ISO/IEC JTC 1/SC 29/WG 1 N 2415
Date: October 18, 2002

ISO/IEC JTC 1/SC 29/WG 1
(ITU-T SG8)

Coding of Still Pictures

JBIG
Joint Bi-level Image Experts Group

JPEG
Joint Photographic Experts Group

TITLE: JasPer Software Reference Manual (Version 1.600.0)

SOURCE: Michael D. Adams
Assistant Professor
Dept. of Electrical and Computer Engineering
University of Victoria
Victoria, BC, Canada
E-mail: mdadams@ieee.org

PROJECT: JPEG 2000

STATUS:

REQUESTED ACTION:

DISTRIBUTION: Public

Contact:
ISO/IEC JTC1/SC29/WG1 Convener—Dr. Daniel Lee
Yahoo!, 3420 Central Expressway, Santa Clara, California 95051, USA
Tel: +1 408 992 7051, Fax: +1 253 830 0372, E-mail: dlee@yahoo-inc.com

Contents

1 Introduction

JasPer is a collection of software (i.e., a library and application programs) for the coding and manipulation of images. Using the JasPer software, one can easily encode and decode image data in a number of popular formats. For example, the JasPer software can handle image data in the following formats (amongst others): JPEG-2000 JP2, JPEG-2000 code stream, JPEG, Portable Bitmap, Windows BMP, and Sun Rasterfile.

The JasPer software is written in the C programming language [1, 2]. This language was chosen mainly due to the availability of C development environments for most of today's computing platforms. The JasPer software consists of about 30k lines of code in total. This code is spread across a library and several application programs.

2 License

The JasPer software may only be used under the terms of the license below.

D. User agrees that it shall not commence any action against Image Power, Inc., the University of British Columbia, Michael David Adams, or any other contributors (collectively "Licensors") for infringement of any intellectual property rights ("IPR") held by the User in respect of any technology that User owns or has a right to license or sublicense and which is an element required in order to claim compliance with ISO/IEC 15444-1 (i.e., JPEG-2000 Part 1). "IPR" means all intellectual property rights worldwide arising under statutory or common law, and whether or not perfected, including, without limitation, all (i) patents and patent applications owned or licensable by User; (ii) rights associated with works of authorship including copyrights, copyright applications, copyright registrations, mask work rights, mask work applications, mask work registrations; (iii) rights relating to the protection of trade secrets and confidential information; (iv) any right analogous to those set forth in subsections (i), (ii), or (iii) and any other proprietary rights relating to intangible property (other than trademark, trade dress, or service mark rights); and (v) divisions, continuations, renewals, reissues and extensions of the foregoing (as and to the extent applicable) now existing, hereafter filed, issued or acquired.

E. If User commences an infringement action against any Licensor(s) then such Licensor(s) shall have the right to terminate User’s license and all sublicenses that have been granted hereunder by User to other parties.

F. This software is for use only in hardware or software products that are compliant with ISO/IEC 15444-1 (i.e., JPEG-2000 Part 1). No license or right to this Software is granted for products that do not comply with ISO/IEC 15444-1. The JPEG-2000 Part 1 standard can be purchased from the ISO.

THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF THE JASPER SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER. THE JASPER SOFTWARE IS PROVIDED BY THE LICENSORS AND CONTRIBUTORS UNDER THIS LICENSE ON AN ‘‘AS-IS’’ BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE JASPER SOFTWARE IS FREE OF DEFECTS, IS MERCHANTABLE, IS FIT FOR A PARTICULAR PURPOSE OR IS NON-INFRINGING. THOSE INTENDING TO USE THE JASPER SOFTWARE OR MODIFICATIONS THEREOF FOR USE IN HARDWARE OR SOFTWARE PRODUCTS ARE ADVISED THAT THEIR USE MAY INFRINGE EXISTING PATENTS, COPYRIGHTS, TRADEMARKS, OR OTHER INTELLECTUAL PROPERTY RIGHTS. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE JASPER SOFTWARE IS WITH THE USER. SHOULD ANY PART OF THE JASPER SOFTWARE PROVE DEFECTIVE IN ANY RESPECT, THE USER (AND NOT THE INITIAL DEVELOPERS, THE UNIVERSITY OF BRITISH COLUMBIA, IMAGE POWER, INC., MICHAEL DAVID ADAMS, OR ANY OTHER CONTRIBUTOR) SHALL ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL THE INITIAL DEVELOPER, THE UNIVERSITY OF BRITISH COLUMBIA, IMAGE POWER, INC.,

```
MICHAEL DAVID ADAMS, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF THE
JASPER SOFTWARE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO
THE USER OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR
CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION,
DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR
MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF
SUCH PARTY HAD BEEN INFORMED, OR OUGHT TO HAVE KNOWN, OF THE POSSIBILITY
OF SUCH DAMAGES.  THE JASPER SOFTWARE AND UNDERLYING TECHNOLOGY ARE NOT
FAULT-TOLERANT AND ARE NOT DESIGNED, MANUFACTURED OR INTENDED FOR USE OR
RESALE AS ON-LINE CONTROL EQUIPMENT IN HAZARDOUS ENVIRONMENTS REQUIRING
FAIL-SAFE PERFORMANCE, SUCH AS IN THE OPERATION OF NUCLEAR FACILITIES,
AIRCRAFT NAVIGATION OR COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL, DIRECT
LIFE SUPPORT MACHINES, OR WEAPONS SYSTEMS, IN WHICH THE FAILURE OF THE
JASPER SOFTWARE OR UNDERLYING TECHNOLOGY OR PRODUCT COULD LEAD DIRECTLY
TO DEATH, PERSONAL INJURY, OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE
("HIGH RISK ACTIVITIES").  LICENSOR SPECIFICALLY DISCLAIMS ANY EXPRESS
OR IMPLIED WARRANTY OF FITNESS FOR HIGH RISK ACTIVITIES.  USER WILL NOT
KNOWINGLY USE, DISTRIBUTE OR RESELL THE JASPER SOFTWARE OR UNDERLYING
TECHNOLOGY OR PRODUCTS FOR HIGH RISK ACTIVITIES AND WILL ENSURE THAT ITS
CUSTOMERS AND END-USERS OF ITS PRODUCTS ARE PROVIDED WITH A COPY OF THE
NOTICE SPECIFIED IN THIS SECTION.

__END_OF_JASPER_LICENSE__
```

3 Version Identification

As the JasPer software continues to evolve over time, it is important to be able to identify particular releases of the software. Every release of the JasPer software is named by a version identifier. A version identifier is comprised of three integers separated by dots. In order, the three integers correspond to the major, minor, and micro version numbers for the software. For example, the version identifier "1.500.0" corresponds to a major version of 1, a minor version of 500, and a micro version of 0. In instances where the micro version is zero, the version identifier may be truncated after the minor version number. For example, the version identifier "1.500" is completely valid and simply an abbreviation for "1.500.0".

Given two different releases of the JasPer software, one can easily determine which one is more recent by comparing the version identifiers, as follows: 1) if the major version numbers differ, the release with the higher major version number is newer; 2) if the major version numbers are equal and the minor version numbers differ, the release with the higher minor version number is newer; or 3) if the major version numbers are equal and the minor version numbers are equal, the release with the higher micro version is newer.

4 Obtaining the Software

The latest version of the JasPer software can be downloaded from the following locations:

- JasPer Project Home Page (i.e., http://www.ece.uvic.ca/~mdadams/jasper)

- JPEG Web Site Software Page (i.e., http://www.jpeg.org/software)

5 Extracting the Software

The JasPer software is distributed in the form of a Zip file. Therefore, in order to extract the contents of this file, a program capable of handling Zip archives is required. Such software is readily available for many different computing platforms, and can be obtained from:

- Info-Zip Web Site (i.e., http://www.info-zip.org). Unzip software for many different computing platforms (e.g., UNIX, DOS, MacOS, etc.).
- WinZip Web Site (i.e., http://www.winzip.com). Zip/Unzip software for Microsoft Windows.

6 Building the Software

Obviously, before the software can be built, the contents of the archive file containing the JasPer distribution must be extracted.

The JasPer code should compile and run on any platform with a C language implementation conforming to ISO/IEC 9899:1999 [2] (i.e., the ISO C language standard) and supporting a subset of ISO/IEC 9945-1 [3] (i.e., the POSIX C API). Only limited POSIX support is required (i.e., the open, close, read, write, and lseek functions must be supported).

Portability was a major consideration during the design of the JasPer software. For this reason, the software makes minimal assumptions about the runtime environment. The code uses very little floating-point arithmetic, most of which can be attributed to floating-point conversions in printf's. This minimal use of floating-point arithmetic should make the code much easier to port to platforms lacking hardware support for floating-point arithmetic.

