دانشگاه صنعتی اصفهان

دانشکده مهندسی برق و کامپیوتر

عنوان: تکلیف سوم درس مبانی بینایی کامپیوتر

نام و نام خانوادگی: علیرضا ابره فروش

شماره دانشجویی: ۹۸۱۶۶۰۳

نیم سال تحصیلی: بهار ۱۴۰۰/۱۴۰۱

مدرّس: دکتر نادر کریمی

دستیاران آموزشی: بهنام ساعدی - محمدرضا مزروعی

۱

۱.۱ الف

۱.۱.۱ mirror padding

در این روش مقادیر خارج از مرز با مقادیر متناظرشان (قرینه نسبت به محور مرزی افقی یا عمودی) مقداردهی می‌شوند.

۲.۱.۱ میانگین پیکسل‌های همسایه‌های موجود

در این روش میانگین سطح روشنایی بین پیکسل‌هایی از کرنل گرفته می‌شود که وجود دارند. به عبارت دیگر به پیکسل‌هایی که در کرنل وجود ندارند مقدار میانگین پیکسل‌های موجود را می‌دهیم و فیلتر میانگین را روی همه‌ی پیکسل‌های مرزی اجرا می‌کنیم.

۲.۱ ب

وزن پیکسل‌های نزدیک به مرکز در فیلتر گوسی بیشتر از وزن این پیکسل‌ها در فیلتر میانگین است. از آنجایی که در zero-padding، پیکسل‌های خراج از مرز با صفر مقداردهی می‌شوند، هنگام اعمال فیلتر میانگین، وزن آن‌ها (پیکسل‌های صفر) نسبت به فیلتر گوسی بیشتر است و نقاط مرزی را تیره‌تر می‌کند.

۲

چون جمع وزن پیکسل‌ها در کرنل میانگین برابر یک است پس مجموع سطح روشنایی پیکسل‌ها قبل و بعد از اعمال فیلتر میانگین باهم برابر است. در مورد فیلتر گوسی نیز همین قضیه رخ می‌دهد. از آنجایی که انتگرال $\inf - \inf$ تا $\inf$ تابعِ $w(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$ برابر یک است، پس قبل و بعد از اعمال فیلتر گوسی مجموع سطح روشنایی پیکسل‌ها باهم برابر می‌شود. توجه شود که در مثال مطرح شده در آموزش مجازی این مجموع اندکی قبل و بعد از اعمال فیلترها متفاوت است و به این دلیل است که با مقادیر تقریبی سروکار داریم برای مثال در فیلتر میانگین وقتی سطح روشنایی ۱۰۰ را بر ۹ تقسیم می‌کنیم، مقدارِ $11.\overline{1}$ به دست می‌آید، اما در محاسبات از $0.\overline{1}$ صرف نظر می‌شود. به همین ترتیب درمورد فیلتر گوسی این اتفاق رخ میدهد.

۳

۱.۳ Algorithm

ابتدا تصاویر آبی و قرمز معیار برای ارقام ۱ تا ۹ تولید می‌کنیم (پیش‌پردازش). هر تصویر را می‌خوانیم و در هر مرحله (تا زمانی که پیکسل آبی یا قرمز وجود داشته باشد) موقعیت مکانی رقم رنگی‌ای که بالاتر و چپ‌تر است را پیدا می‌کنیم و مقدار MSE (متناظرا نرم ۱) بین آن قطعه از تصویر و تمام ارقام معیار هم‌رنگ تولید شده در مرحله‌ی پیش‌پردازش (و دارای ابعاد برابر با قطعه‌ی پیدا شده (با استفاده از resize)) را محاسبه می‌کنیم. مینیمم این مقادیر با احتمال بسیار بالا مربوط به رقم مورد نظر است. آن را به آرایه‌ی ارقام درون تصویر اضافه می‌کنیم و سپس آن قطعه را تماما سیاه می‌کنیم تا در مرحله بعد رقم بعدی پیدا شود. به این شکل جمع ارقام موجود در همه تصاویر با دقت ۱۰۰ درصد به دست می‌آید.

