Understanding Image Representations

Computer Vision (CSCI 4220U)

Faisal Z. Qureshi

http://vclab.science.ontariotechu.ca

What is an Image?

- **Definition**: An image is a 2D matrix of pixel values.
- **Mathematical Form**:

  An image I can be represented as:

  $$I(i, j) = \text{pixel intensity at row } i \text{ and column } j$$

- **Dimensions**:
  - Grayscale: $M \times N$ matrix
  - RGB: $M \times N \times 3$ tensor (Red, Green, Blue channels)

Binary Images

- **Representation**: Each pixel is 0 (black) or 1 (white).
- **Mathematics**:
  Binary image $B(i,j)$:

$$B(i,j) = \begin{cases} 0 & \text{if pixel is black} \\ 1 & \text{if pixel is white} \end{cases}$$

Pros:

- Simple to store and process.
- Efficient for document scans and binary graphics.

Cons:

- Cannot represent complex visual information.
- Limited to black-and-white images.

Grayscale Images

- **Representation**: Pixel intensities range from 0 (black) to 255 (white) for 8-bit images.
- **Mathematics**:
  Grayscale image $G(i, j)$:

$$G(i, j) \in [0, 255]$$

Pros:

- More detail than binary images.
- Suitable for single-channel image analysis, like medical imaging.

Cons:

- No color information.
- Larger file size compared to binary images.

RGB Images

- **Representation**: Each pixel is a combination of Red (R), Green (G), and Blue (B) intensities.
- **Mathematics**:
  RGB image $R(i,j), G(i,j), B(i,j)$:

$$I(i,j) = [R(i,j), G(i,j), B(i,j)]$$

Pros:

- Captures full-color information.
- Widely used in photography and video.

Cons:

- Larger file size due to multiple channels.
- Limited scalability; resolution is fixed.

GIF Representation

- **GIF (Graphics Interchange Format)**: Supports animated images.
- **Uses**: Indexed color (256 colors max) with a color palette.
- **Allows**: Multiple frames to create animations.
- **Mathematical Representation**:

$$I(\text{frame}, i, j) = \text{color\_index from palette for pixel } (i, j)$$

$$\text{Color} = \text{Palette[color\_index]}$$

Pros:

- Lightweight format for animations.
- Supported across most platforms and devices.

Cons:

- Limited to 256 colors per frame.
- Not suitable for high-quality images or videos.

Python Libraries for Image Processing (Part 1)

- **Pillow (PIL)**:
  - General-purpose library for opening, manipulating, and saving images.
  - Example:

    ```python
    from PIL import Image
    img = Image.open("example.jpg")
    img_gray = img.convert("L")  # Convert to grayscale
    img_gray.save("example_gray.jpg")
    ```

  - **Pros**: Simple and lightweight.
  - **Cons**: Limited for advanced processing.

Python Libraries for Image Processing (Part 2)

- **OpenCV**:
  - Advanced library for image and video processing.
  - Example:

    ```
    import cv2
    img = cv2.imread("example.jpg")
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    cv2.imwrite("example_gray.jpg", gray)
    ```

  - **Pros**: Extensive functions for computer vision tasks.
  - **Cons**: Requires understanding of BGR vs RGB.

Python Libraries for Image Processing (Part 3)

- **Scikit-Image**:
  - Built on NumPy, SciPy, and Matplotlib for image analysis.
  - Example:

    ```
    from skimage import io, color
    img = io.imread("example.jpg")
    img_gray = color.rgb2gray(img)
    io.imsave("example_gray.jpg", img_gray)
    ```

  - **Pros**: Focused on image analysis.
  - **Cons**: Not as optimized for video or real-time processing.

Python Libraries for Image Processing (Part 4)

- **NumPy**:
  - Low-level library for pixel manipulation.
  - Example:

    ```
    import numpy as np
    from PIL import Image
    img = Image.open("example.jpg")
    img_array = np.array(img)
    img_array[:, :, 0] = 0  # Remove red channel
    new_img = Image.fromarray(img_array)
    new_img.save("example_no_red.jpg")
    ```

  - **Pros**: Great for custom manipulations.
  - **Cons**: No built-in image-specific functionality.

Summary of Python Libraries

Library	Use Case	Strengths	Weaknesses
Pillow (PIL)	Basic image editing	Lightweight, easy to use	Limited advanced features
OpenCV	Advanced processing	Wide range of tools	Learning curve for new users
Scikit-Image	Image analysis	NumPy-based, excellent algorithms	Limited for non-analysis tasks
NumPy	Custom manipulation	High flexibility	Requires manual implementation
Matplotlib	Visualization	Easy to create plots	Not optimized for large datasets

Copyright and License