MTR 070053

MITRE TECHNICAL REPORT

IQF (IMAGE QUALITY OF FINGERPRINT) SOFTWARE APPLICATION

May 2007

Norman B. Nill

Sponsor: DOJ/FBI
Dept. No.: G036
Contract No.: W15P7T-07-C-F600
Project No.: 0707E02X

MITRE
Center for Integrated Intelligence Systems
Bedford, Massachusetts

(This page intentionally left blank)

Abstract

Knowledge of the image quality of a fingerprint is important and useful information in a variety of applications; it can, for example, help determine the probability of successful matching, alert an operator to a poor quality enrollment, or aid problem diagnostics/performance assessment of capture devices. This document describes a quantitative measure of image quality (developed by the author) which is tailored to measure the visual quality of a digital fingerprint image. This Image Quality of Fingerprint (IQF) metric is available as a freeware software application.

Table of Contents

List of Figures

1 Introduction

The Image Quality of Fingerprint (IQF) software application is tailored to measure the visual quality of a digital fingerprint image, i.e., the apparent quality of the softcopy displayed image presented to a human observer who is knowledgeable in fingerprint assessment[1].

IQF can find uses in such areas as:

- real time quality feedback during fingerprint capture (weed-out, reject low quality captures)
- diagnosis of fingerprint capture system problems related to quality
- indicating prospects for successful matching of latent fingerprint with a stored inked/livescan fingerprint (e.g. in latent workstation)
- archiving/cataloging fingerprints using quality as a parameter
- assessment of fingerprint images submitted for FBI certifications of fingerprint capture devices [PIV], [AppF].

IQF is a fingerprint-tailored version of MITRE's general purpose Image Quality Measure (IQM) [Nill & Bouzas], which in turn is based on earlier work by the author [Nill-1]. IQF and IQM compute image quality based on the two-dimensional, spatial frequency power spectrum of the digital image. The power spectrum, which is the square of the magnitude of the Fourier transform of the image, contains information on the sharpness, contrast, and detail rendition of the image and these are components of visual image quality. In IQF, the power spectrum is normalized by image contrast, average gray level (brightness), and image size; a visual response function filter is applied, and the pixels per inch (ppi) resolution scale of the fingerprint image is taken into account[2]. The fundamental output of IQF is a single-number image quality value which is the sum of the filtered, scaled, weighted power spectrum values. The power spectrum normalizations allow valid intercomparisons between disparate fingerprint images.

---

[1] It is assumed that a good quality softcopy display is used, which is set-up for optimum viewing in terms of contrast, brightness, ambient room lighting, view distance, etc.

[2] IQF does not include the unneeded IQM components of directional scale computation, image degradation tests, computation of 1-D ring and wedge spectra, or specialized noise filtering.

2 IQF Processing

This section describes the processing steps that are illustrated in the IQF flow chart in Figure 2-1.

