1. Point Processing

Pixel Intensity is a weighted average of RGB content due to human responses to different colors:

$$g\left(\begin{bmatrix} R \\ G \\ B \end{bmatrix}\right) = 0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B$$

To create a negative image: $\boxed{v' = 1.0 - v}$
Contrast relates to the ability to distinguish objects in an image. High contrast implies a wider range of intensity values:

$$\text{High contrast} \Leftrightarrow \text{High variance of pixel intensities}$$

Adjusting pixel intensity non-linearly: $\boxed{s = c \cdot r^\gamma}$
Histograms summarize pixel intensity distribution, aiding in contrast stretching or compression.
Piecewise-Linear Contrast Stretching: Apply different linear transformations to different intensity ranges:

$$s(r) = \begin{cases} \alpha r & \text{for } 0 \leq r \leq r_1 \\ \beta(r - r_1) + s_1 & \text{for } r_1 < r \leq r_2 \\ \gamma(r - r_2) + s_2 & \text{for } r_2 < r \leq C_{\max} \end{cases}$$

Where:

$$\alpha = \frac{s_1}{r_1},\ \beta = \frac{s_2 - s_1}{r_2 - r_1},\ \gamma = \frac{C_{\max} - s_2}{C_{\max} - r_2}$$

2. Resizing

General Resizing Formula:

$$(x, y) = \left(\frac{(x' - 1) \cdot (w - 1)}{w' - 1} + 1, \frac{(y' - 1) \cdot (h - 1)}{h' - 1} + 1\right)$$

Nearest Neighbor For a floating point location (x, y), round to the nearest pixel:

$$\text{value} = f(\text{round}(x), \text{round}(y))$$

Pros and cons of using nearest neighbors: Easy and fast.Can result in several locations having the same value.
Bi-Linear Interpolation: Weighted average of the four nearest pixels to a non-integer location:

$$f(x, y_1) = (x_2 - x)f(A) + (x - x_1)f(B)$$

$$f(x, y_2) = (x_2 - x)f(C) + (x - x_1)f(D)$$

$$f(x, y) = \frac{(y_2 - y)}{y_2 - y_1}f(x, y_1) + \frac{(y - y_1)}{y_2 - y_1}f(x, y_2)$$

This simplifies for discrete locations where $x_2 - x_1 = y_2 - y_1 = 1$. Blend factors can be seen as percentages. The weights will add up to 1.
Bi-Linear Interpolating Edge Cases: Handling edge cases when the target location (x, y) falls on integer coordinates:

- If both x and y are integers, use the pixel at that location.
- If x is an integer but y is not, blend the above and below pixels.
- If y is an integer but x is not, blend the left and right pixels.

3. Filtering

Sigma and Blur: Increasing sigma increases the blur effect.
Mean Filtering: Simplest form of smoothing.
Gaussian Filtering:

$$G(x, y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Gaussian Kernel Size Rule of Thumb:

$$K = 2 \cdot \lceil 2\sigma \rceil + 1$$

Bilateral Filtering: Preserves edges by weighting pixels based on their spatial and intensity distance.
Weights: Combines spatial and intensity weights for edge-preserving smoothing.
Normalization:

$$I'(n, m) = \frac{1}{\|W\|}\sum_{i,j} W(i, j) \cdot I\left(n + i - \left\lfloor\frac{K}{2}\right\rfloor, m + j - \left\lfloor\frac{K}{2}\right\rfloor\right)$$

Where: $\|W\|$ is a normalization factor.
Convolution: Core operation in many filtering techniques.

$$G(x, y) = \omega * F(x, y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx, dy) \cdot F(x - dx, y - dy)$$

4. Edges

Derivative Kernels

$$\frac{\partial}{\partial x} = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & 0 & 0 \end{bmatrix}, \quad \frac{\partial}{\partial y} = \begin{bmatrix} 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 \end{bmatrix}$$

Gradient (Sobel Kernel):

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Magnitude and Direction:

$$\text{Magnitude: } M = \sqrt{G_x^2 + G_y^2} = |G_x| + |G_y|, \quad \text{Direction: } \Theta = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Non-Maximum Suppression

- A method to thin out the edges detected in the image to ideally be 1-pixel wide.
- Implemented after edge detection to mitigate the "fat sausage" effect where edges appear thick and blurred due to smoothing.
- The process involves examining the angle of the gradient at each pixel and comparing the gradient magnitude to the pixel's neighbors along the gradient direction.
  - At each pixel, compute the angle of the gradient.
  - Examine the pixels in the direction of the gradient and its opposite.
  - If the pixel's gradient magnitude is greater than the magnitudes of both neighbors in the gradient direction, keep it as an edge pixel.
  - Otherwise, suppress (set to zero) the gradient magnitude of the pixel, effectively removing it as an edge candidate.
- Note: In image processing, it is common to assume an image coordinate system where the positive-y axis points downwards.

Considerations in Non-Maximum Suppression:

- The application of filters like bilateral filters can preserve edge strength during smoothing operations.
- The size of the smoothing filter can affect the thickness of the detected edges; larger filters may result in thicker edges.
- Non-maximum suppression is a crucial step to ensure that the resulting edges are as close to the ideal of being 1-pixel wide as possible.

Hysteresis

- Part of Canny edge detection for identifying strong and weak edges.
- Pixels with gradient magnitudes less than the lower threshold T_L are marked as not edge pixels.
- Pixels with gradient magnitudes greater than or equal to the upper threshold T_H are marked as edges.
- Pixels with gradient magnitudes such that $T_L \leq |G| < T_H$ are "potential edges".
  - Look at the region around them. If there exists a pixel with gradient magnitude $\geq$ to the high threshold, mark this as an edge pixel.
  - Otherwise, mark as not an edge pixel.

Canny Edge Detector

1. Apply Gaussian smoothing to reduce noise.
2. Find gradients (possibly with derivative of Gaussian).
3. Apply non-maximum suppression.
4. Apply thresholding and/or hysteresis.