

INTRODUCTION TO COMPUTER VISION

PROJECT DOCUMENTATION



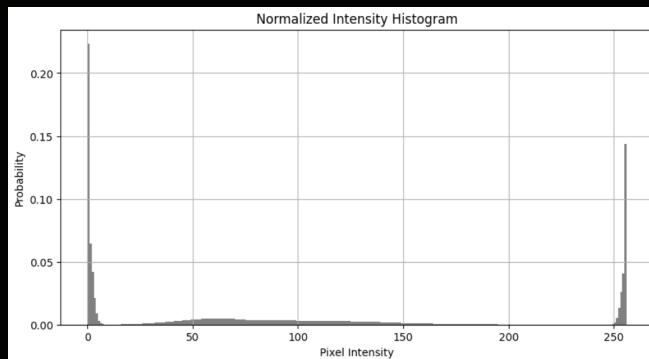
Ferdowsi University of Mashhad
Department of Computer Engineering

SPRING 2025

نام و نام خانوادگی	شماره دانشجویی
امیرحسین افشار	۴۰۱۲۶۲۱۹۶

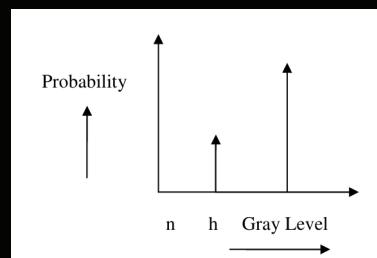
(۱) سوال اول: معماه داوینچی

در ابتدا برای حل پازل، عملیات denoising *processed_img_part_1.jpg* را برای تصویر *denoising* انجام شد. بدین منظور، از هیستوگرام تصویر بهره گرفته شده است [۱]؛ هیستوگرام تصویر با سطوح خاکستری، بر اساس نرمال کردن میزان پیکسل های با شدت بین ۰ تا ۲۵۵ ساخته می شود تا شکل function ۱ این شکل است:



شکل ۱: پلات هیستوگرام تصویر نویزی شماره ۱

این شکل، بیان می کند که تعداد پیکسل ها با مقدار نزدیک به صفر و مقدار نزدیک به ۲۵۵ بسیار زیاد است. همچنین هیستوگرام احتمال قابل انتظار برای تصاویر نویزی با نوع نویز نمک فلفل به شکل زیر است:



شکل ۲: هیستوگرام قابل انتظار برای تصاویر نویزی نمک و فلفل

[۱] Asoke Nath: Image Denoising Algorithms: A Comparative Study of Different Filtration Approaches Used in Image Restoration

با مقایسه شکل ۱ و ۲ می توان نتیجه گرفت که نویز تصویر شماره ۱ از نوع نمک و فلفل می باشد. بنابراین، برای کردن تصویر، بهترین گزینه، استفاده از فیلتر median می باشد. شایان ذکر است که در عملیات denoise کردن تصویر برای اطمینان بیشتر از نوع نویز، از انواع فیلترها استفاده شد که به شرح زیر هستند:

Category	Filter Name	Function
Smoothing Filters	Box Filter	denoise_box_filter
	Gaussian Filter	denoise_gaussian_filter
Statistical Filters	Median Filter	denoise_median_filter
	Max Filter	denoise_max_filter
	Min Filter	denoise_min_filter
Advanced Filters	Bilateral Filter	denoise_bilateral_filter
	Non-Local Means	denoise_nl_means
	Wavelet Filter	denoise_wavelet_filter
	Total Variation Filter	denoise_total_variation

جدول ۱: انواع فیلتر ها و توابع denoising به کار گرفته شده

همانطور که از شکل ۱ انتظار می رفت، بهترین عملکرد خروجی با استفاده از فیلتر میانه یا همان median بدست می آید. در نهایت، برای نهایی کردن بهترین خروجی با استفاده از فیلتر میانه، دو روش پیگیری شد:

۱. استفاده از چند مرحله فیلتر median با ابعاد یکسان

۲. استفاده از یک فیلتر median با کرنل بزرگتر

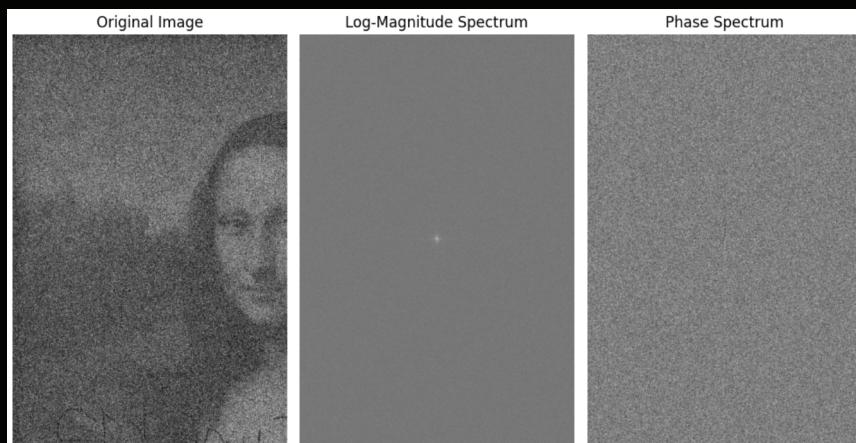
روش اول، به طور کلی برای حفظ جزئیات تصویر مناسب تر است؛ زیرا هرگونه کرنل بزرگ با ریسک از دست دادن جزئیات همراه است. از طرفی، ممکن است که همگرا شدن تصویر پس از اعمال چند بار فیلتر میانه، به کندی پیش روی و حتی برخی نویز ها در تصویر باقی بمانند که این مشکل، در روش دوم وجود ندارد.

در نهایت، با توجه به مسئله که تنها نیاز است یک متن از تصویر استخراج شود (و حفظ سایر جزئیات دارای اهمیت کمتری است)، از روش دوم استفاده شد.



شکل ۳: شکل نهایی تصویر denoise شده

همچنین شایان ذکر است که برای درک بهتر نویز، از حوزه‌ی فرکانس نیز کمک گرفته شد^[1] که بتوان نوع نویز را حدس زد و در نهایت با استفاده از فیلترهای notch و band-pass یا band-reject از نویز تصویر کاست. شکل سوم بیانگر این موضوع است.



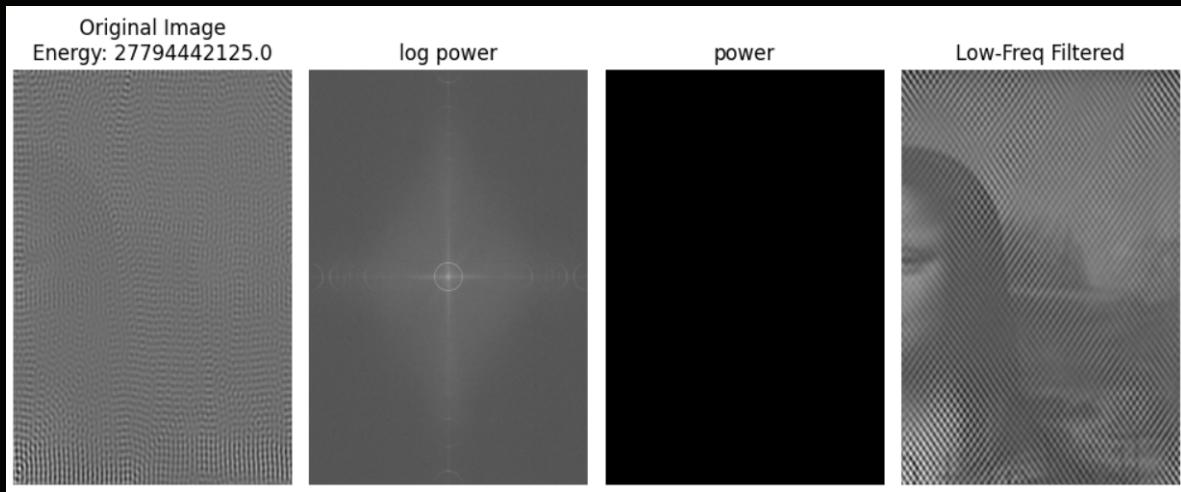
شکل ۴: تصویر شماره ۱ در حوزه فرکانس

با توجه به magnitude تصویر در حوزه فرکانس، نمی‌توان نویز خاصی را قائل شد که نهایتاً، همان فیلتر میانه که از هیستوگرام تصویر درک شده بود، استفاده شد.

