```
import subprocess

from typing import Tuple

import hiearchical_mamba.depth_pro as depth_pro

from loguru import logger



class DepthProRunner:

    """

    A class to handle installation, setup, and running of the Depth-Pro model.


    Attributes:

    -----------

    image_path : str

        Path to the image for depth prediction.


    Methods:

    --------

    install_dependencies():

        Installs dependencies via pip.


    download_pretrained_models():

        Downloads pretrained models via a shell script.


    run_from_commandline():

        Runs the depth model using the command line interface on a single image.
```

```
run_from_python():
    Runs the depth model programmatically from within Python.
"""


def __init__(self, image_path: str = "./data/example.jpg"):
    self.image_path = image_path
    logger.info(
        f"Initialized DepthProRunner with image path: {self.image_path}"
    )


def install_dependencies(self):
    """Installs the required dependencies via pip."""
    logger.info("Installing dependencies...")
    try:
        subprocess.run(["pip", "install", "-e", "."], check=True)
        logger.info("Dependencies installed.")
    except subprocess.CalledProcessError as e:
        logger.error(f"Failed to install dependencies: {e}")


def download_pretrained_models(self):
    """Downloads pretrained models by running a shell script."""
    logger.info("Downloading pretrained models...")
    try:
        subprocess.run(
            ["bash", "get_pretrained_models.sh"], check=True
        )
```

```
            logger.info("Pretrained models downloaded.")
        except subprocess.CalledProcessError as e:
            logger.error(f"Failed to download pretrained models: {e}")

    def run_from_commandline(self):
        """Runs the depth prediction on a single image via command line."""
        logger.info(
            f"Running depth prediction from command line on image: {self.image_path}"
        )
        try:
            subprocess.run(
                ["depth-pro-run", "-i", self.image_path], check=True
            )
            logger.info(
                "Depth prediction completed from command line."
            )
        except subprocess.CalledProcessError as e:
            logger.error(f"Command line depth prediction failed: {e}")

    def run_from_python(self) -> Tuple:
        """
        Runs the depth model programmatically from within Python.

        Returns:
        --------
        Tuple containing depth in meters and focal length in pixels.
```

```
"""
logger.info("Running depth prediction from Python...")
try:
    # Load model and preprocessing transform
    model, transform = depth_pro.create_model_and_transforms()
    model.eval()

    # Load and preprocess the image
    image, _, f_px = depth_pro.load_rgb(self.image_path)
    image = transform(image)

    # Run inference
    prediction = model.infer(image, f_px=f_px)
    depth = prediction["depth"]  # Depth in meters
    focallength_px = prediction[
        "focallength_px"
    ]  # Focal length in pixels

    logger.info(
        f"Depth prediction successful. Depth: {depth} m, Focal length: {focallength_px} px"
    )
    return depth, focallength_px
except Exception as e:
    logger.error(f"Depth prediction failed: {e}")
    return None
```

```
if __name__ == "__main__":
    runner = DepthProRunner(image_path="swarmslogobanner.png")

    # Install dependencies, download models, and run the model
    runner.install_dependencies()
    runner.download_pretrained_models()

    # Uncomment to run from commandline
    # runner.run_from_commandline()

    # Uncomment to run from Python
    depth, focal_length = runner.run_from_python()
    print(depth)
    print(focal_length)
```