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

National Institute of Standards and Technology
U.S. Department of Commerce
INFORMATION TECHNOLOGY LABORATORY

Code Changes in NFIQ 2.1

…and Beyond

Greg Fiumara, Ralph Lessmann (HID Global), Andrew Figlarz (UMD), …
greg@nist.gov | nfiq2@nist.gov

15 June 2021
EAB Workshop on Fingerprint Image Quality

What’s New?

A lot.

Users

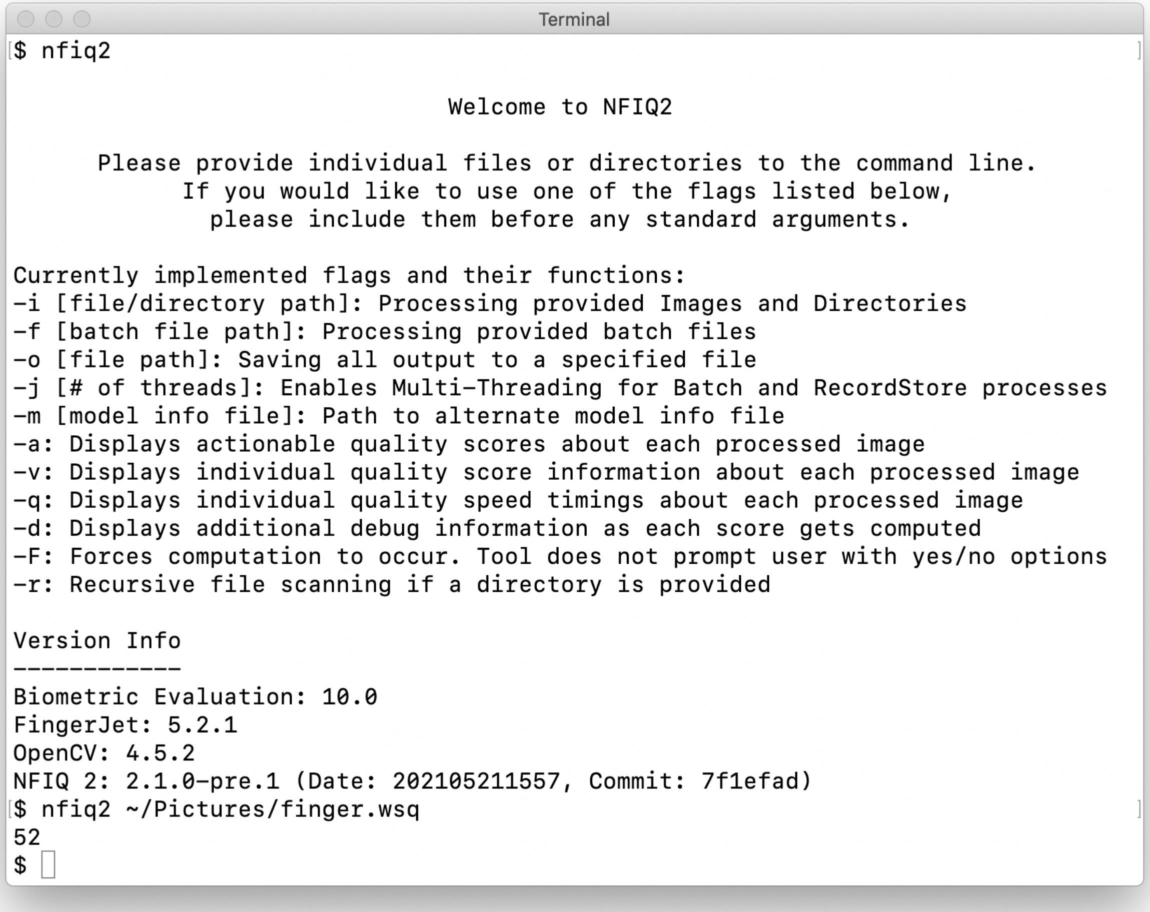
- Command-line interface
- Cross platform support
- Error messages/handling
- Multithreading