۲.۳ Preprocessing

```
clc
clear
close all
imtool close all
%%%%%%%%%%%%%%%%%
%generating template image of numbers to compare each element of image with
for k = 0: 9
    I = imread(['images\' num2str(k) '.png']);
    min_row = 1000;
    max_row = 0;
    min_column = 1000;
    max_column = 0;
    for i = 1: size(I, 1)
        for j = 1: size(I, 2)
            if (I(i, j, 1) < 200)
                if (i < min_row)
                    min_row = i;
                end
                if (i > max_row)
                    max_row = i;
                end
                if (j < min_column)
                    min_column = j;
                end
                if (j > max_column)
                    max_column = j;
                end
            end
        end
    end
    Red = uint8(zeros(max_row - min_row + 1, max_column - min_column + 1, 3));
    Blue = uint8(zeros(max_row - min_row + 1, max_column - min_column + 1, 3));
    for i = min_row: max_row
        for j = min_column: max_column
            Red(i - min_row + 1, j - min_column + 1, 1) = 255;
            Red(i - min_row + 1, j - min_column + 1, 2: 3) = I(i, j, 2: 3);
```

```
            Blue(i - min_row + 1, j - min_column + 1, 1: 2) = I(i, j, 1: 2);
            Blue(i - min_row + 1, j - min_column + 1, 3) = 255;
        end
    end

%     imtool(I);
%     imtool(Red);
%     imtool(Blue);
    imwrite(Red, ['images\p' int2str(k) '.png']);
    imwrite(Blue, ['images\n' int2str(k) '.png']);
end
```

۳.۳ Function

```
function sum_of_values = sumOnImage(I)
%SUMONIMAGE Summary of this function goes here
    X = I;
    values = 0;
    digit = 1;
    while(true)
        [min_row, max_row, min_column, max_column, digit] = getDigit(X);
        if (digit == 0)
            break;
        end
        X(min_row: max_row, min_column: max_column, :) = 0;
        %imtool(X);
        values(end + 1) = digit;
    end
    sum_of_values = sum(values);
end

function [min_row, max_row, min_column, max_column, digit] = getDigit(X)
%GETDIGIT Summary of this function goes here
    [min_row, max_row, min_column, max_column, sign] = getBoundaries(X);
    if (min_row == 100000)
        digit = 0;
    else
```

```
        J = X(min_row: max_row, min_column: max_column, :);
        mse_of_numbers = zeros(1, 9);
        if sign == 0
            for i = 1: 9
                num_pic = imread(['images\p' num2str(i) '.png']);
                mse_of_numbers(i) = immse(J, imresize(num_pic, size(J(:, :, 1))));
            end
        else
            for i = 1: 9
                num_pic = imread(['images\n' num2str(i) '.png']);
                mse_of_numbers(i) = immse(J, imresize(num_pic, size(J(:, :, 1))));
            end
        end
        [min_value, digit] = min(mse_of_numbers);
        if sign == 1
            digit = -digit;
        end
    end
end

function [min_row, max_row, min_column, max_column, sign] = getBoundaries(I)
%GETBOUNDARIES Summary of this function goes here
    min_row = 100000;
    max_row = -100000;
    min_column = 100000;
    max_column = -100000;
    sign = 0;
    white_pixel = 255 * ones(1, 1, 3);
    black_pixel = zeros(1, 1, 3);
    top_row = 0;
    top_column = 0;
    break_flag = 0;
    for i = 1: size(I, 1)
        for j = 1: size(I, 2)
            if (~isequal(I(i, j, :), white_pixel) && ~isequal(I(i, j, :), black_pixel))
                top_row = i;
```

```
60                  top_column = j;
61                  if (I(i, j, 1) == 255)
62                      sign = 0;
63                  end
64                  if (I(i, j, 3) == 255)
65                      sign = 1;
66                  end
67                  break_flag = 1;
68                  break;
69              end
70          end
71          if (break_flag)
72              break;
73          end
74      end
75      break_flag = 0;
76      if sign == 0
77          for i = top_row: top_row + 50
78              if (i <= 0)
79                  continue;
80              end
81              for j = top_column - 35: top_column + 35
82                  if (j <= 0)
83                      continue;
84                  end
85                  if (~isequal(I(i, j, :), white_pixel) && ~isequal(I(i, j, :), black_pixel
    ))
86                      if (i < min_row)
87                          min_row = i;
88                      end
89                      if (i > max_row)
90                          max_row = i;
91                      end
92                      if (j < min_column)
93                          min_column = j;
94                      end
```

```
95                      if (j > max_column)
96                          max_column = j;
97                      end
98                  end
99              end
100         end
101     else
102         for i = top_row: top_row + 50
103             if (i <= 0)
104                 continue;
105             end
106             for j = top_column - 35: top_column + 35
107                 if (j <= 0)
108                     continue;
109                 end
110                 if (~isequal(I(i, j, :), white_pixel) && ~isequal(I(i, j, :), black_pixel
    ))
111                     if (i < min_row)
112                         min_row = i;
113                     end
114                     if (i > max_row)
115                         max_row = i;
116                     end
117                     if (j < min_column)
118                         min_column = j;
119                     end
120                     if (j > max_column)
121                         max_column = j;
122                     end
123                 end
124             end
125         end
126     end
127 end
```

