Digital Image Processing Mini Project #4

Image Restoration

Seyed Hesamoddin Hosseini

خلاصه

معمولاً در فرایند تصویربرداری، تصویر به دلایل مختلفی ممکن است تار شود. تاری ممکن است به دلیل حرکت دوربین، حرکت شیء، دیفوکوس، شرایط جوّی و یا عوامل دیگر باشد. معمولاً تاری ناشی از حرکت شیء در حین تصویربرداری با دو پارامتر طول و زاویه حرکت تعریف می‌شود. چنانچه فقط احتمال حرکت در جهت عمودی و یا افقی وجود داشته باشد، تنها پارامتر توصیف کننده حرکت، اندازه و جهت آن است. در هر سیستم تصویربرداری، یک فاصله کانونی تعریف می‌شود و نقاطی از صحنه که فاصله متفاوتی تا دوربین دارند، دچار تاری دیفوکوس می‌شوند.

بناست در مرحله اول این مینی پروژه، تصویری را که به علت حرکت، به صورت خطی تاره شده است را ترمیم نموده و تخمینی از تابع خراب آن بدست آوریم. این عمل را یک بار با در نظر گرفتن تصویر اصلی و یک بار با فرض اینکه تصویر اصلی را نداریم، ترمیم می کنیم.

در مرحله دوم این مینی پروژه، تصاویری داریم که دچار تاری دی فوکوس شده اند. به کمک دو تصویر موجود تلاش می کنیم توابع تاری را تخمین بزنیم و سپس تصویر رفع تارشده را بدست آوریم.

برای پیاده سازی این پروژه از Matlab استفاده می نماییم.

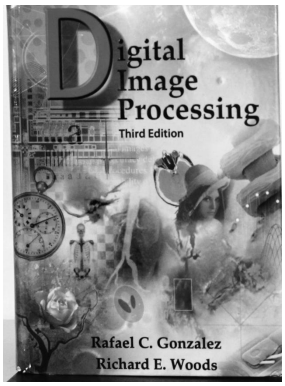
شرح تکنیکال:

مرحله اول:

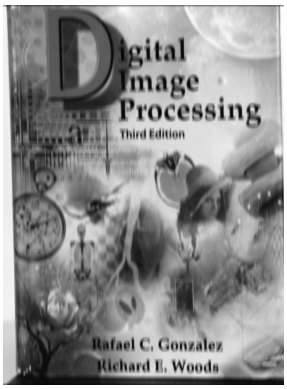
با توجه به توضیحات داده شده معمولاً در فرایند تصویربرداری، تصویر به دلایل مختلفی ممکن است تار شود. در اینجا فرض بر این است که تاری ممکن است به دلیل حرکت دوربین یا حرکت شیء باشد. در ابتدا همانطور که بیان کردیم هدف از ایجاد این سامانه این است که تصویری را که به علت حرکت، به صورت خطی تاره شده است را ترمیم نموده و تخمینی از تابع خراب آن بدست آوریم.

حالت الف:

در این حالت فرض می کنیم هم تصویر تار را داریم و هم تصویر اصلی را داریم. می خواهیم به کمک این دو تصویر، تابع خرابی را تخمین بزنیم.



شکل 1 - تصویر اصلی



شکل 1 - تصویر تار شده ناشی از حرکت

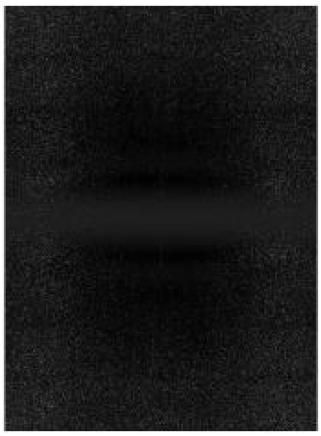
می دانیم که رابطه بین تصویر اصلی و تصویر تخریب شده در حوزه مکان و حوزه فرکانس به ترتیب به صورت زیر می باشد:

$$g(x,y) = h(x,y) * f(x,y) + \eta(x,y)$$
$$G(u,v) = H(u,v)\,F(u,v) + N(u,v)$$

از آنجا که عمل ضرب نسبت به عمل کانوالو کردن، راحت تر می باشد، تصاویر بالا را به حوزه فرکانس می بریم.
در اینجا فرض می کنیم هیچ نویزی وجود ندارد. پس به کمک فرمول زیر، سعی می کنیم تابع خرابی را بدست آوریم:

$$H_s(u,v) = \frac{G_s(u,v)}{F_s(u,v)}$$

نتیجه حاصل تا اینجای کار به صورت زیر می باشد:



شکل 3 - تابع خرابی در حوزه فرکانس

قطعه کد مربوط به صورت زیر می باشد:

```
Fuv = fftshift(fft2(fxy));
Guv = fftshift(fft2(gxy));

Huv = Guv ./ Fuv;
```

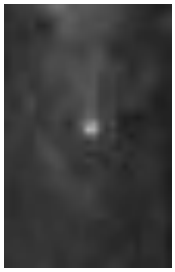
حالت ب:

در این حالت فرض می کنیم که تصویر اصلی (شکل 1) را نداشته باشیم و تنها تصویر تار (شکل 2) را در اختیار داشته باشیم، با دو مجهول تابع تاری و تصویر اولیه مواجه هستیم. در واقع، حالا با blind image deconvolution مواجه هستیم. در این حالت، پیشنهادی برای تخمین تابع تاری ارائه می دهیم.

طرح پیشنهادی: تخمین مدل خرابی با استفاده ازمشاهده

باید قسمتی از تصویر تار را برش بزنیم. به طوری که بتوانیم حالت تار نشده ی آن را به راحتی بازسازی نماییم.
بدین منظور بخشی از تصویر که شبیه نقطه ای سفید رنگ در پس زمینه سیاه رنگ می باشد را crop می نماییم و اصلاح شده ی آن را شبیه سازی می نماییم.
نتیجه به صورت زیر می باشد:

شکل 4- بخشی از تصویر تار شده



شکل 5 - بازسازی تصویر فوق

از آنجا که عمل ضرب نسبت به عمل کانوالو کردن، راحت تر می باشد، تصاویر بالا را به حوزه فرکانس می بریم.
در اینجا فرض می کنیم هیچ نویزی وجود ندارد. پس با تقسیم دو تصویر بر یکدیگر، سعی می کنیم تابع خرابی را بدست آوریم:

```
zF = fftshift(fft2(zf));
zG = fftshift(fft2(zg));

zH = zF ./ zG;
```

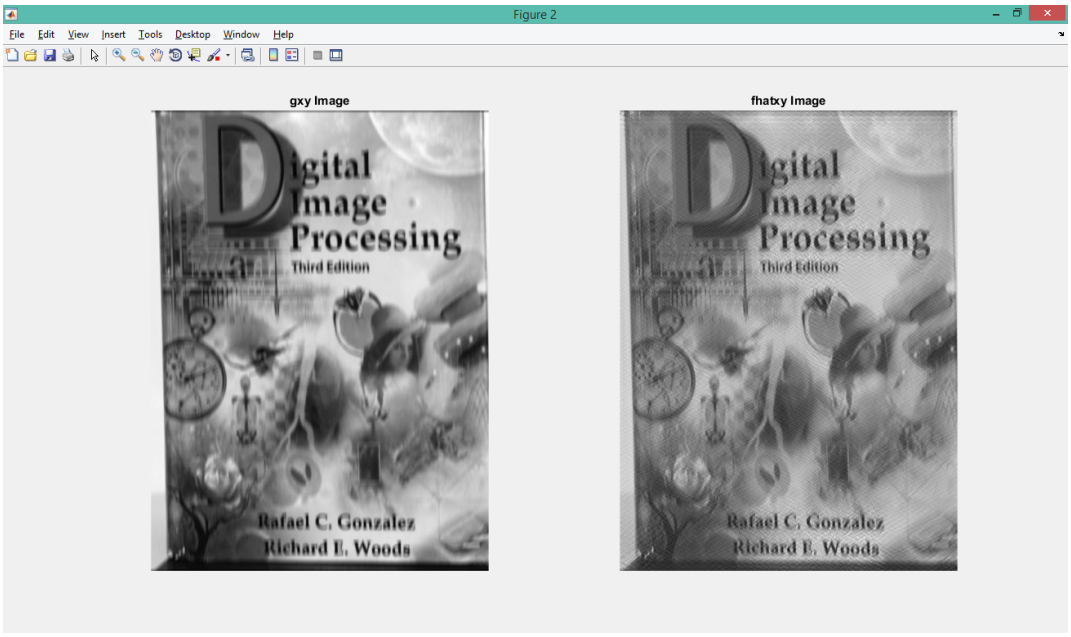
حال می خواهیم به کمک H بدست آمده در بالا، تصویر اولیه را از روی تصویر تار شده تخمین بزنیم. و در نهایت با تقسیم تصویر تار شده در حوزه فرکانس بر تابع خرابی، تصویر اولیه را بازیابی نماییم. بدین منظور ابتدا به کمک تابع imresize تابع خرابی را هم اندازه با تصویر می نماییم:

```
Huv = imresize(Huv,[gxy_row_size, gxy_col_size]);
```

