NIST Special Database 30
Dual Resolution Images from Paired Fingerprint Cards

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

Dual Resolution Images from Paired Fingerprint Cards

C. I. Watson

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

1. INTRODUCTION

This new NIST fingerprint database offers the user complete paired fingerprint cards that include all ten rolled fingerprints and the plain impressions at the bottom of the card scanned at both 19.7 ppmm (500 ppi) and 39.4 ppmm (1000 ppi). Paired fingerprint cards are two sets of fingerprints for one individual captured at different dates. This database allows a user to compare algorithm results on two resolutions of the same image and specifically for adjusting the WSQ compression algorithm to work with 39.4 ppmm images. This database has 36 paired fingerprint cards scanned at both resolutions and segmented into individual fingerprint images. The fingerprint cards scanned at 19.7 ppmm are stored on the first CDROM and the 39.4 ppmm scanned cards are stored on the other three CDROMs (12 cards per CDROM). The segmented images are compressed using lossless JPEG (JPEGL) compression and stored in the ANSI/NIST data format[6]. Reference information is included in comment fields to help reconstruct the fingerprint card image if desired as well as a human defined classification for each fingerprint image.

Software is included to access the data directly from the ANSI/NIST format or to parse the images into individual compressed files. The included source code is written in 'C', and has been developed to compile and execute under the Linux operating system[9] using the GNU `gcc` compiler and `gmake` utility[10]. The source code may also be installed on Win32 platforms that have the Cygwin[11] library and associated tools installed[1].

The 19.7 ppmm cards are a subset of the original data (before WSQ compression) used in NIST Special Database 29. File names in this database are not sequentially numbered but they do contain the same persons fingerprints for same named files in SD29. Many of the fingerprint cards in this database were previously used to make NIST Special Database 4 [2]. Links are given so users that may have previously done testing using SD4 can compare to results from this different scanned version of the data if desired.

2. DATABASE CONTENT AND ORGANIZTION

2.1 CDROM Layout

The top level of the directory structure has four directories: **doc**, **man, src** and **data**. The PDF version of this document is included in **doc** and manual/help pages for the software commands are included in **man.** The **src** directory and its subdirectories contain software that can access the ANSI/NIST formatted data. Since space was available the **doc**, **man**, and **src** directories appear on all four CDROMs. The **data** directory contains the paired fingerprint cards stored in ANSI/NIST format stored in subdirectories of 500 for the 19.7 ppmm images and 1000 for the 39.4 ppmm

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

images. The first CDROM contains the 19.7 ppmm images and the last three contain the 39.4 ppmm images. The filenames (a and b for the first characters in the pair) have numbers that match the sequence used in Special Database 29 with the extension ".an2". Since this data is only a subset from SD29 the file numbers are not sequential. All images in the ANSI/NIST records are compressed using JPEGL compression. Figure 1 shows the fingerprint card image for file a042.an2 from the database.

