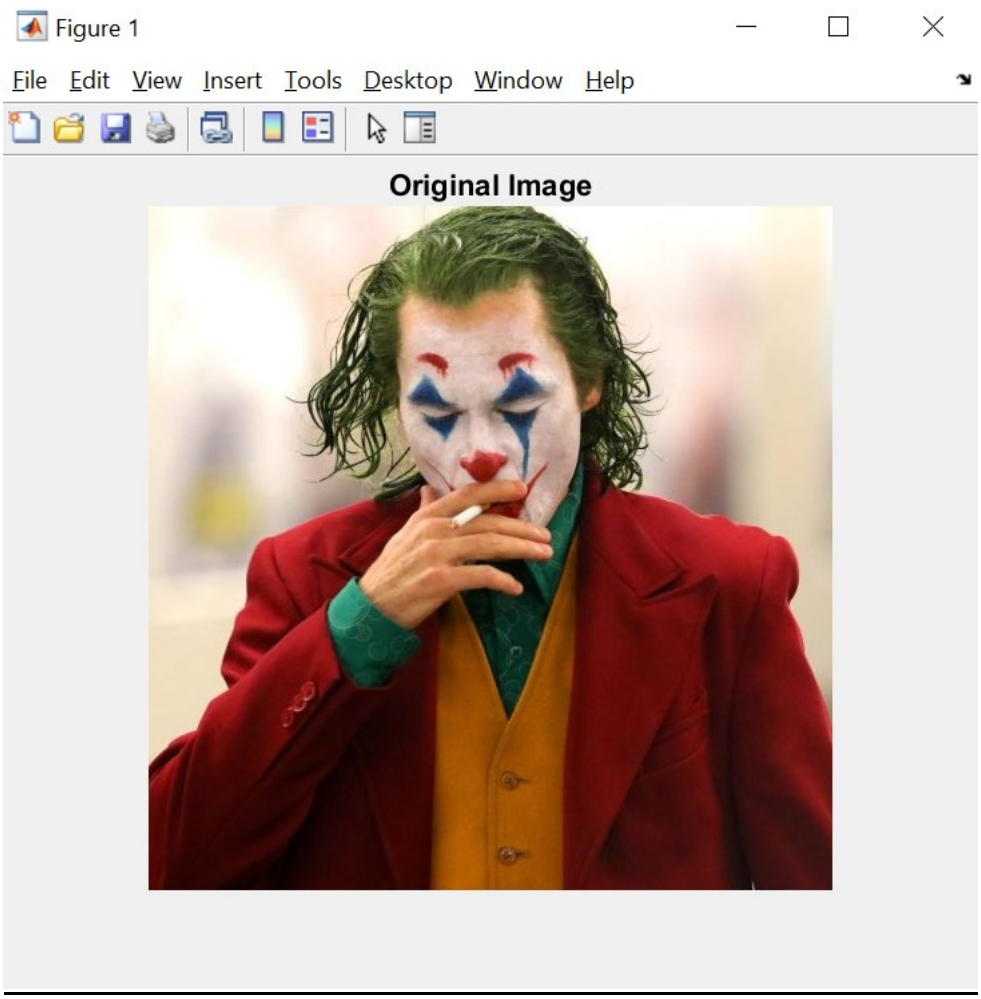
CMPE 362 - SIGNAL PROCESSING

HOMEWORK 3

REPORT

IMAGE PROCESSING



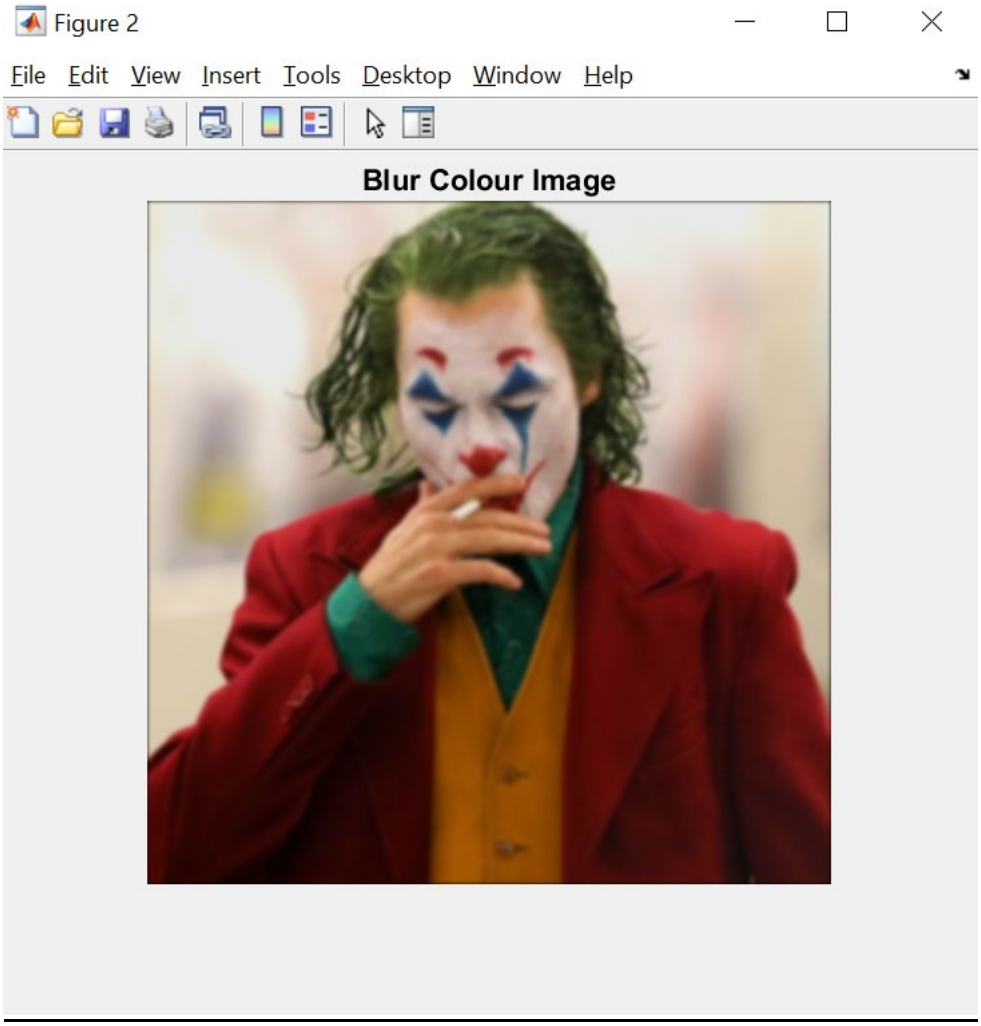
In this homework, I learned to use convolution filtering on images. I split the joker image into its 3 RGB channels. Then, I convolve the image with a 2D function and plot the results.

PART A - Design a kernel that adds blur to your image

```
%% BLUR IMAGE
img=imread('jokerimage.png');
% Extract the individual red, green, and blue color channels.
redChannel = img(:, :, 1);
greenChannel = img(:, :, 2);
blueChannel = img(:, :, 3);
```

```
imshow(img),title('Original Image');

%adds blur to redChannel

I = double(redChannel);
sigma = 1.76; %Standard Deviation
sz = 3; %Box size
[x,y] = meshgrid(-sz:sz,-sz:sz);
M = size(x,1)-1;
N = size(y,1)-1;
%Gaussian
Exp_comp=-(x.^2+y.^2)/(2*sigma*sigma);
Kernel=exp(Exp_comp)/(2*pi*sigma*sigma);
