NIST Special Database 14
Mated Fingerprint Cards Pairs 2
Version 2

Craig I. Watson (cwatson@nist.gov)

National Institute of Standards and Technology
Bldg. 225, Rm. A216
100 Bureau Drive, Mail Stop 8940
Gaithersburg, MD 20899-8940

ACKNOWLEDGEMENTS

I would like to acknowledge the Federal Bureau of Investigation who provided funding and resources in conjunction with NIST to support the development of this fingerprint database.

TABLE OF CONTENTS

Mated Fingerprint Cards Pairs 2
Version 2

C. I. Watson

Keywords: ANSI/NIST format, FBI, fingerprint, grayscale, image, rolled, plain impressions, WSQ compression, resolution.

1. INTRODUCTION

This document describes the NIST fingerprint database, NIST Special Database 14, version 2. Version 2 of the database uses a certifiable version of the wavelet scalar quantization (WSQ) compression code [1] and all the images are recompressed from the original scanned data. The database is being distributed for use in the development and testing of automated fingerprint classification systems. Also, the first 13,500 pairs of images are the same fingerprints which were encoded using a lossless technique and stored in NIST Special Database 9 Volumes 1-5 giving users the potential to evaluate the WSQ compression algorithm. NIST Special Database 14 version 2 consist of 4 discs, each containing 6,750 pairs of fingerprint images for a total of 54,000 images. The prints are 832 (w) X 768 (h) pixels and the storage required on each CD-ROM is approximately 550 megabytes when the prints are compressed and 8.3 gigabytes of storage when uncompressed (15 : 1 average compression ratio, using a bitrate of 0.75).

The data consists of mated fingerprint card pairs, which are two cards from the same individual, drawn from current work at the FBI Manual Image Comparison - Technical Section (MIC-TEC) verify desk. Fingerprint examiners from the Technical Section have attempted to take these card pairs at random, thus approximating a horizontal slice at the card level (see APPENDIX B for class distribution statistics). However, because the work is stacked by sex and unit at MIC-TEC verify desk, a true card level horizontal slice can not be guaranteed. A unit is a major grouping of the Henry Classifications [2].

The fingerprints are classified using the National Crime Information Center (NCIC) classes assigned by the FBI. Valid classes include: Arch, Tented Arch, Radial Loop (Ridge Counts), Ulnar Loop (Ridge Counts), Plain Whorls (all Whorls have Inner, Outer or Meeting Ridge Tracings), Central Pocket Whorl, Double Loop Whorl, Accidental Whorl, Scar and Amputation [2]. All classes and references are stored in a comment field in the WSQ compressed file, allowing for comparison with hypothesized classes.

2. DATABASE CONTENT AND ORGANIZATION

The four CD-ROMs in NIST Special Database 14 version 2 each contain 13,500 8-bit gray scale fingerprint images stored in the data directory. Included with the fingerprint data are software and documentation. See Figure 1 for the directory layout. A copy of this document (sd14ver2.pdf) is on the CDROM.

2.1 Database File Hierarchy

The top level of the file structure contains three directories src, man, and data. The code needed to decompress and use the image data is contained in the src directory with man (manual/help) pages

for the source code stored in the man directories. The data directory contains the fingerprint images stored in two levels of subdirectories for easier access and clarity (see Figure 2). The first level

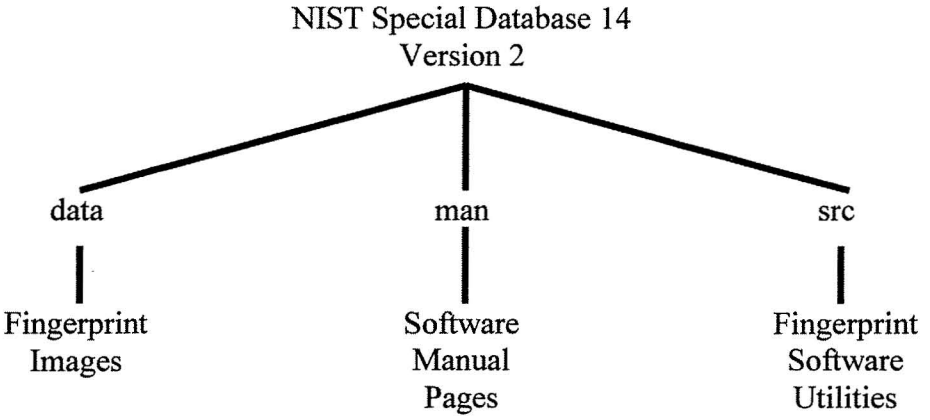


Figure 1 Top level directory tree for each CD-ROM in NIST Special Database 14.

consists of nine subdirectories, mfcp2_1 - mfcp2_9 (mated fingerprint card pairs 2), which divide the mated card pairs into groups of seventy five. The next level has a subdirectory for each of the seventy five mated card pairs stored there, with each subdirectory containing the segmented fingerprint pairs for that mated fingerprint card pair (20 total fingerprints per card level subdirectory). The names of the "card level" subdirectories (2nd level in the data directory) are sequentially continuous throughout the database.

Fingerprints are stored with filenames containing one letter, seven digits and a ".wsq" extension. The first character in the filename is always an "f" or "s" distinguishing if the card, from which the fingerprint was segmented, was a file or search card. The next seven characters indicate the sequence number, i.e., 0000001 - 0027000. The finger number is given by the last of the seven digits, with zero representing digit ten on the fingerprint card (see Figure 3 for fingerprint card layout). Every ten prints in sequential order are a group of prints from the same card, i.e., digits 1-10. The card and fingerprint numbers are sequentially continuous from the first CD-ROM through the last CD-ROM in the database.

2.2 Automated Name and Technical Search Cards

The first character in the second field of the FBI tape filename in the "HISTORY" tagged field or the IHEAD parent field (see Section 3) indicates if the card type was an automated name (n) or technical (t) search. A t means the current Identification Division Automated System (IDAS) found the ident by means of Automated Technical Search (ATS). The card was non-identifiable in Automated Name Search (ANS) and was classified by any one of a large number of fingerprint examiners. Thus database cards defined as search cards (indicated by an "s" in the filename) that are t cards represent a broad base of the Technical Section's (TECH) current manual classification practice. Database file cards (indicated by an "f" in the filename) that are t cards represent a broad base of TECH historical manual classification practice. The fact that the ident of the search and file cards was based in part on manual classification, means that both the search card and the file card

are classifiable at some minimum level of confidence, and that the classifications are the same, within the error permitted by ATS.

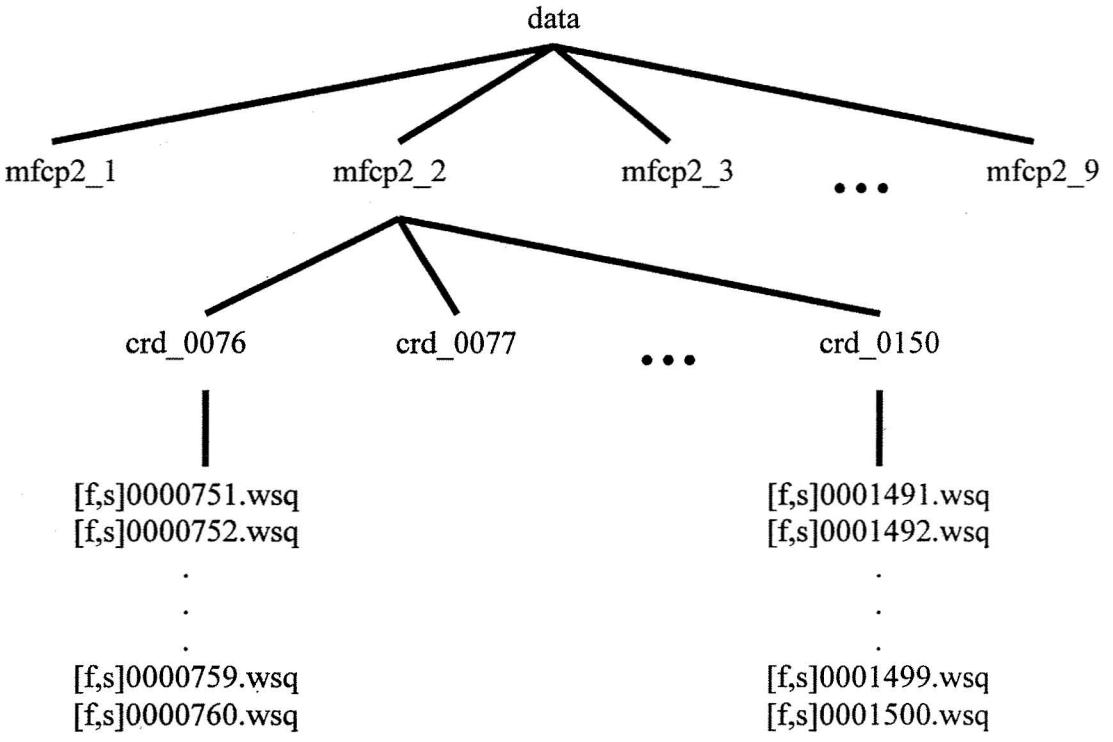


Figure 2 Arrangement of fingerprint files in data directory.
(Note: This example is from CD-ROM 1 of 4.)

1. R. Thumb	2. R. Index	3. R. Middle	4. R. Ring	5. R. Little
6. L. Thumb	7. L. Index	8. L. Middle	9. L. Ring	10. L. Little

Figure 3 Layout of fingerprint card numbers.

