انتقال داده‌ها

موضوع پروژه : اضافه و حذف کردن نویز به تصویر

استاد درس: ابولفضل دیانت

نام دانشجو: فرزان رحمانی

نیم‌سال دوم

سال تحصیلی ۱۴۰۲-۱۴۰۱

فهرست مطالب

فهرست تصاویر

فصل ۱

سیگنال تصویر و اضافه کردن نویز

۱.۱ گام اول

از قبل برنامه متلب را در کامپیوتر خود نصب داشتم و با آن کار کرده بودم. چرا که در ترم سوم برای درس مدار الکتریکی به آن نیاز داشتم.

۲.۱ گام دوم

در این گام عکسی را به عنوان ورودی برنامه‌مان انتخاب می‌کنیم.

شکل ۱.۱: عکس ورودی

۱.۲.۱ بررسی کد

حال کد متلب این گام را بررسی می‌کنیم :

```
clc;
close all;

% STEP 2
filePath = 'E:\uni\6th term\Data transmission\project\p1\
    farzan.jpg';
img = imread(filePath);
figure;
imshow(img)
% ----------------
```

حال به بررسی خطوط کد می‌پردازیم:

- اول مسیر فایل داده‌شده را مشخص میکنیم
- سپس با دستور $imread$ آن فایل را می‌خوانیم
- و با دستور $imshow$ فایل مرحله قبل را در خروجی نشان می‌دهیم.

۳.۱ گام سوم

در این گام با توجه به راهنمایی گفته شده در داک پروژه از تابع پیش تعریف‌شده $rgb2gray$ استفاده می‌کنیم.

۱.۳.۱ بررسی کد

حال کد متلب این گام را بررسی می‌کنیم :

```
% STEP 3
grayImg = rgb2gray(img);
% ----------------
```

۲.۳.۱ انواع تصاویر

- `binary images` : این نوع تصاویر در یک ماتریس $m * n$ ذخیره می‌شوند که رنگ سیاه در آن معادل صفر و رنگ سفید معادل یک می‌باشد
- `indexed images` : تصاویر نمایه‌ای یک نوع فرمت تصویر است که برای ذخیره‌ی تصاویر با پالت رنگی محدود استفاده می‌شود. در این نوع تصاویر، رنگ هر پیکسل به عنوان یک شاخص به جای مقدار رنگ RGB ذخیره می‌شود. با استفاده از یک جدول نمایه ، هر شاخص به مقدار واقعی رنگ تصویر تبدیل می‌شود.

به عبارت دیگر، تصاویر نمایه‌ای شامل مجموعه‌ای از شاخص‌ها هستند که به رنگ‌های مختلف متناظر هستند. این شاخص‌ها معمولاً عدد صحیح بین ۰ و ۲۵۵ هستند، که هر عدد نمایانگر یک رنگ است. از این روش برای کاهش حجم فایل تصویر و افزایش سرعت پردازش در برنامه‌ها استفاده می‌شود، زیرا نیاز به ذخیره‌سازی مستقیم مقادیر RGB برای هر پیکسل وجود ندارد.

به عنوان مثال، فرمت GIF یکی از نمونه‌های تصاویر نمایه‌ای است که می‌تواند تا ۲۵۶ رنگ را در هر تصویر ذخیره کند. اما توجه داشته باشید که در تصاویر نمایه‌ای، تعداد رنگ‌ها محدود است و ممکن است منجر به کاهش کیفیت و دقت رنگ در تصویر شود.

