Interpolacion aplicada a la transformación de imágenes

Contenidos

Transformaciones lineales

Una aplicación de la interpolación es la transformación *lineal* también llamada *rígida*. Si (i,j) es un pixel de la imagen y $I(i,j)$ su intensidad, una transformación rígida es una transformación lineal de las coordenadas de los pixeles, es decir

$$\begin{pmatrix} p \\ q \end{pmatrix} = A \begin{pmatrix} i \\ j \end{pmatrix}$$

donde A es una matriz 2×2.

Entre las transformaciones lineales se encuentran el grupo de transformaciones afines, como escalado, rotación y cizallado. Veamos unos ejemplos con Matlab. Para ello, definimos una matriz 3×3 de la forma

$$\begin{pmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

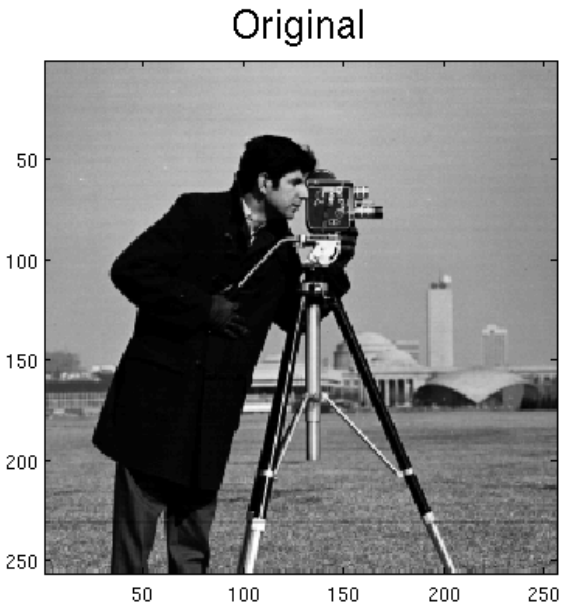
y usamos las órdenens `maketform` y `imtransform` con las matrices respectivas:

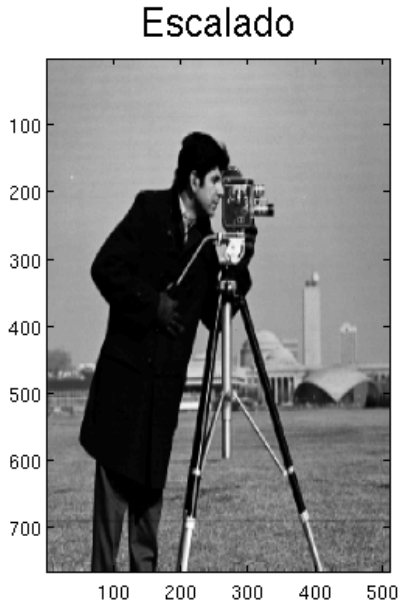
$$\text{Escalado}: \begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \text{Cizallado}: \begin{pmatrix} 1 & sh_1 & 0 \\ sh_2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \text{Rotación}: \begin{pmatrix} \cos(\theta) & \text{sen}(\theta) & 0 \\ -\text{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

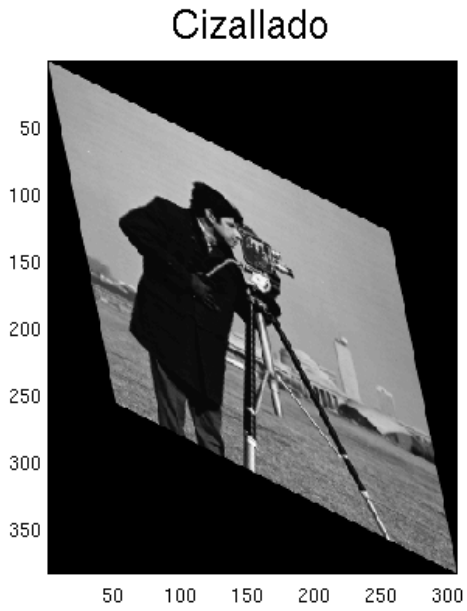
donde s_1 y s_2 son los factores de escala según los ejes X e Y respectivamente, sh_1 y sh_2 son los factores de cizalla paralela a los ejes X e Y respectivamente y donde θ es el ángulo a rotar.