The n means that IDAS found the ident using ANS. The search card is not classified by the main pool of examiners. Instead, the search card is classified for this project by an expert examiner directly associated with data collection. The database search cards that are n cards represent current TECH manual classification practice as done by a single individual and database file cards that are

n cards have the same classification characteristics as file cards that are t cards. Considered as search/file pairs, the n cards are not guaranteed to ident in ATS, although they are in fact identical.

2.3 NCIC Classifications

The classes stored in the "HISTORY" tagged field or the IHEAD id field (see Section 3) are the NCIC classes [2], including any references (see Figure 5), that were assigned by IDAS. A listing of the possible class codes is given in Figure 4. Note that the classification ac (approximate class) means that the classification immediately after the ac is the best classification that can be assigned to the print given the information shown in the image.

Classification name (**class code**)
Arch (**aa**)
Tented Arch (**tt**)
Ulnar Loop Ridge Counts (**01 - 49**)
Radial Loop Ridge Counts (**51 - 99**)
Plain Whorl Inner Ridge Tracing (**pi**)
Plain Whorl Outer Ridge Tracing (**po**)
Plain Whorl Meeting Ridge Tracing (**pm**)
Central Pocket Whorl Inner Ridge Tracing (**ci**)
Central Pocket Whorl Outer Ridge Tracing (**co**)
Central Pocket Whorl Meeting Ridge Tracing (**cm**)
Double Loop Whorl Inner Ridge Tracing (**di**)
Double Loop Whorl Outer Ridge Tracing (**do**)
Double Loop Whorl Meeting Ridge Tracing (**dm**)
Accidental Whorl Inner Ridge Tracing (**xi**)
Accidental Whorl Outer Ridge Tracing (**xo**)
Accidental Whorl Meeting Ridge Tracing (**xm**)
Approximate Class (**ac**) followed by a valid class
Amputation or Missing (**xx**)
Scar or Mutilation (**sr**)

Figure 4 Classification codes for NIST Special Database 14.

2.4 Class Referencing

The referencing of a fingerprint is caused by a variety of ambiguities such as a scar occurring in the fingerprint, the quality of the print rolling, or the print having ridge structures characteristic of two different classes. The referenced prints could easily cause a wrong classification when used in testing an automated classification system, but could provide a challenge in the later stages of development. NIST Special Database 14 version 2 contains prints with references (see APPENDIX B for referencing statistics). The classifications of these prints contain the primary fingerprint class

followed by any references. As an example the classification for fingerprint f0000042.wsq is "**co/09**" and the corresponding print is shown in Figure 5. The fingerprint is classified as a **co**, but the right delta is near the core making it similar to an ulnar loop so it was given a **09** reference.

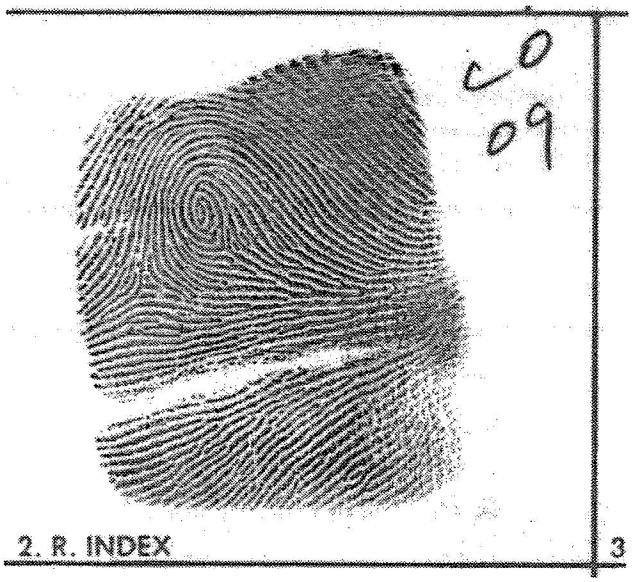


Figure 5 Fingerprint (f0000042.wsq) with a cross-referenced classification "co/09".

2.5 Segmenting the Fingerprint Images

The individual fingerprint images were obtained by scanning all ten prints on a card in one large image (4096 X 1536 pixels) and then segmenting each individual image from that larger image. The individual images were segmented at the same exact points on all the larger (full card) images. The image size of 832 X 768 pixels was selected to allow the user to reconstruct the image of the fingerprint card and resegment the individual fingerprint images if desired. The images overlap by 32 pixels with horizontally adjacent images and by 18 pixels with vertically adjacent images which has to be accounted for when reconstructing the image of the fingerprint card.

2.6 Inked and Live Scan Printed

The fingerprints were scanned from two types of prints. The first type were fingerprint patterns created by rolling the individual's ink-covered finger on the fingerprint card, denoted by an i in the "SCAN_TYPE" tagged field or the IHEAD id field (see Section 3). The second type were patterns taken with a live scanning device and then printed onto a fingerprint card (denoted by an l). This is important in that the quality of the image is affected by the resolution of the printer used to print the live scanned image onto the fingerprint card.

3. FINGERPRINT FILE FORMAT

The fingerprint images are stored on the CDROM's as WSQ compressed files. The IHEAD structure is no longer used on the compressed image file. The user can decompress the images into either IHEAD formatted files or raw data files. The decompression routines **dwsq** and **dwsq14v2** create a companion file with the extension ".ncm" for each file decompressed. This ".ncm" file is a

tagged text file that contains information about the fingerprint image file (see Figure 6) that may be especially useful if the user chooses a raw pixel format for the decompressed image data.

The WSQ compression algorithm was written using the "WSQ Gray-scale Fingerprint Image Compression Specification" [1]. The algorithm is a certifiable implementation of the WSQ specification. Based on the current classification system (PCASYS) used by the Image Recognition Group [13], the decompressed images are acceptable when training and testing classification and matching algorithms.

3.1 Compressed File Format

The compressed files are stored in the format specified in the WSQ compression standard. The file's distinguishing information such as classification, sex, scan type (inked or live), and historical link to the original data collection are stored in a comment field in the compressed file header. The comment uses tags to identify the information. Figure 6 shows a printout of the tagged comment from file f0000042.wsq. This information can be accessed using the "**rdwsqcom**" command and gets printed to a file when using the commands "**dwsq**" and "**dwsq14v2**"(see section 4.2).

The first tag "NIST_COM" indicates there are 14 "tagged" values in the comment. "SD_ID" references NIST Special Database 14. "HISTORY" contains the original file name (f0000042.pct), the NCIC classification (co/09), a link to the original data collection (tape3.n1116018.02), and the original card image size (4096x1536). "FING_CLASS" is a conversion of the first NCIC classification assigned the print to a set of six possible classes (Arch, Left-Loop, Right-Loop, Scar, Tented-Arch, or Whorl). "SEX" is the gender of the fingerprint subject. "SCAN_TYPE" is for inked or live-scan as discussed in section 2.6. "PIX_WIDTH", "PIX_HEIGHT", "PIX_DEPTH"., "PPI" are pixel attributes stored in case the user decompresses to a raw image file format (see section 4.2). "LOSSY" is set to 1 if any lossy compression technique is used on the pixel data and "COLORSPACE" simply states the image is gray-scale data. "COMPRESSION" and "WSQ_BITRATE" describe the type of compression used and the bitrate used to compress the fingerprint pixel data.

```
NIST_COM 14
SD_ID 14
HISTORY f0000042.pct co/09 tape3.n1116018.02 4096x1536
FING_CLASS W
SEX m
SCAN_TYPE i
PIX_WIDTH 832
PIX_HEIGHT 768
PIX_DEPTH 8
PPI 500
LOSSY 1
COLORSPACE GRAY
COMPRESSION WSQ
WSQ_BITRATE 0.750000
```

Figure 6 Tagged comment in compressed file.

3.2 IHEAD Decompressed File Format

Numerous image formats exist, but most image formats are proprietary. Some are widely supported on small personal computers and others on larger workstations. A header format named IHead has been developed for use as a general purpose image interchange format. The IHead header is an open image format which can be universally implemented across heterogeneous computer architectures and environments. Both documentation and source code for the IHead format are publicly available and included with this database. IHead has been designed with an extensive set of attributes in order to represent adequately both binary and gray level images, to represent images captured from different scanners and cameras, and to satisfy the image requirements of diversified applications. These applications include, but are not limited to, image archival/retrieval, character recognition, and fingerprint classification. Figure 7 illustrates the IHead format.

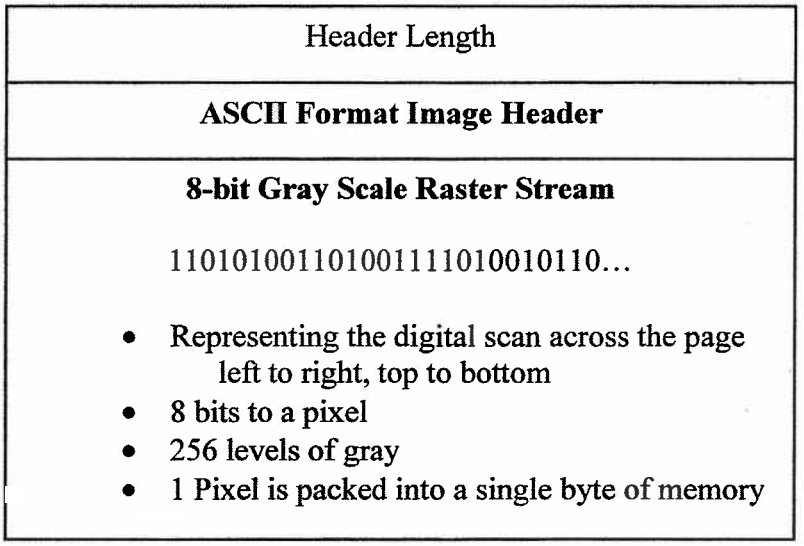


Figure 7 An illustration of the IHead raster file format.