The specific procedure required to build the JasPer software depends on the type of system involved. Only two different build methods are supported. The first method is based on the well known make utility and should work on most UNIX (or UNIX-like) systems. The second method is specifically tailored to the needs of Microsoft Visual C under Microsoft Windows. In passing, we note that by using free software such as Cygwin (http://www.cygwin.com), one can also create a UNIX-like environment under Microsoft Windows in which to build JasPer using the first method.

If you are unfortunate enough to have a compiler that is not compliant with ISO/IEC 9899:1999, you may need to make some changes to the code. Unfortunately, even some of the most popular C language implementations do not strictly comply with the standard. One such example is Microsoft Visual C 6.0. Due to the popularity of Visual C, however, several workarounds have been added to the JasPer code to ensure that it will compile successfully with Visual C.

The current version of the JasPer software is known to compile in the following environments:

- Red Hat Linux 7.3, GNU C 2.96, GNU Make 3.79.1
- SunOS/SPARC 5.7, GNU C 2.95, SunOS make
- Windows 2000 Professional, Microsoft Visual C 6.0

Of course, the software should compile successfully in many other environments as well. For example, past versions of JasPer have been reported to build successfully in the following environments:

- Apple Macintosh PPC G3, Rhapsody 5.6, GNU C 2.7.2.1
- Power Macintosh, GNU/Linux 2.2.15pre9, GNU C 2.95.2
- Sun SPARC, Solaris 2.7, GNU C 2.95.2
- Compaq/DEC Alpha, OSF/1 4.0F, GNU C 2.95.2
- IBM PowerPC, AIX 4.2, GNU C 2.95.2
- HP 9000/712, HP-UX 10.20, GNU C 2.95.2
- SGI Origin 200, IRIX 6.5, GNU C 2.95.2
- OpenVMS Alpha 7.2-1 with Compaq C compiler V6.2-008 (after creating custom makefiles)

6.1 Build Process for UNIX (or UNIX-Like) Systems

The JasPer software is intended to be built using the standard UNIX make utility (in conjunction with a configure script).

If you need a C compiler that is reasonably compliant with the ISO/IEC 9899:1999 standard, you can obtain GNU C from the GNU Project web site (i.e., `http://www.gnu.org`). If you need an implementation of the make utility, you can also obtain GNU Make from the the GNU Project web site. All GNU software is free software.

In what follows, `$TOPDIR` denotes the top level directory of the JasPer software distribution (i.e., the directory containing the file named `configure`). To build the software, the following steps are required (in order):

1. *Set the current working directory to the top level directory of the JasPer software distribution.*

   To set the current working directory as required, type:

   ```
   cd $TOPDIR
   ```

   (where `$TOPDIR` is defined as described earlier in this section).

2. *Perform the initial configuration of the software.*

   This is accomplished by running the configure script. This process allows important information about the system configuration to be determined. The configure script also generates makefiles from makefile templates. In theory, it should not be necessary to manually edit any of the makefile templates (i.e., the Makefile.in files) used by the configure program. To invoke the configure program, type:

   ```
   ./configure
   ```

The configure script accepts a number of options. These options can be listed by invoking the configure command with the `--help` option. Unless you know what you are doing (or have problems with the default build settings), it is *strongly* recommended that you not override the default settings for configure.

In some cases, it may be necessary to explicitly disable the use of the IJG JPEG library (i.e., libjpeg). This is accomplished by supplying the `--disable-libjpeg` option to configure. For example, such action may be required if the version of the JPEG library installed on your system is not compatible with the version of JasPer being built. Also, when building under the Cygwin environment, it may be neccessary to explicitly disable the use of the JPEG library.

In some situations, it may be necessary to explicitly disable the use of the OpenGL libraries. This is accomplished by supplying the `--disable-opengl` option to configure.

3. *Compile and link the software.*

   This is accomplished via the make command. To run the make program, type:

   ```
   make
   ```

4. *Install the software.*

   This step may require special (e.g., superuser/administrator) privileges depending on the target directory for installation. (The default installation directories are normally under `/usr/local`.) To install the executables, libraries, include files, and other auxiliary data, type:

   ```
   make install
   ```

Presuming that the build was successful, the executables for the JasPer application programs can be found in the directory `$TOPDIR/src/appl`.

6.2 Build Process for Microsoft Visual C Studio under Microsoft Windows

With Microsoft Visual C, the entire build process is driven from workspace and project files. For the sake of convenience, all of the workspace and project files necessary to build the JasPer software are provided.

