دانشکده مهندسی کامپیوتر

هلیا وفایی – 99522347

نیم سال 1401-1402

گزارش پروژه اول

درس انتقال داده

فهرست مطالب

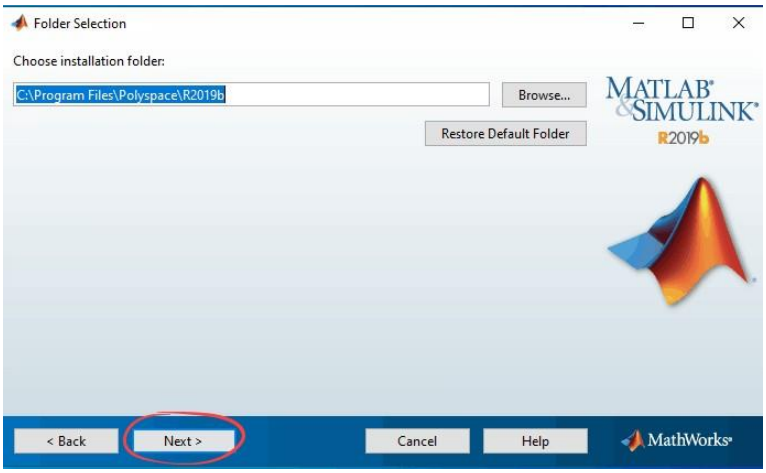
1 هدف پروژه

هدف از انجام این پروژه، بکارگیری مفاهیم آموزش داده شده در مبحث سیگنال‌ها در پردازش تصویر بود. به طوری که در این پروژه باید ابتدا با ابزار وارد کردن و خواندن یک فایل تصویری در نرم‌افزار MATLAB آشنا می‌شدیم. سپس با استفاده از ابزارها و دستورهای مناسب، در تصویر مربوطه نویز ایجاد کرده و درمرحله آخر نیز این نویز را از بین ببریم.

این پروژه درعمل شامل ۸ گام مختلف بود که در هر گام به پیاده‌سازی موارد خواسته‌شده پرداخته شد که در این گزارش به تفصیل کارهای انجام‌شده در هر گام خواهیم پرداخت.

2 گام اول

در گام اول، فارغ از نوع سیستم عاملی که در اختیاز داریم، باید به نصب نرم‌افزار MATLAB می‌پرداختیم.



شکل 1: نصب نرم‌افزار

3 گام دوم

در این گام، یک تصویر رنگی از کامپیوتر خود با دستور imread(path) می خوانیم .

شکل 2: خواندن فایل تصویری از حافظه کامپیوتر

4 گام سوم

در این گام اقدام به تبدیل تصویر رنگی از فرمت RGB به Gray-Scale 8 bit می نماییم. این کار را با استفاده از دستور rgb2gray(image) انجام می دهیم.

شکل 3: تبدیل فرمت از RGB به Gray-scale 8 bit

تکته 1: در این گام، در توضیح انواع تصاویر و تفاوت‌های آنها می‌توان گفت بطور کلی 3 نوع تصویر داریم:

- تصاویر binary : این تصاویر ساده ترین نوع تصاویر هستند که هریک از پیکسل‌های آن فقط 2 مقدار سیاه یا سفید را می‌گیرند. این تصاویر با استفاده از threshold تولید می‌شوند و در واقع هر پیکسلی که بالاتر از threshold باشد مقدار سفید (1) و در غیر این صورت مقدار سیاه (0) را می گیرد.

- تصاویر Gray-scale: این تصاویر تک رنگ هستند و فقط یک رنگ خاکستری دارند که درواقع هر پیکسل سطوح مختلف رنگ خاکستری موجود را تعیین می‌کند. یک تصویر Grayscale معمولی دارای مقیاس 8 bit/pixel است که درواقع دارای 652 سطح خاکستری مختلف می‌باشد. البته در تصاویر پزشکی و نجوم از مقیاس‌های بالاتر ازجمله 21 یا 61 bit/pixel استفاده می‌کنند.