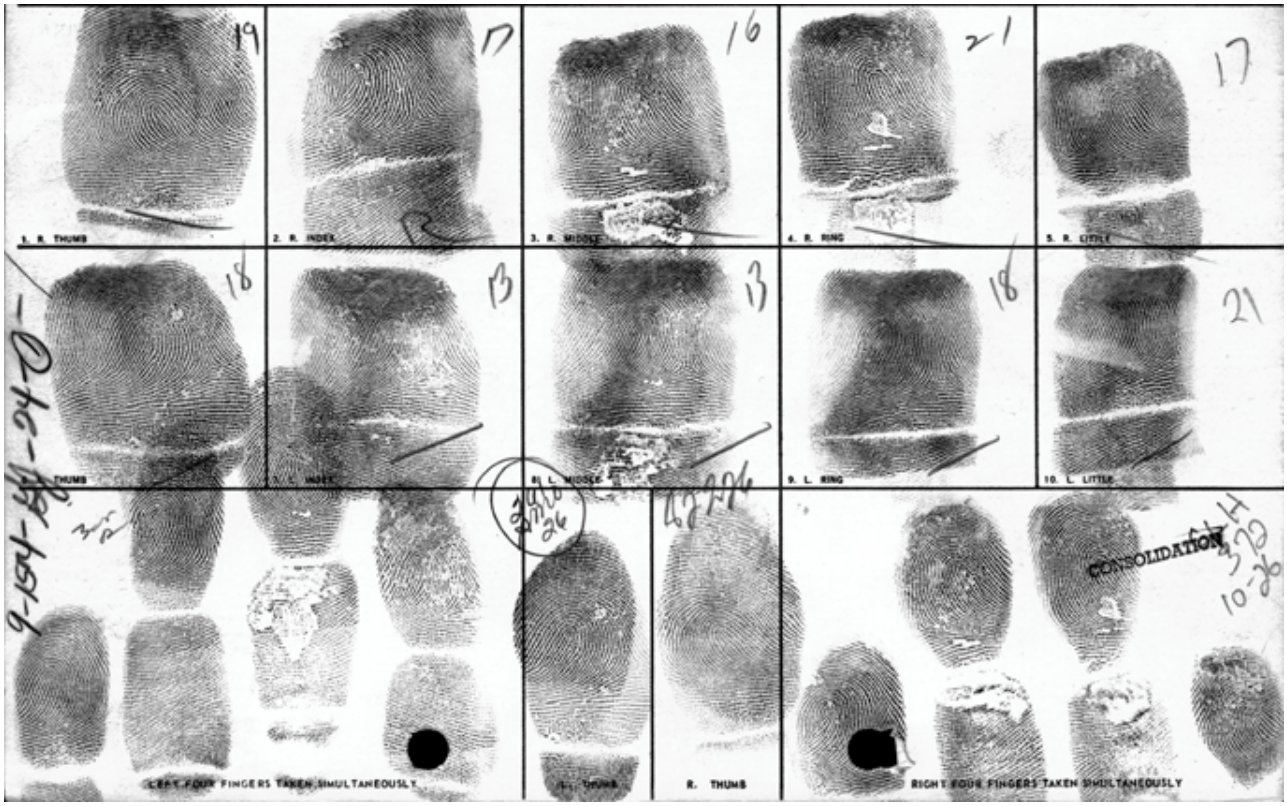

Figure 1 Image of fingerprint card a042.an2 from the database.

2.2 Fingerprints from NIST Special Database 4

Many of the fingerprint cards used to make this database are from the set used to make NIST Special Database 4 (SD4)[2]. The file **doc/sd4_db30.log** contains a complete list of the 316 fingerprint images from SD4 that are contained in one of the fingerprint cards in this new database. The information is provided for users that have done previous testing with images from SD4. They can compare those results to the same images captured using a different scanner/resolution.

3. DATA FORMAT

3.1 Card Segmentation

This data came from an archive of fingerprint cards that contained several varieties of fingerprint card styles. The fingerprint boxes on the cards varied in position and style as shown when comparing Figure 1 and Figure 2. Segmentation was done by detecting the "fingerprint layout"

independently for each fingerprint card. Since each fingerprint card layout was different, the sizes of the segmented images also varied. Figure 3 shows the rolled images for the left and right thumbs and the left four plain impressions as segmented from the fingerprint card in Figure 2. The 14 segmented images contain all the pixel data from the original fingerprint card, so a user could reconstruct the card and resegment the data. An adaptive segmentation would be most useful for the plain impressions at the bottom of the card because the fingerprint impressions occasionally crossed over the boxed area used for simple template segmentation as shown in Figure 4 (file b027.an2).

