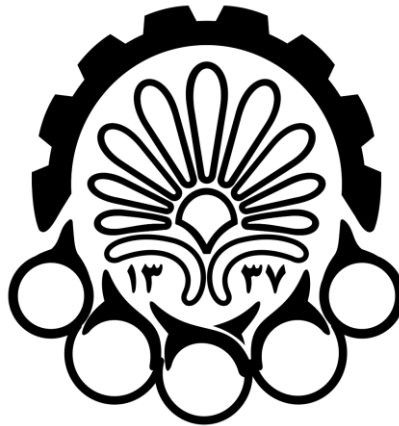


«*In The Name Of GOD*»



دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

[HW-04-Report]

[DIGITAL IMAGE PROCESSING]

Hasan Masroor | [403131030] | May 29, 2025

"فهرست مطالب تمرین 04"

Problem 1	3
A)	3
B)	4
Problem 2	5
A)	5
B)	6
C)	7
D)	10
E)	10
Problem 3	11
A)	11
B)	12
C)	14
D)	15
E)	17
F)	18

Problem 4	19
A)	19
B)	21
Problem 5	23
Problem 6	25
A)	25
B)	25
C)	26
D)	27
E)	28
F)	28

Problem 1: Getting Familiar with the applications of Image Interpolation

A.

در این تمرین با کاربردهای مهم درونیابی تصویر آشنا می‌شویم. ابتدا با اعمال اعوجاج‌های مختلف مانند موجی، پیچشی و اعوجاج لنزی روی تصویر، فرآیند تغییر Image Warping را تجربه می‌کنیم. سپس با استفاده از تکنیک مورفینگ، به‌صورت پیوسته یک تصویر را به تصویر دیگر تبدیل می‌کنیم و مراحل میانی این تغییر را تولید و نمایش می‌دهیم.

ابتدا کتابخانه‌های مورد نیاز را import کردیم و تصویر "Brad.jpg" را لود کردیم، سپس سه نوع الگوریتم وارپینگ طراحی و پیاده‌سازی کردیم تا جلوه‌های متفاوتی از اعوجاج تصویر ایجاد کنیم. در روش اول، با اعمال یک تابع سینوسی بر مختصات افقی پیکسل‌ها، یک اعوجاج موجی در راستای افقی پدید آوردیم. در روش دوم، با استفاده از مدل اعوجاج بشک‌های (Barrel Distortion)، پیکسل‌ها را بر اساس فاصله آن‌ها از مرکز تصویر به بیرون یا داخل جابه‌جا کردیم و جلوه‌ای شبیه به لنزهای واید ایجاد شد. در روش سوم، با تغییر زاویه پلار نقاط بر اساس فاصله‌شان از مرکز، افکت چرخشی یا مارپیچی (Swirl) تولید کردیم. برای همه این تبدیل‌ها، با استفاده از تابع cv2.remap و تکنیک درونیابی دوبعدی، مختصات جدید پیکسل‌ها به تصویر اصلی اعمال شد و تصاویر نهایی تولید شدند. در نهایت برای مشاهده تأثیر هر نوع وارپ نتایج را در خروجی نمایش دادیم:

Original Image



Wave Warp



Barrel Distortion



Swirl Warp



B.

در این بخش، هدف ما پیاده‌سازی تکنیک Image Morphing برای تبدیل تدریجی تصویر "Clinton.jpg" به تصویر "Cruz.jpg" بود. ابتدا هر دو تصویر را بارگذاری و به فضای رنگی RGB تبدیل کردیم، سپس تصویر مقصد را به اندازه تصویر مبدا تغییر اندازه دادیم تا ابعاد یکسانی داشته باشند. برای انجام مورفینگ، از تکنیک ترکیب خطی بر پایه پارامتر آلفا استفاده کردیم؛ به این صورت که با افزایش تدریجی مقدار آلفا از ۰ تا ۱، یک دنباله از تصاویر میانی تولید شد که هرکدام ترکیبی از دو تصویر اصلی با نسبت مشخصی بودند. در مجموع ۱۲ تصویر (شامل تصویر شروع، ۱۰ تصویر میانی و تصویر پایان) تولید و به صورت شبکه‌ای به نمایش درآمدند. خروجی این بخش و تبدیل تدریجی تصویر اولیه Clinton به Cruz را در ادامه می‌توانیم مشاهده کنیم:



Problem 2: Getting Familiar with the Wiener filtering



در این تمرین، هدف شبیه‌سازی و بازسازی تصاویر دچار تاری حرکتی است. ابتدا باید با استفاده از یک کرنل PSF مناسب (طول ۱۵ پیکسل و زاویه ۳۰ درجه) تاری حرکتی را روی تصویر اعمال کرده و نویز گاوسی به آن اضافه کنیم. سپس با محاسبه تبدیل فوریه دوبعدی تصویر اصلی و تصویر مخدوش، طیف فرکانسی آن‌ها را مقایسه کرده و از طریق مشاهده خطوط تاریک موازی در دامنه فرکانس، پارامترهای طول حرکت و زاویه را تخمین بزنیم. در ادامه، با استفاده از این مقادیر، بازسازی تصویر را به روش فیلتر معکوس ساده و فیلتر وینر در حوزه فرکانس انجام داده و کیفیت بازیابی را از نظر باقی‌مانده تاری، نویز و اثر رینگینگ بررسی نماییم.

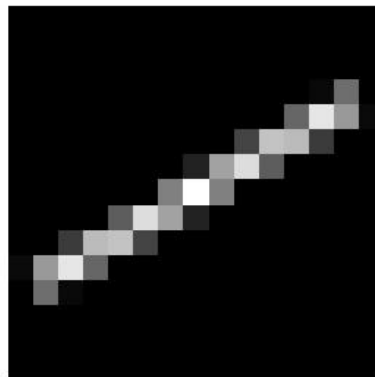
ابتدا کتابخانه‌های مورد نیاز را import کردیم و در این بخش، هدف ما مدل‌سازی یک فرایند تخریب تصویر است که شامل دو مؤلفه اصلی می‌باشد Motion Blur و Gaussian Noise می‌باشد. ابتدا با استفاده از پارامترهای مشخص شده شامل طول تاری $L=15$ و زاویه $\theta=30$ ، یک تابع پخش نقطه‌ای (PSF) طراحی کردیم. این PSF به صورت یک کرنل خطی در مرکز ماتریس ایجاد شده و با استفاده از rotation به زاویه دلخواه، شکل‌دهی می‌گردد؛ سپس کرنل نرمال‌سازی می‌شود. در گام بعد، تصویر "cameraman.png" با کرنل تاری به صورت دوبعدی کانوالو شده و بدین ترتیب یک تصویر تار به دست می‌آید که رفتار تاری حرکتی را شبیه‌سازی می‌کند. به این تصویر تار، نویز گاوسی با میانگین صفر و انحراف معیار $\sigma=1$ افزوده می‌شود تا شرایط تخریب در دنیای واقعی که شامل هم تاری و هم نویز است، بهتر مدل شود.

در نهایت هر چهار مرحله شامل تصویر اصلی، کرنل تاری (PSF)، تصویر مات شده و تصویر نهایی تخریب شده را برای درک بهتر در ادامه در خروجی نمایش دادیم:

Original Image



Motion PSF



Motion Blurred



Degraded (Blur + Noise)



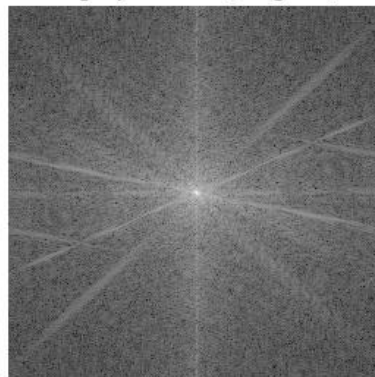
B.