BluredImageRed=zeros(size(I));
I = padarray(I,[sz sz]);
%Convolution
for i=1:size(I,1)-M
    for j=1:size(I,2)-N
        Temp=I(i:i+M,j:j+M).*Kernel;
        BluredImageRed(i,j)=sum(Temp(:));
    end
end
%convert that array to an array of type uint
BluredImageRed=uint8(BluredImageRed);


%adds blur to greenChannel

I = double(greenChannel);
sigma = 1.76; %Standard Deviation
sz = 3; %Box size
[x,y] = meshgrid(-sz:sz,-sz:sz);
M = size(x,1)-1;
N = size(y,1)-1;
%Gaussian
Exp_comp=-(x.^2+y.^2)/(2*sigma*sigma);
Kernel=exp(Exp_comp)/(2*pi*sigma*sigma);
BluredImageGreen=zeros(size(I));
I = padarray(I,[sz sz]);
%Convolution
for i=1:size(I,1)-M
    for j=1:size(I,2)-N
        Temp=I(i:i+M,j:j+M).*Kernel;
        BluredImageGreen(i,j)=sum(Temp(:));
    end
end

BluredImageGreen=uint8(BluredImageGreen);

%adds blur to blueChannel

I = double(blueChannel);
sigma = 1.76; %Standard Deviation
sz = 3; %Box size
[x,y] = meshgrid(-sz:sz,-sz:sz);
```

```
M = size(x,1)-1;
N = size(y,1)-1;
%Gaussian
Exp_comp=-(x.^2+y.^2)/(2*sigma*sigma);
Kernel=exp(Exp_comp)/(2*pi*sigma*sigma);
BluredImageBlue=zeros(size(I));
I = padarray(I,[sz sz]);
%Convolution
for i=1:size(I,1)-M
    for j=1:size(I,2)-N
        Temp=I(i:i+M,j:j+M).*Kernel;
        BluredImageBlue(i,j)=sum(Temp(:));
    end
end
%convert that array to an array of type uint8
BluredImageBlue=uint8(BluredImageBlue);

%Convert gray image to color image

rgbImage = cat(3, BluredImageRed, BluredImageGreen,
BluredImageBlue);

figure,imshow(rgbImage),title('Blur Colour Image');
```



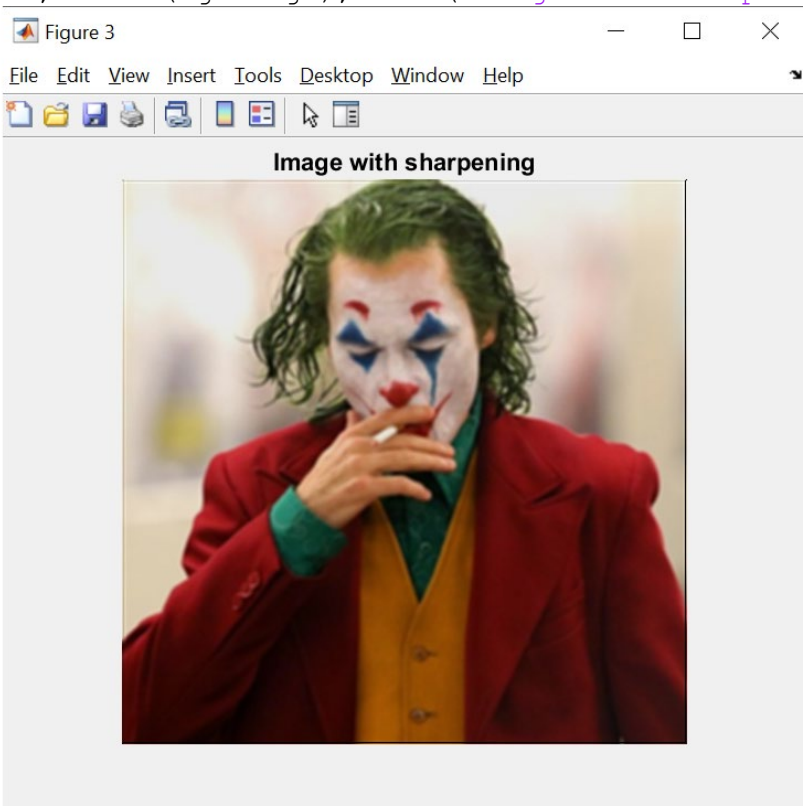
PART B - Design a kernel that sharpens your image to get rid of the blur

```
%% SHARPENING IMAGE
% Extract the individual red, green, and blue color channels.