^[1] Digital Image Processing By Gonzalez 2nd Edition 2002, chapter 4.2

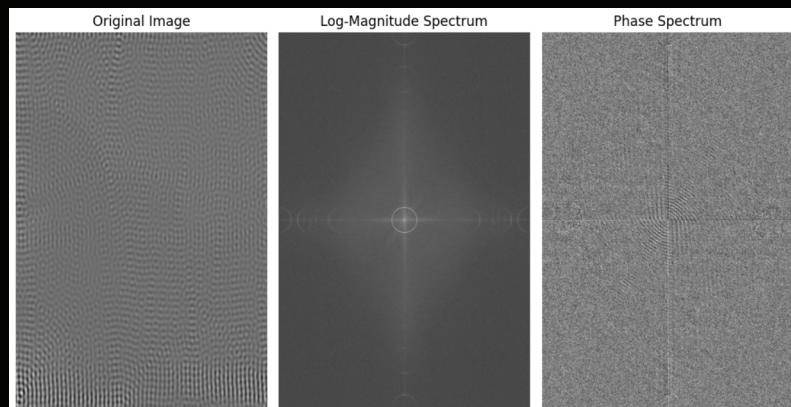
۰ بخش دوم

برای این بخش باید تصویر *processed_img_part_2.jpg* اصلاح می شد. بدین منظور و در ابتدا برای تشخیص نویز، از انرژی و *magnitude* تصویر استفاده شد؛ زیرا تصویر کاملا پوشیده از نویز بود و پیش بینی نوع نویز را سخت می کرد. همچنین برای درک بهتر کلیت تصویر، با حذف فرکانس های بالا، یک تخمین از کلیت تصویر به دست آمده است. برای این منظور، پلات زیر رسم شده است:



شکل ۵: پیش بینی نوع نویز در تصویر دوم

با دقت کردن به لگاریتم *power* در پلات (شکل شماره ۵) متوجه یک حلقه می شویم که با الگوی سینوسی نویز در تصویر در حوزه مکان تطابق دارد. برای درک بهتر این نویز، به طور مستقل برای تصویر در حوزه فرکانس *magnitude* و *phase* رسم شده است که به شکل زیر است:

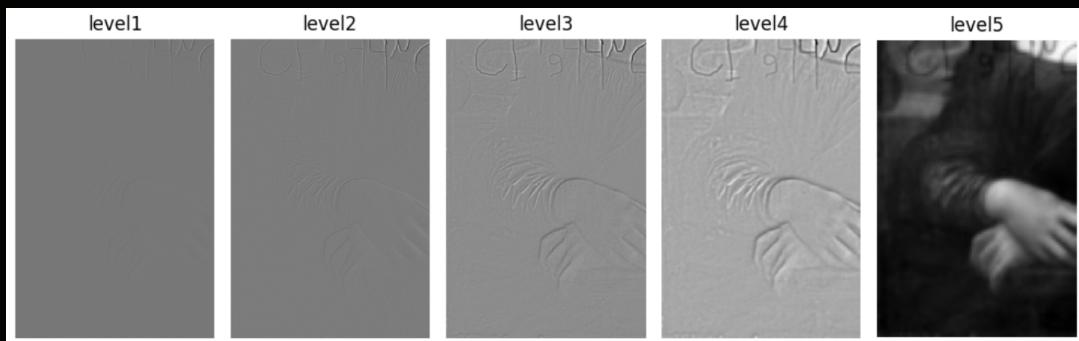


شکل ۶: تصویر دوم در حوزه فرکانس

که ایده اولیه را تایید می کند. بدین منظور، تنها و به سادگی با ساختن یک فیلتر *band-reject* در حوزه فرکانس، این مشکل حل شده و تصویر نهایی ساخته می شود.