Since the header is represented by the ASCII character set, IHead has been successfully ported and tested on several systems including UNIX workstations and servers, DOS personal computers, and VMS mainframes. All attribute fields in the IHead structure are of fixed length with all multiple character fields null-terminated, allowing the fields to be loaded into main memory in two distinct ways. The IHead attribute fields can be parsed as individual characters and null-terminated strings, an input/output format common in the `C' programming language, or the header can be read into main memory using record-oriented input/output. A fixed-length field containing the size in bytes of the header is prefixed to the front of an IHead image file as shown in Figure 7.

The IHead structure definition written in the `C' programming language is listed in Figure 8. Figure 9 lists the header values from an IHead file corresponding to the structure members listed in Figure 8. This header information belongs to the database file f0000042.wsq shown in Figure 5. Referencing the structure members listed in Figure 8, the first attribute field of IHead is the

```
/*************************************************************
        File Name: IHead.h
        Package:  NIST Internal Image Header
        Author:   Michael D. Garris
        Date:     2/08/90
*************************************************************/
/* Defines used by the ihead structure */
#define IHDR_SIZE  288     /* len of hdr record (always even bytes) */
#define SHORT_CHARS    8       /* # of ASCII chars to represent a short */
#define BUFSIZE    80      /* default buffer size */
#define DATELEN    26      /* character length of data string */

typedef struct ihead{
  char id[BUFSIZE];                /* identification/comment field */
  char created[DATELEN];           /* date created */
  char width[SHORT_CHARS];               /* pixel width of image */
  char height[SHORT_CHARS];              /* pixel height of image */
  char depth[SHORT_CHARS];               /* bits per pixel */
  char density[SHORT_CHARS];             /* pixels per inch */
  char compress[SHORT_CHARS];            /* compression code */
  char complen[SHORT_CHARS];             /* compressed data length */
  char align[SHORT_CHARS];               /* scanline multiple: 8|16|32 */
  char unitsize[SHORT_CHARS];            /* bit size of image memory units */
  char byte_order;                 /* 0->highlow | 1->lowhigh*/
  char sigbit;                     /* 0->sigbit first | 1->sigbit last */
  char pix_offset[SHORT_CHARS];          /* pixel column offset */
  char whitepix[SHORT_CHARS];            /* intensity of white pixel */
  char issigned;                   /* 0->unsigned data | 1->signed data */
  char rm_cm;                      /* 0->row maj | 1->column maj */
  char tb_bt;                      /* 0->top2bottom | 1->bottom2top */
  char lr_rl;                      /* 0->left2right | 1->right2left */
  char parent[BUFSIZE];            /* parent image file */
  char par_x[SHORT_CHARS];               /* from x pixel in parent */
  char par_y[SHORT_CHARS];               /* from y pixel in parent */
}IHEAD
```

Figure 8 The IHead `C' programming language structure definition.

identification field, id. This field uniquely identifies the image file, typically by a file name. The identification field in this example not only contains the image's file name, but also the sex of the individual, if the image was scanned from an inked or live scan printed image, and the NCIC classification of the fingerprint, along with any references to other classifications. This convention enables an image recognition system's hypothesized classification to be automatically scored against the actual classification.

IMAGE FILE HEADER

Identity	: f0000042.wsq m i co/09
Header Size	: 288 (bytes)
Date Created	: Mon Dec 7 18:39:44 1992
Width	: 832 (pixels)
Height	: 768 (pixels)
Bits per Pixel	: 8
Resolution	: 500 (ppi)
Compression	: 7 (code)
Compress Length	: 36649 (bytes)
Scan Alignment	: 8 (bits)
Image Data Unit	: 8 (bits)
Byte Order	: High-Low
MSBit	: First
Column Offset	: 0 (pixels)
White Pixel	: 255
Data Units	: Unsigned
Scan Order	: Row Major, Top to Bottom, Left to Right
Parent	: tape3.n1116018.02 4096x1536
X Origin	: 0 (pixels)
Y Origin	: 0 (pixels)

Figure 9 The IHead values for the fingerprint data file f0000042.wsq.

The attribute field, created (skip over the Header Size field in Figure 9), is the date on which NIST received the digitized image. The next three fields hold the image's pixel width, height, and depth. A binary image has a pixel depth of 1 whereas a gray scale image containing 256 possible shades of gray has a pixel depth of 8. The attribute field, density, contains the scan resolution of the image; in this case, 19.7 pixels/mm (500 pixels/inch). The next two fields deal with compression.

In the IHead format, images may be compressed with virtually any algorithm. Whether the image is compressed or not, the IHead is always uncompressed. This enables header interpretation and manipulation without the overhead of decompression. The compress field is an integer flag signifying which compression technique, if any, has been applied to the raster image data which follow the header. If the compression code is zero, then the image data are not compressed, and the data dimensions, width, height, and depth, are sufficient to load the image into main memory. However, if the compression code is nonzero, then the complen field must be used in addition to the image's pixel dimensions. The images create by the decompression software have a compression code of 0 signifying that the images are uncompressed data.

The attribute field, align, stores the alignment boundary to which scan lines of pixels are padded. Pixel values of 8-bit gray scale images are stored 1 byte (or 8 bits) to a pixel, so the images will automatically align to an even byte boundary.

The next three attribute fields identify data interchanging issues among heterogeneous computer architectures and displays. The unitsize field specifies how many contiguous bits are bundled into a single unit by the digitizer. The sigbit field specifies the order, most significant bit first or least significant bit first, in which bits of significance are stored within each unit. The last of these three fields is the byte_order field. If unitsize is a multiple of bytes, then this field specifies the order in which bytes occur within the unit. Given these three attributes, data incompatibilities across computer hardware and data format assumptions within application software can be identified and dealt with effectively.

The pix_offset attribute defines a pixel displacement from the left edge of the raster image data to where a particular image's significant image information begins. The whitepix attribute defines the value assigned to the color white. For example, the gray scale image described in Figure 9 is gray print on a white background and the value of the white pixel is 255. This field is particularly useful to image display routines. The issigned field is required to specify whether the units of an image are signed or unsigned. This attribute determines whether an image with a pixel depth of 8, should have pixel values interpreted in the range of -128 to +127, or 0 to 255. The orientation of the raster scan may also vary among different digitizers. The attribute field, rm_cm, specifies whether the digitizer captured the image in row-major order or column-major order. Whether the scan lines of an image were accumulated from top to bottom, or bottom to top, is specified by the field, tb_bt, and whether left to right, or right to left, is specified by the field, rl_lr.

The final attributes in IHead provide a single historical link from the current image to its parent image. The images used in this database were renamed from their original filenames, given by the FBI, and the `link' to the original filename was stored in the parent field as well as the size of the ten print image (columns x rows) before the individual fingerprints were segmented. The FBI filename consists of three fields separated by periods. The first field contains a tape number, i.e., tape1, tape2, ..., tapen, indicating the FBI streamer tape the file was stored on. The second field contains 8 characters. The first character in the second field is always an n or t indicating an automated name or technical search (see Section 2.2). The next four characters (all digits) are the month and day which the card was scanned and the last three characters of the field indicate the number of the card being scanned on the date given by the previous four digits. The last field indicates the finger number of the file (01-10).

The par_x and par_y fields contain the coordinates of the origin, upper left hand corner pixel coordinate, from where the extraction took place from the parent image. These fields provide a historical thread through successive generations of images and subimages.

We believe that the IHead image format contains the minimal amount of ancillary information required to successfully manage binary and gray scale images.

4. SOFTWARE

The source code in this distribution has been developed using the GNU project's `gcc` compiler and `gmake` utility[11]. Manual pages giving a detailed description of the software utilities are included in APPENDIX C. The software has been tested under LINUX [10]and Windows NT using the Cygwin library [12] and associated tools.[1] Most of the software came from existing code that is

[1] Specific software products identified in this paper were used to adequately support the development of the technology described in this document. In no case does such identification imply recommendation or endorsement by the National

distributed on our NIST Fingerprint Image Software CDROM [13]. If code was needed from a NFIS "library" (src/lib/*) then all the source code for that library was copied into this CDROM's **src** directory. This will reduce confusion if the user is using both CDROMs, by not having two versions of the code. The software tools (src/bin/*) **cjpegb, dwsq, dpyimage, and rdwsqcom** are all copied from the NFIS CDROM. Software tools **dwsq14v2 and dumpihdr** are new source code that only exists on this CDROM. The NFIS CDROM contains other tools for editing/converting ANSI/NIST records and compressing/decompressing color and gray images in lossless/baseline JPEG and WSQ.

The baseline JPEG code in **src/lib/jpegb** uses the Independent JPEG Group's compression/decompression code. For details on its copyright and redistribution, see the included file **src/lib/jpegb/README**.

4.1 Software Compilation

The software can be installed and compiled by first copying the contents of the **src** directory on the CD-ROM to a read/writable disk partition on your computer. The directory to which you copy is referred to as the *installation directory* <install_dir>.

