컴퓨터 비전 세미나

12191584 김창수

3. Image Processing

3-1. Point operators

3-2. Linear filtering

3. Image Processing

- to preprocess the image and convert it into a form, suitable for further analysis.

- Examples of such operations include exposure(노출) correction and color balancing, reducing image noise, increasing sharpness(선명), or straightening(똑바로) the image by rotating it.
- Additional examples include image warping(뒤틂) and image blending, which are often used for visual effects.

- most computer vision applications, require care in designing the image processing stages in order to achieve acceptable results.

3-1. Point operators (point processes)

- where each output pixel's value depends on only the corresponding input pixel value plus, potentially, some globally collected information or parameters

- ex) brightness and contrast adjustments / color correction and transformations

3-1-1. Pixel Transforms

- general image processing operator is a function that takes one or more input images and produces an output image

$$g(\mathbf{x}) = h(f(\mathbf{x})) \quad \text{or} \quad g(\mathbf{x}) = h(f_0(\mathbf{x}), \ldots, f_n(\mathbf{x})),$$

- x is in the D-dimensional domain (usually D = 2 for images)
- f and g operate over some range,
  which can either be scalar or vector-valued for color images or 2D motion.

$\mathbf{x} = (i, j)$, and we can write

$$g(i, j) = h(f(i, j)).$$

3-1-1. Pixel Transforms

commonly used point operators

multiplication and addition with a constant $g(\mathbf{x}) = af(\mathbf{x}) + b.$ $g(\mathbf{x}) = a(\mathbf{x})f(\mathbf{x}) + b(\mathbf{x}),$

- a and b are often called the gain and bias parameters; are said to control contrast and brightness

dyadic (two-input) operator (linear blend operator) $g(\mathbf{x}) = (1-\alpha)f_0(\mathbf{x}) + \alpha f_1(\mathbf{x}).$

- perform a temporal cross-dissolve(흩어짐) between two images or videos, as seen in slide shows and film production

gamma correction $g(\mathbf{x}) = [f(\mathbf{x})]^{1/\gamma}$

- highly used non-linear transform that applied to images before further processing
- is used to remove the non-linear mapping between input radiance(빛) and quantized pixel values
- gamma 2.2 is a reasonable fit for most digital cameras.

3-1-2. Color Transforms

- In fact, adding the same value to each color channel not only increases the apparent intensity of each pixel, it can also affect the pixel's hue(색조) and saturation(채도)

- Chromaticity coordinates(색 좌표) or even simpler color ratios(색 비율) can first be computed and then used after manipulating the luminance(밝기) Y
to re-compute a valid RGB image with the same hue and saturation.

- Similarly, Color balancing can be performed either by multiplying each channel with a different scale factor or by the more complex process.

3-1-3. Compositing(합성) and matting(지우기)

- The process of extracting the object from the original image is often called matting, while the process of inserting it into another image is called compositing.

- The intermediate representation used for the foreground object between these two stages is called an alpha-matted color image

(b) (c)

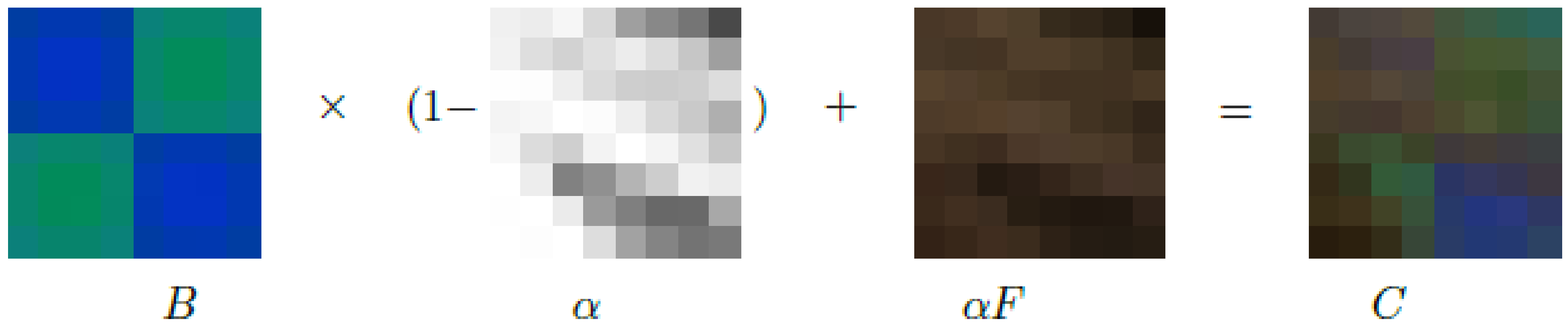
- In addition to the three color RGB channels, an alpha-matted image contains a fourth alpha channel alpha (or A) that describes the relative amount of opacity(불투명도) or fractional coverage at each pixel

3-1-3. Compositing and matting

- Pixels within the object are fully opaque (a = 1), while pixels fully outside the object are transparent (a = 0).
- Pixels on the boundary of the object vary smoothly between these two extremes.

