Segundo Trabalho de Implementação Visão Computacional

|  |  |  |
| --- | --- | --- |
| Ícaro Goulart Faria Motta França | Semestre 01/17 |  |

# Problema

Implementar o algoritmo do Shape from Shading Linear de Pentland, nas versões das equações (10) e (11) do artigo de referência, testando a implementação com as imagens fornecidas:

## Introdução

Neste trabalho usamos 2 funções nativas do Matlab para realizarmos as transformadas de Fourier e a Transformada Inversa de Fourier.

A função fft2 (2-D Fast Fourier Transform) do Matlab foi usada para fazer a Transformada de Fourier (F) das imagens.

A função ifft2 (2-D Inverse Fast Fourier Transform) do Matlab foi usada para fazer a Transformada de Fourier Inversa (f) da frequência da imagem.

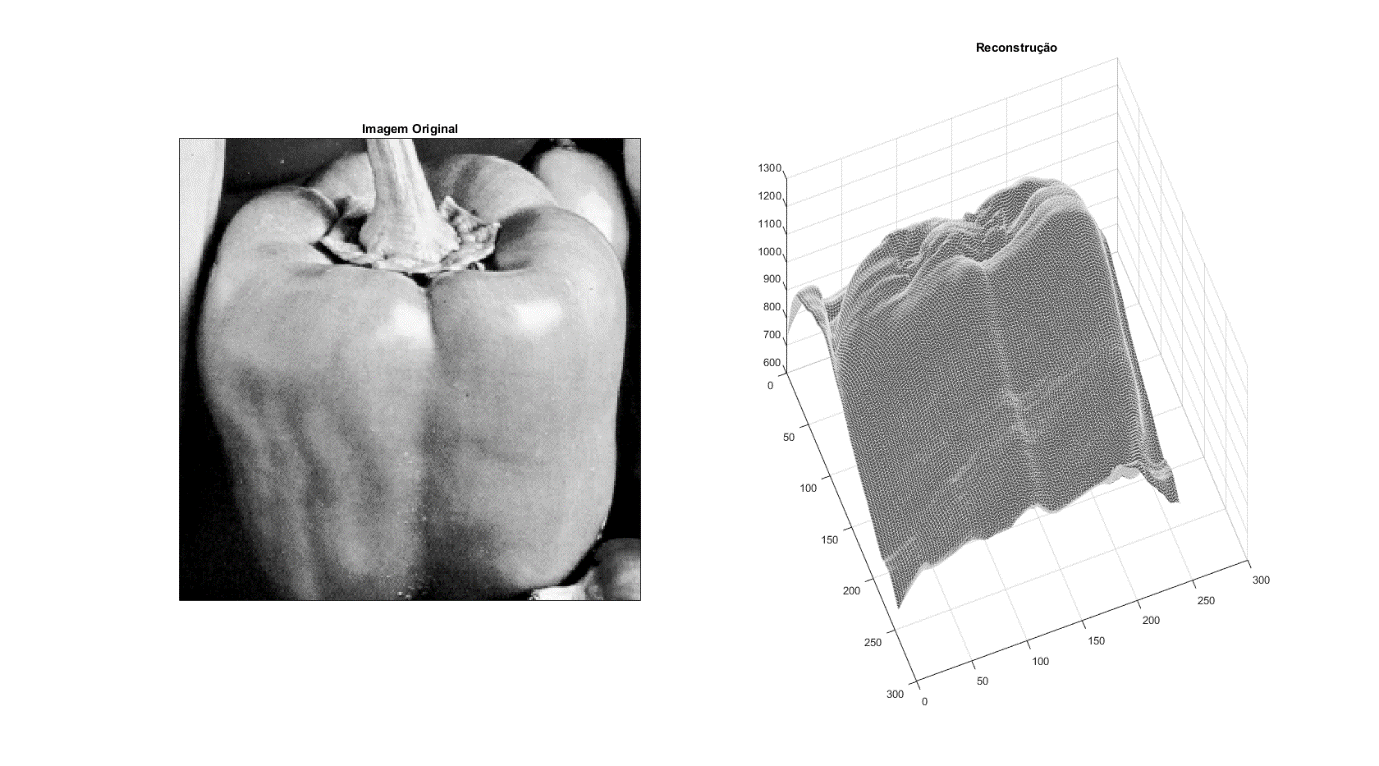
As imagens de entrada e saída foram colocadas na sessão correspondente a cada equação de Pentland implementada. Também foram agrupadas numa tabela de 1 linha e 2 colunas, em que a primeira coluna refere-se ao domínio espacial, a segunda à reconstrução da imagem em 3D.

