Assignment 0 – Interpolation and Error Measures

Image Processing and Pattern Recognition

Deadline: October 25th, 2024

1 Goal

In this exercise you should become familiar with popular error measures for images. You should implement the peak signal to noise ratio (PSNR) and structural similarity (SSIM) index and use them to compare different interpolation strategies.

2 Error Measures

In order to quantify the quality of digital images with respect to some reference, we use *error measures*. An error measure assigns a scalar value that quantifies agreement between some *ground truth* solution and some degraded (or restored) image. In this assignment we only consider discrete images $X, Y \in \mathbb{R}^{M \times N}$.

2.1 Mean Squared Error

The mean squared error (MSE) is well known in most fields that relate to signal processing. Specifically for images $X, Y \in \mathbb{R}^{M \times N}$ it is the mean of the pixel-wise squared differences:

$$\mathrm{MSE}(X, Y) \coloneqq \frac{1}{MN} \|X - Y\|_F^2 = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (X_{i,j} - Y_{i,j})^2. \tag{1}$$

2.2 Peak Signal to Noise Ratio

The PSNR relates the energy of the highest possible signal value m to the energy of the noise (or error). Specifically, the PSNR "normalizes" the MSE by the maximal intensity and scales the result logarithmically:

$$\mathrm{PSNR}(X, Y) \coloneqq 10 \log_{10} \left(\frac{m^2}{\mathrm{MSE}(X, Y)} \right). \tag{2}$$

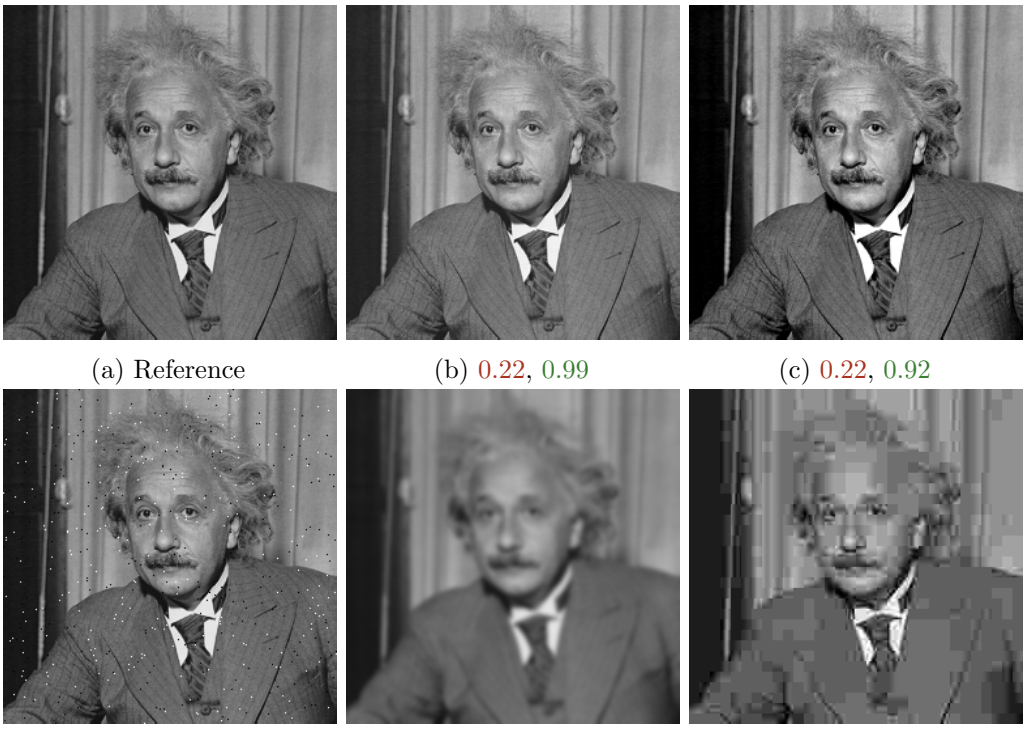
In contrast to the MSE, the PSNR increases as X approaches Y and at $X = Y$ we define $\mathrm{PSNR}(X, Y) = \infty$.

Note that, in image processing, the energy of the highest possible signal value is the "white image", and m depends on which digital representation is used. Typically, if the images are represented using floating point numbers, $m = 1$. On the other hand, if an integral data type is used, $m = 255$.