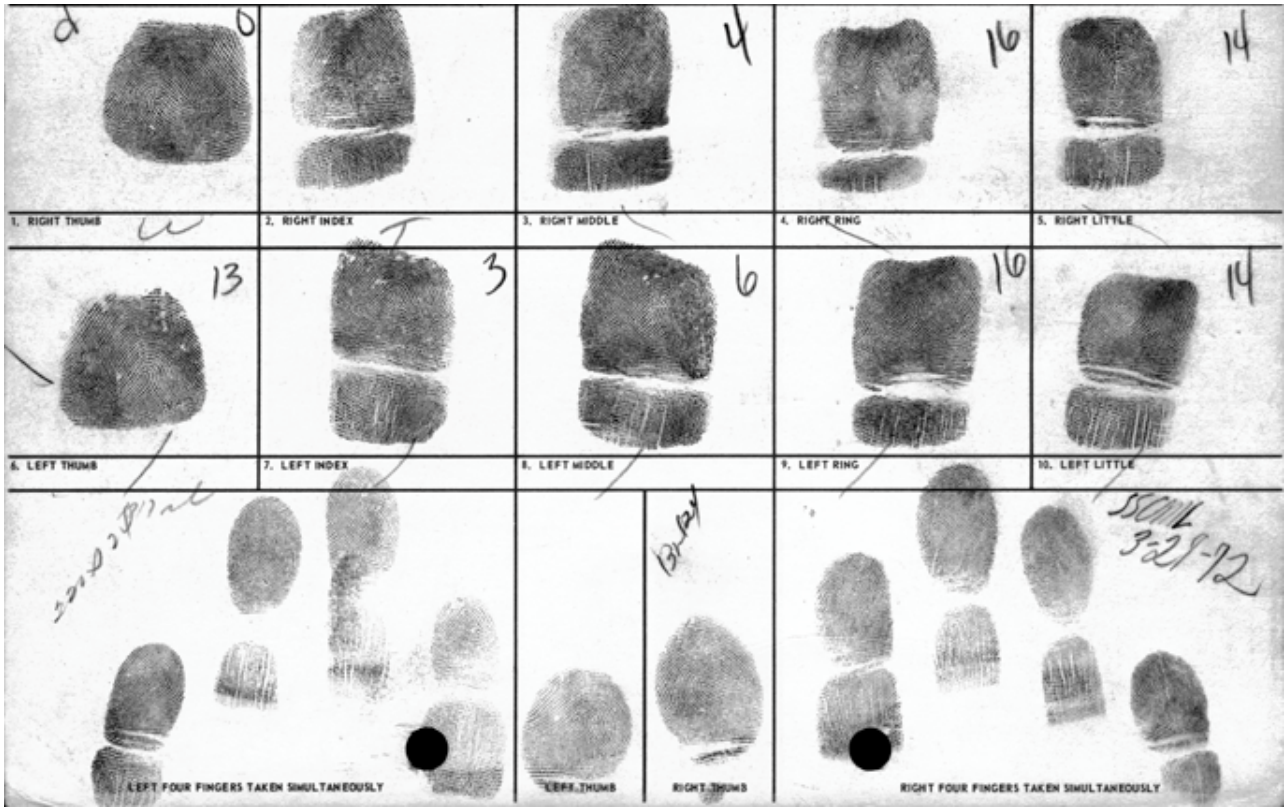

Figure 2 Fingerprint card a021.an2, shows a different card format to segment.

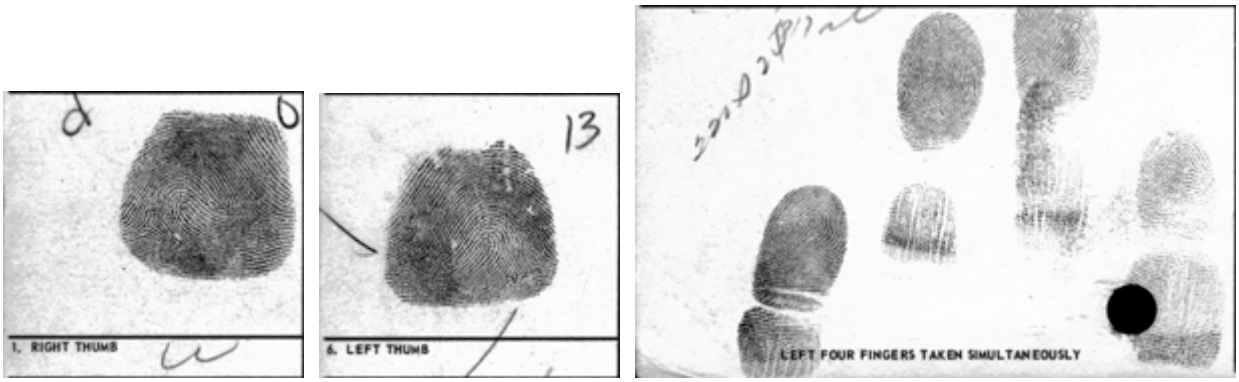

Figure 3 Segmented images for right thumb, left thumb, and the four left plain impressions.

Figure 4 Image where segmentation cuts through the inked fingerprint.

3.2 ANSI/NIST Format

The fingerprint cards are stored using the ANSI/NIST type-14 record format. Each fingerprint card is contained in one ANSI/NIST formatted file that has one type-1 record (required by standard) and fourteen type-14 records containing the ten rolled fingerprint images and the four sets of plain impressions. The four fingers from each hand are contained in one set of plain impressions (Figure 3 and Figure 4) and the users are left to segment them on their own. Table 1 shows the layout of the type-1 record and Table 2 shows the layout for the type-14 record (Table 16 in the standard)[6]. APPENDIX A contains a printout of the ANSI/NIST data for the 19.7 ppmm (500 ppi) version of file a021.an2.

For the type-1 record the optional fields (1.010, 1.013-1.015) are not used in the database. The following section describes the information in the mandatory fields of the type-1 record. Field 1.001 is the total byte length of the record and 1.002 is the version number defined by the standard to be "0300". Field 1.003 gives a listing of all records present in the ANSI/NIST file and it starts with a 1 to identify the type-1 record. The next information item in 1.003 is the number of other records present in the file, 14 for the full fingerprint card. Finally, for each subsequent record in the file its type is listed (all are type-14 records for the database) along with a unique IDC (Image Designation Character), which is referenced in the type-14 records. For the database, field 1.004 is defined to be "NISTDATA" and 1.005 is the date the ANSI/NIST file was made. Field 1.006 is the priority assigned to the transaction which was arbitrarily for the database set to "1". The DAI in field 1.007 was again arbitrarily set to "DAI00000" as was the ORI in field 1.008 set to "MDNISTIG". Field 1.009 contains the name of the fingerprint card file as the TCN. Finally, fields 1.011 and 1.012 were set to the scanning resolution of "19.69" or "39.4". Because all the ANSI/NIST files in the database contain the complete fingerprint card, the only variations in the type-1 records are in field 1.009 all other fields are the same (they were all created on the same day).