## Equação 10 de Pentland

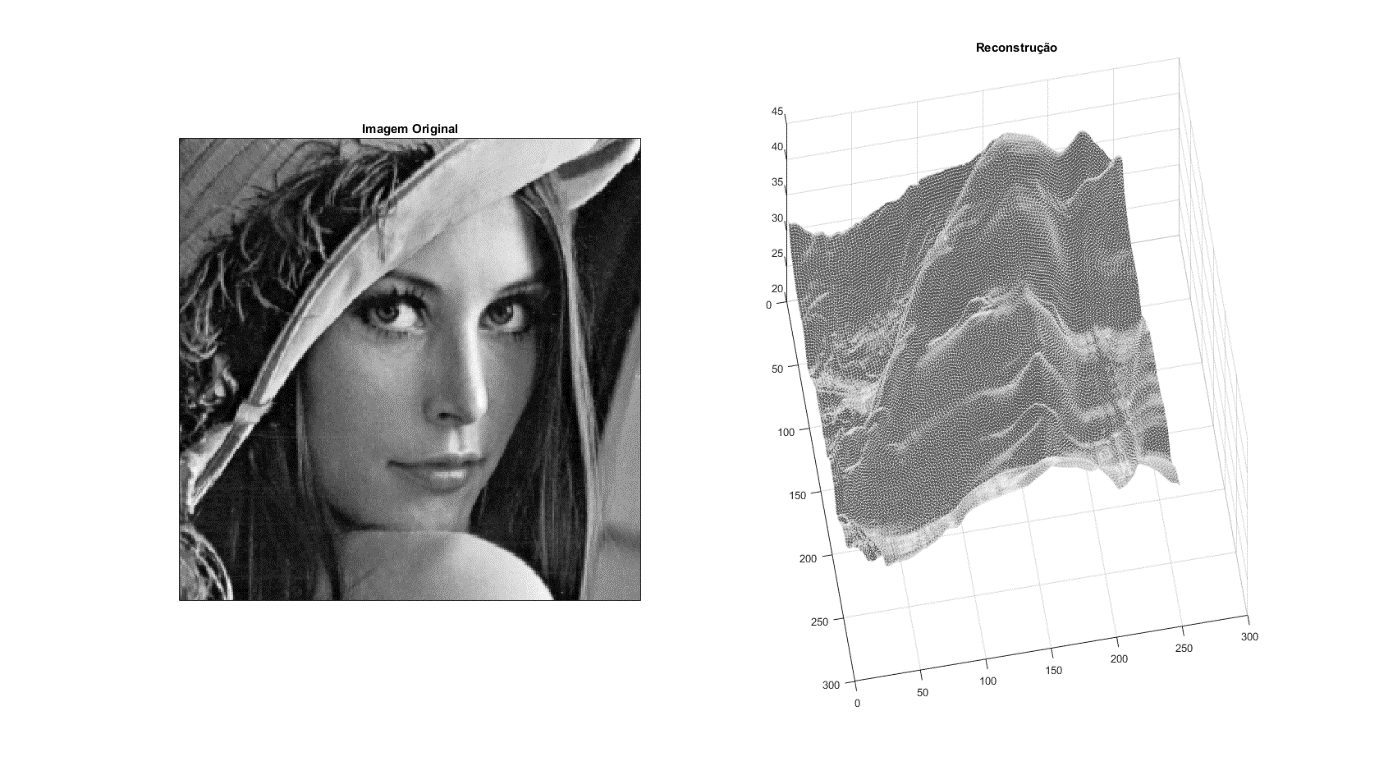
* Primeira imagem:
  + O slant usado nessa equação foi 1.
  + O tilt usado foi 20.
* Segunda imagem:
  + O slant usado nessa equação foi 22.
  + O tilt usado foi 30.

