UNIVERSIDADE FEDERAL DE OURO PRETO - UFOP
DEPARTAMENTO DE COMPUTAÇÃO - DECOM

Disciplina: PROCESSAMENTO DE IMAGENS - BCC 326
Prof.: Guillermo Cámara Chávez
Curso: Ciência da Computação

LISTA 05

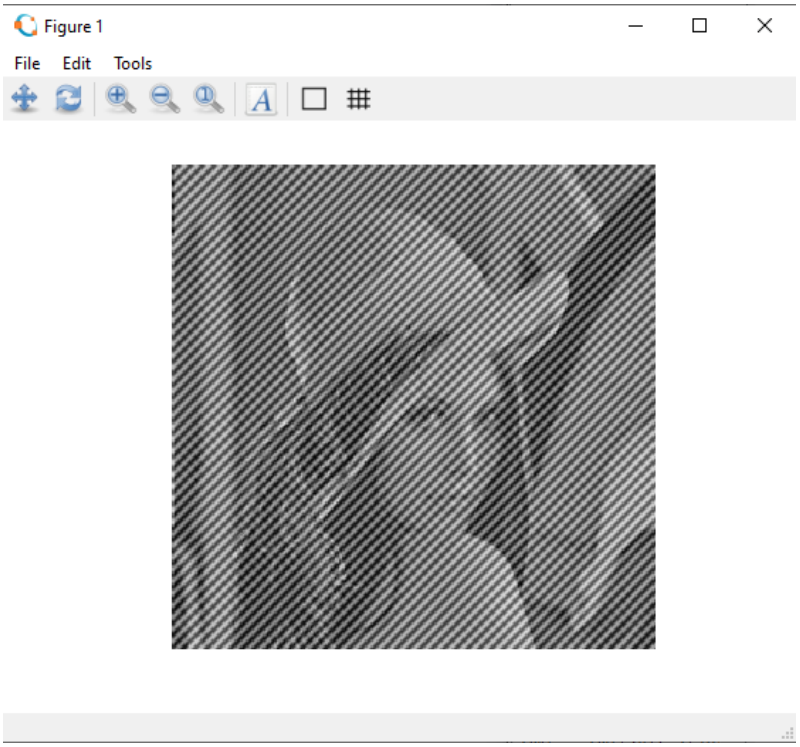
Gustavo Lucas Moreira[1]
gustavo.lucas@aluno.ufop.edu.br

17.2.4289

1. Utilize a filtragem no domínio da frequência, primeiro calcule o espectro de Fourier (Figura b) e elimine a região ao redor dos “spikes” (assinalados com a seta vermelha). Os “spikes” estão localizados nas coordenadas (88,88) e (170,170).

Operações:

```
>> img = imread('lenna_periodico2.png');
>> imshow(img);
```



[1] Graduando do Curso de Ciência da Computação da Universidade Federal de Ouro Preto

```
>> [linha, coluna, ~] = size(img);
>> matriz= fftshift(fft2(img));
>> matriz(88-5:88+5,88-5:88+5)=0;
>> imshow(log(abs(matriz)+1),[]);
```

