OFIQ Library Build Instructions

This document provides instructions for building the `ofiq_lib.dll` native library.

Prerequisites

Required Tools

- **Python 3.10.12+** with pip
- **CMake 3.26+**
- **Visual Studio 2019** or **Visual Studio 2022**
- **Conan 2.0.17** (recommended) or external dependencies

Optional Tools

- Git (for source control)
- 7-Zip or similar (for extracting archives)

Build Methods

Method 1: Using Conan (Recommended)

Conan handles all dependency management automatically.

1. **Install Conan**:

   ```
   pip install conan==2.0.17
   ```

2. **Build using Python script**:

   ```
   cd scripts
   python build_ofiq.py
   ```

3. **Alternative: Use batch script**:

   ```
   cd scripts
   build.cmd
   ```

Method 2: Building from Source

This method requires downloading external dependencies manually.

1. **Download external dependencies**:

   - Download the full OFIQ release from the ISO portal
   - Extract to the `extern/` directory

2. **Build using Python script**:

```
cd scripts
python build_ofiq.py --no-conan
```

Method 3: Using Available Dependencies (Experimental)

This method attempts to build with whatever dependencies are available. Note that this may not work for all functionality.

1. **Check available dependencies**:

```
cd scripts
python build_ofiq.py --no-conan --no-download --skip-deps
```

2. **If dependencies are missing, the script will show what's available and what's missing**

3. **For minimal builds, you can try with available dependencies only**:

```
cd scripts
python build_ofiq.py --no-conan --no-download --skip-deps
```

Note: This approach requires that all necessary dependencies are already available in the `extern/` directory. If dependencies are missing, the build will fail.

Build Options

Python Script Options

```
python build_ofiq.py [options]
```

Option	Description	Default
`--arch x64|x86`	Target architecture	`x64`
`--compiler 16|17`	Visual Studio version (16=2019, 17=2022)	`16`
`--debug`	Build Debug configuration	Release
`--no-conan`	Build dependencies from source	Use Conan
`--no-download`	Skip downloading external files	Download

Examples

```
# Default build (x64, Release, VS2019 with Conan)
python build_ofiq.py

# Build 32-bit version
python build_ofiq.py --arch x86

# Use Visual Studio 2022
python build_ofiq.py --compiler 17

# Build Debug configuration
python build_ofiq.py --debug

# Build without Conan (requires external dependencies)
python build_ofiq.py --no-conan

# Build without downloading (requires pre-downloaded dependencies)
python build_ofiq.py --no-conan --no-download
```

Output Files

After successful build, you'll find:

- `ofiq_lib.dll` - Main library file
- `OFIQSampleApp.exe` - Sample application
- **Header files** - In `install_x86_64/Release/include/`
- **Model files** - In `data/models/` (downloaded automatically)

Troubleshooting

Common Issues

1. **Missing Conan**:

   ```
   Error: Missing required tools: conan
   ```

   Solution: Install Conan: `pip install conan==2.0.17`

2. **Missing external dependencies**:

   ```
   OpenCV source not found at: ...\extern\opencv-4.5.5
   ```

   Solution: Download external dependencies or use Conan

3. **CMake not found**:

```
Error: Missing required tools: cmake
```

Solution: Install CMake 3.26+ from cmake.org

4. **Visual Studio not found**:

```
CMake Error: Could not create named generator Visual Studio 16 2019
```

Solution: Install Visual Studio 2019 or 2022

Dependency Locations

- **Conan profiles**: `conan/` directory
- **External dependencies**: `extern/` directory
- **Model files**: `data/models/` directory
- **Build output**: `build/` and `install_x86_64/` directories

Verification

After build, verify the output:

1. Check for `ofiq_lib.dll` in `install_x86_64/Debug/bin/` (Debug) or `install_x86_64/Release/bin/` (Release)
2. Run the sample application:

```
cd "install_x86_64/Debug/bin"
.\OFIQSampleApp.exe -c "../../../data/ofiq_config.jaxn" -i
"../../../data/tests/images/b-01-smile.png"
```

Expected Output: Quality assessment scores for the test image, including UnifiedQualityScore, BackgroundUniformity, IlluminationUniformity, and other metrics.

Integration with C# Wrapper

The built `ofiq_lib.dll` can be used with the C# wrapper:

1. Copy `ofiq_lib.dll` to your C# project output directory
2. Ensure the C# wrapper can find the native library
3. Use the `OFIQEngine` class from the C# wrapper

Support

For build issues:

1. Check the BUILD.md file
2. Verify all prerequisites are installed

3. Ensure sufficient disk space (build requires ~2GB)
4. Check internet connection for dependency downloads