Obs: O slant mesmo possuindo um valor alto não influenciou muito na reconstrução.

### Primeira Imagem: pepino



### Segunda Imagem: Lena



### Código:

clear all;

%Lê a imagem original do arquivo

isPepper = 1;

filename1 = 'pepper256.png';

filename2 = 'lena.png';

if (isPepper == 1)

filename = filename1;

else

filename = filename2;

end

I = imread(filename);

%Recupera só um canal da imagem caso a imagem não esteja em tons de cinza

if size(I, 3) == 3

I = double(rgb2gray(I));

else

I = double(mat2gray(I));

end

%configurando plot

originalImage = I;

figure(1);

subplot(1,2,1);

imagesc(originalImage);

axis square;

colormap gray(256);

title('Imagem Original');

set(gca, 'XTick', [], 'YTick', []);

%transformada de Fourier de I

Fi = fft2(I);

%recuperamos o número de linhas e colunas da imagem

[M, N] = size(I);

%damos valor para os angulos slant e tilt

slant = degtorad(22);

tilt = degtorad(30);

if (isPepper == 1)

slant = degtorad(1);

tilt = degtorad(20);

end

%Modo 2 de calcular

[x,y] = meshgrid(1:N, 1:M);

%preenchemos uma matriz discretizada com vários valores para wx e wy

wx = (2.\*pi.\*x) ./ N;

wy = (2.\*pi.\*y) ./ M;

k1 = cos(tilt) \* sin(slant);

k2 = sin(tilt) \* sin(slant);

F3 = abs(Fi);

Fz2 = (abs(Fi) ./ (k1\*wx + k2\*wy)) .\* exp(1i\*(angle(Fi) - pi/2));

%computamos a transformada inversa de fourier para recuperar a superfície

Z2 = abs(ifft2(Fz2));

subplot(1,2,2);

mySurf =surfl(Z2);

h = rotate3d;

h.RotateStyle = 'box';

h.Enable = 'on';

colormap gray(256);

title('Reconstrução');

shading faceted;

set(gca,'Ydir','reverse');

set(gca,'Xdir','normal');

set(gca,'Zdir','normal');

%camlight('headlight');

set(mySurf,'LineStyle',':');

mySurf.EdgeColor = [0.9 0.9 0.9];

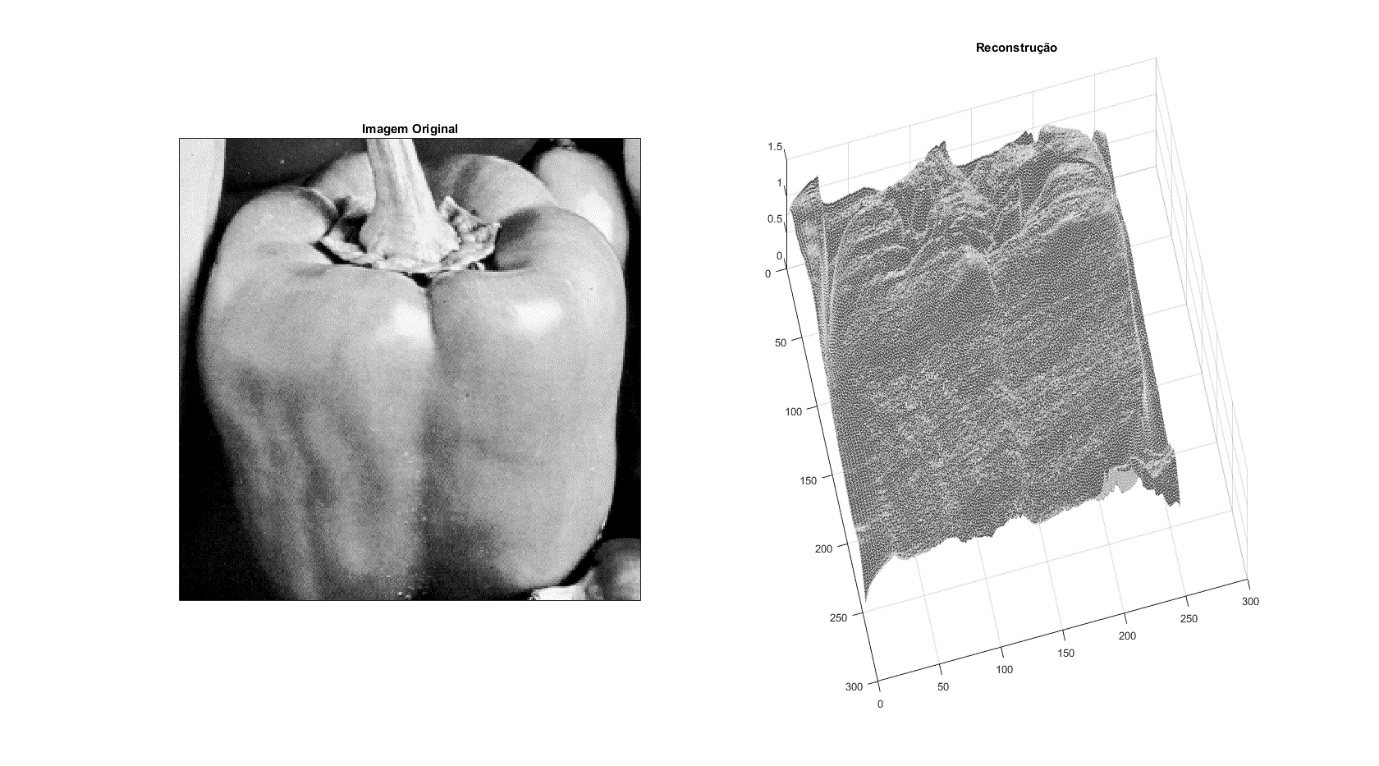
mySurf.FaceColor = 'flat';