- `grayscale images` : در این نوع تصاویر که در یک ماتریس $m * n$ ذخیره می‌شوند هر عضو آن شدت رنگ آن پیکسل را نشان می‌دهد. که بزرگترین عدد برای رنگ سفید و کوچکترین عدد برای رنگ سیاه می‌باشد که با توجه به تایپ داده ها می‌توانند رنج مختلفی برای خود بگیرند
  - [0,1] : single or double
  - [0,..,255] : uint8
  - [0,...,65535] : uint16
  - [-32768,...,32768] : int
- `truecolor images(rgb images)` : این نوع تصاویر در یک ماتریس سه بعدی یعنی $m * n * 3$ ذخیره می شوند که به جای ذخیره کردن عدد در مرحله قبل یک ماتریس ۳ عضوی متشکل از شدت رنگ های rgb (قرمز، آبی و سبز) را در خود نگه می دارد.

۴.۱ گام چهارم

تصویر تبدیل‌شده در مرحله قبل را با استفاده از تابع imshow نشان داده و آن را با استفاده از تابع imwrite ذخیره می‌کنیم.

۱.۴.۱ بررسی کد

```
% STEP 4
figure;
imwrite(grayImg, 'GrayImage.jpg');
```

```
% imwrite(grayImg,'GrayImage.png');
imwrite(grayImg,'GrayImage.bmp');
imwrite(grayImg,'GrayImage.tiff');
imshow(grayImg);
% ----------------
```

شکل ۲.۱: تصویر خاکستری‌شده

۲.۴.۱ تفاوت فرمت های متفاوت عکس

- `jpg` : این نوع تصاویر از نوع فشرده‌سازی با اتلاف [۱] می‌باشند. در واقع حجم فایل را تا حد زیادی کاهش می‌دهد این نوع داده برای نگه‌داری و فرستادن مناسب است. این فرمت فایل از فشرده سازی با اتلاف برای تصاویر دیجیتال شما پشتیبانی می کند. این بدان معنی است که حجم فایل شما کاهش می یابد (فشرده سازی بیشتر مربوط به اندازه فایل های کوچکتر است). می توانید تصاویر را به میزان قابل توجهی کوچکتر کنید. قربانی این امر این است که کیفیت تصاویر شما کاهش می یابد (کمتر واضح

[۱] lossy compression

هستند) زیرا فشرده سازی برخی از داده های اصلی را دور می اندازد تا تصویر کوچکتر شود. این به ویژه هر چه کوچکتر بروید صادق است.

- `png` : این نوع تصایر از نوع فشرده‌سازی بدون اتلاف [۲] می‌باشند. در واقع این نوع فشرده‌سازی مخالف فشرده‌سازی قبلی می‌باشد و می‌توان به بازسازی دوباره داده اصلی از فایل فشرده رسید. یک تصویر PNG برعکس فشرده سازی با اتلاف JPG است. از فشرده سازی بدون اتلاف پشتیبانی می کند. همانطور که از نام آن پیداست، "lessless" به توانایی آن در نگه داشتن تمام داده های اصلی در قالب فایل اشاره دارد. در نتیجه، کیفیت تصویر اصلی شما تغییر نمی کند یا کاهش نمی یابد. با این حال، معاوضه برای این مزیت این است که اندازه فایل بزرگتر از JPG خواهد بود. این می تواند در مورد سرعت وب سایت و UX برای بازدیدکنندگان وب سایت شما مشکل ساز باشد، زیرا تصاویر بزرگتر باعث می شود کل وب سایت زمان بیشتری برای بارگذاری طول بکشد.
- `bmp` : توسط شرکت Microsoft توسعه یافته‌شده است. حجم فایل بیشتری دارد و از نوع فشرده‌سازی بدون اتلاف می‌باشد. فرمت فایل بیت مپ یک فرمت جالب است زیرا استانداردی است که توسط مایکروسافت توسعه یافته و کلاسیک در نظر گرفته شده است. این بدان معنی است که به طور کلی در رابط های کاربری گرافیکی قدیمی استفاده می شود. طبیعتاً این یک فایل تصویری بزرگتر است. با این حال، می‌توانید آن را با فناوری ZIP تحت فشرده‌سازی بدون تلفات قرار دهید.
- `tiff` : این نوع نسبت به نوع های‌های قبل بیشترین حجم فایل را دارد و از نوع فشرده‌سازی بدون اتلاف می‌باشد و معمولا در صنعت هنر و عکس‌های حرفه‌ای استفاده می‌شود. فرمت تصویر TIFF از دهه ۱۹۸۰ وجود داشته است. در حالی که از این نظر یک کلاسیک است، اما محبوبیت آن کم نشده است. در هر صورت، با گذشت دهه ها محبوبیت بیشتری پیدا کرده است. این به این دلیل است که افراد حرفه ای در صنایعی مانند هنرهای گرافیک، عکاسی و انتشارات کار می کنند. یکی از بزرگترین مزایای آن انعطاف پذیری کامل آن است که به کاربران اجازه می دهد تصاویر و داده ها را در یک فایل مدیریت کنند. همچنین از فشرده سازی بدون اتلاف و بدون اتلاف پشتیبانی می کند و خلاقیت ها را با آزادی زیادی در هنگام کار با آن توانمند می کند.