redChannel = rgbImage(:, :, 1);
greenChannel = rgbImage(:, :, 2);
blueChannel = rgbImage(:, :, 3);

%Red channel

mask = [0 -1 0 ; -1 5 -1 ; 0 -1 0];
[rSize, cSize] = size(mask);

[nrow, ncol] = size(redChannel);
redChannel = cast(redChannel, 'double');
newImgRed = zeros(nrow,ncol);

% Corners
subMask = mask(2:3,2:3);
i = 1;
j = 1;
subMat = redChannel(i:i+1,j:j+1);
newImgRed(i,j) = sum(sum(subMask.*subMat));

subMask = mask(1:2,1:2);
i = nrow;
j = ncol;
subMat = redChannel(i-1:i,j-1:j);
newImgRed(i,j) = sum(sum(subMask.*subMat));

% Edges
subMask = mask(2:3,1:3);
i = 1;
for j = 2:ncol-1
    subMat = redChannel(i:i+1,j-1:j+1);
    newImgRed(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:2,1:3);
i = nrow;
for j = 2:ncol-1
    subMat = redChannel(i-1:i,j-1:j+1);
    newImgRed(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:3,2:3);
j = 1;
for i = 2:nrow-1
    subMat = redChannel(i-1:i+1,j:j+1);
    newImgRed(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:3,1:2);
j = ncol;
for(i = 2:nrow-1)
```

```
    subMat = redChannel(i-1:i+1,j-1:j);
    newImgRed(i,j) = sum(sum(subMask.*subMat));
end

% Inner
for i = 0.5*(rSize+1) : nrow - 0.5*(rSize+1)
    for j = 0.5*(cSize+1) : ncol - 0.5*(cSize+1)
        subMat = redChannel(i-1:i+1,j-1:j+1);
        newImgRed(i,j) = sum(sum(subMat.*mask));
    end
end
%convert that array to an array of type uint
newImgRed = cast(newImgRed, 'uint8');


%Green channel

mask = [0 -1 0 ; -1 5 -1 ; 0 -1 0];
[rSize, cSize] = size(mask);

[nrow, ncol] = size(greenChannel);
greenChannel = cast(greenChannel, 'double');
newImgGreen = zeros(nrow,ncol);

% Corners
subMask = mask(2:3,2:3);
i = 1;
j = 1;
subMat = greenChannel(i:i+1,j:j+1);
newImgGreen(i,j) = sum(sum(subMask.*subMat));

subMask = mask(1:2,1:2);
i = nrow;
j = ncol;
subMat = greenChannel(i-1:i,j-1:j);
newImgGreen(i,j) = sum(sum(subMask.*subMat));


% Edges
subMask = mask(2:3,1:3);
i = 1;
for j = 2:ncol-1
    subMat = greenChannel(i:i+1,j-1:j+1);
    newImgGreen(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:2,1:3);
i = nrow;
for j = 2:ncol-1
    subMat = greenChannel(i-1:i,j-1:j+1);