In what follows, `$TOPDIR` denotes the top level directory of the JasPer software distribution (i.e., the directory containing the file named `configure`). To build the software, the following steps are required (in order):

1. *If necessary, install the OpenGL and GLUT libraries on your system.*

   These libraries are required in order to build the jiv application program which is included in the JasPer software. The JasPer library itself and the other application programs included with JasPer can be built without the OpenGL and GLUT libraries, however.

2. *Run Microsoft Visual C.*

3. *Open the JasPer workspace file.*

   The JasPer workspace file is called `jasper.dsw` and can be found in the directory `$TOPDIR/src/msvc`. The workspace file can be opened using the “File” menu.

4. *Build the code.*

   From the "Build Menu", select the "Batch Build" item. Choose the projects/configurations that are to be built, and then, click on the "Build" button. If you do not have the OpenGL and GLUT libraries (and their associated header files) installed on your system, you should not attempt to build the jiv application.

Presuming that the build was successful, the release and debug versions of the executables for the JasPer software can be found in the directories `$TOPDIR/src/msvc/Win32_Release` and `$TOPDIR/src/msvc/Win32_D` respectively.

6.3 Dependencies on Other Software

In order to have access to the full functionality of the JasPer sofware, you may need to install some additional software on your system. This software must be installed before you attempt to build JasPer.

In order to compile the JasPer software with JPEG support, you will need to download and install the free IJG JPEG library which is available from the IJG web site (i.e., `http://www.ijg.org`). Then, use the `jpg_enc.c` and `jpg_dec.c` files instead of `jpg_dummy.c` to build the JasPer library.

In order to build the jiv application, you will need the OpenGL and GLUT libraries installed on your system. It would appear that Windows 2000 and Windows XP ship with the OpenGL library from Microsoft. Most recent versions of UNIX ship with OpenGL support included. The GLUT library is less common and, therefore, may not be installed on your system. To obtain the GLUT library, one can visit: `http://www.opengl.org/developers/documentation/glut`. For more information on the OpenGL library, see: `http://www.opengl.org`.

At the time of this writing, a binary distribution of the GLUT library for Windows is available from the OpenGL web site:

`http://www.opengl.org/developers/documentation/glut/glutdlls37beta.zip`

This archive file contains three files of interest: 1) `glut.h`, 2) `glut32.lib`, and 3) `glut32.dll`. The file `glut.h` file should be installed in a directory in which the C compiler searches for header files (e.g., `$TOPDIR/src/include/jasper`). The `glut32.lib` file should be installed in a directory in which the C compiler searches for libraries. The `glut32.dll` file should be installed in a directory in which the system looks for dynamically-linked libraries.

7 The libjasper Library

The libjasper library is the heart of the JasPer software. This library provides means to encode, decode, and otherwise manipulate image data. Several different codecs are supported as detailed in the section on codecs (i.e., Section 13). The support for codecs is provided in a very modular manner. This allows support for other codecs to be easily added.

The library consists of two distinct categories of code: 1) base/core code, and 2) codec drivers. The base code provides generic routines for manipulating images and provides a framework for constructing codec drivers. The codec drivers provide a means for encoding/decoding images in

specific formats. The library has been designed to be extensible so that adding support for new image formats is a straightforward exercise.

In order to use libjasper, a C source file should include the main JasPer library header file `jasper/jasper.h`. This can be accomplished with the following preprocessor directive:

```
#include <jasper/jasper.h>
```

The main header file includes all of the other library header files. Therefore, in order to insulate application code from possible changes to the names of the other library header files, one should only ever include the main library header directly.

The first usage of the library must always be to initialize it. This is accomplished by calling `jas_init`. If any other functionality of the library is used before this initialization is performed, the resulting behavior is undefined.

Briefly, the JasPer library provides the following key classes:

- An image class (i.e., `jas_image_t`). This class is used to represent an image, and also provides access to codec drivers for encoding/decoding image data in various formats.
- A sequence/matrix class. This class provides matrix and two-dimensional sequence classes.
- A I/O stream class (i.e., `jas_stream_t`). This class provides I/O streams similar to that of the standard C library, but with additional functionality required by other code in JasPer.
- A fixed-point number class. This templated class is used for performing fixed-point arithmetic.
- A tag-value parser class. This class facilitates the parsing of tag-value pairs. A tag-value pair is a string of the form "tag=value". Such pairs are used by some interfaces within JasPer in order to pass parameters.
- A command line parser class. This class allows the parsing of command lines. This code is inspired by the getopt function available on most UNIX systems.

