CSE463
Computer Vision: Fundamentals and Applications
Lecture 4
Point Operators and Filtering

Point Operators

Point operators are basic image processing transformations that apply adjustments to each individual pixel independently, without considering neighboring pixels. They modify pixel values based on a specific formula or parameter, such as brightness or contrast, to adjust the appearance of an image.

Pixel Transforms

3.1.1 Pixel transforms

A general image processing *operator* is a function that takes one or more input images and produces an output image. In the continuous domain, this can be denoted as

$$g(\mathbf{x}) = h(f(\mathbf{x})) \quad \text{or} \quad g(\mathbf{x}) = h(f_0(\mathbf{x}), \ldots, f_n(\mathbf{x})), \tag{3.1}$$

where $\mathbf{x}$ is in the D-dimensional (usually $D = 2$ for images) *domain* of the input and output functions f and g, which operate over some *range*, which can either be scalar or vector-valued, e.g., for color images or 2D motion. For discrete (sampled) images, the domain consists of a finite number of *pixel locations*, $\mathbf{x} = (i, j)$, and we can write

$$g(i, j) = h(f(i, j)). \tag{3.2}$$

Pixel transforms are the building blocks of point operators, allowing the adjustment of brightness and contrast. Two commonly used point processes are multiplication and addition with a constant, a > 0 and b are often called the gain and bias parameters.

1. **Brightness and Contrast Adjustment**:
   - **Multiplicative Transformation**: Adjusts the contrast by scaling pixel values. The contrast gain factor controls how much contrast is increased or decreased.

$$g(\mathbf{x}) = af(\mathbf{x}) + b.$$

   - **Additive Transformation**: Adjusts brightness by adding a bias value `b` to each pixel, shifting its intensity uniformly across the image.

$$g(\mathbf{x}) = a(\mathbf{x})f(\mathbf{x}) + b(\mathbf{x})$$

2. **Spatially Varying Gain and Bias**:
   - For effects like vignetting, the gain and bias values can vary across the image rather than being constant. This allows for more localized adjustments, often used for creative or artistic effects.

Linear Operations and Image Blending

Linear operators use the **superposition principle**, meaning the transformation applied to a combination of inputs equals the sum of the transformations applied to individual inputs. A simple example is adjusting the brightness using a multiplicative gain.

$$h(f_0 + f_1) = h(f_0) + h(f_1).$$

- **Linear Blend Operator**: This is often used in **image blending** and **cross-dissolving**, where two images are blended based on an alpha value, `α`, that varies from 0 to 1. For instance:

$$g(\mathbf{x}) = (1 - \alpha)f_0(\mathbf{x}) + \alpha f_1(\mathbf{x}).$$

   - If `α = 0`, only the background image is visible.
   - If `α = 1`, only the foreground image is visible.
   - When `0 < α < 1`, the images blend proportionally to `α`.

Non-Linear Transformations: Gamma Correction

Gamma correction is a critical step in image processing and display technology. It addresses the nonlinear relationship between the intensity values of image pixels and their perceived brightness. The goal is to ensure that images appear natural and consistent across different devices and lighting conditions.

1. Why is Gamma Correction Needed?

Human vision does not perceive brightness linearly. For example, doubling the intensity of light does not double the perceived brightness. Similarly:

- Most display devices (like monitors and TVs) do not render brightness linearly due to hardware constraints.
- Camera sensors capture light linearly, which does not align with how humans perceive brightness.

Gamma correction adjusts the pixel intensity values to align with the nonlinear perception of brightness by the human eye and the nonlinear response of display devices.

2. Gamma Function

The transformation is mathematically represented as:

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

For practical purposes:

- **Gamma > 1**: Makes darker regions brighter.
- **Gamma < 1**: Makes brighter regions darker.
- **Gamma ≈ 2.2**: Common standard used in digital displays to compensate for their default nonlinear response.