اگر بخواهیم از بین این فرمت ها یک فرمت را به عنوان فشرده‌سازی بدون اتلاف انتخاب کنیم کدام فرمت می‌باشد؟ `.tiff`

[۲] lossless compression

۵.۱ گام پنجم

با توجه به جستجوهای صورت‌گرفته تصویرها از نوع سیگنال توان نیستند اما از نوع سیگنال انرژی می‌باشند. در پی جستجوهای مختلف متوجه شده‌ام هر عضو ماتریس نشان‌دهنده انرژی آن می‌باشد که با جمع مربع آن‌ها می‌توان به انرژی کل تصویر رسید. در واقع تصاویر از نوع سیگنال انرژی هستند و عدد هر پیکسل نشان دهنده انرژی آن می باشد، پس انرژی کل تصریر برابر با مجموع اعداد همه پیکسل ها است. همچنین چون جمع تعداد محدودی اعداد صحیح مثبت است پس انرژی کل تصویر نیز محدود می‌باشد. لذا سیگنال تصویر از نوع سیگنال انرژی می‌باشد. توجه کنیم که چون تصاویر در فضای گسسته هستند بجای انگرال از جمع استفاده می‌شود. دو راه را برای این‌کار انجام داده‌ام یکی که طبق نکته ذکرشده بالا و دیگری طبق رابطه پارسوال اول تبدیل فوریه آن را محاسبه کرده و سپس انرژی این سینگال را در حوزه فرکانس (جمع مربع مقادیر آن در حوزه فرکانس) حساب می‌کنیم. اما جواب های متفاوتی کسب کرده‌ام .

۱.۵.۱ بررسی کد

```
% STEP 5
totalEnergy = sum(grayImg(:).^2);
display(totalEnergy);
% alternative
F = fft2(grayImg);
magImage = abs(F).^2;
energy = sum(magImage(:));
display(energy);
% ----------------------
```

```
totalEnergy : 593518031, energy : 6.8558e+16
```

۶.۱ گام ششم

در این گام نیز با توجه به تابع از پیش تعریف‌شده *imnoise* که مقادیر پیش‌فرضمقادیر 0.01 برای واریانس و صفر برای میانگین در نظر گرفته‌شده است می‌توان به تصویر نویز اضافه کرد. و به عنوان ورودی دوم تابع نویز *gaussian* را به آن می‌دهیم و در ادامه نویز اضافه شده به تصویر را نمایش می‌دهیم.

