Nist Special Database 9

Mated Fingerprint Card Pairs Users' Guide

C. I. Watson

National Institute of Standards and Technology

Advanced Systems Division

Image Recognition Group

February 16, 1993

Users' Guide

Contents

1.0 INTRODUCTION

This report describes the NIST fingerprint database, *NIST Special Database 9*, which will consist of multiple volumes. Currently, five volumes are scheduled to be released. Volumes will be released periodically after the initial release of volume one. The database is being distributed for use in the development and testing of automated fingerprint classification systems on a common set of images. Each volume has 3 disks containing 8-bit gray scale fingerprint images.Each CD- ROM contains 1800 (900 pairs/90 card pairs) fingerprints stored in NIST's IHead raster data format and compressed using a modified JPEG lossless [1] compression algorithm. The prints are 832 (w) X 768 (h) pixels (see Appendix A) and each CD-ROM requires approximately 630 to 660 megabytes of storage when the prints are compressed and 1.2 gigabytes of storage when uncompressed (1.85:1 average compression ratio).

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 C for class distribution statistics on the first five volumes). However, because the work is stacked by sex and unit (A unit is a major grouping of the Henry Classifications [2].) at MIC-TEC verify desk, a true card level horizontal slice can not be guaranteed.

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 the NIST IHead **id** field of each file, allowing for comparison with hypothesized classes.

After five volumes of the data is collected, a supplemental dataset may be built to fill in the less frequent and transitional NCIC fingerprint classes at the finger level. This supplemental data will be drawn directly from the technical master file and consist of single fingerprint images of interest. These images will not be matched pairs. The composition of the supplemental data will be based on a statistical analysis of the main body of the data.

Back to Contents

2.0 MODIFIED JPEG LOSSLESS COMPRESSION

The compression used was developed from techniques outlined in the WG 10 "JPEG" (draft) standard [1] for 8-bit gray scale images with modifications to the compressed data format. The NIST IHead format already contained most of the information needed in the decompression algorithm, so the JPEG compressed data format was modified to contain only the information needed when reconstructing the Huffman code tables and identifying the type of predictor used in the coding process. Codes used to compress and decompress the images are still developed per the draft standard, but only applied to 8-bit gray scale images.

The standard uses a differential coding scheme and allows for seven possible ways of predicting a pixel value. Tests showed that predictor number 4 provided the best compression on the fingerprint images; therefore, this predictor was used to compress all of the images.

3.0 DATABASE REFLECTANCE CALIBRATION

The reflectance values for the fingerprint database, *NIST Special Database* 9, were calibrated using a reflection step table [3]. A plot of the reflectance values obtained using this step table is shown in Appendix B. Also shown on the plot and below is an equation used to predict the reflectance of a given datapoint. The plot in Appendix B shows that this predicted reflectance closely follows the actual reflectance obtained using the reflection step table.

$$\text{predicted \% reflectance} = -5.1 + (.36 * \text{grayscale pixel value})$$

Back to Contents

4.0 FINGERPRINT FILE FORMAT [4][5]

Image file formats and effective data compression and decompression are critical to the usefulness of image archives. Each fingerprint was digitized in 8-bit gray scale form at 19.6850 pixels/mm (500 pixels/inch), 2-dimensionally compressed using a modified JPEG lossless algorithm, and temporarily archived onto computer magnetic mass storage. Once all prints were digitized, the images were mastered and replicated onto ISO-9660 formatted CD-ROM discs for permanent archiving and distribution.

After digitization, certain attributes of an image are required to correctly interpret the 1-dimensional pixel data as a 2-dimensional image. Examples of such attributes are the pixel width and pixel height of the image. These attributes can be stored in a machine readable header prefixed to the raster bit stream. A program which manipulates the raster data of an image is able to first read the header and determine the proper interpretation of the data which follow it.

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 adequately represent both binary and gray level images, to represent images captured from different scanners and cameras, and to satisfy the image requirements of diversified applications including, but not limited to, image archival/retrieval, character recognition, and fingerprint classification. Figure 1 illustrates the IHead format.