- To composite a new image on top of an old image, the over operator is used.

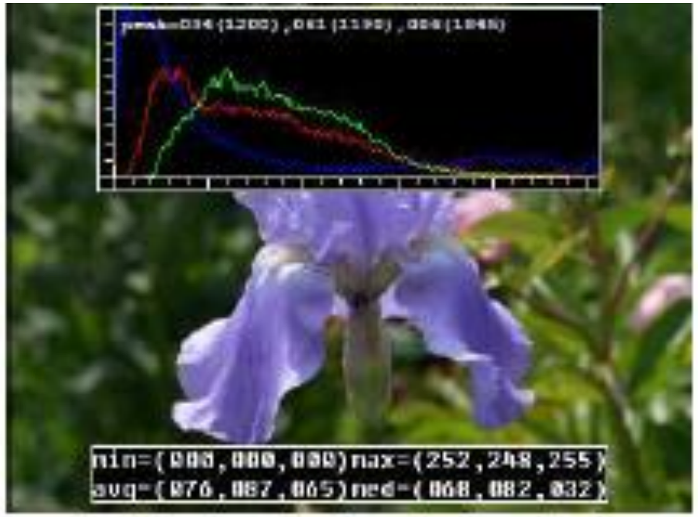
$$C = (1 - \alpha)B + \alpha F.$$

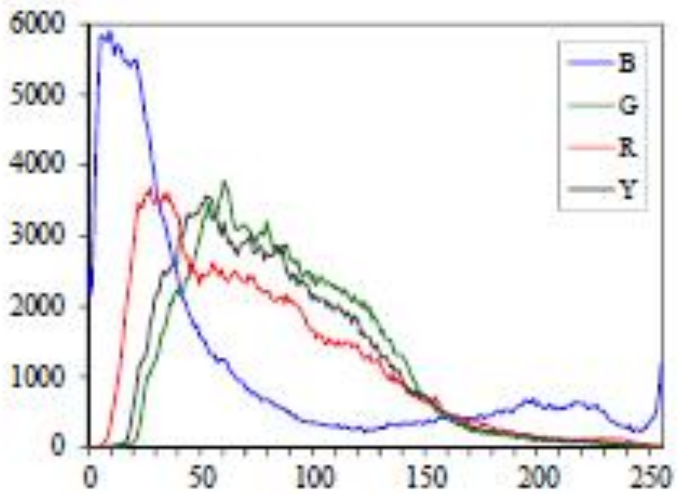


- a = 1 인 픽셀 : 배경 부분을 연하게(1 – a = 0) + 물체를 진하게(a = 1) => 물체 픽셀
- a = 0 인 픽셀 : 배경 부분을 진하게(1 – a = 1) + 물체를 연하게(a = 0) => 배경 픽셀

3-1-4. Histogram equalization(균일화)

- We can visualize the lightness values in an image by plotting the histogram of the color channels and luminance values.





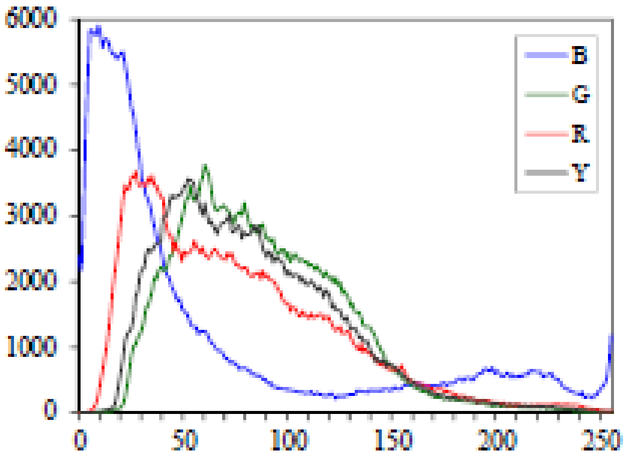
- Notice that the image in Figure has both an excess of dark values and light values, but that the mid-range values are largely under-populated.

