JASPER: A PORTABLE FLEXIBLE OPEN-SOURCE SOFTWARE TOOL KIT FOR IMAGE CODING/PROCESSING

Michael D. Adams[†] *and Rabab K. Ward*[‡]

[†]Dept. of Elec. and Comp. Engineering
University of Victoria
Victoria, BC, V8W 3P6, Canada
E-mail: mdadams@ece.uvic.ca

[‡]Dept. of Elec. and Comp. Engineering
University of British Columbia
Vancouver, BC, V6T 1Z4, Canada
E-mail: rababw@ece.ubc.ca

ABSTRACT

JasPer, a portable flexible open-source software tool kit for handling image data is described. This software provides a means for representing images, and facilitates the manipulation of image data as well as its import/export in various formats (such as JPEG 2000). JasPer is proving to be an extremely useful tool in a wide variety of applications, ranging from image coding/processing research to open-source and proprietary software development.

1. INTRODUCTION

Digital imagery is used in many of today's computer software applications. Consequently, software modules that facilitate the handling of such data are often needed. Almost any application program that deals with images must address the problem of image interchange and import/export. That is, a means must exist for moving image data between an application and its external environment. Moreover, many applications must be capable of rendering an image for display on a particular output device (such as a monitor or printer) with reasonably accurate color/tone reproduction.

Although image import/export and rendering are very basic functionalities, they are often not easily implemented. Usually, an image is represented in a coded format (such as JPEG-2000 JP2 [1] or JPEG [2]). Since coding formats are often quite complex, the import/export of image data can be a daunting task. Rendering an image in such a way as to accurately reproduce color/tone requires a color management scheme of some sort. Unfortunately, developing an effective color management engine can require considerable effort.

In addition, the development of efficient coding schemes for image interchange is a complex and time-consuming process. In fact, many researchers expend considerable energy developing new coding schemes. Normally, part of this process requires the implementation of the proposed coding algorithm. Although the implementation of the coding algorithm itself can be quite time consuming, some framework in which to implement the algorithm is also needed. This framework typically provides data types for representing images and other related information, allows for the import/export of image data using one or more standard formats, and provides some level of support for image rendering. Unfortunately, the development of this additional framework requires considerable effort, and detracts from the researcher's objective of testing/evaluating the new coding algorithm.

To address the problems/issues described above, we have developed a software tool kit known as JasPer [3]. In this paper, we introduce the JasPer software, and describe its functionality and structure, as well as its development history. We also briefly characterize the current user base for the software, and give examples of its use. In so doing, we demonstrate that JasPer provides an attractive solution for handling image data in many diverse applications.

2. HISTORICAL PERSPECTIVE

Before proceeding further, a short digression concerning the history of JasPer is appropriate. The JasPer software was initially developed in order to provide a free reference implementation of the JPEG-2000 Part-1 codec (as described in [4]). Work on the software began in September 1999, and in December 2000, the software was first released in source code form to the general public. The JasPer software has since been published in the JPEG-2000 Part-5 standard [5] as a reference implementation of the JPEG-2000 Part-1 codec.

Since the time of [4] (circa JasPer version 0.016.0), JasPer has continued to evolve, and in so doing, the motivations/objectives behind the development of this software have grown more ambitious. Although originally conceived as simply a JPEG-2000 reference implementation (i.e., as described in [4]), JasPer has further evolved into a general-purpose tool kit for handling image data that is useful in a wide variety of applications. In the remainder of this paper, we examine JasPer as it exists today, and briefly mention some of the many changes made as JasPer developed from its earlier versions to the current day version.

3. SOFTWARE OVERVIEW

In simple terms, JasPer is a software tool kit for the handling of image data. The software provides a means for representing images, and facilitates the manipulation of image data, as well as the import/export of such data in numerous formats (e.g., JPEG-2000 JP2 [1], JPEG [2], PNM [6], BMP [7], Sun Rasterfile [8], and PGX [9]). The import functionality supports the auto-detection (i.e., automatic determination) of the image format, eliminating the need to explicitly identify the format of coded input data. A simple color management engine is also provided in order to allow the accurate representation of color. Partial support is included for the ICC color profile file format [10], an industry standard for specifying color.

The JasPer software consists of a library and several application programs. The code is written in the C programming language [11]. This language was chosen primarily due to the availability of C development environments for most of today's computing platforms. At present, JasPer consists of approximately 40K lines of code. Although written in C, the JasPer library can be easily integrated into applications written in the C++ programming language as well.

Portability has been (and continues to be) a major consideration in the development of JasPer. For this reason, the software makes minimal assumptions about the compile- and run-time environments. The code should compile and run on any platform with a C language implementation that is reasonably compliant with the (most recent) ISO C language standard [11] and has some limited support for the POSIX C API [12]. For example, JasPer compiles and runs (without modification) under Microsoft Windows and most mainstream flavors of UNIX (e.g., Red Hat Linux, Solaris, AIX, HP-UX, and IRIX).