Table 1 – Type-1 transition information record

Ident	Cond code	Field number	Field name	Char type
LEN	M	1.001	LOGICAL RECORD LENGTH	N
VER	M	1.002	VERSION NUMBER	N
CNT	M	1.003	FILE CONTENT	N
TOT	M	1.004	TYPE OF TRANSACTION	AN
DAT	M	1.005	DATE	N
PRY	M	1.006	PRIORITY	N
DAI	M	1.007	DESTINATION AGENCY IDENTIFIER	AN
ORI	M	1.008	ORIGINATING AGENCY IDENTIFIER	AN
TCN	M	1.009	TRANSACTION CONTROL NUMBER	AN
TCR	O	1.010	TRANSACTION CONTROL REFERENCE	AN
NSR	M	1.011	NATIVE SCANNING RESOLUTION	N
NTR	M	1.012	NOMINAL TRANSMITTING RESOLUTION	N
DOM	O	1.013	DOMAIN NAME	AN
GMT	O	1.014	GRENNWICH MEAN TIME	AN
DCS	O	1.015	DIRECTORY OF CHARACTER SETS	AN

Key: M = Mandatory; O = Optional; N = Numeric; A = Alphabetic; AN = Alphanumeric

For the type-14 record, optional fields 14.014-14.199 and 14.204-14.998 are not used. Field 14.001 is the total byte length of the record and 14.002 is the IDC that was given in field 1.003 when defining the number and type of records present in the ANSI/NIST file. Field 14.003 is the impression type from the image, defined by the standard as "3" for "nonlive-scan rolled" and "2" for "nonlive-scan plain." The source agency in 14.004 is set to "MDNISTIG", the same value as 1.008 and the capture date is set in field 14.005. Fields 14.006 and 14.007 are the horizontal and vertical pixel size of the image. The scale units (14.008) is defined by the standard as "2" for pixels per centimeter, so the values of 14.009 and 14.010 are "197" or "394." The compression algorithm (14.011) is set to "JPEGL" and field 14.012 is "8" bits per pixel. The finger positions (14.013) are defined in the standard as described in Table 3.

The standard allows user defined comments in fields 14.200-14.998. In this database, reference information is stored in fields 14.200-14.203. First, field 14.200 contains the pixel size of the fingerprint card image before segmentation and 14.201 has the (x,y) location for the upper/left corner of the fingerprint in the fingerprint card image. Field 14.202 contains the classification for the fingerprint image. The classes used are from the ANSI/NIST standard as shown below in Table 4. The plain impressions that contain four fingers from a hand have four classes (subfields) in this field. In addition to the primary classification, each class may also have one reference class associated with it. File a021.an2 shown in APPENDIX A has fingers with reference classes. Finally, field 14.203 contains the sex of the individual that contributed the fingerprints.

The last field in the record contains the actual JPEGL compressed fingerprint image. This image can be stored in a file by itself and tools are provided to parse all the images from the ANSI/NIST formatted file. The reference information is repeated in a comment field in the JPEGL compressed image so that information is not lost.

Table 2 – Type-14 variable-resolution tenprint record layout

Ident	Cond code	Field number	Field name	Char type	Field size per occurrence		Occur count		Max byte count
					min	max	min	Max	
LEN	M	14.001	LOGICAL RECORD LENGTH	N	4	8	1	1	15
IDC	M	14.002	IMAGE DESIGNATION CHARACTER	N	2	5	1	1	12
IMP	M	14.003	IMPRESSION TYPE	A	2	2	1	1	9
SRC	M	14.004	SOURCE AGENCY / ORI	AN	10	21	1	1	28
TCD	M	14.005	TENPRINT CAPTURE DATE	N	9	9	1	1	16
HLL	M	14.006	HORIZONTAL LINE LENGTH	N	4	5	1	1	12
VLL	M	14.007	VERTICAL LINE LENGTH	N	4	5	1	1	12
SLC	M	14.008	SCALE UNITS	N	2	2	1	1	9
HPS	M	14.009	HORIZONTAL PIXEL SCALE	N	2	5	1	1	12
VPS	M	14.010	VERTICAL PIXEL SCALE	N	2	5	1	1	12
CGA	M	14.011	COMPRESSION ALGORITHM	A	5	7	1	1	14
BPX	M	14.012	BITS PER PIXEL	N	2	3	1	1	10
FGP	M	14.013	FINGER POSITION	N	2	3	1	6	25
RSV		14.014 14.019	RESERVED FOR FUTURE DEFINITION	--	--	--	--	--	--
COM	O	14.020	COMMENT	A	2	1 28	0	1	128
RSV		14.021 14.199	RESERVED FOR FUTURE DEFINITION	--	--	--	--	--	--
UDF	O	14.200 14.998	USER-DEFINED FIELDS	--	--	--	--	--	--
DAT	M	14.999	IMAGE DATA	B	2	--	1	1	--

Key: M = Mandatory; O = Optional; N = Numeric; A = Alphabetic; AN = Alphanumeric; B = Binary

Table 3 – Finger position code

Finger position	Finger code
Unknown	0
Right thumb	1
Right index finger	2
Right middle finger	3
Right ring finger	4
Right little finger	5
Left thumb	6
Left index finger	7
Left middle finger	8
Left ring finger	9
Left little finger	10
Plain right thumb	11
Plain left thumb	12
Plain right four fingers	13
Plain left four fingers	14

