Crear imágenes con transparencia en MATLAB

Para hacer una imagen png como se verá a continuación, se requiere saber unas cosas.

Se sabe que una imagen a color tiene 3 canales RGB, y cada uno de estos canales es una matriz del tamaño de la imagen, pero cuando hablamos de transparencia, hablamos de que la imagen se le agrega un canal extra que representará la transparencia, entonces tendremos 4 canales y ya no solamente 3 como convencionalmente tiene una imagen a color.

"Una imagen con transparencia es un conjunto de 4 matrices del mismo tamaño".

Teniendo esto en cuenta el codigo tendra mas sentido, pues para agregar el canal de transparencia, necesitamos generar una matriz del tamaño de la imagen con valor 0 a las regiones que queremos sean transparentes.

```
% Leemos la imagen y la mostramos
n = imread('patitos.jpg');
h = imshow(n)
```

```
h =
  Image with properties:

         CData: [853×1280×3 uint8]
  CDataMapping: 'direct'

  Show all properties
```

```
% Mostramos la informacion del pixel por el que pasa el mouse,
% en la parte inferior izquierda
impixelinfo(h)

% El comando gitinput es para obtener las coordenadas del mouse
% cuando hace clic, estas coordenadas se almacenan en 2 variables
% x,y, el fin de esto es hacer un rectángulo/cuadrado que donde el
% área será transparente (Orden va de izquierda a derecha)
[x,y] = ginput(2)
```


```
x = 2×1
   75.0000
  565.0000
y = 2×1
  177.0000
  612.0000
```

```
% Redondeamos cada elemento de x,y al entero más cercano, para
% tener áreas de pixeles enteros
x = fix(x)
```

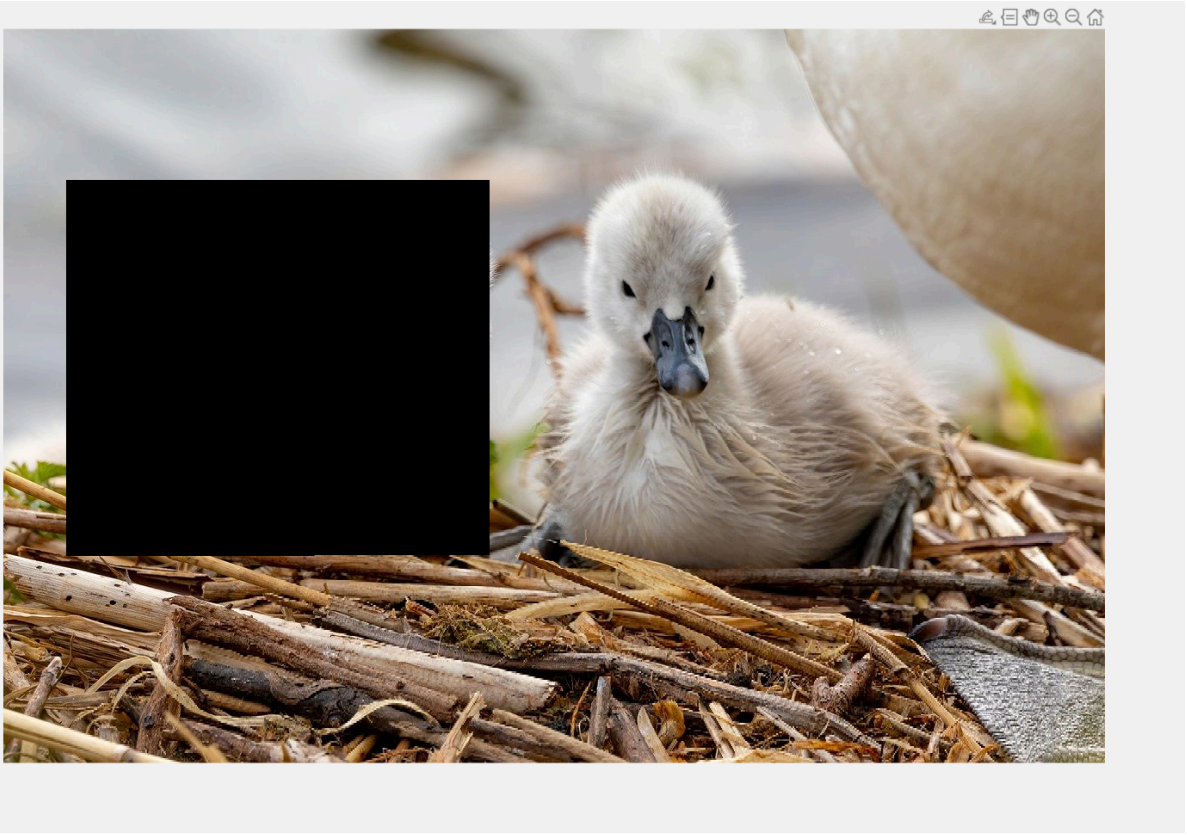
```
x = 2×1
    74
   565
```

```
y = fix(y)
```

```
y = 2×1
   176
   612
```

```
%Creamos el área del rectángulo del cual se creará la transparencia
cor = n(y(1):y(2),x(1):x(2));

% Del tamaño del rectangulo creamos una matriz de ceros que seran el
% 4to canal
z = zeros(size(cor));

% Creamos una imagen de respaldo para no editar la imagen original
m=n;
```

```
% Se separa la imagen de los patitos en 3 matrices que corresponden
% al RGB de cada canal
r = m(:,:,1);
g = m(:,:,2);
b = m(:,:,3);

% A cada canal se hace cero el rectángulo que seleccionamos usando las
% coordenadas del ginput, este paso es necesario, si hacemos ceros a la
% imagen original no funcionará, primero tienen que ser a cada una de sus
% componentes RGB y de ahí se unen en una sola matriz
r(y(1):y(2),x(1):x(2))=0;
g(y(1):y(2),x(1):x(2))=0;
b(y(1):y(2),x(1):x(2))=0;

% Se unen los canales en una sola matriz, así como antes se separo para
% aplicar el proceso a cada canal, se deben de juntar
M(:,:,1) = r;
M(:,:,2) = g;
M(:,:,3) = b;

% Condicionales para aplicar la transparencia, A2 es que todos los pixeles,
% se pueden jugar con estos parámetros para obtener diferentes resultados
A1 = double(r==1);
A2 = double(r>1);
A3 = double(r<1);

% La variable M se guarda con el nombre PNGlisto.png, y el canal alpha
% se le agrega el valor de la variable condicional A2
imwrite(M,'PNGlisto.png','Alpha', A2);
M2 = M;

% Mostramos la imagen
imshow(M)
```



```
cla
%Muestro la imagen original y la transparente
subplot(1,2,1), imshow(n)
title("Imagen original")
subplot(1,2,2), imshow(M)
title("Imagen png con transparencia")
```

Imagen original

Imagen png con transparencia

Cuando la imagen se abre en un programa de edición de imagen como Photoshop, Krita, entre otros ya se podrá apreciar la transparencia.

Este programa es solo un ejemplo de cómo se puede implementar la transparencia en este programa, no es la única forma, también el área del rectángulo se puede adaptar para hacer un círculo, o un área amorfa, pero eso ya necesita otra lógica más compleja.

Un desperfecto es que todo píxel negro se considera como transparente, y este error quedara arreglado en futuros ejemplo.

Fuente de la foto: https://pixabay.com/es/photos/cygnets-cisnes-nido-aves-6233080/