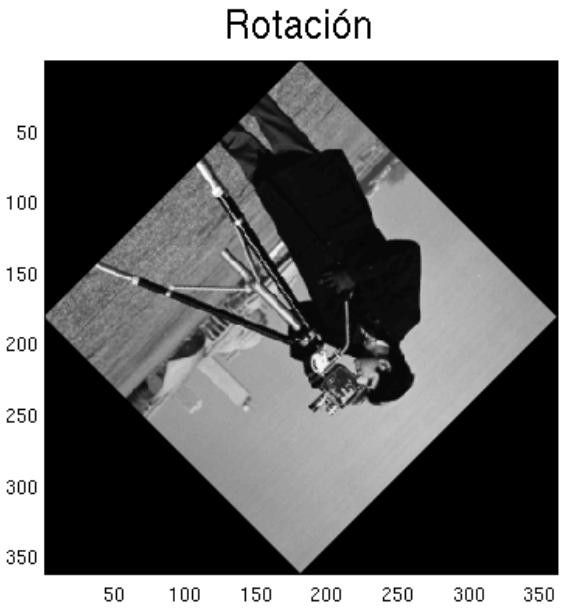
```
I=imread('cameraman.tif');
I=im2double(I);

s=[2,3];
tform1 = maketform('affine',[s(1) 0 0; 0 s(2) 0; 0 0 1]);       % Escalado
I1 = imtransform(I,tform1);

sh=[0.5 0.2];
tform2 = maketform('affine',[1 sh(1) 0; sh(2) 1 0; 0 0 1]);     % Cizallado
I2 = imtransform(I,tform2);

theta=3*pi/4;                                                   % Rotación
A=[cos(theta) sin(theta) 0; -sin(theta) cos(theta) 0; 0 0 1];
tform3 = maketform('affine',A);
I3 = imtransform(I,tform3);

figure
imagesc(I),axis image
title('Original','FontSize',18)
colormap(gray)
figure
imagesc(I1),axis image
title('Escalado','FontSize',18)
colormap(gray)
```

```
figure
imagesc(I2),axis image
title('Cizallado','FontSize',18)
colormap(gray)
figure
imagesc(I3),axis image
title('Rotación','FontSize',18)
colormap(gray)
```









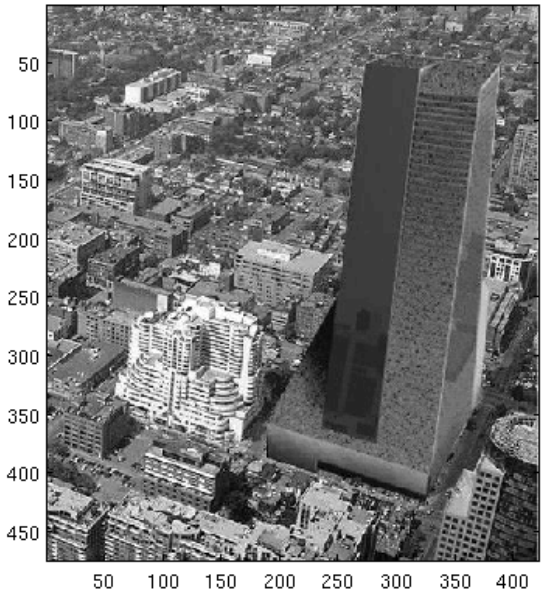
Ejercicio 1 Queremos enderezar la torre de la fotografía tower_bw.jpg. Para ello la tenemos que rotar un ángulo adecuado. Pero una vez rotada, aparece una región negra que no estaba definida. Por lo tanto tenemos que recortar la figura para eliminar esta región.

Escribir un programa que realice estas tareas. Mostrar la imagen inicial, la imagen final y el ángulo rotado (en grados).

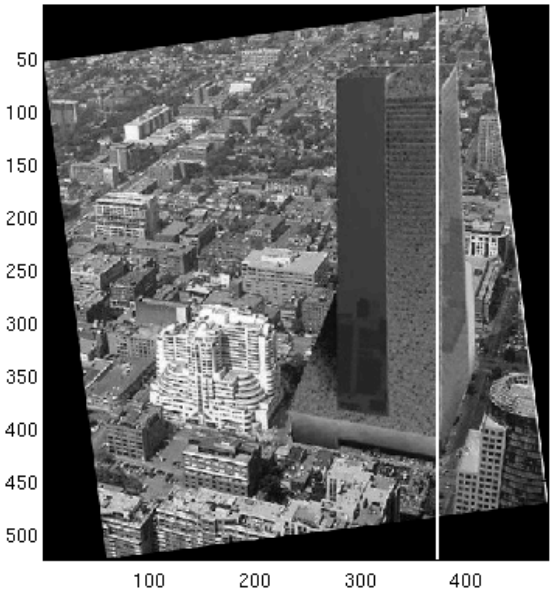
Nota: Usar una línea vertical para decidir que la torre está enderezada. Por ejemplo, si la imagen rotada es I, utilizar $I(:, 9:12) = 1$ para dibujar una línea blanca vertical que tenga de ancho de la columna 9 a la 12.