Table 4 – Pattern classification

Description	Code
Plain arch	PA
Tented arch	TA
Radial loop	RL
Ulnar loop	UL
Plain whorl	PW
Central pocket loop	CP
Double loop	DL
Accidental whorl	AW
Whorl, type not designated	WN
Right slant loop	RS
Left slant loop	LS
Scar	SR
Amputation	XX
Unknown or unclassifiable	UN

4. SOFTWARE

The source code in this distribution has been developed using the GNU project's `gcc` compiler and `gmake` utility[10]. Manual pages giving a detailed description of the software utilities are included in APPENDIX B. The software has been tested under LINUX [9]and Windows NT using the Cygwin library [11] and associated tools.[2] Most of the software came from existing code that is distributed on our NIST Fingerprint Image Software CDROM [12]. 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, djpegl, dpyan2k, dpyimage, and rdjpgcom** are all copied from the NFIS CDROM. Software tools **an2kcard, an2kimgs, and rec_card** and the library **src/lib/db30** 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. 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
```

where 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
```

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.

---

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

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>

where 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 images in the ANSI/NIST records.

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

an2kcard - Reconstructs the fingerprint card image from the ANSI/NIST record and writes it to a file as raw pixel data.

an2kimgs - Parses the 14 images from the ANSI/NIST record and writes them to stand alone files. The images remain compressed the same as they were in the ANSI/NIST record (WSQ for this database).

rec_card - Reconstructs the fingerprint card image from the 14 images files created by an2kimgs and writes the image to a file as raw pixel data.

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.

djpegl - Decompresses a JPEGL compressed image.

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

dpyan2k - Will parse and display the images in an ANSI/NIST formatted file. Only works in an X windows environment. (Note: No minutiae information is included with data in this database. This command onlyh displays the images from the ANSI/NIST formatted file.)

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

REFERENCES

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

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

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

[4] C. Watson, "*NIST Special Database 10*: Supplemental Fingerprint Card Data (SFCD) for NIST Special Database 9," CD-ROM & documentation, June 1993.

[5] C. Watson, "*NIST Special Database 14*: Mated Fingerprint Card Pairs 2," CD-ROM & documentation, September 1993.

[6] 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.

[7] "Electronic Fingerprint Transmission Specification," CJIS-RS-0010 (V7). Available from Criminal Justice Information Services Division, Federal Bureau of Investigation, 935 Pennsylvania Avenue, NW, Washington D.C. 20535.

[8] "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.

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

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

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

[12] M.D. Garris, C.I. Watson, “NIST Fingerprint Image Software,” CDROM and documentation, 2001.

APPENDIX A. Sample ANSI/NIST Formatted File.

The *"$^{U}_{S}$"* separator shall separate multiple information items within a field or subfield, the *"$^{R}_{S}$"* separator shall separate multiple subfields, and the *"$^{G}_{S}$"* separator shall separate information fields.

Type-1 Transition Record

1.001:223$^{G}_{S}$

1.002:0300$^{G}_{S}$

1.003:1$^{U}_{S}$14$^{R}_{S}$14$^{U}_{S}$00$^{R}_{S}$14$^{U}_{S}$01$^{R}_{S}$14$^{U}_{S}$02$^{R}_{S}$14$^{U}_{S}$03$^{R}_{S}$
14$^{U}_{S}$04$^{R}_{S}$14$^{U}_{S}$05$^{R}_{S}$14$^{U}_{S}$06$^{R}_{S}$14$^{U}_{S}$07$^{R}_{S}$14$^{U}_{S}$
08$^{R}_{S}$14$^{U}_{S}$09$^{R}_{S}$14$^{U}_{S}$10$^{R}_{S}$14$^{U}_{S}$11$^{R}_{S}$14$^{U}_{S}$12$^{R}_{S}$
14$^{U}_{S}$13$^{G}_{S}$

1.004:NISTDATA$^{G}_{S}$

1.005:20010808$^{G}_{S}$

1.006:1$^{G}_{S}$

1.007:DAI00000$^{G}_{S}$

1.008:MDNISTIG$^{G}_{S}$

1.009:a021.raw$^{G}_{S}$

1.011:19.69$^{G}_{S}$

1.012:19.69$^{G}_{S}$

Type-14 Variable Resolution Tenprint Records

1st Type-14 Record (Right Thumb)

14.001:40654$^{G}_{S}$

14.002:00$^{G}_{S}$

14.003:3$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:792$^{G}_{S}$

14.007:756$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:1$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:0$^{U}_{S}$0$^{G}_{S}$

14.202:WN$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_01.wsq$^{G}_{S}$

2nd Type-14 Record (Right Index)

14.001:32606$^{G}_{S}$

14.002:01$^{G}_{S}$

14.003:3$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:796$^{G}_{S}$

14.007:756$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:2$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:792$^{U}_{S}$0$^{G}_{S}$

14.202:RS$^{U}_{S}$TA$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_02.wsq$^{G}_{S}$

3rd Type-14 Record (Right Middle)

