Trab 5 - Octave

Anny Caroline Correa Chagas
Ciência da Computação, UERJ

21 de Julho de 2019

Transformação Projetiva em Imagem

1. Crie uma função no Octave que deforme uma imagem de acordo com uma transformação projetiva.
2. A função deve ter como parâmetros o caminho de um arquivo com a imagem a ser deformada e uma matriz que indica quatro pares de pontos correspondentes na imagem original e na imagem deformada.
3. Abaixo, um exemplo da matriz de pares de pontos. A primeira linha indica que o ponto 1,1 (linha 1 e coluna 1) na imagem original corresponde ao ponto 50,50 (linha 50, coluna 50) na imagem deformada.

   [1 1 50 50;
   1 350 1 300;
   390 1 390 1;
   390 350 300 350];
4. Considere que a imagem original pode ser uma imagem de tons de cinza (monocromática) ou com três bandas (tricromática). Você tem que descobrir isso.
5. Gere duas imagens deformadas. As intensidades dos pixels nas imagens deformadas devem ser calculadas através de: (1) núcleo de reconstrução constante (vizinho mais próximo); (2) núcleo de reconstrução triangular 2D (bilinear).
6. A função deve exibir a imagem original e as imagens deformadas.
7. Submeta o código da função e de um script que execute a função (em arquivos .m) dentro de um arquivo .pdf onde se mostre também um ou mais exemplos de resultados.

Resposta:

```
function deforma(caminhoIMG, P)
  IMG = imread(caminhoIMG);
  IMG = im2double(IMG);
  BANDA_IMG = IMG(:, :, 1);

  L = [P(1:size(P)(1), 3:4)];
  L = transpose(L)(:);

  A1 = [P(:,1)              P(:,2)              ones(size(P)(1),1)  zeros(size(P)(1),1)
   ↪  zeros(size(P)(1),1) zeros(size(P)(1),1) -P(:,1).*P(:,3) -P(:,2).*P(:,3)];
  A2 = [zeros(size(P)(1),1) zeros(size(P)(1),1) zeros(size(P)(1),1) P(:,1)
   ↪  P(:,2)              ones(size(P)(1),1)  -P(:,1).*P(:,4) -P(:,2).*P(:,4)];
  A = reshape([A1 A2]', 8,  size(P)(1)*2)';

  x = inv(A'*A) * A' * L;
  x(9) = 1;
  T = reshape(x',3,3)';
```

```
16
17    #Calcula canvas
18    Ptio = [ T * [size(IMG,1) 1 1]', T * [1 1 1]',  T * [1 size(IMG,2) 1]' , T *
       ↪  [size(IMG,1) size(IMG,2) 1]'];
19
20    Ptio(:,1) = Ptio(:,1)/Ptio(3,1);
21    Ptio(:,2) = Ptio(:,2)/Ptio(3,2);
22    Ptio(:,3) = Ptio(:,3)/Ptio(3,3);
23    Ptio(:,4) = Ptio(:,4)/Ptio(3,4);
24
25    sobra_lin = min( (Ptio(1,:)<0) .* Ptio(1,:)) * (-1);
26    sobra_col = min( (Ptio(2,:)<0) .* Ptio(2,:)) * (-1);
27
28    Ptio(1,:) = Ptio(1,:) .+ sobra_lin;
29    Ptio(2,:) = Ptio(2,:) .+ sobra_col;
30
31    lin = max(Ptio(1,:));
32    col = max(Ptio(2,:));
33
34    canvasNRC = zeros(lin, col, size(IMG, 3));
35    canvasNRT = zeros(lin, col, size(IMG, 3));
36
37    # Calcula T inverso
38    Tinv = inv(T);
39
40    #Pega cada pixel da imagem com zeros
41    for k=1:size(IMG, 3)
42      for i=1:lin
43        for j=1:col
44          coord = Tinv * [i-sobra_lin j-sobra_col 1]';
45          coord = coord ./ coord(3);
46
47          # NRC - se coord dentro imagem original
48          coord = round(coord);
49          if coord(1) > 0 && coord(1) <= size(IMG,1) && coord(2) >0 && coord(2) <=
             ↪  size(IMG,2)
50            canvasNRC(i,j,k) = IMG(coord(1), coord(2), k);
51            canvasNRT(i,j,k) = nrt(IMG, coord(1), coord(2));
52          endif
53
54        endfor
55      endfor
56    endfor
57
58    figure;
59    imshow(IMG);
60    title('Original');
61
62    figure;
63    imshow(canvasNRC);
64    title('Nucleo de reconstrucao constante (vizinho mais proximo)');
65
66    figure;
67    imshow(canvasNRT);
68    title('Nucleo de reconstrucao triangular 2D (bilinear)');
69  end
```

```
function fc = nrt(f, x, y)
  j = floor(x);
  k = floor(y);
  u = x - j;
  v = y - k;

  #{
  fa = (1-v)*f(j,k) + v*f(j,k+1);
  fb = (1-v)*f(j+1,k) + v*f(j+1,k+1);
  fc = (1-u)*fa + u*fb;
  #}

  fa = (1-v)*f(j,k);

  if ((j+1) <= size(f,1))
    fb = (1-v)*f(j+1,k);
  else
    fb = 0;
  endif

  if (((j+1) <= size(f,1)) && ((k+1) <= size(f,2)))
    fb = fb + v*f(j+1,k+1);
  endif

  if ((k+1) <= size(f,2))
    fa = fa + v*f(j,k+1);
  endif

  fc = (1-u)*fa + u*fb;
end
```