The permissions on the copied subdirectories and the compilation scripts (named "*makefile.mak*") should then be changed to read/writable. Also note that the distribution was originally written and developed with all directories and file names in lowercase characters. The CD-ROM is in ISO-9660 format and some operating systems (such as the Win32 family) copy the CD-ROM's content converting directory and file names to all uppercase characters. Before compilation, these directories and file names should be changed to all lowercase characters. A Bourne Shell (sh) script has been provided to automatically change permission on directories and compilation scripts, and it combs the installation directory, converting subdirectory and file names containing uppercase characters to all lowercase characters. The script may be invoked from a UNIX or Cygwin shell window. Within the directory containing the script, type the following:

```
% sh setup.sh <install_dir>
```

The text <install_dir> is replaced by your specific installation directory path. Once copied and permissions changed, the software can be compiled by executing the following commands in the top-level installation directory on a Linux machine:

```
% make -f makefile.mak PROJDIR=<install_dir> depend

% make -f makefile.mak PROJDIR=<install_dir> install
```

The text <install_dir> is replaced by your specific installation directory path. Alternatively, on a Win32 machine with the Cygwin library and utilities installed, type the following commands:

```
% make -f makefile.mak PROJDIR=<install_dir> \
      X11_EXST=0 EXEEXT=.exe depend

% make -f makefile.mak PROJDIR=<install_dir> \
      X11_EXST=0 EXEEXT=.exe install
```

Institute of Standards and Technology, nor does it imply that the equipment identified is necessarily the best available for the purpose.

Successful compilation under Linux will produce the executable files stored in the top-level bin directory. To invoke these utilities you can specify a full path to these files, or you may add the top-level bin directory to your environment's execution path.

A manual page is provided for each utility in the top-level man directory. To view a man page on a Linux machine or a Win32 machine running a Cygwin shell, type:

% man -M <install_dir>/man <executable>

The text <install_dir> is replaced by your specific installation directory path and <executable> is replaced by the name of the utility of interest.

4.2 Software Tools

Some basic software tools are included on this CDROM to help the user access the WSQ compressed images.

The software for these commands is only on this database CDROM:

dumpihdr – Prints the IHEAD image structure to the screen. The IHEAD structure is attached to the uncompressed output image unless the "raw" option in chosen during decompression.

dwsq14v2 – Decompresses a WSQ compressed image. If the image is from NIST Special Database 14 version 2 it will format the IHEAD structure to contain the same information used in the original version of the database. If the user already has software to read the IHEAD format used in the original version of the database this command should probably be used instead of **dwsq**. This command also creates a companion tagged text file with a ".ncm" extension that contains information about the compressed file (see Figure 6).

The software for these commands was copied from the NFIS CDROM:

cjpegb - Convert the image to a baseline JPEG compressed file. Format is compatible with most image viewers. Warning: WSQ is a lossy compression and recompressing the images to baseline JPEG creates more data loss, so this should only be used, if needed, as an aid for viewing the images.

dwsq - Decompresses a WSQ compressed image. If the user wants the IHEAD structure of the uncompressed image to be formatted like the original version of the database use **dwsq14v2**. This command also creates a companion tagged text file with a ".ncm" extension that contains information about the compressed file (see Figure 6).

dpyimage - Will display a WSQ/JPEGL/JPEGB compressed image. Only works in an X windows environment.

rdwsqcom - Will read and display the comment fields in a WSQ compressed image. Useful for accessing the fingerprint reference information stored in the comment field.

REFERENCES

[1] WSQ Gray-scale Fingerprint Image Compression Specification, IAFIS-IC-0110v2, Criminal Justice Information Services, Federal Bureau of Investigation, February 1993.

[2] The Science of Fingerprints, Rev. 12-84, U.S. Department of Justice, Federal Bureau of Investigation. Available from U.S. Government Printing Office, Washington D.C. 20402.

[3] C.L. Wilson, G.T. Candela, P.J. Grother, C.I. Watson, and R.A. Wilkinson. Massively Parallel Neural Network Fingerprint Classification System. Technical Report NISTIR 4880, National Institute of Standards and Technology, July 1992.

[4] National Bureau of Standards, Standard Reference Materials, Reflection Step Table 2601.

[5] M.D. Garris, Design and Collection of a Handwriting Sample Image Database, Social Science Computing Journal, Vol. 10, to be published, 1992.

[6] C. Watson, "*NIST Special Database 4*: 8-bit Gray Scale Images of Fingerprint Image Groups," CD-ROM & documentation, March 1992.

[7] R.M. McCabe, and R.T. Moore, "Data Format for Information Interchange", American National Standard ANSI/NBS-ICST 1-1986, August 1986.

[8] C. Watson, "*NIST Special Database 9*: 8-Bit Gray Scale Images of Mated Fingerprint Card Pairs," Vol. 1-5, CD-ROM & documentation, May 1993.

[9] R.M. McCabe, "Data Format for the Interchange of Fingerprint, Facial, Scar Mark & Tattoo (SMT) Information," American National Standard ANSI/NIST-ITL 1-2000, July 2000. Available from R.M. McCabe at NIST, 100 Bureau Drive, Stop 8940, Gaithersburg, MD 20899-8940.

[10] Linux - a freely available clone of the UNIX operating system. Learn more at http://www.linux.org.

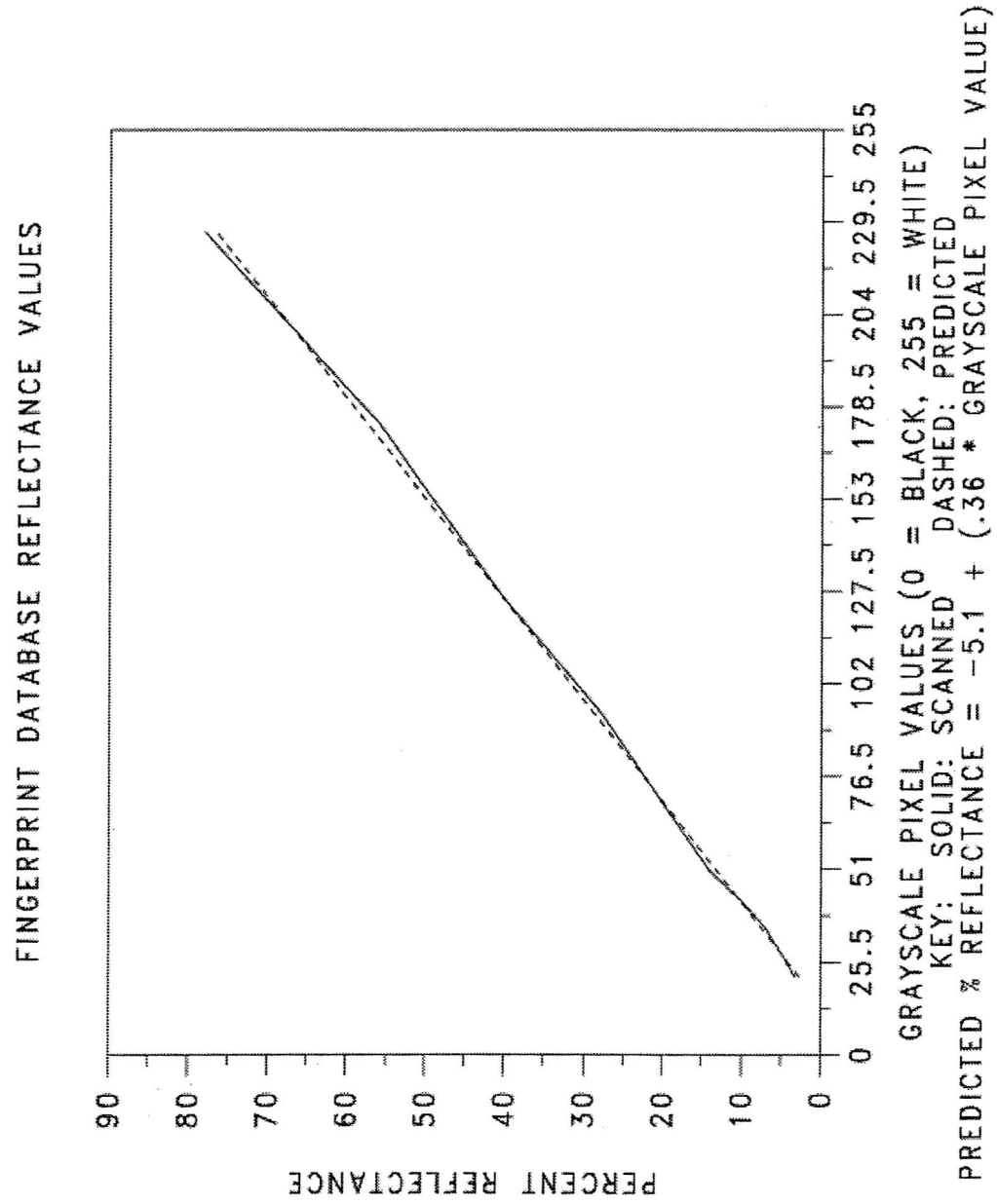
[11] GNU project - free UNIX-like utilities. Learn more at http://www.gnu.org.

[12] Cygwin tools - free GNU utility port for Win32 machines. Learn more at http://sourceware.cygnus.com/cygwin/.

[13] M.D. Garris, C.I. Watson, "NIST Fingerprint Image Software," CDROM and documentation, 2001.