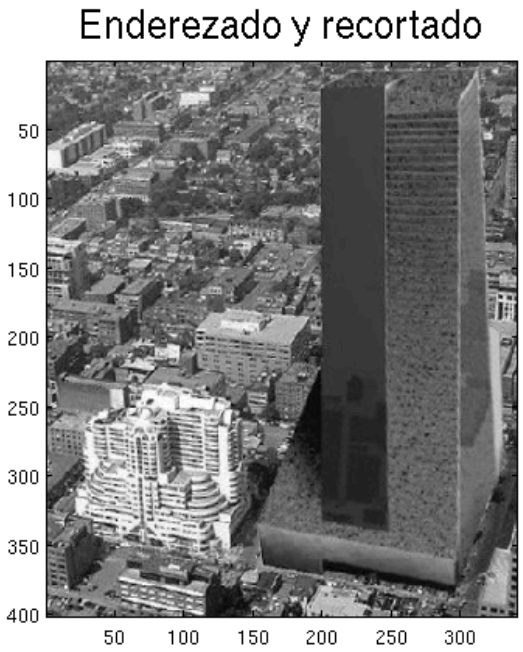
```
Ejercicio1
```

Original



Enderezado sin recortar





Transformaciones no lineales

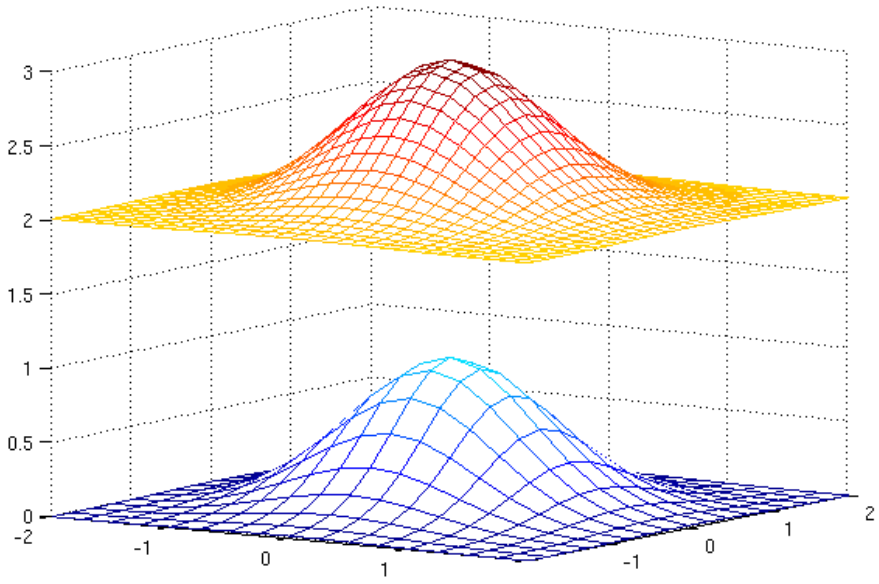
En este caso, en la transformación

$$\begin{pmatrix} p \\ q \end{pmatrix} = A \begin{pmatrix} i \\ j \end{pmatrix}$$

la matriz A no tiene que ser la misma para todos los pixeles sino que puede depender del pixel sobre el que está actuando.

