Relatório do laboratório 2 de Processamento Digital de Imagens

Aluno: Tomás Rosário Rosemberg 14/0087567

Questão 1)(Arquivos dentro da pasta Questão1)

Para a letra A foi criada a função para filtrar a imagem “filtrogeral.m”, porém, como ele utiliza 5 for’s para efetuar a filtragem, acaba demorando bastante, desta forma foi criada a função “filtro3X3.m” para usar os filtros requeridos na imagem “lena512color.tiff”. A filtragem da imagem resultou nos arquivos “LetraA.png”, “LetraB.png”, “LetraC.png”, “LetraD.png” e “LetraE.png”, os quais foram utilizados os filtros 1,2,3,4,5 respectivamente.

```
function [novaimg] = filtro3X3 (imagem, filter)
  img = imread(imagem);           %le imagem
  tamimg = size(img);             %vetor com o tamanho da imagem
  tamfil = size(filter);          %vetor com tamanho do filtro

  sumfilter = sum(sum(filter));    %multiplicando o vetor pelo inverso da soma da matriz
  if sumfilter ~= 0
    filter = filter/sumfilter;
  end
  filter = flip(filter,1);         %duas linhas para espelhar a matriz, primeira
  filter = flip(filter,2);         %espelha em relacao as linhas e seguna as colunas

  imgtemp = zeros(tamimg(1) + 2*tamfil(1) , tamimg(2) + 2*tamfil(2), tamimg(3));%cria uma imagem temporaria com o tamanho da imagem mais duas vezes o tamanho do filtro de altura e largura

  imgtemp (tamfil(1)+1:tamimg(1) + tamfil(1),tamfil(2)+1:tamimg(2) + tamfil(2), 1:tamimg(3))...
  =img(1:tamimg(1),1:tamimg(2),1:tamimg(3)); %preenche a imagem temporaria para que tenha uma borda ao redor da imagem original, sendo tal borda do tamanho do filtro

  novaimg = zeros(tamimg(1),tamimg(2),tamimg(3));%cria a nova imagem que sera a filtrada
  novaimg = double(novaimg);
  img = double(img);
  imgtemp = double(imgtemp);
  for(prof = 1:1:tamimg(3)) %for variando o canal da imagem de 1 a 3
    for(linha = tamfil(1)+1:1:tamimg(1) + tamfil(1)) %for para varrer as linhas
      for(coluna = tamfil(2)+1:1:tamimg(2) + tamfil(2)) %for para varrer as colunas
        novaimg(linha-tamfil(1),coluna-tamfil(2),prof) = imgtemp(linha-1,coluna-1,prof)* filter(1,1)+imgtemp(linha-1,coluna,prof)* filter(1,2)+imgtemp(linha-1,coluna+1,prof)* filter(1,3)...
        +imgtemp(linha,coluna-1,prof)* filter(2,1)+imgtemp(linha,coluna,prof)* filter(2,2)+imgtemp(linha,coluna+1,prof)* filter(2,3)...
        +imgtemp(linha+1,coluna-1,prof)* filter(3,1)+imgtemp(linha+1,coluna,prof)* filter(3,2)+imgtemp(linha+1,coluna+1,prof)* filter(3,3);
      end
    end
  end
  novaimg = uint8(novaimg);
  img = uint8(img);
  imgtemp = uint8(imgtemp);
  imshow(img);
  figure;
  imshow(novaimg);
  imwrite(novaimg,"passaalta.png");
endfunction
```

