Contrast Adjustment

علیرضا لرستانی

اطلاعات گزارش

تاریخ: 1399/08/23

واژگان کلیدی:
هیستوگرام
همسان سازی(متعادل سازی)

چکیده

در این مقاله، ابتدا به پیاده سازی برنامه ای خواهیم پرداخت که در آن هیستوگرام یک تصویر را محاسبه میکنیم سپس برنامه با تصویر camera man تست کرده و به شرح نتایج خواهیم پرداخت. در ادامه به همسان سازی هیستو گرام بر روی تصویر مذبور خواهیم پرداخت. در بخش سوم این مقابه به بررسی تفاوت بین دو تابع histeq و imadjust خواهیم پرداخت و در پایان به بهبود کنتراست محلی بر روی 4 تصویر تست خواهیم پرداخت.

1-مقدمه

نوشتار حاضر، به بررسی الگوریتم پیاده سازی هیستو گرام یک تصویر و همسان سازی آن به جهت افزایش کنتراست و وضوح عکس میپردازیم. در پیاده سازی ها از زبان برنامه نویسی پایتون استفاده شده که در پایان مقاله کد ها نیز قرار داده خواهند شد. همچنین تصاویر تست و نتیجه ی آنها در متن مقاله قرار داده خواهد شد.

2-شرح تکنیکال

• 2.1.1 : محاسبه هیستوگرام

- هیستوگرام چیست؟

هیستوگرام در واقع نوعی نمایش گرافیکی است که نشان میدهد چند مرتبه مقادیر مختلف رنگ در تصویر تکرار شده.

برای محاسبه هیستوگرام کافیست به ازای هر intensity level که در بازه 0 تا 255 قرار میگیرند، تعداد دفعات تکرار آن را در تمام پیکسل های تصویر به دست آوریم.

• 2.1.2 : متعادل سازی

- متعادل سازی هیستوگرام به دو روش محلی و سراسری صورت میپذیرد.
- در روش سراسری، عمل متعادل سازی بر روی هیستوگرام کل تصویر اعمال میشود. در حالیکه در روش محلی پنجره هایی با سایز متفاوت را در نظر میگیریم و متعادل سازی بر روی این پنجره ها اعمال میشود.
- در روش سراسری ابتدا مطابق آنچه پیش از این توضیح داده شد هیستوگرام تصویر را به دست می آوریم. سپس احتمال توزیع یا pdf هر پیکسل را محاسبه میکنیم. سپس cdf هر پیکسل که مجموع pdf پیکسل های قبلی است را محاسبه میکنیم. سپس

* پست الکترونیک نویسنده alirezalorestani2010@gmail.com
1. دانشجو، دانشکده مهندسی، دانشگاه فردوسی مشهد

مقدار نهایی جدید را برای هر پیکسل با ضرب مقدار قبلی در cdf به دست می آوریم.

- **2.1.3 : تفاوت histeq و imadjust**

تابع imadjust مقادیر intensity در یک تصویر grayscale را به مقادیر جدیدی تبدیل میکند. به طور پیشفرض 1 درصد پایینی و 1 درصد بالایی intensity level را اشباع میکند. این عمل باعث افزایش کنتراست عکس میشود. همچنین امکان تعیین بازه برای مقادیر جدید و همچنین تغییر میزان گاما در عکس خروجی را فراهم می آورد.

J = imadjust (I, [low_in high_in],[low_out high_out],gamma)

تابع histeq هیستوگرام تصویر grayscale را به گونه ای تغییر میدهدکه هیستوگرام تصویر خروجی فلت تر باشد. با فراخوانی J = histeq (I, hgram) هیستوگرام تصویر I را به hgram تبدیل میکند.

- **۲.۲.۱ : همسان سازی محلی**

- تفاوت همسان سازی محلی با همسان سازی سراسری در این است که در همسان سازی سراسری هیستوگرام کل تصویر را متعادل میکنیم اما در همسان سازی محلی ابتدا پنجره ای با سایز دلخواه انتخاب میکنیم همسان سازی را روی این پنجره اعمال میکنیم سپس پنجره را جابجا میکنیم و دوباره متعادل سازی را اعمال میکنیم و این عمل را انقدر تکرار میکنیم تا کل تصویر با پنجره موردنظر پوشش داده شود. بر اساس همین جابجایی متعادل سازی محلی به دو روش پوشا و غیر پوشا انجام میشود.