Aplicação do código para a imagem *Coins.png*

```
P = [
 1   1   50  50;
 1   350 1   300;
 390 1   390 1;
 390 300 300 300;
 390 350 300 350
];

deforma("imgs/Coins.png",P);
```

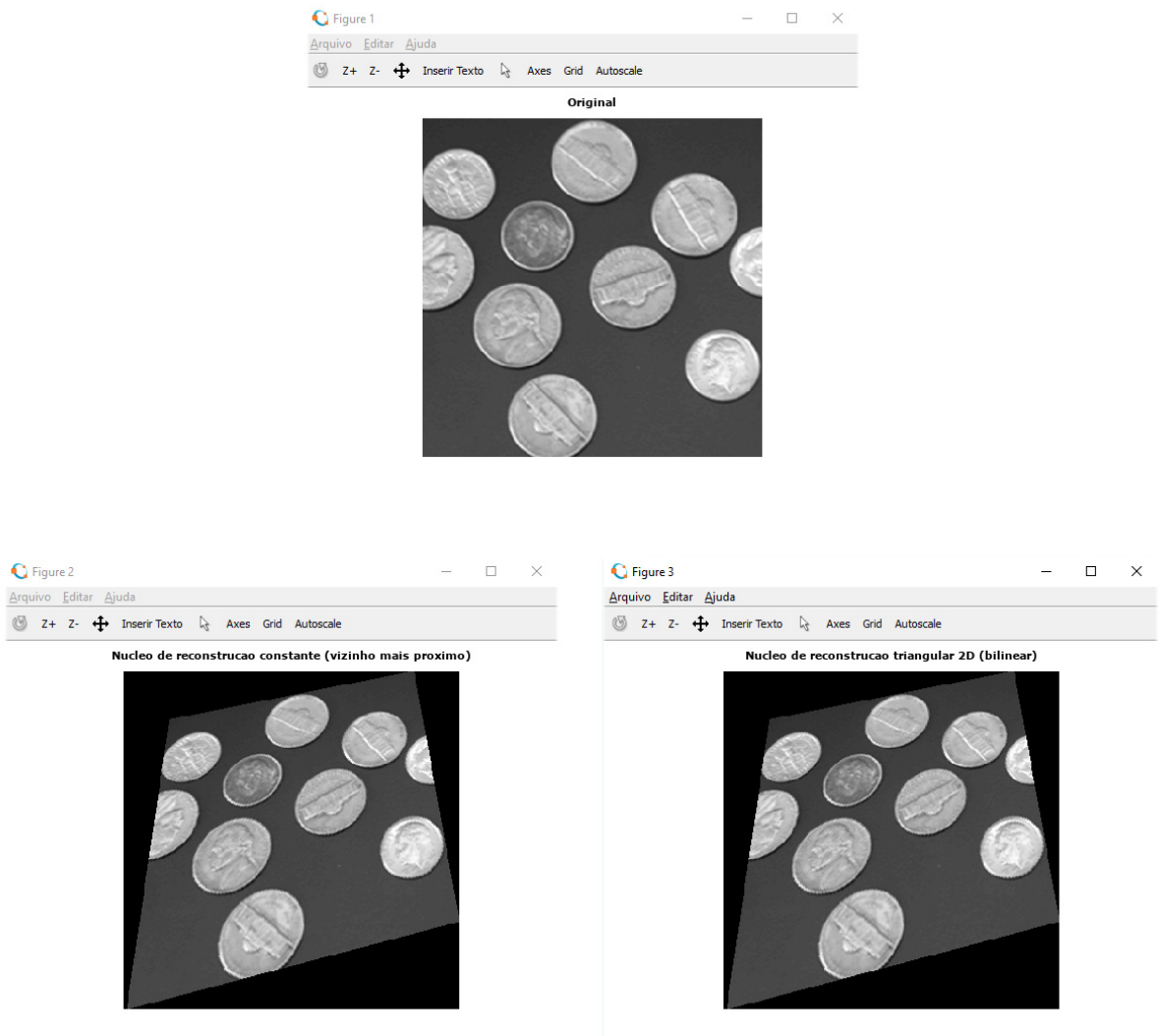

Figura 1: Resultado da execução do código t5 para a imagem *Coins.png*

Aplicação do código para a imagem *woman.png*

```
P = [
 1   1   50  50;
 1   350 1   300;
 390 1   390 1;
 390 300 300 300;
 390 350 300 350
];

deforma("imgs/woman.png",P);
```