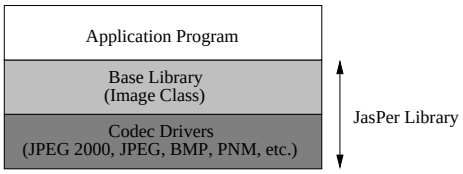

Fig. 1. Software structure.

Two different build methods for the software are supported. The primary build method (for UNIX and UNIX-like systems) is based on the well-known make, autoconf, libtool, and automake tools. This build process can generate dynamic or static libraries. A second build method is provided in order to accommodate the building of JasPer with Microsoft Visual C (MSVC) under Microsoft Windows. This build method is based on MSVC workspace and project description files (which are provided with JasPer).

For maximal portability, JasPer does not rely on any non-standard proprietary libraries. Only standard libraries and other free software are employed. In order to provide JPEG support, JasPer relies on the popular IJG JPEG library [13]. For graphics support, JasPer makes use of the OpenGL [14] and GLUT [15] libraries. These particular libraries were chosen due to their availability on many platforms, including Microsoft Windows and most flavors of UNIX.

The JasPer software is available for download from the Internet. For further details on obtaining the software, the reader is referred to Appendix A.

4. JASPER LIBRARY

The heart of the JasPer software is the JasPer library. In fact, most of the code in JasPer is associated with this library (as opposed to the JasPer sample application programs). The JasPer library provides classes for representing images, color profiles (i.e., color space definitions), and other related entities. Each of these classes has a well-defined interface through which an application may interact with class objects. The library can be used to manipulate images, import/export image data in a variety of formats, and perform basic color management operations.

Conceptually, the JasPer library is structured as shown in Fig. 1. The library consists of two distinct types of code: 1) core code, and 2) codec drivers. The core code provides the basic framework upon which the library is built, while the codec drivers only provide the means for encoding/decoding image data in various formats. All application interfaces are through the core code. The codec drivers are only ever directly called by the core code, never by an application.

The codec support in the JasPer library is both modular and extensible. A well-defined interface exists between the core code and codec drivers. Moreover, support for a new image format can be easily added without having to modify the library in any way. To do so, a codec driver for the new format simply needs to be provided. Furthermore, an application need only include codec drivers for the image formats that it will use. In this way, an application can avoid the cost of increased memory consumption for codec drivers that are never to be employed.

4.1. Core Code

The core code provides a number of key classes as described below. (To avoid name space collisions, all of the identifiers used by the core code are prefixed with either `jas_` or `JAS_`.)

1. Image class (i.e., `jas_image_t`). This class is used to represent an image. Methods are provided for such things as: image creation/destruction, querying general image properties (e.g., reference grid width and height, color profile), querying component properties (e.g., width, height, grid offset, grid spacing, component type, sample precision/signedness), setting various image properties, loading and saving an image (i.e., encoding/decoding), copying an image, adding and deleting components, and reading and writing component data.
2. Color profile class (i.e., `jas_colorprof_t`). This class is used to define a color space. Such a definition is normally made relative to a reference color space such as CIE XYZ or CIE Lab [10]. Methods are provided for such things as: profile creation/destruction, loading and saving profiles, connecting profiles (to facilitate color space conversion), and applying a profile to image data (i.e., performing a color space conversion).
3. Stream class (i.e., `jas_stream_t`). This class provides I/O streams similar to that of standard C library [11], but with additional functionality required by other code in the JasPer library. This extra functionality includes: 1) the ability to associate an object other than a file descriptor with a stream (such as a memory buffer), and 2) multi-character unget.
4. Fixed-point number class. This templated class (i.e., a set of macros) provides a fixed-point number class. Support is provided for basic arithmetic operations, type conversion, and rounding.
5. Tag-value parser class (i.e., `jas_tvp_t`). This class is used to parse tag-value pairs (i.e., strings of the form "TAG=VALUE"). Tag-value pairs are used to pass options to codec drivers for encoding/decoding operations. Methods are provides for such things as: creation/destruction, finding the next tag-value pair in a string, and querying the current tag and value.

In addition to the above classes, some other functionality is provided, including command line parsing routines (similar in spirit to UNIX `getopt`).

4.2. Codec Drivers

The core code provides a framework for housing codec drivers. A codec driver provides the means for encoding/decoding of image data in a particular format. Each driver provides three methods: 1) an encoding, 2) decoding, and 3) validation method. The encoding method emits the coded version of an image (i.e., a `jas_image_t` object) to a stream (i.e., a `jas_stream_t` object). The decoding method creates an image (i.e., a `jas_image_t` object) from the coded data in a stream. The validation method is used to quickly test if the data in a stream is formatted correctly for the image format in question. This particular method is used for the autodetection of image formats.