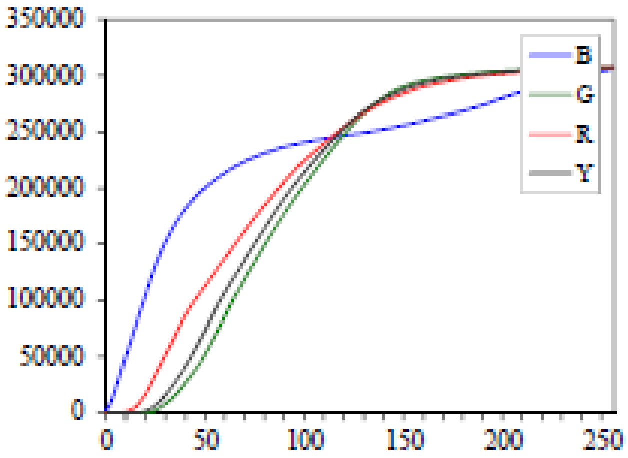
- how to simultaneously brighten some dark values and darken some light values, while still using the full extent of the available dynamic range?
- One popular answer to this question is to perform histogram equalization(균일화)

3-1-4. Histogram equalization

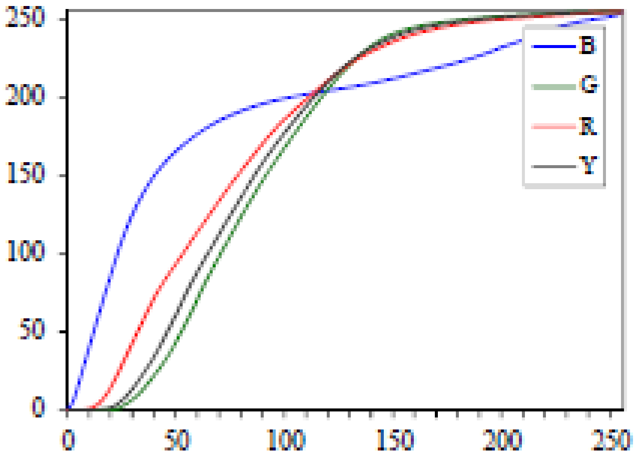
$$c(I) = \frac{1}{N}\sum_{i=0}^{I} h(i) = c(I-1) + \frac{1}{N}h(I),$$

- Cumulative distribution(누적 분포) c(I) is the integrated original distribution h(I).





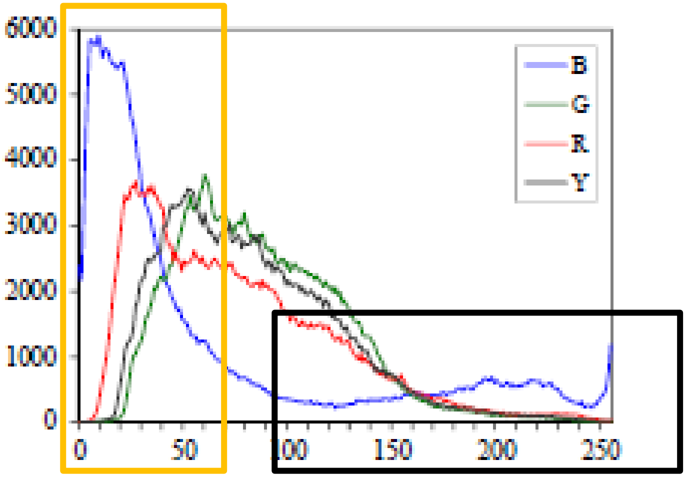
-> scaled



- We can look up its corresponding percentile c(I) and determine the final value that pixel should take.
- When working with eight-bit pixel values, the I and c axes are rescaled from [0, 255]