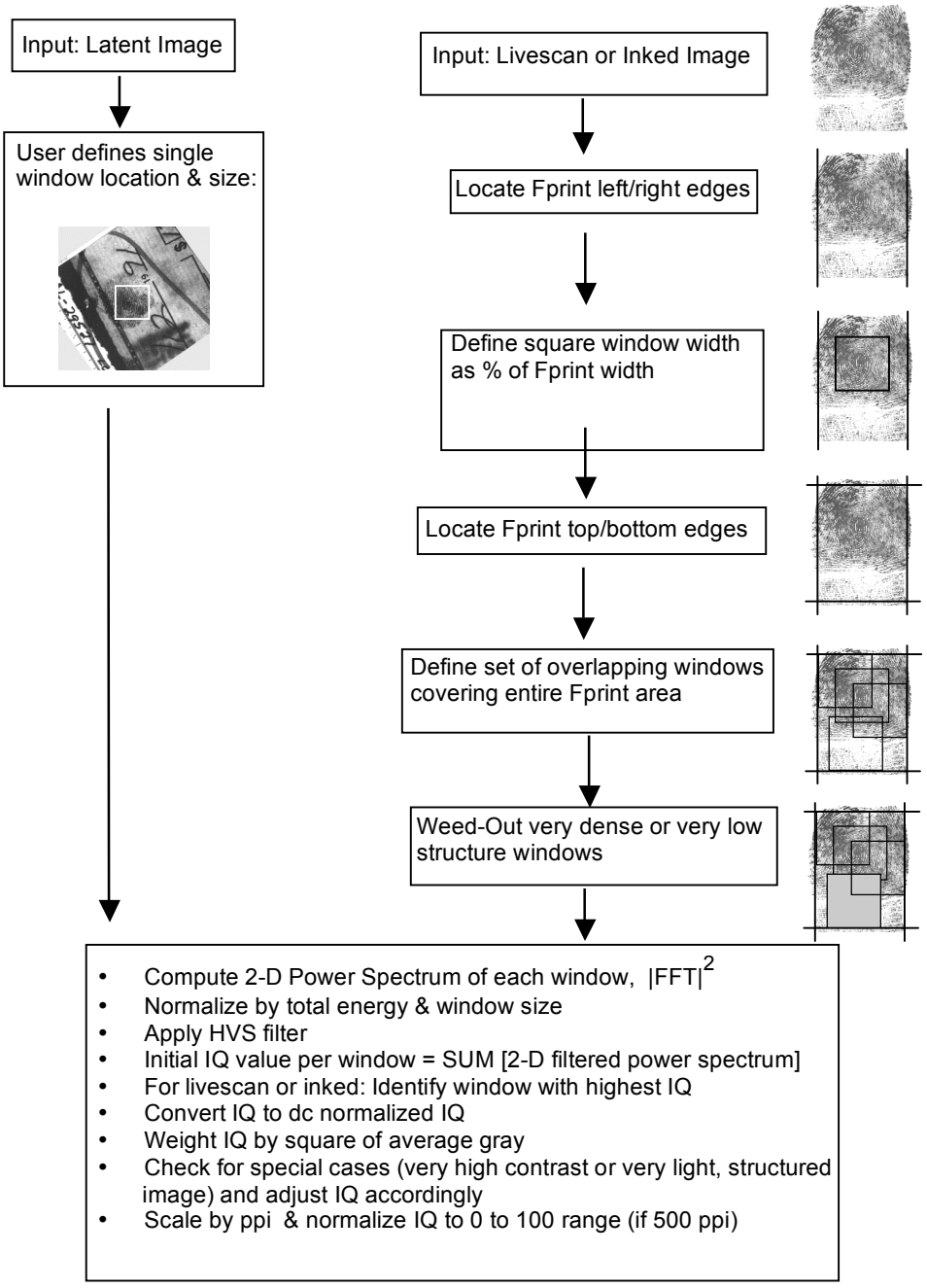


Figure 2-1. IQF Flow Chart

2.1 Pre-Processing to Establish Measurement Windows

2.1.1 Inked or Livescan Image

The digitized inked image or digital livescan image is first read-in, either as a raw image or in PGM format. IQF assumes the image contains only one fingerprint[3], which is upright or slanted upright (not horizontal), is 8 bits per pixel (bpp) grayscale, and has the polarity: dark gray ridges with light gray valleys.

IQF first locates the approximate left and right vertical edges of the fingerprint, using comparisons of the average values of narrow vertical strips formed across the image width. The correct left and right edges are easily located for a livescan with a plain white background, whereas an inked fingerprint, which may contain printblock lines, text, and a noisy background, is more problematic. The technique generally works on inked images, but if the algorithm believes it cannot find reasonable left and right edge locations, then it defines the entire image width (minus a safety margin) as the fingerprint width. Note that the program does not actually identify a region as a fingerprint, i.e., ridges, bifurcations, cores, or other fingerprint minutia are neither detected nor recognized as a fingerprint, per se.

A fingerprint window is defined as a certain percentage of the computed fingerprint width; this defines, for the given image, a constant-size square measurement window in which the quality value will be computed. Testing has indicated that a window width that is 60% of the fingerprint width works well, i.e., it includes a substantial enough part of the fingerprint to compute a quality value indicative of the fingerprint, while being small enough to (usually) avoid the quality computation being swamped by smudged areas, low structure areas, etc[4]. However, there is built-in logic to adjust the width if certain conditions exist, e.g., the window width can never be more than 0.8 inches nor less than 0.4 inches (changeable in GlobalData)[5]. The window width is set equal to the minimum value (0.4 inches) if the computed fingerprint width is suspect. Also, a small adjustment of a few pixels is made to the window width if it happens to fall on one of the program-catalogued long-compute widths associated with the specific FFT algorithm used.

Next, the top and bottom horizontal edges of the fingerprint are found, in a manner similar to that used for the left/right vertical edges. The fingerprint height is clamped to a maximum of 1.4 inches to avoid potentially extraneous areas, such as pronounced finger

---

[3] The program will still run if the inked/livescan image contains more than one fingerprint, but in such a case there is no assurance that the computed quality value will correspond to the highest quality fingerprint area in the image.

[4] When actual parameter values are given in this document they are with respect to IQF version 1.0.

[5] Conversion: inches = pixels / ppi

creases. At this stage the fingerprint area is defined by a rectangular box formed by the computed left, right, top and bottom edges, and the fingerprint window is a smaller square lying within this rectangle.

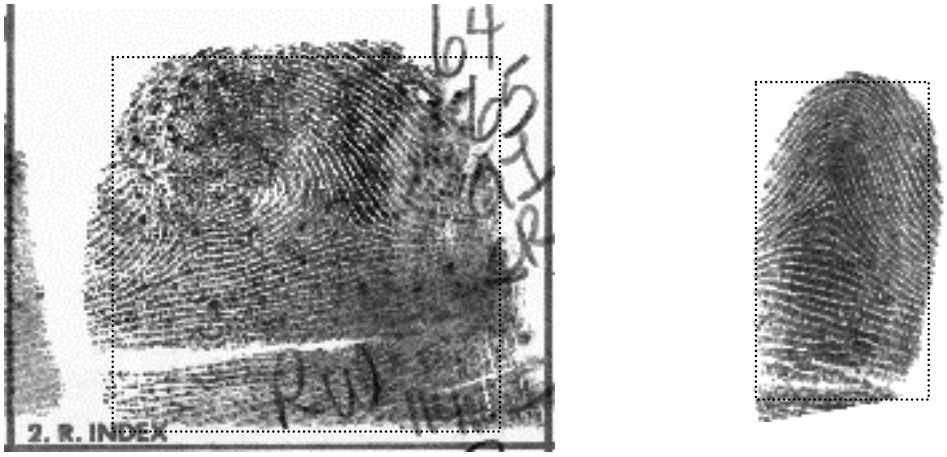
Next, the square fingerprint window is marched across and down the fingerprint rectangle in small (0.1 inch), overlapping increments and the coordinates of each window location are recorded. Larger increments will produce fewer windows and result in faster processing time, but fewer windows will result in a rougher estimate of the best quality area in the given fingerprint. Increments less than 0.1 inch seldom have any benefit while increments greater than about 0.2 inches will often result in noticeably lower quality values.