در هر دو حالت گوشه بالا و سمت چپ پنجره در گوشه بالا و سمت چپ تصویر قرار میگیرد و آنقدر جابجا میشود که گوشه پایین سمت راست پنجره بر روی گوشه پایین سمت راست تصویر قرار بگیرد.

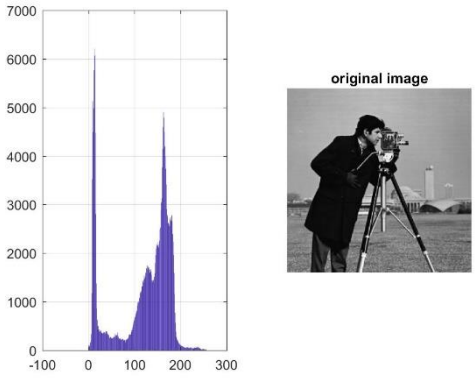
تفاوت دو روش پوشا و غیر پوشا در این است که در روش غیر پوشا پنجره ها نباید روی هم overlap کنند اما در روش پوشا چنین محدودیتی نداریم.

3- شرح نتایج و نتیجه گیری

۲.۱ - همسان سازی هیستوگرام

1تصویر "Camera Man" ، تصویر اصلی

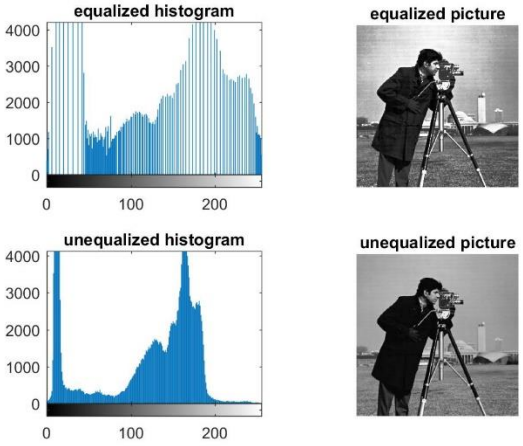
- ۲.۱.۱



2هیستوگرام تصویر 1

در بالا تصویر اصلی Camera Man به همراه هیستوگرام آن را مشاهده میکنید. همانطور که مشخص است رنج تیره ی تصویر (نزدیک به صفر) فراوانی بالاتری دارد.

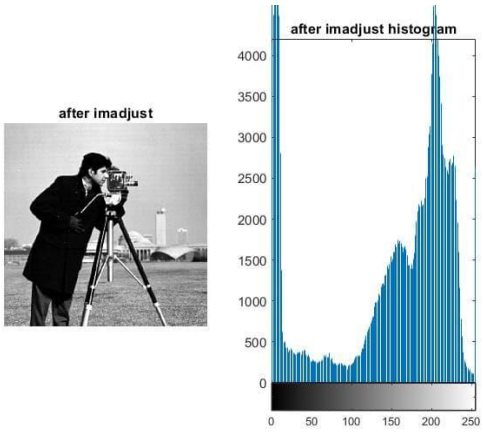
- ۲.۱.۲



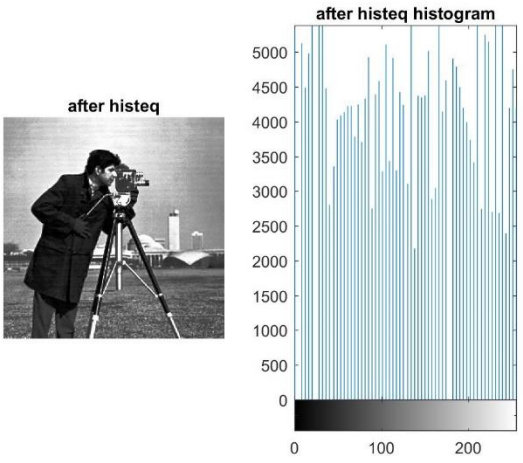
3تصویر اصلی به همراه تصویر همسان سازی شده و هیستوگرام های آن ها

تصویر ۳ نشان میدهد وضوح تصویر پس از اعمال همسان سازی سراسری افزایش یافته است.