Imagem 1)Filtro3x3.m

```
function [novaimg] = filtrogeral (imagem, filter)
  img = imread(imagem);           %le imagem
  tamimg = size(img);             %vetor com o tamanho da imagem
  tamfil = size(filter);          %vetor com tamanho do filtrogeral

  sumfilter = sum(sum(filter));    %multiplicando o vetor pelo inverso da soma da matriz
  if sumfilter ~= 0
    filter = filter/sumfilter;
  end
  filter = flip(filter,1);         %duas linhas para espelhar a matriz, primeira
  filter = flip(filter,2);         %espelha em relacao as linhas e seguna as colunas

  imgtemp = zeros(tamimg(1) + 2*tamfil(1) , tamimg(2) + 2*tamfil(2), tamimg(3));%cria uma imagem temporaria com o tamanho da imagem mais duas vezes o tamanho do filtrogeral de altura e largura

  imgtemp (tamfil(1)+1:tamimg(1) + tamfil(1),tamfil(2)+1:tamimg(2) + tamfil(2), 1:tamimg(3))...
  =img(1:tamimg(1),1:tamimg(2),1:tamimg(3)); %preenche a imagem temporaria para que tenha uma borda ao redor da imagem original, sendo tal borda do tamanho do filtrogeral

  novaimg = zeros(tamimg(1),tamimg(2),tamimg(3));%cria a nova imagem que sera a filtrada
  for(prof = 1:1:tamimg(3)) %for variando o canal da imagem de 1 a 3
    for(linha = tamfil(1)+1:1:tamimg(1) + tamfil(1)) %for para varrer as linhas
      for(coluna = tamfil(2)+1:1:tamimg(2) + tamfil(2)) %for para varrer as colunas
        for(x=1:1:tamfil(1))%for para varrer o tamanho do filtrogeral em linha
          for(y=1:1:tamfil(2)) %for para varrer o tamanho do filtrogeral em coluna
            novaimg(linha-tamfil(1),coluna-tamfil(2),prof) = novaimg(linha-tamfil(1),coluna-tamfil(2),prof)+imgtemp(linha-ceil(tamfil(1)/2)+x,coluna-ceil(tamfil(2)/2)+y,prof)* filter(x,y);
          end
        end
      end
    end
  end
  imshow(img);
  figure;
  novaimg = uint8(novaimg);
  imshow(novaimg);
  imwrite(novaimg,"imagemfiltrada.png")
endfunction
```


Imagem 2)Filtrogeral.m

Imagem 3)Imagens resultantes da aplicação do filtro 1,2,3 e 4 respectivamente.

Imagem 4) Imagem resultante da aplicação do filtro 5

Podemos perceber que os filtros 1 e 2 são filtros passa-baixa, pois acabam agindo em zonas de alta frequência, onde ha uma mudança muito brusca de valores na matriz, amenizando essa mudança utilizando os valores próximos.

O filtro 3 é um filtro de alto-reforço.

Podemos perceber que o filtro 4 é um filtro passa-alta, que transforma em zero os valores de frequência baixos, ou seja, zera lugares onde há pouca variações de valores e realça lugares onde há muita variação de valores.

O filtro 5 é uma máscara de nitidez.

Caso o filtro fosse com o m maior, a imagem perderia mais e mais seus detalhes, conforme podemos perceber se olharmos a imagem ”Filtro11X11.png”. O arquivo “filtro11X11.png” foi feito para gerar a imagem ”Filtro11X11.png” pois utilizar a função filtrogeral iria demorar muito para terminar de rodar no computador.

