Code Changes in NFIQ 2.1
...and Beyond

Greg Fiumara
greg@nist.gov | nfiq2@nist.gov

15 June 2021
EAB Workshop on Fingerprint Image Quality
Code Changes in NFIQ 2.1
...and Beyond

Greg Fiumara
greg@nist.gov | nfiq2@nist.gov

Ralph Lessmann (HID Global), Andrew Figlarz (UMD), ...
What’s New?

A lot.

Users
- Command-line interface
- Cross platform support
- Error messages/handling
- Multithreading

Developers
- Cross platform support
- Simplified, consistent API
- Build system
- Compliance test

Any mention of commercial products is for information only and does not imply recommendation or endorsement by NIST.
Command-line Interface

Currently implemented flags and their functions:

- **-i** [file/directory path]: Processing provided Images and Directories
- **-f** [batch file path]: Processing provided batch files
- **-o** [file path]: Saving all output to a specified file
- **-j** [# of threads]: Enables Multi-Threading for Batch and RecordStore processes
- **-m** [model info file]: Path to alternate model info file
- **-a**: Displays actionable quality scores about each processed image
- **-v**: Displays individual quality score information about each processed image
- **-d**: Displays additional debug information as each score gets computed
- **-f**: Forces computation to occur. Tool does not prompt user with yes/no options
- **-r**: Recursive file scanning if a directory is provided

**Version Info**

Biometric Evaluation: 10.0
FingerJet: 6.2.1
OpenCV: 4.5.2
NFIQ 2: 2.1.0-pre.1 (Date: 202105211557, Commit: 7f1efad)

```bash
$ nfiq2
Welcome to NFIQ2
Please provide individual files or directories to the command line.
If you would like to use one of the flags listed below, please include them before any standard arguments.
```

```
NFIQ2(1) National Institute of Standards and Technology NFIQ2(1)

NAME
nfiq2 -- Compute quality of fingerprint images

SYNOPSIS
nfiq2 OPTIONS file/dir/rs [...]
nfiq2 -f batch [-f ...] OPTIONS [file/dir/rs ...]

DESCRIPTION
nfiq2 is a tool for computing the NIST Fingerprint Image Quality (NFIQ) 2 of fingerprint images. It is used to produce quality scores for fingerprint images encoded in formats supported by Biometric Evaluation framework (BMP, JPEG, JPEGOL, JPEG-2000, PBM, PNG, TIFF, and WSQ). Images can be encoded stand alone, within an ANSI/NIST ITL Type 14 record, or any combination of the two within a Biometric Evaluation Record Store. nfiq2 serves as a reference implementation of ISO/IEC 29794-4:2017.

nfiq2 Prints scores in a Comma Separated Value (CSV) format. The default output includes information about the image including: the image’s name, finger position (if applicable), quality score, error mes-
```
Command-line Interface

NFIQ2 SINGLE finger.wsq WSQ false false
• Codecs **decoupled** from library
  - Developer note: library *only* supports raw pixels
• BMP, JPEG, JPEG 2000, PBM, PNG, TIFF, WSQ
• ISO/IEC 19794-4
• Binary EBTS and ANSI/NIST-ITL: Type 14

• Directories, batch files

• Uses Biometric Evaluation Framework*
  
*CAVEAT on lossy image compression
Resolution | CLI

500 PPI only, and enforced

• “What about 1000 PPI?”
  • Integration with NIST Fingerprint Image Resampler (NFIR), but caveat emptor.