3-1-4. Histogram equalization

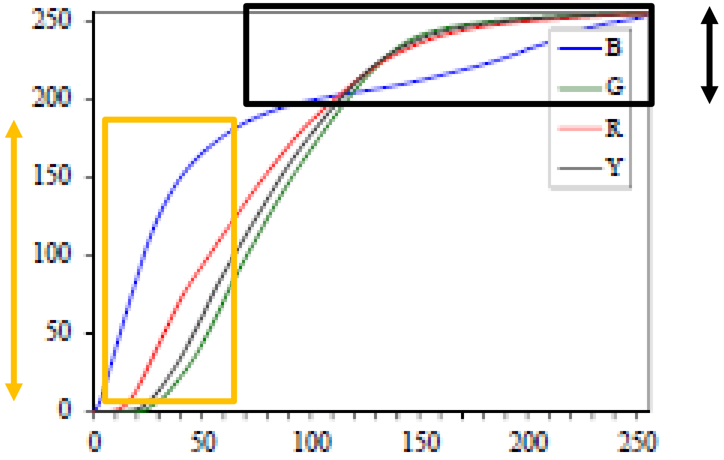
Blue Line 기준



요소 많음

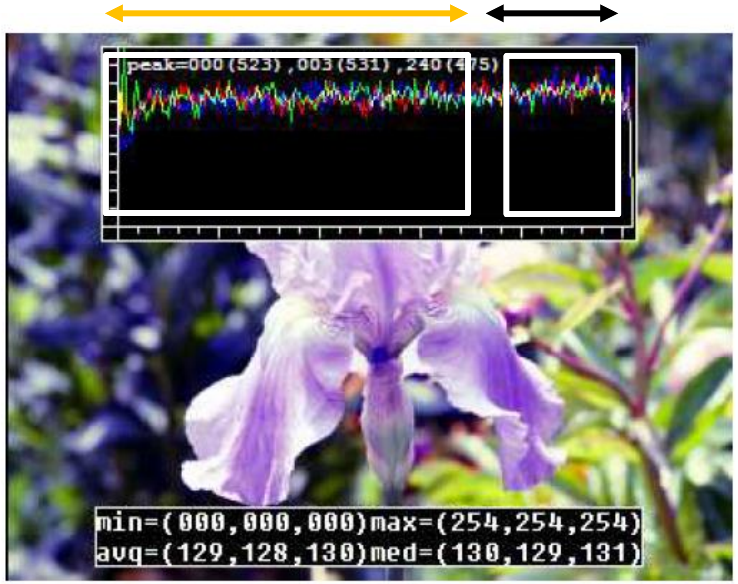
요소 적음

요소 적음
-> 좁은 범위로 매핑



요소 많음
-> 넓은 범위로 매핑

요소가 많은 부분이 흩어짐
요소가 적은 부분이 모임



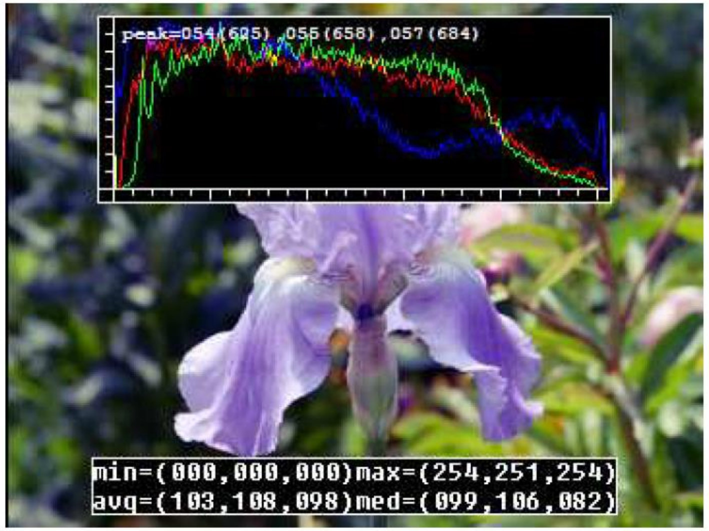
ex) 밝기 0부터 50을 가지는 요소가 10000개였다면, 밝기 0부터 200을 가지는 요소가 10000개가 됨

3-1-4. Histogram equalization

- As we can see, the resulting histogram is flat; (it is “flat” in the sense of a lack of contrast and being muddy(탁한) looking).
- One way to compensate for this is to only partially compensate for the histogram unevenness

$$f(I) = \alpha c(I) + (1-\alpha)I$$

- which is a linear blend between the cumulative distribution function and the identity transform (a straight line)



- Another potential problem with histogram equalization(in general, image brightening) is that noise in dark regions can be amplified and become more visible.

3-1-4. Locally Adaptive Histogram equalization

- While global histogram equalization can be useful,
  for some images it might be preferable to apply different kinds of equalization in different regions.