14.001:36622$^{G}_{S}$

14.002:02$^{G}_{S}$

14.003:3$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:800$^{G}_{S}$

14.007:756$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:3$^{G}_{S}$

14.200:3972$^{G}_{S}$2492$^{G}_{S}$

14.201:1588$^{G}_{S}$0$^{G}_{S}$

14.202:RS$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_03.wsq$^{G}_{S}$

4th Type-14 Record (Right Ring)

14.001:37715$^{G}_{S}$

14.002:03$^{G}_{S}$

14.003:3$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:800$^{G}_{S}$

14.007:756$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:4$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:2388$^{U}_{S}$0$^{G}_{S}$

14.202:RS$^{U}_{S}$WN$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_04.wsq$^{G}_{S}$

5th Type-14 Record (Right Little)

14.001:42004$^{G}_{S}$

14.002:04$^{G}_{S}$

14.003:3$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:784$^{G}_{S}$

14.007:756$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:5$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:3188$^{U}_{S}$0$^{G}_{S}$

14.202:RS$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_05.wsq$^{G}_{S}$

6th Type-14 Record (Left Thumb)

14.001:34005$^{G}_{S}$

14.002:05$^{G}_{S}$

14.003:3$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:792$^{G}_{S}$

14.007:752$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:6$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:$0^{U}_{S}756^{G}_{S}$

14.202:LS^{G}_{S}

14.203:F_{S}

14.999:a021_06.wsq^{G}_{S}

7th Type-14 Record (Left Index)

14.001:36279^{G}_{S}

14.002:06^{G}_{S}

14.003:3^{G}_{S}

14.004:$MDNISTIG^{G}_{S}$

14.005:20010808^{G}_{S}

14.006:796^{G}_{S}

14.007:752^{G}_{S}

14.008:2^{G}_{S}

14.009:197^{G}_{S}

14.010:197^{G}_{S}

14.011:$WSQ20^{G}_{S}$

14.012:8^{G}_{S}

14.013:7^{G}_{S}

14.200:$3972^{U}_{S}2492^{G}_{S}$

14.201:$792^{U}_{S}756^{G}_{S}$

14.202:$TA^{U}_{S}LS^{G}_{S}$

14.203:F^{G}_{S}

14.999:a021_07.wsq^{G}_{S}

8th Type-14 Record (Left Middle)

14.001:36429^{G}_{S}

14.002:07^{G}_{S}

14.003:3^{G}_{S}

14.004:$MDNISTIG^{G}_{S}$

14.005:20010808^{G}_{S}

14.006:800^{G}_{S}

14.007:752^{G}_{S}

14.008:2^{G}_{S}

14.009:197^{G}_{S}

14.010:197^{G}_{S}

14.011:$WSQ20^{G}_{S}$

14.012:8^{G}_{S}

14.013:8^{G}_{S}

14.200:$3972^{U}_{S}2492^{G}_{S}$

14.201:$1588^{U}_{S}756^{G}_{S}$

14.202:LS^{G}_{S}

14.203:F^{G}_{S}

14.999:a021_08.wsq^{G}_{S}

9th Type-14 Record (Left Ring)

14.001:38668^{G}_{S}

14.002:08^{G}_{S}

14.003:3^{G}_{S}

14.004:$MDNISTIG^{G}_{S}$

14.005:20010808^{G}_{S}

14.006:800^{G}_{S}

14.007:752^{G}_{S}

14.008:2^{G}_{S}

14.009:197^{G}_{S}

14.010:197^{G}_{S}

14.011:$WSQ20^{G}_{S}$

14.012:8^{G}_{S}

14.013:9^{G}_{S}

14.200:$3972^{U}_{S}2492^{G}_{S}$

14.201:$2388^{U}_{S}756^{G}_{S}$

14.202:LS^{G}_{S}

14.203:F^{G}_{S}

14.999:a021_09.wsq^{G}_{S}

10th Type-14 Record (Left Little)

14.001:41225^{G}_{S}

14.002:09^{G}_{S}

14.003:3^{G}_{S}

14.004:$MDNISTIG^{G}_{S}$

14.005:20010808^{G}_{S}

14.006:784^{G}_{S}

14.007:752^{G}_{S}

14.008:2^{G}_{S}

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:10$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:3188$^{U}_{S}$756$^{G}_{S}$

14.202:LS$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_10.wsq$^{G}_{S}$

11th Type-14 Record (Left Four Fingers Plain Impression)

14.001:106697$^{G}_{S}$

14.002:10$^{G}_{S}$

14.003:2$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:1588$^{G}_{S}$

14.007:984$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:14$^{G}_{S}$

14.200:3972$^{G}_{S}$2492$^{G}_{S}$

14.201:0$^{G}_{S}$1508$^{G}_{S}$

14.202:LS$^{R}_{S}$LS$^{R}_{S}$LS$^{R}_{S}$TA$^{U}_{S}$LS$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_14.wsq$^{G}_{S}$

12th Type-14 Record (Left Thumb Plain Impression)

14.001:29523$^{G}_{S}$

14.002:11$^{G}_{S}$

14.003:2$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:400$^{G}_{S}$

14.007:984$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:12$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:1588$^{U}_{S}$1508$^{G}_{S}$