۱.۶.۱ بررسی کد

```
% STEP 6
figure;
noisedPicture = imnoise(grayImg,'gaussian');
imshow(noisedPicture);
imwrite(noisedPicture,'noisyImage.jpg');
% ----------------
```

شکل ۳.۱: تصویر با نویز

۲.۶.۱ SNR(signal to noise ratio)

در این قسمت به محاسبه نسبت سینگال به نویزاضافه شده می پردازیم. نسبت سیگنال به نویز، سطح توان سیگنال را با سطح توان نویز مقایسه می کند و معمولا بر حسب دسیبل بیان می شود. هرچه مقدار نسبت سیگنال به نویز بیشتر باشد، برای یک سیستم مشخصه بهتری محسوب می شود؛ زیرا اطلاعات مفید بیشتری در قالب سیگنال، نسبت به اطلاعات ناخواسته یا نویز دریافت می شود.
`SNR` : میزان قدرت یک سیگنال نسبت به نویز پس‌زمینه آن می‌باشد. که با با واحد db اندازه‌گیری می‌شود.
$SNR = 10 \cdot \log_{10} \frac{P_{signal}}{P_{noise}}$

این نسبت می‌تواند هر عددی باشد که اعداد بزرگتر از صفر نشان دهنده این است که سطح سیگنال اصلی بیشتر از سطح noise است و اعداد کوچکتر برعکس. در واقع هر چه این نسبت بزرگتر باشد کیفیت سیگنال بهتر است.

توضیح مفصل تر:

Ratio Signal-to-Noise یک معیار است که در مورد نویز در عکس استفاده می‌شود. SNR نسبت سیگنال به نویز را نشان می‌دهد و از طریق اندازه‌گیری قدرت سیگنال نسبت به قدرت نویز، کیفیت سیگنال را ارزیابی می‌کند.

در مفهوم ،SNR سیگنال نشان‌دهنده قسمت مورد نظر و معنادار تصویر است که می‌خواهیم دریافت کنیم. نویز، خراب‌کننده سیگنال است و می‌تواند از منابع مختلفی نظیر اندازه‌گیری ناقص، تداخل الکترومغناطیسی یا خطاهای فرآیند تولید تصویر، به وجود آید.

با اندازه‌گیری ،SNR می‌توانیم نسبت قدرت سیگنال به قدرت نویز را محاسبه کنیم. این نسبت به صورت لگاریتمی بیان می‌شود و معمولاً به واحد دسی‌بل (dB) تبدیل می‌شود. بالاترین SNR ممکن، یعنی بی‌نویز بودن، معادل یک SNR بالغ بر بی‌نهایت است. در عمل، هدف ما افزایش SNR و کاهش نویز است تا سیگنال مورد نظر را بهتر و دقیق‌تر دریافت کنیم.

به عنوان مثال، در عکس‌برداری دیجیتال، SNR برای سنسور دوربین معیاری است که نشان می‌دهد درصد سیگنال واقعی تصویر (نسبت به سیگنال نویز) در میان اثرات نویزی ناشی از سنسور (نویز سیستمی) و شرایط نوری ضعیف (نویز محیطی) است. SNR بالا به معنای وجود کمترین نویز و سیگنال واضح و با کیفیت بالاست.

۳.۶.۱ بررسی کد

```
% SNR % https://www.mathworks.com/help/signal/ref/snr.
    html
x = im2double(grayImg);
y = im2double(abs(noisedPicture - grayImg));
signalNoiseRatio = snr(x,y);
display(signalNoiseRatio);
% ----------------
```

در این بخش اگر دو سیگنال را به صورت مستقیم به تابع از پیش تعریف‌شده `snr` میدادم به `syntax error` برمی‌خوردم که می‌گفت باید ورودی های تابع `snr` باید `double` باشد در نتیجه هر دو سیگنال را به این نوع تبدیل کردم .
علت اینکه دو نویز را از هم کم کرده‌ام این بود که باید نویز خالص را از این روش به دست می‌آوردم.

