EE604: Assignment 3

Prof. Tushar Sandhan
sandhan@iitk.ac.in

Semester-I, 2022

Due date: 7^{th} November, 2022 Weight: 25%
Due time: 07:07AM Submission: MookIT

Introduction

Here you will analyze research papers, image processing software and design creative questions.

References

- [1] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images", IEEE International conference on computer vision (ICCV), 1998.
- [2] Y. Yao, H. Zhou and M. Erol-Kantarci, "Joint Sensing and Communications for Deep Reinforcement Learning-based Beam Management in 6G", IEEE Globecom, 2022.
- [3] J. Kopf, M. F. Cohen, D. Lischinski and M. Uyttendaele, "Joint Bilateral Upsampling", ACM Transactions on Graphics, 2007.

1 Stubble Burning: `denoiser` [5%]

Stubble burning has been one of the prime reasons for air pollution in northern Indian regions. Light-weight burnt flakes get dispersed in the environment and remain floated for months. Surrounding images captured during this time, include peculiar diffused speckled noise as shown in Fig. 1. Denoise the provided 2 noisy images, one is of smaller size and another is of average size. We will use any one of these images for evaluation. You can refer [1] or the topics covered in the class. Write a python program `denoiser.py`, which accepts command line file path for input image and directly stores the output `denoised.jpg` file at current location where the command {`$python denoiser.py ./noisy1.jpg`} is being executed.

Figure 1: Input `noisy1.jpg`

2 6G Network: `localizer` [6%]

User location is a crucial piece of information for 6G network management and control. In real world, location uncertainty in mmWave networks is unavoidable and leads to localization errors. Image processing aided 6G systems show much improved performance by reducing location uncertainty. Refer [2] for designing computationally light-weight fast localizer of 3 geographical locations: building=1, grass=2 and road=3, as shown in Fig.2. Different input image given to your algorithm (e.g. `location2.jpg`), will entirely contain only one of these 3 locations. Write a python program `localizer.py`, which accepts command line file path for input image and directly outputs the location number (either 1, 2, or 3) after the command `{$python localizer.py ./location2.jpg}` is executed.

Figure 2: Input `location.jpg`

3 Image Colorization for Improved JPEG: `colorizer` [7%]

'Imbalance and rush' create distressed black & white life; but purity with '**b**alance and **r**est', make stress-free colorful life. Similarly chroma with C**b** and C**r** components of the YCbCr color transform make image colorful. However in JPEG compression, both Cb and Cr are decimated greatly i.e. their pixel area is $\frac{1}{4}$ that of Y. That means original color information is lost.

Figure 3: Y of `flyingelephant.jpg`

In extreme JPEG compression, Cb and Cr can be decimated by the factor of 4 or 8 i.e. their pixel area is $\frac{1}{16}$ and $\frac{1}{64}$ that of Y respectively. Thus one can consider, almost all color information is lost and estimating that back is almost the problem of image colorization from gray image (i.e. Y channel). For transmission over low bandwidth network, the colorful image `flyingelephant.jpg` is extremely decimated in both Cb and Cr channels, whereas original Y channel is preserved, as shown in Fig. 3. Thus inputs available are Y channel and extremely decimated (very small size) Cb and Cr channels, from which the original colorful image `flyingelephant.jpg` need to be estimated.

Refer [3] for resurrecting Cb and Cr channels. Once bilateral upsampling of Cb and Cr is performed independently, we can combine them to Y channel to obtain underlying image in YCbCr space. Convert it to RGB space to produce colorful `flyingelephant.jpg`.

Write a python program `colorizer.py`, which accepts command line file path for all 3 input images (viz. Y channel original shape image, 4 times decimated Cb and Cr channel images) and directly stores the colorful output image `flyingelephant.jpg` file at current location where the command `{$python colorizer.py ./Y.jpg ./Cb4.jpg ./Cr4.jpg}` is being executed.

4 Software and Questions for a Question: swq4q [7%]

You have to explore any image processing (IP) software (SW) for analyzing some of its functionalities. For this question, you will also be designing analytical and objective questions & answers. A3Q4 should be submitted as a pdf file `swq4q.pdf`, and its total weight is distributed as below,

- `sw [3%]`: For IP in scientific domain (microscopy, remote sensing), professional photography and handheld devices, a plethora of SW is available (open source, paid, trial versions, restricted, etc.). Choose any useful SW, list its URL and include screenshots demonstrating its operation. Explore one of its intriguing image processing functions for producing output. Explain this function briefly and include input-output image results in the report.
- `q4q`: Design below questions related to IP. Describe the questions in detail and answer them. You may refer to any resources or design on your own and it may be related to any IP topic (covered or not-covered in the class).
  - `q4q-analytical [2%]`: One analytical question (something similar to midterm sec.3)
  - `q4q-mcq [2%]`: Two MCQ questions (something similar to midterm sec.2)

Submission

For each question use the original images that are provided to you separately apart from this document. The only purpose of the figures here is to help with the proper explanations.

If your program does not produce output and throws errors, then you will receive a score of 0 for that question. You alone are accountable for the submitted program's proper operation. Kindly make sure it executes and outputs just the things which are being asked for.

Depending on the output quality, partial to full credit will be awarded if the program executes correctly. For plagiarized responses, even if your program does not execute or unexpectedly produces correct answer, in all cases you will be awarded full marks with an honorarium prefix as a negative sign.

Compress only the files listed below into `rollno_A3.zip` (eg.`191234_A3.zip`) and upload it.

- `denoiser.py`
- `localizer.py`
- `colorizer.py`
- `swq4q.pdf`

■