Developers

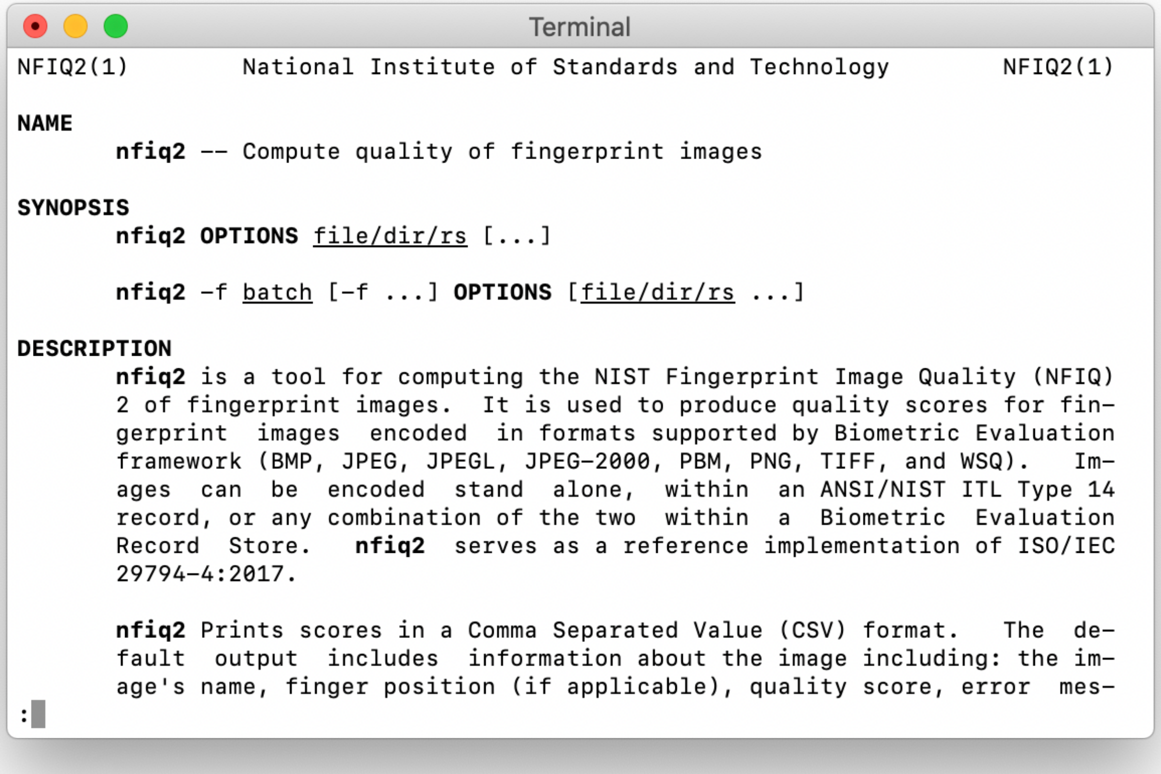
- Cross platform support
- Simplified, consistent API
- Build system
- Compliance test

