2.1. What is an Image

August 28, 2024

Jarrian Vince G. Gojar
Instructor I
College of Information and Communications Technology, Sorsogon State University, Philippines

1 Introduction

An image is a 2D representation of a scene or object. It is a collection of pixels. Each pixel is a small square of illumination. Each pixel has a specific location and a specific color. The color of a pixel is defined by the intensity of the light at that location. The intensity of light is defined by the amount of red, green, and blue light.

1.1 Types of Images

- **Binary Image**: An image with only two colors, black and white.
- **Grayscale Image**: An image with shades of gray.
- **Color Image**: An image with multiple colors.

1.2 Python Libraries for Image Processing

- **OpenCV**: Open Source Computer Vision Library.
  - It is used for image processing, video analysis, and machine learning.
  - It is written in C++ and has Python bindings.
  - In image processing, it is used for reading, writing, and processing images.
- **PIL**: Python Imaging Library, also known as Pillow.
  - It is used for opening, manipulating, and saving many different image file formats.
  - It is written in C and has Python bindings.
  - In image processing, it is used for basic image processing tasks.
- **Scikit-Image**: A collection of algorithms for image processing.
  - It is built on top of NumPy, SciPy, and Matplotlib.
  - In image processing, it is used for filtering, segmentation, and feature extraction.
- **Matplotlib**: A 2D plotting library for Python.
  - It is used for creating static, animated, and interactive visualizations.
  - In image processing, it is used for displaying images and plots.
- **NumPy**: A library for numerical computing in Python.
  - It is used for creating and manipulating arrays.
  - In image processing, it is used for representing images as arrays.
- **SciPy**: A library for scientific computing in Python.
  - It is used for scientific and technical computing.

– In image processing, it is used for image processing algorithms.

2 Setup

[ ]:
```
%pip install opencv-python opencv-contrib-python matplotlib pillow numpy␣
↪scikit-learn scikit-image scipy
```

3 Python Implementation of OpenCV

Installation of OpenCV in Python

```
pip install opencv-python
```

```
pip install opencv-contrib-python
```

OpenCV is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source BSD license.

OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch, and Caffe. It has C++, C, Python, and Java interfaces and supports Windows, Linux, Android, and MacOS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured GPU interface is under development.

Read More:

- OpenCV Website

[ ]:
```
# Importing the OpenCV library
import cv2

# Reading an image from the file using OpenCV
imageOpenCV = cv2.imread('../../assets/images/parrot.jpg')

# Create a window and display the image
cv2.namedWindow('OpenCV Image', cv2.WINDOW_NORMAL)

# Resizing the window to 1/4th of the image size
cv2.resizeWindow('OpenCV Image', imageOpenCV.shape[1] // 4, imageOpenCV.
↪shape[0] // 4)

# Displaying the image
cv2.imshow('OpenCV Image', imageOpenCV)

# Waiting for a key press and then closing the window
cv2.waitKey(0)
cv2.destroyAllWindows()
```

OpenCV reads an image in BGR format by default. The code above reads an image from the file

using OpenCV, creates a window, and displays the image in the window. The window is resized to 1/4th of the image size and the image is displayed in the window. The window is closed when a key is pressed.

The image is read using `cv2.imread()` function. The `cv2.namedWindow()` function is used to create a window with a name. The `cv2.resizeWindow()` function is used to resize the window to 1/4th of the image size. The `cv2.imshow()` function is used to display the image in the window. The `cv2.waitKey()` function is used to wait for a key press. The `cv2.destroyAllWindows()` function is used to close the window.

Important codes to remember:

- `cv2.imread()`
  - This function reads an image from the file.
  - The image is read in the BGR color format.
  - Syntax: `cv2.imread(filename, flags)`
- `cv2.namedWindow()`
  - This function creates a window with a specified name and flags.
  - The window can be configured using flags. Read More
  - Syntax: `cv2.namedWindow(winname, flags)`
- `cv2.resizeWindow()`
  - This function resizes the window.
  - The window name and the width and height of the window are passed as arguments.
  - Syntax: `cv2.resizeWindow(winname, width, height)`
- `cv2.imshow()`
  - This function displays the image in the window.
  - The window name and the image are passed as arguments.
  - Syntax: `cv2.imshow(winname, mat)`
- `cv2.waitKey()`
  - This function waits for a key press.
  - The delay in milliseconds is passed as an argument.
  - If the delay is 0, the function waits indefinitely for a key press.
- `cv2.destroyAllWindows()`
  - This function closes all the windows.
  - The window name can be passed as an argument to close a specific window.
  - Syntax: `cv2.destroyAllWindows(winname)`