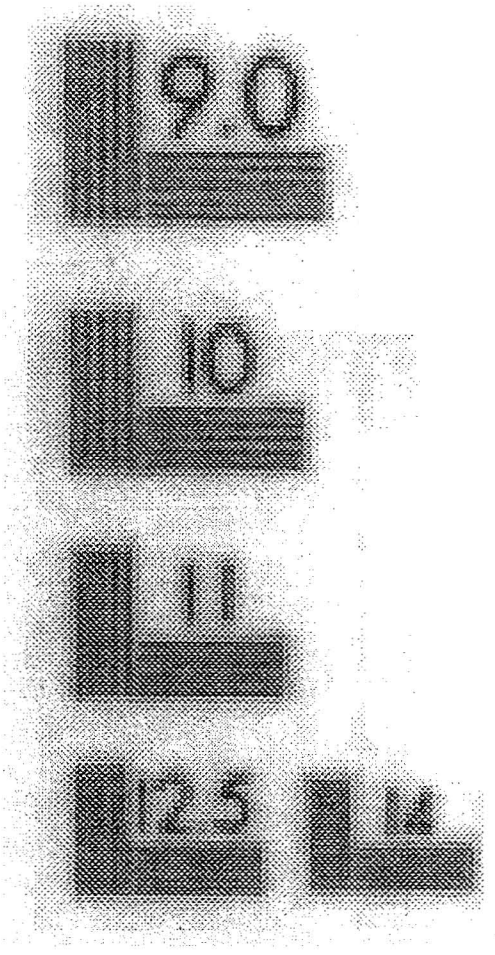
APPENDIX A. Database Reflectance and Resolution Calibration

The reflectance values for the fingerprint database, NIST Special Database 14, were calibrated using a reflection step table [4]. A plot of the reflectance values obtained using this step table is shown below. Also shown on the plot is an equation used to predict the reflectance of a given datapoint. The plot shows that this predicted reflectance closely follows the actual reflectance obtained using the reflection step table.



Database Resolution Calibration

The resolution of the scanner used to create the database was calibrated using a NBS 1010A resolution chart. A portion of this image is shown below (Figure B.2) after being magnified for viewing (Note: Printing has significantly reduced the quality of this image.). The scan of the table showed that the vertical resolution was approximately 10.5 line pairs/mm and the horizontal resolution was approximately 11.0 line pairs/mm.



APPENDIX B. Fingerprint Class Distribution Statistics

Natural Fingerprint Class Distribution

The data in this table give an estimate of the natural distribution of the listed fingerprint classes. This is from a very large sample of data given to NIST by the FBI and should give a good estimate of the "a priori" probabilities. See Figure 4 for the class codes.

Note:

UL (1-5 ridge count ulnar loops)
UM (6-30 ridge count ulnar loops)
UH (31-49 ridge count ulnar loops)
(L,M,H) also used for radial loops (R)

All whorl ridge tracings (I,M,O) for a whorl class are counted together (i.e. PI,PM,PO are counted in PW).

Class	Count	% of Total
UL	17063157	7.6638
UM	119530343	53.6860
UH	26312	0.0118
RL	3340194	1.5002
RM	5617095	2.5229
RH	3807	0.0017
PW	44989517	20.2066
CW	8727311	3.9198
DW	7890156	3.5438
XW	257264	0.1155
AA	8125907	3.6497
TT	6548416	2.9412
SR	187927	0.0844
XX	187927	0.1526
Total	222647130	

Class Distribution with Referenced Classes Not Included

The following data represent the class breakdown of NIST Special Database 14. The data totals are shown for all cards, file cards (F) and search cards (S). The data is based on the prime class of each print. Fingerprints with references are not included in the prime class totals. The number of referenced prints is given in the "RF" class. See Figure 4 for the class codes.

Note:

UL (1-5 ridge count ulnar loops)
UM (6-30 ridge count ulnar loops)
UH (31-49 ridge count ulnar loops)
(L,M,H) also used for radial loops (R)

All whorl ridge tracings (I,M,O) for a whorl class are counted together (i.e. PI,PM,PO are counted in PW).

Class	All Files	% of Total	F files	% of Total	S files	% of Total
UL	3452	6.9657	1680	6.8757	1772	7.0533
UM	27113	54.7107	13450	55.0462	13663	54.3844
UH	4	0.0081	2	0.0082	2	0.0080
RL	521	1.0513	256	1.0477	265	1.0548
RM	1087	2.1934	538	2.2018	549	2.1852
RH	1	0.0020	1	0.0041	0	0.0000
PW	11744	23.6980	5757	23.5614	5987	23.8308
CW	1440	2.9057	726	2.9713	714	2.8420
DW	1802	3.6362	889	3.6384	913	3.6341
XW	40	0.0807	20	0.0819	20	0.0796
AA	1764	3.5595	864	3.5361	900	3.5824
TT	544	1.0977	230	0.9413	314	1.2499
SR	45	0.0908	21	0.0859	24	0.0955
XX	0	0.0000	0	0.0000	0	0.0000
RF	4443		2566		1877	
Total	54000		27000		27000	

Distribution with Referenced Classes Included

The following data represent the class breakdown of NIST Special Database 14. The data is based on the prime class of each print without regard to references. The data totals are shown for all cards, file cards (F) and search cards (S). See Figure 4 for the class codes.

Note:

UL (1-5 ridge count ulnar loops)
UM (6-30 ridge count ulnar loops)
UH (31-49 ridge count ulnar loops)
(L,M,H) also used for radial loops (R)

All whorl ridge tracings (I,M,O) for a whorl class are counted together (i.e. PI,PM,PO are counted in PW).

Class	All Files	% of Total	F files	% of Total	S files	% of Total
UL	4136	7.6593	2128	7.8815	2008	7.4370
UM	27580	51.0741	13744	50.9037	13836	51.2444
UH	4	0.0074	2	0.0074	2	0.0074
RL	743	1.3759	386	1.4296	357	1.3222
RM	1192	2.2074	598	2.2148	594	2.2000
RH	2	0.0037	2	0.0074	0	0.0000
PW	12507	23.1611	6184	22.9037	6323	23.4185
CW	1918	3.5519	1020	3.7778	898	3.3259
DW	2155	3.9907	1096	4.0593	1059	3.9222
XW	54	0.1000	31	0.1148	23	0.0852
AA	1932	3.5778	972	3.6000	960	3.5556
TT	1732	3.2074	816	3.0222	916	3.3926
SR	45	0.0833	21	0.0778	24	0.0889
XX	0	0.0000	0	0.0000	0	0.0000
Total	54000		27000		27000	

Finger Class Distribution Comparison (Includes References)

The following data represent the class distribution comparison of NIST Special Database 14. The data is based on the prime class of each print without regard to references. The data shows the distribution for a "natural" dataset, which was calculated from a large data sample given to NIST by the FBI, and the fingerprints in NIST Special Database 14. The "natural" dataset should be large enough to give a good estimate of the "a priori" probabilities. The distribution for NIST Special Database 14 is shown for "all" the prints and for the prints from the file ("f") and search ("s") cards.

Class	Natural	% of Total	All files	% of Total	F files	% of Total	S files	% of Total
01	1184450	0.531985	277	0.513	166	0.615	111	0.411
02	2910419	1.307189	712	1.319	374	1.385	338	1.252
03	3611045	1.621869	916	1.696	456	1.689	460	1.704
04	4535306	2.036993	1038	1.922	519	1.922	519	1.922
05	4821937	2.165731	1193	2.209	613	2.270	580	2.148
06	4877860	2.190848	1263	2.339	627	2.322	636	2.356
07	5179856	2.326487	1392	2.578	708	2.622	684	2.533
08	6167910	2.770262	1495	2.769	729	2.700	766	2.837
09	6905017	3.101328	1576	2.918	808	2.993	768	2.844
10	8176735	3.672509	1838	3.404	915	3.389	923	3.419
11	9184468	4.125123	2157	3.994	1069	3.959	1088	4.030
12	10275618	4.615203	2347	4.346	1189	4.404	1158	4.289
13	10963356	4.924095	2399	4.443	1195	4.426	1204	4.459
14	10773739	4.838930	2462	4.559	1187	4.396	1275	4.722
15	9230044	4.145593	2164	4.007	1084	4.015	1080	4.000
16	9855852	4.426669	2089	3.869	1050	3.889	1039	3.848
17	7998642	3.592520	1713	3.172	858	3.178	855	3.167
18	5810787	2.609864	1361	2.520	672	2.489	689	2.552
19	4326753	1.943323	971	1.798	496	1.837	475	1.759
20	3409114	1.531174	852	1.578	422	1.563	430	1.593
21	2244371	1.008039	545	1.009	259	0.959	286	1.059
22	1596525	0.717065	376	0.696	191	0.707	185	0.685
23	1039230	0.466761	243	0.450	111	0.411	132	0.489
24	618620	0.277848	156	0.289	78	0.289	78	0.289
25	391637	0.175900	76	0.141	37	0.137	39	0.144
26	229292	0.102984	48	0.089	27	0.100	21	0.078
27	131863	0.059225	17	0.031	9	0.033	8	0.030
28	76709	0.034453	21	0.039	12	0.044	9	0.033
29	40995	0.018413	9	0.017	5	0.019	4	0.015
30	25350	0.011386	10	0.019	6	0.022	4	0.015
31	11772	0.005287	0	0.000	0	0.000	0	0.000
32	6754	0.003034	1	0.002	1	0.004	0	0.000
33	3468	0.001558	1	0.002	0	0.000	1	0.003

