Sensor Technology and Image Analysis Lab 1

Environment Setup Guide for Image Processing and Machine Learning

Atieh Sahraeidolatkhaneh: atieh.sahraeidolatkhaneh@hv.se

Yongcui Mi: yongcui.mi@hv.se

Overview

This guide explains how to set up a clean Python environment using Miniconda, how to launch JupyterLab each time you work, and provides a simple example at the end to help you access basic image information and perform essential image operations.

1. Python Environment Quickstart

Recommended to follow this quickstart to create a ready-to-use Conda setup. (If you already have your own Python setup, you are free to use it — just make sure you can import and run the same packages (opencv-python, numpy, matplotlib).)

1.1. Install Miniconda

Miniconda provides an isolated and reliable Python environment, preventing conflicts between different libraries.

- Visit: `https://www.anaconda.com/docs/getting-started/miniconda/install`
- Download the installer based on your system, for the computers in the lab, select **Miniconda3 Windows 64-bit**.

Note: You can follow this video: `https://www.youtube.com/watch?v=AgnAs0nPEVg`

1.2. Open Miniconda Prompt

Open the Start Menu and search for **Miniconda Prompt**. You will see:

```
(base) C:\Users\YourName>
```

1.3. Create a Course Environment

Creating a separate environment (e.g.,*stb600*) ensures all course packages remain isolated and reproducible.

Run:

```
conda create -n stb600 python=3.11
```

Press **Y** when asked.

1.4. Activate the Environment

Activating the environment ensures Jupyter and Python use the correct packages.

```
conda activate stb600
```

You should see:

```
(stb600) C:\Users\YourName>
```

1.5. Install Required Libraries

Install the core packages needed for image processing and machine learning.

```
conda install numpy scipy matplotlib scikit-image scikit-learn
```

Install OpenCV:

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

Install JupyterLab:

```
conda install jupyterlab
```

1.6. Launching JupyterLab

Each time you start working:

1. Open Miniconda Prompt.
2. Activate the environment:

```
conda activate stb600

```

3. Move to your project folder, for example:

```
cd C:\Users\YourName\Documents\image_processing_course

```

4. Launch JupyterLab:

```
jupyter lab

```

Starting JupyterLab from the correct folder keeps notebooks and images organized.

1.7. Verify Setup

```
# test_setup.py
import sys
import cv2
import numpy as np
import matplotlib.pyplot as plt

print("Python version:", sys.version)
print("OpenCV version:", cv2.__version__)
print("NumPy version:", np.__version__)

# Create a simple test image (grayscale gradient)
img = np.linspace(0, 255, 256, dtype=np.uint8)
img = np.tile(img, (256, 1))

# Show image using OpenCV and Matplotlib
cv2.imshow("OpenCV Test Window", img)
cv2.waitKey(1000)  # show for 1 second
cv2.destroyAllWindows()

plt.imshow(img, cmap='gray')
plt.title("Matplotlib Display Test")
plt.show()

print("Environment test completed successfully!")
```

1.8. Summary of Commands

```
# One-time installation
conda create -n stb600 python=3.11
conda activate stb600
conda install numpy scipy matplotlib scikit-image scikit-learn
pip install opencv-python opencv-contrib-python
```

```
conda install jupyterlab

# Each time you start working
conda activate stb600
cd <your-folder>
jupyter lab
```

2. Simple Example: Basic Image Operations

Below is an example demonstrating how to load, display, modify, and save an image.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Read image
img = cv2.imread("test.jpg")##test.jpg should be in the same folder as
    your *.ipynb file

# Check if image loaded
if img is None:
    raise ValueError("Image not found. Place test.jpg in the same folder
    as your notebook.")

# Display image (convert BGR RGB)
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title("Original Image")
plt.axis("off")
plt.show()

# Print image information
print("Image shape:", img.shape)      # (H, W, C)
print("Data type:", img.dtype)
print("Min/Max values:", img.min(), img.max())

# Take a region of interest (ROI)
roi = img[50:150, 100:200]
plt.imshow(cv2.cvtColor(roi, cv2.COLOR_BGR2RGB))
plt.title("ROI")
plt.axis("off")
```

```
plt.show()

# Modify pixel values: draw a red square
img[50:150, 50:150] = [0, 0, 255]  # BGR format

# Display modified image
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title("Modified Image")
plt.axis("off")
plt.show()

# Save result
cv2.imwrite("modified_output.jpg", img)
print("Saved: modified_output.jpg")
```