

Rio de Janeiro, 26 de dezembro de 2017

Universidade do Estado do Rio de Janeiro

Instituto de Matemática e Estatística

**Computação Gráfica**

**Trabalho 5 – Transformação Projetiva em Imagem**

**Aluno:** Leonardo Lima Marinho

**Matrícula:** 2014.1.00506.11

**Código**

pkg load image;

function rt2d = reconstrucaoTriangular2D(M, x, y)

i = floor(x);

j = floor(y);

a = x - i;

b = y - j;

fa = (1 - b) \* M(i, j) + b \* M(i, j + 1);

fb = (1 - b) \* M(i + 1, j) + b \* M(i + 1, j + 1);

rt2d = (1 - a) \* fa + a \* fb;

end

function transformar(caminho\_img, M)

% L = A \* X

% Observacoes

L = [

M(1,3)

M(1,4)

M(2,3)

M(2,4)

M(3,3)

M(3,4)

M(4,3)

M(4,4)

];

% Coeficientes (sistema de equacoes)

A = [ M(1,1) M(1,2) 1 0 0 0 -M(1,1)\*M(1,3) -M(1,2)\*M(1,3)

0 0 0 M(1,1) M(1,2) 1 -M(1,1)\*M(1,4) -M(1,2)\*M(1,4)

M(2,1) M(2,2) 1 0 0 0 -M(2,1)\*M(2,3) -M(2,2)\*M(2,3)

0 0 0 M(2,1) M(2,2) 1 -M(2,1)\*M(2,4) -M(2,2)\*M(2,4)

M(3,1) M(3,2) 1 0 0 0 -M(3,1)\*M(3,3) -M(3,2)\*M(3,3)

0 0 0 M(3,1) M(3,2) 1 -M(3,1)\*M(3,4) -M(3,2)\*M(3,4)

M(4,1) M(4,2) 1 0 0 0 -M(4,1)\*M(4,3) -M(4,2)\*M(4,3)

0 0 0 M(4,1) M(4,2) 1 -M(4,1)\*M(4,4) -M(4,2)\*M(4,4)

];

% Coeficientes (transformacao projetiva)

x = inv(A'\*A) \* A' \* L;

% Matriz de transformação

T = [x(1:3)'; x(4:6)'; cat(1, x(7:8), [1])'];

% Inversa da matriz de transformação

T\_inv = inv(T);

img = imread(caminho\_img);

banda\_img = img(:, :, 1);

sup\_esq = T \* [1; 1; 1];

sup\_esq = sup\_esq./sup\_esq(3);

sup\_dir = T \* [columns(banda\_img);1;1];

sup\_dir = sup\_dir./sup\_dir(3);

inf\_esq = T \* [1;rows(banda\_img);1];

inf\_esq = inf\_esq./inf\_esq(3);

inf\_dir = T \* [rows(banda\_img);columns(banda\_img);1];

inf\_dir = inf\_dir./inf\_dir(3);

borda = [sup\_esq'; sup\_dir'; inf\_esq'; inf\_dir'];

linhas = 1 + ceil(max(borda(:, 1)) - min(borda(:, 1)));

colunas = 1 + ceil(max(borda(:, 2)) - min(borda(:, 2)));

imgResultadoVMP = uint8(zeros(linhas, colunas, size(img, 3)));

imgResultadoNRT = uint8(zeros(linhas, colunas, size(img, 3)));

% Cálculo das intensidades dos pixels nas imagens deformadas

for i = 1:size(img, 3) % 1 se for monocromática, 3 se for tricromática.

banda\_img = img(:, :, i);

for l = 1:linhas % Varre a imagem original, ponto a ponto

for c = 1:colunas

p = T\_inv \* [l; c; 1];

p = p./p(3);

lins = rows(banda\_img);

cols = columns(banda\_img);

if ( (p(1) < 1) || (p(1) > lins) || (p(2) < 1) || (p(2) > cols) )

continue;

end

% imgResultadoVMP: nucleo de reconstrução constante (vizinho mais proximo)

vmp = round(p);

if (not( (vmp(1) < 1) || (vmp(1) > lins) || (vmp(2) < 1) || (vmp(2) > cols) ))

imgResultadoVMP(l, c, i) = banda\_img(vmp(1), vmp(2));

end

% imgResultadoNRT: nucleo de reconstrucao triangular 2D (bilinear)

imgResultadoNRT(l, c, i) = reconstrucaoTriangular2D(banda\_img, p(1), p(2));

end

end

end

figure;

imshow(img);

title('Original');

figure;

imshow(imgResultadoVMP);

title('Nucleo de reconstrucao constante (vizinho mais proximo)');

figure;

imshow(imgResultadoNRT);

title('Nucleo de reconstrucao triangular 2D (bilinear)');

end

% Le a imagem

caminho\_img = 'coins.png';

img = imread(caminho\_img);