سپس با تقسیم تصویر تار شده در حوزه فرکانس بر تابع خرابی، تصویر اولیه را بازیابی نماییم.

```
Fhatuv = Guv ./ Huv;
fhatxy = ifft2(ifftshift(Fhatuv));
```

نتیجه حاصل به صورت زیر می باشد:

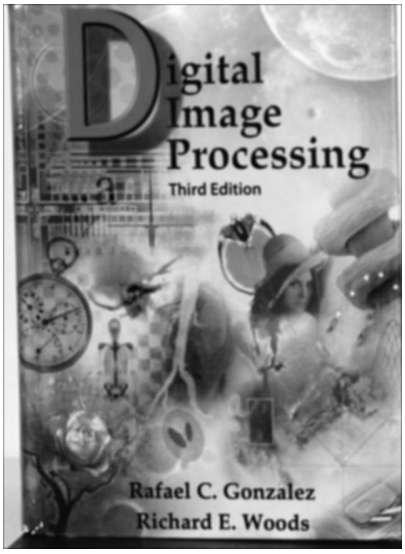


شکل 6 - تخمین تصویر اصلی با استفاده از مشاهده

مرحله دوم:

با توجه به توضیحات داده شده در هر سیستم تصویربرداری، یک فاصله کانونی تعریف می‌شود و نقاطی از صحنه که فاصله متفاوتی تا دوربین دارند، دچار تاری دیفوکوس می‌شوند. تابع تاری دیفوکوس را می‌توان با تابع pillbox توصیف نمود که با تک پارامتر شعاع تاری قابل توصیف است.

دو تصویر تارشده به دلیل دی فوکوس داریم که تاره شده ی شکل 1 با دو تابع دیفوکوس با شعاع متفاوت هستند.



شکل 7 - تصویر دیفوکوس شده با شعاع کوچکتر

شکل 8 - تصویر دیفوکوس شده با شعاع بزرگتر

فرض می کنیم تصویر اصلی در اختیار نیست. به کمک دو تصویر موجود تلاش می کنیم توابع تاری را تخمین بزنیم و سپس تصویر رفع تارشده را بدست آوریم.

رویکردهای قابل بررسی:

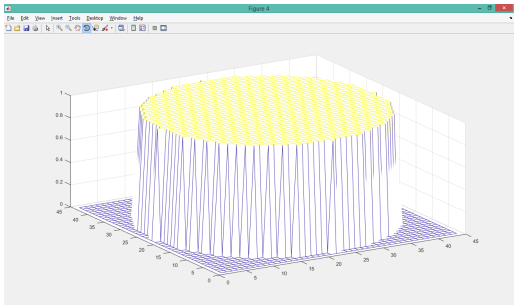
- اصل بازسازی هردو تصویر، یک تصویر واحد می‌شود.
- حاصل تاری هر تصویر با تابع تاری تصویر دیگر با هم تقریباً برابر است:

$$\left.\begin{array}{ll} im_3 = k_1 \otimes im_1 + \omega_1 & \omega_1 \sim N(0,\sigma^2) \\ im_4 = k_2 \otimes im_1 + \omega_2 & \omega_2 \sim N(0,\sigma^2) \end{array}\right\} \rightarrow k_2 \otimes im_3 \cong k_1 \otimes im_4$$

انحراف معیار نویز اضافه شده به تصاویر 0.001 است.