۰ بخش سوم

برای بخش سوم از پازل، باید تصویر اصلی را از ۵ زیر تصویر reconstruct می کردیم. بدین منظور، تصاویر در ابتدا و در کنار یکدیگر plot شده اند:



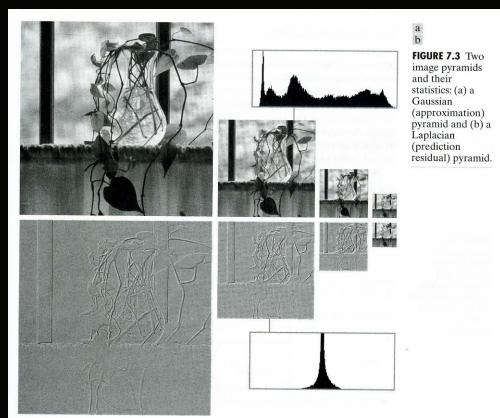
شکل ۷: زیرتصاویر بخش سوم پازل

همچنین ابعاد هر کدام از تصاویر در جدول زیر بیان شده است:

Image File	Width (px)	Height (px)
processed_img_part_3_Level_0.jpg	1796	1201
processed_img_part_3_Level_1.jpg	898	601
processed_img_part_3_Level_2.jpg	449	301
processed_img_part_3_Level_3.jpg	225	151
processed_img_part_3_Level_4.jpg	113	76

جدول ۲: ابعاد زیر تصاویر در بخش سوم پازل

با توجه به جدول شماره ۲ که بیانگر ابعاد پازل است، میتوان متوجه شد که در هر مرحله با تقریب هم طول و هم عرض تصویر نصف می شود. این موضوع، وجود نوعی pyramid را در ذهن تداعی می کند^[۱]. این موضوع، در زیربخش سوم فصل هفتم کتاب گونزالس^[۲] به این شکل آمده است:

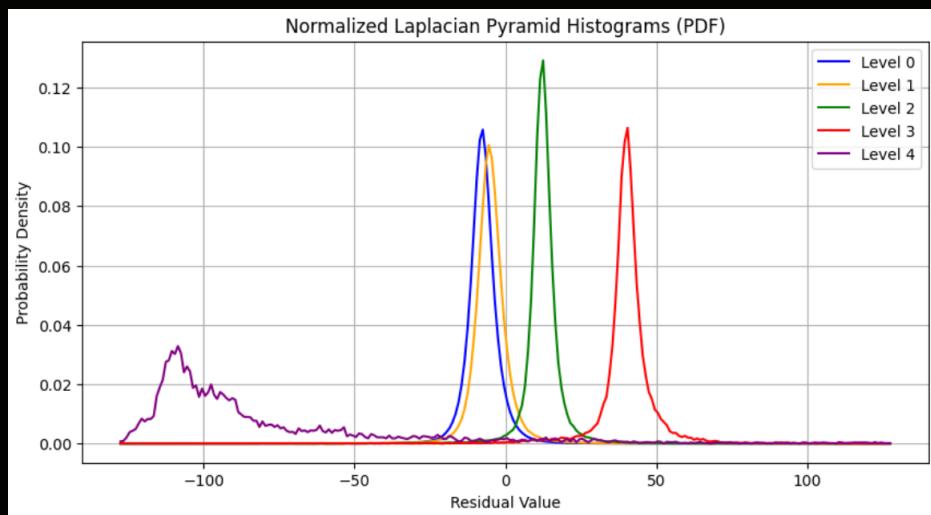


شکل ۸: انواع pyramid ها: کتاب گونزالس، زیربخش سوم فصل هفتم

[۱] medium: A Beginners Guide to Computer Vision (Part 4)- Pyramid

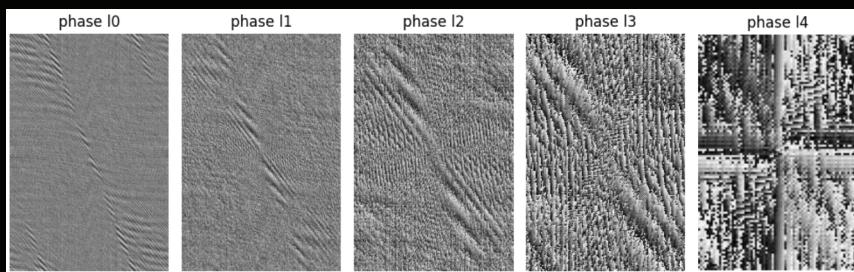
[۲] Digital Image Processing By Gonzalez 2nd Edition 2002, chapter 7.3