در این بخش، با هدف بررسی تأثیر تخریب حاصل از تاری حرکتی و نویز گوسی بر محتوای فرکانسی تصویر، اقدام به محاسبه و تحلیل طیف لگاریتمی بزرگی تصویر اصلی و تصویر تخریب شده می‌کنیم. ابتدا با استفاده از تبدیل فوریه دوبعدی (2D FFT) و سپس با اعمال جابه‌جایی مرکز طیف (FFT shift)، طیف فرکانسی تصاویر را به دست می‌آوریم. در ادامه، برای افزایش قابلیت مشاهده جزئیات، از مقیاس لگاریتمی برای نمایش بزرگی طیف استفاده می‌کنیم. هر دو تصویر (اصلی و تخریب شده) به نوع داده‌ی float32 تبدیل شده‌اند تا دقت کافی در محاسبات تضمین شود. سپس طیف‌های حاصل را به صورت تصویری نمایش داده‌ایم. این نمایش بصری به ما امکان می‌دهد تا نقاط قوت انرژی، الگوهای تناوبی و اثرات مخرب نویز و تاری را در حوزه فرکانس به وضوح مشاهده کنیم. در نهایت نیز نتایج خروجی و طیف‌ها را در ادامه نمایش دادیم:

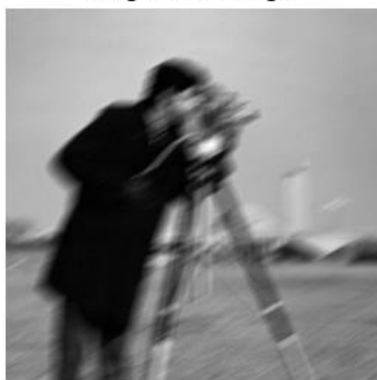
Original Image



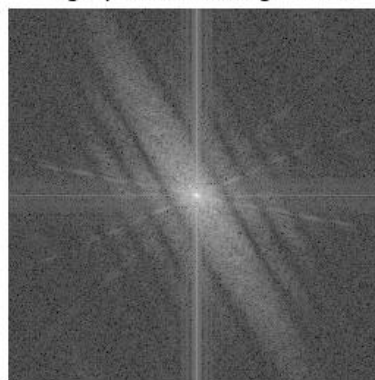
Log Spectrum (Original)



Degraded Image



Log Spectrum (Degraded)



طیف‌های بزرگ‌نمایی لگاریتمی تصویر اصلی و تخریب‌شده تفاوت‌های فرکانسی را به‌خوبی نشان می‌دهند. طیف تصویر اصلی با پراکندگی گسترده فرکانس‌های بالا و شدت بیشتر، حاکی از وجود جزئیات واضح (مانند لبه‌ها) است. در مقابل، طیف تصویر تخریب‌شده کاهش قابل‌توجهی در فرکانس‌های بالا و الگوهای کشیده (به دلیل تاری حرکت) دارد که نشان‌دهنده از بین رفتن جزئیات ریز و تأثیر محوشدگی و نویز است.

C.

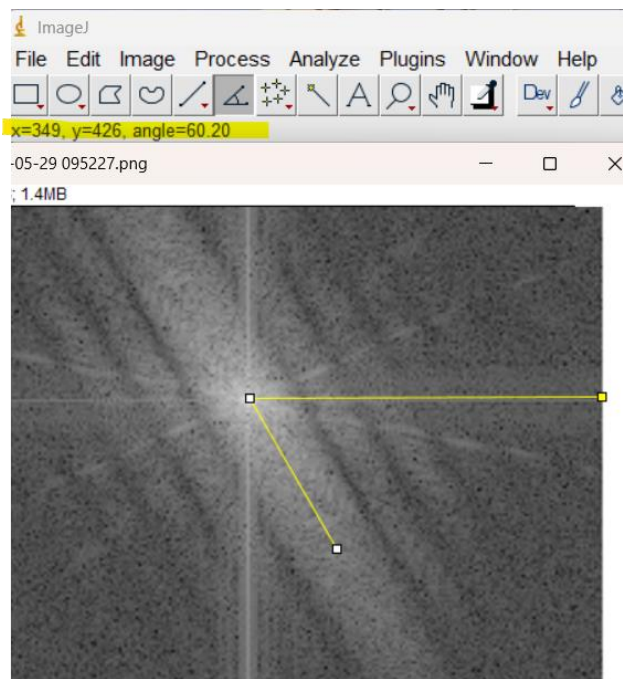
این بخش نیازمند اندازه‌گیری دستی زاویه حرکت (θ) و طول تاری (L) از طیف فوریه تصویر تخریب‌شده بخش قبل است. این فرآیند با بررسی خطوط تیره موازی (صفرهای PSF) و بهره‌گیری از روابط زیر می‌باشد:

$$\theta = \phi_{\text{lines}} + 90^\circ, \quad L = \frac{1}{\Delta u}$$

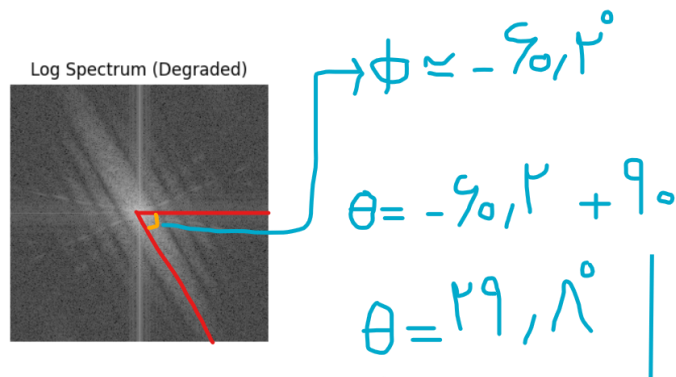
ابتدا مقدار ϕ را با استفاده از طیف تصویر تخریب‌شده در پارت قبلی پیدا می‌کنیم، سپس θ را طبق فرمول بالا بدست می‌آوریم و در نهایت نیز مقدار L را محاسبه کردیم.

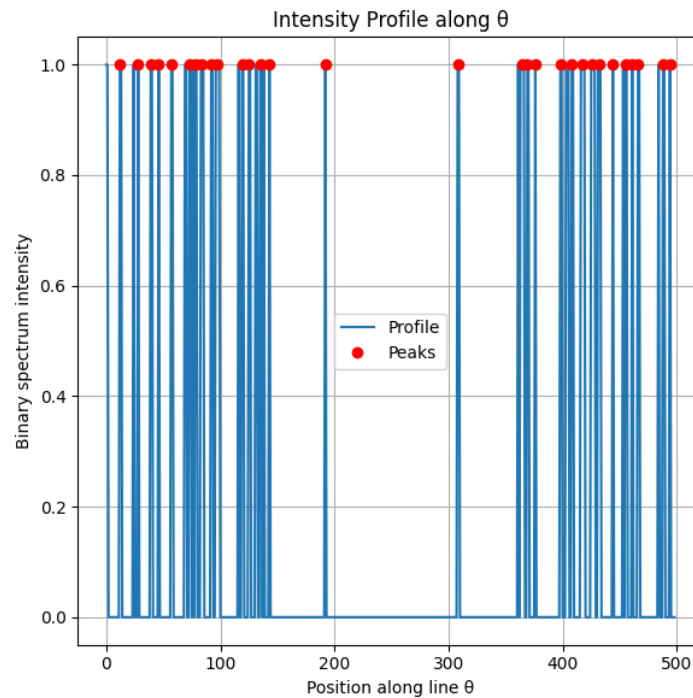
برای محاسبه‌ی زاویه حرکت θ از روی تصویر طیف فوری تصویر تخریب شده از نرم افزار ImageJ استفاده کردیم که ابتدا زاویه ϕ را بدست بیاوریم. مراحل که انجام دادیم به شرح زیر است:

تصویر طیف فوری (log-magnitude spectrum) در ImageJ بارگذاری شد. با استفاده از ابزار Angle Tool در ImageJ، زاویه‌ی بین خطوط تاریک فرکانسی (که مربوط به صفرهای تابع انتقال PSF هستند) اندازه گیری شد. زاویه‌ی حاصل شده از خطوط تاریک در طیف، مقدار $\phi \approx -60.2$ بدست آمد:



$$\theta = \phi + 90^\circ$$





Distance between dark lines (d): 16.62 pixels
Delta u (frequency spacing): 0.0649
Estimated motion length (L): 15.40 pixels

07 اگر تصویر $N \times N$ باشد و فاصله بین دو خط تیره در تصویر طیف هم برابر d باشد مقدار Δu برابر $\frac{d}{N}$ می باشد و وقتی Δu را داشتیم می توانیم L را نیز حساب کنیم:

$$L = \frac{1}{\Delta u}$$

09 تصویر cameraman 256×256 است پس N برابر 256 است و فقط کافی است d را پیدا کنیم و بر آن محاسبه آن از

10 گویاتون موجود در پارت C استفاده کردیم که فاصله بین دو خط تیره را حدود 16,62 پیکسل بدست آوردیم و حال

داریم:

$$\Delta u = \frac{d}{N} = \frac{16,62}{256} \approx 0,06492$$

12

$$L = \frac{1}{\Delta u} = \frac{1}{0,06492} \approx 15,4 \text{ پیکسل}$$

13

2 $L = 15,4$ و $\theta = 29,8^\circ$

D, E

در این بخش با هدف بازسازی تصویر تخریب شده، دو روش فیلتر معکوس (Inverse Filtering) و فیلتر وینر (Wiener Filtering) را پیاده سازی کرده ایم. ابتدا با فرض آگاهی تقریبی از طول و زاویه تاری، PSF مربوط به تاری حرکتی به صورت دستی تولید شده و برای هم ترازی با ابعاد تصویر، در مرکز تصویر صفر پد و جابه جا شده است. سپس طیف فوریه هر دو تصویر تخریب شده و PSF محاسبه شده و با استفاده از آن ها، بازسازی اولیه با فیلتر معکوس انجام شده است. برای جلوگیری از ناپایداری در فرکانس هایی که پاسخ PSF به صفر میل می کند، یک آستانه ی کوچک (ϵ) لحاظ شده است. در ادامه، با تخمین واریانس نویز و تصویر، نسبت K مورد نیاز برای فیلتر وینر محاسبه گردیده و در صورت خروج از بازه ی پایدار، مقدار پیش فرض مناسب جایگزین شده است. در نهایت، با استفاده از فیلتر وینر که به صورت تطبیقی نسبت به میزان نویز عمل می کند، تصویر به صورت مؤثرتری بازسازی شده و نتایج هر دو روش به صورت بصری در پایین نشان داده شده اند و در ادامه به تجزیه و تحلیل آنها خواهیم پرداخت:

Wiener Filtered Image



Inverse Filtered Image



همانطور که در تصویر بالا هم مشاهده کردیم، تصاویر بازسازی شده با فیلتر معکوس (Inverse Filtering) و فیلتر وینر (Wiener Filtering) تفاوت های قابل توجهی در کیفیت و پایداری را نشان می دهند. تصویر بازسازی شده با فیلتر معکوس، به دلیل حساسیت بالای آن به نویز، الگوهای نویزی شدید و خطوط ناخواسته ای را نمایش می دهد که جزئیات اصلی (مانند دوربین و سه پایه) را تحت الشعاع قرار داده و وضوح را کاهش داده است. در مقابل، تصویر بازسازی شده با فیلتر وینر، با بهره گیری از تخمین نسبت نویز به سیگنال (K)، نویز را به طور مؤثری سرکوب کرده و جزئیات مانند لبه های دوربین و پس زمینه را بهتر حفظ کرده است، هرچند همچنان اثرات باقی مانده از تاری قابل مشاهده است.

Problem 3: Adding Different Types of Degradations to Images



در این تمرین، به بررسی فرآیندهای تخریب تصویر و روش‌های پیشرفته بازسازی آن می‌پردازیم تا درک عمیق‌تری از افت کیفیت تصاویر و تکنیک‌های اصلاح آن‌ها کسب کنیم. ابتدا تاری حرکتی به همراه نویز را شبیه‌سازی کرده و تصویر را با استفاده از تابع گسترش نقطه‌ای (PSF) مخدوش می‌سازیم و با بهره‌گیری از فیلترهای فرکانسی مانند فیلتر Wiener، بازسازی Non-blind تصویر را انجام می‌دهیم. در ادامه، بازسازی Blind Deconvolution را به منظور تخمین خودکار PSF و بازیابی تصویر پیاده‌سازی می‌کنیم. در نهایت نیز روش‌های مختلف بازسازی شامل فیلتر معکوس، Wiener، کمترین مربعات محدود شده و فیلترهای میانگین‌گیری را مقایسه و ارزیابی می‌کنیم. همچنین، فیلترهای حفظ‌کننده لبه مانند فیلتر bilateral و guided برای بازسازی تصاویر با حفظ جزئیات بررسی می‌شوند و همچنین یک پایپلاین کامل برای بازسازی تصاویر واقعی با تخمین PSF و بهینه‌سازی نهایی تصویر طراحی و پیاده‌سازی می‌گردد.

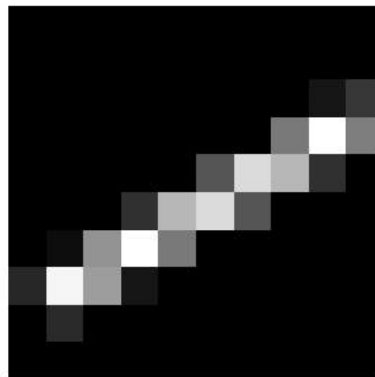
****چون در سوال به ورودی اشاره‌ای نشده بود و همچنین برای این تمرین ورودی به ما داده نشده بود از همان تصویر Cameraman مربوط به سوال دوم در اینجا نیز استفاده کردیم.****

ابتدا کتابخانه‌های مورد نیاز را import کردیم و در این بخش، فرآیند تخریب تصویر با شبیه‌سازی تاری حرکت و نویز گاوسی سفید بر روی تصویر cameraman.png انجام شد. ابتدا تابع PSF برای تاری خطی با طول $L=10$ پیکسل و زاویه $\theta=30$ به صورت دستی با استفاده از یک هسته خطی و چرخش طراحی شد و از پیچش دوبعدی برای اعمال آن بر تصویر استفاده شد؛ سپس نویز گاوسی با واریانس 0.002 به تصویر محوشده اضافه شد تا نسخه تخریب‌شده تولید شود. نتایج شامل تصویر اصلی، PSF، تصویر محوشده و تصویر محوشده با نویز به صورت بصری را برای درک بهتر در ادامه نمایش دادیم و در بخش‌های بعدی از آن استفاده می‌کنیم:

Original



Motion Blur PSF



Motion Blurred



Motion Blurred + Noise

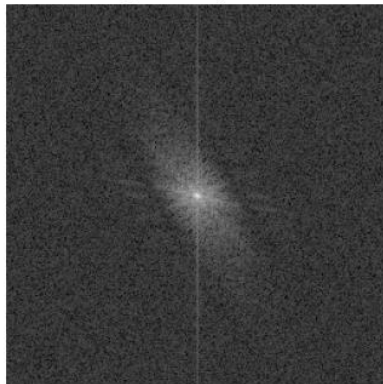


B.