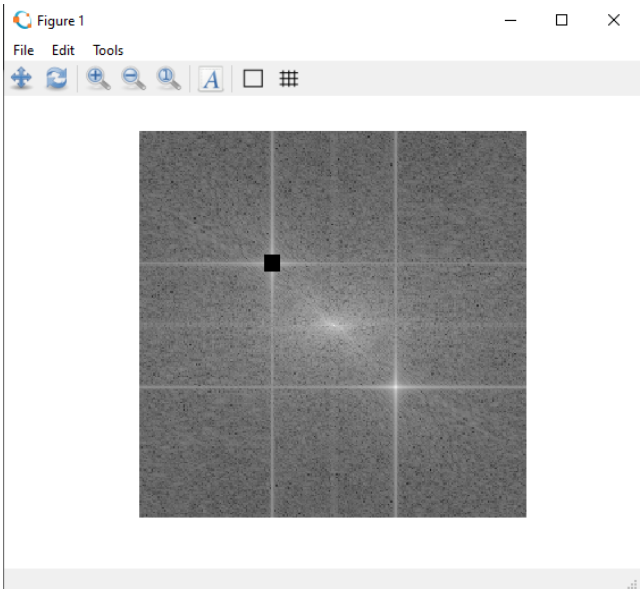
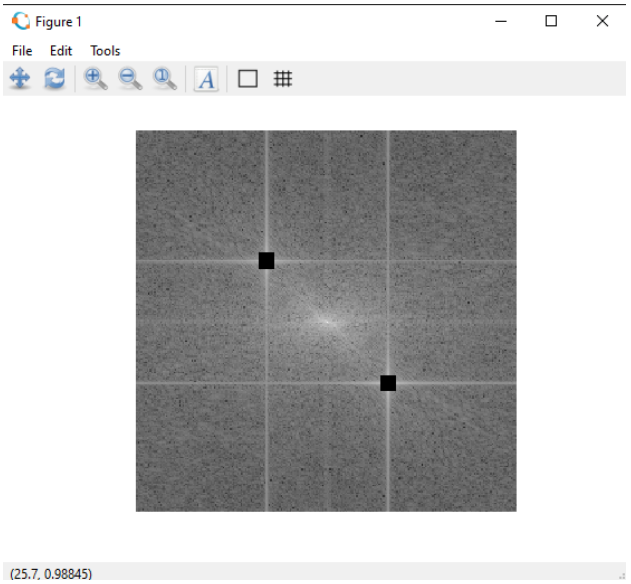
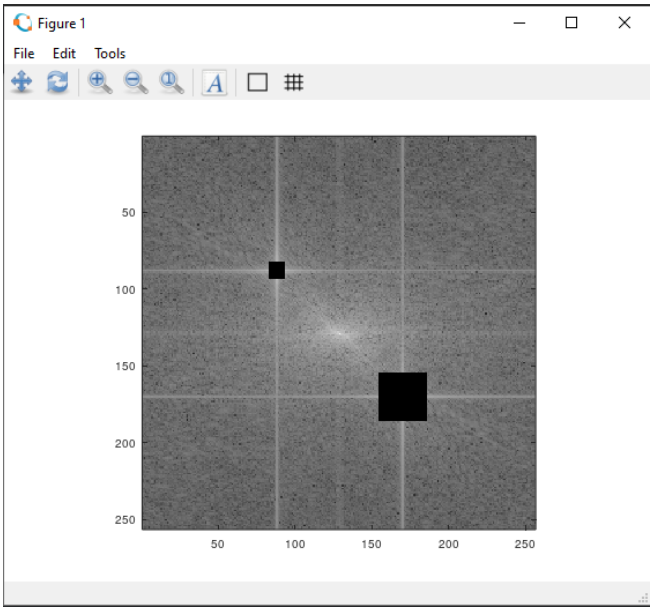


Imagem resultado.

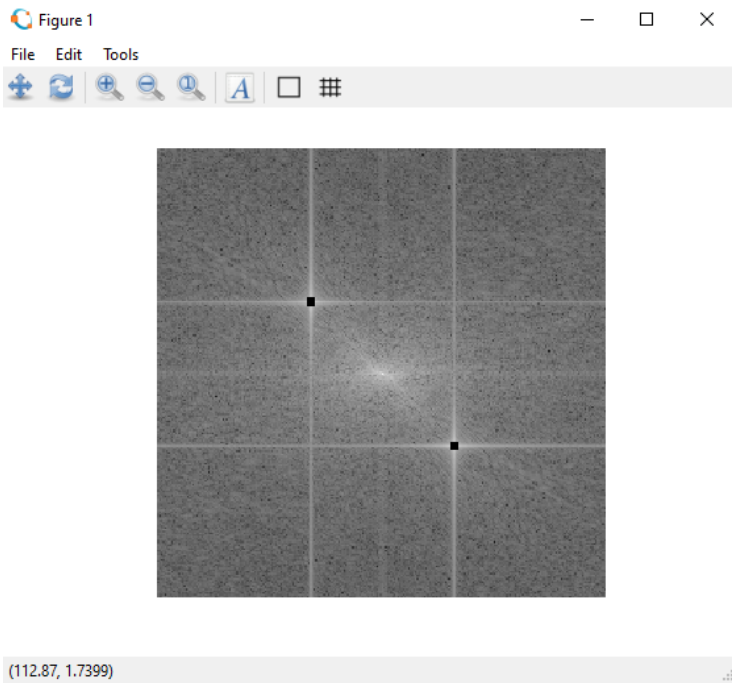
```
>> matriz(170-5:170+5,170-5:170+5)=0;
>> imshow(log(abs(matriz)+1),[]);
```




```
>> matriz(88-5:88+5,88-5:88+5)=0;
>> matriz(170-15:170+15,170-15:170+15)=0;
>> imshow(log(abs(matriz)+1),[]);
```




```
>> matriz = fftshift(fft2(img));
>> matriz(88-2:88+2,88-2:88+2)=0;
>> matriz(170-2:170+2,170-2:170+2)=0;
>> imshow(log(abs(matriz)+1),[]);
```