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

Command-line Interface

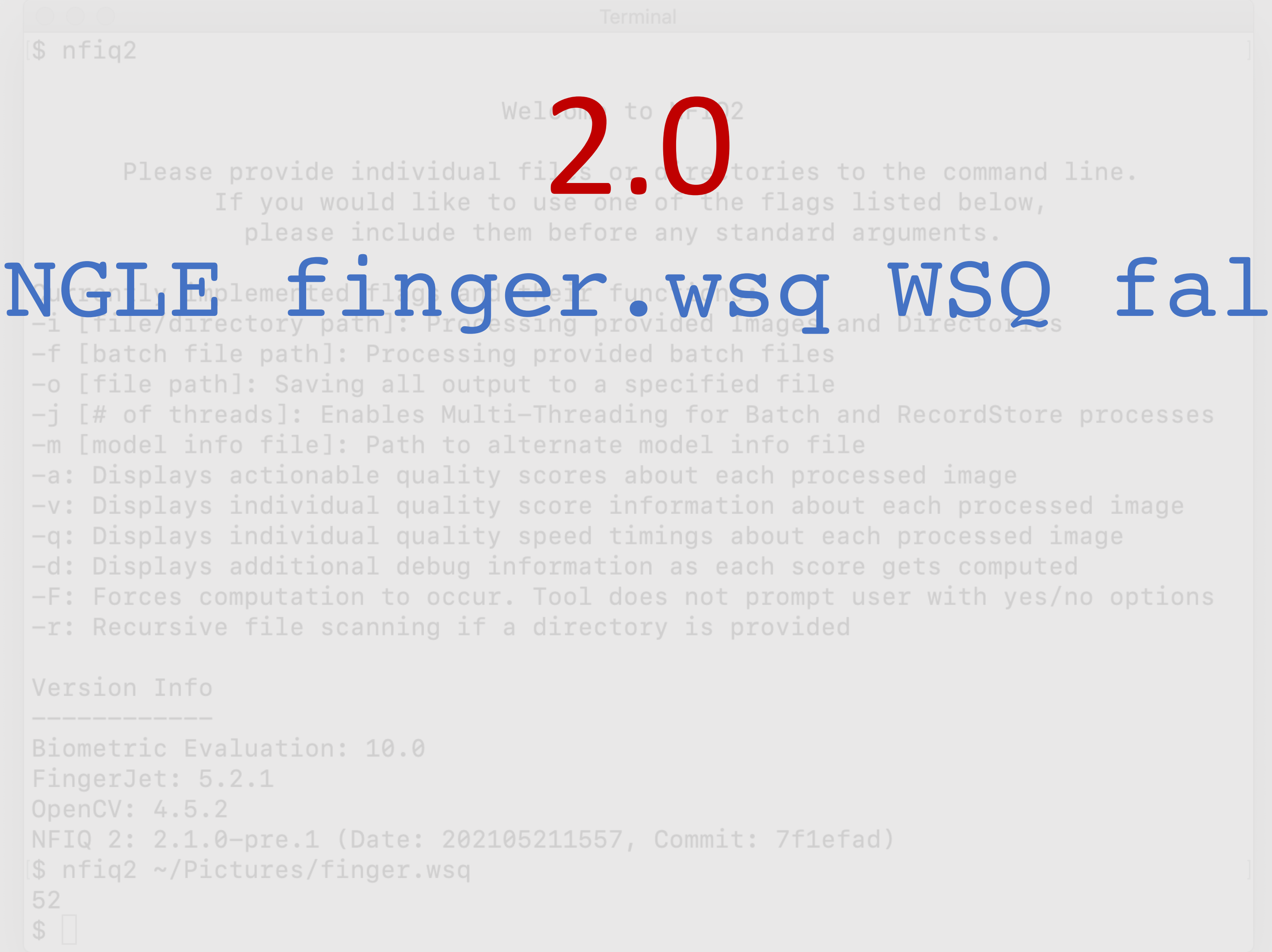


```
$ nfiq2

                               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.

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
-q: Displays individual quality speed timings 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: 5.2.1
OpenCV: 4.5.2
NFIQ 2: 2.1.0-pre.1 (Date: 202105211557, Commit: 7f1efad)
$ nfiq2 ~/Pictures/finger.wsq
52
$
```



```
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 fin-
       gerprint  images  encoded  in formats supported by Biometric Evaluation
       framework (BMP, JPEG, JPEGL, JPEG-2000, PBM, PNG, TIFF, and WSQ).   Im-
       ages  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  de-
       fault  output  includes  information about the image including: the im-
       age's name, finger position (if applicable), quality score, error  mes-
:
```

Command-line Interface



```
$ 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.

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
-q: Displays individual quality speed timings 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: 5.2.1
OpenCV: 4.5.2
NFIQ 2: 2.1.0-pre.1 (Date: 202105211557, Commit: 7f1efad)
$ nfiq2 ~/Pictures/finger.wsq
52
$ 
```

