

Dados do relatório Lista 1

Matéria: Processamento Digital de Imagens (PDI) **código:** CMP1084 A01

Aluno: Vitor de Almeida Silva

Matrícula: 20161003305497

ENUNCIADO

1. Aplique a operação de espelhar a imagem horizontalmente e verticalmente.
2. Aplique a operação de deformação, a sua escolha.
3. Aplique a operação de cortar a imagem.
4. Aplique a operação de translação em uma parte da imagem.
5. Aplique a operação de aumento de escala da imagem.
6. Crie um método que reduza a escala da imagem pela metade e outro que rotacione a imagem original em 180°.
7. Conceitue *morphing*.
8. Utilize a imagem resultante da questão 2 em seu método de rotação, rotacione em apenas 90°.

A imagem utilizada é mostrada na Figura 1.

Figura 1: Figura selecionada para procedimentos



Fonte: Edward (RUBI_SAMA, 2018)

DESENVOLVIMENTO

1. Aplique a operação de espelhar a imagem horizontalmente e verticalmente.

Resposta:

A operação de espelhamento pode ser feita utilizando métodos ponto a ponto. Isto é, através de um método de binarização é possível se inverter as linhas e colunas de modo a provocar o efeito de espelhamento vertical e horizontal. Deste modo os dois tipos de espelhamento pedidos foram feitos da seguinte forma:

- **Horizontal:** pega-se os pixels iniciando da última coluna de cada linha, e transfere eles para a primeira posição de uma nova matriz;
- **Vertical:** pega-se os pixels iniciando da última linha (N, M), e transfere eles para uma nova matriz começando do início (1,1);

A Figura 2 mostra os resultados das duas operações

Figura 2: figura original, espelhamento vertical e horizontal



Fonte: Autoral

Código questão1:

```
%0) lendo e mostrando uma imagem do disco

img=imread('C:\Users\Vitor\Desktop\faculdade\9 nono
período\PDI\Lista 1\edward.png');
imshow(img(:,:,:));
title('img original');
figure();

[x y k] = size(img);

%1. Aplique a operação de espelhar a imagem horizontalmente e
verticalmente.

%Espelhamento na vertical
img2=img;
for i=x: -1: 1
    for j=y: -1: 1
        img2(421-i,511-j,:) = img(i,j,:);
    end
end
imshow(img2(:,:,:));
title('img esp-vertical');
figure();

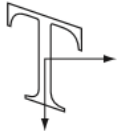
%Espelhamento na horizontal
img2=img;
for i=1: x
    for j=y: -1: 1
        img2(i,511-j,:) = img(i,j,:);
    end
end
imshow( img2(:,:,:) );
title('img esp-horizontal');
figure();
```

2. Aplique a operação de deformação, a sua escolha.

Resposta:

O tipo de distorção escolhida para o exercício se chama cisalhamento. Tal distorção, consiste em definir uma constante que será aplicada em uma das dimensões da matriz de imagem seguida de uma soma. A Figura 3 mostra as definições retiradas do livro texto.