As an example, a 500 ppi rolled fingerprint might be defined by a 457 pixel wide by 631 pixel high rectangle, in which case the square fingerprint window would be 274 pixels wide (0.548 inches), and there would be 45 overlapping fingerprint windows within the rectangle area, each one covering 26% of the fingerprint area.

Some of the windows may have very little fingerprint structure or have substantial areas with very dense or opaque fingerprint content. These windows are identified and discarded from further consideration at this stage in the processing, as follows:
The fingerprint rectangle area is gridded into very small adjacent, non-overlapping square blocks, 0.08 by 0.08 inches each. The average and standard deviation is computed for each block and these values are used in a series of tests to determine if it is a very dense block or a very low structure block. The number of such unacceptable blocks in each fingerprint window is determined and if it exceeds a threshold value, then that fingerprint window is discarded. In the previous example this might cut the number of candidate windows from 45 to perhaps 30, which will avoid later problems in identifying the true peak image quality window from the group of windows, and will shorten the total processing time for that image.

Figure 2-2 illustrates 2 results for the IQF-determined fingerprint area rectangle: an inked fingerprint with a fairly noisy background and a livescan with a clean background. Note that the fingerprint area locator algorithm is smart enough to avoid printblock boundary lines near the perimeter of the image, but it has no way of detecting/avoiding text that overlays the fingerprint itself[6].

[6] A potential alternative approach might be to substitute a fingerprint core-finder algorithm for the current fingerprint rectangle algorithm; then construct a single measurement window centered around the core. However, this presupposes that the fingerprint image contains a clearly defined core and that the best quality part of the fingerprint is centered on the core.



Inked Roll Live Plain

Figure 2-2. IQF-Determined Fingerprint Rectangle (Dashed Line): Noisy Inked Image (Left); Clean Livescan Image (Right)

2.1.2 Latent Image

For a latent digital/digitized image, the user pre-defines the size and location of a single square window containing the fingerprint area of interest, instead of having the program define the multiple candidate windows described in section 2.1.1 for inked or livescan images. The difference in the two approaches is due to the assumption that the latent image contains a very substantial amount of extraneous structure, some of which may occlude part of the fingerprint. Under these conditions, IQF's algorithm for finding the fingerprint within the image cannot be relied upon.

Figure 2-3 illustrates a latent fingerprint with a typically very noisy background. When it is treated as an inked/live image, then the IQF fingerprint rectangle finder algorithm is invoked, which does a poor job of isolating the true fingerprint for measurement. In such a case the user needs to pre-define the true fingerprint location and associated square measurement area, which corresponds to a "latent image" run of IQF.

However, if a set of latent fingerprints are known to be relatively free of a noisy background, then time and effort can be saved by instructing IQF to treat them as inked/live images and let the program find the fingerprint area. Alternatively, the user may want to pre-define a specific size/location for a single square measurement window

on an image, regardless of whether it is an inked, livescan, or latent image; this can be done by telling the program that it is a latent image.

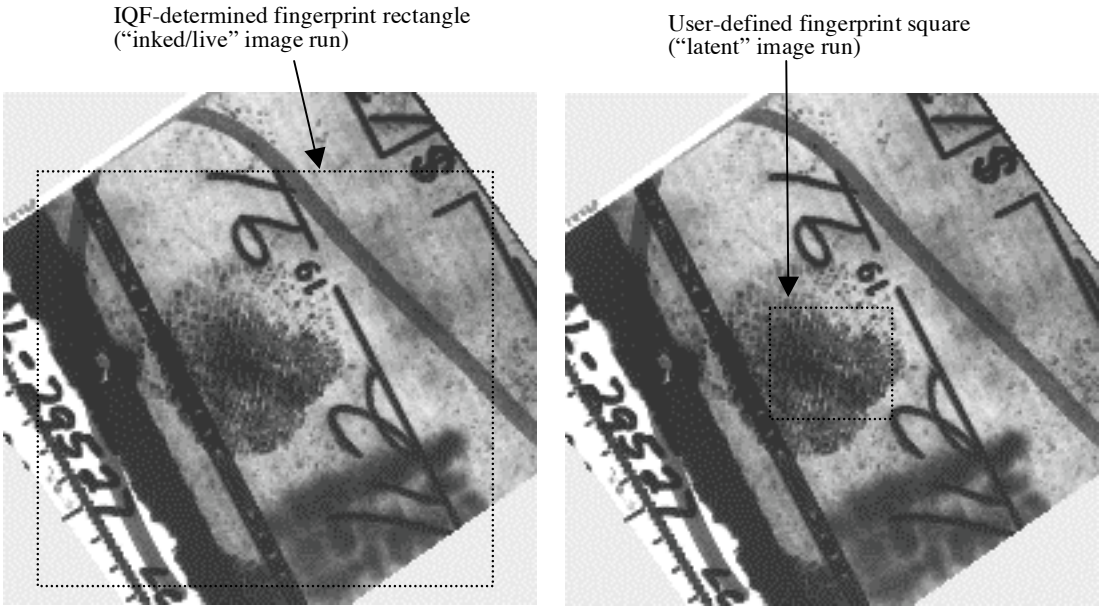


Figure 2-3. Fingerprint Rectangle Determined by IQF (Left) and by User (Right) (Latent Image with Very Noisy Background)

2.2 Image Quality Assessment Processing

Each measurement window is treated as an independent subimage to which the following processing steps are applied.