۴.۳ Driver code

```
clc
clear
close all
imtool close all
%%%%%%%%%%%%%%%%%
dirinfo = dir("images\Q3");
name_of_images = {dirinfo.name};
ground_truth = zeros(1, 100);
my_result = zeros(1, 100);
box_color = {'green'};
for i = 3: size(name_of_images, 2)
    current_image_name = char(name_of_images(i));
    temp = strsplit(current_image_name, "_");
    kemp = strsplit(char(temp(3)), ".");
    g_t = str2double(kemp(1));
    ground_truth(i) = g_t;
    I = imread(['images\Q3\' current_image_name]);
    J = I;
    m_r = sumOnImage(I);
    my_result(i) = m_r;
    J = insertText(J, [350 756;], num2str(m_r), 'boxColor', box_color, 'FontSize', 22);
    imwrite(J, ['images\A3\' current_image_name]);
end
correct_guesses = my_result == ground_truth;
percentage = 100 * (sum(correct_guesses) - 2) / (size(my_result, 2) - 2)
```

۴

۱.۴ Algorithm

برای هر پیکسل دارای نویز فلفل نمکی (سطح روشنایی ۰ و ۲۵۵) یک کرنل ۳ در ۳ نظر می‌گیریم و آن را با میانگین سطح روشنایی پیکسل‌های همسایه‌اش در کرنل ۳ در ۳ که دارای نویز نیستند (در صورت وجود) مقداردهی می‌کنیم.سطح روشنایی سایر پیکسل‌ها (که یا همه‌ی همسایه‌هاشان نویز هستند یا خودشان فاقد نویز) را بدون تغییر می‌گذاریم. با فرض اینکه تصویر اصلی فاقد سطح روشنایی ۰ یا ۲۵۵ باشد (یا به تعداد کم) الگوریتم را تا جایی که هیچ پیکسل با سطح روشنایی ۰ یا ۲۵۵ باقی نماند تکرار می‌کنیم. این کار در تصاویر با نویز بالا که در آن بسیاری از پیکسل‌ها در همسایگی‌های ۳ در ۳ی خود همسایه‌ی غیر نویز ندارند می‌تواند تا حدی کیفیت تصاویر را ارتقا دهد. توجه شود که در صورتی که تصویر اصلی تعداد زیادی پیکسل ۰ یا ۲۵۵ داشته باشد آنگاه این الگوریتم ناکارآمد است. همچنین با توجه به اینکه ممکن است الگوریتم به تعداد زیادی اجرا شود (با توجه به درصد نویز و میزان

پراکندگی آن) در کاربردهایی که زمان اهمیت دارد می‌تواند نا کارآمد باشد.

