

## Relatório projeto III – Danilo Celestino – 201207140064

### PARTE I

- a) O código exemplo lê a imagem utilizada e atribui a matriz correspondente à variável  $I$ . Em seguida carrega-se a matriz correspondente ao filtro *Unsharp* na variável  $h$ . A função *imfilter* faz a correlação entre o filtro e a imagem original, e o resultado é armazenado na variável  $I2$ . Em seguida é exibida a imagem original e a filtrada em janelas diferentes.

b)  $h = \begin{bmatrix} -0.15 & -0.6 & -0.15 \\ -0.6 & 3.9 & -0.6 \\ -0.15 & -0.6 & -0.15 \end{bmatrix}$

- c) O filtro unsharp do Matlab/Octave possui um padrão de números 1.5, 6 e 39 divididos por 9, onde o maior número é o central e é positivo, os demais são negativos e os menores valores ficam nos "cantos". Os números negativos é o que ressalta o contraste e os números menores torna o efeito de borda mais brando. O filtro  $h$  criado possui o mesmo padrão, mas dividido por 10. A seguir há a imagem original e a filtrada, respectivamente.

Original Image



Filtered Image



- d) Em termos de sinais, o filtro é similar a um sinal invertido de menor frequência e baixa amplitude que ao ser correlacionado com a imagem original cancela parte dos sinais em baixa-frequência do sinal original. Portanto, é um filtro passa-alta.

### PARTE II

Os passos foram seguidos e estão relativamente bem explicados no código *p2.m* em anexo. Os resultados obtidos para a imagem *coins.jpg* pode ser vista a seguir:

Ruído aplicado	Valor de M	Média	Mediana
Salt & Pepper (density 0.06)	3	29.7764	36.3901
	5	29.2048	35.8182
	7	29.0081	35.5364
Salt & Pepper (density 0.005)	3	33.2324	36.5725
	5	32.0533	35.8738
	7	31.1211	35.5656
Gaussian (variância 0.001)	3	33.4224	34.7432
	5	32.3458	34.7182
	7	31.2729	34.4999
Gaussian (variância 0.03)	3	28.6805	29.1662
	5	29.0041	30.0494
	7	28.9486	30.4061

À escala decimal os valores obtidos pelo código podem apresentar certa variação, provavelmente devido a limitações computacionais.

**1) Por que para o ruído salt & pepper o filtro de mediana é tão melhor que o de média?**

Para o ruído Salt & Peper, que é um ruído que afeta alguns pixels em dada região, a mediana é mais eficiente pois é menos sensível as discrepâncias causadas por este ruído, se comparado a média, que é um filtro linear. Como no ruído gaussiano quase todos os pixels são afetados, ele perde eficiência, mas ainda assim produz resultados mais fiéis que o filtro de média.

**2) A PSNR é uma figura de mérito objetiva, mas analisando as imagens de forma subjetiva, qual o filtro que conserva melhor as bordas dos objetos na imagem?**

Em geral, o filtro de mediana e com ordens menores ( $M=3$ , por exemplo) preservam mais as bordas. É possível chegar a esta conclusão analisando as imagens processadas por cada filtro.

**3) Quando o ruído é Gaussiano e a PSNR é relativamente alta, os filtros não parecem ajudar muito (melhorar a PSNR). Mas quando a PSNR é baixa, eles parecem melhorar. Você concorda? Se sim, qual o motivo deste**

**comportamento? Se não, em qual aspecto o raciocínio está incompleto?**

A PSNR é a relação sinal ruído de pico. Quando o ruído é gaussiano a PSNR é menor que quando o ruído é Salt & Peper (e quanto maior a variância, pior). A PSNR é uma mensurável de sucesso do filtro em recuperar a imagem original. Não é a PSNR ser alta que torna o filtro ineficiente, e sim o contrário: Se o filtro for adequado ao tipo de ruído presente no sinal a PSNR se torna alta.

**4) Pensando no custo computacional dos processos de filtragem, qual o filtro que demanda mais poder de processamento, o de média ou mediana?**

A Mediana necessita de cerca de 3x mais processamento que o filtro de média. Esta informação foi obtida através do tempo de processamento necessário para o Matlab/Octave realizar cada função.