1) The two-dimensional (2-D) Fast Fourier Transform (FFT) of the subimage is computed, from which the subimage power spectrum is computed (squared modulus of FFT).

2) The power spectrum is normalized by the total power spectrum energy and by the subimage area. Total power normalization works best for later identifying the single window that has the best quality, within a given image. Subimage area normalization makes results obtained comparable between different subimage sizes across different fingerprints.

3) The normalized power spectrum is multiplied by a filter representing the spatial frequency response of the Human Visual System (HVS) [Nill & Bouzas]. Inclusion of such a filter makes the final quality values more closely correspond to human observer assessments of relative quality.

4) The 2-D normalized, filtered power spectrum values at non-zero frequencies are summed, resulting in a single quality number for the given subimage.

5) For an inked or livescan image, with its multiple candidate windows, the window having the highest IQ value is identified and all other windows are then discarded. A latent image starts with and ends with only one window, so only one IQ value exists.

6) Additional normalizations and weightings on the single IQ value identified in step 5 are applied, to ensure that IQ magnitudes are intercomparable between disparate fingerprint images, and to scale all 500 ppi fingerprint images to a 0 to 100 range, where 0 is worst quality, 100 is best quality. These operations include the following:

7) The total power normalized IQ performs well in identifying the highest IQ window from a set of candidate windows in a given image, especially for oddball/difficult cases, partly because it somewhat suppresses the influence of contrast. However, for the final IQ value a stronger influence from contrast is desirable, which is obtained by converting the total-power-normalized IQ to zero-frequency-power-normalized IQ, i.e., normalization by "dc power" [Nill-2]. This dc-normalized IQ is then weighted by multiplying by the square of the average gray level in the given window, which is done to counterbalance the dc-normalized IQ's propensity to over-estimate the true IQ of dark images and under-estimate the true IQ of light images.

8) The dc-normalized, average gray weighted IQ suffices in most cases, but is deficient in two special cases which requires further adjustment: a very high contrast/binary image, or a very light image which retains good fingerprint structure. Without further adjustment, the quality of the very high contrast/binary image would be over-estimated and the quality of the very light but structured image would be under-estimated. The following adjustments are therefore made when such images are detected:

> -- A high contrast/binary subimage is detected if its IQ is higher than the very best valid IQ, or if the subimage contrast[7] is > 1.6. In either case, the dc-normalized, average gray weighted IQ is deweighted by a multiplicative constant, HCfactor = 0.17. [In some specialized applications a binary fingerprint image may be more prized than a grayscale fingerprint image; in such cases, one might want to increase HCfactor to a value of ~0.5.]

[7] IQF computed contrast is: square root of [nonzero frequency power / zero frequency power], which equals the standard deviation of image gray levels divided by the average gray level.

-- A very light but structured image is detected if the average gray level is > 200 and the contrast > 0.1. In this case the IQ is upweighted by multiplying it by 2.5.

9) The IQ value is scaled by the ppi resolution level.

It can be shown on theoretical grounds that the power spectrum of a scene (e.g. finger) is inherently scale-independent. Scale independence is not a problem if all fingerprints are captured at the same scale, say 500 ppi. However, if, for example, one were to run a very high quality 1000 ppi image along with a very high quality 500 ppi image and left the scale independence intact, then the computed quality value of the 500 ppi image could be as high or higher than the 1000 ppi image quality value, which would be an incongruous result. IQF therefore makes the quality value scale-dependent by multiplying the pre-final IQ value by ppi/500. This ensures that a high quality 1000 ppi image will have a higher quality value than a high quality 500 ppi image. [This scale factor is consistent with the scaling approach discussed in [Nill & Bouzas], with reference to equation 13].

However, it would be wrong to assume that quality strictly increases with ppi resolution level, regardless of that resolution level, and regardless of the type of fingerprint image: inked, livescan, or latent. For example, an inked fingerprint has finite information content; its quality depends on such variables as the ink (viscosity, etc.), finger pressure on the ink pad, transfer pressure to the card stock, and properties of the card stock itself (fiber content, bleeding, etc.). Scanning the card at an ever increasing ppi resolution level will not produce an ever increasing image quality level in the digitized image. On the other hand, livescan quality could theoretically strictly increase with ppi resolution level, as long as the opto-electronic (or other) device noise level can be kept low enough and the resolution increase is a "true" resolution of the device. This is so because the surface structure of the actual finger is being captured in a livescan, which has information content down to the molecular level, in the extreme[8]. Latent fingerprints can be captured by a variety of means, and some approaches will have more information content and resolution capability than other approaches.

