Nome: Yuri Tavares dos Passos
Disciplina: Visão Computacional

Respostas da Lista de exercícios 2

Exercícios Teóricos

Questão 1:

Ruído são valores aleatórios que surgem em cada pixel da imagem. São valores que não são necessários para o cômputo de algoritmos para imagens e que muitas vezes atrapalham nos resultados.

Questão 2:

- Ruido aleatório: devido a natureza discreta da luz, a carga elétrica convertida pelo CCD pode apresentar variações.
- Ruido térmico: devido ao movimento térmico dos átomos, e a flutuação rápida e se origina na existência de estados de vida longa na interface silício-óxido.
- Ruido de cargas espúrias: A alteração de potenciais para a leitura introduz cargas espúrias.
- Corrente de escuro: na ausência de luz, pares elétrons-buracos se formam dentro do sensor, geralmente com uma dependência exponencial da temperatura.
- Erros devido à discretizacão: O ganho de um CCD é um parâmetro que pode alterar no circuito eletrônico de saída, e determina quantos elétrons coletados em cada pixel são necessários para cada contagem na imagem.
- Franjas de interferência: se o comprimente de onda da luz incidente for da mesma ordem que a espessura do CCD, ou interferências pela reflexão de luz quase monocromática dentro do CCD, ocorrem franjas de interferência, que normalmente têm a mesma forma mas com amplitude proporcional à intensidade da luz incidente.
- Efeitos cosméticos no CCD: pixels ruins, píxeis quentes, colunas mortas e bits congelados podem causar problemas na calibração e análise das imagens.

Questão 3:

Filtros Lineares podem ser representados pela convolução de uma matriz A pela matriz I, correspondente a imagem. A operação de convolução de I por A é dada por $I * A$. A operação convolução é linear pois:

$$\alpha[(I + J) * A] = \alpha I * A + \alpha J * A \tag{1}$$

Filtros não-lineares não obedecem estas afirmações. O filtro da mediana é um exemplo, pois não é possível separar linearmente filtros da mediana como na Equação 1.

Questão 4:

Filtros que minimizam o erro quadrático podem ser encontrados resolvendo sistemas lineares. Esta técnica é aplicada para respostas arbitárias de frequências. Além disso, restrições lineares sobre os coeficientes são facilmente inclusas.

As vantagens são:

- O filtro é ótimo no que diz respeito ao critério de erro quadrático.
- Método simples e não-iterativo
- Soluções análiticas possíveis em alguns casos. No casos em que isso não é possível, pode ser obtida por um sistema de equações lineares.
- Permite o uso de uma função dependente de frequência.
- Fácil de incluir restrições arbritárias lineares.

Para construir um filtro usando este método deve-se seguir os seguintes passos

1. Especificar N, o tamanho do filtro.
2. Estabelecer a frequência de corte.
3. Construir uma matriz Q de tal forma que $a = Q^{-1}b$ será o filtro desejado

Questão 5:

O algoritmo de Roberts pode ser resumido nos seguintes passos:

1. Aplique o filtro Gaussiano para remoção dos ruídos
2. Filtre a imagem resultante com as seguintes máscaras:

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

   obtendo as imagens I e J.

3. Estime os gradiantes das duas imagens como sendo

$$G(i,j) = \sqrt{I(i,j)^2 + J(i,j)^2}$$

4. Marque os pontos em que $G(i,j) > t$, onde t é o limiar.

Com este algoritmo, caso haja muito ruído, a o limiar t deve ser diminuido, pois este detector é muito sensível a ruídos. A Figura 1 apresenta a aplicação do detector Roberts.