ابتدا تابع myfspecial را طراحی می نماییم تا به کمک آن تابع pillbox را تولید نماییم.

برای این کار از معادله ریاضی دایره استفاده مینماییم. خروجی به شکل زیر می باشد:



شکل 9- فیلتر pillbox

حال می خواهیم به کمک تابع myfspecial با شعاع های مختلف تابع pillbox را تولید نماییم و آن را بر روی تصاویر تار شده تست نماییم.

از آنجا که در این قسمت فرض کردیم که تصویر دارای نویز گوسی با واریانس 0.001 می باشد، پس فرمول بدست آوردن تصویر نهایی به شکل زیر می باشد:

$$\hat{F}(u,v)=\left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2+K}\right]G(u,v)$$

بدین منظور فیلتر pillbox و دو تصویر فوق را به حوزه فرکانس می بریم.و تصویر را بر فیلتر تقسیم می نماییم تا تخمینی از تصویر اصلی بدست آوریم.

```
Guv3 = fft2(gxy3);
Guv4 = fft2(gxy4);

h3 = myfspecial(1.5);
hxy = zeros([gxy_row_size, gxy_col_size]);
[h_row, h_col]=size(h3);
hxy(1:h_row , 1:h_col) = h3(1:h_row , 1:h_col);
Huv3 = fft2(hxy);

h4 = myfspecial(2.5);
hxy = zeros([gxy_row_size, gxy_col_size]);
[h_row, h_col]=size(h4);
hxy(1:h_row , 1:h_col) = h4(1:h_row , 1:h_col);
Huv4 = fft2(hxy);

k =0.001;

Fhatuv3 = (((real(Huv3) .^ 2) ./ ((real(Huv3) .^ 2) + k)) ./ Huv3) .* Guv3;
fhatxy3 = ifft2(Fhatuv3);

Fhatuv4 = (((real(Huv4) .^ 2) ./ ((real(Huv4) .^ 2) + k)) ./ Huv4) .* Guv4;
fhatxy4 = ifft2(Fhatuv4);
```

پس از آزمایش بر روی مقادیر مختلف شعاع و K بهترین نتیجه ای که به دست آوردیم به صورت زیر می باشد:

برای شکل 7 مقدار شعاع برابر 1.5 و مقدار K برابر 1000 بدست آمد.

برای شکل 8 مقدار شعاع برابر 2.5 و مقدار K برابر 1000 بدست آمد.

نتیجه آزمایش مقادیر بالا به صورت زیر می باشد:

شکل 10- تخمین تصاویر دی فوکوس شده

نتیجه گیری:

در مرحله اول قسمت الف، به این نتیجه رسیدیم که اگر تصویر اولیه را داشته باشیم، بدست آوردن تابع خرابی به صورت کاملا دقیق انجام می شود و خروجی کاملا ایده آل خواهد بود.

در مرحله اول قسمت ب، به این نتیجه رسیدیم که اگر تصویر اولیه را نداشته باشیم، بدست آوردن تابع خرابی به صورت کاملا دقیق انجام نمی شود و خروجی کاملا ایده آل نخواهد بود. اما با انتخاب بخش مناسبی از تصویر خراب شده که شبیه به یک ضربه دو بعدی باشد، می توان تخمین بسیار خوبی بدست آورد.

در مرحله دوم، به این نتیجه رسیدیم که اگر تصویر اولیه را نداشته باشیم، بدست آوردن تابع خرابی به صورت کاملا دقیق انجام نمی شود و بسیار سخت خواهد بود. و خروجی کاملا ایده آل نخواهد بود. اما با آزمون و خطا و انتخاب پارامتر های مناسب نظیر شعاع، می توان تخمین بسیار خوبی بدست آورد.