برای درک بهتر هرم، هیستوگرام احتمال برای هر کدام از زیر تصاویر رسم شده است:



شکل ۹: هیستوگرام در هرم

نمودار هیستوگرام نرمال شده هرم، توزیع آماری مقدارهای باقیمانده در هر سطح از هرم را نمایش می‌دهد. در فرآیند ساخت هرم، هر تصویر در یک سطح با نسخه‌ی بزرگ نمایی شده‌ی تصویر سطح بعدی کم می‌شود، و نتیجه این اختلاف، تصویر باقیمانده‌ای است که تنها حاوی اطلاعات لبه‌ها، جزئیات کوچک است. برای نمایش بهتر این اطلاعات، از یک آفست استفاده شده و همچنین نرمال سازی انجام شده است. در تایید این موضوع، از تبدیل به حوزه فوریه و نمایش فاز هر کدام از زیر تصاویر نیز استفاده شده است:



شکل ۱۰: هیستوگرام در هرم

همانطور که مشخص است، لبه‌های کوچک در سطوح بالا، دارای فاز یکنواخت تری هستند و هرچه به لایه‌های پایین‌تر می‌رویم، این یکنواختی کمتر می‌شود و تغییرات فرکانس وضوح می‌یابند. با دقیق‌تر بررسی این هیستوگرام، (و همچنین زیرتصاویر ارائه شده در شکل پنجم) می‌توان نتیجه گرفت که هرم از نوع لاپلاسی است؛ هر مرحله از *downsample* شدن تصویر و محاسبه اختلاف با لایه قبلی محاسبه می‌شود که برای ساختن تصویر اصلی، کافیست معکوس این کار را انجام دهیم.

برای درست کردن تصویر، در ابتدا تلاش شد که با طی کردن فرایند به شکل معکوس، تصویر اصلی ساخته شود، اما تصویر خروجی شامل کیفیت مناسب نبود. بدین منظور، از تابع `resize` از کتابخانه `cv2` استفاده شد که به صورت خطی تصویر را تغییر سایز می‌دهد. همچنین، برای بهبود نهایی، مقادیر `grayscale` به `uint8` تغییر یافته اند؛ به عبارتی، مقادیر قبلی سطوح خاکستری با این کار، به بازه (۰، ۲۵۵) گسترش یافتند که باعث بهبود کنترast شد. تصویر نهایی ساخته شده نیز در شکل ۹ آمده شده است:



شکل ۱۱: تصویر نهایی ساخته شده از زیرتصاویر

(۲) پروژه دوم: تصاویرنویزی/غیرنویزی

در این پروژه، هر دو روش یادگیری عمیق و بینایی ماشین کلاسیک پیاده سازی شده است. در ابتدا به تفصیل، روش یادگیری عمیق توضیح داده می شود:

(۳) یادگیری عمیق در پروژه دوم

در ابتداء، برای پیاده سازی سیستم تشخیص نویزها، این نیاز دیده می شد که دیتاست های در اختیار قرار گرفته، دیتاست های کوچکی هستند و برای روش های ml و یا dl مناسب نیستند. به گونه ای که حتی با augment کردن داده ها و پیاده سازی بر روی مدل های pre-trained و با tune fine آنها نیز، کفایت نمی کرد. بنابراین، با بررسی دیتاست های قرار داده شده کشاورزی که از نوع برگ درختان هستند، دیتاست های کاندیدا از وبسایت kaggle به شرح زیر هستند:

1. <https://www.kaggle.com/datasets/ichhadhari/leaf-images>
2. <https://www.kaggle.com/datasets/mdwaquarazam/agricultural-crops-image-classification/data>
3. <https://www.kaggle.com/datasets/alexo98/leaf-detection>