2. Filtro notch: São filtros capazes de rejeitar uma faixa bastante estreita de frequências. Sua utilização é recomedada quando o sinal a ser atenuado é bem definido

```
function nimg = notch(img, d0)
  [lin, col] = size(img);
```

```
  u0 = (lin / 2 - 88);
  v0 = (col / 2 - 88);

  d1 = zeros(lin, col);
  for i = 1 : lin
    for j = 1 : col
      d1(i, j) = sqrt((i - lin /2 -u0)^2 + (j-col / 2 - v0) ^ 2);
    end
  end
  d2 = zeros(lin, col);
  for i = 1 : lin
    for j = 1 : col
      d2(i, j) = sqrt((i - lin / 2 + u0)^2+(j-col/2+v0) ^ 2);
    end
  end

  h = zeros(lin, col);
  for i = 1 : lin
    for j = 1 : col
      h(i, j) = 1 - exp((-1 / 2) * ((d1(i, j) * d2(i,j))/(d0^2)));
    end
  end

  nimg = h .* fftshift(fft2(img));
  nimg = abs(ifft2(nimg));
end
```

3. Repita o processo de remoção do ruído periódico da questão anterior utilizando os seguintes filtro passa-bandas: ideal, Butterworth e Gaussiano.

Filtro Ideal:

```
function nimg = ideal(img, d0, w)
  [lin, col] = size(img);
    d = zeros(lin, col);
  for i = 1 : lin
    for j = 1 : col
      d(i, j) = sqrt(lin^ 2 + col^ 2);
    end
  end

  h = zeros(lin, col);
```

```
  for i = 1 : lin
    for j = 1 : col
        if ((d(i,j) >= (d0 - w/2)) && (d(i,j) <= (d0 + w/2)))
            h(i, j) = 0;
        else
            h(i, j) = 1;
        end
    end
  end

  nimg = h .* fftshift(fft2(img));
  nimg = abs(ifft2(nimg));
end
```

Filtro Butterworth

```
function nimg = butterworth(img, d0, w, n)
  [lin, col] = size(img);

  d = zeros(lin, col);
  for i = 1 : lin
    for j = 1 : col
      d(i, j) = sqrt(lin^ 2 + col^ 2);
    end
  end

  h = zeros(lin, col);
  for i = 1 : lin
    for j = 1 : col
      h(i, j) = 1 / 1 +((d(i, j) * w)/(d(i, j) ^2 - d0 ^
2))^(2*n);
    end
  end

  nimg = h .* fftshift(fft2(img));
  nimg = abs(ifft2(nimg));
end
```

Filtro Gaussiano

```
function nimg = gaussiano(img, d0, w)
  [lin, col] = size(img);
  d = zeros(lin, col);
  for i = 1 : lin
```

```
    for j = 1 : col
      d(i, j) = sqrt(lin.^ 2 + col.^ 2);
    end
  end

  h = zeros(lin, col);
  for i = 1 : lin
    for j = 1 : col
      h(i, j) = 1 - exp(-((d(i, j) ^ 2 - d0 ^ 2)/(d(i, j)*w))^2);
    end
  end

  nimg = h .* fftshift(fft2(img));
  nimg = abs(ifft2(nimg));
end
```