۲.۴ Function

```
function K = removeNoise(J)
%REMOVENOISE Summary of this function goes here
    K = J;
    for i = 1: size(K, 1)
        for j = 1: size(K, 2)
            if (K(i, j) == 0 || K(i, j) == 255)%comment this to include
%             all pixels
                arr = [];
                for k = i - 1: i + 1
                    for l = j - 1: j + 1
                        if (k > 0 && k < size(K, 1) && l > 0 && l < size(K, 2))
                            if (K(k, l) > 0 && K(k, l) < 255)
                                arr(end + 1) = K(k, l);
                            end
                        end
                    end
                end
                K(i, j) = mean(arr);
            end%comment this to include all pixels
        end
    end
end
```

۳.۴ Driver code

```
clc
clear
close all
imtool close all
%%%%%%%%%%%%%%%%%
I = imread("images\Q4\House.tif");
my_psnr_values = [];
med_psnr_values = [];
for d = 0.1: 0.1: 0.9
```

```
    J = imnoise(I, 'salt & pepper', d);
    K = removeNoise(J);
    while (ismember(0, K) || ismember(255, K))
        K = removeNoise(K);
    end
    L_prime = medfilt2(J);
    L = J;
    for i = 1: size(L, 1)%to apply to all pixels comment this loop and use L_prime
    instead of L
        for j = 1: size(L, 2)
            if (L(i, j) == 0 || L(i, j) == 255)
                L(i, j) = L_prime(i, j);
            end
        end
    end
    my_psnr_values(end + 1) = psnr(K, I);
    med_psnr_values(end + 1) = psnr(L, I);
end
my_psnr_values
mean(my_psnr_values)
med_psnr_values
mean(med_psnr_values)
```

۴.۴ Results

۱.۴.۴ اعمال الگوریتم روی پیکسل‌های دارای نویز

مقدار نویز	مقدار PSNR							
	Bridge		Boat		Peppers		House	
	Median	روش شما	Median	روش شما	Median	روش شما	Median	روش شما
۱۰٪	34.2685	35.5608	36.9305	38.4299	39.5897	39.8929	37.5208	41.2446
۲۰٪	29.1947	32.3510	30.8115	35.1030	31.6369	36.9182	30.8793	38.0098
۳۰٪	24.2216	30.3418	24.9933	33.0565	25.7074	35.0742	25.3376	35.6030
۴۰٪	19.8592	28.7748	20.4020	31.3515	20.7401	33.0959	20.3952	33.6473
۵۰٪	16.2266	27.2865	16.6014	29.9160	16.5719	31.6668	16.4084	31.8117
۶۰٪	13.1765	25.8880	13.4030	28.4420	13.3855	30.1246	13.6071	30.3997
۷۰٪	10.6323	24.3311	10.9112	26.7086	10.7682	28.1435	10.8164	28.2111
۸۰٪	8.4913	22.5335	8.7927	24.7861	8.6039	25.9394	8.7580	25.6044
۹۰٪	6.7577	20.0949	6.9537	21.8315	6.8026	22.3678	6.9176	22.6665
میانگین	**18.0921**	**27.4625**	**18.8666**	**29.9583**	**19.3118**	**31.4693**	**18.9600**	**31.9109**

۲.۴.۴ اعمال الگوریتم روی تمام پیکسل‌ها

مقدار نویز	مقدار PSNR							
	Bridge		Boat		Peppers		House	
	Median	روش شما	Median	روش شما	Median	روش شما	Median	روش شما
۱۰٪	26.4097	26.4315	29.5318	28.9522	33.0908	30.9779	31.0373	31.5807
۲۰٪	24.7094	26.1935	26.7574	28.6520	28.4362	30.5863	27.0972	31.1686
۳۰٪	21.6243	25.8358	22.7045	28.2411	23.4262	30.1847	22.2786	30.6399
۴۰٪	17.9888	25.3857	18.5878	27.7388	18.7552	29.6208	18.3987	29.9686
۵۰٪	14.6814	24.7770	14.9653	27.0795	15.1718	28.8453	14.8342	27.7914
۶۰٪	11.8506	23.1835	12.1876	26.0641	12.2067	27.7857	12.0371	27.6941
۷۰٪	9.7641	22.8620	9.9448	24.3833	9.8579	26.2016	9.9807	25.6126
۸۰٪	7.9810	21.1757	8.1638	22.9033	7.9377	23.6133	8.0554	23.6689
۹۰٪	6.4092	19.2518	6.6635	20.7581	6.4928	20.6154	6.6111	21.1087
میانگین	**15.7132**	**23.8996**	**16.6118**	**26.0858**	**17.2639**	**27.6034**	**16.7034**	**27.6926**

منابع