Class	Natural	% of Total	All files	% of Total	F files	% of Total	S files	% of Total
34	1877	0.000843	0	0.000	0	0.000	0	0.000
35	1065	0.000478	2	0.004	1	0.004	1	0.004
36	562	0.000252	0	0.000	0	0.000	0	0.000
37	268	0.000120	0	0.000	0	0.000	0	0.000
38	157	0.000071	0	0.000	0	0.000	0	0.000
39	91	0.000041	0	0.000	0	0.000	0	0.000
40	105	0.000047	0	0.000	0	0.000	0	0.000
41	74	0.000033	0	0.000	0	0.000	0	0.000
42	18	0.000008	0	0.000	0	0.000	0	0.000
43	20	0.000009	0	0.000	0	0.000	0	0.000
44	31	0.000014	0	0.000	0	0.000	0	0.000
45	19	0.000009	0	0.000	0	0.000	0	0.000
46	8	0.000004	0	0.000	0	0.000	0	0.000
47	5	0.000002	0	0.000	0	0.000	0	0.000
48	11	0.000005	0	0.000	0	0.000	0	0.000
49	7	0.000003	0	0.000	0	0.000	0	0.000
51	355982	0.159886	89	0.165	45	0.167	44	0.163
52	755477	0.339316	213	0.394	111	0.411	102	0.378
53	798428	0.358607	189	0.350	100	0.370	89	0.330
54	799378	0.359034	154	0.285	77	0.285	77	0.285
55	630929	0.283376	98	0.181	53	0.196	45	0.167
56	455045	0.204379	91	0.169	43	0.159	48	0.178
57	370298	0.166316	60	0.111	33	0.122	27	0.100
58	341372	0.153324	79	0.146	38	0.141	41	0.152
59	366137	0.164447	77	0.143	40	0.148	37	0.137
60	343091	0.154096	82	0.152	43	0.159	39	0.144
61	299996	0.134741	79	0.146	33	0.122	46	0.170
62	372938	0.167502	68	0.126	37	0.137	31	0.115
63	400723	0.179981	113	0.209	54	0.200	59	0.219
64	394334	0.177112	77	0.143	40	0.148	37	0.137
65	387294	0.173950	75	0.139	40	0.148	35	0.130
66	418185	0.187824	75	0.139	42	0.156	33	0.122
67	366353	0.164544	88	0.163	44	0.163	44	0.163
68	298189	0.133929	65	0.120	30	0.111	35	0.130
69	233616	0.104927	49	0.091	25	0.093	24	0.089
70	195708	0.087901	30	0.056	12	0.044	18	0.067
71	134253	0.060299	27	0.050	14	0.052	13	0.048
72	89073	0.040006	24	0.044	11	0.041	13	0.048
73	56760	0.025493	14	0.026	7	0.026	7	0.026
74	37274	0.016741	6	0.011	4	0.015	2	0.007
75	23134	0.010390	5	0.009	4	0.015	1	0.004
76	13765	0.006182	6	0.011	3	0.011	3	0.011
77	8436	0.003789	0	0.000	0	0.000	0	0.000
78	5281	0.002372	0	0.000	0	0.000	0	0.000

Class	Natural	% of Total	All files	% of Total	F files	% of Total	S files	% of Total
79	3253	0.001461	0	0.000	0	0.000	0	0.000
80	2587	0.001162	2	0.004	1	0.004	1	0.004
81	1270	0.000570	0	0.000	0	0.000	0	0.000
82	877	0.000394	2	0.004	2	0.007	0	0.000
83	551	0.000247	0	0.000	0	0.000	0	0.000
84	355	0.000159	0	0.000	0	0.000	0	0.000
85	275	0.000124	0	0.000	0	0.000	0	0.000
86	158	0.000071	0	0.000	0	0.000	0	0.000
87	75	0.000034	0	0.000	0	0.000	0	0.000
88	70	0.000031	0	0.000	0	0.000	0	0.000
89	37	0.000017	0	0.000	0	0.000	0	0.000
90	40	0.000018	0	0.000	0	0.000	0	0.000
91	37	0.000017	0	0.000	0	0.000	0	0.000
92	15	0.000007	0	0.000	0	0.000	0	0.000
93	8	0.000004	0	0.000	0	0.000	0	0.000
94	5	0.000002	0	0.000	0	0.000	0	0.000
95	5	0.000002	0	0.000	0	0.000	0	0.000
96	7	0.000003	0	0.000	0	0.000	0	0.000
97	5	0.000002	0	0.000	0	0.000	0	0.000
98	7	0.000003	0	0.000	0	0.000	0	0.000
99	10	0.000004	0	0.000	0	0.000	0	0.000
AA	8125907	3.649680	1932	3.578	972	3.600	960	3.556
TT	6548416	2.941163	1732	3.207	816	3.022	916	3.393
XX	339724	0.152584	0	0.000	0	0.000	0	0.000
SR	187927	0.084406	45	0.083	21	0.078	24	0.089
PI	17938974	8.057132	5129	9.498	2513	9.307	2616	9.689
CI	4629016	2.079082	930	1.722	503	1.863	427	1.581
DI	4585388	2.059487	1294	2.396	665	2.463	629	2.330
XI	100401	0.045094	15	0.028	9	0.033	6	0.022
PM	9045208	4.062576	2582	4.781	1305	4.833	1277	4.730
CM	270727	0.121595	53	0.098	31	0.115	22	0.081
DM	453692	0.203772	126	0.233	67	0.248	59	0.219
XM	67205	0.030185	15	0.028	8	0.030	7	0.026
PO	18005335	8.086938	4796	8.881	2366	8.763	2430	9.000
CO	3827568	1.719118	935	1.731	486	1.800	449	1.663
DO	2851076	1.280536	735	1.361	364	1.348	371	1.374
XO	89658	0.040269	24	0.044	14	0.052	10	0.037
Total	222647130		54000		27000		27000	

Referencing Information Tables

The following data gives information about referencing of the fingerprint classes in NIST Special Database 14. The totals are shown for all cards, file cards, and search cards. The primary class is the first class given followed by reference class(es). The only class/reference combinations given are ones that exist in the database; therefore all combinations not shown, do not exist in the database. It is also important to know that if a primary class has two references, the order of the references is not significant. That is, UL/TT/AA is the same as UL/AA/TT and, due to the order of search through the classes for references, both would be included in the UL/AA/TT total. Scars and amputations are treated differently in that a count of the total number is given at the end of the list.

Note:

UL (1-5 ridge count ulnar loops)
UM (6-30 ridge count ulnar loops)
UH (31-49 ridge count ulnar loops)
(L,M,H) also used for radial loops (R)

All whorl ridge tracings (I,M,O) for a whorl class are counted together (i.e. PI,PM,PO are counted in PW).

Class/Reference	All Cards	File Cards	Search Cards
UL	3452	1680	1772
UL/UM	101	88	13
UL/PW	3	1	2
UL/CW	5	1	4
UL/TT	539	339	200
UL/UM/TT	5	4	1
UL/RL/AA	1	1	-
UL/CW/TT	1	-	1
UL/AA/TT	2	2	-
UM	27113	13450	13663
UM/UL	47	42	5
UM/PW	1	1	-
UM/CW	201	107	94
UM/DW	59	39	20
UM/XW	1	-	1
UM/TT	9	6	3
UM/UL/TT	2	2	-
UM/PW/CW	3	2	1
UM/CW/DW	11	8	3
UM/UL/CW/DW/XW	1	1	-
UH	4	2	2
RL	521	256	265
RL/RM	11	6	5
RL/CW	9	6	3
RL/DW	1	-	1

Class/Reference	All Cards	File Cards	Search Cards
RL/TT	189	115	74
RL/PW/CW	1	-	1
RM	1087	538	549
RM/RL	4	4	-
RM/PW	1	1	-
RM/CW	59	34	25
RM/DW	7	4	3
RM/XW	5	2	3
RM/TT	3	1	2
RM/RL/TT	1	-	1
RM/PW/DW	1	-	1
RM/CW/DW	2	2	-
RM/DW/XW	1	-	1
RH	1	1	-
RH/DW	1	1	-
PW	11744	5757	5987
PW/UL	2	1	1
PW/UM	7	2	5
PW/CW	289	150	139
PW/DW	420	247	173
PW/XW	12	9	3
PW/UM/CW	2	1	1
PW/UM/DW	2	1	1
PW/RL/XW	1	1	-
PW/CW/DW	6	3	3
PW/CW/TT	1	1	-
PW/DW/XW	6	4	2
CW	1440	726	714
CW/UL	3	2	1
CW/UM	126	80	46
CW/RL	1	1	-
CW/RM	33	16	17
CW/PW	265	167	98
CW/DW	31	15	16
CW/UM/PW	2	1	1
CW/UM/DW	6	4	2
CW/RM/DW	1	1	-
CW/PW/DW	3	3	-
CW/DW/XW	2	2	-
DW	1802	889	913
DW/UL	1	1	-

Class/Reference	All Cards	File Cards	Search Cards
DW/UM	36	18	18
DW/RM	5	1	4
DW/PW	258	160	98
DW/CW	16	8	8
DW/XW	12	5	7
DW/UM/CW	6	4	2
DW/UM/XW	3	3	-
DW/RM/CW	1	-	1
DW/PW/CW	1	-	1
DW/PW/XW	2	1	1
DW/CW/XW	1	1	-
DW/CW/PW/UM	1	-	1
DW/CW/PW/UL/UM	1	1	-
XW	40	20	20
XW/UM	3	2	1
XW/RM	2	2	-
XW/PW	3	1	2
XW/DW	5	5	-
XW/UL/DW	1	1	-
AA	1764	864	900
AA/TT	162	105	57
AA/UL/TT	4	3	1
AA/RL/TT	1	-	1
TT	544	230	314
TT/UL	644	300	344
TT/UM	3	2	1
TT/RL	270	138	132
TT/RM	5	3	2
TT/CW	1	1	-
TT/XW	1	-	1
TT/AA	191	93	98
TT/UL/RL	1	1	-
TT/UL/CW	1	1	-
TT/UL/AA	50	36	14
TT/RL/RM	1	-	1
TT/RL/CW	1	1	-
TT/RL/AA	5	4	1
TT/RL/XW/DW	1	-	1
TT/UL/XW/DW	2	1	1
TT/RL/CW/DW	1	-	1
Total Scars	276	152	124

APPENDIX C. Software Manual Pages.