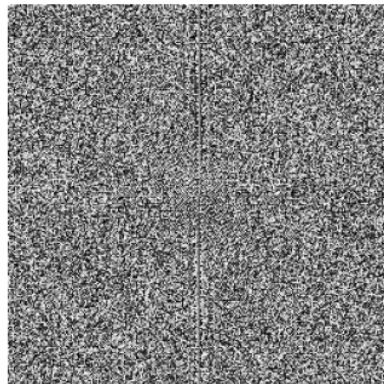
در این پارت فیلتر وینر برای بازسازی تصویر تخریب شده که تحت تأثیر تاری حرکت و نویز گاوسی قرار داشت، پیاده سازی شد. تابع PSF در مرکز تصویر پد شده و تبدیل های فرکانسی لازم اعمال شدند. با استفاده از یک پارامتر ثابت برای تنظیم نویز، تصویر بازسازی شد و کیفیت آن با معیارهای PSNR و SSIM ارزیابی گردید. همچنین طیف های فرکانسی (بزرگ نمایی و فاز) تصویر تخریب شده و بازسازی شده به منظور تحلیل عمیق تر نمایش داده شدند. در نهایت، تصاویر اصلی، تخریب شده و بازسازی شده را نمایش دادیم تا در ادامه به تجزیه و تحلیل آنها بپردازیم:

Degraded Image (DFT)

Magnitude Spectrum

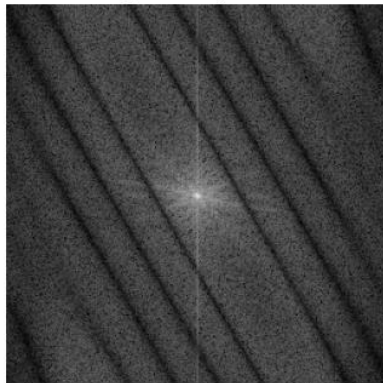


Phase Spectrum

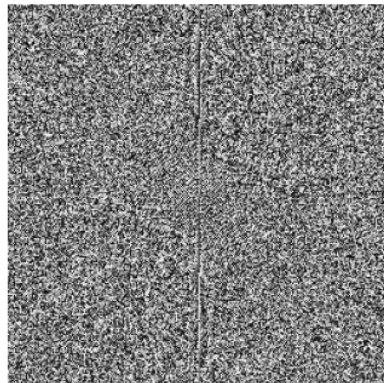


Restored Image (DFT)

Magnitude Spectrum



Phase Spectrum



Original



Degraded



Restored (Wiener)



همانطور که بالاتر گفتیم فیلتر وینر برای بازسازی تصویر تخریب شده cameraman.png که تحت تأثیر تاری حرکت ($L=10$ و $\theta=30$) و نویز گاوسی (واریانس 0.002) قرار داشت، با استفاده از پارامتر ثابت اعمال شد که منجر به تولید تصویر بازسازی شده ای شد. ارزیابی کمی نشان داد که PSNR برابر 19.02 دسی بل و SSIM برابر 0.2644 است، مقادیری که حاکی از بهبود نسبی کیفیت نسبت به تصویر تخریب شده است، هرچند SSIM پایین نشان دهنده محدودیت در حفظ ساختارهای جزئی تصویر است. طیف فرکانسی تصویر تخریب شده، با تمرکز شدت در فرکانس های پایین و پراکندگی ناشی از نویز کاهش وضوح را تأیید می کند، در حالی که طیف تصویر بازسازی شده الگوهای خطی ناشی از تاری را کاهش داده و بازایی نسبی فرکانس های بالا را نشان می دهد. مقایسه بصری تصاویر اصلی، تخریب شده و بازسازی شده نیز بیانگر کاهش نویز و بهبود جزئیات (مانند لبه های دورین و سه پایه) در تصویر وینر است، هرچند اثرات باقی مانده تاری و نویز همچنان مشهود بوده و نشان دهنده نیاز به تنظیم دقیق تر پارامتر K یا استفاده از روش های تکمیلی است.

C.

در این بخش، الگوریتم دیکانولوشن Blind با استفاده از روش ریچاردسون-لوسی برای بازسازی تصویر تخریب شده (degraded_output.png) و تخمین تابع PSF پیاده سازی شد. ابتدا تصویر لود و نرمال سازی شد، سپس PSF اولیه با یک تابع دلتا مقداردهی اولیه گردید. فرآیند بازسازی طی 50 تکرار انجام شد که در هر مرحله، تصویر و PSF به صورت متناوب به روزرسانی شدند (تصویر با استفاده از نسبت تصویر محو به پیش تخمینی اصلاح شد و PSF با استخراج بخش مرکزی پیش معکوس و اعمال تنظیم کننده به روزرسانی گردید). برای پایداری، PSF تخمین شده با فیلتر گاوسی (اندازه 3×3 ، $\sigma=0.5$) صاف و نرمال سازی شد. در نهایت، تصویر بازسازی شده و PSF تخمین شده تولید شدند و یک تابع ارزیابی برای مقایسه PSF تخمین شده با PSF واقعی (در صورت وجود) طراحی شد که در اینجا به دلیل نبود PSF واقعی، صرفاً برای تحلیل بصری آماده شد.



D

در این بخش فرایند تخریب و بازسازی تصویر را با ترکیب دو نوع تخریب رایج، شامل تاری حرکتی همراه با نویز گوسی و تاری گوسی همراه با نویز پواسون پیاده‌سازی کردیم. برای ایجاد تاری حرکتی، یک تابع PSF با طول و زاویه مشخص تولید کرده و با استفاده از تبدیل فوریه، آن را روی تصویر اعمال کردیم. برای شبیه‌سازی شرایط واقعی‌تر، نویز گوسی به تصویر تار شده افزودیم. در سناریوی دوم تصویر را با فیلتر گوسی مات کرده و سپس نویز پواسون را به آن اضافه کردیم. برای بازسازی تصاویر تخریب شده، چهار روش مختلف فیلتر معکوس (Inverse Filtering)، فیلتر وینر (Wiener Filtering)، فیلتر حداقل مربعات محدود (Constrained Least Squares) و فیلتر میانه (Median Filtering) را به کار گرفتیم. در حالت تاری گوسی، برای کاهش نویز اولیه، ابتدا پیش‌پردازشی شامل اعمال فیلتر گوسی با سیگمای کم انجام دادیم که کمی نتایج خروجی بهبود پیدا کند.