2.0

```
NFIQ2 SINGLE finger.wsq WSQ false false
```

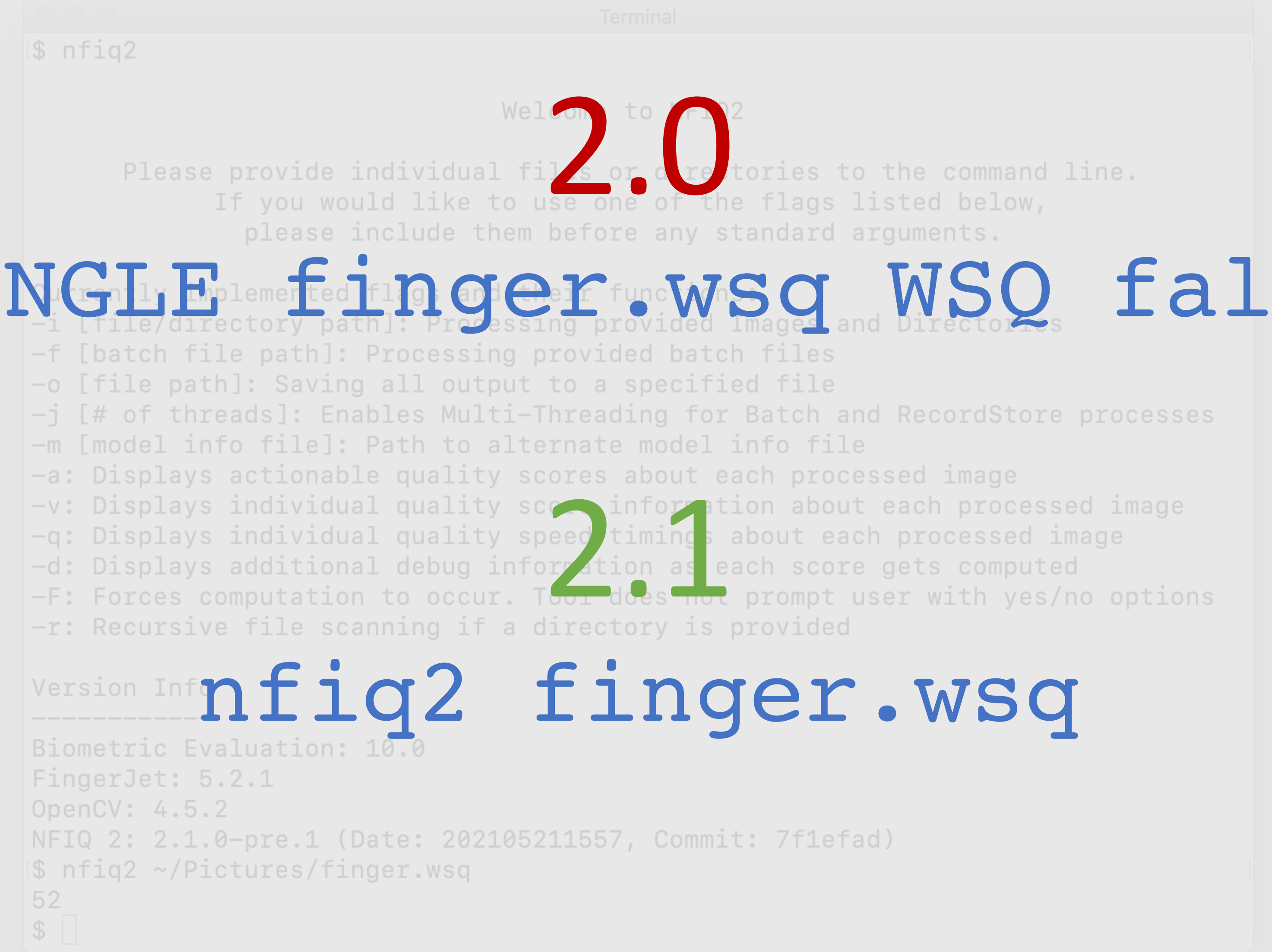
Command-line Interface

2.0

```
NFIQ2 SINGLE finger.wsq WSQ false false
```

2.1

```
nfiq2 finger.wsq
```



```
$ 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.

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
-q: Displays individual quality speed timings 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: 5.2.1
OpenCV: 4.5.2
NFIQ 2: 2.1.0-pre.1 (Date: 202105211557, Commit: 7f1efad)
$ nfiq2 ~/Pictures/finger.wsq
52
$ 
```

Input | CLI

- 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

Dimensions | CLI

- Imposed by FingerJet FX OSE
  - 196 < width < 800
  - 196 < 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`

Downloads | CLI

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

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

- 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!