Color Transforms

Color transforms modify an image's color properties, adjusting individual channels to alter brightness, and balance, or to convert between color spaces.

1. **Understanding Color Channels**: Color images consist of correlated signals (RGB channels) due to the interaction of light, sensors, and human perception.
2. **Brightness Adjustment**:
   - Adding a constant to each RGB channel uniformly increases brightness but may alter the color balance.
   - **Chromaticity Coordinates**: Adjustments using chromaticity coordinates help maintain perceptual color qualities without affecting hue or saturation.
3. **Color Balancing**: Corrects lighting discrepancies (e.g., yellowish hue from incandescent lighting) by scaling each channel separately or applying a **3×3 color twist matrix** for complex color transformations.
4. **Color Spaces**:
   - **RGB**: Common in digital displays and image files.
   - **YCbCr**: Often used in video compression for efficient storage.
   - **HSV**: An intuitive color space for tasks involving hue manipulation.
   - **L*a*b***: Designed for color accuracy and perceptual uniformity in color-critical applications.

Image Matting and Compositing

Image matting and compositing are techniques to extract and seamlessly blend objects from one image into another.

1. **Alpha Matting**:
   - Uses an **alpha matte**, a grayscale image, where each pixel indicates transparency level.
   - **Alpha Channel (α)**: Controls transparency, with
   - $\alpha = 1$: Fully opaque (inside object)
   - $\alpha = 0$: Fully transparent (outside object)
   - $0 < \alpha < 1$: Smooth transitions on object boundaries to avoid visual artifacts (e.g., "jaggies"). Ie. for smooth transitions on edges.

2. **Compositing Formula – The Over Operator**:

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

   - Introduced by Porter and Duff (1984), the **Over operator** defines a way to blend a foreground object over a background using alpha values.
   - **Formula**: Combines RGB and alpha channels to blend images while avoiding harsh edges, ensuring smooth transitions.

Image Filtering

(*Read section 3.2 pg 119-122 of Richard Szeliski)

Image filtering adjusts pixel values based on neighboring pixels, making it essential for tasks like noise reduction, blurring, sharpening, and edge detection.

1. **Linear Filtering**:

$$g(i, j) = \sum_{k,l} f(i + k, j + l)h(k, l).$$

   - **Linear filters** apply a weighted sum of neighborhood pixel values to compute each output pixel.
   - **Kernel or Mask**: The weights for neighboring pixels, also called filter coefficients, determine the filter's effect.

2. **Correlation and Convolution**:

$$g(i, j) = \sum_{k,l} f(i + k, j + l)h(k, l). \tag{3.12}$$

The entries in the weight *kernel* or *mask* $h(k, l)$ are often called the *filter coefficients*. The above *correlation* operator can be more compactly notated as

$$g = f \otimes h. \tag{3.13}$$

A common variant on this formula is

$$g(i, j) = \sum_{k,l} f(i - k, j - l)h(k, l) = \sum_{k,l} f(k, l)h(i - k, j - l), \tag{3.14}$$

where the sign of the offsets in f has been reversed, This is called the *convolution* operator,

$$g = f * h, \tag{3.15}$$

and h is then called the *impulse response function*.[5] The reason for this name is that the kernel function, h, convolved with an impulse signal, $\delta(i, j)$ (an image that is 0 everywhere except at the origin) reproduces itself, $h * \delta = h$, whereas correlation produces the reflected signal. (Try this yourself to verify that it is so.)

- **Correlation**: Applies the kernel directly across the image.
- **Convolution**: Similar to correlation, but with reversed kernel offsets, making it shift-invariant.
- Both correlation and convolution can be represented in matrix form, where each image is flattened into a vector, enabling computational efficiency.

Exercises (from the book)

Ex 2.7: Gamma correction in image stitching. Here's a relatively simple puzzle. Assume you are given two images that are part of a panorama that you want to stitch (see Section 8.2). The two images were taken with different exposures, so you want to adjust the RGB values so that they match along the seam line. Is it necessary to undo the gamma in the color values in order to achieve this?