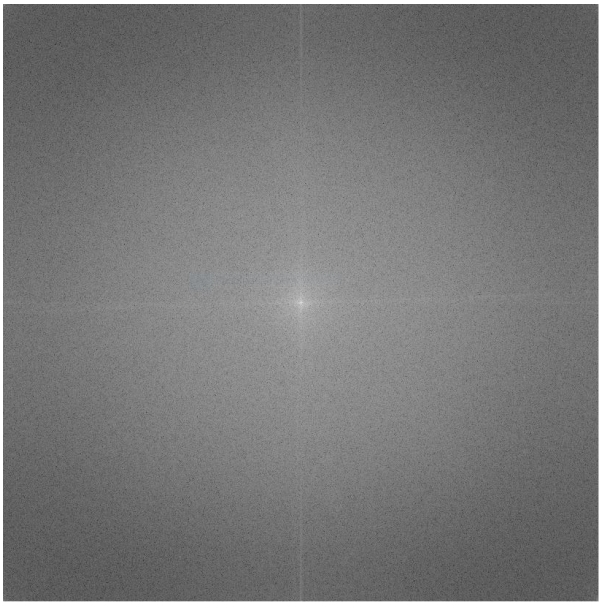
```
signalNoiseRatio = 15.6827
```

۷.۱ گام هفتم

در این گام سعی به گرفتن تبدیل فوریه از تصویر خاکستری شده و آن را طبق راهنمایی گفته شده رسم می‌کنیم. همچنین برای حل این سوال با سرچ و خواندن مستندات سایت متلب و همچنین دانشی که از درس بینایی کامپیوتر کسب کرده ام استفاده کردم.

۱.۷.۱ بررسی کد

```
% STEP 7
figure;
ft = fftshift(log(abs(fft2(grayImg)))); % log is for
    reducing the large difference between max and data
imshow(ft, []);
imwrite(ft, "fourierImage.jpg")
% ----------------------
```



شکل ۴.۱: تصویر تبدیل فوریه گرفته شده

۲.۷.۱ تحلیل عکس بالا

طی جست‌وجو های پی‌درپی نتایج دریافت‌شده از آنها را در قالب چند نکته اشاره می‌کنم:

- هر پیکسل در عکس تبدیل فوریه نشان دهنده یک موج سینوسی دو بعدی با فرکانس وابسته به فاصله از مرکز می‌باشد.
- نتیجه تصویر نشان می‌دهد که عکس حاوی تمامی فرکانس‌ها می‌باشد اما بزرگی آن هر چه از مرکز تصویر دورتر می‌شویم کمتر می‌شود و در نتیجه فرکانس بیشتر می‌شود.
- فرکانس‌های کمتر حاوی اطلاعات بیشتری نسبت به فرکانس های بالاتر هستند.
- به طور معمول وسط عکس شامل فرکانس های بالاتر بوده ولی بعد از استفاده از دستود اوضاعfftshift تغییر کرد و وسط عکس نشانگر فرکانس های low شد.

۳.۷.۱ پاسخ به سوال‌ها

۱. **مرکز تصویر چرا از همه نقاط دیگر نورانی تر است؟** به دلیل اینکه بخش زیادی از عکس فرکانس کمتری دارند و مرکز تصویر حاوی اطلاعات بیشتری است. به تعبیری دیگر بخش زیادی از تصویر اصلی ما داری تغییرات کم فرکانس هستند یعنی رنگ آنها به یک‌باره از سفید به سیاه تغییر نمی‌کند. همچنین نقطه مرکز دارای فرکانس صفر است و از جمع مقادیر پیکسل ها (میانگین پیکسل ها) به دست می آیدو

۲. **چرا هر چه از مرکز دورتر می‌شویم نقاط کم نورتر می شوند؟** چون این نقاط فرکانس بیشتر را در بر می‌گیرند در واقع این نفاط نشان‌دهنده تغییر ناگهانی (فرکانس بالا) در عکس اصلی می‌باشند.

۳. **بالا و پایین‌ترین فرکانس در تصویر کدام نقاط است؟** مرکز دارای کمترین فرکانس و هرچه از مرکز دورتر می‌شویم فرکانس بیشتر می‌شود. بنابراین گوشه های تصویر دارای بیشترین فرکانس می‌باشند.