    newImgGreen(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:3,2:3);
j = 1;
for i = 2:nrow-1
```

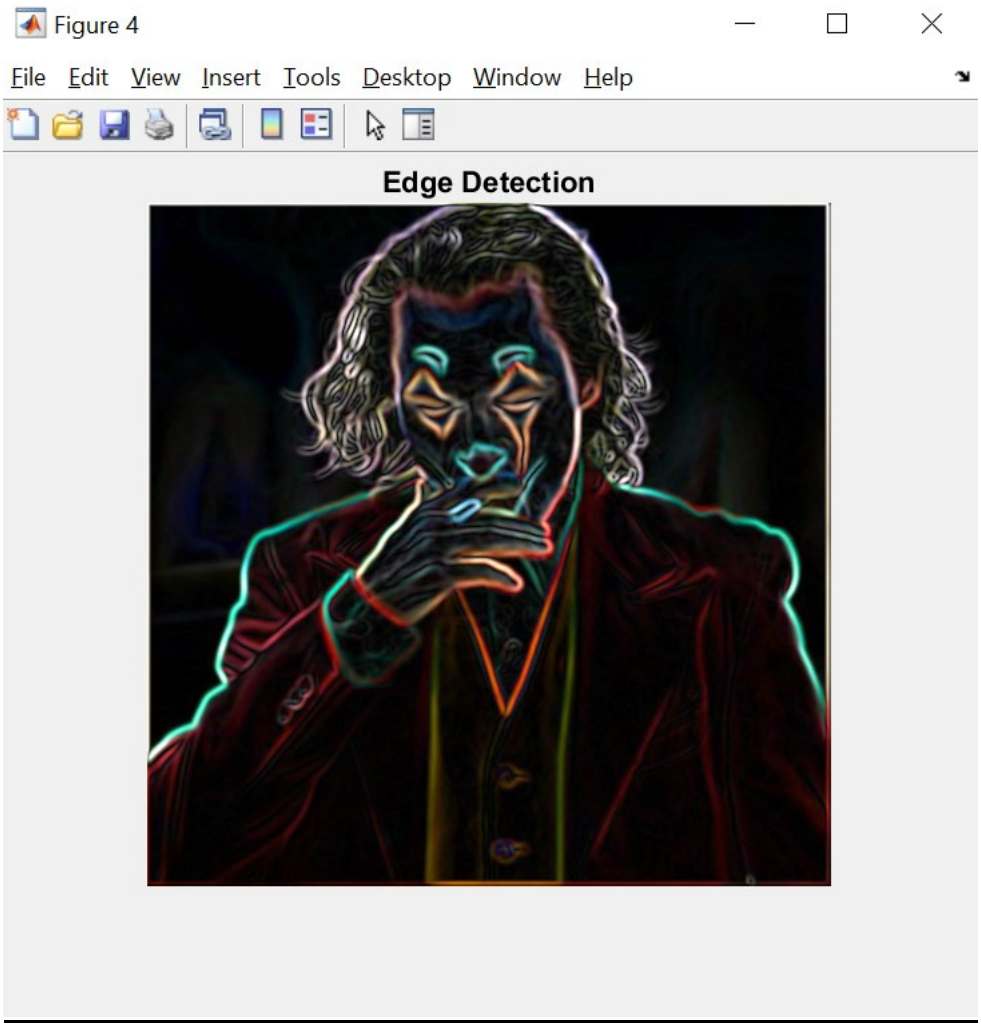
```
    subMat = greenChannel(i-1:i+1,j:j+1);
    newImgGreen(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:3,1:2);
j = ncol;
for(i = 2:nrow-1)
    subMat = greenChannel(i-1:i+1,j-1:j);
    newImgGreen(i,j) = sum(sum(subMask.*subMat));
end

% Inner
for i = 0.5*(rSize+1) : nrow - 0.5*(rSize+1)
    for j = 0.5*(cSize+1) : ncol - 0.5*(cSize+1)
        subMat = greenChannel(i-1:i+1,j-1:j+1);
        newImgGreen(i,j) = sum(sum(subMat.*mask));
    end
end
%convert that array to an array of type uint
newImgGreen = cast(newImgGreen, 'uint8');

%Blue channel

mask = [0 -1 0 ; -1 5 -1 ; 0 -1 0];
[rSize, cSize] = size(mask);

[nrow, ncol] = size(blueChannel);
blueChannel = cast(blueChannel, 'double');
newImgBlue = zeros(nrow,ncol);

% Corners
subMask = mask(2:3,2:3);
i = 1;
j = 1;
subMat = blueChannel(i:i+1,j:j+1);
newImgBlue(i,j) = sum(sum(subMask.*subMat));

subMask = mask(1:2,1:2);
i = nrow;
j = ncol;
subMat = blueChannel(i-1:i,j-1:j);
newImgBlue(i,j) = sum(sum(subMask.*subMat));


% Edges
subMask = mask(2:3,1:3);
i = 1;
for j = 2:ncol-1
    subMat = blueChannel(i:i+1,j-1:j+1);
    newImgBlue(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:2,1:3);
i = nrow;
for j = 2:ncol-1
    subMat = blueChannel(i-1:i,j-1:j+1);
```

```
    newImgBlue(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:3,2:3);
j = 1;
for i = 2:nrow-1
    subMat = blueChannel(i-1:i+1,j:j+1);
    newImgBlue(i,j) = sum(sum(subMask.*subMat));
end

subMask = mask(1:3,1:2);
j = ncol;
for(i = 2:nrow-1)
    subMat = blueChannel(i-1:i+1,j-1:j);
    newImgBlue(i,j) = sum(sum(subMask.*subMat));
end

% Inner
for i = 0.5*(rSize+1) : nrow - 0.5*(rSize+1)
    for j = 0.5*(cSize+1) : ncol - 0.5*(cSize+1)
        subMat = blueChannel(i-1:i+1,j-1:j+1);
        newImgBlue(i,j) = sum(sum(subMat.*mask));
    end
end
%convert that array to an array of type uint
newImgBlue = cast(newImgBlue, 'uint8');

rgbImage = cat(3, newImgRed, newImgGreen, newImgBlue);

% Display the sharpened color image.
