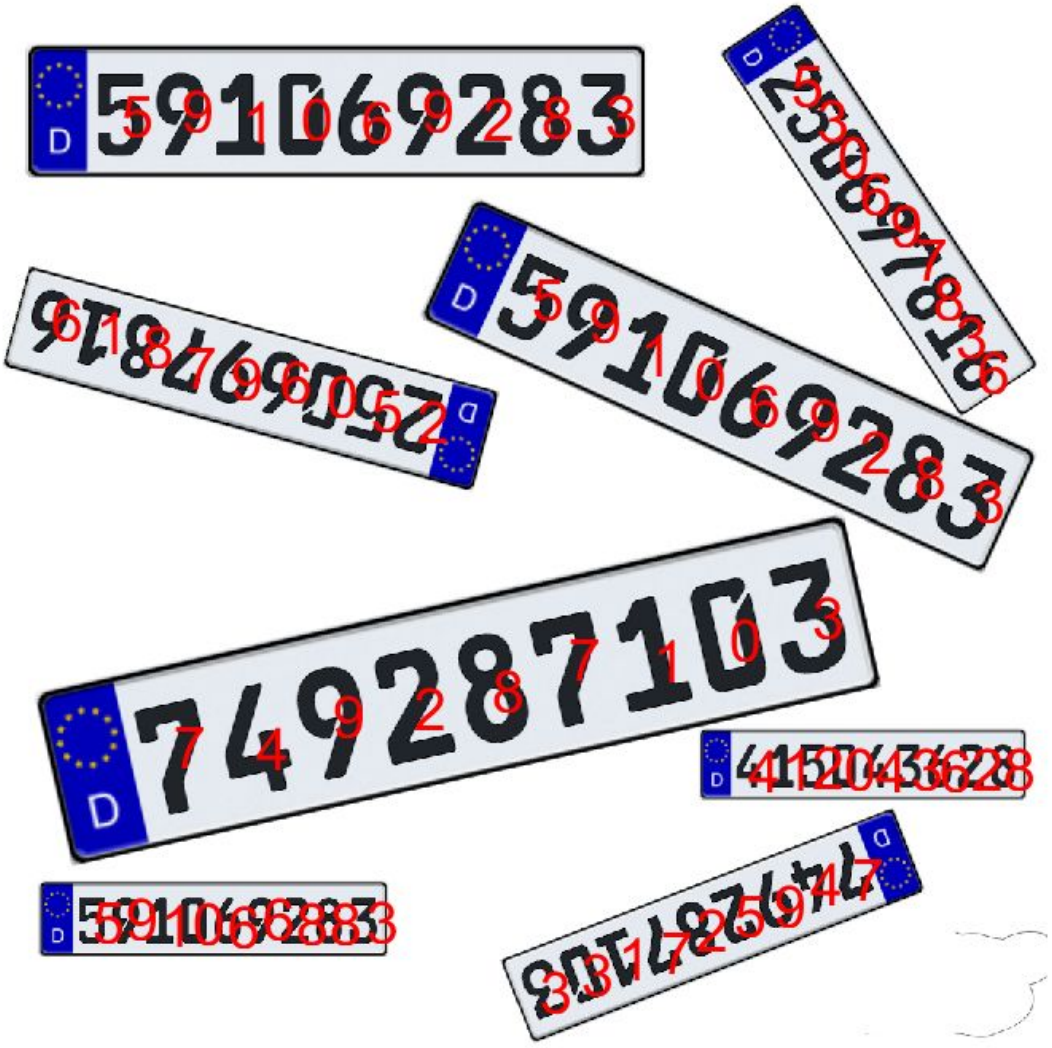
```
z9 = [Hu_ord1_media(10); Hu_ord2_media(10); Hu_ord3_media(10);
Hu_ord4_media(10)];
```

5. **Clasificación.**

```
%% Clasificacion
%Para clasificar la imagen, primero obtenemos los momentos de Hu
%de cada dígito que contenga
[X, centroids] = moments2('Imagenes/Matriculas.png');
%Calculamos las distancias a cada prototipo
dists = [sqrt(sum((X-z0).^2));...
         sqrt(sum((X-z1).^2));...
         sqrt(sum((X-z2).^2));...
         sqrt(sum((X-z3).^2));...
         sqrt(sum((X-z4).^2));...
         sqrt(sum((X-z5).^2));...
         sqrt(sum((X-z6).^2));...
         sqrt(sum((X-z7).^2));...
         sqrt(sum((X-z8).^2));...
         sqrt(sum((X-z9).^2))];

 %Obtenemos la distancia mínima y en que posición está
 [C,I] = min(dists);

 %Mostramos los resultados
 img = imread('Imagenes/Matriculas.png');
 figure()
 imshow(img);
 hold on
 for i = 1:length(I)

text(centroids(i).Centroid(1),centroids(i).Centroid(2),int2str(I(i
)-1),'FontSize',28,'Color','red');

 end
 hold off
```

6. **Resultados.**

La siguiente imagen muestra los resultados obtenidos.



No se ha conseguido una clasificación perfecta de todos los dígitos. Con una matriz de verificación podemos comprobar como de acertado es nuestro algoritmo. La probabilidad de acierto global es del 87,3%.

real/clasificado	0	1	2	3	4	5	6	7	8	9
0	7/8			1/8						
1		7/8		1/8						
2			5/8			2/8			1/8	
3				6/6						

4					3/3					
5			1/6	1/6		4/6				
6							8/8			
7								6/6		
8			1/8						7/8	
9							1/10			9/10