14.202:LS$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_12.wsq$^{G}_{S}$

13th Type-14 Record (Right Thumb Plain Impression)

14.001:24483$^{G}_{S}$

14.002:12$^{G}_{S}$

14.003:2$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:400$^{G}_{S}$

14.007:984$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:11$^{G}_{S}$

14.200:3972$^{U}_{S}$2492$^{G}_{S}$

14.201:1988$^{U}_{S}$1508$^{G}_{S}$

14.202:WN$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_11.wsq$^{G}_{S}$

14th Type-14 Record (Right Four Fingers Plain Impression)

14.001:113204$^{G}_{S}$

14.002:13$^{G}_{S}$

14.003:2$^{G}_{S}$

14.004:MDNISTIG$^{G}_{S}$

14.005:20010808$^{G}_{S}$

14.006:1584$^{G}_{S}$

14.007:984$^{G}_{S}$

14.008:2$^{G}_{S}$

14.009:197$^{G}_{S}$

14.010:197$^{G}_{S}$

14.011:WSQ20$^{G}_{S}$

14.012:8$^{G}_{S}$

14.013:13$^{G}_{S}$

14.200:3972$^{G}_{S}$2492$^{G}_{S}$

14.201:2388$^{G}_{S}$1508$^{G}_{S}$

14.202:RS$^{U}_{S}$TA$^{R}_{S}$RS$^{R}_{S}$RS$^{U}_{S}$WN$^{R}_{S}$RS$^{G}_{S}$

14.203:F$^{G}_{S}$

14.999:a021_13.wsq$^{G}_{S}$

APPENDIX B. Software Manual Pages.

NAME

an2kcard − Takes an ANSI/NIST 2000 file from NIST Special Databases 29 and 30 and combines the 14 fingerprint images to reconstruct the original fingerprint card image.

SYNOPSIS

an2kcard <*ansi_nist file*>

DESCRIPTION

An2kcard Reconstructs the fingerprint card image for a standard compliant ANSI/NIST-ITL 1-2000 file from NIST Special Databases 29 and 30 and writes its contents to a new file. The output file is just the raw pixel values for the fingerprint card image. The size of the fingerprint card image is printed to the screen so the user can use **cjpegb** and convert the image into a viewable format.

OPTIONS

<*ansi_nist file*>

the input ANSI/NIST file

% an2kcard a001.an2

SEE ALSO

an2kimgs(1), **rec_card**(1)

AUTHOR

NIST/ITL/DIV894/Image Group

NAME

an2kimgs − Takes an ANSI/NIST 2000 file from NIST Special Database 29 and 30 and parses the 14 fingerprint images from the file and writes them out as individual files.

SYNOPSIS

an2kimgs <*ansi_nist file*>

DESCRIPTION

An2kimgs Parses the fingerprint images from a standard compliant ANSI/NIST-ITL 1-2000 file and writes them into individual stand alone files. The files remain in the same compressed format as they were in the ANSI/NIST file. The output file names will be the prefix of the ANSI/NIST file with an "_" followed by the finger number (0-14) and an extension based on the compression type "jpl" for JPEGL and "wsq" for WSQ. For example a001.an2 might produce files a001_01.wsq thru a001_14.wsq. The decompression utilities **dwsq** and **djpegl** can be used to decompress the files if desired. The utilities **rdwsqcom** and **rdjpgcom** will print out the comment fields of the compressed files so fingerprint reference information can be viewed.

OPTIONS

<*ansi_nist file*>
the input ANSI/NIST file

% an2kimgs a001.an2

SEE ALSO

an2kcard(1), **rec_card**(1)

AUTHOR

NIST/ITL/DIV894/Image Group

NAME

rec_card − Takes the set of 14 segemented stand alone fingerprint images (with the correct reference information in the comment field) and combines them to reconstruct the original fingerprint card image.

SYNOPSIS

rec_card <*ofile*> <*14 card image files*>

DESCRIPTION

Rec_card Reconstructs the fingerprint card image from the set of 14 segmented fingerprint images (with correct reference information in the comment field) and writes its contents to a new file. The output file is just the raw pixel values for the fingerprint card image. The size of the fingerprint card image is printed to the screen so the user can use **cjpegb** and convert the image into a viewable format.

OPTIONS

<*ofile*> the output raw fingerprint image file

<*14 card image files*>
the 14 segemented fingerprint image files

% rec_card a001.raw a001_01.jpl ... a001_14.jpl

SEE ALSO

an2kcard(1), **an2kimgs**(1)

AUTHOR

NIST/ITL/DIV894/Image Group

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

djpegl − decompresses a Lossless JPEG (JPEGL) grayscale or color image.

SYNOPSIS

djpegl <*outext*> <*image file*>
[**-raw_out** [**-nonintrlv**]]

DESCRIPTION

Djpegl takes as input a file containing a Lossless JPEG (JPEGL) compressed grayscale or color image. Once read into memory, the compressed pixmap is decoded and reconstructed to its original condition prior to compression.