Exercícios Práticos

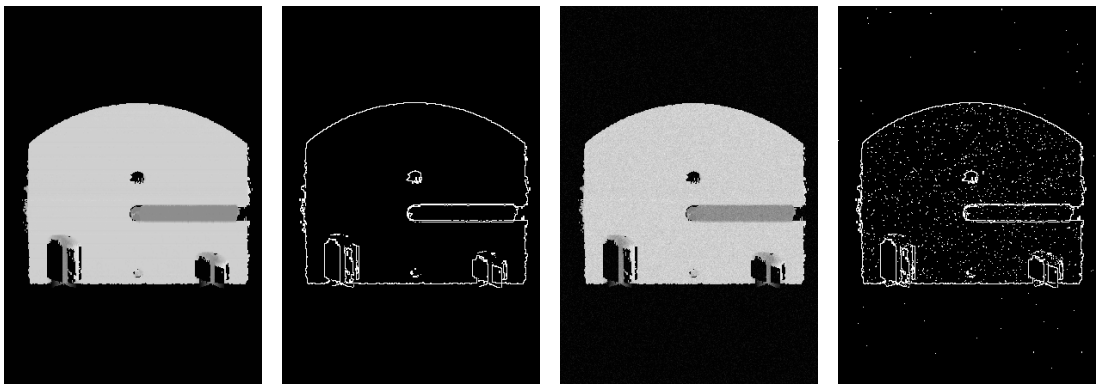
Questão 1:

Abaixo o código para Scilab que gera a máscara para uma dimensao.

```
// WEIGHTED LEAST SQUARE LOWPASS FILTER

// filter length
N = 5;
M = (N-1)/2;

// set band-edges and stop-band weighting
wp = 1.999;
ws = 2.001;
K = 10;
```



(a) Imagem original (b) Bordas detectadas de 1(a) com t=20 (c) Imagem 1(a) com ruído (d) Bordas detectadas da imagem 1(c) com t=20

Figura 1: Detector de bordas Roberts

```
// normalize band-edges for convenience
fp = wp/%pi;
fs = ws/%pi;

// construct q(k)
q  = [fp+K*(1-fs), fp*sinc(fp*[1:2*M])-K*fs*sinc(fs*[1:2*M])];


// construct Q1, Q2, Q
Q1 = toeplitz(q([0:M]+1));
Q2 = hank(size(Q1,1),size(Q1,2),[q([0:M]+1),q([M:2*M]+1)]);
Q  = (Q1 + Q2)/2;

// construct b
b  = fp*sinc(fp*[0:M]');

// solve linear system to get a(n)
a  = Q\b;

// form impulse response h(n)
h  = [a(M+1:-1:2)/2; a(1); a(2:M+1)/2];
```

A máscara h é convoluida para linhas e h^T nas colunas. A frequencia de corte é aproximadamente 2 rad/s. O código de filtragem usando esta máscara foi:

```
I=gray_imread('lenna_noise.jpg');
I2 = zeros(I);
for i=1:size(I,1)
  Y = convol(h',I(i,:));
  I2(i,:) = Y(1+2:512+2);
end
for j=1:size(I,2)
  Y = convol(h',I2(:,j)');
  Z = Y';
  I2(:,j) = Z(1+2:512+2);
end
imwrite(normal(I2,1,0),'lenna_filtro1.png');
```

Para o filtro de Butterworth foi usado o seguinte código:

```
n=1;
wc = 2;
```

(a) Imagem original (b) Com filtro dos mínimos quadrados (c) Com filtro Butterworth

Figura 2: Filtragens para frequencia de corte igual a 2 rad/s

```
I=gray_imread('lenna_noise.jpg');
If = fft(I);
for i=1:size(If,1)
  for j=1:size(If,1)
    w=2*%pi * (i+j)/(size(If,1) + size(If,2));
    B(i,j) = 1/sqrt(1+(w/wc)^(2*n));
  end
end

I2f = B.*If;
I2=real(ifft(I2f));
imwrite(I2,'lenna_butterworth.png');
```

A Figura 2 mostra os resultados para ambos

Exercícios de Pesquisa

Questão 1:

SIFT é lento e não é bom em mudanças de iluminação, mas é invariante a rotação, mudanças de escala e transformações afins. SURF é rápido e tem boa performance assim como o SIFT possue, mas não é estável a rotação e mudanças de iluminação.