Laboration 3

Bildreproduktion och Bildkvalitet
(TNM097)

Bildkvalitet

Objective of the Lab

The purpose of this lab is to develop a deeper understanding of several important image quality metrics by applying them to evaluate a set of images.

After completing this lab, you should have a clearer understanding of the following concepts:

- Color gamut
- Mathematical quality metrics, such as MSE, SNR, and SSIM
- Human Visual System (HVS) models and why they are needed
- S-CIELab quality metric
- Correlation between objective quality metrics and subjective quality assessments (DMOS — Differential Mean Opinion Score)
- Training objective metrics based on subjective DMOS data

Introduction:

In this lab, you will evaluate several image quality metrics using different grayscale and color images. You will examine how well these metrics correlate with your own visual judgment of image quality, compare them with the DMOS values, and perform training of objective metrics based on DMOS data.
All metrics covered in this lab have been discussed in Lecture 3 and are described in the corresponding lecture notes and literature available on Lisam.
All necessary images and files are provided in the archives *Lab3.zip* and *scielab.zip*, available under Course Documents → Labs → Lab3 on Lisam.
Remember to scale all images to the interval $[0, 1]$ after loading them into MATLAB — for example, by using the function *im2double*.

1. Color Gamut:

Here you are supposed to plot the color gamut for two different devices in the same plot. You can use the function *plot_chrom(XYZ, col)*, which takes the CIEXYZ data for the device primaries and a string *col* to specify the color of the graph. Notice that in this function it is assumed that XYZ is a $3 \times T$ matrix, where T is the number of the primaries of the device. Therefore, each column contains the CIEXYZ values for a primary.

Device 1: A Dell computer screen. The CIEXYZ data of the RGB-primaries for this screen is saved in *Dell.mat*.

Device 2: An Inkjet printer (Canon IPF 6450) using seven primary inks, C, M, Y, K, R, G and B. The CIEXYZ data of the primaries for this printer is saved in *Inkjet.mat*. In order to easily plot this the data in the chromaticity diagram, the CIEXYZ for black is not included in this data.

Discuss the differences between these two devices and try to figure out which device is better in reproducing a specific color, for example red.
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….

2. Mathematical metrics (MSE/SNR)

Start by reading a grayscale image (for example *peppers_gray.tif*) and test one of the metrics *MSE* or *SNR* for two different “distortions”. You can either use Matlab’s pre-defined function *snr* or the function *mysnr* provided to you, **just note that in both of them the first argument is**

the original signal/image and the second argument is the noise (**noise:** the difference between the original and the reproduction).

2.1 Interpolation

Use the function *imresize* to down-sample and then up-sample the image by 0.25 and 4, respectively. Use three different interpolations, "*nearest*", "*bilinear*" and "*bicubic*".

Hint: *imresize(imresize(p,0.25,'nearest'),4,'nearest')*; does the intended operation using nearest neighbor interpolation.

Look at these three images and apply MSE or SNR to them to find how similar they are to the original image. Does the result of these metrics correlate with your judgment of the quality?

…………………………………………………………………………………………………….
…………………………………………………………………………………………………….
…………………………………………………………………………………………………….
…………………………………………………………………………………………………….
…………………………………………………………………………………………………….

2.2 Halftoning

Halftone the grayscale image by thresholding it with 0.5 and also error diffusion (use Matlab function *dither*).
Hint: $a >= 0.5$, thresholds image *a* with 0.5.
Hint: The resulting halftones will be of class *logical*, make them double in order to be able to use them in your calculations.

Examine the two halftone images and use either **MSE** or **SNR** to measure how similar they are to the original image. Do the results from these metrics align with your own visual assessment of image quality? According to these metrics, is there any halftone that appears more similar to the original image than the one thresholded at 0.5? Discuss why.

…………………………………………………………………………………………………….
…………………………………………………………………………………………………….
…………………………………………………………………………………………………….
…………………………………………………………………………………………………….
…………………………………………………………………………………………………….

3. Mathematical metrics involving HVS

You are given a function *snr_filter*, which is the SNR function modified by a model for HVS. Make sure that you understand how this function operates by reading the comments and also having a look at the other functions that are called inside this function.
Apply this metric to the two halftones in assignment 2.2. Does the result correlate better with your judgment of quality?

…………………………………………………………………………………………………….
…………………………………………………………………………………………………….
…………………………………………………………………………………………………….

4. S-CIELab

Among the files put at Lisam you have a file called *scielab.zip*, in which you have all necessary codes to run the function called *scielab*. For more information about the method, see: Zhang & Wandell, 1997, *A Spatial Extension of CIELAB for Digital Color-Image Reproduction*.

S-CIELab as a full-reference metric

The function *scielab* accepts several optional arguments. To use it as a full-reference metric in this lab, you need to provide five arguments:

1. *sampPerDeg* (samples per degree): Refer to the lecture notes to understand how this value relates to the device resolution and viewing distance.
2. Original image in CIEXYZ color space. You can convert your RGB images to CIEXYZ using the predefined MATLAB function *rgb2xyz*.
3. Reproduction image in CIEXYZ color space.
4. White point vector for the intended light source (CIEXYZ values). For example, for CIED65, use $[95.05, 100, 108.9]$. This is required for computing the CIELab values.
5. Image format, which in this lab is 'xyz'.

The output of the function is an image of the same size as the inputs, containing the S-CIELab difference between the two images. To obtain a single quality value, you can compute the average of the output image. Smaller values indicate smaller differences, i.e., higher similarity between the original and reproduction.