Considering the above, the scaling in IQF is considered to be valid up to about 1000 ppi; beyond that resolution level the IQF magnitude is suspect, at least for inked or latent images. The program therefore prints-out a warning if the input image resolution is greater than 1010 ppi.

10) The scaled IQ is normalized by the highest valid IQ at 500 ppi, i.e., maxIQ500 = 4850. The result is raised to the 1/3 power (cube root) and then multiplied by 100, such that a 500 ppi image has a quality range of 0 (worst quality) to 100 (best quality).

[8] There are, however, many practical device design and fabrication problems in present-day livescan devices that precludes attainment of the theoretical principle: that quality strictly increases with resolution.

[The 1/3 power tends to make the IQ values better approximate the visual sense of relative quality levels.]

Additional Considerations:

• The image quality computations performed on the peak window of in an inked or livescan image, are identical to the image quality computations performed on the user-defined window in a latent image. So, if by chance the inked/livescan peak window and user-defined latent window had identical fingerprint content, then the resultant IQ values would be identical.

• An inked/livescan image can only contain a single fingerprint; the program will still run to completion if there is more than one fingerprint in an image, but the result will generally be meaningless. [If one wants to process a ten-print card image, an algorithm would need to be added to IQF to locate each rollblock in the image, then crop each rollblock out as a separate image, then run the 10 rollblock images, one after another.]

• There are quite a few constants in the program (principally in GlobalData, Fprint, and IQF subroutines) and many of them are inter-dependent. It may happen that one image may produce a quality value that doesn't seem 'right', and it might be found that by adjusting one or another constant makes the quality value 'better'. However, that change of a single constant could make the quality assessment of 100 other images (that were not considered) incorrect, because of the interdependencies of the constants.

2.3 Program Subroutines

All of the following IQF subroutines were written by the author (in Fortran-90/95), except for the FFT subroutine.

IQF
- main subroutine, calls other subroutines and performs functions:
- user interface via menu display, or call command line interface
- compute 2-dimensional (2-D) power spectrum from output of FFT
- identify peak IQ window from set of windows for given image
- renormalize peak IQ and weight by average gray
- detect special cases where additional IQ weightings are needed
- normalize IQ to 0 to 100 quality range (500 ppi image)
- print-out to file & display

ReadAux
- read-in the input data file containing list of images

CommandLine
- parse the data that user entered on command line

PGM
- if input image is in PGM format, retrieve header bytes, width pixels, height pixels from header

ReadImage
- read-in image as raw, given header bytes, width pixels, height pixels; store entire image in memory

Fprint
- find fingerprint in image, define rectangle encompassing it
- define single size square window (for given image) and multiple window locations within fingerprint rectangle area
- weed-out windows with very poor or no fingerprint structure content

FFT
- compute 2-D fast Fourier transform of subimage contained in square measurement window; output real and imaginary parts in spatial frequency space
- this is a modification of the freeware Brenner/Rader FFT; updated to Fortran-90/95 and modified/streamlined for processing 2-D images

computeIQ
- weight 2-D power spectrum by visual system filter, normalize by total power, and sum all spectrum data to get preliminary IQ for given subimage
- compute subimage contrast and average gray from power spectrum

WriteImage
- write entire input image to a new file in PGM format, with a boundary line around the selected peak IQ window (diagnostic/verbose output option)

Help
- explanation of runtime menu items
- how to setup input data file
- other program notes

BadImageList
- running count of all images listed in input data file that cannot be processed (for any number of reasons); image file names listed to output.

GlobalData
- defines global constants, parameters, switches, etc.

2.4 Program Compilation

• Compile for running in command line mode:

cmdline = 'Y' (in GlobalData.f95)
activate line: CALL COMMANDLINE(.) (in iqf.f95)
include CommandLine.f95 in compiler files list
Compiler must recognize the Unix "Getarg()" subroutine, e.g., by adding Unix library to compiler options.

For Apple Mac OS 10, command line mode means activating the Unix terminal mode. For Microsoft Windows OS, command line mode means activating the "command prompt" window.

• Compile for running as double-click app:

cmdline = 'N' (in GlobalData.f95)
comment-out line: CALL COMMANDLINE(.) (in iqf.f95)
exclude CommandLine.f95 from compiler files list

Note that the IQF source code does not contain any GUI code. We compile with Absoft Pro Fortran v10 (32/64 bit), which wraps a Mac OS GUI or Windows OS GUI into the binary ("executable") application; other compilers may or may not have this feature.

• The InputDataFile formats and all output files are the same, whether running from the command line mode or from displayed menu in double-click app mode.

3 Program Testing

Program debugging and algorithm optimization were performed with various sets of image types, e.g., binary images, cluttered background inked images, variable contrast images, and light-gray images, among others, with a variable number of images in a set (usually less than 100). The internal algorithms were modified as a result of running these sets, and the intermediate IQF version was run against a carefully selected, visually-ranked base set of 29 inked and livescan images, which included a large range of visual quality levels, contrast, and lightness/darkness (mostly 500 ppi images with a few at 1000 ppi), resulting in further adjustments[9].