• “DEVICE doesn’t encode resolution in its images!”
  • “No resolution” images detected
  • –F flag to force 500 PPI
• Imposed by FingerJet FX OSE
  • $196 < \text{width} < 800$
  • $196 < \text{height} < 1000$

• Zero minutia bug in 2.0 corrected

• Dimensions after whitespace crop
  • “The first step in the segmentation is to remove areas of the input image which are near constant.”
Consistent Output | CLI

• Single image ➔ single number

• CSV Output
  • –v: Show quality feature values
  • –q: Show quality module speed
  • –a: Show “actionable feedback”
    • Remains undocumented and experimental

• Error messages, NA
Pre-compiled:
• Command-line interface application
• Development library/headers

• Platforms
  • CentOS 7, 8 (.rpm)
  • macOS 10.15 (.pkg)
  • macOS 11 (.pkg)
    • Intel, M1
  • Windows 10 (.msi)
    • 32-bit, 64-bit
  • Ubuntu 20.04 LTS (.deb)

https://github.com/usnistgov/NFIQ2/releases
Naming and Versioning

<table>
<thead>
<tr>
<th>NFIQ 2</th>
<th>Quality scoring algorithms from ISO/IEC 29794-4:2017</th>
</tr>
</thead>
<tbody>
<tr>
<td>NFIQ 2.0</td>
<td>NIST software (CLI + lib) version, 2016</td>
</tr>
<tr>
<td>NFIQ 2.1-pre</td>
<td>NIST pre-release software (CLI + lib) version, Jan 2021</td>
</tr>
<tr>
<td>NFIQ 2.1-pre.1</td>
<td>NIST pre-release software (CLI + lib) version, May 2021</td>
</tr>
<tr>
<td>NFIQ 2.1</td>
<td>NIST software (CLI + lib) version, June 2021</td>
</tr>
</tbody>
</table>

- Bug fixes and performance improvements: *patch*
  - NFIQ 2.1.1, 2.1.2, 2.1.#
- Scoring changes†, library API: *minor*
  - NFIQ 2.2, 2.3, 2.#
  - Developers: constants in NFIQ2::Version namespace

For simplicity, CLI and library will maintain identical versions.

†Hopefully never again!
Changes for Developers

Warning: code samples ahead!
Score Stability 1

- Standardized on Latest OpenCV (4.5.x)
- Disabled OpenCV Features + Optimizations
  - Start with blank slate
    - No Intel TBB
    - No Intel IPP
    - No Graph API
    - No OpenMP
    - No OpenCL

GitHub Issue #265: *Can we enable some or all of these for better performance?*
Algorithm 7: rvu algorithm

Input: Fingerprint image I
Output: rvu quality score $Q_{rvu}$

for each block $V$ in I do
  1. Determine dominant ridgeflow orientation $\theta(V)$
  2. Rotate $V$ such that $\theta(V)$ is perpendicular to x-axis
  3. Crop $V$ such that no invalid regions are included
  4. With $V$ obtain the ridge-valley signature $T$ (eq. (11))
  5. Determine $DT$ using linear regression on $T$
  6. For each $T(x)$ compute threshold $P(x) = DT(1) \ast DT(0)$ Binarize $T$ using $P$
  7. Classify ridge and valley in $T$ as $S(x) = \begin{cases} 1, & \text{if } T(x) < P(x) \\ 0, & \text{otherwise} \end{cases}$
  8. Compute ridge-valley transition vector as $C(x) = \begin{cases} 1, & \text{if } S(x - 1) \neq S(x) \\ 0, & \text{otherwise} \end{cases}$
  9. Drop first and last transition from $T$ using $C$ to remove incomplete ridges or valleys and obtain $T'$
  10. Compute $Q_{RVU}$ as the ratio between widths of ridge and valleys in $T'$
end

Figure 7: Processing steps of Ridge Valley Uniformity quality algorithm. (a) crop of current block; (b) average profile of block; (c) average profile with regression line; (d) local quality score as the standard deviation of local ridge to valley ratios.