- ۲.۱.۳

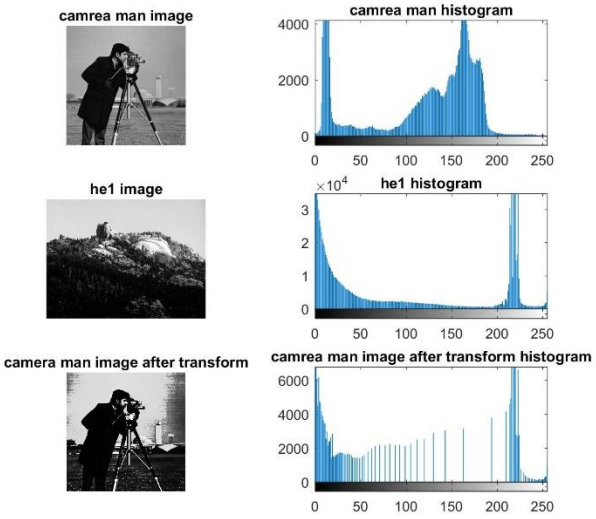


4پس از اعمال imadjust



5پس از اعمال histeq

همانطور که مشاهده میکنید پس ار اعمال histeq هیستوگرام حاصل فلت تر شده.

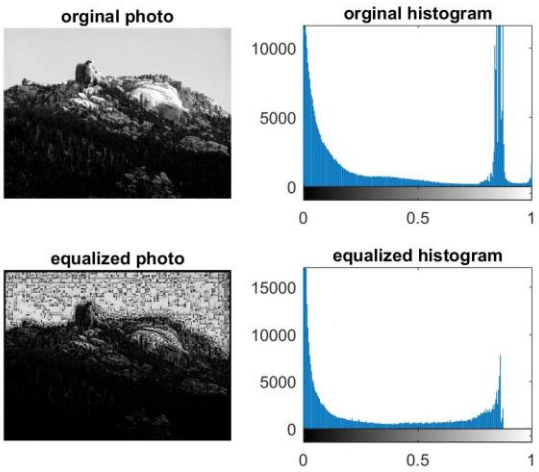


6 هیستوگرام تصویر camera man به هیستوگرام تصویر he1 تبدیل شده.

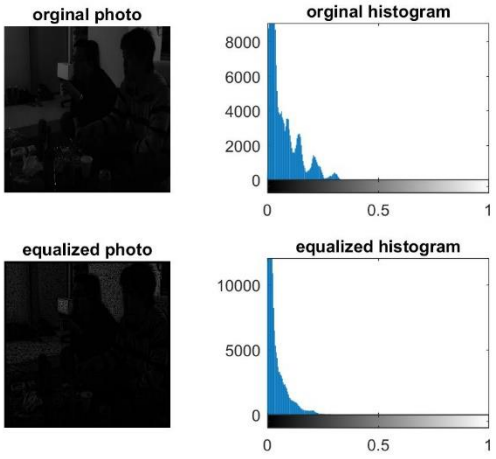
در تصویر شماره ۶ در ردیف اول تصویر اصلی camera man به همراه هیستوگرام آن را مشاهده میکنیم. در ردیف دوم تصویر he1 به همراه هیستوگرام آن آمده است. سپس به کمک تابع histeq هیستوگرام camera man را به هیستوگرام تصویر he1 تبدیل میکنیم.

در ردیف سوم تصویر حاصل به همراه هیستوگرام آن را مشاهده میکنید.

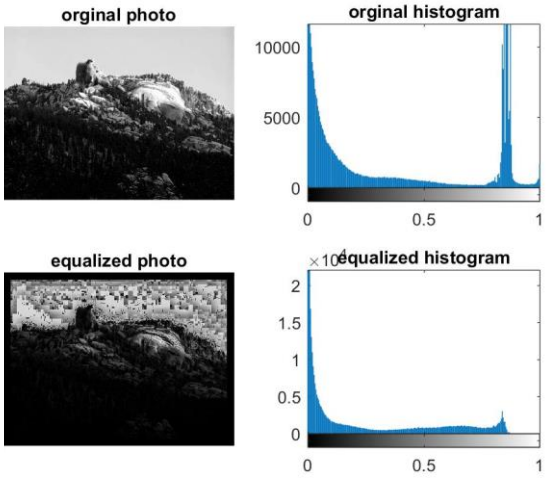
• ۲.۲.۱



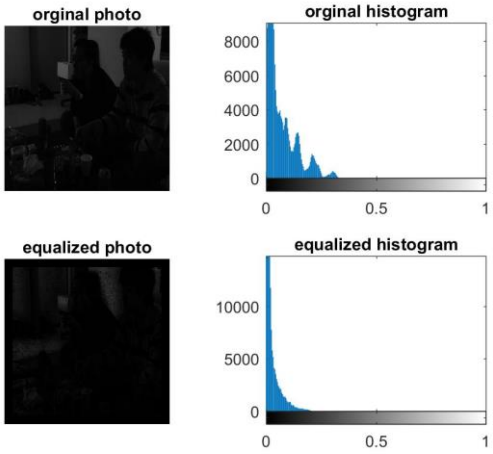
7تصویر **he1** به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۱۵