4 Python Implementation of OpenCV Image using Matploblib

Installation of Matplotlib

```
pip install matplotlib
```

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

Matplotlib is used to display the image in the form of a plot. The image is converted to RGB format before displaying it using Matplotlib. An OpenCV image is in BGR format, so it needs to be converted to RGB format before displaying it using Matplotlib.

Read More:

- Matplotlib Image

[ ]:
```
# Importing the OpenCV and Matplotlib libraries
import cv2
import matplotlib.pyplot as plt

# Reading an image from the file using OpenCV
imageOpenCVwithMatplotlib = cv2.imread('../../assets/images/parrot.jpg')

# # Convert color from BGR to RGB
# imageOpenCVwithMatplotlib = cv2.cvtColor(imageOpenCVwithMatplotlib, cv2.
  ↪COLOR_BGR2RGB)

# Displaying the image using Matplotlib
print("OpenCV Image with Matplotlib")
plt.figure('OpenCV with Matplotlib Image')
plt.imshow(imageOpenCVwithMatplotlib)
plt.title('OpenCV Image with Matplotlib', y=-0.1)
plt.axis('off')
plt.show()
```

```
OpenCV Image with Matplotlib
```

OpenCV Image with Matplotlib

The code above reads an image from the file using OpenCV and displays it using OpenCV and Matplotlib. When displaying an image using OpenCV, the image is displayed in a window that is resizable. However, when displaying an image using Matplotlib, the image is displayed in a Matplotlib figure that is not resizable. The `plt.figure()` function is used to create a new figure with the given name. The image is displayed using the `plt.imshow()` function of Matplotlib, which takes the image as input and displays it in a figure. The `plt.axis()` function is used to turn off the axis of the figure. The `plt.show()` function is used to display the figure. The code above displays the image using OpenCV and Matplotlib. The image is displayed in a window using OpenCV and in a figure using Matplotlib.

Image read using OpenCV are in BGR format. To display the image using Matplotlib, the color of the image needs to be converted from BGR to RGB format. This can be done using the `cv2.cvtColor()` function of OpenCV. The `cv2.cvtColor()` function takes the image and the color conversion code as input and returns the image in the desired color format. The color conversion code `cv2.COLOR_BGR2RGB` is used to convert the color of the image from BGR to RGB format.

Important codes to remember:

- `cv2.cvtColor()`
  - This function is used to convert the color of the image from one color format to another.
  - OpenCV has many options for color conversion. Read More
  - Syntax: `cv2.cvtColor(src, code[, dst[, dstCn]]) -> dst`
- `plt.figure()`
  - This function is used to create a new figure with the given name.
  - Syntax: `plt.figure(num=None, figsize=None, dpi=None, facecolor=None, edgecolor=None, frameon=True, FigureClass=<class 'matplotlib.figure.Figure'>, clear=False, **kwargs)`
- `plt.imshow()`
  - This function is used to display the image in the figure.
  - Syntax: `plt.imshow(X, cmap=None, norm=None, aspect=None, interpolation=None, alpha=None, vmin=None, vmax=None, origin=None, extent=None, shape=<deprecated parameter>, filternorm=1, filterrad=4.0, imlim=<deprecated parameter>, resample=None, url=None, *, data=None, **kwargs)`
- `plt.axis()`
  - This function is used to control the axis of the figure.
  - Syntax: `plt.axis(*v, **kwargs)`
- `plt.show()`
  - This function is used to display the figure.
  - Syntax: `plt.show(*args, **kwargs)`

5 Python Implementation of PIL or Pillow Library

Installation of Pillow

```
pip install pillow
```

PIL (Python Imaging Library) is a free library for the Python programming language that adds

support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X, and Linux. The current version of the library is known as Pillow.

Pillow is a fork of the original PIL library and is actively maintained. It is recommended to use the Pillow library instead of the PIL library.