Header Length

ASCII Format Image Header

8-bit Gray Scale Raster Stream

11010100110100111110010010110...

- Representing the digital scan across the page left to right, top to bottom.
- 8 bits to a pixel
- 256 levels of gray
- 1 Pixel is packed into a single byte of memory.

Figure 1: 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 computrs, 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 1.

```
/*************************************************************************
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 sigbit;                      /* 0->sigbit first | 1->sigbit last */
    char byte_order;                  /* 0->highlow | 1->lowhigh*/
    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 2: The IHead 'C' programming language structure definition.

Back to Contents

The IHead structure definition written in the 'C' programming language is listed in Figure 2. Figure 3 lists the header values from an IHead file corresponding to the structure members listed in Figure 2. This header information belongs to the database file **f0000001.pct** (see Figure A.1 in Appendix A). Referencing the structure members listed in Figure 2, the first attribute field of 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, with any references to another classification (see Figure 8 and Section 5.4 for an example of class referencing and Figure 7 for the class codes). This convention enables an image recognition system's hypothesized classification to be automatically scored against the actual classification.

IMAGE FILE HEADER

Identity	: f0000001.pct m i 20
Header Size	: 288 (bytes)
Date Created	: Mon Dec 718:15:24 1992
Width	: 832 (pixels)
Height	: 768 (pixels)
Bits per Pixel	: 768 (pixels)
Resolution	: 500 (Ppi)
Compression	: 6 (code)
Compress Length	: 307195 (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.t1116010.01 4096x1536
X Origin	: 0 (pixels)
Y Origin	: 0 (pixels)

Figure 3: The IHead values for the fingerprint data file **f0000001.pct**.

The attribute field, **created**, 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, 19.6850 pixels/mm (500 pixels/inch). The next two fields deal with compression.

The IHead format, images may be compressed with vinually 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 which signifies which compression technique, if any, has been applied to the raster image data which follows the header. If the compression code is zero, then the image data is 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. For example, the images in this database have a compression code of 6 signifying that modified JPEG lossless compression has been applied to the image data prior to file creation. In order to load the compressed image data into main memory, the value in **complen** is used to determine the size of the compressed block of image data.

Back to Contents

Once the compressed image data has been loaded into memory, JPEG lossless decompression can be used to produce an image which has the pixel dimensions consistent with those stored in its header. Using JPEG lossless compression and this compression scheme on the images in this database, an average compression ratio of 1.8 to 1 was achieved.

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 in which bits of significance are stored within each unit; most significant bit first or least significant bit first. 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 effectively dealt with.

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 3 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, ..., tape*n*), 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 5.2). 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 origin, upper left hand comer 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.

Back to Contents

5.0 DATABASE CONTENT AND ORGANIZATION

Each CD-ROM in *NIST Special Database* 9 contains 1800 8-bit gray scale fingerprint images stored in the **data** directory (see Figure 4). The images, which use approximately 640 megabytes of storage compressed and 1.2 gigabytes of storage uncompressed, are distributed on an ISO-9660 formatted CD-ROM and compressed using a modified IPEG loss less compression algorithm. Included with the fingerprint data are software and documentation.

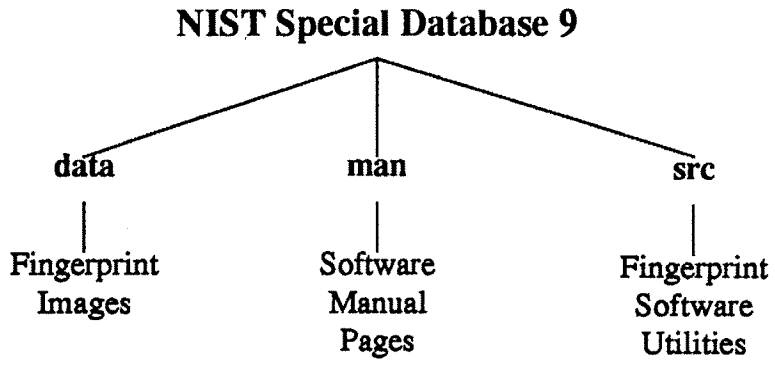