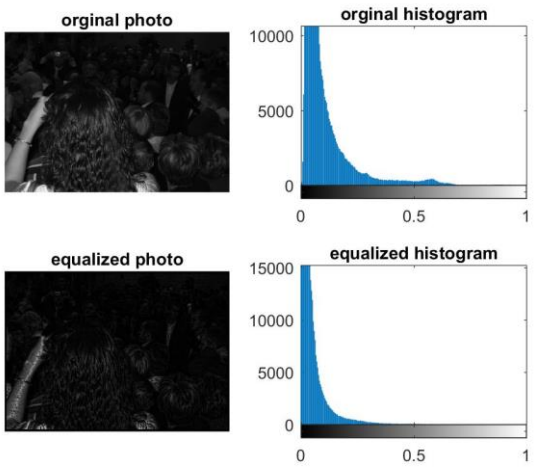
۹ تصویر **he2** به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۱۵



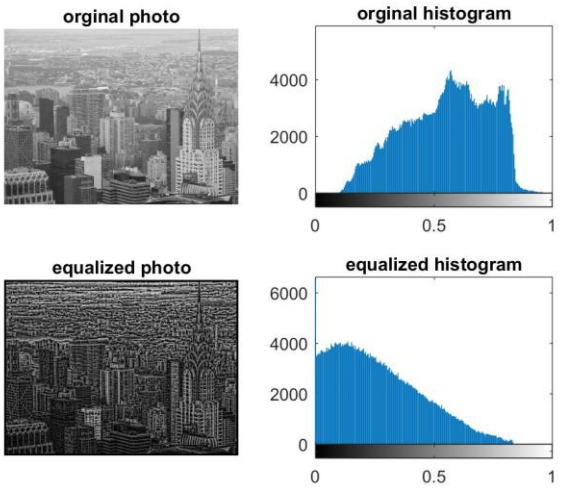
۸ تصویر **he1** به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۵۰



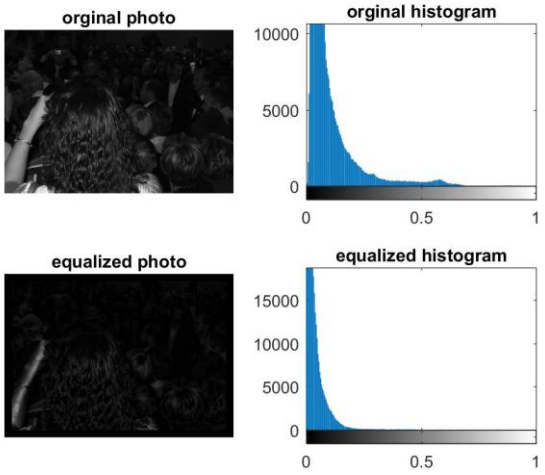
10تصویر **he2** به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۵۰



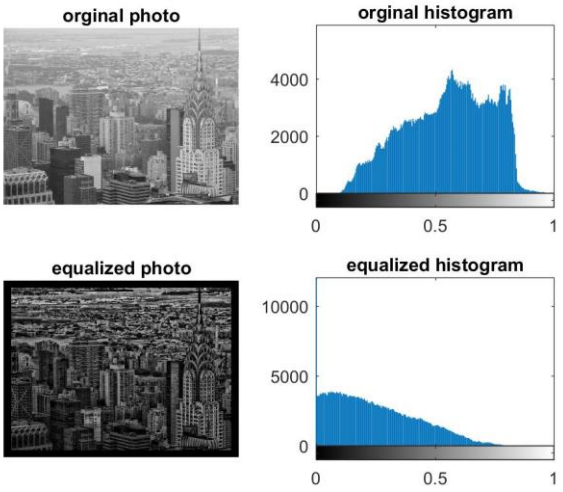
۱۱ تصویرhe3 به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۱۵



۱۳ تصویر he4 به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۱۵



12تصویر he3 به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۵۰



۱۴ تصویر he4 به همراه هیستوگرام و تصویر بعد از اعمال همسان سازی محلی با سایز پنجره ۵۰

در تست های انجام شده به نظر میرسد تصاویر خروجی همسان سازی سراسری نتایج مطلوب تری دارند.