figure,imshow(rgbImage),title('Image with sharpening');
```



PART C - Design a kernel that highlights edges in your image

```
%% EDGE DETECTION
C=double(redChannel);
for i=1:size(C,1)-2
for j=1:size(C,2)-2
%Sobel mask for x-direction
Gx=((2*C(i+2,j+1)+C(i+2,j)+C(i+2,j+2))-
(2*C(i,j+1)+C(i,j)+C(i,j+2)));
%Sobel mask for y-direction
Gy=((2*C(i+1,j+2)+C(i,j+2)+C(i+2,j+2))-
(2*C(i+1,j)+C(i,j)+C(i+2,j)));
%The gradient of the image
redChannel(i,j)=sqrt(Gx.^2+Gy.^2);
end
end

C=double(greenChannel);
for i=1:size(C,1)-2
for j=1:size(C,2)-2
%Sobel mask for x-direction
Gx=((2*C(i+2,j+1)+C(i+2,j)+C(i+2,j+2))-
(2*C(i,j+1)+C(i,j)+C(i,j+2)));
%Sobel mask for y-direction
Gy=((2*C(i+1,j+2)+C(i,j+2)+C(i+2,j+2))-
(2*C(i+1,j)+C(i,j)+C(i+2,j)));
%The gradient of the image
greenChannel(i,j)=sqrt(Gx.^2+Gy.^2);
end
end


C=double(blueChannel);
for i=1:size(C,1)-2
for j=1:size(C,2)-2
%Sobel mask for x-direction
Gx=((2*C(i+2,j+1)+C(i+2,j)+C(i+2,j+2))-
(2*C(i,j+1)+C(i,j)+C(i,j+2)));
%Sobel mask for y-direction
Gy=((2*C(i+1,j+2)+C(i,j+2)+C(i+2,j+2))-
(2*C(i+1,j)+C(i,j)+C(i+2,j)));
%The gradient of the image
blueChannel(i,j)=sqrt(Gx.^2+Gy.^2);
end
end


% Recombine separate color channels into a single, true color RGB
image.
rgbImage2 = cat(3, uint8(redChannel), uint8(greenChannel),
uint8(blueChannel));
% Display the blurred color image.
figure,imshow(rgbImage2),title('Edge Detection');
```



PART D – Design a kernel that makes your image embossed

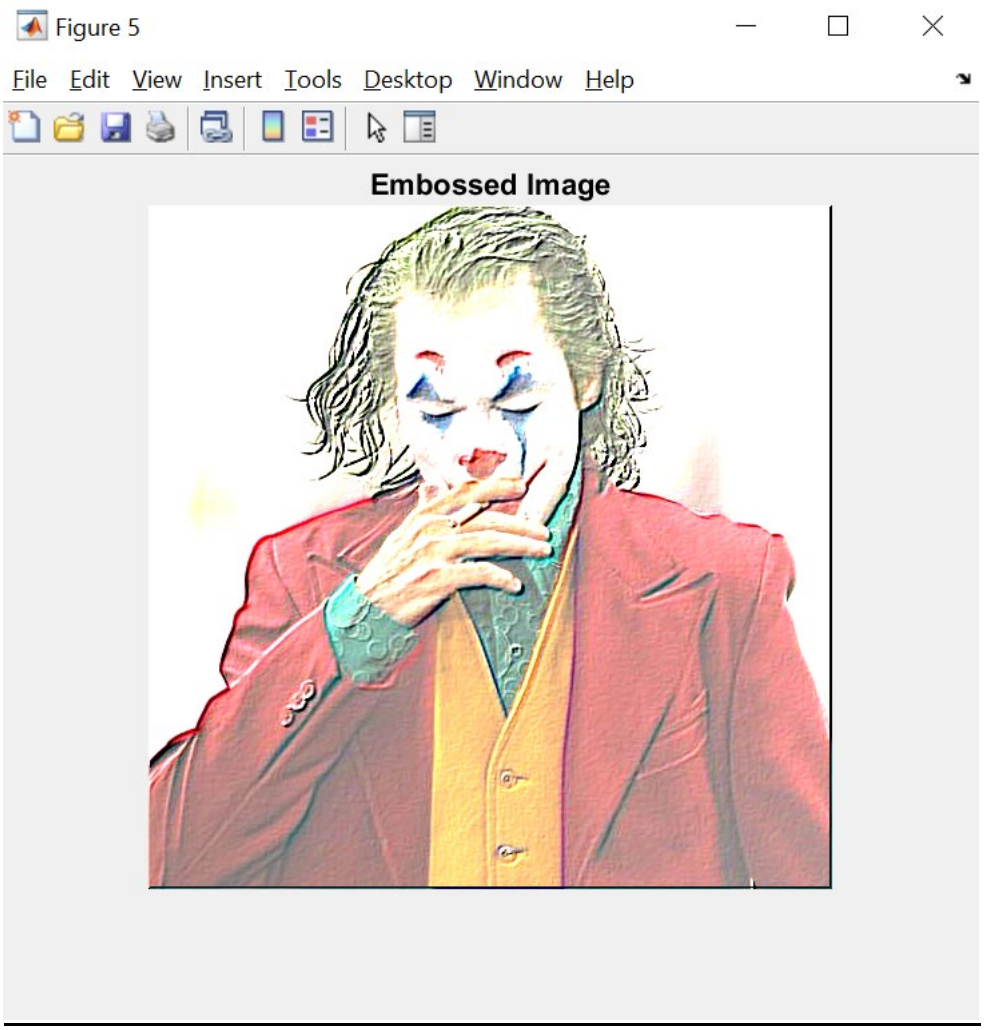
```
%% EMBOSSING
img = imread('jokerimage.png');

% Extract the individual red, green, and blue color channels.
redChannel = img(:, :, 1);
greenChannel = img(:, :, 2);
blueChannel = img(:, :, 3);

%Red channel
mask = [-2 -1 0 ; -1 1 1 ; 0 1 2];
[rSize, cSize] = size(mask);

[nrow, ncol] = size(redChannel);
redChannel = cast(redChannel, 'double');
newImgRed = zeros(nrow,ncol);

% Corners
subMask = mask(2:3,2:3);
i = 1;
j = 1;
subMat = redChannel(i:i+1,j:j+1);
```

```