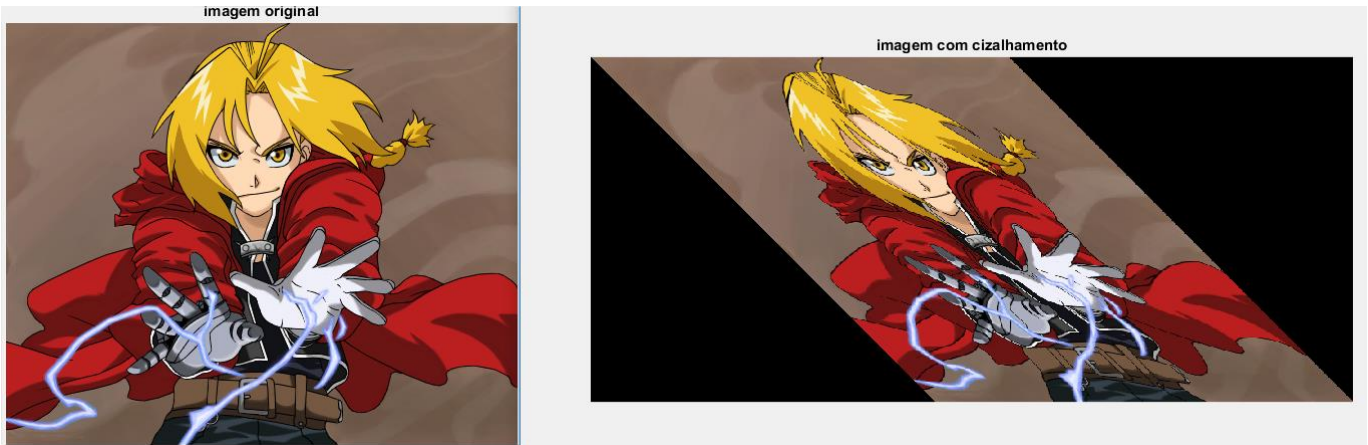
Figura 3: definição da formula e matriz

Cisalhamento (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v + s_v w$ $y = w$	
----------------------------	---	-------------------------	---

Fonte: (GONZALEZ, WOODS, 2010)

Esse processo é repetido pixel a pixel o que provoca um deslocamento das linhas fazendo assim uma deformação da imagem. A Figura 4 mostra o resultado do cisalhamento com constante 1.

Figura 4: imagem original e imagem com cisalhamento



Fonte: Autoral

Código questão2:

```
%2. Aplique a operação de deformação, a sua escolha.

%0) lendo e mostrando uma imagem do disco
img=imread('C:\Users\Vitor\Desktop\faculdade\9 nono
periodo\PDI\Lista 1\edward.png');

img2=img;
imshow(img2(:,:));
title('imagem original');
figure();

[xo yo k] = size(img2);
constCisi=1;           %constante de cisalhamento

imgNova=uint8(zeros([xo (yo*constCisi + xo) 3 ]));

for i=1: xo
    for j=1 : yo
        x=i;
        y=constCisi*i+j;
        imgNova(x,y,:)=img2(i,j,:);
    end
end

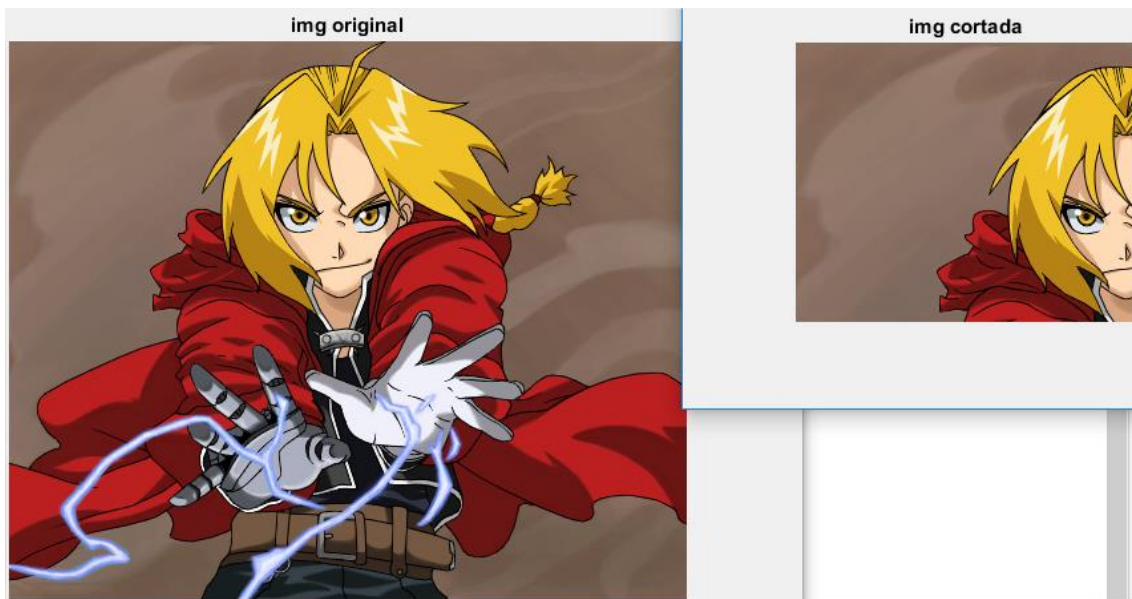
imshow(imgNova(:,:));
title('imagem com cizalhamento');
figure();
```

3. Aplique a operação de cortar a imagem.

Resposta:

A operação de cortar a imagem foi realizada através de uma operação ponto a ponto. Em tal operação, foram estabelecidas as dimensões x e y de onde seria realizado o corte da imagem. Resumidamente, é necessário apenas definir as dimensões e realizar um duplo for coletando os pixels da região definida e colocando em uma nova imagem. A Figura 5 mostra o resultado.