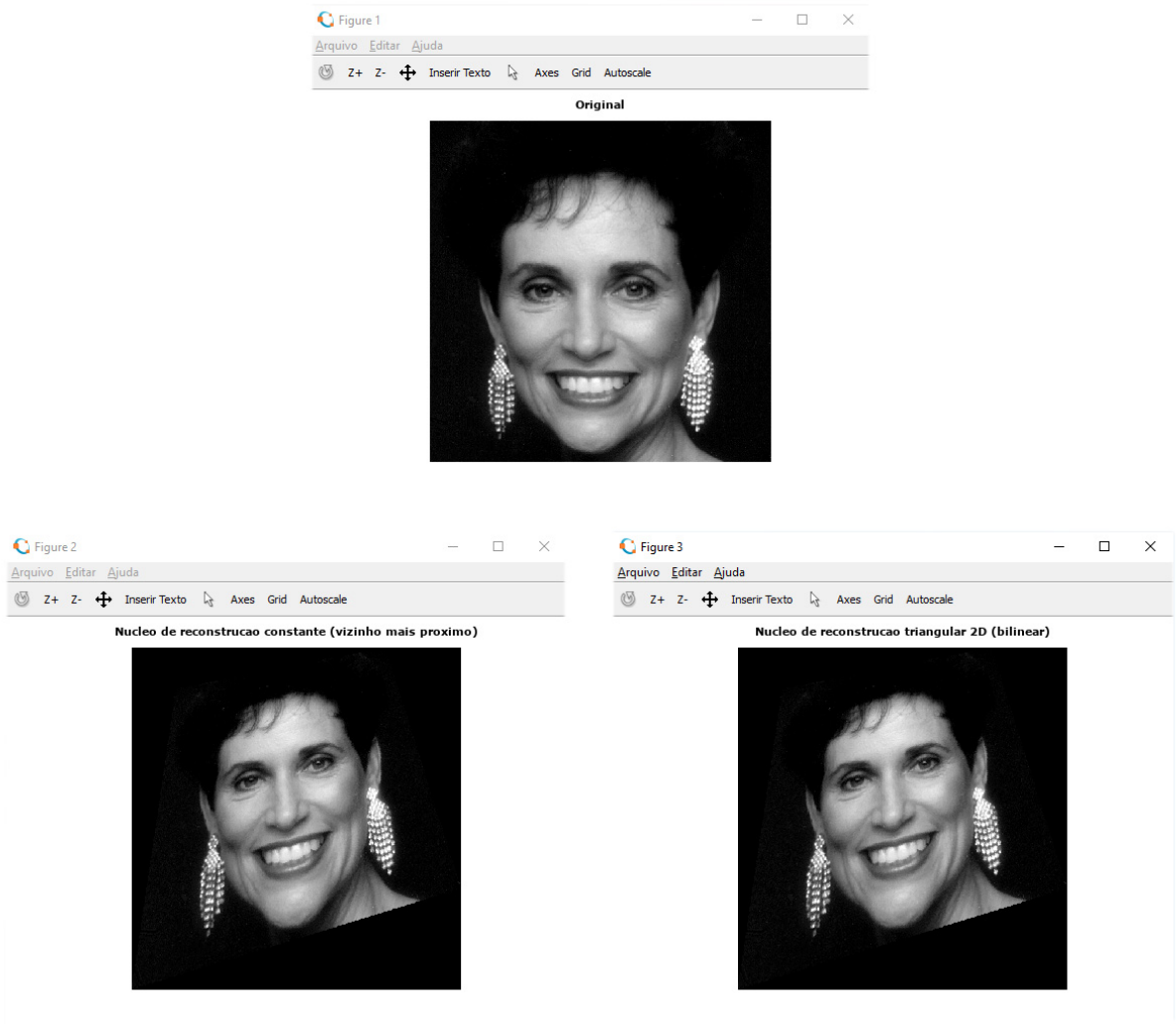

Figura 2: Resultado da execução do código t5 para a imagem *woman.png*

Aplicação do código para a imagem *Ferrari.jpg*

```
P = [
 1   1   50  50;
 1   350 1   300;
 390 1   390 1;
 390 300 300 300;
 390 350 300 350
];

deforma("imgs/Ferrari.jpg",P);
```