که طبق بررسی انجام شده، بهترین و مشابه ترین دیتاست، مربوط به دیتاست leaf-detection است. دقت شود که این موضوع که دیتاست ها باید برای استفاده این پروژه به شکل کاملی شخصی سازی شوند (به عنوان مثال، نویز به تصاویر اضافه شود) کاملاً واضح است و از ابتداء صرفاً به دنبال دیتاست هایی صرفاً با زمینه مرتبط بودم و سپس دیتاست ها را به شکل مناسبی برای استفاده از این پروژه شخصی سازی کردم. مجدداً تاکید می شود که دیتاست انتخاب شده، صرفاً انواع برگ های گوناگون بوده است و هیچ هدفی از نویز/دینویز کردن تصاویر در این دیتاست دنبال نشده است.

بررسی دیتاست آورده شده:

با توجه به این که این دیتاست، صرفاً شامل تصاویری از برگ های ۱۰ نوع درخت هندی هستند، بدون توجه به لیبل های آن برگ ها، و صرفاً با در نظر گرفتن همه آنها از یک نوع، (همگی را به شکل یکسان و «برگ» در نظر گرفتم) به آنها انواع گوناگونی از نویز ها پیاده سازی کردم. در ابتداء، نویزهای تکی، یعنی:

سپس انواع نویز های دو تایی (تمامی جایگشت ها لحاظ نشده است؛ به علت محدودیت پردازشی گوگل کولب به ۱۶ گیگ حافظه رم، بیش از این نوع نمیتوان جایگشت های مختلفی از نویز در نظر گرفت):

No.	Combinations	Noise Type
1	1	Gaussian
2	1	Salt & Pepper
3	1	Poisson
4	1	Speckle
5	1	Uniform
6	2	Gaussian + Salt & Pepper
7	2	Gaussian + Poisson
8	2	Gaussian + Speckle
9	2	Gaussian + Uniform
10	2	Salt & Pepper + Speckle
11	2	Salt & Pepper + Uniform
12	2	Poisson + Speckle
13	2	Poisson + Uniform
14	2	Speckle + Uniform
15	3	Gaussian + Salt & Pepper + Speckle
16	3	Gaussian + Poisson + Uniform
17	3	Salt & Pepper + Speckle + Uniform

جدول ۳: لیست انواع نویز های پیاده شده

در ابتدا، به کمک کتابخانه *cv2* تابع زیر را برای دریافت فیچرها تکمیل کردیم.

```
1 def extract_features(image_path)
2     # input is an image and the ouput is an array of features.
3
```

Listing 1: extract feature function



شکل ۱۴: new caption

شکل ۱۳: new caption

شکل ۱۲: caption another

منابع

1. Image Denoising Algorithms: A Comparative Study of Different Filtration Approaches Used in Image Restoration. <https://ieeexplore.ieee.org/abstract/document/6524379/>
2. Digital Image Processing By Gonzalez 4th <https://elibrary.pearson.de/book/99.150005/9781292223070>
3. Automatic identification of noise in ice images using statistical features https://www.researchgate.net/figure/Simple-pattern-classifier-to-identify-noise-types-of-Gauss-tbl1_258714501
4. medium: A Beginners Guide to Computer Vision (Part 4)- Pyramid <https://medium.com/analytics-vidhya/a-beginners-guide-to-computer-vision-part-4-pyramid-3640edeffb00>
5. medium: A Beginners Guide to Computer Vision (Part 4)- Pyramid <https://medium.com/analytics-vidhya/a-beginners-guide-to-computer-vision-part-4-pyramid-3640edeffb00>
6. medium: A Beginners Guide to Computer Vision (Part 4)- Pyramid <https://medium.com/analytics-vidhya/a-beginners-guide-to-computer-vision-part-4-pyramid-3640edeffb00>
7. medium: A Beginners Guide to Computer Vision (Part 4)- Pyramid <https://medium.com/analytics-vidhya/a-beginners-guide-to-computer-vision-part-4-pyramid-3640edeffb00>