۸.۱ گام هشتم

برای حل این سوال با توجه به صورت سوال ابتدا از تصویر تبدیل فوریه می گیریم. سپس آن را شیفت می دهیم. بعد از آن فرکانس های پایین که در مرکز تصویر تبدیل فوریه قرار دارند را نگه می داریم و بقیه آن ها را دور میریزیم. اصطلاحا آن را ماسک می کنیم. برای پیاده سازی نیز از سایت geeks for geeks کمک گرفتم که در لینک های آخر گزارش قابل مشاده است. پس از ماسک کردن فرکانس های بالا و نگه داشتن فرکانس های پایین از تصویر تبدیل فوریه معکوس میگیریم و قسمت حقیقی آن را نگه می داریم. در تصویر حاصل نویز رفع شده است ولی کمی کیفیت تصویر کاهش یافته است و لبه های آن از بین رفته اند .

همچنین روش دوم یا جایگزینی هم وجود دارد که به شرح زیر است. با کمک گرفتن از مستند اصلی سایت

متلب از روش $median$ (فیلتر میانه) استفاده می‌کنیم که این متد با کمک $3-by-3 neighborhood$ پیاده‌سازی شده‌است. فیلتر میانه که یک فیلتر مرتبه ای است برای نویز نمک و فلفل بسیار مناسب است. همچنین این فیلتر برای حذف نویز هایی که در تصویر ما وجود دارد بسیار مناسب است.

۱.۸.۱ بررسی کد

```
% STEP 8

%%% 1st method %%% https://www.geeksforgeeks.org/matlab-ideal-lowpass-filter-in-image-processing/

% Saving the size of the input_image in pixels-
% M : no of rows (height of the image)
% N : no of columns (width of the image)
[M, N] = size(noisedPicture);

% Getting Fourier Transform of the input_image
% using MATLAB library function fft2 (2D fast fourier transform)
FT_img = fft2(double(noisedPicture));

% Assign Cut-off Frequency
% D0 = 30; % one can change this value accordingly
D0 = 270; % 120 - 150 - 250 - 300 - 350

% Designing filter
u = 0:(M-1);
idx = find(u>M/2);
u(idx) = u(idx)-M;
v = 0:(N-1);
```

```
23 idy = find(v>N/2);
24 v(idy) = v(idy)-N;
25
26 % MATLAB library function meshgrid(v, u) returns
27 % 2D grid which contains the coordinates of vectors
28 % v and u. Matrix V with each row is a copy
29 % of v, and matrix U with each column is a copy of u
30 [V, U] = meshgrid(v, u);
31
32 % Calculating Euclidean Distance
33 D = sqrt(U.^2+V.^2);
34
35 % Comparing with the cut-off frequency and
36 % determining the filtering mask
37 H = double(D <= D0);
38
39 % Convolution between the Fourier Transformed
40 % image and the mask
41 G = H.*FT_img;
42
43 % Getting the resultant image by Inverse Fourier
       Transform
44 % of the convoluted image using MATLAB library function
45 % ifft2 (2D inverse fast fourier transform)
46 output_image = real(ifft2(double(G)));
47
48 % Displaying Input Image and Output Image
49 % subplot(2, 1, 1), imshow(noisedPicture),
50 % subplot(2, 1, 2), imshow(output_image, [ ]);
51 figure;
```

```
imshow(noisedPicture)
figure;
imshow(output_image, [ ])
imwrite(output_image, "1st.jpg")

% ----------------

%%% 2nd method %%% https://www.mathworks.com/help/images/ref/medfilt2.html
Kmedian = medfilt2(noisedPicture);
figure;
imshow(Kmedian);
imwrite(Kmedian, "2nd.jpg")
```