% obtem dimensoes da imagem

a = size(img, 1); % altura

l = size(img, 2); % largura

% Pares de pontos

M = [ 1 1 50 50

1 l 1 300

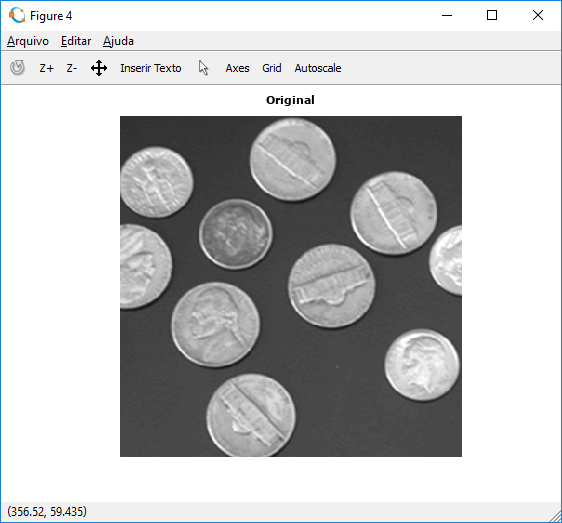
a 1 390 1

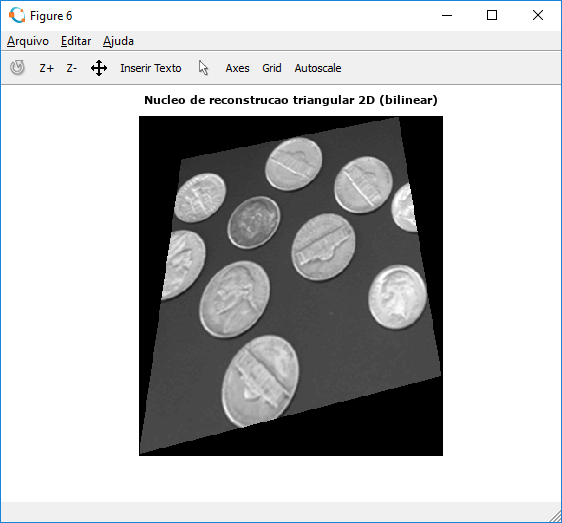
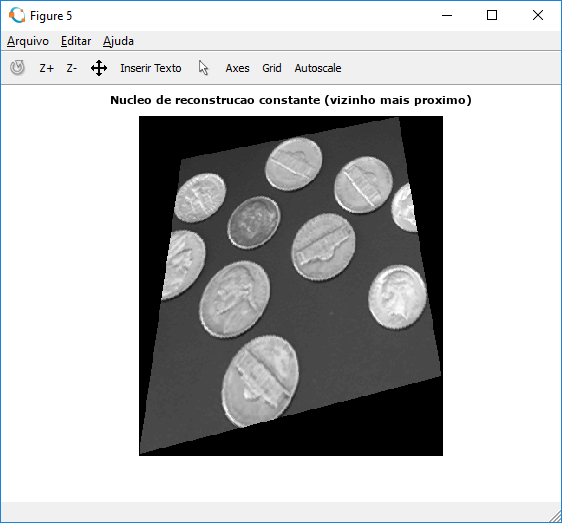
a l 300 350

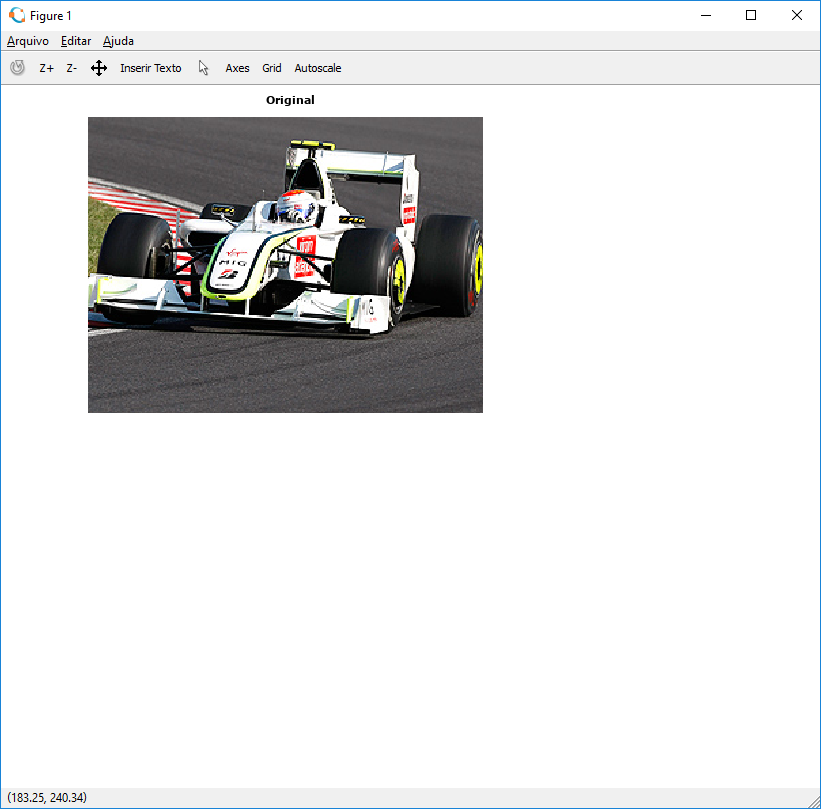
];