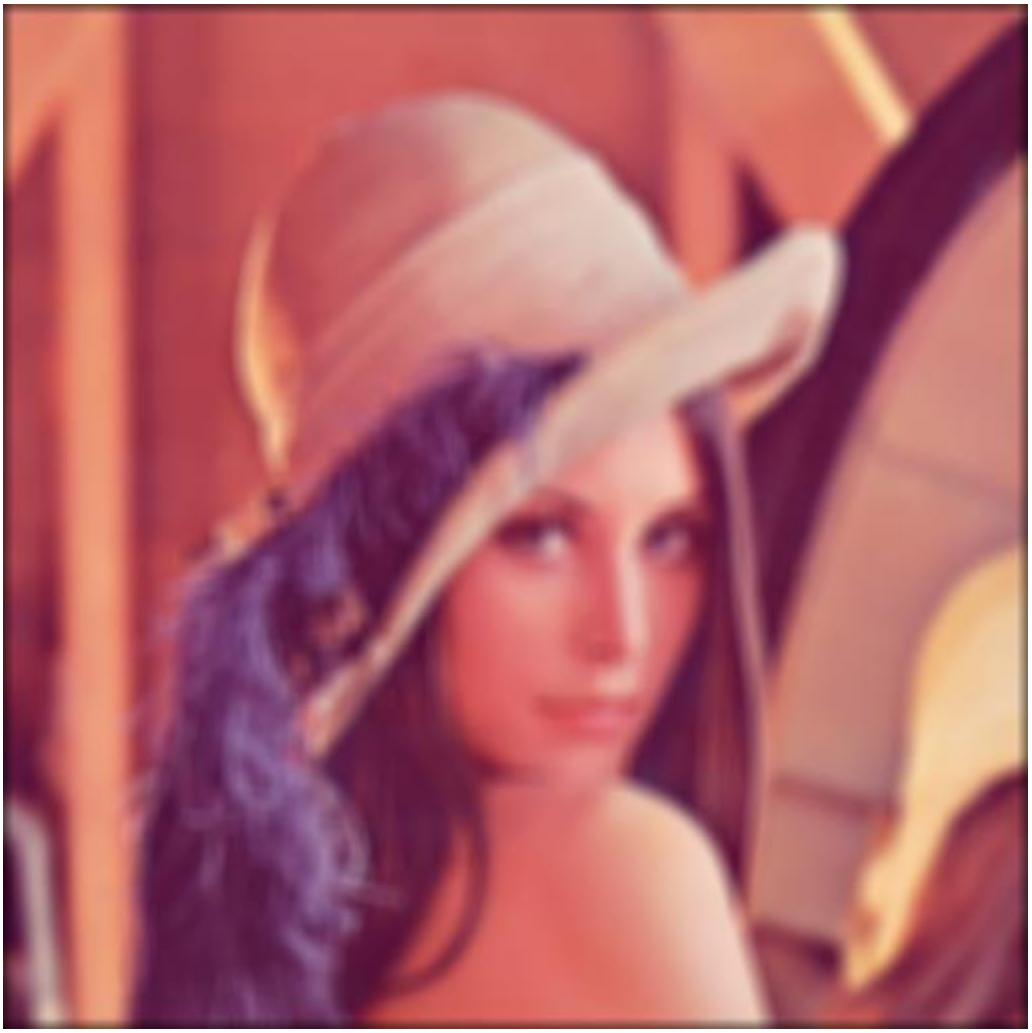

Imagem 5)Imagem após a aplicação de um filtro 11x11

Questão2) (arquivos dentro da pasta Questão2)

A função que faz a redução e a aumenta a resolução espacial usando o método do ponto mais próximo é a função “dimensionamentoMPV.m”, desta forma foram geradas as reduções por 2 vezes, 4 vezes, 8 vezes e 16 vezes que acarretaram nas imagens "reduz2X.png", "reduz4X.png", "reduz8X.png" e "reduz16X.png" respectivamente. Após isso, ampliamos novamente as imagens reduzidas pelo mesmo fator, gerando as imagens “MVPamplia2x.png”, “MVPamplia4x.png”, “MVPamplia8x.png” e“MVPamplia16x.png”. Este procedimento todo acarretou numa grande perda da qualidade da imagem, pois houve o descarte do valor de vários pixels e na ampliação houve a cópia de valores dos pixels ao invés de reobtenção dos valores descartados.

```
function [novaimg] = dimensionamentoMVP (imagem, dime)
  img = imread(imagem);
  tamimg = size(img);
  novaimg = zeros(tamimg(1)*dime,tamimg(2)*dime,tamimg(3));
  tamnimg = size(novaimg);
  img = double(img);
  novaimg = double(novaimg);
  if(dime>1)
    for(x=1:dime)
      for(y=1:dime)
        novaimg(y:dime:tamnimg(1),x:dime:tamnimg(2),1:1:tamnimg(3))=img(1:tamimg(1),1:tamimg(2),1:1:tamnimg(3));
      end
    end
  elseif(dime == 1)
    novaimg = img;
  else
    novaimg(1:1:tamnimg(1),1:1:tamnimg(2),1:tamnimg(3))=img(1:(1/dime):tamimg(1),1:(1/dime):tamimg(2),1:tamimg(3));
  end
  img = uint8(img);
  novaimg = uint8(novaimg);
endfunction
```

Imagem 6) DimensionamentoMVP.m

Imagem 7) Reduzida 2 vezes e reduzida 4 vezes

Imagem 8) Reduzida 8 e 16 vezes


Imagem 9) Ampliada pelo método do ponto mais próximo 2 vezes e 4 vezes respectivamente

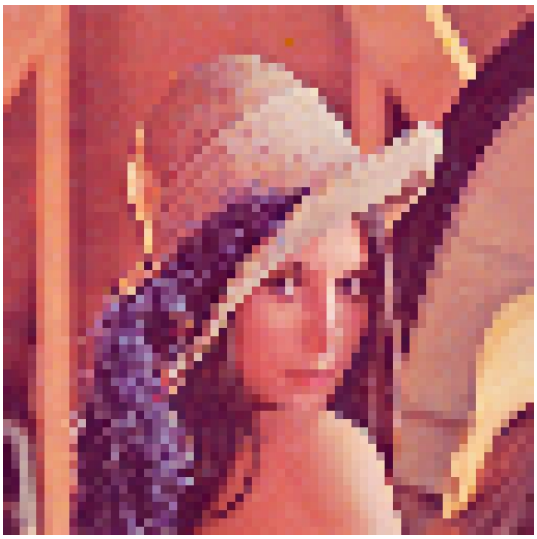


Imagem 10) Ampliada pelo método do ponto mais proximo 8 vezes e 16 vezes respectivamente