شکل ۵.۱: تصویر با نویز رفع‌شده

۲.۸.۱ عملکرد رفع نویز

با توجه به راهنمایی گفته‌شده و تابع ازپیش‌تعریف‌شده $PSNR$ به بررسی این تابع می‌پردازیم . هرچه مقدار خروجی این تابع بیشتر باشد ما بهتر عمل کرده‌ایم

range PSNR

- برای تصویر ها و ویدئو های فشرده شده بین ۳۰ تا ۵۰ دسی‌بل می‌باشد که ۸ بیتی است.
- برای ۱۲ بیتی خروجی از ۶۰ به بالا خوب است
- مقدارهای قابل‌قبول برای انتقال‌های بی‌سیم بین ۲۰ تا ۲۵ می‌باشد.

```
% PEAK SNR % https://www.mathworks.com/help/images/ref/
    psnr.html
[peaksnr, outputSNR] = psnr(noisedPicture , grayImg);
fprintf('\n The Peak-SNR value is %0.4f', peaksnr );
% ----------------------------
```

```
The Peak-SNR value is 20.4128
```

توضیح تکمیلی PSNR

PSNR یک معیار است که برای ارزیابی کیفیت تصاویر مورد استفاده قرار می‌گیرد. این معیار با توجه به نسبت بیشینه سیگنال به نویز، ارزیابی کیفیت تصویر را انجام می‌دهد.

PSNR بر اساس مقایسه سیگنال واقعی (تصویر اصلی یا مرجع) با تصویر بازسازی شده (تصویر پردازش شده) اندازه‌گیری می‌شود. برای محاسبه PSNR، ابتدا نسخه بازسازی شده تصویر باید با تصویر اصلی مقایسه شود و سیگنال خطا (تفاوت بین دو تصویر) یا نویز محاسبه می‌شود. سپس با استفاده از فرمول مربوطه، PSNR بر اساس مقدار بیشینه ممکن برای سیگنال واقعی (با توجه به بیت‌های نمایش) و مقدار میانگین مربعات خطا (MSE) محاسبه می‌شود.

به طور کلی، مقادیر PSNR بیشتر نشان دهنده کیفیت بالاتر تصویر هستند. افزایش مقدار PSNR به معنای کاهش نویز و افزایش تطابق تصویر با تصویر اصلی است. با این حال، باید توجه داشت که PSNR تنها یکی از معیارهای موجود برای ارزیابی کیفیت تصویر است و ممکن است به عنوان یک معیار کمی، به برخی جنبه‌های کیفیت تصویر، مانند واقعیت رنگ یا وضوح جزئیات، توجه نکند.

بنابراین، PSNR می‌تواند به عنوان یک معیار کمکی برای ارزیابی کیفیت تصاویر مورد استفاده قرار گیرد، اما برای ارزیابی کامل و جامع تصاویر، معیارهای دیگری مانند $SSIM(StructuralSimilarityIndex)$ و $VIF(VisualInformationFidelity)$ نیز ممکن است مورد استفاده قرار بگیرند.

Bibliography

[1] *2-D median filtering*. URL: https://www.mathworks.com/help/images/ref/medfilt2.html.

[2] *Add noise to image*. URL: https://www.mathworks.com/help/images/ref/imnoise.html.

[3] *Basic Image Import, Processing, and Export*. URL: https://www.mathworks.com/help/images/image-import-and-export.html.

[4] *chat GPT-3*. URL: https://chat.openai.com/.

[5] *Convert RGB image or colormap to grayscale*. URL: https://www.mathworks.com/help/matlab/ref/rgb2gray.html.

[6] *JPG, PNG, BMP, and TIFF Images*. URL: https://creativemarket.com/blog/difference-between-jpg-png-bmp-tiff-images.

[7] *MATLAB - Ideal Lowpass Filter in Image Processing*. URL: https://www.geeksforgeeks.org/matlab-ideal-lowpass-filter-in-image-processing/.

[8] *Signal-to-Noise Ratio Calculator*. URL: https://www.omnicalculator.com/physics/signal-to-noise-ratio.

[9] *What Is "Energy" in Image Processing?* URL: https://www.baeldung.com/cs/energy-image-processing.