Figura 5: Imagem original e imagem cortada em $x/2$ e $y/2$



Fonte: Autoral

Código questão 3:

```
%3. Aplique a operação de cortar a imagem
%dimensões de corte

%0) lendo e mostrando uma imagem do disco
img=imread('C:\Users\Vitor\Desktop\faculdade\9 nono periodo\PDI\Lista
1\edward.png');

img2=img;
imshow(img2(:,:,:));
title('imagem original');
figure();

[xo yo k] = size(img2);

X1=xo/2;
Y1=yo/2;

img3=uint8(zeros([X1 Y1 3 ]));

for i=1: X1
    for j=1: Y1
        img3(i,j,:) = img(i,j,:);
    end
end

imshow(img3(:,:,:));
title('img cortada na metade');
figure();
```

4. Aplique a operação de translação em uma parte da imagem.

Resposta:

A translação, para essa questão, consiste de se definir uma dimensão x e y, que será translada e dois fatores de translação Tx e Ty que indicam o quanto serão transladas na direção de x e de y. A Figura 6 indica a definição do livro texto.

Figura 6: definição de translação

Translação	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$x = v + t_x$ $y = w + t_y$	
------------	---	-----------------------------	---

Fonte: (GONZALEZ, WOODS, 2010)

A Figura 7 mostra a translação aplicada na imagem definida para a lista.

Figura 7: Translação



Fonte: Autoral

Código questão 4:

```
%0) lendo e mostrando uma imagem do disco
img=imread('C:\Users\Vitor\Desktop\faculdade\9 nono periodo\PDI\Lista
1\edward.png');
imshow(img(:,:, :));
figure();

img2=img;

[xo yo] = size(img2);

%constantes de translação
tx=xo/2;
ty=yo/400;

%delimitações da translação
X1= xo/4;
Y1= yo/6;

for i=1: X1
    x=ceil(i+tx);
    for j=1: Y1
        y=ceil(j+ty);
        if(x>=1 || x<=xo && y>=1 || y<=yo) %verifica os cantos da figura
            img2(i,j,:)=0;
            img2(x,y,:)=img(i,j,:);
        end
    end
end
end
```

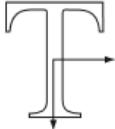
Fonte: Autoral

5. Aplique a operação de aumento de escala da imagem.

Resposta:

A operação de escala consiste em aumentar as dimensões da imagem em alguma ordem (escala). No Livro texto é exposta uma forma simplificada de fazer este processo. Os valores dos pontos da imagem antiga são calculados tendo em vista a escala de forma a serem colocados na próxima imagem. A Figura 8 mostra as definições de escala.