- Instead of computing a single curve, subdivide the image into M x M pixel blocks
  and perform separate histogram equalization in each sub-block

- One way to eliminate blocking artifacts is to use a moving window.

(a) (b) (c)

Figure 3.8 *Locally adaptive histogram equalization: (a) original image; (b) block histogram equalization; (c) full locally adaptive equalization.*

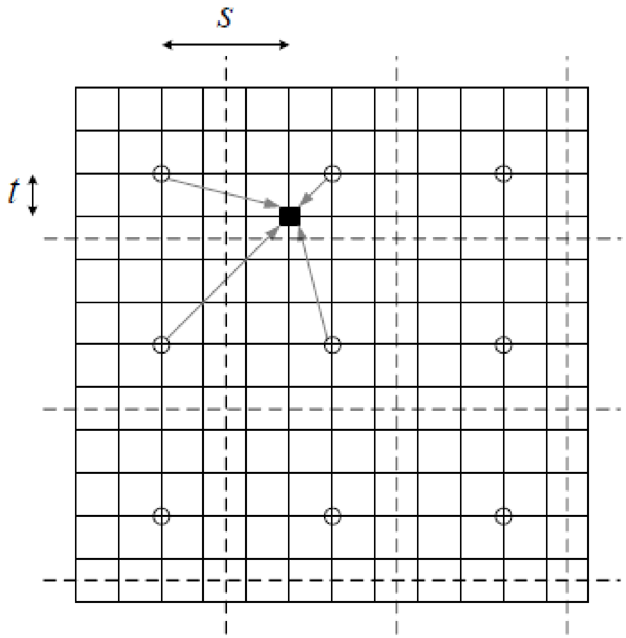
3-1-4. Histogram Interpolation(보간)

- A more efficient approach is to compute non-overlapped block-based equalization(균일화) functions as before, but to then smoothly interpolate the transfer functions as we move between blocks.
- This technique is known as adaptive histogram equalization (AHE)

- The weighting function for a given pixel can be computed as a function of its horizontal and vertical position (s, t) within a block.

$$f_{s,t}(I) = (1-s)(1-t)f_{00}(I) + s(1-t)f_{10}(I) + (1-s)tf_{01}(I) + stf_{11}(I)$$

- Instead of blending the four lookup tables for each output pixel, we can instead blend the results of mapping a given pixel through the four neighboring lookups.

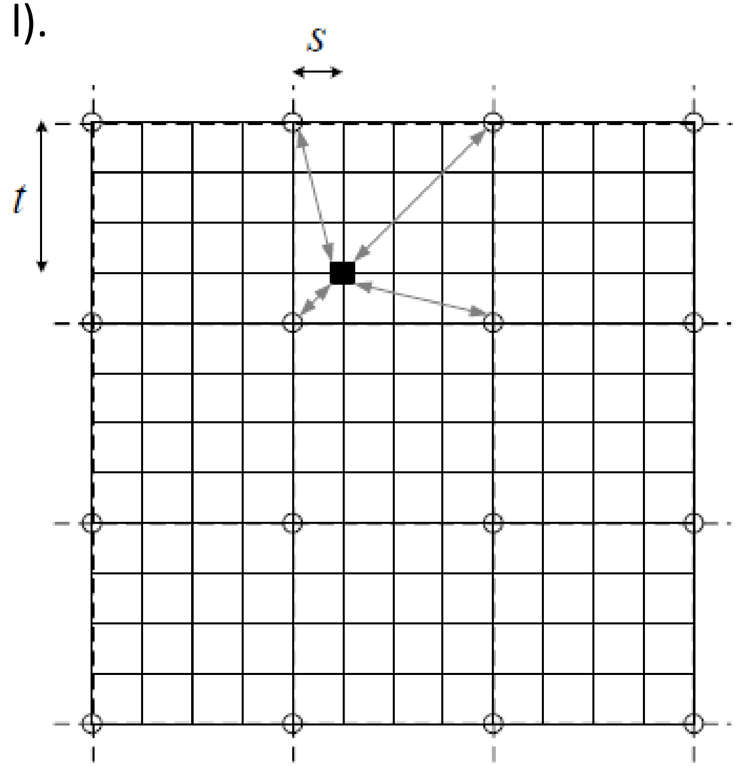


3-1-4. Histogram Interpolation(보간)

- A variant on this algorithm is to place the lookup tables at the corners of each M x M block.
- In addition to blending four lookups to compute the final value,
  we can also distribute each input pixel into four adjacent lookup tables.