In a comparison of the IQF values of these test images with their visual image quality, the IQF magnitudes appear to loosely correspond to the 4 usability classifications defined by the Biometric Application Programming Interface Consortium for biometric quality metrics having a 0 to 100 magnitude range [BioAPI]. For 500 ppi fingerprint images this correspondence is:

BioAPI Quality Score (~IQF value)	BioAPI Biometric Utility:
0 - 25	unacceptable
26 - 50	marginal
51 - 75	adequate
76 - 100	excellent

Finally, as stated at the outset in section 1, IQF is tailored to correlate with the visual sense of quality and verification has been in that sense; no experimentation has been performed against automated computer matchers. Other fingerprint quality metrics, such as NIST's Fingerprint Image Quality metric [NFIQ], are tailored specifically for predicting automated computer matcher performance. Correlation between IQF and NFIQ, with their divergent purposes, has often been found to be poor on an individual livescan basis, with somewhat better correlation after averaging results over sets of livescans.

[9] These fingerprint images were drawn from a variety of sources, most of which are not publicly releasable.

4 Running IQF

IQF can be run with either a single image or multiple images; either as a double-click app or from the command line. The user-selected output is either concise (just IQ) or diagnostic/verbose (IQ, contrast, average gray, image with scribed-in window location, etc.).

4.1 Image File Formats

IQF only directly reads PGM format, for all other image formats the number of header bytes, width in pixels, and height in pixels must be determined before the IQF run and those images are read-in as raw images. In all cases the image must be uncompressed and 8 bpp grayscale.

IQF reads the so-called "P5" PGM format for 8 bpp grayscale images described at:

http://badc.nerc.ac.uk/data/claus/PGM_format.html

For recognition by IQF, the PGM image filename must end with the non-case sensitive tag: **.**pgm

All latent images must be in PGM format, but inked or livescan images can be in other formats, such as TIFF. One freeware image viewer application that will give the total header size, width and height of a TIFF image is,

ImageJ for Mac OSX and Windows, available at:

http://rsb.info.nih.gov/ij/

[Before opening the TIFF image with *ImageJ*, enable the "debug mode" via edit/options/miscellaneous, then open the image and inspect the log window, where "offset" or "stripoffset" is the header bytes, and "imagewidth", "imagelength" is the width and height, respectively.]

A BMP format inked or livescan image can also be read-in as raw, but note that the image is generally stored 'upside down' in BMP format, i.e., the first row of pixels in the file is really the bottom row of the image. If it is input to IQF as-is (with correct header size), the program will assume the image is right side up, giving questionable results. To avoid this, either flip the image before saving as BMP, or convert a non-flipped BMP image to TIFF, PGM, or true raw format, before input to IQF. Freeware that extracts BMP image properties is *ImageInfo*, available at:

http://schmidt.devlib.org/image-info/

Finally, note that for non-PGM images, the program attempts to decipher the image polarity, but the method is not foolproof (see algorithm in *readimage* subroutine). When running in the diagnostic/verbose output mode discussed in the next section, the gray level of the upper left corner pixel is output (*pix1*), which can be used to verify that the program selected the correct polarity. That is, if the true *pix1* value, as determined by an independent image viewer program, is: (255 - IQF's printout *pix1* value)
then there is a high probability that IQF read the image with the wrong polarity. If IQF's printout *pix1* value is wrong and subtracting it from 255 does not make it correct, then there is a high probability that the wrong image header size was used (wrong offset number of bytes to first pixel in image file).

4.2 Run-Time Menu Options and Output

In the double-click app mode, the user sees the following display and enters the RunCase and ImageType:

RunCase:
s Single image run
c Concise output with multiple images (I/O files)
d Diagnostic/verbose output with multiple images (I/O files)
h Help

ImageType:
k Inked or Livescan
t Latent
h Help

enter: RunCase ImageType

single image run (s):

if inked or livescan, user types in:
ppi ImageFileName

then if image is not in PGM format, user types in:
header width height

if latent image, user types in:
ppi WindowWidth ULcol ULrow ImageFileName
(ULcol, ULrow defines upper left corner of square measurement window)

IQ value is displayed (no file output).

multiple images run (c or d):

The user first constructs a data file containing the needed image information. In the following input data file example for inked and/or livescan images, the entries are:
ppi header imagewidth imageheight imagefilename
(PGM image does not require header, width, height entries):

500 186 800 750 img2.tif
1000 0 1600 1500 img3.raw
500 img1.pgm

For a latent image run, the user needs to define the location and size of the single square window to be measured for quality in each image, before running IQF. The latent image must be in PGM format; the corresponding input data file entries are:
ppi windowWidth Upperleft col of window Upperleft row of window imagefilename:

500 200 165 323 img4.pgm

With the concise option **c** run, the program's output file contains:
IQ ppi imagefilename:

90 500 img1.pgm
83 500 img2.tif
140 1000 img3.raw

With the diagnostic/verbose option **d**, the program's output file contains additional information, as in the following example:

IQ	Contrast	avgGry	n	Width	Col	Row	pix1	ppi	
90	0.6361	151.2	45	280	260	125	252	500	img1.pgm

where **IQ**, **Contrast**, and **avgGray** (average gray) are for the peak IQ window, **n** is the number of useful candidate windows from which the peak window was selected, **Width** is the window width, **Col** & **Row** define the upper left corner of the peak IQ window, and **Pix1** is the gray level of the pixel in the upper left corner of the entire image, which can be used to verify that the program read image polarity correctly, or that the correct header size was used (see end of section 4.1).

[It is also possible to output the above information for all **n** candidate windows defined in the fingerprint area: in GlobalData subroutine, set SubImageInfo = 'Y'.]

