Image Forensics Project: Tampering Localization & Source Camera Identification

This project integrates two essential computer vision techniques in digital forensics:

1. Tampering Localization: Detecting and segmenting manipulated regions in an image using a ConvNeXt-based segmentation model.
2. Source Camera Identification (SCI): Identifying the source camera of an image using PRNU (Photo-Response Non-Uniformity) patterns extracted via denoising (FFDNet) and classified using a ResNet model.

Project Structure

```
Project Structure:

project-root/
 configs/                  # Config files for MMSegmentation (ConvNeXt)
 data/
    CASIA2/              # Dataset for tampering localization
    custom/              # SCI dataset (D01D35 folders)
 mmseg/                   # MMSegmentation framework
 ffdnet_tf/               # FFDNet implementation in TensorFlow
 prnu_utils/              # PRNU extraction and camera ID helpers
 train.py                 # Training script for ConvNeXt segmentation
 test.py                  # Evaluation script for segmentation
 sci_train.py             # Training script for SCI (ResNet + PRNU)
 sci_test.py              # Testing script for SCI
 interface.py             # Optional GUI (Tkinter-based)
 README.md
```

1. Environment Setup

```
1. Environment Setup

Python Environment:
$ conda create -n image-forensics python=3.8 -y
$ conda activate image-forensics

Dependencies:
$ pip install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu121
$ pip install mmcv-full==1.6.2 -f
https://download.openmmlab.com/mmcv/dist/cu121/torch2.0/index.html
$ pip install mmsegmentation==0.29.0
$ pip install tensorflow==2.10.0
$ pip install scikit-image opencv-python matplotlib tqdm pandas seaborn
```

2. Tampering Localization

```
2. Tampering Localization (ConvNeXt + MMSeg)

Dataset Structure:
data/CASIA2/
 images/
 masks/
 ...

To Train:
$ python train.py

To Test:
$ python test.py
```

3. Source Camera Identification

```
3. Source Camera Identification (PRNU + FFDNet + ResNet)

Dataset:
Organized in 'data/custom/' with D01D35 folders for each device.

Preprocessing:
- Denoising with FFDNet (TensorFlow)
- PRNU Extraction (Residual = Original - Denoised)
- Feature extraction with ResNet (Keras or PyTorch)

To Train:
$ python sci_train.py

To Test:
$ python sci_test.py
```