Image Deblurring by Frequency Domain Inverse Filter Design

Asked 3 days ago Active 2 days ago Viewed 52 times

1

I have a task about using a Gaussian Filter to blur an image then using the corresponding inverse Gaussian Filter to deblur the blurred image. The detailed instructions are shown below:

Apply 2D-DFT to the spatial domain Gaussian filter, resulting in a frequency domain Gaussian filter. Create a frequency domain inverse Gaussian filter by taking the reciprocal value of all frequency domain Gaussian filter coefficients. Apply inverse 2DDFT and take the real-parts to create a spatial domain inverse Gaussian filter (the imaginary parts are theoretically 0, but may not be completely 0 due to numerical errors).

The size of the blurring filter is 21x21, and the size of the original image is 256x256. Here is my blurring code on Matlab.

```
Img = imread('text.tif');
img = im2double(Img);
gf = zeros(21);
for i = 1:21
    for j = 1:21
        gf(i,j) = (1/(2*pi))*exp(-((i-10.5)^2+(j-10.5)^2)/2);
    end
end
Convig = conv2(img,gf);
```

The output image is shown below. Blurred Image

Then following the instructions, my deblurring code is shown below:

```
dftgf =fft2(gf);
idftgf = 1./dftgf;
inverse = ifft2(idftgf);
gf2 =real(inverse);
Convig2 = conv2(Convig,gf2);
```

The result of blurring seems to be correct. But it is strange that the deblurring image is incorrect Deblurring Attempt. Could someone help me with my code? Thanks in advance.

matlab image-processing gaussianblur

edited 2 days ago
MichaelTr7
834 2 9

asked Oct 18 at 17:22
Rosaria
21 2

1 Answer

Active Oldest Votes

1

Okay finally I found the mistake. For the formula of Guassian filter, it is said that

```
gf(i,j) = (1/(2*pi*sigma^2))*exp(-((i-center)^2+(j-center)^2)/(2*sigma^2);
```

but here I used "10.5" directly. If I use "11" as the center, the result image is correct.

So the deblurring pseudo code should be:

```
[m,n] = size(filter)
centerx = m+1/2
centery = n+1/2
gf(i,j) = (1/(2pi*sigma^2))*exp(-((i-centerx)^2+(j-centery)^2)/(2*sigma^2);
```

However, even if I found the mistake, I still do not know why it cannot be decimals. Maybe image processing is a kind of discrete programming?

answered 2 days ago

Rosaria

21 2

New contributor

Nice catch, usually the centre filter kernels has to be an integer number. During convolution, the centre pixel determines the position of the resultant. Here is a gif showing the process commons.wikimedia.org/wiki/File:2D_Convolution_Animation.gif. – MichaelTr7 2 days ago

As an aside, the 11th pixel is the centre pixel since to it's left there will be 10 pixels and to it's right there will another 11 pixels. – MichaelTr7 2 days ago