This Photo by Unknown Author is licensed under CC BY

Score Stability 1

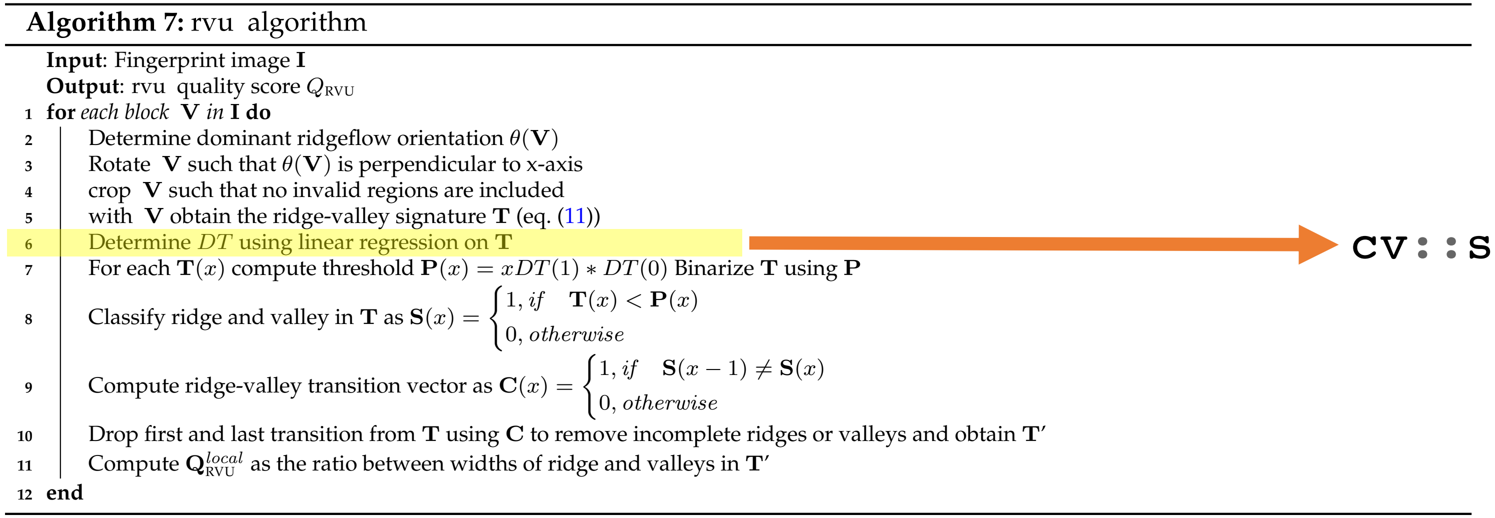
- Standardized on Latest OpenCV (4.5.x)

- 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?

Score Stability 2



Algorithm 7: rvu algorithm

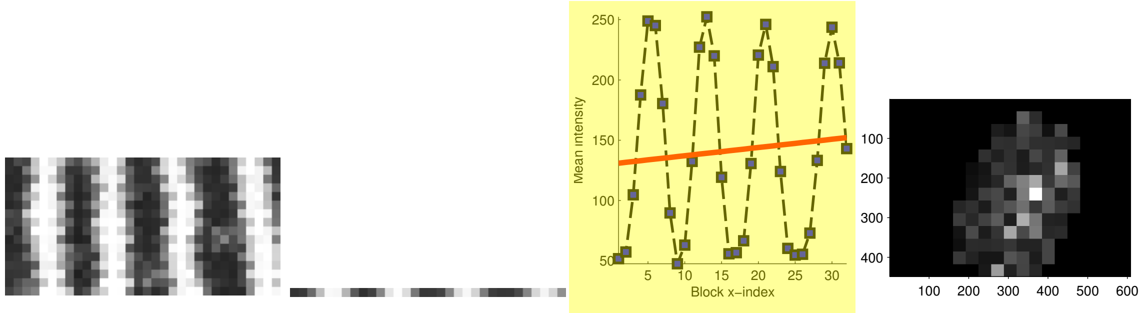
```
Input: Fingerprint image $\mathbf{I}$
Output: rvu quality score $Q_{\text{RVU}}$
for each block $\mathbf{V}$ in $\mathbf{I}$ do
    Determine dominant ridgeflow orientation $\theta(\mathbf{V})$
    Rotate $\mathbf{V}$ such that $\theta(\mathbf{V})$ is perpendicular to x-axis
    crop $\mathbf{V}$ such that no invalid regions are included
    with $\mathbf{V}$ obtain the ridge-valley signature $\mathbf{T}$ (eq. (11))
    Determine $DT$ using linear regression on $\mathbf{T}$
    For each $\mathbf{T}(x)$ compute threshold $\mathbf{P}(x) = xDT(1) * DT(0)$ Binarize $\mathbf{T}$ using $\mathbf{P}$
    Classify ridge and valley in $\mathbf{T}$ as $\mathbf{S}(x) = \begin{cases} 1, \textit{if} & \mathbf{T}(x) < \mathbf{P}(x) \\ 0, \textit{otherwise} \end{cases}$
    Compute ridge-valley transition vector as $\mathbf{C}(x) = \begin{cases} 1, \textit{if} & \mathbf{S}(x-1) \neq \mathbf{S}(x) \\ 0, \textit{otherwise} \end{cases}$
    Drop first and last transition from $\mathbf{T}$ using $\mathbf{C}$ to remove incomplete ridges or valleys and obtain $\mathbf{T}'$
    Compute $\mathbf{Q}_{\text{RVU}}^{local}$ as the ratio between widths of ridge and valleys in $\mathbf{T}'$
end
```

```
cv::solve( ..., T, DT, ... );
```

6.126157964974621864831796187463473478...