## Equação 11 de Pentland

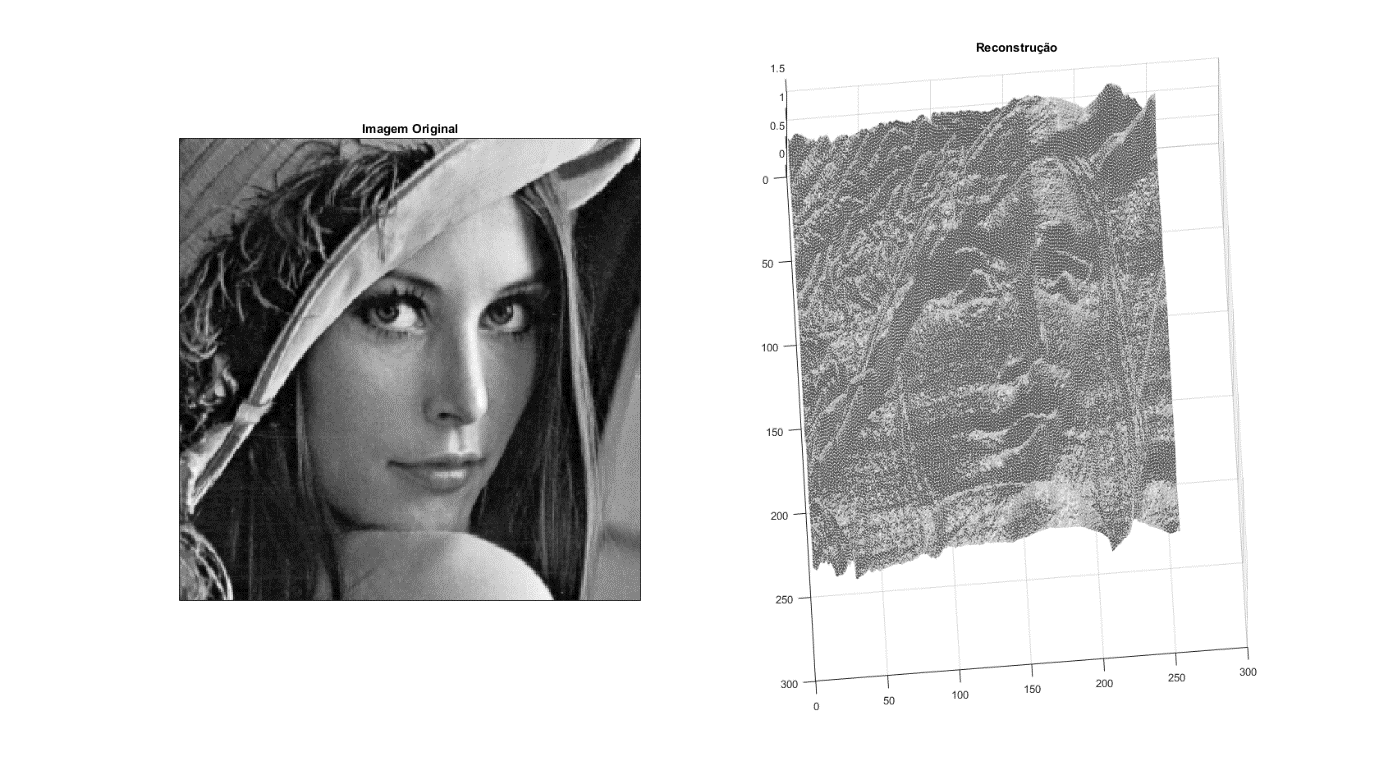
* Primeira imagem:
  + O slant usado nessa equação foi 76.
  + O tilt usado foi 35.
* Segunda imagem:
  + O slant usado nessa equação foi 85.
  + O tilt usado foi 55.