Source: NFIQ 2 Report (Draft), Tabassi, NIST.
2.0: What is the NFIQ 2 score for this image?

```cpp
std::list<NFIQ::ActionableQualityFeedback> actionableQuality{};
std::list<NFIQ::QualityFeatureData> features{};
std::list<NFIQ::QualityFeatureSpeed> speeds{};

FingerprintImageData image{ /* ... */ };

NFIQ::NFIQ2Algorithm algorithm{};
const unsigned int qualityScore = algorithm.computeQualityScore(image,
    false, actionableQuality,
    false, features,
    false, speeds);
```
2.0: What is the NFIQ 2 score for this image?

```cpp
FingerprintImageData image{ /* ... */ };  
NFIQ::NFIQ2Algorithm algorithm{};  
const unsigned int qualityScore = algorithm.computeQualityScore(image,
false, actionableQuality,
false, features,
false, speeds);
```
2.1: What is the NFIQ 2 score for this image?

```cpp
const NFIQ2::FingerprintImageData image{ /* ... */ };
const NFIQ2::Algorithm algorithm{ /* ... */ };

const unsigned int qualityScore = algorithm.computeQualityScore(image);
```

OpenCV installation is not required.
2.1: Feature Vectors

```cpp
const NFIQ2::FingerprintImageData image{ /* ... */ };
const auto features =
    NFIQ2::QualityFeatures::computeQualityFeatures(image);
```
2.1: Feature Vectors

```cpp
const NFIQ2::FingerprintImageData image{ /* ... */ };
const auto features =
    NFIQ2::QualityFeatures::computeQualityFeatures(image);

const NFIQ2::Algorithm algorithm{ /* ... */ };
const unsigned int qualityScore =
    algorithm.computeQualityScore(features);
```
2.1: Obtain a Specific Feature

```cpp
const NFIQ2::FingerprintImageData image{ /* ... */ };  
const auto features =  
    NFIQ2::QualityFeatures::computeQualityFeatures(image);  

std::cout << "Number of minutiae: " <<  
    features[NFIQ2::Identifiers::QualityFeatures::Minutiae::Count] << '\n';
```
2.1: Obtain a Specific Feature

std::unordered_map<std::string, double> const NFIQ2::FingerprintImageData image{ /* ... */ };
const auto features =
    NFIQ2::QualityFeatures::computeQualityFeatures(image);

std::cout << "Number of minutiae: " <<
    features[NFIQ2::Identifiers::QualityFeatures::Minutiae::Count] << '\n';

Constants for modules, features, ...
/* No model (or embedded model, compilation dependent) */
NFIQ2::Algorithm algorithm{};

/* Instantiate from ModelInfo */
algorithm = NFIQ2::Algorithm{NFIQ2::ModelInfo{
   "/usr/local/nfiq2/share/nist_plain_tir-ink.txt"}};
std::optional<unsigned int> qualityScore{};
try {
    qualityScore = algorithm.computeQualityScore(image);
} catch (const NFIQ2::Exception &e) {
    std::cout << "{ERROR}: " << e.what() << '\n';
}
Reduced Dependencies | API

No more:
• NBIS
• BiomDI

Build-only dependencies:
• FingerJet FX OSE
• OpenCV

We distribute a regression-tested development package. 

*Please use it!*
Public API fully documented and browsable.

https://pages.nist.gov/NFIQ2/docs
Compliance Test

• **New** compliance test
  • 200+ images sampled from NIST Special Databases 300 and 302
  • Exact values **required** for all operating systems
  • Quality scores 0-97 + errors represented
    • Optical TIR and ink scan for each value, where possible
  • Download: https://nigos.nist.gov/datasets/nfiq2_compliance

• GitHub releases run compliance test *and* 
  ≅2m sequestered regression test
• GitHub commits **run compliance**
  • Ubuntu 18, 20
  • Windows 2019 (32/64-bit)
  • macOS 10.15 (Intel)
Future | API

- C API
  - Facilitating easier Python, C#, Java, ... integration
- \( \geq \text{C++17} \)?
- CLI: Native image codecs
  - Beware of lossy compression
- Continued API and build system improvements
  - No guarantees about breaking code

- CLI: Check for updates
- Officially support Android, iOS, MinGW, ...
- CMake `find_package` support
- Shared object build
- Code optimization
- Package manager support
- Code signing

Want to help? https://github.com/usnistgov/NFIQ2/issues and email nfiq2@nist.gov