Figure 4: Top level directory tree for each CD-ROM in *NIST Special Database* 9.

5.1 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** pages for the source code stored in the **man** directories. The **data** directory contains the fingerprint images stored in 2 levels of subdirectories for easier access and clarity (see Figure 5). The first level consists of nine subdirectories, **mfcp_0 - mfcp_8** (mated fingerprint card pairs), which divide the mated card pairs into groups of ten. The next level has a subdirectory for each of ten 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 "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 ".pct" (picture) 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- 00001800). The finger number is given by the last of the seven digits, with zero representing digit ten on the fingerprint card (see Figure 6 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 in volume 1 of the database, *NIST Special Database 9*, through the last CD-ROM in the last volume of the database.

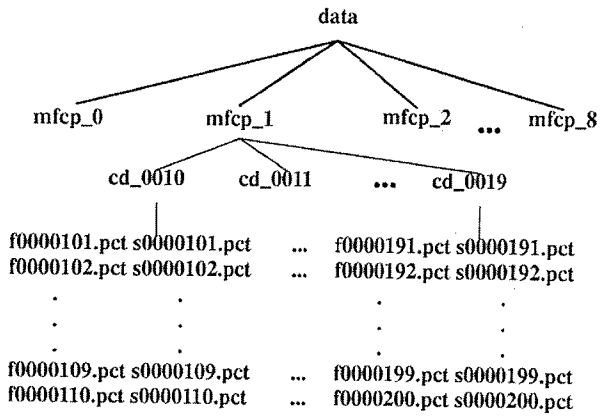

Figure 5: Arrangement of fingerprint files on each CD-ROM in NIST Special Database 9. (Note: This example is from CD-ROM 1 of 3 in Volume 1 of NIST Special Database 9.)

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

Figure 6: Layout of fingerprint card numbers.

Back to Contents

5.2 Automated Name and Technical Search Cards

The first character in the second field of the FBI filename in the **parent** field (see Section 4.0) 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 anyone 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 Sections (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 evel of confidence, and that the classifications are the same, within the error permitted by ATS.

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 search/file pairs, the **n** cards are not guaranteed to ident in ATS, although they are in fact identical.

5.3 NCIC Classifications

The classes stored in the NIST Ihead **id** field are the NCIC classes [2], including any references (see Figure 8), that were assigned by the FBI (IDAS). A listing of the possible class codes is given below. 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 7: Classification codes for *NIST Special Database 9.*

Back to Contents

5.4 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 9* contains prints with references (see Appendix C for referencing statistics). The **id** fields of these prints contain the primary fingerprint class followed by any references. An example IHead header is shown in Figure 8 (**f0000042.pct**) and the corresponding print is shown in Figure A.2. 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.

IMAGE FILE HEADER

Identify	: f0000042.pct 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	: 6 (code)
Compress Length	: 386882 (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 8: The IHead values for the fingerprint data file **f0000042.pct**.

Back to Contents

5.5 Segmenting the Fingerprint Images

The individual fingerprint images were obtained by scanning all ten prints on a card into one large image (4096 × 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

× 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.

5.6 Inked and Live Scan Printed

The fingerprints were scanned from two types of prints. The first type were fingerprint patterns created by rolling the individuals ink covered finger on the fingerprint card (denoted by an **i** in the **id** field). The second type were patterns taken with a live scanning device and then printed onto a fingerprint card (denoted by an **l** in the **id** field). This is important in that the quality of the image is by the resolution of the printer used to print the live scanned image onto the fingerprint card.

6.0 SOFTWARE FOR ACCESSING DATABASE

Included with the fingerprint images are documentation and software written in the 'C' programming language. The software was developed on a SUN 4/470 sparc station and has only been tested on that platform. Four programs are included in the **src** directory: **dumpihdr, ihdr2sun, sunalign, and dcpljpg**. These routines are provided as an example to software developers of how IHead images can be manipulated and used. Descriptions of these programs and their subroutines are given below as well as in the included man pages located in the **man** directory. Copies of the manual pages are also included in Appendix D.