Figura 8: escala

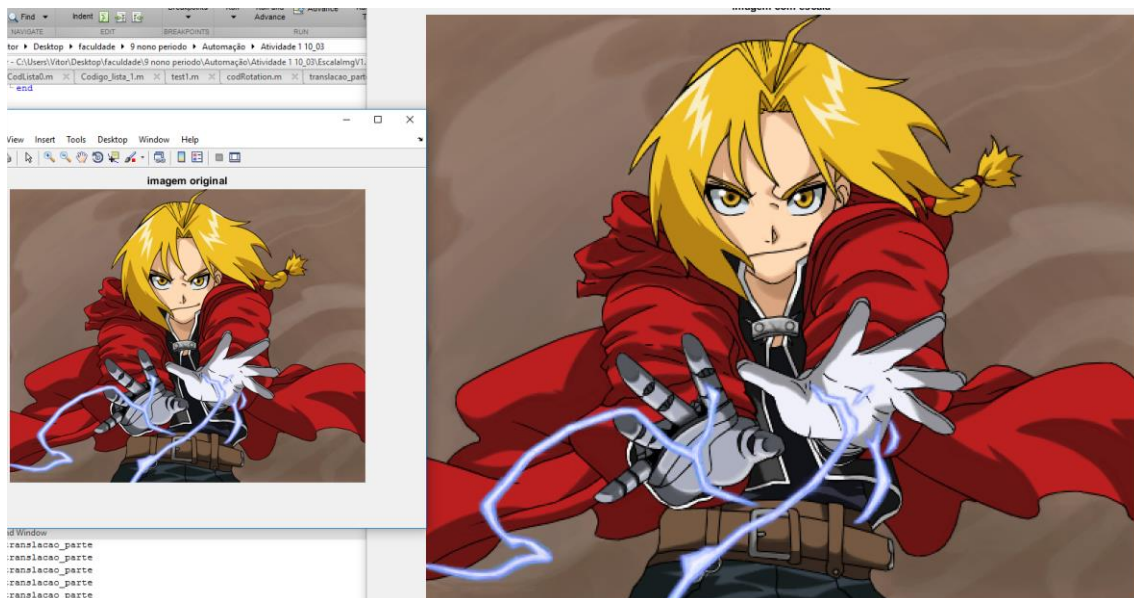
Escala	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= c_x v \\ y &= c_y w \end{aligned}$	
--------	---	--	---

Fonte: (GONZALEZ, WOODS, 2010)

Somente com este método não é possível ter um aumento de escala de forma mais fiel. Isso se dá por conta de que sobram diversos pixels preenchidos com preto na imagem nova. Para solucionar este problema pode-se utilizar algum método de repetição de pixel (repetir o pixel $n-1$ na lacuna preta, por exemplo) ou utilizar método envolvendo conceitos de vizinhança de pixel.

O que foi feito para este exercício, foi a criação de um duplo for, no qual, os pontos na imagem original eram coletados de índices i e j divididos pela escala definida e inseridos na posição de pixel 0 da nova Matriz. A Figura 9 mostra o resultado do aumento da escala.

Figura 9: Aumento da escala no fator de 2 (imagem original/imagem com escala)



Fonte: Autoral

Código questão 5:

```
%5. Aplique a operação de aumento de escala da imagem.

%0) lendo e mostrando uma imagem do disco
img=imread('C:\Users\Vitor\Desktop\faculdade\9 nono periodo\PDI\Lista
1\edward.png');

img2=img;
imshow(img2(:,:,:));
title('imagem original');
figure();

escalaX=2;
escalaY=2;
[xo yo k] = size(img2);

%para correção do ponto preto em matriz não quadrada
%pega a posição atual de C, divide x e y pela escala e pega essa posição na
%imagem A para preencher o ponto preto na imagem C

imgNova= uint8(zeros([escalaX*xo escalaY*yo 3 ]));

%não é necessário realizar a verificação de fora da img por conta que o
%tamanho vai ser aumentado linearmente
for i=1: xo
    for j=1: yo
        x=escalaX*i;
        y=escalaY*j;
        imgNova(x,y,:)=img2(i,j,:);
    end
end

%dimensões de linha/coluna da nova imagem
X1=size(imgNova,1);
Y1=size(imgNova,2) ;

%Correção dos pontos pretos
for i=1:X1
    for j=1:Y1
        if(imgNova(i,j,:)==0)
            imgNova(i,j,:)=img2(ceil(i/escalaX),ceil(j/escalaY),:);
        end
    end
end

imshow(imgNova(:,:,:));
title('imagem com escala');
figure();
```

6. Crie um método que reduza a escala da imagem pela metade e outro que rotacione a imagem original em 180°.

Resposta:

Para reduzir a escala da imagem pela metade, basta utilizar o mesmo método usado na questão 5, porém, com valores de escala fracionários. Para realização da rotação, foi adotado o método do livro no qual se utiliza dos valores de seno e cosseno definindo o centro da imagem e girando-a em torno deste. A Figura 10 mostra a definição do livro em relação a rotação.