With option **d**, in addition to the output data file, the entire input image is reconstructed in PGM format with a visible boundary line drawn around the peak IQ measurement window, as illustrated in Figure 4-1. This image has the tag: _IQ.pgm appended to the

end of the original image file name. [This image reconstuction option can be turned-off, see "PGMwrite" parameter in GlobalData.]

Finally, the output data file lists all images that could not be processed, for any of a variety of reasons. If this number exceeds a threshold during runtime, then program execution is halted (maxbadimages=100 in GlobalData).

Figure 4-1. IQF Output Image (Run Option "d"), Square shows Peak IQ Subimage Window

In the command line mode, the 5 valid argument sets for inked/livescan images are given in the following; for a latent image, substitute **st**, **ct**, or **dt** for **s**, **c**, or **d**:

s ppi ImageFileName	(single image in PGM format)
s ppi hdr width height ImageFileName	(single image not in PGM format)
c InputDataFile	(concise output, multiple images)
d InputDataFile	(diagnostic output, multiple images)
hh	(help)

4.3 Run-Time Problems

Image was not processed:

Verify that input data file has correct linebreak for given app on given operating system.

Verify that input data file has correct data format and correct values.

Make sure that image width and height are each greater than the minimum window size (see MinWindowWidth in GlobalData subroutine).

Make sure computer has adequate memory available for the program.

IQ value does not even approximately reflect visual quality (way too high or low):

Run image in diagnostic mode (option d).

Check peak IQ window location, it should cover 'best' part of fingerprint.

Check that program read image polarity correctly (use pix1 output value).

Check that the correct ppi level was input.

For some inked images with high structure background, the program may not find the best window within the fingerprint itself; in such a case, user can predefine the best window and run as latent image.

Compute time is slow:

There are a number of possible ways to speed-up compute time:

- increase distance between successive windows ("increment" in Fprint subroutine); but see comments in section 2.1.1

- recompile with more optimized compiler/settings

- use a faster FFT algorithm

- the obvious: use a faster computer

List of References

PIV
Personal Identity Verification: Image Quality Specifications for Single Finger Capture Devices, FBI PIV-071006, July 2006
http://www.fbi.gov/hq/cjisd/iafis/piv/pivspec.pdf

AppF
Image Quality Specifications, Appendix F of the FBI's Electronic Fingerprint Transmission Specification, IAFIS-DOC-01078-7.1, May 2, 2005.
http://www.fbi.gov/hq/cjisd/iafis/efts71/efts71.pdf

Nill & Bouzas
Objective Image Quality Measure Derived from Digital Image Power Spectra, N.B.Nill and B.H.Bouzas, Optical Engineering, April 1992, vol.31, pp. 813-825.

Nill-1
Scene Power Spectra: the Moment as an Image Quality Merit Factor, N.B.Nill, Applied Optics, November 1976, vol.15, pp. 2846-2854.

Nill-2
Contrast Effect on Imagery Power Spectra, N.B.Nill, Applied Optics, 1 July 1979, vol.18, pp. 2147-2151 [and errata: N. B. Nill, Applied Optics, 15 December 1980, vol.19, pg. 4135].

BioAPI
BioAPI Specification Version 1.1, 16 March 2000, The BioAPI Consortium
http://www.bioapi.org/DownloadsPage1.html
(also published as ANSI/INCITS Standard 358-2002)

NFIQ
Fingerprint Image Quality, E.Tabassi, C.L.Wilson, & C.I.Watson,
NISTIR 7151, August 2004;
document and software available at (under NBIS heading):
http://www.itl.nist.gov/iad/894.03/nigos/nigos.html

Appendix A

Computational Flow in IQF

Terminology:

• A term in *italics* refers to the IQF subroutine that performs the given computation.

• A term in **boldface** refers to an IQF code parameter name.

• Most terminology follows that given in [Nill & Bouzas], where more detailed descriptions, derivations, background can be found.

h(x,y) = two-dimensional square image, i.e., gray level of pixel at column x, row y
h(x,y) is a given square fingerprint window that was either determined in *Fprint* (inked or livescan fingerprint processing), or was defined by the user (latent fingerprint processing).
x = horizontal direction, across image width, across image pixel columns
y = vertical direction, across image height, across image pixel rows
H(u,v) = two-dimensional Fourier transform of h(x,y)
v = spatial frequency in vertical direction (in Fourier domain)
u = spatial frequency in horizontal direction
ρ= radial spatial frequency
M = number of pixels across h(x,y) square image width or height
P (u,v) = unnormalized image power spectrum

The Fourier transform of h(x,y) is computed in *FFT*:

$$H(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{M-1} \exp[-2\pi i y\frac{v}{M}]\exp[-2\pi i x\frac{u}{M}]\, h(x,y), \qquad u,v = -\frac{M}{2}\bullet\bullet\bullet\frac{M}{2}$$

Comment: in the actual code, h(x,y) is first converted to a one-dimensional array for compatible input to the Brenner *FFT* routine, then the one-dimensional array output of *FFT* is reconverted back to the two-dimensional array, H(u,v) (conversions performed in *iqf*).