NAME

cjpegb – compresses a grayscale or color (RGB) image using *lossy* Baseline JPEG (JPEGB).

SYNOPSIS

cjpegb *<q=20=95> <outext> <image file>*
[**-raw_in** *w,h,d,[ppi]*
[**-nonintrlv**]]
[*comment file*]

DESCRIPTION

Cjpegb takes as input a file containing an uncompressed grayscale or color (RGB) image. Two possible input file formats are accepted, NIST IHead files and raw pixmap files. If a raw pixmap file is to be compressed, then its image attributes must be provided on the command line as well. Once read into memory, the grayscale or color pixmap is then *lossy* compressed to a specified level of reconstruction quality using the Independent JPEG Group's (IJG) library for Baseline JPEG (JPEGB). The JPEGB results are then written to an output file.

Note that **cjpegb** calls the IJG library in a default color mode where one of the compression steps includes a colorspace conversion from RGB to YCbCr, and then the Cb & Cr component planes are downsampled by a factor of 2 in both dimensions. Due to this colorspace conversion, **cjpegb** should only be used to compress RGB color images.

The color components of RGB pixels in a raw pixmap file may be interleaved or non-interleaved. By default, **cjpegb** assumes interleaved color pixels. (See INTERLEAVE OPTIONS below.) Regarding color pixmaps, the NIST IHead file format only supports interleaved RGB images.

OPTIONS

All switch names may be abbreviated; for example, **-raw_in** may be written **-r**.

<q=20-95>
specifies the level of quality in the reconstructed image as a result of lossy compression. The integer quality value may range between 20 and 95. The lower the quality value, the more drastic the compression.

<outext>
the extension of the compressed output file. To construct the output filename, **cjpegb** takes the input filename and replaces its extension with the one specified here.

<image file>
the input file, either an IHead file or raw pixmap file, containing the grayscale or color (RGB) image to be compressed.

-raw_in *w,h,d,[ppi]*
the attributes of the input image. This option must be included on the command line if the input is a raw pixmap file.

w the pixel width of the pixmap

h the pixel height of the pixmap

d the pixel depth of the pixmap

ppi the optional scan resolution of the image in integer units of pixels per inch.

-nonintrlv
specifies that the color components in an *input* raw pixmap file image are non-interleaved and stored in separate component planes. (See INTERLEAVE OPTIONS below).

comment file
an optional user-supplied ASCII comment file. (See COMMENT OPTIONS below.)

INTERLEAVE OPTIONS

The color components of RGB pixels in a raw pixmap file may be interleaved or non-interleaved. Color components are interleaved when a pixel's (R)ed, (G)reen, and (B)lue components are sequentially adjacent in the image byte stream, ie. RGBRGBRGB... . If the color components are non-interleaved, then all (R)ed components in the image are sequentially adjacent in the image byte stream, followed by all (G)reen components, and then lastly followed by all (B)lue components. Each complete sequence of color components is called a *plane*. The utilities **intr2not** and **not2intr** convert between interleaved and non-interleaved color components. By default, **cjpegb** assumes interleaved color components, and note that all color IHead images must be interleaved.

COMMENT OPTIONS

Upon successful compression, this utility generates and inserts in the compressed output file a specially formatted comment block, called a NISTCOM. A NISTCOM is a text-based attribute list comprised of (name, value) pairs, one pair per text line. The first line of a NISTCOM always has name = "NIST_COM" and its value is always the total number of attributes included in the list. The utility **rdjpgcom** scans a JPEG compressed file for any and all comment blocks. Once found, the contents of each comment block is printed to standard output. Using this utility, the NISTCOM provides easy access to relevant image attributes. The following is an example NISTCOM generated by **cjpegb**:

```
NIST_COM 12
PIX_WIDTH 768
PIX_HEIGHT 1024
PIX_DEPTH 24
PPI -1
LOSSY 1
COLORSPACE YCbCr
NUM_COMPONENTS 3
HV_FACTORS 2,2:1,1:1,1
INTERLEAVE 1
COMPRESSION JPEGB
JPEGB_QUALITY 50
```

Cjpegb also accepts an optional comment file on the command line. If provided, the contents of this file are also inserted into the compressed output file. If the comment file is a NISTCOM attribute list, then its contents are merged with the NISTCOM internally generated by **cjpegb** and a single NISTCOM is written to the compressed output file. Note that **cjpegb** gives precedence to internally generated attribute values. If the user provides a non-NISTCOM comment file, then the contents of file are stored to a separate comment block in the output file. Using these comment options enables the user to store application-specific information in a JPEG file.

EXAMPLES

From *test/imgtools/execs/cjpegb/cjpegb.src*:

% cjpegb 50 jpb face08.raw -r 768,1024,8
compresses a grayscale face image in a raw pixmap file.

% cjpegb 50 jpb face24.raw -r 768,1024,24
compresses a color face image in a raw pixmap file.

SEE ALSO

cjpeg(1E), **cjpegl**(1D), **djpegb**(1D), **dpyimage**(1D), **intr2not**(1D), **jpegtran**(1E), **not2intr**(1D), **rdjpgcom**(1E), **wrjpgcom**(1E)

AUTHOR

NIST/ITL/DIV894/Image Group

NAME

dpyimage – displays the image contents of Baseline JPEG, Lossless JPEG, WSQ, IHead, and raw pixmap files.

SYNOPSIS

dpyimage [*options*] *image-file ...*

-r *w,h,d,wp*
-A
-s *n*
-a *n*
-v
-x
-b *n*
-N *n*
-O
-k
-W *n*
-H *n*
-X *n*
-Y *n*
-n
-T *title*
-t
-D *dir*
-d *display*

DESCRIPTION

Dpyimage reads various image file formats, decompresses and reconstructs pixmaps as needed, and displays image contents in an X11 window. Supported file formats include Baseline JPEG (lossy), Lossless JPEG, WSQ (lossy), NIST IHead, and raw pixmap files. Raw pixmaps containing either grayscale or interleaved RGB color pixels are supported. This utility automatically differentiates between these different formats.

If only one file (or the *-n* option) is specified on the command line, the image or images are simply read from disk and then displayed. If multiple files are specified, **dpyimage** attempts to minimize the display waiting time by forking a background process to pre-read images from disk. By default, the child transfers images to the parent via a pipe. This always allows at least one image to be read in from disk while the user is viewing the current image. Since a process writing on a pipe is blocked (until a read on the other end of the pipe) after transferring four kilobytes, the child will only be one image ahead of the parent except when handling smaller images.

If the *-t* option appears on the command line, the processes use temporary files as the means of exchanging image data. Therefore, the child is not constrained on the number of images it may pre-read for the parent. However, the filesystem on which the directory for temporary files resides must have enough space for copies of all images in uncompressed state or an error may occur. This is the suggested mode for viewing compressed images for which decompression takes considerably longer than disk I/O.

If the image is too large to be displayed on the screen, the upper lefthand corner will be displayed and the rest of the image can be moved into view by holding down a mouse button, moving in the direction desired, and then releasing the button. Button presses when another button(s) is already down and button releases when another button(s) is still down are ignored.

Users may exit from the program by striking keys 'x' or 'X'. Advancing to the next image is accomplished by any other keystroke.

OPTIONS

-r *w,h,d,wp*
raw pixmap attributes:
w - pixel width,
h - pixel height,
d - pixel depth,
wp - white pixel value
bi-level wp=0|1
grayscale wp=0|255
RGB wp=0 (value ignored)

-A automatically advances through images.

-s *n* in automatic mode, sleeps *n* seconds before advancing to the next image [2].

-a *n* sets drag accelerator to *n* — changes in pointer position will result in *n* shifts in the displayed image [1].

-v turns on verbose output.

-x turns on debug mode, causing a core dump when an X11 error occurs.

-b *n* sets border width to *n* pixels [4].

-N *n* the child I/O process is niced to level *n*.

-O overrides the redirect on windows (no window manager).

-k informs utility that there is no keyboard input.

-W *n* displays image in a window of width *n* pixels.

-H *n* displays image in a window of height *n* pixels.

-X *n* positions image window with top-left corner *n* pixels to the right of the display's top-left corner [0].

-Y *n* positions image window with top-left corner *n* pixels below the display's top-left corner [0].

-n does not fork to display multiple images.