All memory allocation in the libjasper library is performed via the functions `jas_malloc`, `jas_realloc`, `jas_calloc`, and `jas_free`. If one is trying to port the JasPer code to an embedded platform, it might be necessary to change these functions to use a platform-specific memory allocator, rather than `malloc` and friends.

Support for new image formats can be easily added to JasPer. In order to add support for a new image format, one must provide three functions:

1. an encoding function,
2. a decoding function, and
3. a validation function.

The encoding function emits the coded version of an image (i.e., an jas_image_t object) to a stream (i.e., a jas_stream_t object). The decoding function creates an image (i.e., an jas_image_t object) from the coded data in a stream (i.e., a jas_stream_t object). The validation function is used to quickly test if a data stream is formatted correctly for the image format in question. (This functionality is necessary for the autodetection of image formats.)

The exact types and arguments taken by the encoding, decoding, and validation functions can be found by examining the code for the image formats already included in JasPer (e.g., PNM, BMP, JPEG). Once the above functions have been written, the JasPer library can be made aware of the new image format through a call to jas_image_addfmt. This call, of course, must be made after the JasPer library has been initialized. The code for the jas_init function gives several examples of how the jas_image_addfmt function might be invoked.

8 Demo Application Programs

The JasPer software consists of the following programs:

1. jasper. The JasPer transcoder.
2. imgcmp. An image comparison utility.
3. imginfo. An image information utility.
4. jiv. A simple image viewer.

The use of these commands is described in the sections that follow.

9 The jasper Command

Synopsis

jasper [options]

Description

The jasper command converts image data from one format to another. In other words, this command functions as a general-purpose transcoder. Since the JPEG-2000 format is supported by this software, it can be used as a JPEG-2000 encoder and/or decoder.

Options

The jasper program accepts the following options:

--help
: Print usage information and exit.

--version
: Print the version and exit.

`--input` *file*
`-f` *file*
Read the input image from the file named *file*. By default, the input image is read from standard input.

`--input-format` *format*
`-t` *format*
Specify the format of the input image as *format*. In most circumstances, this option should not be needed, as the format is normally autodetected by examining the image data directly or deduced from the input file name extension if an input file is specified (via the `--input` option).

`--input-option` *option*
`-o` *option*
Provide the option *option* to the decoder. The valid values for the argument *option* are determined by the input image format. See below for more details.

`--output` *file*
`-F` *file*
Write the encoded image to the file named *file*. By default, the encoded image is written to standard output.

`--output-format` *format*
`-T` *format*
Produce the output image in the format indicated by *format*. The output format must be specified if an output file is not given (via the ”–output” option). If an output file is given and no output format is specified, an attempt will be made to deduce the correct format from the output file name extension.

`--output-option` *option*
`-O` *option*
Provide the option *option* to the encoder. The valid values for the argument *option* are determined by the output format. See below for more details.

`--verbose`
Enable verbose mode.

The argument *format* must have one of the following values:

Value	Description
bmp	Windows BMP
jp2	JPEG-2000 JP2
jpc	JPEG-2000 Code Stream
jpg	JPEG
pgx	PGX
pnm	PNM/PGM/PPM
mif	My Image Format
ras	Sun Rasterfile

Examples

1. Suppose that we have an image stored in the PNM/PPM format in a file called lena.ppm. To encode this image (losslessly) in the JPEG-2000 JP2 format, and store the result in a file called lena.jp2, type:

   ```
   jasper --input lena.ppm --output lena.jp2 --output-format jp2
   ```

   Or, alternately (using short option names), type:

   ```
   jasper -f lena.ppm -F lena.jp2 -T jp2
   ```

2. Suppose that we have a RGB color image stored in the JPEG-2000 JP2 format in a file called lena.jp2. To encode this image in the PNM/PPM format, and store the result in a file called lena.ppm, type:

   ```
   jasper --input lena.jp2 --output lena.ppm --output-format pnm
   ```

   Or, alternately (using short option names), type:

   ```
   jasper -f lena.jp2 -F lena.ppm -T pnm
   ```

3. Suppose that we have an image stored in the BMP format in a file called lena.bmp. To encode this image in a lossy manner at 100:1 compression in the JPEG-2000 (code stream) format, and store the result in a file called lena_lossy.jpc, type:

   ```
   jasper -f lena.bmp -F lena_lossy.jpc -T jpc -O rate=0.01
   ```

4. Suppose that we have an image stored in a file called sachie.pnm in the PNM/PPM format, and we want to encode the image in the JPEG-2000 (code stream) format and store the result in a file named sachie_new.jpc. Further, suppose that we want the JPEG-2000 format to employ the following parameters:

   - code blocks are 64 samples in width and 32 samples in height
   - no multicomponent transform is to be employed
   - 4 resolution levels should be employed for each component
   - the compression is lossy at 64:1

   In order to accomplish the above, type:

   ```
   jasper -f sachie.pnm -F sachie_new.jpc -T jpc -O cblkwidth=64 -O cblkheight=32
   -O nomct -O numrlvls=4 -O rate=0.015625
   ```

10 The imgcmp Command

Synopsis

imgcmp [options]

Description

The imgcmp command compares two images. The two images being compared must have the same geometry (i.e., the same width, height, number of components, component subsampling factors, etc.).

Options

The following options are supported:

`-f` *file*
: Read the primary (i.e., reference) image (for comparison purposes) from the file named *file*.

`-F` *file*
: Read the secondary image (for comparison purposes) from the file named *file*.

`-m` *metric*
: Use the difference metric specified by *metric*. The *metric* argument may assume one of the following values:

Value	Description
psnr	peak signal to noise ratio (PSNR)
mse	mean squared error (MSE)
rmse	root mean squared error (RMSE)
pae	peak absolute error (PAE)
mae	mean absolute error (MAE)
equal	equality

The `-f` and `-F` options must always be specified. There is currently no way to explicitly specify the format of the images. If the format of either image cannot be autodetected, the command will exit with an error.

Examples

1. Suppose that we have two slightly different versions of an image stored in files `original.pgm` and `reconstructed.pgm`. In order to calculate the difference between these images using the PSNR metric, type:

```
imgcmp -f original.pgm -F reconstructed.pgm -m psnr
```

11 The imginfo Command

Synopsis

`imginfo` [options]

Description

The imginfo command displays information about an image. This command is really only intended to be used from shell scripts for testing purposes.

12 The jiv Command

Synopsis

`jiv` [options] [file1 file2 ...]

Description

The jiv command displays an image. Basic pan and zoom functionality is provided. Components of an image may be viewed individually. Color components may also be viewed together as a composite image. At present, the jiv image viewer has only trivial support for color. It recognizes RGB and YCbCr color spaces, but does not use tone reproduction curves and the like in order to accurately reproduce color. For basic testing purposes, however, the color reproduction should suffice.

Options

The following options are supported:

`--help`
: Print help information and exit.

`--wait` *n*
: Automatically step from one image to the next, pausing for *n* seconds in between.

13 Codecs

The JasPer software provides implementations of several popular codecs. Likely, the ones of most interest are those associated with the JPEG-2000 standard. The sections that follow describe the various codecs in more detail.

13.1 BMP Codec

One of the most popular image formats on the Microsoft Windows platform is Microsoft's BMP format. The BMP codec in JasPer was written without the benefit of the BMP format specification from Microsoft. This means that the BMP support will inevitably not work correctly for all valid BMP files.

Encoder Options

The encoder does not support any special options.

Decoder Options

The decoder does not support any special options.

13.2 JP2 Codec

One of the two image formats specified in the JPEG-2000 Part-1 standard (i.e., ISO/IEC 15444-1 [4]) is the so called "JP2" format.

Encoder Options

The encoder supports all of the same options as the JPC encoder.

Decoder Options

The decoder supports all of the same options as the JPC decoder.

13.3 JPC Codec

One of the two image formats specified in the JPEG-2000 Part-1 standard (i.e., ISO/IEC 15444-1 [4]) is the so called JPEG-2000 code stream format. The JPC codec in JasPer implements this format.

The design of the JPEG-2000 codec implementation was driven by several key concerns: execution speed, memory usage, robustness, portability, modularity, maintainability, and extensibility. In some cases, however, during the design process, modularity, portability, and understandability of the code were weighed more heavily than execution speed and memory usage. Code understandability and portability were critical considerations since this software was intended to be used as a reference implementation of the JPEG-2000 Part-1 codec in the JPEG-2000 Part-5 standard [5].

Since the JPEG-2000 standard does not specify any means for encoding color space information in a JPEG-2000 code stream, the decoder must make certain assupmtions about the color space of an image. If accurate color representation is important, the JPEG-2000 code stream format should not be employed. The JPEG-2000 JP2 format should be used instead.