در همسان سازی محلی شاهد کاهش نویز پس از افزایش اندازه پنجره یافت در حالی که با کاهش سایز پنجره جزئیات تصاویر افزایش یافت.

۴- پیوست

2.1.1

```
function histogram(img)

h = img;
[Row, Column] = size(h);
t = 1:256;
n = 0:255;
count = 0;
for z = 1:256
    for i = 1:Row
        for j = 1:Column

            if h(i, j) == z - 1
                count = count + 1;
            end
        end
    end
    t(z) = count;
    count = 0;
end
figure;
subplot(1, 2, 1);
bar(n, t);
grid on;
subplot(1, 2, 2), imshow(img), title('original image');
end
```

تابع محاسبه و رسم هیستوگرام

2.1.2

```
function histogrameq(img)
orginal = img;
[rows, columns, ~] = size(orginal);
finalResult = uint8(zeros(rows, columns));
pixelNumber = rows * columns;
frequncy = zeros(256, 1);
pdf = zeros(256, 1);
cdf = zeros(256, 1);
cummlative = zeros(256, 1);
outpic = zeros(256, 1);
for i = 1:1:rows
    for j = 1:1:columns
        val = orginal(i, j);
        frequncy(val + 1) = frequncy(val + 1) + 1;
        pdf(val + 1) = frequncy(val + 1) / pixelNumber;
    end
end
sum = 0;
%we want the 256 - 1 that's why we initailzed the intensityLevel with 255
%instead of 256
intensityLevel = 255;

for i = 1:1:size(pdf)
    sum = sum + frequncy(i);
    cummlative(i) = sum;
    cdf(i) = cummlative(i) / pixelNumber;
    outpic(i) = round(cdf(i) * intensityLevel);
end

for i = 1:1:rows
    for j = 1:1:columns
        finalResult(i, j) = outpic(orginal(i, j) + 1);
    end
end
subplot(2, 2, 2), imshow(finalResult), title('equalized picture');
subplot(2, 2, 1), imhist(finalResult), title('equalized histogram');
subplot(2, 2, 4), imshow(orginal), title('unequalized picture');
subplot(2, 2, 3), imhist(orginal), title('unequalized histogram');

end
```

تابع همسان ساز سراسری

2.1.3

```
function transformCameraManToHe1(img, he1)
subplot(3, 2, 1), imshow(img), title('camrea man image');
subplot(3, 2, 2), imhist(img), title('camrea man histogram');
subplot(3, 2, 3), imshow(he1), title('he1 image');
subplot(3, 2, 4), imhist(he1), title('he1 histogram');
subplot(3, 2, 5), imshow(histeq(img, imhist(he1))), title('camera man image after transform');
subplot(3, 2, 6), imhist(histeq(img, imhist(he1))), title('image after transform histogram');

end
```

تابع تطبیق هیستوگرام camera man به he1

2.2.1

```
% Create an empty array
Ieq = zeros(size(I, 1), size(I, 2));
% Apply this over a NxN region around each pixel (N is odd)
n = floor(windowsize / 2); % <-- N is the half size of the region ie. [-N:N,-N:N]
for r = 1 + n:size(I, 1) - n
    for c = 1 + n:size(I, 2) - n
        % -- INSERT YOUR CODE BELOW ----------------------------------------
        % NOTE: For pixels near the boundary, ensure the NxN neighbourhood is still
        % inside the image (this means for pixels near the boundary the pixel may
        % not be at the centre of the NxN neighbourhood).
        if r - n <= 1
            fromrow = 1;
            torow = r + n;
        else
            fromrow = abs(r - n);
            if n + r >= size(I, 1)
                torow = size(I, 1);
            else
                torow = r + n;
            end
        end
        if c - n <= 1
            fromcol = 1;
            tocol = c + n;
        else
            fromcol = abs(c - n);
            if c + n > size(I, 2);
                tocol = size(I, 2);
            else
                tocol = c + n;
            end
        end
        neighbour = I(fromrow:torow, fromcol:tocol);
        lessoreq = neighbour(neighbour <= I(r, c));
        sumofval = sum(lessoreq);
        pixval = sumofval / (size(neighbour, 1) * size(neighbour, 2));
        Ieq(r, c) = pixval;
        % -- INSERT YOUR CODE ABOVE ----------------------------------------
    end
end
```

تابع همسان ساز محلی