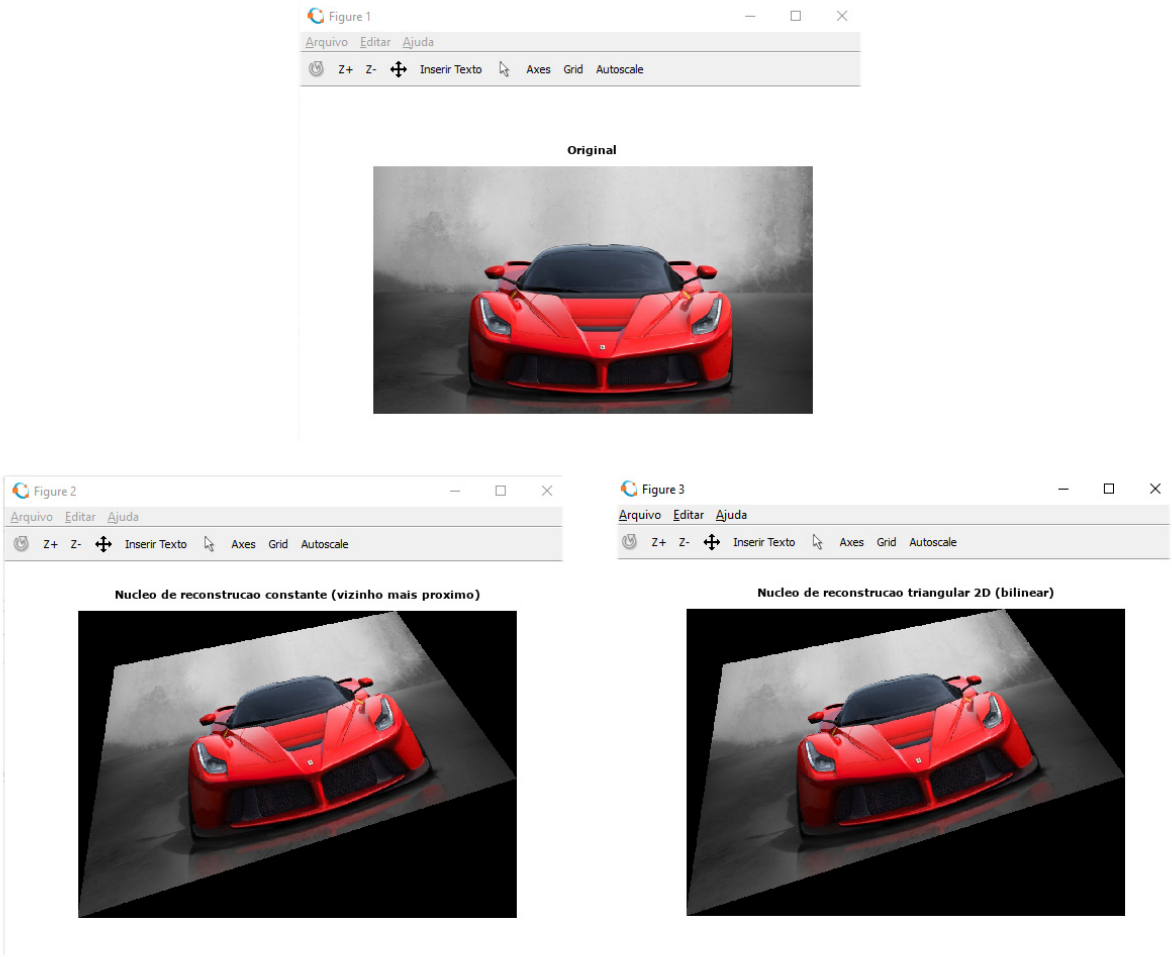

Figura 3: Resultado da execução do código t5 para a imagem *Ferrari.jpg*

Aplicação do código para a imagem *Ferrari2.jpg*

```
P = [
 1   1   50  50;
 1   350 1   300;
 390 1   390 1;
 390 300 300 300;
 390 350 300 350
];

deforma("imgs/Ferrari2.jpg",P);
