High-Fidelity C++ Face Restoration Engine

This project is a high-performance, native C++ application designed to restore and enhance low-quality or blurred facial images. It leverages a state-of-the-art Generative Facial Prior GAN (GFPGAN) model, ported to a C++ environment for maximum speed and efficiency. The application intelligently detects faces within an image, processes them through the deep learning model, and seamlessly integrates the restored face back into the original image.

Key Features

1. AI-Powered Restoration: Utilizes a pre-trained GFPGAN model to generate realistic facial details.
2. High-Performance C++: Built entirely in C++ for optimal execution speed and low overhead.
3. Cross-Platform: Uses CMake for a consistent build process across Windows, Linux, and macOS.
4. Standalone Inference: Operates independently of Python after an initial model conversion step.

Technology Stack

1. C++17: For modern, efficient, and robust code.
2. LibTorch (PyTorch C++ API): For loading and executing the deep learning model.
3. OpenCV: For all image processing tasks, including I/O and face detection.
4. CMake: For cross-platform build automation and dependency management.

Step-by-Step Guide to Execution

Step 1: System Prerequisites

Ensure the following components are installed on your system: C++ Compiler & CMake (Visual Studio 2019+, build-essential, or Xcode), OpenCV Library, and LibTorch Library.

Step 2: Model Preparation

Convert the Python .pth model to a TorchScript .pt model using the provided Python script to generate gfpgan_v1.4_traced.pt.

Step 3: Project Setup

Organize your project with the required structure including model and face detector files (prototxt and caffemodel).

Step 4: Build Configuration

Edit CMakeLists.txt to set OpenCV_DIR and Torch_DIR paths.

Step 5: Compile and Run

Build and execute using CMake and make (Linux/macOS) or Visual Studio (Windows).

Usage

Once compiled, the program can be run from the command line with the following syntax:

```
./enhancer
```