$$h_{k,l}(I(i,j)) \mathrel{+}= w(i,j,k,l),$$

- w(I, j, k, l) is the bilinear weighting function between pixel (I, j) and lookup table (k, l).
- notice that the gray arrows in Figure point both ways
- This is an example of soft histogramming.



original ->

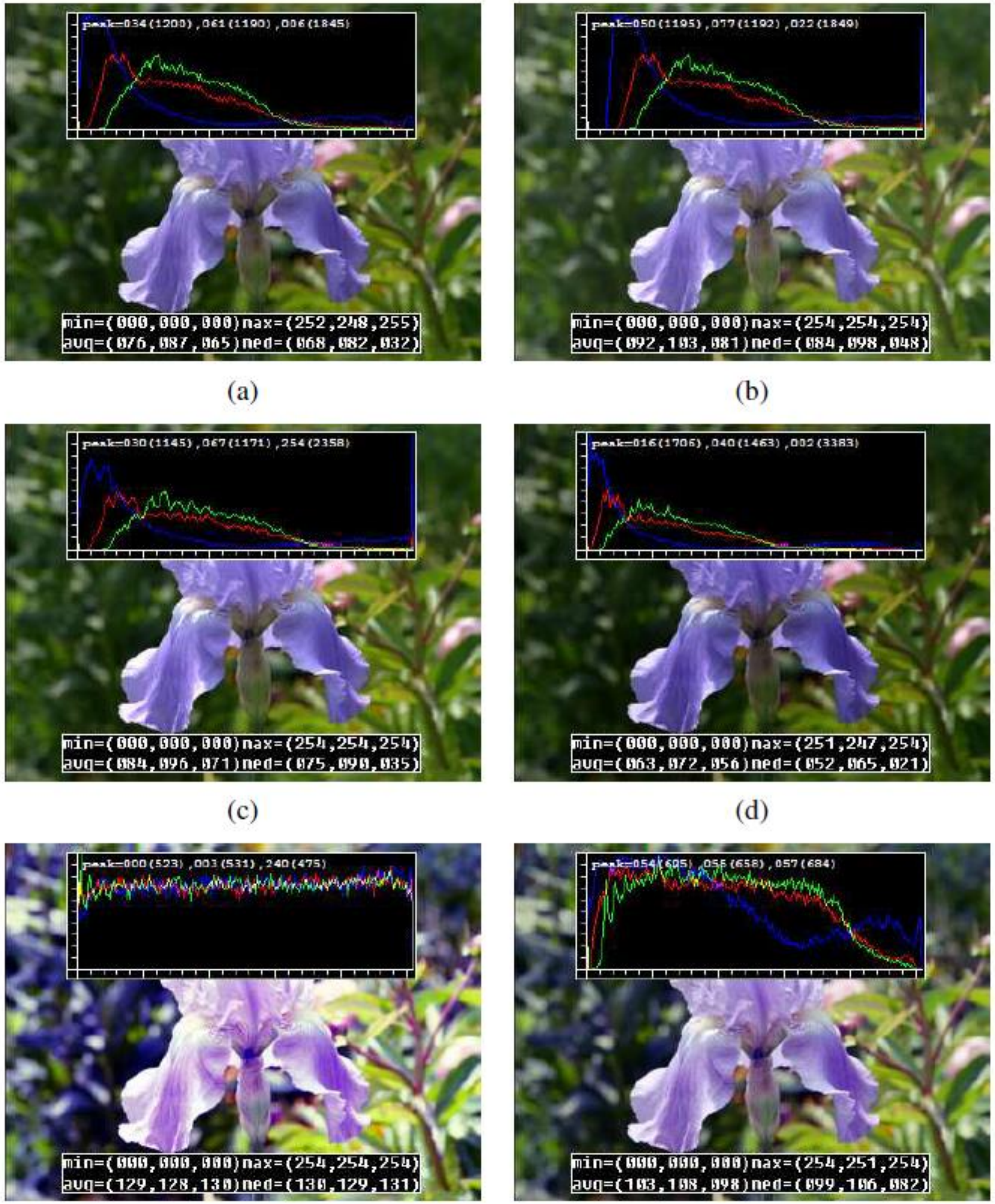
brightness increased
(additive offset, b = 16)

contrast increased
(multiplicative gain, a = 1.1)

gamma (partially)
linearized (= 1:2)

full histogram
equalization

partial histogram
equalization



(a) (b) (c) (d)