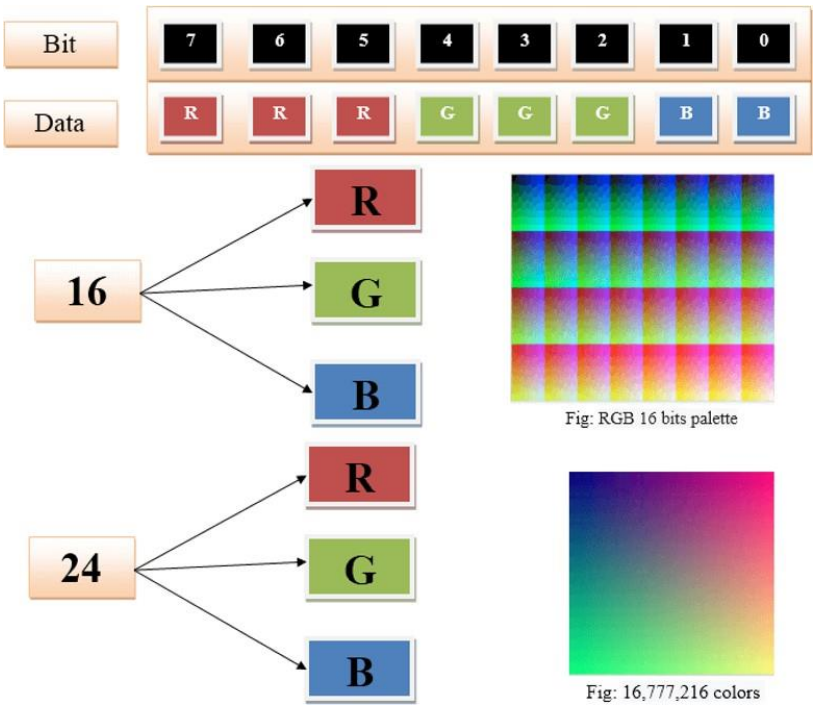
- تصاویر رنگی: این تصاویر، تصاویر تک رنگ سه باندی هستند که هر باند دارای رنگ متفاوتی می باشد و اطلاعات واقعی در تصویر دیجیتال ذخیره می شود. این تصایر با مقیاس 24 bit/pixel یعنی 8 بیت برای هر یک از 3 باند RGB نمایش داده می شود.

همچنین تصاویر رنگی از نظر تعداد بیت نیز انواع مختلف دارند ازجمله:

- رنگ 8 بیتی: برای ذخیره اطلاعات تصویر در حافظه کامپیوتر یا در فایل یک تصویر استفاده می شود. در این فرمت، هر پیکسل یک بایت 8 بیتی را نشان می دهد. دارای طیف رنگی 552-0 است که در آن از 0 برای مشکی، 552 برای رنگ سفید و 721 برای رنگ خاکستری استفاده می شود.

- رنگ 61 بیتی: این فرمت نیز به عنوان فرمت رنگ high شناخته می شود. این فرمت دارای 63556 سایه رنگ مختلف است و در سیستم‌های توسعه یافته ی مایکروسافت استفاده می شود. فرمت رنگی 61 بیتی نیز به 3 فرمت قرمز، سبز و آبی تقسیم می شود که به فرمت RGB نیز معروف است.

- رنگ 42 بیتی: این فرمت به عنوان فرمت رنگ واقعی نیز شناخته می شود. فرمت رنگی 42 بیتی نیز در قرمز، سبز و آبی توزیع شده است. همان طور که 42 را می توان به طورمساوی بر روی 8 تقسیم کرد، بنابراین به طور مساوی بین 3 رنگ مختلف مانند 8 بیت برای R 8 بیت برای G و 8 بیت برای B توزیع می شود.



شکل 4: انواع فرمت تصویر های رنگی ازنظر تعداد بیت

5 گام چهارم

در این گام، تصویر Grayscale به دست آمده از گام قبل را با استفاده از دستورات imshow نمایش‌می‌دهیم و با دستور imwrite در کامپیوتر خود ذخیره می‌کنیم.