6.1 Compilation

CD-ROM, used for *NIST Special Database* 9, is a read only storage medium. The files in the **src** directory must be copied to a read-writable partition prior to compiling. After copying these files, executable binaries can be produced by invoking the UNIX utility **make** to execute the included makefile. An example of this command follows.

make -f **makefile.mak**

6.2 Dumpihdr <Ihead file>

Dumpihdr is a program which reads an image's IHead data from the given file and formats the header data into a report which is printed to standard output. The report shown in Figure 3 was generated using this utility. The main routine for **dumpihdr** is found in the file **dumpihdr.c** and calls the external function **readihdr()**.

Readihdr() is a function responsible for loading an image's IHead data from a file into main memory. This routine allocates, reads, and returns the header information from an open image file in an initialized Iliead structure. This function is found in the file **ihead.c**. The IHead structure definition is listed in Figure 2 and is found in the file **ihead.h**.

Back to Contents

6.3 Ihdr2sun <Ihead file>

Ihdr2sun converts an image from NIST IHead format to Sun rasterfile format. **Ihdr2sun** loads an IHead formatted image from a file into main memory and writes the raster data to a new file appending the data to a Sun rasterfile header. The main routine for this program is found in the file **ihdr2sun.c** and calls the external function **ReadIheadRaster()** which is found in the file **rasterio.c**.

ReadIheadRaster() is the procedure responsible for loading an IHead image from a file into main memory. This routine reads the image's header data returning an initialized IHead structure by calling **readihdr()**. In addition, the image's raster data is returned to the caller uncompressed. The images in this database have been 2-dimensionally compressed using a modified JPEG loss less compression algorithm, therefore **ReadIheadRaster()** invokes the external procedure **jpglldcp()** which is responsible for decompressing the raster data. Upon completion, **ReadIheadRaster()** returns an initialized IHead structure, the uncompressed raster data, the image's width and height in pixels, and pixel depth.

Jpglldcp() accepts image raster data compressed using the modified JPEG lossless compression algorithm and returns the uncompressed image raster data. **Jpglldcp()** was developed using techniques described in the WG10 "JPEG" (draft) standard [1] and adapted for use with this database. Source code for the algorithm is found in **jpglldcp.c**.

6.4 Dcplljpg <lossless JPEG compressed file>

Dcplljpg is a program which decompresses a fingerprint image file (approximately 10 seconds per image, for images from this database, on a scientific workstation) that was compressed using the modified JPEG compression routine. The routine accepts a compressed image in NIST IHead format and writes the uncompressed image to the same filename using the NIST IHead format. The main routine is found in **dcplljpg.c** and calls the external functions **ReadIheadRaster()** (see section 6.3 for ReadIheadRaster description) and **writeihdrfile()**.

Writeihdrfile() is a routine that writes an IHead image into a file. This routine opens the passed filename and writes the given IHead structure and corresponding data to the file. **Writeihdrfile()** is found in the src file **rasterio.c**.

Back to Contents

References

[1] WG10 "JPEG", committee draft ISO/IEC CD 10198-1, "Digital Compression and Coding of Continuous-Tone Still Images," March 3, 1991.

[2] *The Science of Fingerprints.* U.S. Department of Justice, Washington, D.C., 1984.

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

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

[5] C.I. Watson and C.L. Wilson, "NIST Special Database 4, Fingerprint Database," National Institute of Standards and Technology, March 15, 1992.

Appendices

Back to Contents | **Database Description** | **SRD Data Home**

[Online Databases] [New and Updated Databases]
[Database Price List] [JPCRD] [CODATA] [FAQ] [Comments] [NIST] [Data]

Create Date: 2/03
Last Update: Monday, 05-May-03 09:35:54