Read More:

- Pillow Documentation

[ ]:
```
# Importing PIL and Matplotlib libraries
import matplotlib.pyplot as plt
from PIL import Image

# Reading an image from the file using PIL
imagePIL = Image.open('../../assets/images/parrot.jpg')

# # Displaying the image using PIL
# print("Displaying the image using PIL")
# imagePIL.show()

# Displaying the image using Matplotlib
print("PIL Image with Matplotlib")
plt.figure('PIL with Matplotlib Image')
plt.imshow(imagePIL)
plt.title('PIL Image with Matplotlib', y=-0.1)
plt.axis('off')
plt.show()
```

```
PIL Image with Matplotlib
```

PIL Image with Matplotlib

The code above reads an image from the file using PIL by using the `Image.open()` function. The image is then displayed using the `image_object.show()` method of the Image class. The image can also be displayed using Matplotlib.

Important codes to remember:

- `Image.open()`
  - This method is used to open an image file.
  - Syntax: `Image.open(file)`
- `image_object.show()`
  - This method is used to display the image using the default image viewer.
  - Syntax: `image_object.show()`

6 Python Implementation of Scikit-Image

Installation of Scikit-Image

```
pip install scikit-learn scikit-image
```

Scikit-Image is a collection of algorithms for image processing. It is built on top of NumPy, SciPy, and Matplotlib. It is a free software and is distributed under the BSD license. It is a part of the larger SciPy ecosystem.

Scikit-Image is used for image processing tasks such as image filtering, image segmentation, and

image restoration. It is also used for feature extraction, image registration, and image enhancement.

Scikit-Image is a powerful library that provides a wide range of image processing algorithms. It is widely used in the field of computer vision, medical imaging, and remote sensing.

Read More:

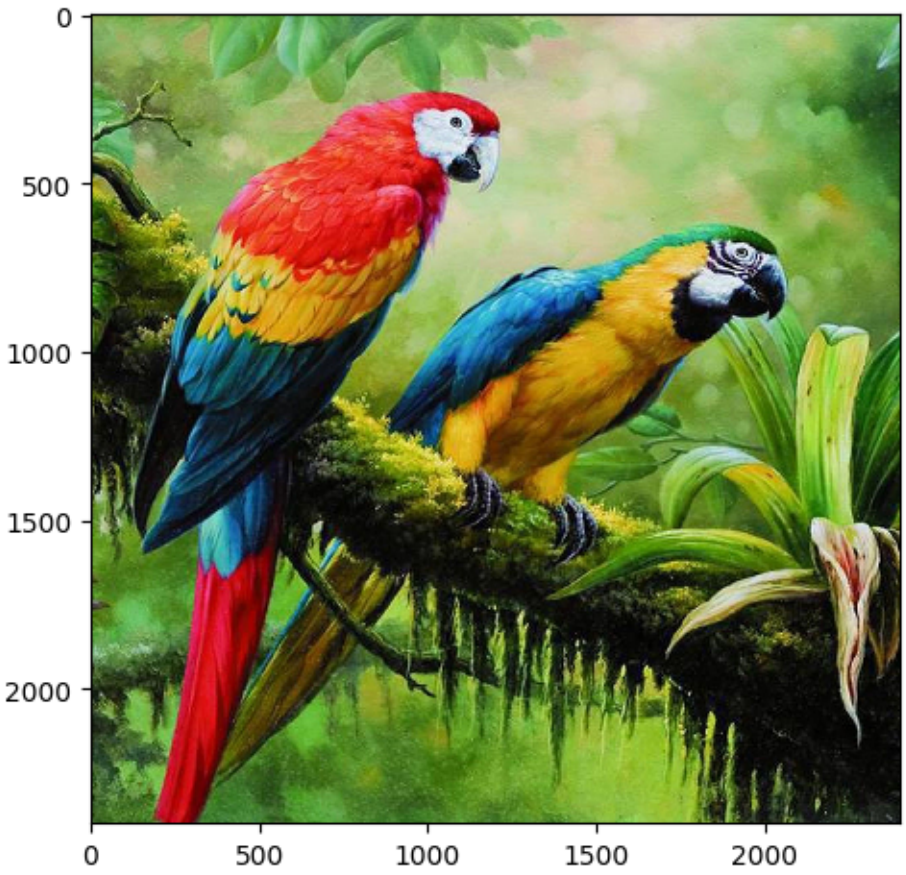
- Scikit-Image Documentation

[ ]:
```
# Importing the Scikit-Image library
from skimage import io

# Reading an image from the file using Scikit-Image
imageScikit = io.imread('../../assets/images/parrot.jpg')

# Displaying the image using Scikit-Image
print("Scikit-Image Image")
io.imshow(imageScikit)
io.show()
```

```
Scikit-Image Image
```



The code above reads an image from the file using the `io.imread()` function of the Scikit-Image library. The image is then displayed using the `io.imshow()` function of the Scikit-Image library.