newImgRed(i,j) = sum(sum(subMask.*subMat)) + 128;

subMask = mask(1:2,1:2);
i = nrow;
j = ncol;
subMat = redChannel(i-1:i,j-1:j);
newImgRed(i,j) = sum(sum(subMask.*subMat)) + 128;


% Edges
subMask = mask(2:3,1:3);
i = 1;
for j = 2:ncol-1
    subMat = redChannel(i:i+1,j-1:j+1);
    newImgRed(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:2,1:3);
i = nrow;
for j = 2:ncol-1
    subMat = redChannel(i-1:i,j-1:j+1);
    newImgRed(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:3,2:3);
j = 1;
for i = 2:nrow-1
    subMat = redChannel(i-1:i+1,j:j+1);
    newImgRed(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:3,1:2);
j = ncol;
for i = 2:nrow-1
    subMat = redChannel(i-1:i+1,j-1:j);
    newImgRed(i,j) = sum(sum(subMask.*subMat)) + 128;
end

% Inner
for i = 0.5*(rSize+1) : nrow - 0.5*(rSize+1)
    for j = 0.5*(cSize+1) : ncol - 0.5*(cSize+1)
        subMat = redChannel(i-1:i+1,j-1:j+1);
        newImgRed(i,j) = sum(sum(subMat.*mask)) + 128;
    end
end
%convert that array to an array of type uint
newImgRed = cast(newImgRed, 'uint8');

%Green channel
mask = [-2 -1 0 ; -1 1 1 ; 0 1 2];
[rSize, cSize] = size(mask);

[nrow, ncol] = size(greenChannel);
greenChannel = cast(greenChannel, 'double');
newImgGreen = zeros(nrow,ncol);
```

```
% Corners
subMask = mask(2:3,2:3);
i = 1;
j = 1;
subMat = greenChannel(i:i+1,j:j+1);
newImgGreen(i,j) = sum(sum(subMask.*subMat)) + 128;

subMask = mask(1:2,1:2);
i = nrow;
j = ncol;
subMat = greenChannel(i-1:i,j-1:j);
newImgGreen(i,j) = sum(sum(subMask.*subMat)) + 128;


% Edges
subMask = mask(2:3,1:3);
i = 1;
for j = 2:ncol-1
    subMat = greenChannel(i:i+1,j-1:j+1);
    newImgGreen(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:2,1:3);
i = nrow;
for j = 2:ncol-1
    subMat = greenChannel(i-1:i,j-1:j+1);
    newImgGreen(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:3,2:3);
j = 1;
for i = 2:nrow-1
    subMat = greenChannel(i-1:i+1,j:j+1);
    newImgGreen(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:3,1:2);