En estas transformaciones se suele usar el comando de Matlab `interp2` que interpola funciones en dos dimensiones. Por ejemplo, definimos la función $f(x,y) = e^{-x^2-y^2}$ en una rejilla gruesa y luego hallamos los valores en los puntos intermedios por interpolación y representamos el resultado:

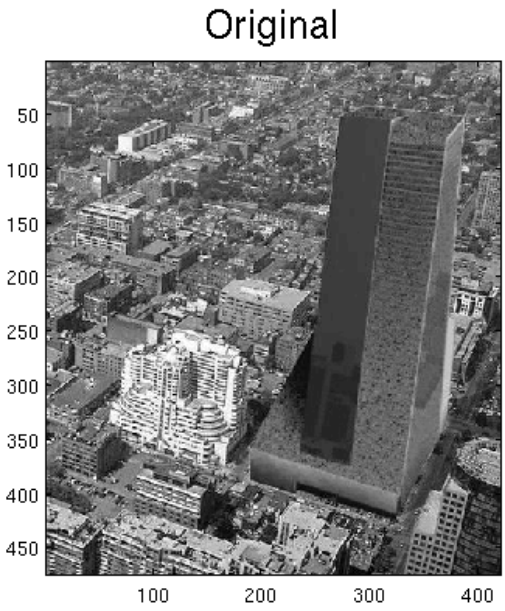
```
clear
[x,y] = meshgrid(-2:0.25:2);            % rejilla gruesa
z = exp(-x.^2-y.^2);                    % para cada punto (x,y) definimos el z de acuerdo con la función
[p,q] = meshgrid(-2:0.125:2);           % rejilla fina
zfinal = interp2(x,y,z,p,q);            % interpolamos los puntos intermedios
mesh(x,y,z), hold on                    % dibujamos la superficie utilizando una malla gruesa
mesh(p,q,zfinal+2)                      % dibujamos la superficie utilizando una malla más fina
colormap(jet)
view([34,10])
```

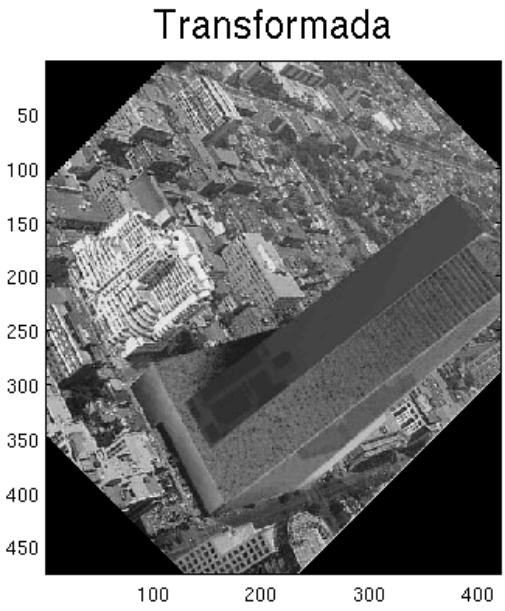


La orden `interp2` puede usarse también para transformaciones afines. Por ejemplo, podemos si rotamos los píxeles de una imagen alrededor de un punto (x_0, y_0) un ángulo θ, para cada pixel realizaremos la transformación:

$$\begin{pmatrix} p \\ q \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \text{sen}(\theta) \\ -\text{sen}(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$$

```
clear
I=imread('tower_bw.jpg');
I=im2double(I);

[m,n]= size(I);
[x,y] = meshgrid(1:n,1:m);                  % rejilla heredada de la imagen original

theta=pi/4;                                 % ángulo de rotación
x0=fix(m/2);y0=fix(n/2);                    % centro de rotación
p=(x-x0)*cos(theta)+(y-y0)*sin(theta)+x0;   % coordenadas transformadas (nueva posición de los pixeles)
q=-(x-x0)*sin(theta)+(y-y0)*cos(theta)+y0;

Ifinal=interp2(x,y,I,p,q,'bicubic');        % valores de la intensidad interpolada en la nueva posición

figure
imagesc(I),axis image
title('Original','FontSize',18)
colormap(gray)
figure
imagesc(Ifinal),axis image
title('Transformada','FontSize',18)
colormap(gray)
```