Answer:

Yes, it is necessary to undo the gamma in the color values to achieve accurate exposure matching along the seam when stitching images taken with different exposures.

In image stitching, particularly when dealing with two images taken with different exposures, it's crucial to ensure that the RGB values align along the seam for a smooth transition. This process often involves adjusting the brightness and color balance between the images. Regarding gamma correction, it depends on how the images were captured and stored:

1. **Gamma Correction**: Most images are encoded with gamma correction to adjust for the nonlinear response of display devices and human vision. This means that the pixel values in the image are stored in a gamma-corrected space, which is typically darker in the shadows and lighter in the highlights than a linear color space.
2. **Undoing Gamma**: To align the two images for stitching, you often need to perform operations like brightness adjustment, blending, or histogram matching. These operations should ideally be done in a *linear color space*, where the RGB values represent the actual intensities of light.
   If the images you're stitching have gamma correction applied, it's advisable to **undo the gamma correction** before performing the adjustments. This is because adjusting RGB values in a gamma-corrected color space might lead to inaccurate results, especially when manipulating pixel intensities to match exposure levels. By working in the linear space, you ensure that the color and brightness adjustments are applied correctly, and then you can apply the gamma correction back to the result after the adjustment.

Ex 2.8: White point balancing—tricky. A common (in-camera or post-processing) technique for performing white point adjustment is to take a picture of a white piece of paper and to adjust the RGB values of an image to make this a neutral color.

1. Describe how you would adjust the RGB values in an image given a sample "white color" of (R_w, G_w, B_w) to make this color neutral (without changing the exposure too much).

2. Does your transformation involve a simple (per-channel) scaling of the RGB values or do you need a full 3×3 color twist matrix (or something else)?

Answers:

1. Adjusting the RGB Values to Make the White Color Neutral

To adjust the RGB values in an image so that a sample "white color" (Rw, Gw, Bw) appears neutral, you need to scale the RGB channels in a way that the white point becomes a neutral white (where all channels are equal, typically RGB = 1 or RGB = (1, 1, 1) in normalized space).

The general approach is to compute a scaling factor for each color channel based on the ratio of the target white color to the observed white color in the image. This scaling should correct for any color casts without significantly affecting the exposure.

2. Simple Scaling vs 3x3 Color Twist Matrix

In this case, the transformation involves simple per-channel scaling of the RGB values. Since you're adjusting the RGB channels independently (multiplying each channel by a constant factor), a full 3x3 color twist matrix is not necessary for this operation. A full 3x3 matrix would be used for more complex color transformations that mix the RGB channels, but here, you're adjusting each channel independently. So, the correct transformation in this case is a simple per-channel scaling, which is a diagonal scaling operation.

Exercise (not from book)

1. What is gamma correction, and why is it important in digital image processing?
2. Explain why gamma correction is necessary to align digital images with human perception of brightness.
3. Describe how the value of gamma affects the brightness and contrast in an image.
4. Given an image pixel with an original intensity value of 0.5, calculate the gamma-corrected intensity value for gamma = 2.2 and gamma = 0.5.
5. If the gamma value is set to greater than 1, what impact does it have on the darker regions of an image? Why?
6. Describe the process of forward and inverse gamma correction. Why are both types of correction necessary?
7. Why do digital displays use a gamma value of approximately 2.2 as the standard?
8. How does gamma correction ensure consistency of brightness across different devices?
9. What are the challenges associated with gamma correction when displaying images on multiple devices with varying gamma responses?
10. How does gamma correction help in reducing data size in video compression?
11. Explain how gamma correction improves the clarity and realism of images, especially in areas with very high or low brightness levels.
12. What are some real-world applications where gamma correction is crucial, and why?
13. How would an image look if it is displayed without gamma correction on a digital screen?
14. In what way does gamma correction bridge the gap between linear intensity captured by cameras and nonlinear human visual perception?