Digits inconsistent across architecture + OS. Truncate.

6.1261579649



(a) (b) (c) (d)

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.

Simplification: Round 1 | API

2.0: What is the NFIQ 2 score for this image?

```
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);
```

Simplification: Round 1 | API

2.0: What is the NFIQ 2 score for this image?

```
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);
```

Declared but unused

Why list?

Global namespace

RF parameters embedded

Member method

Copied + not const

NFIQ 1 or NFIQ 2?

Not const

Duplicative naming

C-style optional out parameters

Simplification: Round 1 | API

2.1: What is the NFIQ 2 score for this image?

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

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

Still a member method.
More on this later.

OpenCV installation is not required.

Simplification: Round 2 | API

2.1: Feature Vectors

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

Simplification: Round 2 | API

2.1: Feature Vectors

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

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

Simplification: Round 3 | API

2.1: Obtain a Specific Feature

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

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

Simplification: Round 3 | API

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, …

Model Swap | API

```
/* 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"}};
```

See presentation
*From Excitement
to Practice* by
Ralph Lessmann

Error Handling | API

```
std::optional<unsigned int> qualityScore{};
try {
        qualityScore = algorithm.computeQualityScore(image);
} catch (const NFIQ2::Exception &e) {
        std::cout << "[ERROR]: " << e.what() << '\n';
}
```

Reduced Dependencies | API

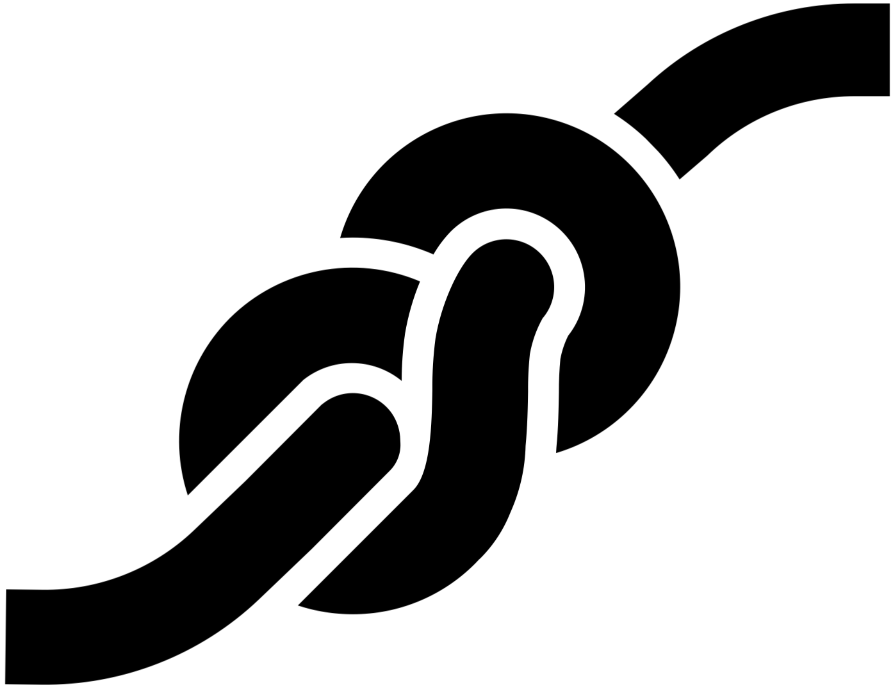
No more:

- NBIS
- BiomDI

Build-only dependencies:

- FingerJet FX OSE
- OpenCV

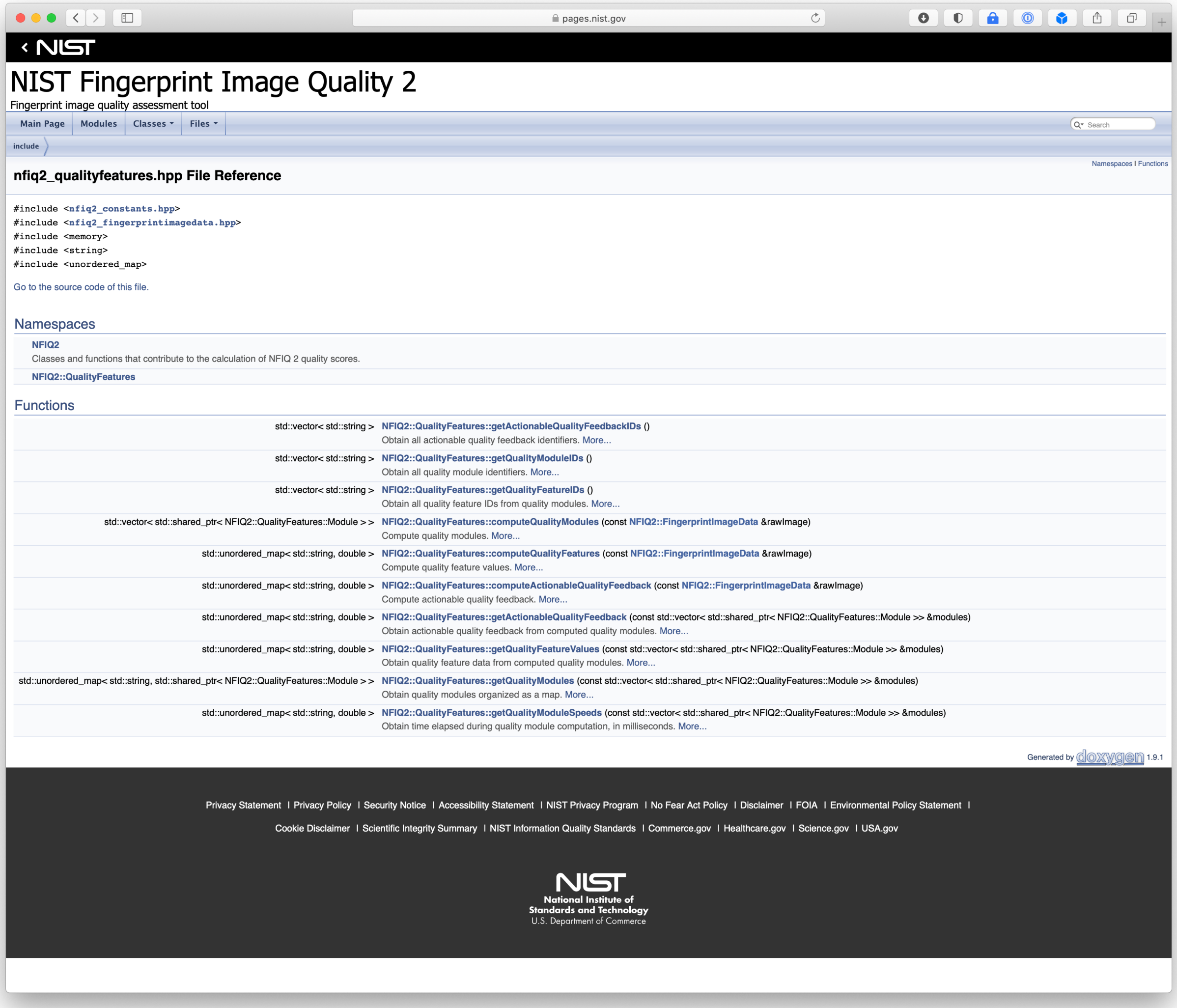
We distribute a regression-tested development package.
Please use it!



Documentation | API

Public API fully documented and browsable.

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



NIST Fingerprint Image Quality 2

Fingerprint image quality assessment tool

nfiq2_qualityfeatures.hpp File Reference

```
#include <nfiq2_constants.hpp>
#include <nfiq2_fingerprintimagedata.hpp>
#include <memory>
#include <string>
#include <unordered_map>
```

Go to the source code of this file.

Namespaces

NFIQ2
Classes and functions that contribute to the calculation of NFIQ 2 quality scores.