- در ادامه، کیفیت تصاویر بازسازی شده را با استفاده از معیارهای PSNR و SSIM ارزیابی کردیم و نتایج مربوط به Motion blur + Gaussian noise را ابتدا نمایش دادیم:

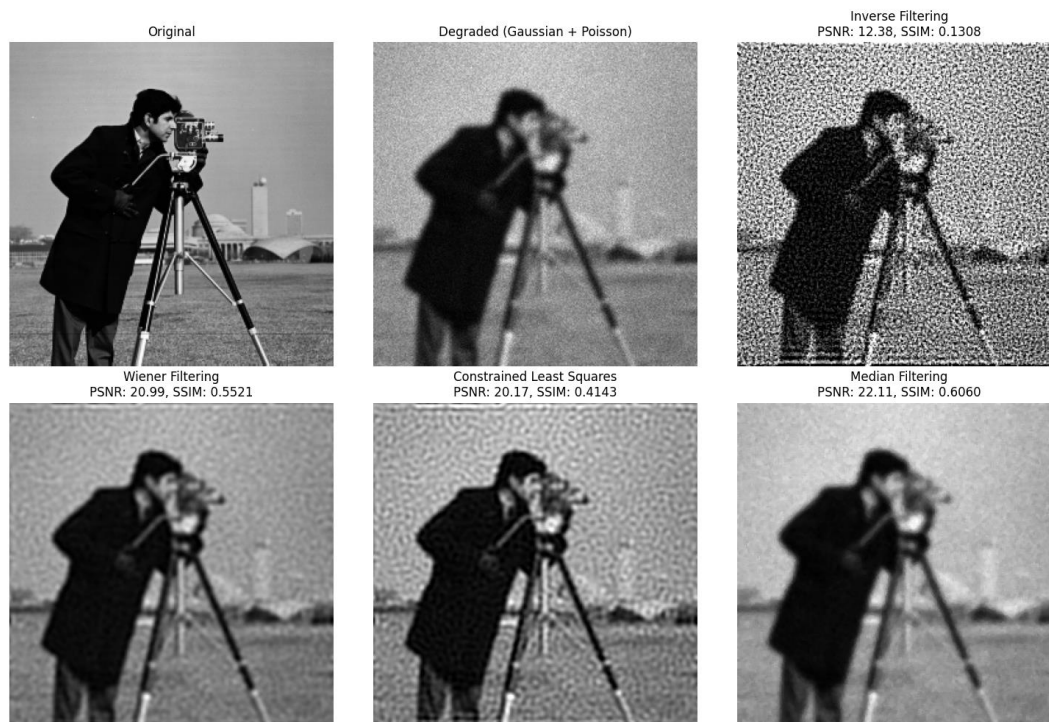


Scores for Motion Blur + Gaussian Noise:

	PSNR	SSIM
Inverse Filtering	16.957988	0.255334
Wiener Filtering	24.418715	0.675984
Constrained Least Squares	25.340268	0.755603
Median Filtering	19.622257	0.595735

ارزیابی کمی تصاویر بازسازی شده با تاری حرکت و نویز گاوسی نشان داد که فیلتر Constrained Least Squares با PSNR برابر 25.34 دسی بل و SSIM برابر 0.7556 بهترین عملکرد را دارد و پس از آن فیلتر ویتر با PSNR برابر 24.42 دسی بل و SSIM برابر 0.6760 قرار گرفت که بیانگر حفظ بهتر جزئیات و کاهش نویز است. فیلتر معکوس با PSNR برابر 16.96 دسی بل و SSIM 0.2553 ضعیف ترین نتیجه را داشت و تحت تأثیر تقویت نویز و رینگینگ قرار گرفت، در حالی که فیلتر میانه با PSNR برابر 19.62 دسی بل و SSIM 0.5957 عملکرد متوسطی ارائه داد، اما با صاف کردن بیش از حد وضوح را کاهش داد. مقایسه بصری تصاویر نیز نشان داد که فیلتر Constrained Least Square و ویتر جزئیات لبه ها را بهتر بازیابی کرده اند، در مقابل فیلتر معکوس نویز شدید و فیلتر میانه افت وضوح را نشان داد.

-در ادامه نیز خروجی و نتایج مربوط به Gaussian blur + Poisson noise را نمایش می دهیم و به تجزیه و تحلیل آنها می پردازیم:



Scores for Gaussian Blur + Poisson Noise:		
	PSNR	SSIM
Inverse Filtering	12.382811	0.130842
Wiener Filtering	20.991626	0.552051
Constrained Least Squares	20.168372	0.414300
Median Filtering	22.108010	0.606007

ارزیابی کمی تصاویر بازسازی شده با تاری گاوسی و نویز پواسون نشان داد که فیلتر میانه با PSNR برابر 22.11 دسی بل و SSIM برابر 0.6060 بهترین عملکرد را دارد که بیانگر توانایی آن در کاهش مؤثر نویز پواسون و حفظ ساختارهای تصویر است. فیلتر وینر با PSNR برابر 20.99 دسی بل و SSIM 0.5521 و فیلتر Constrained Least Squares با PSNR برابر 20.17 دسی بل و SSIM 0.4143 عملکرد متوسطی ارائه کردند که نشان دهنده محدودیت در بازیابی جزئیات ظریف تحت این نوع نویز است. فیلتر معکوس با PSNR 12.38 دسی بل و SSIM 0.1308 ضعیف ترین نتیجه را داشت و تحت تأثیر تقویت نویز شدید قرار گرفت. نتایج بصری بالا نیز همین مواردی که به آنها اشاره کردیم را نشان می دهند.

E

در این بخش نیز هدف ما بررسی عملکرد سه فیلتر پرکاربرد در حذف نویز گاوسی از تصویر است و سه روش متفاوت را روی تصویر نویزی اعمال کردیم: فیلتر Bilateral، فیلتر Gaussian و فیلتر Median.

برای پیاده سازی فیلتر Bilateral، یک نسخه ساده شده و بهینه شده به صورت دستی کدنویسی کردیم که با ترکیب وزن های spatial و intensity-based به حفظ لبه ها کمک می کند. برای فیلترهای گاوسی و میانه نیز از توابع آماده OpenCV استفاده کردیم. پس از اعمال فیلترها، کیفیت نتایج را با دو معیار عددی استاندارد PSNR و SSIM ارزیابی کردیم تا میزان حفظ ساختار تصویر را بررسی کنیم. در ادامه نیز نتایج خروجی این پارت را برای درک و تجزیه و تحلیل بهتر نمایش دادیم:

```
Results:
Bilateral Filter: PSNR = 33.63, SSIM = 0.9191
Gaussian Filter: PSNR = 24.45, SSIM = 0.8077
Median Filter: PSNR = 27.25, SSIM = 0.8702
```

Original



Bilateral

PSNR: 33.64, SSIM: 0.9190



Gaussian

PSNR: 24.45, SSIM: 0.8077



Median

PSNR: 27.28, SSIM: 0.8702



ارزیابی کمی نشان داد که فیلتر Bilateral با PSNR برابر 33.63 دسی بل و SSIM برابر 0.9191 بهترین عملکرد را داشت و به خوبی لبه‌ها را حفظ کرد، در حالی که نویز را به طور مؤثری کاهش داد. فیلتر میانه با PSNR برابر 27.25 دسی بل و SSIM 0.8702 عملکرد خوبی در حذف نویز ارائه داد اما جزئیات لبه‌ها را کمی صاف کرد. فیلتر گاوسی با PSNR 24.45 دسی بل و SSIM 0.8077 ضعیف‌ترین نتیجه را داشت و به دلیل صاف‌سازی یکنواخت، لبه‌ها را بیش از حد محو کرد. مقایسه بصری نیز نشان داد که فیلتر دو طرفه با حفظ وضوح لبه‌ها و کاهش نویز، برتری قابل توجهی نسبت به دو روش دیگر دارد در بازسازی با حفظ جزئیات عملکرد بهتری دارد.