Encoder Options

The following options are supported by the encoder:

`imgareatlx=`*x*
: Set the x-coordinate of the top-left corner of the image area to *x*.

`imgareatly=`*y*
: Set the y-coordinate of the top-left corner of the image area to *y*.

`tilegrdtlx=`*x*
: Set the x-coordinate of the top-left corner of the tiling grid to *x*.

`tilegrdtly=`*y*
Set the y-coordinate of the top-left corner of the tiling grid to *y*.

`tilewidth=`*w*
Set the nominal tile width to *w*.

`tileheight=`*h*
Set the nominal tile height to *h*.

`prcwidth=`*w*
Set the precinct width to *w*. The argument *w* must be an integer power of two. The default value is 32768.

`prcheight=`*h*
Set the precinct height to *h*. The argument *h* must be an integer power of two. The default value is 32768.

`cblkwidth=`*w*
Set the nominal code block width to *w*. The argument *w* must be an integer power of two. The default value is 64.

`cblkheight=`*h*
Set the nominal code block height to *h*. The argument *h* must be an integer power of two. The default value is 64.

`mode=`*m*
Set the coding mode to *m*. The argument *m* must have one of the following values:

Value	Description
int	integer mode
real	real mode

If lossless coding is desired, the integer mode must be used. By default, the integer mode is employed. The choice of mode also determines which multicomponent and wavelet transforms (if any) are employed.

`rate=`*r*
Specify the target rate. The argument *r* is a positive real number. Since a rate of one corresponds to no compression, one should never need to explicitly specify a rate greater than one. By default, the target rate is considered to be infinite.

`ilyrrates=`$r_0[,r_1,\ldots,r_{N-1}]$
Specify the rates for any intermediate layers. The argument to this option is a comma separated list of N rates. Each rate is a positive real number. The rates must increase monotonically. The last rate in the list should be less than or equal to the overall rate (as specified with the `rate` option).

`prg=`*p*

Set the progression order to *p*. The argument *p* must have one of the following values:

Value	Description
lrcp	layer-resolution-component-position (LRCP) progressive (i.e., rate scalable)
rlcp	resolution-layer-component-position (RLCP) progressive (i.e., resolution scalable)
rpcl	resolution-position-component-layer (RPCL) progressive
pcrl	position-component-resolution-layer (PCRL) progressive
cprl	component-position-resolution-layer (CPRL) progressive

By default, LRCP progressive ordering is employed. Note that the RPCL and PCRL progressions are not valid for all possible image geometries. (See [4] for more details.)

`nomct`

Disallow the use of any multicomponent transform.

`numrlvls=`*n*

Set the number of resolution levels to *n*. The argument *n* must be an integer that is greater than or equal to one. The default value is 6.

`sop`

Generate SOP marker segments.

`eph`

Generate EPH marker segments.

`lazy`

Enable lazy coding mode (a.k.a. arithmetic coding bypass).

`termall`

Terminate all coding passes.

`segsym`

Use segmentation symbols.

`vcausal`

Use vertically stripe causal contexts.

`pterm`

Use predictable termination.

`resetprob`

Reset the probability models after each coding pass.

`numgbits=`*n*

Set the number of guard bits to *n*.

Decoder Options

The decoder does not support any special options.

Rate Specification

All rates are specified in terms of compression factors (i.e., as reciprocals of compression ratio) and not as actual bit rates! Although image coding folks frequently use the number of bits per pixel to specify rate, this quantity is often inconvenient to use when dealing with images that have differing sample precisions. Furthermore, the number of bits per pixel is not well defined for multicomponent images with distinct subsampling factors. The compression factor, however, is independent of sample precision and well defined for all types of images. For these reasons, JasPer uses the compression factor and not the number of bits per pixel to specify rates.

13.4 JPG Codec

For lossy coding, one of the most popular image formats is specified in the JPEG standard (i.e., ISO/IEC 10918-1 [6]). In JasPer, the JPG codec implements this format.

The JPEG support in JasPer requires the JPEG library from the Independent JPEG Group (IJG). For legal reasons, the IJG JPEG library source code is not included with JasPer. The source code for this library can be downloaded from the IJG web site (i.e., http://www.ijg.org).

Encoder Options

The JPG encoder does not support any special options.

Decoder Options

The decoder does not support any special options.