NFIQ2::QualityFeatures

Functions

std::vector< std::string >	NFIQ2::QualityFeatures::getActionableQualityFeedbackIDs () Obtain all actionable quality feedback identifiers. More...
std::vector< std::string >	NFIQ2::QualityFeatures::getQualityModuleIDs () Obtain all quality module identifiers. More...
std::vector< std::string >	NFIQ2::QualityFeatures::getQualityFeatureIDs () Obtain all quality feature IDs from quality modules. More...
std::vector< std::shared_ptr< NFIQ2::QualityFeatures::Module > >	NFIQ2::QualityFeatures::computeQualityModules (const NFIQ2::FingerprintImageData &rawImage) Compute quality modules. More...
std::unordered_map< std::string, double >	NFIQ2::QualityFeatures::computeQualityFeatures (const NFIQ2::FingerprintImageData &rawImage) Compute quality feature values. More...
std::unordered_map< std::string, double >	NFIQ2::QualityFeatures::computeActionableQualityFeedback (const NFIQ2::FingerprintImageData &rawImage) Compute actionable quality feedback. More...
std::unordered_map< std::string, double >	NFIQ2::QualityFeatures::getActionableQualityFeedback (const std::vector< std::shared_ptr< NFIQ2::QualityFeatures::Module >> &modules) Obtain actionable quality feedback from computed quality modules. More...
std::unordered_map< std::string, double >	NFIQ2::QualityFeatures::getQualityFeatureValues (const std::vector< std::shared_ptr< NFIQ2::QualityFeatures::Module >> &modules) Obtain quality feature data from computed quality modules. More...
std::unordered_map< std::string, std::shared_ptr< NFIQ2::QualityFeatures::Module > >	NFIQ2::QualityFeatures::getQualityModules (const std::vector< std::shared_ptr< NFIQ2::QualityFeatures::Module >> &modules) Obtain quality modules organized as a map. More...
std::unordered_map< std::string, double >	NFIQ2::QualityFeatures::getQualityModuleSpeeds (const std::vector< std::shared_ptr< NFIQ2::QualityFeatures::Module >> &modules) Obtain time elapsed during quality module computation, in milliseconds. More...

Generated by doxygen 1.9.1

Privacy Statement | Privacy Policy | Security Notice | Accessibility Statement | NIST Privacy Program | No Fear Act Policy | Disclaimer | FOIA | Environmental Policy Statement |

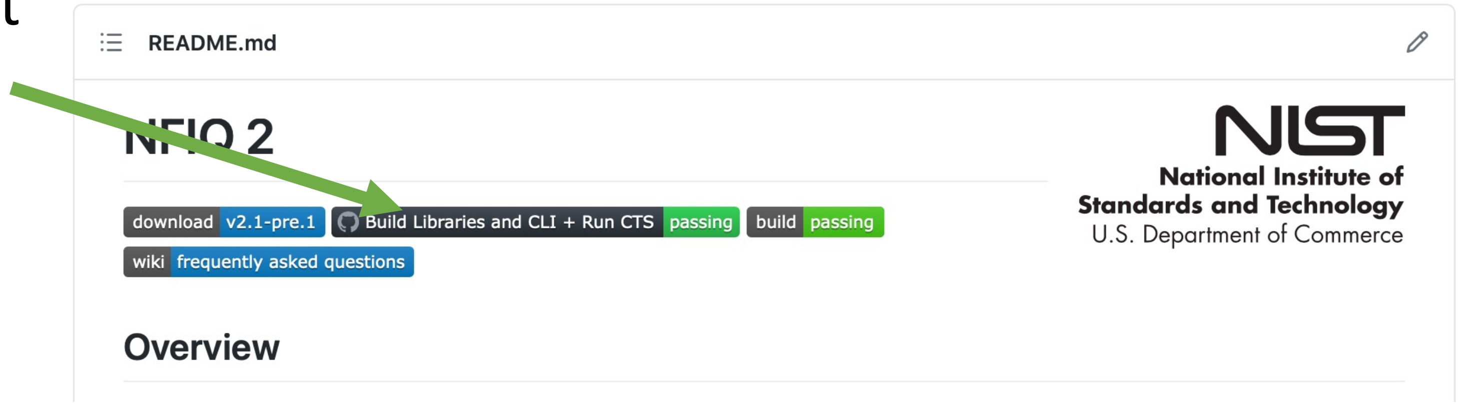
Cookie Disclaimer | Scientific Integrity Summary | NIST Information Quality Standards | Commerce.gov | Healthcare.gov | Science.gov | USA.gov

NIST
National Institute of
Standards and Technology
U.S. Department of Commerce

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)



README.md

NFIQ 2

download v2.1-pre.1 | Build Libraries and CLI + Run CTS passing | build passing

wiki frequently asked questions

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Overview

Future | API

- C API
  - Facilitating easier Python, C#, Java, … integration
- ≥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