نکته 2: برای ذخیره ی یک تصویر به صورت کامل و بدون هیچ گونه فشرده سازی با اتلاف در کامپیوتر خود از فرمت png استفاده می کنیم.

6 گام پنجم

تصویر rgb ذخیره شده با دستور imread خوانده و به یک تصویر خاکستری تبدیل می کنیم.

هیچ توانی در تصویر وجود ندارد و سیگنال تصویر از نوع انرژی است. برای محاسبه مقدار انرژی موجود در تصویر، باید تمامی سطوح خاکستری تصویر (مقادیرذخیره شده برای هر پیکسل که همگی اعدادی بین 0-552 هستند) را با یکدیگر جمع کنیم که این کار را با استفاده از دستور sum روی آرایه‌ی پیکسل های تصویر انجام می دهیم که مقدار آن برابر با 348004888 به دست می آید.

7 گام ششم

در این گام، سعی می کنیم با استفاده از دستور imnoise(image, method) به تصویر خاکستری، یک نویز گاوسی (gaussian) با میانگین صفر و واریانس 0.03 اضافه کنیم.

شکل 5: تصویر ایجادشده پس از اعمال نویز گاوسی

نکته 3: درمورد مفهوم SNR (Signal Noise Ratio) می‌توان گفت SNR یا نسبت سیگنال به نویز، معیاری است که سطح سیگنال موردنظر را با سطح سیگنال پس‌زمینه مقایسه می‌کند که از حاصل تقسیم توان سیگنال اولیه به توان سیگنال ایجاد شده پس از اعمال نویز به دست می‌آید که واحد آن دسی‌بل است.

$$\mathrm{SNR} = \frac{\mathrm{P}signal}{\mathrm{P}noise}$$

از آنجا که بسیاری از سیگنال‌ها دامنه دینامیکی بسیار وسیعی دارند، سیگنال‌ها اغلب با استفاده ازمقیاس دسی‌بل لگاریتمی بیان می‌شوند. یعنی داریم:

$$\mathrm{P}\mathrm{signal},\mathrm{db} = 10\log\frac{\mathrm{P}signal}{10}$$

$$\mathrm{P}nose,db = 10\log\frac{\mathrm{P}nose}{10}$$

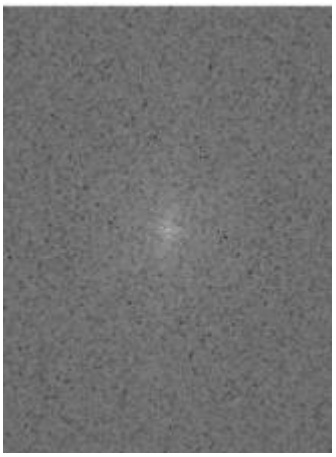
پس می توانیم بگوییم:

$$\mathrm{SNR}db = \mathrm{P}signal,db - \mathrm{P}noise,db$$

این کار را در متلب با استفاده از یکی از توابع آماده‌ی snr(x, y) انجام می دهیم که ورودی اول این تابع، سیگنال reference و ورودی دوم تابع مقدار نویز (اختلاف سیگنال اصلی و نویزدار) می‌باشد. برای هر دو حالت (قبل و بعد از اضافه کردن نویز) مقدار SNR را با توجه به توضیحات داده‌شده محاسبه می کنیم. این مقدار برای حالت قبل از اضافه کردن نویز مقدار بی‌نهایت (چون نویز نداریم و مخرج کسر صفر می‌شود.) و برای بعد از اضافه کردن نویز، برابر با مقدار 348004888 به دست آمد که برابر با انرژی به دست آمده در گام قبل است.