The `io.imread()` function reads an image from the file and returns a NumPy array representing the image. The `io.imshow()` function displays the image using the Matplotlib library.

Important codes to remember:

- `io.imread()`
  - Reads an image from the file and returns a NumPy array representing the image.
  - Syntax: `io.imread(file_path)`
- `io.imshow()`
  - Displays the image using the Matplotlib library.
  - Syntax: `io.imshow(image)`

7 Python Implementation of NumPy

Installation of NumPy

```
pip install numpy
```

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

NumPy is a fundamental package for scientific computing with Python. It contains among other things:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

For Image Processing, NumPy is used to store and manipulate the image data. NumPy provides a high-performance multidimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with Python.

Read More:

- NumPy Documentation
- NumPy Image Processing

[ ]:
```python
# Importing the OpenCV, NumPy, and Matplotlib libraries
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Reading an image from the file using OpenCV
imageOpenCV = cv2.imread('../../assets/images/parrot.jpg')
```

```
# Converting the image to NumPy array
imageNumPy = np.array(imageOpenCV)

# Converting color from BGR to RGB
imageNumPy = cv2.cvtColor(imageNumPy, cv2.COLOR_BGR2RGB)

# Displaying the image using Matplotlib
print("NumPy Image with Matplotlib")
plt.figure('NumPy Image with Matplotlib')
plt.imshow(imageNumPy)
plt.title('NumPy Image with Matplotlib', y=-0.1)
plt.axis('off')
plt.show()
```

```
NumPy Image with Matplotlib
```

NumPy Image with Matplotlib

The code above reads an image from the file using OpenCV, converts the image to a NumPy array, and then displays the image using Matplotlib.

The image is read using the `cv2.imread()` function, which reads the image from the file. The image is then converted to a NumPy array using the `np.array()` function. The image is displayed using Matplotlib with the `plt.imshow()` function.

The `plt.imshow()` function displays the image. The `plt.axis('off')` function removes the axis from the image. The `plt.show()` function displays the image.

To convert the color from BGR to RGB, the `cv2.cvtColor()` function can be used.

Important codes to remember:

- `np.array()`
  - Converts the image to a NumPy array
  - Syntax: `np.array(Image)`

8 Python Implementation of SciPy

Installation of SciPy

```
pip install scipy
```

SciPy is a library that is used for scientific and technical computing. It is built on top of NumPy and provides a large number of functions that operate on NumPy arrays and are useful for different types of scientific and engineering applications. Some of the key features of SciPy include:

- Integration
- Optimization
- Interpolation
- Linear Algebra
- Statistics
- Signal Processing
- Image Processing
- Sparse Matrix Handling
- Clustering
- Spatial Data Structures
- File Input/Output
- Special Functions
- Fast Fourier Transforms
- Genetic Algorithms
- Ordinary Differential Equations
- Partial Differential Equations

For Image Processing, SciPy provides the following modules:

- ndimage: This module provides functions for multi-dimensional image processing.
- misc: This module provides functions for reading, writing, and displaying images.
- io: This module provides functions for reading and writing image files.
- interpolate: This module provides functions for image interpolation.
- signal: This module provides functions for signal processing.
- fftpack: This module provides functions for fast Fourier transforms.