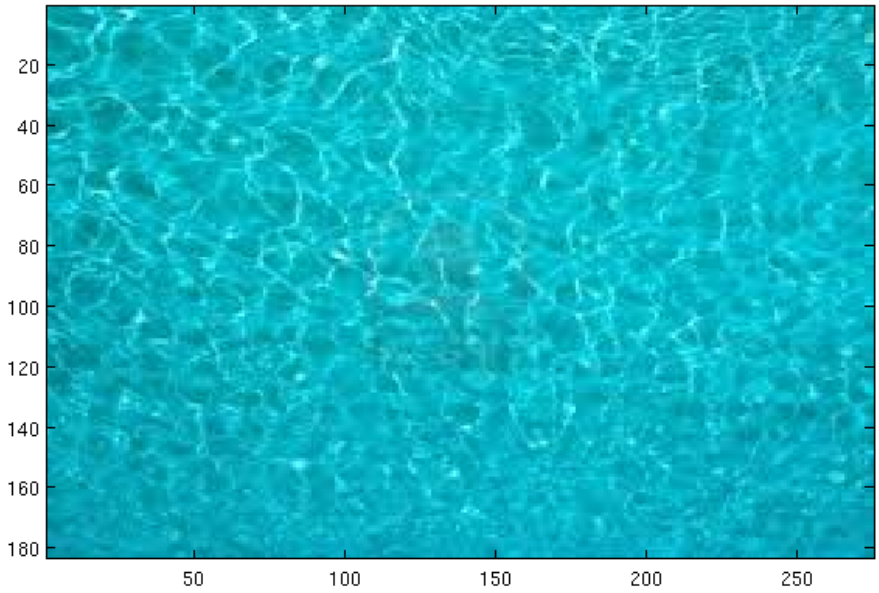




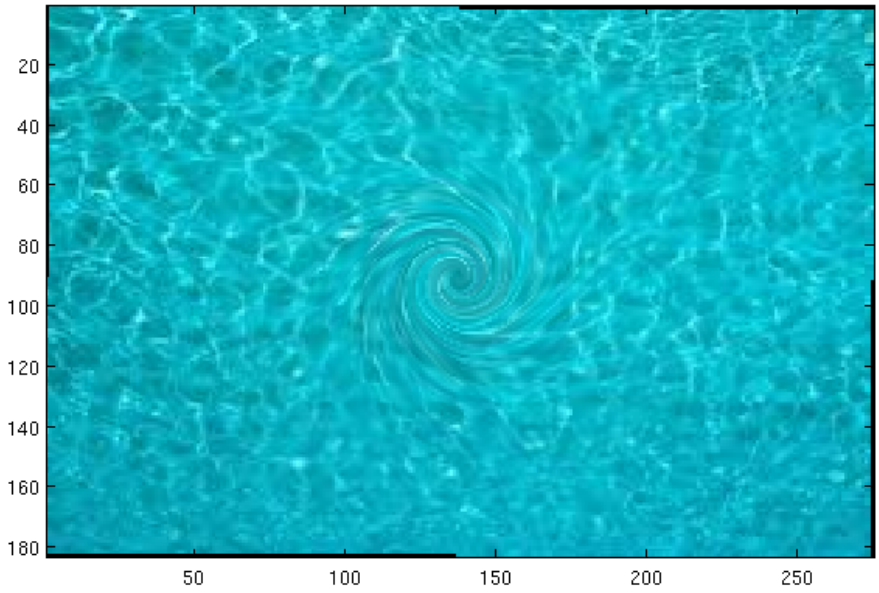
Ejercicio 2 El objetivo de este ejercicio es introuducir un remolino en la piscina, tal como muestra la figura (resultado final del ejercicio)