Figura 10: Rotação

Rotação	$\begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= v \cos \theta - w \sin \theta \\ y &= v \sin \theta + w \cos \theta \end{aligned}$	
---------	--	--	--

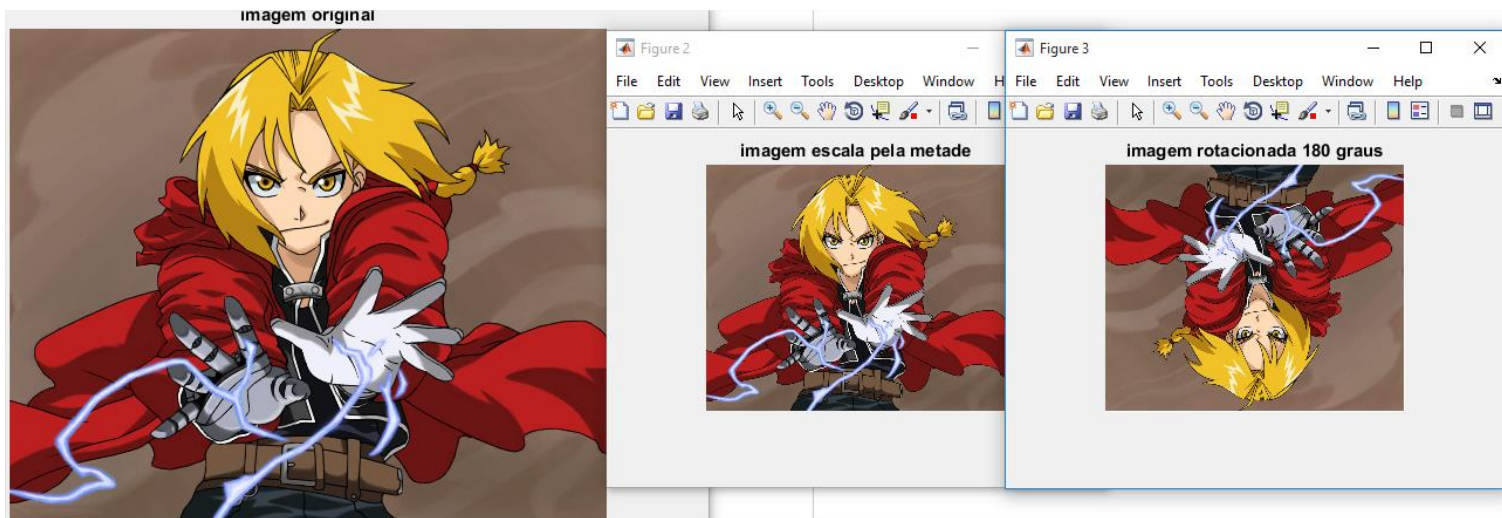
Fonte: (GONZALEZ, WOODS, 2010)

Além de se realizar as operações indicadas na Figura 10, ainda foi necessário definir outras variáveis, como as seguintes:

- **Centro das imagens:** é definido pelas dimensões x e y divididas por dois;
- **Tamanho máximo da imagem resultante:** Para que a imagem original seja rotacionada sem perda de partes, a matriz da imagem que receberá o resultado deve ter o tamanho apropriado;
- **Calculo da posição do pixel:** Cálculo da posição do pixel na imagem nova levando em conta o centro da imagem;

Além de utilizar as definições do livro também foram coletadas informações de funções como *ceil* e *round*, que são dois tipos de arredondamento no site da Mathworks e stackoverflow (MEMBER, 2014). A Figura 11 mostra o resultado das operações requisitadas no exercício.

Figura 11: imagem original/ imagem escala metade/ imagem rotacionada 180



Fonte: Autoral

Código questão 6:

Código da função de escala pela metade:

```
function [ imgNova ] = reduz_Escala_Metade(img2)

escalaX=1/2; %configurado para retornar a metade da escala
escalaY=1/2;
[xo yo k] = size(img2);
imgNova=uint8(zeros([round(xo*escalaX) round(yo*escalaY) 3 ]));

for i=1: xo
    for j=1: yo
        x=ceil(escalaX*i);
        y=ceil(escalaY*j);
        imgNova(x,y,:)=img2(i,j,:);
    end
end

end
```