-T *title* sets window name to *title* [*file*]. **-t uses temporary files to transfer multiple images to parent [via pipe].**

-D *directory*
creates temporary files in *directory* [/tmp].

-d *display*
connects to alternate display.

image-file ...
one or more image files whose pixmaps are to be displayed.

ENVIRONMENT

If the environment variable **TMPDIR** is set and the **-D** option is not set on the command line, **dpyimage** uses this directory as the location for temporary files.

EXAMPLES

From *test/imgtools/execs/dpyimage/dpyimage.src*:

% dpyimage -r 500,500,8,255 ../../data/finger/gray/raw/finger.raw
displays a fingerprint image from a raw pixmap file.

% dpyimage ../../data/finger/gray/jpegl/finger.jpl
displays a reconstructed fingerprint image from a Lossless JPEG file.

% dpyimage ../../data/finger/gray/wsq/finger.wsq
displays a reconstructed fingerprint image from a WSQ file.

% dpyimage ../../data/face/gray/jpegb/face.jpb
displays a reconstructed grayscale face image from a Baseline JPEG file.

% dpyimage -r 768,1024,24,0 ../../data/face/rgb/raw/intrlv/face.raw
displays a color face image from a raw pixmap file.

% dpyimage ../../data/face/rgb/jpegb/face.jpb
displays a reconstructed color face image from a Baseline JPEG file.

% dpyimage ../../data/face/rgb/jpegl/face.jpl
displays a reconstructed color face image from a Lossless JPEG file.

SEE ALSO

an2ktool(1C), **cjpegb**(1D), **cjpegl**(1D), **cwsq**(1D), **djpegb**(1D), **djpegl**(1D), **dpyan2k**(1C), **dwsq**(1D)

AUTHOR

NIST/ITL/DIV894/Image Group

NAME

dumpihdr – takes NIST IHEAD image files and prints their header content to stdout

SYNOPSIS

dumpihdr *ihdrfile* ...

DESCRIPTION

Dumpihdr opens NIST IHEAD rasterfiles and formats and prints their header contents to stdout.

OPTIONS

ihdrfile any NIST IHEAD image file name

EXAMPLES

dumpihdr foo.pct bar.pct

FILES

include/ihead.h NIST's raster header include file

DIAGNOSTICS

Dumpihdr exits with a status of -1 if opening an **ihdrfile** fails.

BUGS

NAME

dwsq – decompresses a WSQ-encoded grayscale fingerprint image.

SYNOPSIS

dwsq *<outext>* *<image file>* **[-raw_out]**

DESCRIPTION

Dwsq takes as input a file containing a WSQ-compressed grayscale fingerprint image. Once read into memory, the lossy-compressed pixmap is decoded and reconstructed using Wavelet Scalar Quantization as described in the FBI's Criminal Justice Information Services (CJIS) document, "WSQ Gray-scale Fingerprint Compressions Specification," Dec. 1997. This is the only fingerprint compression format accepted by the FBI IAFIS system.

Upon completion, two different output image file formats are possible, a NIST IHead file (the default) or a raw pixmap file (specified by the **-raw_out** flag). In addition, a specially formatted text file, called a NISTCOM, is created with extension ".ncm". The NISTCOM file contains relevant image attributes associated with the decoded and reconstructed output image. (See NISTCOM OUTPUT below.)

OPTIONS

All switch names may be abbreviated; for example, **-raw_out** may be written **-r**.

<outext>

the extension of the decompressed output file. To construct the output filename, **dwsq** takes the input filename and replaces its extension with the one specified here.

<image file>

the input WSQ file to be decompressed.

-raw_out

specifies that the decoded and reconstructed image should be stored to a raw pixmap file.

NISTCOM OUTPUT

Upon successful completion, **dwsq**, creates a specially formatted text file called a NISTCOM file. A NISTCOM is a text-based attribute list comprised of (name, value) pairs, one pair per text line. The first line of a NISTCOM always has name = "NIST_COM" and its value is always the total number of attributes included in the list. These attributes are collected and merged from two different sources to represent the history and condition of the resulting reconstructed image. The first source is from an optional NISTCOM comment block inside the WSQ-encoded input file. This comment block can be used to hold user-supplied attributes. The WSQ encoder, **cwsq**, by convention inserts one of these comment blocks in each compressed output file it creates. (The utility **rdwsqcom** can be used to scan a WSQ file for any and all comment blocks.) The second source of attributes comes from the decompression process itself. In general, attribute values from this second source are given precedence over those from the first.

The NISTCOM output filename is constructed by combining the basename of the input WSQ file with the extension ".ncm". By creating the NISTCOM file, relevant attributes associated with the decoded and reconstructed image are retained and easily accessed. This is especially useful when dealing with raw pixmap files and creating image archives. The following is an example NISTCOM generated by **dwsq**:

```
NIST_COM 7
PIX_WIDTH 500
PIX_HEIGHT 500
PIX_DEPTH 8
PPI 500
LOSSY 1
COLORSPACE GRAY
```

EXAMPLES

From *test/imgtools/execs/dwsq/dwsq.src*:

% dwsq raw finger.wsq -r

decompresses a WSQ-encoded fingerprint image and stores the reconstructed image to a raw pixmap file. Note the NISTCOM file, **finger.ncm**, is also created.

SEE ALSO

cwsq(1D), **dpyimage**(1D), **rdwsqcom**(1D), **wrwsqcom**(1D)

AUTHOR

NIST/ITL/DIV894/Image Group

NAME

dwsq14v2 – decompresses a WSQ-encoded grayscale fingerprint image and formats the IHEAD structure of the uncompressed image to be the same as the original version of NIST Special Database 14.

SYNOPSIS

dwsq14v2 *<outext>* *<image file>* [**-raw_out**]

DESCRIPTION

Dwsq14v2 takes as input a file containing a WSQ-compressed grayscale fingerprint image. Once read into memory, the lossy-compressed pixmap is decoded and reconstructed using Wavelet Scalar Quantization as described in the FBI's Criminal Justice Information Services (CJIS) document, "WSQ Gray-scale Fingerprint Compressions Specification," Dec. 1997. This is the only fingerprint compression format accepted by the FBI IAFIS system.

Upon completion, two different output image file formats are possible, a NIST IHead file (the default) or a raw pixmap file (specified by the **-raw_out** flag). In addition, a specially formatted text file, called a NISTCOM, is created with extension ".ncm". The NISTCOM file contains relevant image attributes associated with the decoded and reconstructed output image. (See NISTCOM OUTPUT below.)

OPTIONS

All switch names may be abbreviated; for example, **-raw_out** may be written **-r**.

<outext>

the extension of the decompressed output file. To construct the output filename, **dwsq14v2** takes the input filename and replaces its extension with the one specified here.

<image file>

the input WSQ file to be decompressed.

-raw_out

specifies that the decoded and reconstructed image should be stored to a raw pixmap file.

NISTCOM OUTPUT

Upon successful completion, **dwsq14v2**, creates a specially formatted text file called a NISTCOM file. A NISTCOM is a text-based attribute list comprised of (name, value) pairs, one pair per text line. The first line of a NISTCOM always has name = "NIST_COM" and its value is always the total number of attributes included in the list. These attributes are collected and merged from two different sources to represent the history and condition of the resulting reconstructed image. The first source is from an optional NISTCOM comment block inside the WSQ-encoded input file. This comment block can be used to hold user-supplied attributes. The WSQ encoder, **cwsq**, by convention inserts one of these comment blocks in each compressed output file it creates. (The utility **rdwsqcom** can be used to scan a WSQ file for any and all comment blocks.) The second source of attributes comes from the decompression process itself. In general, attribute values from this second source are given precedence over those from the first.

The NISTCOM output filename is constructed by combining the basename of the input WSQ file with the extension ".ncm". By creating the NISTCOM file, relevant attributes associated with the decoded and reconstructed image are retained and easily accessed. This is especially useful when dealing with raw pixmap files and creating image archives. The following is an example NISTCOM generated by **dwsq14v2**:

```
NIST_COM 12
SD_ID 14
HISTORY f0000001.pct 20 tape3.t1116010.01 4096x1536
FING_CLASS R
SEX m
SCAN_TYPE i
```

```
PIX_WIDTH 832
PIX_HEIGHT 768
PIX_DEPTH 8
PPI 500
LOSSY 1
COLORSPACE GRAY
```

SEE ALSO

cwsq(1D), **dpyimage**(1D), **rdwsqcom**(1D), **wrwsqcom**(1D)

AUTHOR

NIST/ITL/DIV894/Image Group

NAME

rdwsqcom – scans a WSQ-encoded image file for any and all comment blocks, printing their contents to standard output.

SYNOPSIS

rdwsqcom <*image file*>

DESCRIPTION

Rdwsqcom takes as input a file containing a WSQ-compressed image, and *without* decoding and reconstructing the image, the utility scans the file for any and all comment blocks. As a comment block is encountered, its contents is printed to standard output. Comments can be written to a WSQ file by using the **wrwsqcom** command.

OPTIONS

<*image file*>

the input WSQ file to be scanned.

EXAMPLES

From *test/imgtools/execs/rdwsqcom/rdwsqcom.src*:

% **rdwsqcom finger.wsq > finger.com** prints any comments stored in the WSQ fingerprint file to an output file.

SEE ALSO

cwsq(1D), **wrwsqcom**(1D)

AUTHOR

NIST/ITL/DIV894/Image Group