8 گام هفتم

در گام هفتم، تصویر خاکستری نویزدار را به حوزه فرکانس می‌بریم و آن را نمایش می‌دهیم. برای تبدیل به حوزه فرکانس از دستور (*fftshift*(*log*(*abs*(*fft2*(*gaulmg*)))) استفاده می کنیم و با استفاده از دستور imshow آن را نمایش می دهیم.



شکل 6: تصویر دارای نویز در حوزه فرکانس

نکته 4 : در توضیح در رابطه با مفهوم TTF(Fast Fourier Transform) می‌توان گفت هدف آن، تبدیل و جابجایی عکس بین حوزه spatial و frequency است. در حوزه تصویرهای دیجیتال با DFT(Discrete Fourier Transform) سروکار داریم که برای یک تصویر دوبعدی در ابعاد N*N داریم:

$$\mathrm{F(k,l)} = \sum_{\mathrm{m=0}}^{\mathrm{N-1}} \sum_{n=0}^{N-1} f(m,n) e^{-2i\pi(\frac{km}{N}+\frac{nl}{N})}$$

درواقع در FFT ما تابع را بصورت مجموع توابع سینوسی و کسینوسی می نویسیم. هنگامی که از FFT استفاده می کنیم برای اینکه یک عکس را به حوزه فرکانس ببریم، کمترین فرکانس‌ها با peak های بزرگ در وسط داده‌ها نمایش داده می‌شوند و بیشترین فرکانس‌ها در اطراف هستند. در تصویر به وجود آمده نیز

همین موضوع برقرار است. وسط عکس، فرکانس کمتر است و دور عکس فرکانس بالاتر است و به همین

دلیل و نویزهایی که به تصویر اضافه شده، باعث شده است تا تعداد نقاط سفید در وسط عکس بیشتر باشد.

9 گام هشتم

در این گام، با جستجو در اینترنت انجام توانستم دو روش خوب برای رفع نویز تصاویر پیدا کنم وهر 2 روش (که درواقع جزو دستورات متلب بودند) را روی تصویر نویزدار اعمال کردم که نویز تصویررا تا حد خوبی برطرف نمود:

- روش Averaging filter: استفاده از دستور imfilter :

شکل 7: رفع نویز با استفاده از Average filter

- روش Median filter: استفاده از دستور medfilt2 :

شکل 8: رفع نویز با استفاده از Median filter

نکته 5 : برای اینکه ببینیم در زمینه ی رفع نویز چقدر خوب عمل کردیم، نیاز به یک معیار کمی داریم که معیار PSNR (Peak Signal-to-Noise Ratio) می‌تواند در این قسمت ما را یاری کند. این معیار یک اصطلاح مهندسی برای نسبت بین حداکثر توان ممکن یک سیگنال و قدرت نویز مخرب است که بر صحت نمایش آن تأثیر می گذارد. از آنجایی که بسیاری از سیگنال‌ها دارای محدوده دینامیکی بسیار وسیعی هستند، PSNR معمولا به صورت یک کمیت لگاریتمی با استفاده از مقیاس دسی بل بیان می‌شود. هر چه PSNR بالاتر باشد، کیفیت تصویر فشرده یا بازسازی شده بهتر است.

برای محاسبه‌ی PSNR ابتدا نیاز به محاسبه‌ی مقدار خطای میانگین مربع (MSE) داریم که از رابطه ی زیر به دست می‌آید:

$$MSE = \frac{1}{\mathrm{mn}} \sum_{\mathrm{i}=0}^{\mathrm{m}-1} \sum_{j=0}^{n-1} (I(j,k) - K(i,j))^2$$

با استفاده از MSE نیز می‌توان رابطه ی PSNR را به صورت زیر نوشت:

$$PSNR = 10 \log_{10} \frac{max^2}{MSE}$$

برای محاسبه‌ی PSNR در متلب، از یکی از توابع آماده‌ی متلب به نام psnr(noise, ref) استفاده می کنیم که مقدار این معیار برای average filter برابر با 14.7250 و برای Median filter برابر با 15.3544 به دست می آید که نشان از دقت بالاتر روش دوم دارد.