F

در این بخش، هدف طراحی یک پایپ‌لاین کامل برای بازسازی تصویر واقعی مات شده (ناشی از تاری حرکتی یا خارج از فوکوس) است. این پایپ‌لاین شامل سه مرحله اصلی است: تخمین تابع PSF، بازسازی تصویر با استفاده از PSF تخمینی و پردازش Post-processing مانند افزایش وضوح و بهبود کنتراست.

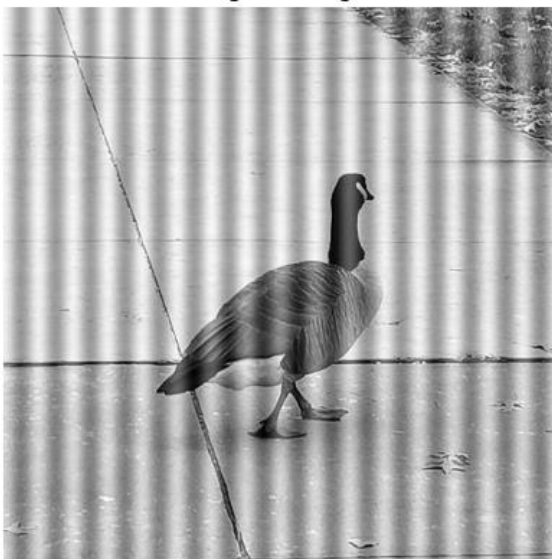
Problem 4: Applying Notch Filters to Remove Periodic Patterns

.A

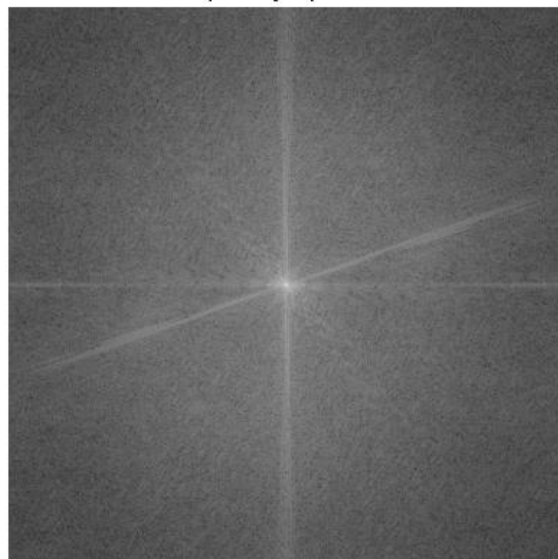
در این تمرین، با استفاده از فیلترهای ناچ (Notch Filters) به حذف الگوهای تناوبی و نویزهای تکرارشونده در تصاویر می‌پردازیم. ابتدا تصاویر داده‌شده را بارگذاری و طیف فرکانسی آن‌ها را نمایش می‌دهیم تا نویزهای تناوبی قابل مشاهده باشند. سپس با طراحی و اعمال فیلترهای ناچ مناسب، این نویزها را تضعیف کرده و تصاویر بازسازی‌شده بدون نویز را نمایش می‌دهیم.

ابتدا کتابخانه‌های مورد نیاز را import کردیم و در این بخش هدف ما حذف نویزهای تناوبی از تصاویر خاکستری با استفاده از فیلتر ناچ (Notch Filter) در حوزه فرکانس بود. ابتدا تصاویر را لود کرده و طیف فرکانسی آن‌ها را با استفاده از تبدیل فوریه دو بعدی محاسبه کردیم. سپس با تحلیل طیف بزرگی، پیک‌های فرکانسی مربوط به نویز را با کمک تکنیک تشخیص قله (Peak Detection) شناسایی کرده و نقاط متقارن آن‌ها را نیز لحاظ کردیم. در ادامه، یک ماسک فیلتر ناچ طراحی نمودیم که نواحی اطراف این پیک‌ها را تضعیف می‌کند و از آن برای فیلتر کردن فرکانس‌های مزاحم استفاده کردیم. در نهایت، با اعمال معکوس تبدیل فوریه، تصویر بازسازی‌شده و نویززدایی‌شده به دست آمد. این فرآیند برای دو تصویر مختلف انجام شد و نتایج به صورت زیر نمایش داده شده‌اند:

Original Image



Frequency Spectrum



Identified Noise Peaks

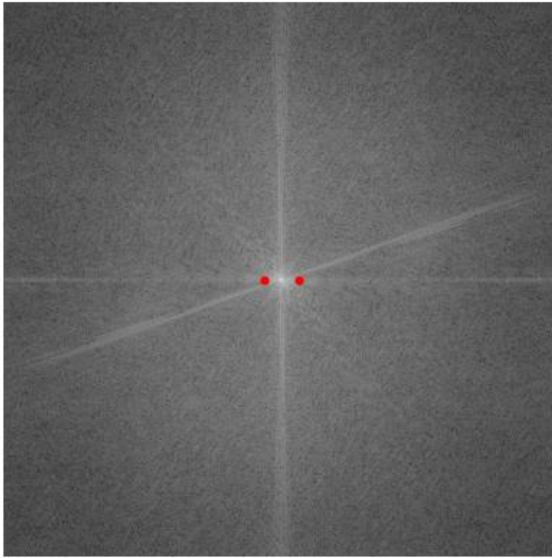
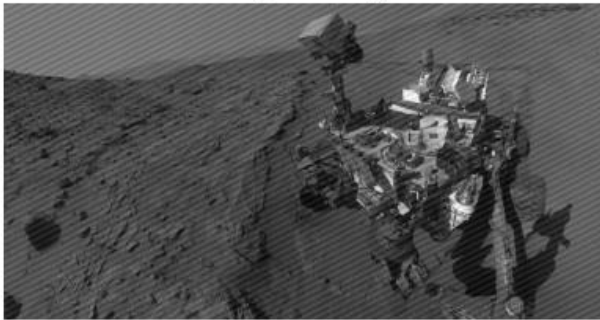