Read More:

- SciPy Documentation
- SciPy Image Processing

[ ]:
```
# Importing the OpenCV SciPy, and Matplotlib libraries
import cv2
import matplotlib.pyplot as plt
from scipy import ndimage

# Reading an image from the file using OpenCV
imageOpenCV = cv2.imread('../../assets/images/parrot.jpg')

# Converting color from BGR to RGB
imageOpenCV = cv2.cvtColor(imageOpenCV, cv2.COLOR_BGR2RGB)

# Rotating the image by 45 degrees
imageSciPyRotated = ndimage.rotate(imageOpenCV, 45)

# Displaying the original and rotated images using Matplotlib
print("Original and Rotated Images using SciPy")
plt.figure('Original and Rotated Images using SciPy')

plt.subplot(1, 2, 1)
plt.imshow(imageOpenCV)
plt.axis('off')
plt.title('(a) Original Image', y=-0.15)

plt.subplot(1, 2, 2)
plt.imshow(imageSciPyRotated)
plt.axis('off')
plt.title('(b) Rotated Image', y=-0.15)

plt.show()
```

```
Original and Rotated Images using SciPy
```

(a) Original Image (b) Rotated Image

The code above reads an image using OpenCV, rotates the image by 45 degrees using the rotate() function from the ndimage module of SciPy, and then displays the original and rotated images using Matplotlib.

The image is read using `cv2.imread()` function from OpenCV. The image is then rotated by 45 degrees using the `rotate()` function from the `ndimage` module of SciPy. The rotated image is displayed using Matplotlib.

To display the original and rotated images side by side, the `plt.subplot()` function is used to create two subplots in a single figure. The original image is displayed in the first subplot, and the rotated image is displayed in the second subplot. The `plt.imshow()` function is used to display the images in the subplots, and the `plt.title()` function is used to set the titles of the subplots. The `plt.axis('off')` function is used to turn off the axes of the subplots.

The `plt.show()` function is used to display the figure containing the original and rotated images. The first subplot displays the original image, and the second subplot displays the rotated image.

To convert the color from BGR to RGB, the `cv2.cvtColor()` function can be used before rotating the image. This is done to ensure that the colors are displayed correctly in the right order.

Important codes to remember:

- `ndimage.rotate()`
  - This function is used to rotate an image by a specified angle.
  - Syntax: `ndimage.rotate(input, angle, reshape=True)`
  - Parameters:
    - `input`: The input image to be rotated.
    - `angle`: The angle by which the image should be rotated.
    - `reshape`: A boolean value that specifies whether the output image should be reshaped to contain the full rotated image.
  - Returns: The rotated image.

- `plt.subplot()`
  - This function is used to create subplots in a single figure.
  - Syntax: `plt.subplot(nrows, ncols, index)`
  - Parameters:
    - `nrows`: The number of rows of subplots.
    - `ncols`: The number of columns of subplots.
    - `index`: The index of the current subplot.
  - Returns: The current subplot.
- `plt.title()`
  - This function is used to set the title of a subplot.
  - Syntax: `plt.title(label)`
  - Parameters:
    - `label`: The title of the subplot.
  - Returns: None.

9 Summary

- An image is a 2D representation of a scene or object.
- It is a collection of pixels, each with a specific location and color.
- An image can be binary, grayscale, or color.
- Python libraries for image processing include OpenCV, PIL, Scikit-Image, Matplotlib, NumPy, and SciPy.
- OpenCV is used for reading, writing, and processing images.
- PIL is used for opening, manipulating, and saving images.
- Scikit-Image is used for filtering, segmentation, and feature extraction.
- Matplotlib is used for displaying images and plots.
- NumPy is used for representing images as arrays.
- SciPy is used for image processing algorithms.

10 References

- Thomas G. (2022). Graphic Designing: A Step-by-Step Guide (Advanced). Larsen & Keller. ISBN: 978-1-64172-536-1
- Singh M. (2022). Computer Graphics and Multimedia. Random Publications LLP. ISBN: 978-93-93884-95-4
- Singh M. (2022). Computer Graphics Science. Random Publications LLP. ISBN: 978-93-93884-03-9
- Singh M. (2022). Computer Graphics Software. Random Publications LLP. ISBN: 9789393884114
- Tyagi, V. (2021). Understanding Digital Image Processing. CRC Press.
- Ikeuchi, K. (Ed.). (2021). Computer Vision: A Reference Guide (2nd ed.). Springer.
- Bhuyan, M. K. (2020). Computer Vision and Image Processing. CRC Press.
- Howse, J., & Minichino, J. (2020). Learning OpenCV 4 Computer Vision with Python 3: Get to grips with tools, techniques, and algorithms for computer vision and machine learning. Packt Publishing Ltd.
- Kinser, J. M. (2019). Image Operators: Image Processing in Python. CRC Press.