2.3 Structural Similarity Index

It is well known that images Y on the hypersphere $\{Y \colon \|X - Y\|_F^2 = r\}$ can have drastically different appearances to the human eye. In Fig. 1 we show images with different corruptions which have the same MSE. In the first two examples (shifted mean intensity and contrast stretching respectively), the MSE overestimates the "visual" error, whereas for the jpeg compressed image the MSE possibly underestimates the "visual" error.

The SSIM tries to combat this issue by defining error measures on three components: luminance, contrast, and structure. To ease notation, let $x, y \in \mathbb{R}^n$ be two flattened



(a) Reference (b) 0.22, 0.99 (c) 0.22, 0.92

(d) 0.22, 0.87 (e) 0.22, 0.83 (f) 0.22, 0.81

Figure 1: Images with the same MSE ($m = 1$, MSE scaled by 100) appear very different to the human eye, while the SSIM represent the human visual system more accurately.

image patches. The luminance of a patch x is the mean intensity

$$\mu_x = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{3}$$

and we define the luminance comparison function l as

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \tag{4}$$

where $C_1 = (K_1 m)^2$ with the constant $K_1 \ll 1$, which we fix as $K_1 = 0.01$. To estimate the contrast of the patches, we compute the sample standard deviation in the patch as

$$\sigma_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_x)^2}. \tag{5}$$

To compare two patches x, y, we define the contrast comparison function

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \tag{6}$$

where again $C_2 = (K_2 m)^2$ and we set $K_2 = 0.03$. To quantify structural similarity, we consider the normalized cross correlation between x and y. Specifically, the structure comparison is

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{7}$$

where $C_3 = \frac{C_2}{2}$ and

$$\sigma_{xy} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y). \tag{8}$$

Finally, we define $\mathrm{SSIM} = l \cdot c \cdot s$ such that

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}. \tag{9}$$

To use the SSIM for image quality assessment, typically it is desirable to apply it on local windows rather than globally, since image features (luminance, contrast, structure) can change drastically over an image. To combine the local quality assessments into a scalar value, we simply compute the mean over all local windows. We denote this as the mean SSIM

$$\mathrm{MSSIM}(X,Y) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\mathrm{SSIM}(w_{r,(i,j)}(X), w_{r,(i,j)}(Y)), \tag{10}$$

where $w_{r,(i,j)}$ extracts windows of radius r at pixel location (i, j). To avoid blocking artifacts, we use a Gaussian weighting kernel $g = \{g_1, \ldots, g_n\}$ such that the estimates are modified according to

$$\mu_x = \sum_{i=1}^{n} g_i x_i, \quad \sigma_x = \sqrt{\tilde{n} \sum_{i=1}^{n} g_i (x_i - \mu_x)^2}, \quad \sigma_{xy} = \tilde{n} \sum_{i=1}^{n} g_i (x_i - \mu_x)(y_i - \mu_y), \tag{11}$$

where $\tilde{n} = \frac{n}{n-1}$. The specific parameters of the Gaussian weighting are given in the assignment sheet.

3 Tasks

You are provided with the skeleton file `interpolation_error.py` in which you find the parts you should implement. In the file we are trying to find

$$\mathrm{PSNR}(X, \uparrow_k(\downarrow_k(X))) \text{ and } \mathrm{MSSIM}(X, \uparrow_k(\downarrow_k(X)))$$

where $\uparrow_k$ and $\downarrow_k$ are two-fold up- and downsampling operators using interpolation degree k. Specifically, you should implement (2) both in "vectorized" form and looping over all pixel values in the image (left and right hand side of (1)). Further, you should implement (9).

3.1 Report

In your report, compare the results of the different interpolation schemes qualitatively (i.e. show the resulting images) and quantitatively by means of the PSNR and SSIM. Which interpolation scheme does best? Which one of the two quantitative error measures do you think is superior?

3.2 Notes

1. Do not change the `import` statements.
2. Notice that σ_x as well as σ_y in (9) is only ever used after it was squared. This insight can ease the computations heavily (compare to identity $\mathrm{Var}[x] = \mathbb{E}[(x - \mathbb{E}[x])^2] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$). This is also seen in the notebook of the first tutorial session. Very similar tricks can be applied to calculate σ_{xy}.
3. In the file `reference_output.txt` you can find the numbers of the reference implementation, i.e. it was generated by
   `$ python interpolation_error.py > reference_output.txt`.
4. Please work with `float64` images $X \in [0, 1]^{M \times N}$ $(m = 1)$. The skeleton file already loads the input image appropriately.

Acronyms

MSE mean squared error 1, 2

PSNR peak signal to noise ratio 1, 4

SSIM structural similarity 1–4