Now, read the color image *peppers_color.tif*. Down-sample and then up-sample the image (using imresize) by factors of 0.25 and 4, respectively, applying all three available interpolation methods. Compare the results using the S-CIELab metric. To make a fair assessment, ensure you know your screen resolution and specify a viewing distance.
Finally, examine the S-CIELab difference images and discuss whether the metric aligns with your visual judgment of image quality.

……………………………………………………………………………………………………….
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….

5. SSIM

In this section, you will test the SSIM (Structural Similarity) image quality metric, for which MATLAB provides a predefined function, *ssim*. Although the SSIM metric performs well in many cases, this exercise demonstrates a situation where SSIM does not behave as expected.
The *ssim* function takes two images as input — the original and the distorted (or reproduced) image — and returns two outputs: A single **SSIM value**, representing the overall structural similarity and An **SSIM map**, showing the local SSIM value for each pixel.
Start by reading the grayscale image *peppers_gray.tif*. To evaluate the behavior of SSIM, you will simulate two types of distortions and compare their results to the original image.

- **Distortion 1:** Add $+0.1$ to all odd rows (i.e., rows 1, 3, 5, …) and -0.1 to all even rows (i.e., rows 2, 4, 6, …) in the original image.
- **Distortion 2:** Add $+0.1$ to the upper half of the original image and subtract 0.1 from the lower half.

Both distortions will produce identical SNR values when compared to the original image — verify this and explain why.
Next, use the *ssim* function to compute how similar each distorted image is to the original. Also, inspect the **SSIM maps** generated for both distortions. What differences do you observe, and what do they tell you about the limitations of SSIM?

……………………………………………………………………………………………………….
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….
……………………………………………………………………………………………………….

6. Correlation between Objective and Subjective Metrics

In this section, you will use a prewritten function to study the correlation between the three objective image quality metrics tested in this lab — SNR, SSIM, and S-CIELab — and the subjective evaluation results DMOS (Difference Mean Opinion Score). **Before you start**, download and extract the zip file ***DMOS_Metric_Training.zip***, which contains all MATLAB code, saved data, and images. Note that all folders containing images must be located in the same directory as the MATLAB code files.
The subjective tests were conducted on 29 original images, each of which was degraded (e.g., blurred or distorted) to produce 779 corrupted images in total. The subjective evaluation was carried out by a group of human observers under controlled viewing conditions. Each participant rated the perceived quality of the distorted images compared to their original versions. The individual ratings were then averaged to obtain the values, which represent the collective human perception of image quality. A lower DMOS value indicates higher visual quality, meaning that the distortion is less noticeable.

(**Source:** LIVE Image Quality Assessment Database — Laboratory for Image and Video Engineering, University of Texas at Austin)

To make all objective and subjective metrics comparable, they have been normalized to the range $[0,1]$, where a higher value corresponds to better image quality. This normalization allows for a more direct and meaningful comparison between the metrics, enabling you to analyze how well the objective measures correlate with human visual perception.
Now run the function $Comparisons\_objective\_Dmos$. This function displays the correlation between the three objective quality metrics and DMOS using two types of correlation: Pearson (measuring linear relationship) and Spearman (measuring rank-order agreement). A high (positive) correlation means that the metric agrees well with subjective human ratings.
For more details on how the function works and the differences between these correlation types, see the comments within the script. Examine the results and consider which objective quality metric shows the strongest correlation with DMOS.
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………

To gain more insight into the results from DMOS and the objective quality metrics, display the images that are ranked as the worst according to each of the four metrics. For example, the index of the image ranked as worst by SNR can be found with $ind = find(SNR\_all == min(SNR\_all))$, and similarly for the other three metrics. Once you have the index *ind*, use it as input to the function $retrieve\_image$ to load and display the corresponding image. This function returns a structure containing the original image, the corrupted image, and the DMOS value corresponding to the given index. Please refer to the comments in the script for instructions on how to display these outputs.
Display all four images and provide your observations about these images.
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………

Now, display the images that are ranked as the best according to each of the four metrics. For each metric, the best image is the one with the maximum value. Show all four images and describe your observations about these images.
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………

7. Objective Metric Training Using DMOS

In this section, you will use the provided script *Comparisons_training.m* to train different prediction models based on the subjective DMOS data. The goal is to investigate how well different objective image quality metrics can predict the DMOS values.
In the script, you can:

- Choose the input metrics to use — **either one of SNR, SSIM, S-CIELab**, **any pair of them**, or **all three together**.
- Select a regression method (Linear, Polynomial (2nd degree), or KNN). You may have encountered these models in a previous course. Below is a brief description of each:

    - Linear Regression: Predicts a target variable as a linear combination of the input features. It assumes a straight-line relationship between inputs and output.
    - Polynomial Regression (2nd degree): Extends linear regression by including squared terms and interactions of input features. This allows the model to capture curved relationships between inputs and output.
    - K-Nearest Neighbors (KNN) Regression: A non-parametric model that predicts the target value based on the average of the $\boldsymbol{k}$ closest training samples in the feature space.

Please read the comments in the code for details on each step.

After running the script, try different combinations of:

- Regression methods
- Input metrics

and briefly describe what you observe:

- Which combination produced the best correlation with the DMOS values?
- How does the choice of model and input affect the prediction performance?

Note: It is not required to change the training rate, but if you want to explore how the results vary with different data splits, you can adjust the variable *train_rate* (default is 0.8, meaning 80% of the data is used for training).

………
………
………
………
………
………
………
………
………
………
………
………