```

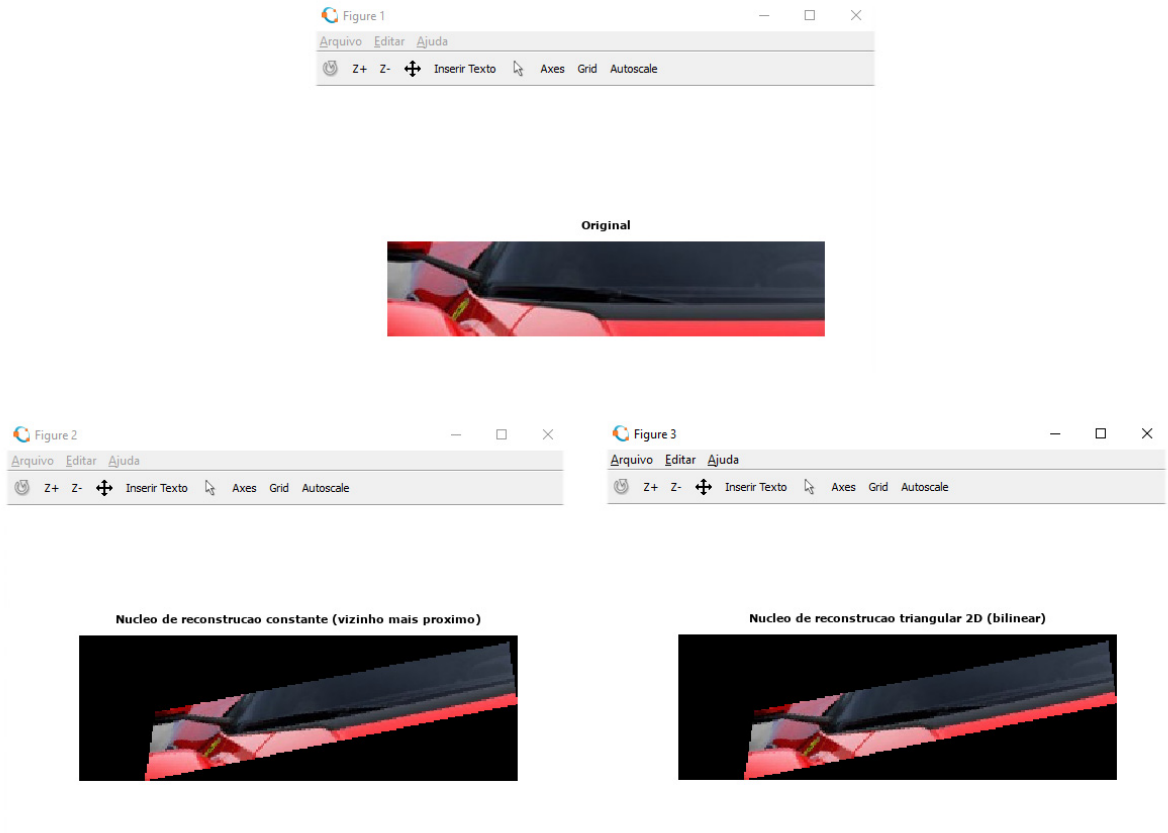

Figura 4: Resultado da execução do código t5 para a imagem *Ferrari2.jpg*

Aplicação do código para a imagem *cores.png*

```
P = [
 1   1   50  50;
 1   350 1   300;
 390 1   390 1;
 390 300 300 300;
 390 350 300 350
];

deforma("imgs/cores.png",P);
```