Image After Noise Removal



Original Image



Frequency Spectrum



Identified Noise Peaks

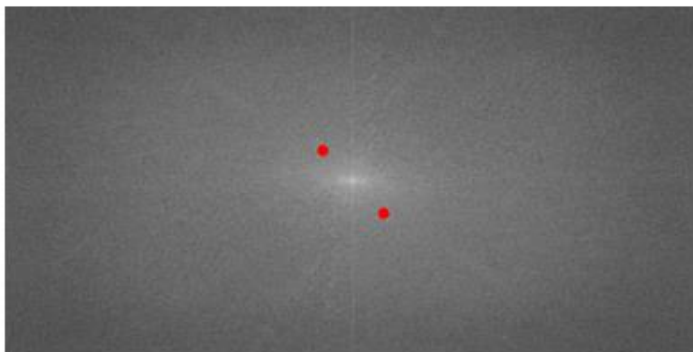


Image After Noise Removal

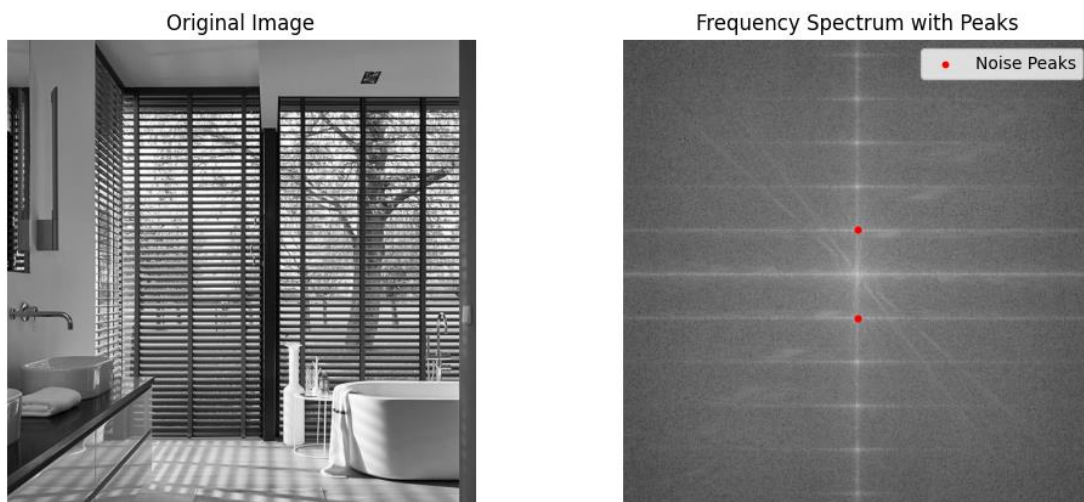


بالتر گفتیم که تکنیک فیلتر ناتچ برای حذف نویز از تصاویر `curiosity_rover_selfie.png` و `Goose.jpg` پیاده‌سازی کردیم تصویر اولیه `Goose` و `curiosity_rover` با الگوهای نواری مواجه بودند و شناسایی پیک‌های نویز در طیف فرکانسی با علامت‌گذاری نقاط قرمز نشان داد که نویز در فرکانس‌های خاص متمرکز است و اعمال فیلتر ناتچ این پیک‌ها را حذف کرد. نتیجه بصری نهایی دو تصویر ورودی مؤثر بودن روش فیلتر ناتچ در حذف نویزهای فرکانسی خاص را نشان دادند و تا حد خیلی خوبی توانستیم این کار را انجام دهیم.

B

در این مرحله با هدف حذف نویز تناوبی از تصویر، فیلتر **Butterworth Band-Reject** را در حوزه فرکانس پیاده‌سازی کردیم. ابتدا تصویر را به صورت سیاه‌وسفید بارگذاری کرده و با اعمال تبدیل فوریه دوبعدی و انتقال مرکز طیف، نمایی از انرژی فرکانسی آن به دست آوردیم. سپس با استفاده از الگوریتم شناسایی قله‌ها، نقاط دارای شدت زیاد در طیف را (غیر از مؤلفه DC) استخراج کرده و نقاط قرینه‌شان نسبت به مرکز را نیز لحاظ کردیم. براساس فاصله این پیک‌ها از مرکز، شعاع حذف (D_0) و پهنای باند (W) را برای طراحی فیلتر تعیین کردیم. ماسک فیلتر **Butterworth** را با این پارامترها تولید کرده و بر طیف فرکانسی تصویر اعمال نمودیم. در نهایت با انجام تبدیل فوریه معکوس، تصویر را بازسازی و نویز را به طور مؤثر حذف کردیم. نتیجه نهایی نشان داد که این روش قادر است مؤلفه‌های تناوبی مزاحم را بدون تخریب محتوای اصلی تصویر کاهش دهد.

خروجی این پارت را در ادامه نمایش دادیم:



Original Image



Filtered Image



همان طور که خروجی های بالا را مشاهده کردیم؛ تصویر فیلترشده با استفاده از فیلتر Notch ، الگوی دوره‌ای کرکره‌ها را با موفقیت کاهش داده و دید بهتری از منظره پشت آن فراهم کرده است. هرچند برخی جزئیات هنوز کمی محو هستند، اما وضوح کلی تصویر بهبود یافته و سایه‌های مزاحم کرکره‌ها به‌طور قابل‌توجهی حذف شده‌اند. در نتیجه عملکرد خوبی از خود نشان داده است.

Problem 5: Getting to Know Some of the Simplified JPEG Compression Steps

A.

در این تمرین، با مراحل ساده‌شده‌ای از الگوریتم فشرده‌سازی تصویر JPEG آشنا می‌شویم که روشی استاندارد و پرکاربرد برای کاهش حجم تصاویر دیجیتال، به‌ویژه در عکس‌هایی با تغییرات ملایم رنگ و روشنایی می‌باشد. یکی از اجزای کلیدی این الگوریتم، تبدیل کسینوسی گسسته (DCT) است که نقش مهمی در تمرکز اطلاعات تصویر و حذف مؤلفه‌های غیرضروری ایفا می‌کند. در این تمرین، تصویر shahnameh.png به بلوک‌های 4×4 تقسیم می‌شود و بر روی هر بلوک تبدیل DCT اعمال می‌گردد. سپس برای شبیه‌سازی مرحله کوانتیزاسیون، ۷۵٪ از ضرایب DCT در هر بلوک حذف شده و تنها بخش کوچکی از اطلاعات اصلی حفظ می‌شود. در ادامه، تصویر با استفاده از ضرایب باقی‌مانده بازسازی شده و برای ارزیابی کیفیت آن، نسبت سیگنال به نویز پیک (PSNR) نسبت به تصویر اصلی محاسبه می‌گردد.

ابتدا کتابخانه‌های مورد نیاز را import کردیم و تصویر ورودی را به فضای رنگی YCrCb تبدیل کرده و هر کانال را جداگانه پردازش می‌کنیم. برای فشرده‌سازی، تصویر را به بلوک‌های 4×4 تقسیم می‌کنیم و با استفاده از تبدیل کسینوسی گسسته (DCT) ضرایب هر بلوک را استخراج می‌نماییم. سپس ۷۵٪ از ضرایب کوچک‌تر را حذف کرده و تنها بزرگ‌ترین ضرایب را نگه می‌داریم تا حجم داده کاهش یابد. پس از بازسازی هر بلوک با استفاده از تبدیل معکوس DCT یا IDCT، کانال‌ها را دوباره ترکیب و به فضای رنگی RGB بازمی‌گردانیم. کیفیت تصویر فشرده‌شده با محاسبه PSNR نسبت به تصویر اصلی ارزیابی شده و نتایج خروجی را نیز در پایین نمایش داده‌ایم:



Compressed Image (RGB)
PSNR: 43.47 dB



اگر به خروجی‌های بالا توجه کنیم می‌بینیم که تصویر فشرده‌شده کانال Y با وجود حذف ۷۵٪ ضرایب DCT، کیفیت مناسبی با PSNR برابر ۴۳.۴۷ دسی‌بل حفظ کرده است. جزئیات کلی تصویر همچنان واضح باقی مانده‌اند، اما در نواحی با تغییرات فرکانسی بالا، آثار بلوکی و نویزهای جزئی قابل مشاهده است.

Problem 6: Hands-On Image Processing: Huffman coding

A.

07 (a) می‌دانیم که فرمول آنتروپی $H = - \sum_{i=1}^n P_i \log_2 P_i$ است و ابتدا P_i که فراوانی پیکسل‌ها هستند را حساب می‌کنیم:

08

مقدار	تعداد وقوع	احتمال P
21	12	$12/32 \rightarrow 0,375$
95	4	$4/32 \rightarrow 0,125$
169	4	$4/32 \rightarrow 0,125$
243	12	$12/32 \rightarrow 0,375$

11 حالا از فرمول بالا استفاده می‌کنیم و مقدار H را بدست می‌آوریم:

12

$$H = - \sum_{i=1}^4 \frac{12}{32} \log_2 \frac{12}{32} + \frac{4}{32} \log_2 \frac{4}{32} + \frac{4}{32} \log_2 \frac{4}{32} + \frac{12}{32} \log_2 \frac{12}{32}$$

13

$$= - \left(-0,5306 \dots 0,375 \dots 0,375 \dots -0,5306 \right)$$

14

$$= 1,8112$$

B.