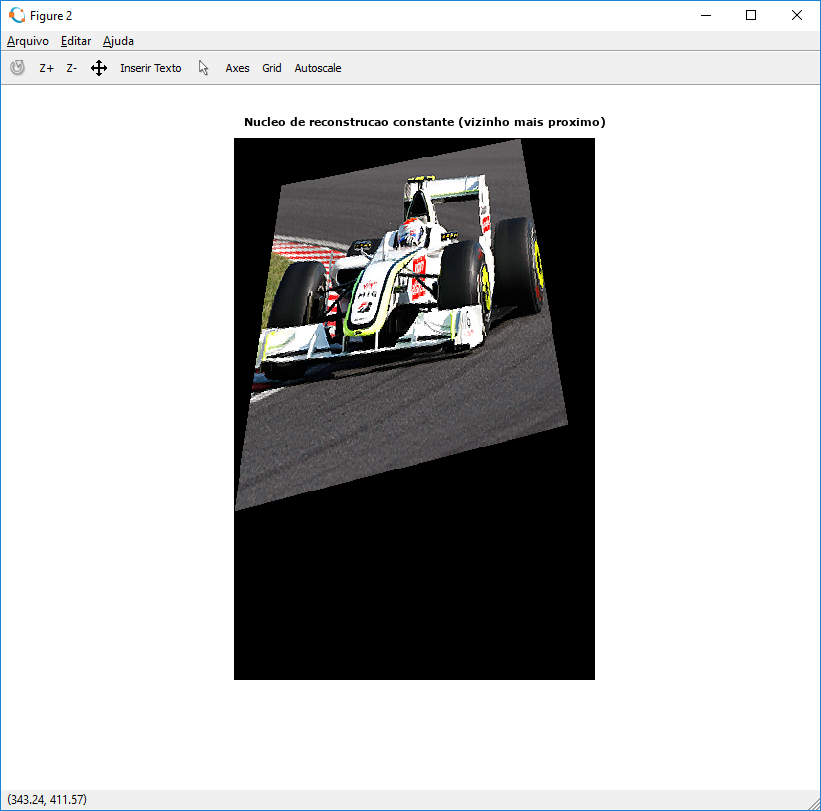
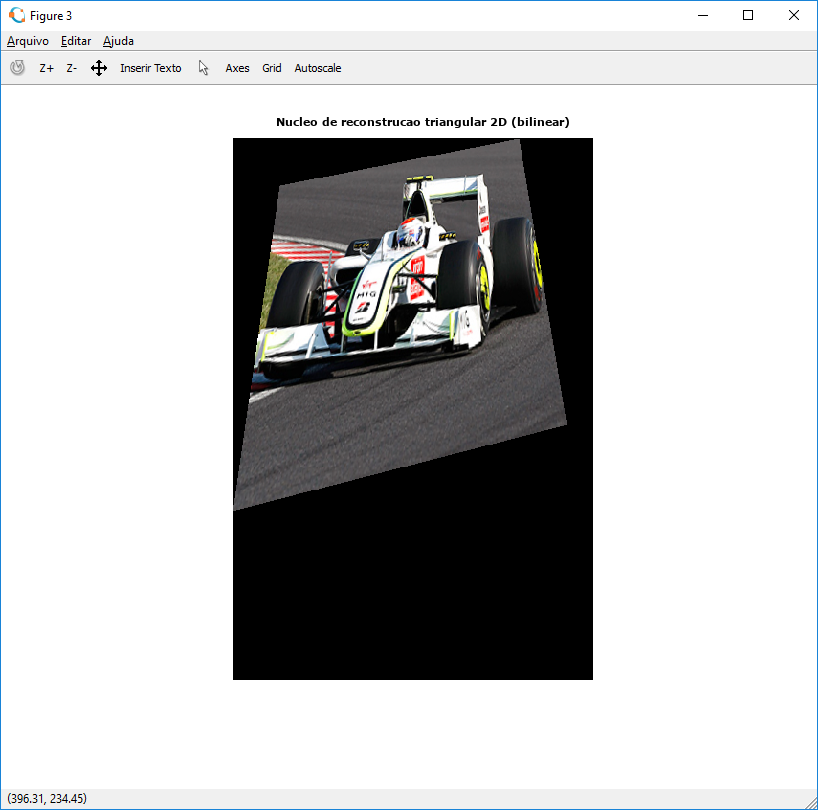
transformar(caminho\_img, M);

**Exemplos:**

****

****

****

****