H(u,v) is converted to the image power spectrum in *iqf*:

$$P(u,v) = |H(u,v)|^2 = \{Re(u,v) + Im(u,v)\}^2$$

Re, Im = real and imaginary parts of Fourier transform, respectively

The zero frequency power (**dcpower**), the total power (**totpower)**, and total non-zero power (**acpower**) are computed in *computeIQ*:

$$\textbf{dcpower} = P(0,0) = |H(0,0)|^2 = \left|\frac{1}{M^2}H(0,0)\right| M^4 = (\textbf{avgGray})^2 M^4$$

$$\textbf{totpower} = \sum_{u=-\frac{M}{2}}^{\frac{M}{2}} \sum_{v=-\frac{M}{2}}^{\frac{M}{2}} P(u,v)$$

acpower = totpower - dcpower

The average gray level and contrast are computed (in *computeIQ*):

avgGray = (**dcpower**)$^{1/2}$ / M^2

contrast = (**acpower** / **dcpower**) $^{1/2}$

Comment: this measure of contrast is equal to:
the standard deviation of image gray levels divided by the average gray level.

A rotationally symmetric Human Visual System filter, A(Tr), (**hvsmtf2** in code) is computed, which is the square of a model of the modulation transfer function of the visual system:

$A(T\rho) = (0.2 + 0.45T\rho) \exp(-0.18T\rho)$

where, T = constant that fixes frequency of peak of $A(T\rho)$
T = (2*viewdist) Tan(0.5^o) / spot
viewdist = pseudo, nominal eye-to-display distance (mm)
spot = pseudo, nominal display pixel width (mm)
see **HVSparm** in *GlobalData*

A(Tρ) is multiplied by the total power normalized power spectrum:
$P_1(\rho,\theta) = P(\rho,\theta)$ / **totpower**

and the product is summed around the 360 degree circle, from ρ = 0.01 to ρ = $1/\sqrt{2}$ cycles per pixel width:

$$\textbf{dbl\_iq} = \sum_{\theta=0}^{360} \sum_{\rho=.01}^{\frac{1}{\sqrt{2}}} A(T\rho)P_1(\rho,\theta)$$

Comment: actual code in *computeIQ* for calculating **dbl_iq** is via a combination of radial and cartesian coordinates and takes advantage of bilateral symmetrism of $P_1(\rho,\theta)$. In terms of cartesian coordinates, summation is from -0.5 to +0.5 cycles per pixel width in x & y directions.

dbl_iq is the preliminary IQ value for the given fingerprint window:

prelim_IQ = **dbl_iq**

The following final steps are performed in *iqf* subroutine for the window identified as having the highest **prelim_IQ** value (contrast, avgGray, etc. refer to the values for that peak window).

The **prelim_IQ** is renormalized to dcpower by multiplying by (1+ $contrast^2$), and is then multiplied by the square of the average gray level, forming the base final IQ value (**xIQdc**):

xIQdc = prelim_IQ * (1+ $contrast^2$) * ($avgGray^2$)

Comment: the output of MITRE's IQM program equals the IQF base value under the following IQM run conditions:

running identically the same single subimage as in IQF
sensor = 4
no noise filter
default preferences (with dc normalization)
then:
IQM output "IQ" value = **xIQdc** / $avgGray^2$ = prelim_IQ * (1+ $contrast^2$)

xIQdc is deweighted if one of several conditions exists, indicating that the image is very high contrast or binary, i.e., if

xIQdc > **maxIQ500**
or
contrast > 1.6

where **maxIQ500** is the maximum valid IQ value (defined in *GlobalData*).

xIQdc is upweighted if the image is very light but still appears to retain good fingerprint ridge structure, i.e., if
avgGray > 200 and contrast > 0.1

The **xIQdc** value, possibly weighted as given above, is normalized by **maxIQ500**, raised to the 1/3 power, scaled by the ppi resolution level, and multiplied by 100, such that a 500 ppi image has a final quality value (**IQfinal)** in the range of 0 to 100, and a 1000 ppi image has a final quality value (**IQfinal)** in the range of 0 to 200.

Comment: raising to the 1/3 power makes the relative **IQfinal** values more closely correspond to visual assessments of relative quality levels, for this type of metric. See section 4 of [Nill & Bouzas] for related background on 1/3 power use.

Appendix B

MITRE Legal Notice

Glossary

AC	Alternating Current (literal)
ANSI	American National Standards Institute
app	(software) application
BMP	Basic Multilingual Plane
bpp	bits per pixel
CJIS	Criminal Justice Information Services (division of FBI)
DC	Direct Current (literal)
DOJ	Department of Justice
FAR	Federal Acquisition Regulations
FBI	Federal Bureau of Investigation
FFT	Fast Fourier Transform
GUI	Graphical User Interface
HVS	Human Visual System
IAFIS	Integrated Automated Fingerprint Identification System
INCITS	InterNational Committee for Information Technology Standards
IQ	Image Quality
IQF	(MITRE) Image Quality of Fingerprint
IQM	(MITRE) Image Quality Measure
mm	millimeter
MTF	Modulation Transfer Function
NBIS	NIST Biometric Image Software
NFIQ	NIST Fingerprint Image Quality
NIST	National Institute of Standards and Technology
NISTIR	NIST Interagency or Internal Report
OS	(computer) Operating System
PGM	Portable GrayMap
PIV	Personal Identity Verification
ppi	pixels per inch

TIFF	Tagged Image File Format
UL	Upper Left (corner)
2-D	Two-Dimensional