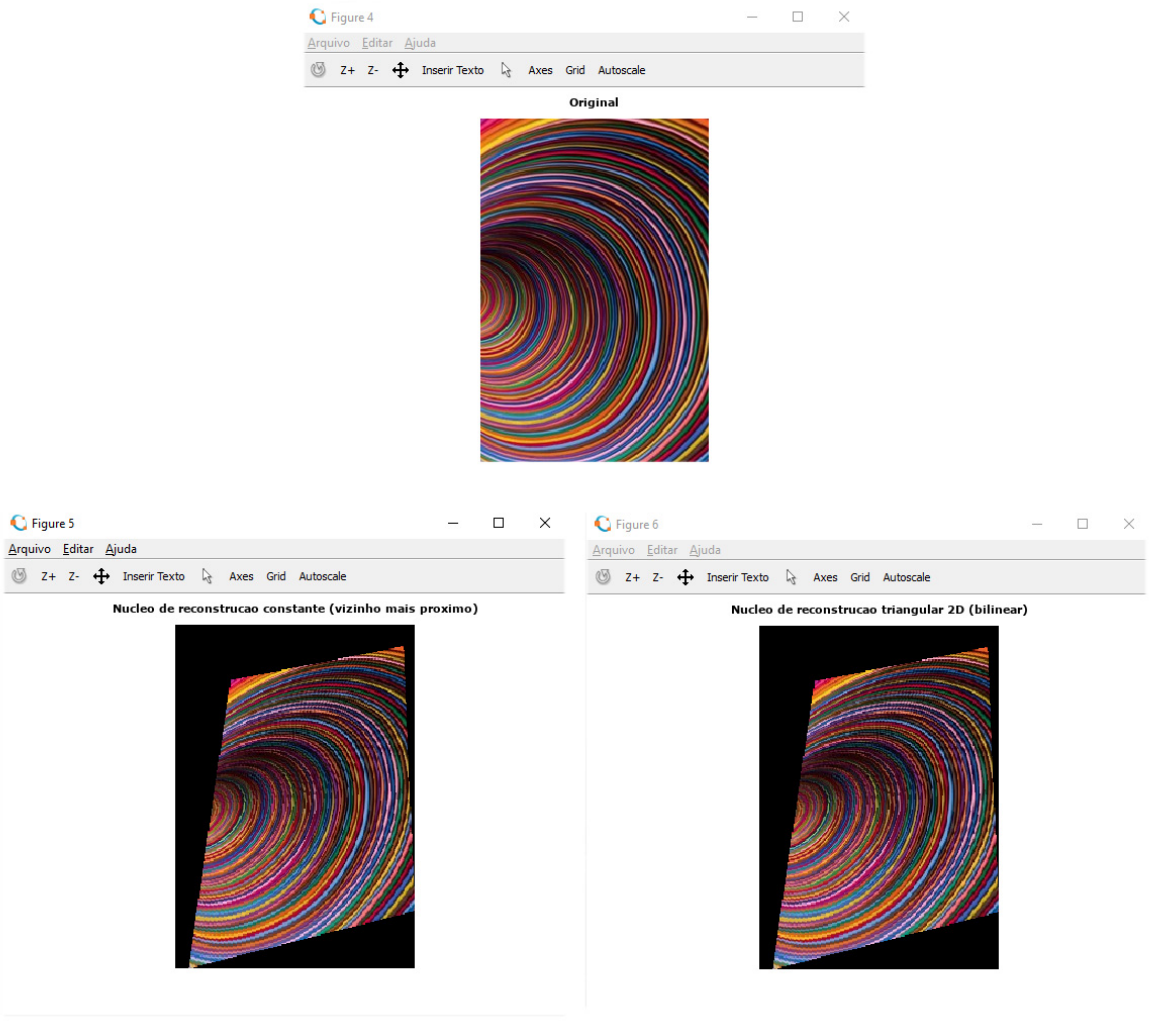

Figura 5: Resultado da execução do código t5 para a imagem *cores.png*