پیوست:

سورس کد پروژه با نرم افزار متلب:

پیاده سازی تابع `myfspecial`

```
function h = myfspecial(Radius)

    r = ceil(Radius);
```

```
    h = zeros(r * 2 + 4, r * 2 + 4);
    [h_row_size, h_col_size] = size(h);

    for row = 1:h_row_size
        for col = 1:h_col_size
            x = row - (r+2);
            y = col - (r+2);

            if sqrt(x .^ 2 + y .^2) > r
               h(row, col) = 0;
            else
               h(row, col) = 1;
            end
        end
    end
end
```

پیاده سازی پروژه:

```
close all;
clear;
%read images
fxy = imread('D:/Im401.jpg');
%fxy = rgb2gray(fxy);

gxy = imread('D:/Im402.jpg');
%gxy = rgb2gray(gxy);

[fxy_row_size, fxy_col_size] = size(fxy);

figure;
subplot(2,2,1);
imshow(fxy);
title('fxy Image');
subplot(2,2,2);
imshow(gxy);
title('gxy Image');

Fuv = fftshift(fft2(fxy));
Guv = fftshift(fft2(gxy));

Huv = Guv ./ Fuv;

subplot(2,2,3);
imshow(abs(log(1 + Huv)),[]);
title('Huv Image');

Fhatuv = Guv ./ Huv;

fhatxy = ifft2(ifftshift(Fhatuv));

subplot(2,2,4);
imshow(fhatxy,[]);
title('fhatxy Image');

%=========================================

clear;
%read images

gxy = imread('D:/Im402.jpg');
%gxy = rgb2gray(gxy);

zf = imread('D:/zf.jpg');
```

```
%gxy = rgb2gray(gxy);
zg = imread('D:/zg.jpg');
%gxy = rgb2gray(gxy);

[gxy_row_size, gxy_col_size] = size(gxy);

[zg_row_size, zg_col_size] = size(zg);

figure
subplot(1,2,1);
imshow(gxy);
title('gxy Image');

Guv = fftshift(fft2(gxy));

zF = fftshift(fft2(zf));
zG = fftshift(fft2(zg));

zH = zG ./ zF;

zh = ifft2(ifftshift(zH));

Huv = zH;

Huv = imresize(Huv,[gxy_row_size, gxy_col_size]);

Fhatuv = Guv ./ Huv;

fhatxy = ifft2(ifftshift(Fhatuv));

subplot(1,2,2);
imshow(fhatxy,[]);
title('fhatxy Image');

%================================

clear;
%read images

gxy3 = imread('D:/Im403.jpg');
gxy4 = imread('D:/Im404.jpg');

gxy3 = im2double(gxy3);
gxy4 = im2double(gxy4);

[gxy_row_size, gxy_col_size] = size(gxy3);

figure
subplot(2,2,1);
imshow(gxy3);
title('gxy3 Image');
subplot(2,2,2);
imshow(gxy4);
title('gxy4 Image');

Guv3 = fft2(gxy3);
Guv4 = fft2(gxy4);

h3 = myfspecial(1.5);
hxy = zeros([gxy_row_size, gxy_col_size]);
[h_row, h_col]=size(h3);
hxy(1:h_row , 1:h_col) = h3(1:h_row , 1:h_col);
```

```
Huv3 = fft2(hxy);

h4 = myfspecial(2.5);
hxy = zeros([gxy_row_size, gxy_col_size]);
[h_row, h_col]=size(h4);
hxy(1:h_row , 1:h_col) = h4(1:h_row , 1:h_col);
Huv4 = fft2(hxy);

k =1;

Fhatuv3 = Guv3 .* Huv3;
fhatxy3 = ifft2(Fhatuv3);

Fhatuv4 = Guv4 .* Huv4;
fhatxy4 = ifft2(Fhatuv4);

subplot(2,2,3);
imshow(fhatxy3,[]);
title('fhatxy Image');
subplot(2,2,4);
imshow(fhatxy4,[]);
title('fhatxy Image');
```