13.5 PGX Codec

The JPEG-2000 Verification Model software employs a non-standard format called PGX. In JasPer, this format is handled by the PGX codec. The PGX format can only handle single components images, and consequently, is of limited use.

Encoder Options

The encoder does not support any special options.

Decoder Options

The decoder does not support any special options.

13.6 MIF Codec

The MIF format is not a standard format. This format was invented solely for the purpose of testing the JasPer software. The support for the MIF format is experimental. It is intended to be used for advanced testing of the JasPer JPEG-2000 codec implementation.

Encoder Options

The encoder does not support any special options.

Decoder Options

The decoder does not support any special options.

13.7 PNM Codec

On UNIX platforms, the Portable Pixmap/Graymap/Bitmap (PNM) format is quite popular for coding image data. The PNM codec in JasPer supports this format. In JasPer, the support for the PNM/PGM/PPM format is complete. Therefore, the use of this format is favored over the BMP format. A (nonstandard) extension has also been added to the support for the PNM format so that it can handle images with signed sample values.

Encoder Options

The following options are provided by the PNM encoder:

`text`
: Use a non-raw (i.e., non-binary) flavor of PNM format.

Decoder Options

The decoder does not support any special options.

13.8 RAS Codec

One popular image format on Sun workstations is the Sun Rasterfile format. The RAS codec in JasPer implements this format.

Encoder Options

The encoder does not support any special options.

Decoder Options

The decoder does not support any special options.

14 Reporting Bugs

If you are unfortunate enough to encounter any problems with the JasPer software, please submit a bug report. In order to ensure that your bug report can be properly processed, always be sure to include *all* of the following information:

- The version of JasPer in which the problem was found.
- The details of the run-time system (i.e., operating system, version number).
- The compiler that you are using (i.e., vendor, version number).
- The exact command line options used when the problem was observed.
- Indicate whether or not the problem is reproducible. If the problem is reproducible, indicate the exact steps required to reproduce the problem.
- A detailed description of the problem that you are experiencing.

It is essential that you include all of the above information. Failure to do so may result in the bug report not being processed. Your complete bug report should be sent to mdadams@ieee.org.

15 Additional JasPer Resources

For more information about the JasPer software, please visit the following web pages:

- JasPer Project Home Page (i.e., http://www.ece.uvic.ca/~mdadams/jasper)
- JasPer Software Announcements Group (i.e., http://groups.yahoo.com/group/jasper-announce)

  By joining this group, one can easily obtain up-to-date information about the JasPer Project in a timely fashion. The associated mailing list is used to announce new releases of the software and disseminate other important information about JasPer.
- JasPer Software Discussion Group (i.e., http://groups.yahoo.com/group/jasper-discussion).

For more information about the JasPer software and JPEG-2000 standard, the reader is referred to [7, 8].

16 Origin of the Name

The JasPer software is named, in part, after Jasper National Park, the largest national park in the Canadian Rockies with 10,878 square kilometres of mountain wilderness. As it happens, jasper is also the name of an opaque cryptocrystalline variety of quartz used for ornamentation or as a gemstone—hence, the implication that the software is precious (i.e., like a gemstone). Lastly, the name "jasper" was also chosen because it contains a letter "J" followed subsequently by a letter "P", not unlike the abbreviation "JP" that is associated with the JPEG-2000 standard.

References

[1] B. W. Kernighan and D. M. Ritchie, *The C Programming Language*, Prentice Hall, Englewood Cliffs, NJ, USA, 2nd edition, 1988.

[2] *ISO/IEC 9899: Programming languages—C*, 1999.

[3] *ISO/IEC 9945-1: Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language]*, 1990.

[4] *ISO/IEC 15444-1: Information technology—JPEG 2000 image coding system—Part 1: Core coding system*, 2000.

[5] *ISO/IEC 15444-1: Information technology—JPEG 2000 image coding system—Part 5: Reference software*, 2002.

[6] *ISO/IEC 10918-1: Information technology—Digital compression and coding of continuous-tone still images: Requirements and guidelines*, 1994.

[7] M. D. Adams and F. Kossentini, "JasPer: A software-based JPEG-2000 codec implementation," in *Proc. of IEEE International Conference on Image Processing*, Vancouver, BC, Canada, Oct. 2000.

[8] M. D. Adams, "The JPEG-2000 still image compression standard," ISO/IEC JTC 1/SC 29/WG 1 N 2412, Sept. 2001, Available from http://www.ece.uvic.ca/~mdadams and distributed with the JasPer software.