Código da função de rotacionar em 180 graus:

```
function [ C ] = rotacao_180( img )

[rowsi,colsi,z]= size(img);

angle=180;

rads=2*pi*angle/360;

%calcula o tamanho dos vetores de linha e coluna da nova imagem. isso é
%feito para que a imagem original sirva dentro das dimensões da proxima
%imagem após a rotação. O uso do valor absoluto garante que não sejam
%retornados valores negativos. caso não seja feita está operação os cantos
%da imagem rotacionada podem ser cortados.

rowsf=ceil(rowsi*abs(cos(rads))+colsi*abs(sin(rads)));
colsf=ceil(rowsi*abs(sin(rads))+colsi*abs(cos(rads)));

% define an array withcalculated dimensionsand fill the array with zeros ie.,black
C=uint8(zeros([rowsf colsf 3 ]));

%C(rowsf,colsf,3)=0;
C(:, :, :)=255; %fundo configurado para cor branca

%calculating center of original and final image
xo=ceil(rowsi/2); %função ceil(x) pega o valor inteiro mais próximo do número,
yo=ceil(colsf/2); %sempre maior ou igual;

midx=ceil((size(C,1))/2);
midy=ceil((size(C,2))/2);

%o centro da imagem é importante pois a nova imagem irá girar em torno
%deste valor

%loop para calcular as coordenadas correspondentes do pixel da imagem nova
%de acordo com cada pixel na imagem original

for i=1:size(C,1)
    for j=1:size(C,2)

        x= (i-midx)*cos(rads)+(j-midy)*sin(rads); %calculo para retornar o valor da linha na nova imagem
        y= -(i-midx)*sin(rads)+(j-midy)*cos(rads); %Calculo para retornar o valor da coluna na nova imagem
        x=round(x)+xo; % A função round arredonda o número para o inteiro mais próximo
        y=round(y)+yo; %soma o ponto com o centro para compensar o deslocamento

        if (x>=1 && y>=1 && x<=size(img,1) && y<=size(img,2) ) %verifica se a cordenada está dentro da
            imagem
                C(i,j,:)=img(x,y,:); %adiciona o pixel na cordenada x,y calculada nos locais
            correspondentes em C ,m
        end
    end
end
end
```

Código principal:

```
%0) lendo e mostrando uma imagem do disco
img=imread('C:\Users\Vitor\Desktop\faculdade\9 nono periodo\PDI\Lista 1\edward.png');

img2=img;
imshow(img2(:, :, :));
title('imagem original');
figure();

imgNova=reduz_Escala_Metade(img2);
imshow(imgNova(:, :, :));
title('imagem escala pela metade');
figure();
```

7. Conceitue *morphing*.

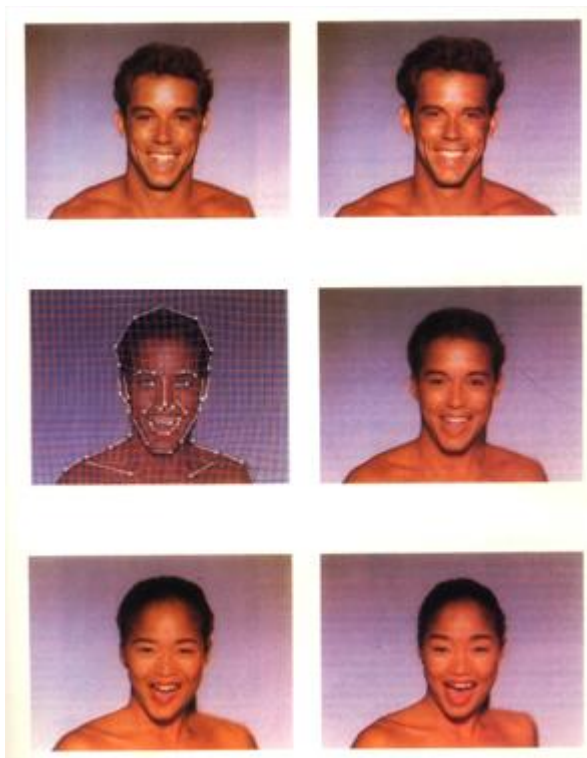
Resposta:

Morphing descreve a ação tomada quando uma imagem digital é transformada em outra imagem, ou em uma imagem de forma mais técnica. *Morphing* descreve a combinação de generalização de características da imagem através do cruzamento ou mistura entre elementos de duas ou mais imagens.