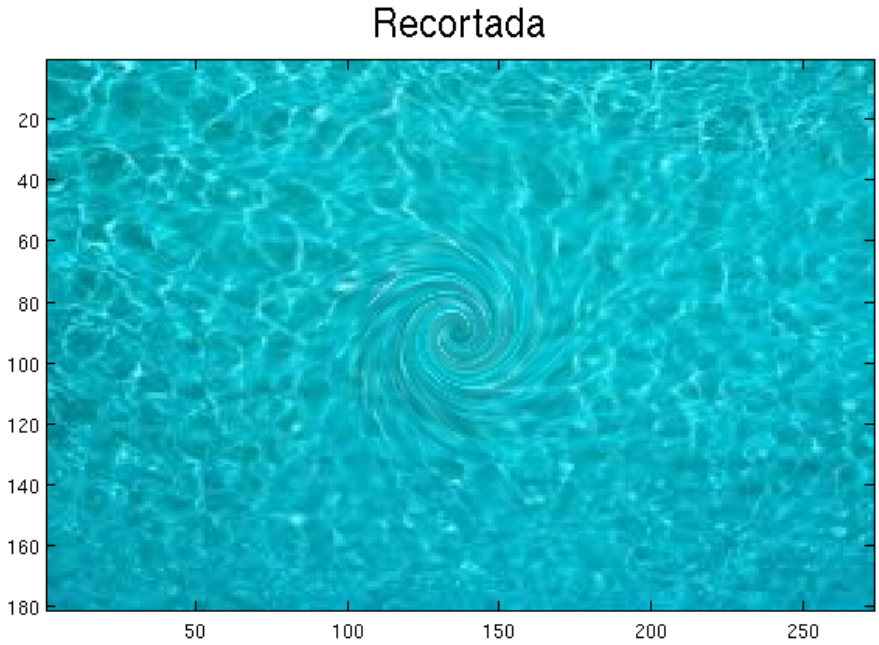
```
Ejercicio2
```

Original



Transformada





Para realizar este ejercicio, usar la rotación del programa anterior e introducir los siguientes cambios:

- Definir una función en línea $s = @(x, y)...$ que represente la circunferencia de radio 1 centrada en (x_0, y_0).
- Definir el ángulo como $\theta = 10 * \exp(-0.1 * s(x, y))$.
- Definir la posición nueva de cada pixel dependiendo de este ángulo, como en el programa anterior.
- Recortar las líneas negras que aparecen en los bordes.

Si la figura fuera en blanco y negro y tuviéramos sólo una intensidad, usaríamos `interp2` como en el programa anterior. Pero como ahora nuestra figura es una imagen rgb, tenemos que realizar la interpolación para cada canal. Lo podemos hacer con un bucle:

```
%Ifinal=zeros(size(Ic));                % inicializa la matriz final a ceros
%for k=1:3                              % tres colores
    %selecciona el canal
    %interpola y genera Ichannel;
    %Ifinal(:,:,k)=Ichannel;            % añade este canal a la matriz de tres capas
%end
```

Ejercicio 3 El objetivo de este ejercicio es escalar la imagen de forma no uniforme. También se busca automatizar el recortado final.

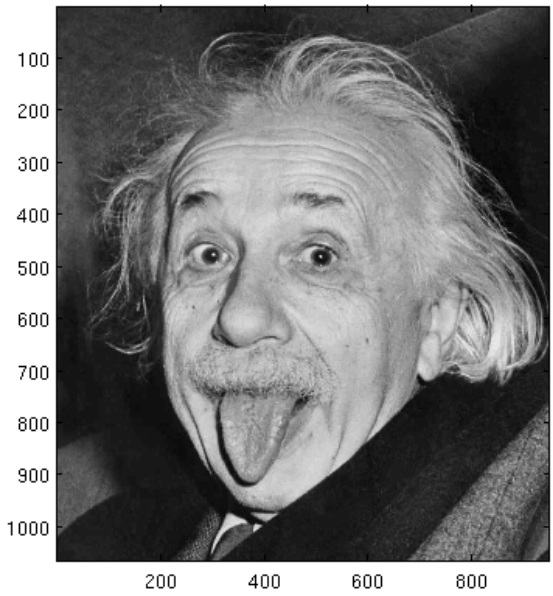
En este caso definimos la transformación como:

- $d = 0.01$,
- $p = d * s(x, y). * (x - x0) + x$ siendo s(x,y) la función definida para el problema anterior.
- $q = d * s(x, y). * (y - y0) + y$.

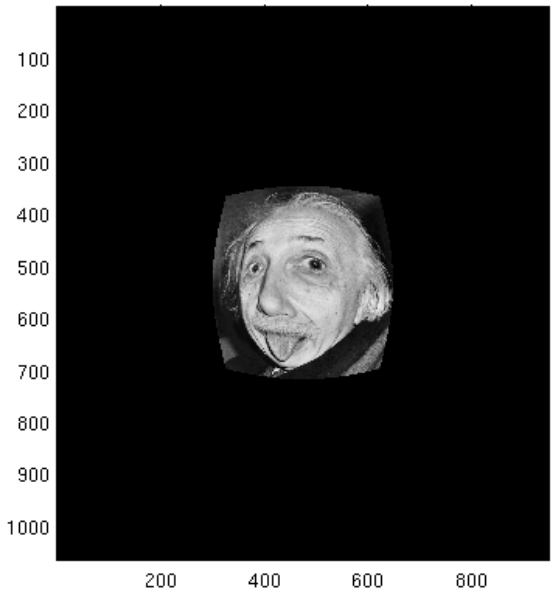
Después de la interpolación, obtenemos una figura pequeña rodeada de una gran zona donde la intensidad no está definida. Pero el resultado final debería ser:

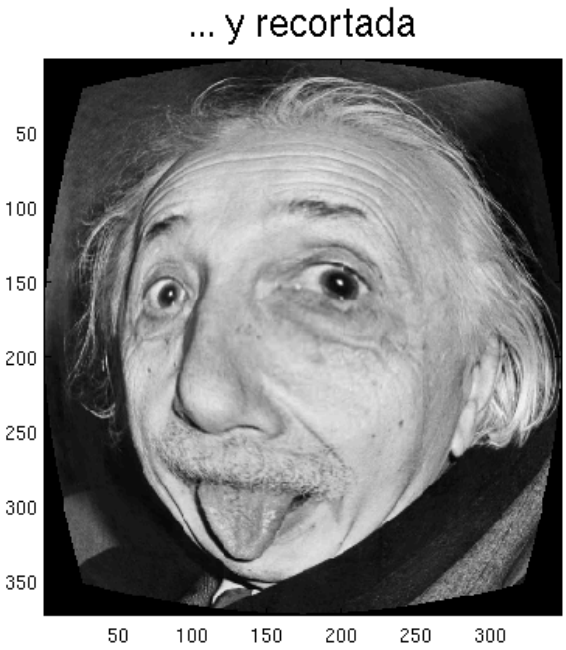
```
Ejercicio3
```

Original



modificada



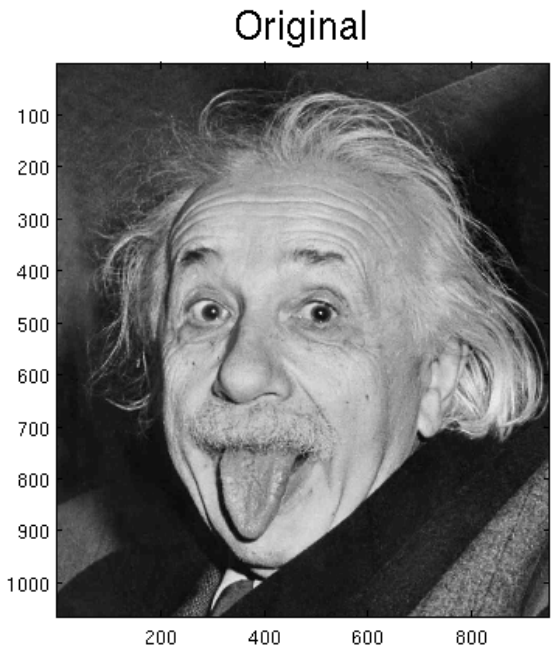


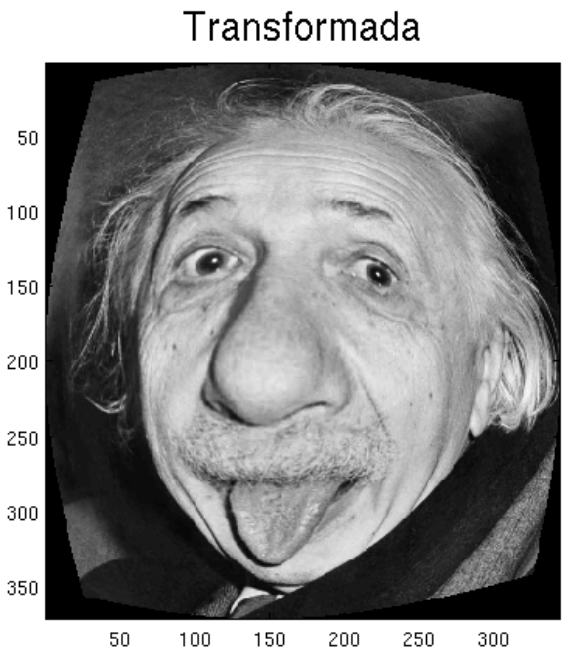
Usando la función `find`, encontrar los pixeles donde la intensidad interpolada es ≥ 0 y definir otra imagen donde solo aparezcan estos píxeles.

Ejercicio 4 Modificar el programa anterior para convertirlo en una función que tenga como argumentos:

- Entrada: una imagen en blanco y negro, el punto central (x_0, y_0), y un parámetro de deformación d.
- Salida: La matriz de la imagen transformada y recortada.

```
I=imread('einstein_bw.jpg');
A=Ejercicio4(I,361,593,0.01);
```





Ficheros:

Cameraman

Rascacielos

Einstein

Agua

Published with MATLAB® 7.14