15 (b) برای رسم درخت هافمن از همان فراوانی‌های بخش قبل استفاده می‌کنیم و ابتدا آنها را مرتب می‌کنیم:

16

21: 0,375

243: 0,375

95: 0,125

169: 0,125

17

سورت

0,625

0,375

0,125

0,125

18

07 حالا درخت هافمن را رسم می کنیم و به عنوان مثال ساختار است را

08 و ساختار چیدمان را 1 در نظر می گیریم:

09

10

11

طول کد	کدها فین	مقدار بیکسل
1 bit	0	21
2 bit	10	243
3 bit	110	95
3 bit	111	169

12

تعداد لای بیت های لازم برای فشرده سازی با هافمن = 60

.C

13 (C) در این سوال به محاسبه نرخ فشرده سازی و اثربخشی الگوریتم هافمن

14 می پردازیم. نرخ فشرده سازی برابر با ساین تقویر اصلی قبل از

15 فشرده سازی تقسیم بر ساین بعد از فشرده سازی است:

16 هر بیکسل 3 بیت است

17

18

1403

$$\frac{(4 \times 8) \times 8}{60} = \frac{256}{60} \approx 4,27$$

یعنی داده ها حدود 4,27 برابر فشرده شده اند.

اثربخشی هم از فرمول آنتروپی تقسیم بر میانگین بیت هر بیکسل

Average bits/pixel = $\frac{60}{32} = 1,875$ داریم:

Efficiency = $\frac{\text{Entropy}}{\text{Average bits/pixel}} = \frac{1,8112}{1,875} \approx 0,966$

d. در این بخش جفت‌ها را غیر همبسته را بدست می‌آوریم:

(21, 21)	(21, 95)	(169, 243)	(243, 243)
(21, 21)	(21, 95)	(169, 243)	(243, 243)
(21, 21)	(21, 95)	(169, 243)	(243, 243)
(21, 21)	(21, 95)	(169, 243)	(243, 243)

جفت	تعداد وقوع	احتمال P_i
(21, 21)	4	$4/16$
(21, 95)	4	$4/16$
(169, 243)	4	$4/16$
(243, 243)	4	$4/16$

$$H = - \sum_{i=1}^4 P_i \log_2 P_i$$

$$H_{Pair} = -4 \times \frac{4}{16} \log_2 \frac{4}{16} = -4 \times (0.5) = 2$$

19 جمعه 1399 Feb 7 Fri 2025

حال آنتروپی بر پیکسل را حساب می‌کنیم و هر جفت شامل 2 پیکسل است:

$$H_{Per\ pixel} = \frac{H_{Pair}}{2} = \frac{2}{2} = 1$$

می‌بینیم که نسبت به آنتروپی پیکسل $a(1, 8112)$ مقدار کمتری دارد و می‌تواند فشرده‌سازی بیشتری انجام دهد.

E

07 (e) در این بخش باید اختلاف بین پیکسل‌های مجاور را بدست بیاوریم و به عنوان مثال برای ردیف اول را حساب می‌کنیم و ردیف‌های 2 و 3 و 4 نیز به همین صورت هستند:

08

21 - 21 = 0
 21 - 21 = 0
 95 - 21 = 74
 169 - 95 = 74
 243 - 169 = 74
 243 - 243 = 0
 243 - 243 = 0

09

0	0	0	74	74	74	0	0
0	0	0	74	74	74	0	0
0	0	0	74	74	74	0	0
0	0	0	74	74	74	0	0

10

11

ستون اول را چون مقداری ندانستیم گذاشتیم و ردیف اول را محاسبه کردیم و ردیف‌های بعدی هم به همین صورت هستند.

13

مقدار	تعداد وقوع	احتمال P_i
0	20	$\frac{20}{32} = 0,625$
74	12	$\frac{12}{32} = 0,375$

15

$H = - \sum_{i=1}^2 P_i \log_2 P_i = - (0,625 \log_2 0,625 + 0,375 \log_2 0,375)$

16

17

بیت بر پیکسل $= - (-0,4238 - 0,5306) = 0,9544$

می‌بینیم که نسبت به آنتروپی پیکسل (1) مقدارش کمتر شد و وقتی که اختلاف بین پیکسل‌ها را نگذاریم می‌کنیم اطلاعات کمتری برای ذخیره‌سازی نیاز داریم و می‌توانیم فشرده‌سازی بهتری داشته باشیم.

F

در بخش (a)، آنتروپی تصویر اولیه بر اساس پیکسل‌های تکی برابر با 1.8112 بیت به ازای هر پیکسل به دست آمد. این مقدار نشان‌دهنده میزان عدم قطعیت (یا تصادفی بودن) تصویر است در حالتی که هیچ گونه وابستگی مکانی بین پیکسل‌ها در نظر گرفته نشده و تنها بر اساس فراوانی شدت روشنایی پیکسل‌ها محاسبه شده است. در بخش (d)، با استفاده از جفت‌های غیرهمپوشان پیکسل، آنتروپی به 1.0 بیت به ازای هر جفت پیکسل کاهش یافت. این کاهش به این دلیل است که در این حالت، الگوهای محلی و تکرار در توالی تصویر تا حدی شناسایی

می‌شوند و مدل ساختار تصویر را بهتر درک می‌کند. بنابراین اطلاعات مورد نیاز برای توصیف تصویر کاهش می‌یابد. در بخش (e)، با ایجاد تصویر اختلافی (اختلاف شدت روشنایی بین پیکسل‌های مجاور)، آنتروپی باز هم کاهش یافته و به 0.9544 بیت به ازای هر پیکسل رسیده است. علت این کاهش بیشتر، بهره‌گیری مؤثر از همبستگی فضایی بین پیکسل‌هاست؛ به‌ویژه در نواحی یکنواخت تصویر، که اختلاف بین پیکسل‌ها معمولاً صفر یا مقدار ثابتی است و این باعث تکرار بالا و در نتیجه کاهش آنتروپی می‌شود.

کاهش تدریجی آنتروپی در مراحل مختلف این تمرین نشان می‌دهد که هرچه مدل‌سازی تصویر هوشمندانه‌تر باشد و بتواند وابستگی‌های درونی تصویر را بهتر تشخیص دهد، میزان اطلاعات مورد نیاز برای نمایش تصویر نیز کاهش می‌یابد. مدل ساده‌ی پیکسل‌تکی هیچ ساختار یا وابستگی‌ای در تصویر را در نظر نمی‌گیرد و آنتروپی بالاست. مدل جفت پیکسل، بخشی از الگوهای تکراری تصویر را ثبت می‌کند و آنتروپی را کاهش می‌دهد. مدل اختلافی، ساختار و همبستگی پیکسل‌ها را به خوبی استخراج کرده و در نتیجه کمترین آنتروپی را دارد. این کاهش آنتروپی به معنی افزایش پتانسیل فشرده‌سازی تصویر است، که یکی از اهداف اصلی در پردازش تصویر و کدگذاری داده‌ها محسوب می‌شود.

آنتروپی هر روش را مجدد برای درک بهتر در ادامه آورده‌ایم:

Method	Entropy (bits/pixel)	
Single pixels	1.8112	(a)
Pixel pairs	1	(d)
Differences	0.9544	(e)

«... خردادماه 1404 ...»