A ideia é pegar uma imagem e distorcer ela de acordo com outra imagem, o que recebe o nome de um dos métodos de *morphing* o *warping*. Ao se fazer esse efeito, um visualizador experiencia uma ilusão que a fotografia está se transformando em algo fluido, surrealista e com frequência de uma forma dramática.

A Figura 12 mostra um exemplo de *morphing*.

Figura 12: exemplo simplificado de *morphing*



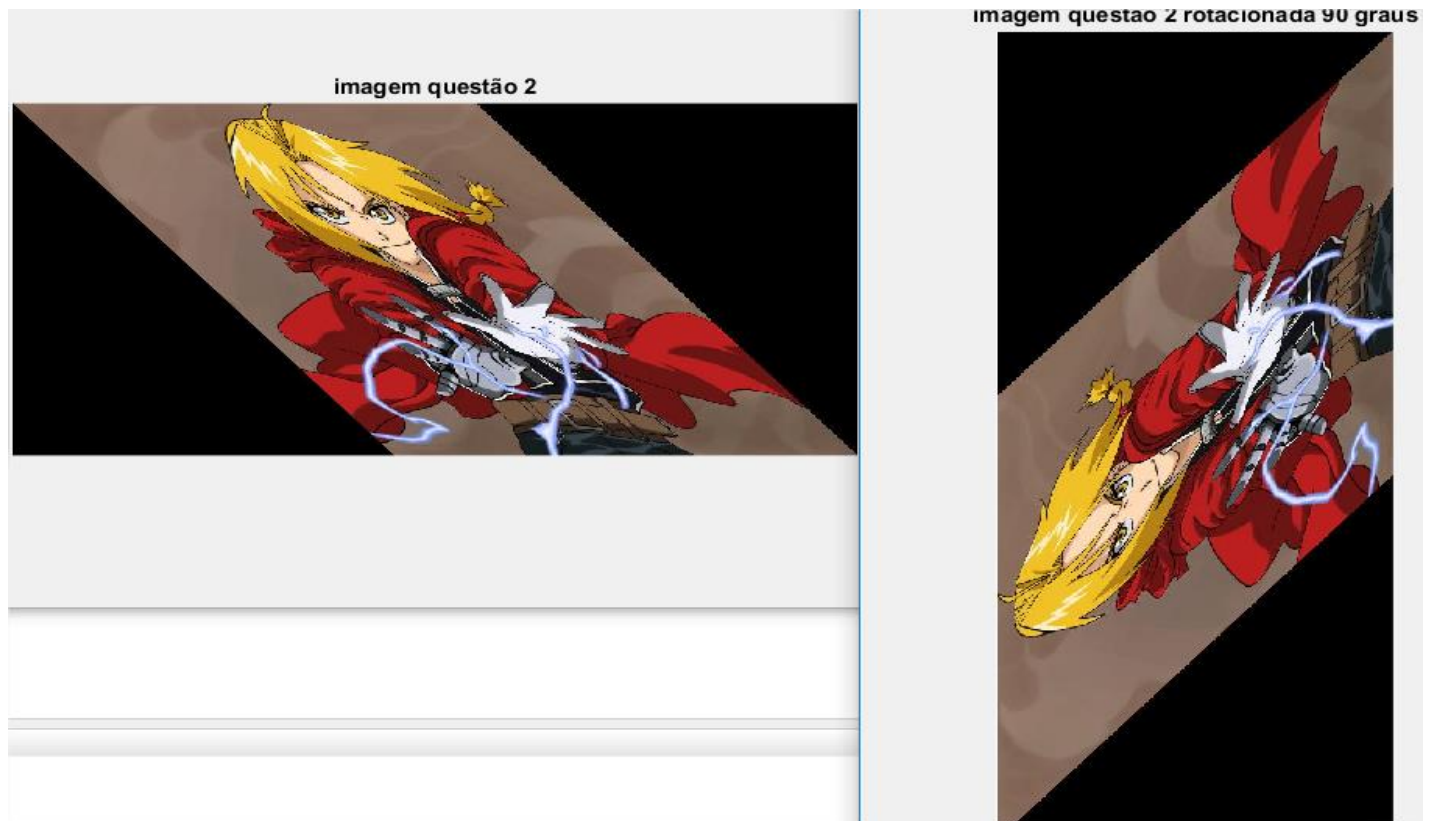
Fonte: (PENKLER, 1997)

8. Utilize a imagem resultante da questão 2 em seu método de rotação, rotacione em apenas 90°.

Reposta:

Procedimento mostrado na Figura 13.