A função para redimensionamento das imagens usando o método bilinear é a função “dimensionamentoBILINEAR.m”, que gerou como saída as imagens “BIamplia2x.png”, “BIamplia4x.png”, “BIamplia8x.png” e “BIamplia16x.png”. A partir delas podemos perceber claramente uma melhora na imagem em relação a ampliação usando o método do vizinho mais próximo.

```
function [novaimg] = dimensionamentoBILINEAR (imagem, dime)
  img = imread(imagem);
  tamimg = size(img);
  imgtemp = zeros(tamimg(1)+dime,tamimg(2)+dime,tamimg(3));
  imgtemp(1:tamimg(1),1:tamimg(2),1:tamimg(3)) = img(1:tamimg(1),1:tamimg(2),1:tamimg(3));
  novaimg = zeros(tamimg(1)*dime,tamimg(2)*dime,tamimg(3));
  tamnimg = size(novaimg);
  img = double(img);
  novaimg = double(novaimg);
  if(dime>1)
    for(y=1:dime)
      novaimg((y:dime:tamnimg(1)),1:dime:tamnimg(2),1:1:tamnimg(3)) = ((dime+1-y)/dime)*img(1:tamimg(1),1:tamimg(2),1:1:tamnimg(3)) + ((y-1)/dime)*imgtemp(2:tamimg(1)+1,1:tamimg(2),1:1:tamnimg(3));
    end
    for(prof = 1 : 1 : tamnimg(3))
      for(coluna = 1: dime: tamnimg(2)-dime)
        for(linha = 1 : 1 : tamnimg(1))
          for(x=1:dime)
            novaimg(linha,coluna+x-1,prof) = ((dime+1-x)/dime)*novaimg(linha,coluna,prof) + ((x-1)/dime)*novaimg(linha,coluna+dime,prof);
          end
        end
      end
    end
    %novaimg(1:dime:tamnimg(1),1:dime:tamnimg(2),1:tamnimg(3)) = img(1:tamimg(1),1:tamimg(2),1:tamimg(3));
  elseif(dime == 1)
    novaimg = img;
  else
    novaimg(1:1:tamnimg(1),1:1:tamnimg(2),1:tamnimg(3))=img(1:(1/dime):tamimg(1),1:(1/dime):tamimg(2),1:tamimg(3));
  end
  img = uint8(img);
  novaimg = uint8(novaimg);
endfunction
```

Imagem 11) DimensionamentoBILINEAR.m

Imagem 12) Ampliada pelo método bilinear 2 vezes e 4 vezes respectivamente

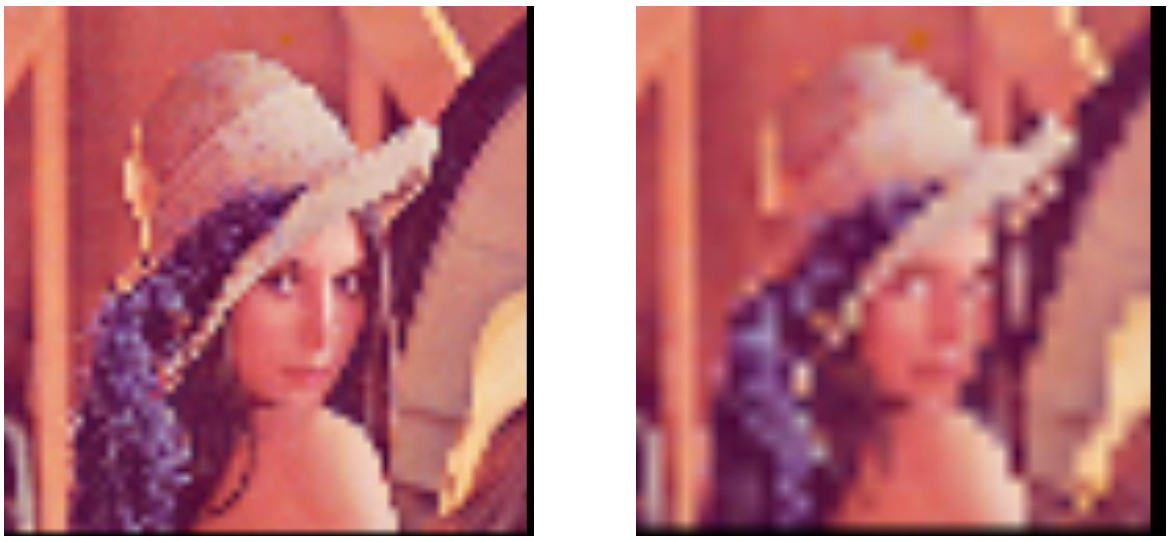

Imagem 13) Ampliada pelo método bilinear 8 vezes e 16 vezes respectivamente