UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA

INF01046 - Fundamentos de Processamento de Imagens

Laboratório Aula 18

1) Faça o download dos scripts "lab_18_01.m" a "lab_17_04.m" e das imagens "book.jpg", "coins.jpg" do link da disciplina e salve-os no diretório "work" do MATLAB.

2) **scripts lab_18_01 e lab_18_02:**

Os scripts lab_18_01 e lab_18_02 exemplificam a implementação do filtro de Wiener (lab_18_01) e do Método de mínimos quadráticos com restrição (lab_18_02).

Estes scripts fazem o seguinte:

- Carregam uma imagem sem degradar
- Calculam a dimensão do padding considerando o tamanho da máscara a ser utilizada
- Aplicam o padding na imagem original
- Geram uma mascara de degradação no domínio espacial
- Geram uma imagem de ruido domínio espacial

- Calcula a DFT da imagem sem degradar, mascara e ruido (todas com padding)

No domínio da frequência:

- Multiplica a DFT da imagem pela função de degradação para obter a DFT da imagem degradada sem ruido.

- Soma o ruido para obter a DFT da imagem degradada com ruido.

- Aplica a filtragem inversa (divide pela função de degradação)

- Aplica o filtro de Wiener (lab_18_01) ou o Método de mínimos quadráticos com restrição (lab_18_02) ambos para vários valores dos parâmetros.

- Apresenta os resultados

É interessante que estude como é gerada uma função de degradação (filtro) no domínio da frequência partindo de uma mascara no domínio espacial.

Em resumo, a função de degradação no domínio da frequência é a transformada de Fourier da máscara no domínio espacial (teorema da convolução).

Os scripts apresentam a forma correta de fazer esto considerando os detalhes de padding e centrado da máscara.

Os passos seguidos são:

1- Determinar o tamanho da máscara no domínio espacial ex. 3x3
2- Calcular o tamanho da imagem com padding (soma o tamanho da imagem mais o tamanho da mascara e procura a dimensão par mais próxima.)
3- Cria a máscara no domínio espacial com a dimensão da imagem mais padding e centrada.
4.- Aplica ifftshit na máscara criada para levar o centro ao origem de coordenadas
5.- Calcula a DFT da máscara

Exercício :
Proponha uma máscara para simular borramento causado por movimento linear da câmera (~21 pixels na direção diagonal da imagem)
- Faça um desenho em papel da máscara centrada com detalhe das áreas de padding, centrado etc.
- Faça um desenho em papel da máscara depois de mover o centro para a origem de coordenadas.

3) scripts lab_18_03 e lab_18_04:

O scripts lab_18_03 e lab_18_04 exemplificam com realizar transformações geomêtricas.

No script **lab_18_03** é utilizada uma função de transformação de coordenadas da forma:

$$
\begin{aligned}
x' &= x \\
y' &= y_0 + \alpha \cdot (y - y_0)
\end{aligned}
$$

Onde x e y são as coordenadas na imagem original e x' e y' na imagem transformada.

Como devemos calcular o valor de cada pixel na imagem transformada necessitamos da expressão inversa que expresse x e y como função de x' e y'.

$$
\begin{aligned}
x &= x' \\
y &= y_0 + \frac{1}{\alpha} \cdot (y' - y_0)
\end{aligned}
$$

Então para cada pixel da imagem transformada calculamos as coordenadas correspondentes na imagem original.

O script utiliza como algoritmo de interpolação, vizinhos mais cercanos.

Uma linha comentada no código permite especificar um valor de alfa para cada x, criando um efeito interessante.

Exercício avançado: Implemente uma função que simule uma lente sobre uma parte da imagem.

O script **lab_18_04** implementa um algoritmo, tomando 4 pontos da imagem original e mapeando estes em outros 4 pontos da imagem transformada.

A forma da transformação proposta e bilinear nas coordenadas x e y:

$$x = c_1 \cdot x' + c_2 \cdot y' + c_3 \cdot x' \cdot y' + c_4$$
$$y = c_5 \cdot x' + c_6 \cdot y' + c_7 \cdot x' \cdot y' + c_8$$

Especificando 4 pontos de referência na imagem original e mapeando estes para 4 pontos na imagem transformada, obtemos a 8 equações:

$$x_0 = c_1 \cdot x'_0 + c_2 \cdot y'_0 + c_3 \cdot x'_0 \cdot y'_0 + c_4$$
$$y_0 = c_5 \cdot x'_0 + c_6 \cdot y'_0 + c_7 \cdot x'_0 \cdot y'_0 + c_8$$

$$x_1 = c_1 \cdot x'_1 + c_2 \cdot y'_1 + c_3 \cdot x'_1 \cdot y'_1 + c_4$$
$$y_1 = c_5 \cdot x'_1 + c_6 \cdot y'_1 + c_7 \cdot x'_1 \cdot y'_1 + c_8$$

$$x_2 = c_1 \cdot x'_2 + c_2 \cdot y'_2 + c_3 \cdot x'_2 \cdot y'_2 + c_4$$
$$y_2 = c_5 \cdot x'_2 + c_6 \cdot y'_2 + c_7 \cdot x'_2 \cdot y'_2 + c_8$$

$$x_3 = c_1 \cdot x'_3 + c_2 \cdot y'_3 + c_3 \cdot x'_3 \cdot y'_3 + c_4$$
$$y_3 = c_5 \cdot x'_3 + c_6 \cdot y'_3 + c_7 \cdot x'_3 \cdot y'_3 + c_8$$

Em forma matricial , este sistema de equações fica:

$$\begin{pmatrix} x_0 \\ y_0 \\ x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \end{pmatrix} = \begin{pmatrix} x'_0 & y'_0 & x'_0 \cdot y'_0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x'_0 & y'_0 & x'_0 \cdot y'_0 & 1 \\ x'_1 & y'_1 & x'_1 \cdot y'_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x'_1 & y'_1 & x'_1 \cdot y'_1 & 1 \\ x'_2 & y'_2 & x'_2 \cdot y'_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x'_2 & y'_2 & x'_2 \cdot y'_2 & 1 \\ x'_3 & y'_3 & x'_3 \cdot y'_3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x'_3 & y'_3 & x'_3 \cdot y'_3 & 1 \end{pmatrix} . \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \end{pmatrix}$$

No script : A = D . C

No matlab resolvemos o sistema de equações dividindo ambos membros por D (esto somente é possível se D é inversível , ou o que é a mesma coisa se o sistema possui uma única solução)

C = D \ A

Tendo calculado os coeficientes C (c1 a c8) podemos aplicar a transformada:

$$x = c_1 \cdot x' + c_2 \cdot y' + c_3 \cdot x' \cdot y' + c_4$$
$$y = c_5 \cdot x' + c_6 \cdot y' + c_7 \cdot x' \cdot y' + c_8$$

Que especifica as coordenada de cada ponto na imagem original correspondente a cada ponto da imagem transformada.

O script utiliza como algoritmo de interpolação, vizinhos mais cercanos.

Exercício:
Desenvolva um algoritmos similar que utilize como referência 3 pontos em cada imagem.