Figura 13: rotação da imagem do exercício 2



Fonte: autoral

Código questão 8:

%8. Utilize a imagem resultante da questão 2 em seu método de rotação, rotacione em apenas 90°.

```
img2=imread('C:\Users\Vitor\Desktop\faculdade\9 nono periodo\Automação\Atividade 1 10_03\imgQuest8.jpg');
imshow(img2(:,:,:));
title('imagem questão 2');
figure();

img2=rotacao_90(img2);
imshow(img2(:,:,:));
title('imagem questão 2 rotacionada 90 graus');
figure();

function [ C ] = rotacao_90( img )

[rowsi,colsi,z]= size(img);

angle=90;

rads=2*pi*angle/360;

%calcula o tamanho dos vetores de linha e coluna da nova imagem. isso é
%feito para que a imagem original sirva dentro das dimensões da proxima
%imagem após a rotação. O uso do valor absoluto garante que não sejam
%retornados valores negativos. caso não seja feita esta operação os cantos
%da imagem rotacionada podem ser cortados.

rowsf=ceil(rowsi*abs(cos(rads))+colsi*abs(sin(rads)));
colsf=ceil(rowsi*abs(sin(rads))+colsi*abs(cos(rads)));

% define an array withcalculated dimensionsand fill the array with zeros ie.,black
C=uint8(zeros([rowsf colsf 3 ]));

%C(rowsf,colsf,3)=0;
C(:,:,:)=255; %fundo configurado para cor branca

%calculating center of original and final image
xo=ceil(rowsi/2); %função ceil(x) pega o valor inteiro mais próximo do número,
yo=ceil(colsi/2); %sempre maior ou igual;

midx=ceil((size(C,1))/2);
midy=ceil((size(C,2))/2);

%o centro da imagem é importante pois a nova imagem irá girar em torno
%deste valor

%loop para calcular as coordenadas correspondentes do pixel da imagem nova
%de acordo com cada pixel na imagem original

for i=1:size(C,1)
    for j=1:size(C,2)

        x= (i-midx)*cos(rads)+(j-midy)*sin(rads); %calculo para retornar o valor da linha na nova imagem
        y= -(i-midx)*sin(rads)+(j-midy)*cos(rads); %Calculo para retornar o valor da coluna na nova imagem
        x=round(x)+xo; % A função round arredonda o número para o inteiro mais próximo
        y=round(y)+yo; %soma o ponto com o centro para compensar o deslocamento

        if (x>=1 && y>=1 && x<=size(img,1) && y<=size(img,2) ) %verifica se a cordenada está dentro da imagem
            C(i,j,:)=img(x,y,:); %adiciona o pixel na cordenada x,y calculada nos locais
correspondentes em C ,m
        end
    end
end

end
```

REFERÊNCIAS

GONZALEZ, Rafael C., WOODS, Richard E. **Processamento Digital de Imagem**. São Paulo: Sv, 2010.

MEMBER, Stack. **Image rotation by Matlab without using imrotate**. 2014. Disponível em: <https://stackoverflow.com/questions/19684617/image-rotation-by-matlab-without-using-imrotate>. Acesso em: 11 fev. 2020.

RUBI_SAMA. **MURAL LIVRE DE RUBI_SAMA**. 2018. Disponível em: https://gartic.com.br/rubi_sama/desenho-livre/edward-elric-p-froid. Acesso em: 11 mar. 2020.

PENKLER, By Thomas. **Morphing**. 1997. Disponível em: <https://old.cescg.org/CESCG97/penkler/morphing.htm>. Acesso em: 11 mar. 2020.