j = ncol;
for i = 2:nrow-1
    subMat = greenChannel(i-1:i+1,j-1:j);
    newImgGreen(i,j) = sum(sum(subMask.*subMat)) + 128;
end

% Inner
for i = 0.5*(rSize+1) : nrow - 0.5*(rSize+1)
    for j = 0.5*(cSize+1) : ncol - 0.5*(cSize+1)
        subMat = greenChannel(i-1:i+1,j-1:j+1);
        newImgGreen(i,j) = sum(sum(subMat.*mask)) + 128;
    end
end
%convert that array to an array of type uint
newImgGreen = cast(newImgGreen, 'uint8');

%Blue channel
mask = [-2 -1 0 ; -1 1 1 ; 0 1 2];
[rSize, cSize] = size(mask);
```

```
[nrow, ncol] = size(blueChannel);
blueChannel = cast(blueChannel, 'double');
newImgBlue = zeros(nrow,ncol);

% Corners
subMask = mask(2:3,2:3);
i = 1;
j = 1;
subMat = blueChannel(i:i+1,j:j+1);
newImgBlue(i,j) = sum(sum(subMask.*subMat)) + 128;

subMask = mask(1:2,1:2);
i = nrow;
j = ncol;
subMat = blueChannel(i-1:i,j-1:j);
newImgBlue(i,j) = sum(sum(subMask.*subMat)) + 128;


% Edges
subMask = mask(2:3,1:3);
i = 1;
for j = 2:ncol-1
    subMat = blueChannel(i:i+1,j-1:j+1);
    newImgBlue(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:2,1:3);
i = nrow;
for j = 2:ncol-1
    subMat = blueChannel(i-1:i,j-1:j+1);
    newImgBlue(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:3,2:3);
j = 1;
for i = 2:nrow-1
     subMat = blueChannel(i-1:i+1,j:j+1);
    newImgBlue(i,j) = sum(sum(subMask.*subMat)) + 128;
end

subMask = mask(1:3,1:2);
j = ncol;
for i = 2:nrow-1
    subMat = blueChannel(i-1:i+1,j-1:j);
    newImgBlue(i,j) = sum(sum(subMask.*subMat)) + 128;
end

% Inner
for i = 0.5*(rSize+1) : nrow - 0.5*(rSize+1)
    for j = 0.5*(cSize+1) : ncol - 0.5*(cSize+1)
        subMat = blueChannel(i-1:i+1,j-1:j+1);
        newImgBlue(i,j) = sum(sum(subMat.*mask)) + 128;
    end
end
%convert that array to an array of type uint
```

```
newImgBlue = cast(newImgBlue, 'uint8');
rgbImage = cat(3, newImgRed, newImgGreen, newImgBlue);

% Display the embossed color image.
figure,imshow(rgbImage),title('Embossed Image');
```



ALİ BATIR - 2015400261