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, **djpegl** takes the input filename and replaces its extension with the one specified here.

<*image file*>
the input JPEGL file to be decompressed.

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

-nonintrlv
specifies that the color components in the reconstructed image should be organized into separate component planes. The **-raw_out** flag must be used with this option, because the IHead format only supports interleaved color pixels. (See INTERLEAVE OPTIONS below.)

INTERLEAVE OPTIONS

For example, given an RGB image, its color components 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, **djpegl** uses interleaved color component pixels in the reconstructed output image. Note that all color IHead images must be interleaved.

NISTCOM OUTPUT

Upon successful completion, **djpegl**, 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 JPEGL-encoded input file. This comment block can be used to hold user-supplied attributes. The JPEGL encoder, **cjpegl**, by convention inserts one of these comment blocks in each compressed output file it creates. (The utility **rdjpgcom** can be used to scan a JPEG 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 JPEGL 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 **djpegl**:

```
NIST_COM 9
PIX_WIDTH 768
PIX_HEIGHT 1024
PIX_DEPTH 24
PPI -1
LOSSY 0
NUM_COMPONENTS 3
HV_FACTORS 1,1:1,1:1,1
INTERLEAVE 1
```

EXAMPLES

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

% djpegl raw finger.jpl -r
decompresses a JPEGL-encoded grayscale fingerprint image and stores the reconstructed image to a raw pixmap file. Note the NISTCOM file, **finger.ncm**, is also created.

% djpegl raw face.jpl -r
decompresses a JPEGL-encoded RGB face image and stores the reconstructed image to a raw pixmap file. Note the NISTCOM file, **face.ncm**, is also created.

SEE ALSO

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

AUTHOR

NIST/ITL/DIV894/Image Group

NAME

dpyan2k − display image and minutiae contents of an ANSI/NIST file.

SYNOPSIS

dpyan2k [*options*] <*ANSI_NIST ...*>

-a *n*
-v
-x
-b *n*
-i
-n
-p *n*
-W *n*
-H *n*
-X *n*
-Y *n*
-T *title*
-d *display*

DESCRIPTION

Dpyan2k displays in a sequence of X11 windows all the image records and overlays any corresponding minutiae from Type-9 records contained in an ANSI/NIST-ITL 1-2000 file.

If multiple input files are specified, *dpyan2k* reads each ANSI/NIST file into memory and displays its contents, one file at a time. Multiple image records within an ANSI/NIST file are displayed simultaneously by forking background window processes, one for each image record.

If an image is too large to be displayed on the screen, the upper left hand 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 remove a displayed image window by striking any key within that window. Once all windows associated with a particular ANSI/NIST file have been removed, the utility proceeds to display the contents of the next ANSI/NIST file listed on the command line.

OPTIONS

-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].

-i directs the utility to use the FBI/IAFIS fields 13-23 in a Type-9 record when overlaying minutiae on an image.

-n directs the utility to use the NIST fields 5-12 in a Type-9 record when overlaying minutiae on an image. This is the default setting.

-p sets the pixel width of overlayed minutia points [3].

-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].

-T *title* sets all image window names to *title*.

-d *display*
connects to an alternate X11 display.

<ANSI_NIST ...>
one or more ANSI/NIST files with images and possbily minutiae to be displayed.

EXAMPLES

From *test/an2k/execs/dpyan2k/dpyan2k.src*:

% dpyan2k ../../data/nist.an2
displays image records and overlays minutia using NIST Type-9 fields.

% dpyan2k -i ../../data/iafis.an2
displays image records and overlays minutia using FBI/IAFIS Type-9 fields.

SEE ALSO

an2ktool(1C), **dpyimage**(1D), **mindtct**(1B)

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

rdjpgcom − display text comments from a JPEG file

SYNOPSIS

rdjpgcom [**−verbose**] [*filename*]

DESCRIPTION

rdjpgcom reads the named JPEG/JFIF file, or the standard input if no file is named, and prints any text comments found in the file on the standard output.

The JPEG standard allows "comment" (COM) blocks to occur within a JPEG file. Although the standard doesn't actually define what COM blocks are for, they are widely used to hold user-supplied text strings. This lets you add annotations, titles, index terms, etc to your JPEG files, and later retrieve them as text. COM blocks do not interfere with the image stored in the JPEG file. The maximum size of a COM block is 64K, but you can have as many of them as you like in one JPEG file.

OPTIONS

−verbose

Causes **rdjpgcom** to also display the JPEG image dimensions.

Switch names may be abbreviated, and are not case sensitive.

HINTS

rdjpgcom does not depend on the IJG JPEG library. Its source code is intended as an illustration of the minimum amount of code required to parse a JPEG file header correctly.

In **−verbose** mode, **rdjpgcom** will also attempt to print the contents of any "APP12" markers as text. Some digital cameras produce APP12 markers containing useful textual information. If you like, you can modify the source code to print other APPn marker types as well.

SEE ALSO

cjpeg(1), **djpeg**(1), **jpegtran**(1), **wrjpgcom**(1)

AUTHOR

Independent JPEG Group