Observação: Provavelmente essa equação não foi implementada corretamente, porque os valores nesse caso foram altos.

### Primeira Imagem: pepino



### Segunda Imagem: Lena



### Código:

clear all;

%Lê a imagem original do arquivo

isPepper = 0;

filename1 = 'pepper256.png';

filename2 = 'lena.png';

if (isPepper == 1)

filename = filename1;

else

filename = filename2;

end

I = imread(filename);

%Recupera só um canal da imagem caso a imagem não esteja em tons de cinza

if size(I, 3) == 3

I = double(rgb2gray(I));

else

I = double(mat2gray(I));

end

%configurando plot

originalImage = I;

figure(1);

subplot(1,2,1);

imagesc(originalImage);

axis square;

colormap gray(256);

title('Imagem Original');

set(gca, 'XTick', [], 'YTick', []);

%transformada de Fourier de I

Fi = fft2(I);

%recuperamos o número de linhas e colunas da imagem

[M, N] = size(I);

%damos valor para os angulos slant e tilt

slant = degtorad(85);

tilt = degtorad(55);

if (isPepper == 1)

slant = degtorad(76);

tilt = degtorad(35);

end

%Modo 2 de calcular

[x,y] = meshgrid(1:N, 1:M);

%preenchemos uma matriz discretizada com vários valores para wx e wy

sinSlant = sin(slant);

wx = (2.\*pi\*sinSlant.\*x) ./ N;

wy = (2.\*pi\*sinSlant.\*y) ./ M;

k1 = cos(tilt) \* sin(slant);

k2 = sin(tilt) \* sin(slant);

%criar uma matriz para s que tem todas as frequencias.

angleX = (2.\*pi.\*x) ./ N;

angleY = (2.\*pi.\*y) ./ M;

s = sign(cos(tilt - (angleX + angleY)));

% 0.5 < d < 0.75

d = 0.5; %se coloco um d < que 0.5 o ruido fica menor e a imagem tende a ficar melhor ?

Fz2 = (abs(Fi) ./ (s\*d + (k1\*wx + k2\*wy))) .\* exp(1i\*(angle(Fi) - pi/2));

%computamos a transformada inversa de fourier para recuperar a superfície

Z2 = abs(ifft2(Fz2));

subplot(1,2,2);

mySurf =surfl(Z2);

h = rotate3d;

h.RotateStyle = 'box';

h.Enable = 'on';

colormap gray(256);

title('Reconstrução');

shading faceted;

set(gca,'Ydir','reverse');

set(gca,'Xdir','normal');

set(gca,'Zdir','normal');

%camlight('headlight');

set(mySurf,'LineStyle',':');

mySurf.EdgeColor = [0.9 0.9 0.9];

mySurf.FaceColor = 'flat';