The codec drivers provided with the JasPer distribution are written in order to accommodate streamed data. In other words, image data streams are always processed in a single pass. This design philosophy eliminates the need for a stream object to be seekable. As a result, it is possible to write application programs that receive data from, or send data to, pipelines or other entities that do not support random access to data.

5. IMAGE MODEL

The set of applications for which JasPer may be a useful tool is dictated, in part, by the image model that JasPer employs. Therefore, it is prudent to introduce this model here. The image model employed by JasPer is quite general and partially inspired by the one used in the JPEG-2000 standard.

An image is comprised of one or more components. In turn, each component consists of rectangular array of samples. This

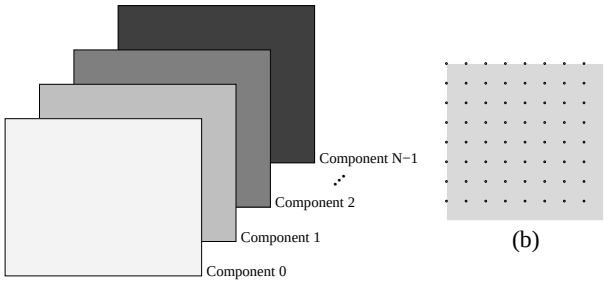

Fig. 2. Image model. (a) An image with N components. (b) Individual component.

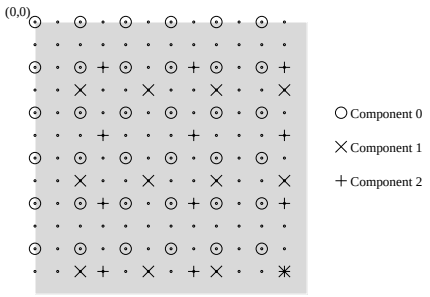

Fig. 3. Alignment of components on the reference grid.

structure is depicted in Fig. 2. The sample values for each component are integer valued, and can be signed or unsigned with precision from 1 to (nominally) 16 bits/sample[1]. The signedness and precision of the sample data are specified on per-component basis. All of the components are associated with same spatial extent in an image, but represent different types of information.

There is considerable flexibility in the interpretation of components. A component may represent spectral information (e.g., a color plane) or auxiliary information (e.g., an opacity plane). For example, a RGB image would have three components, where one component is associated with each of the red, green, and blue color planes. A RGBA (i.e., RGB with transparency) image would have four components, one associated with each of the red, green, blue, and alpha planes. The various components need not be sampled at the same resolution. In other words, different components may have different sampling periods. For example, when color images are represented in a luminance-chrominance color space, it is not uncommon for the luminance information to be more finely sampled than the chrominance information.

Since an image can have a number of components, a means must exist for specifying how these components are combined together in order to form a composite image. For this purpose, we employ an integer lattice known as the reference grid. The reference grid provides an anchor point for the various components of an image, and establishes their alignment relative to one another.

Each component is associated with a rectangular sampling grid. Such a grid is uniquely specified by four parameters: the horizontal offset (HO), vertical offset (VO), horizontal spacing (HS), and vertical spacing (VS). The samples of a component are then mapped onto the points where the sampling grid intersects the reference grid. In this way, sample (i, j) of a component is mapped to the position $(\mathrm{HO} + i\mathrm{HS}, \mathrm{VO} + j\mathrm{VS})$ on the reference grid.

To clarify the above text, we now present an illustrative example. Consider an image with three components. For the kth component, let us denote the horizontal grid offset, vertical grid offset, horizontal grid spacing, and vertical grid spacing, as HO_k, VO_k, HS_k, and VS_k, respectively. Suppose, for example, that these parameters have the following values:

k	$(\mathrm{HO}_k, \mathrm{VO}_k)$	$(\mathrm{HS}_k, \mathrm{VS}_k)$
0	(0, 0)	(2, 2)
1	(2, 3)	(3, 4)
2	(3, 2)	(4, 3)

In this scenario, the component samples would be aligned on the reference grid as illustrated in Fig. 3. Perhaps, it is worth nothing that the above set of parameter values was chosen in order to provide an enlightening example, and is not meant to represent a set of values that is likely to be used with great frequency by applications.

From above, we can see that the image model used by JasPer is quite general. The main constraint imposed by this model is that rectangular sampling must be employed. The vast majority of applications, however, use such sampling. Also, with JasPer, one can easily accommodate grayscale, color, and other multi-band data (with or without opacity information).

6. APPLICATION PROGRAMS

In order to demonstrate how the JasPer library can be used, several sample application programs are provided in the JasPer software distribution. These programs include: `jasper`, `jiv`, `imgcmp`, and `imginfo`. The `jasper` program is an image transcoder (i.e., it converts image data from one format to another). It can handle image data in any format supported by the JasPer library (e.g., JPEG 2000, JPEG, PNM, BMP). The `jiv` program is a simple image viewer (based on OpenGL). It provides basic pan and zoom functionality as well as a slideshow capability. The components of an image may be viewed individually, or, in the case of color images, together as a single composite image. The `imgcmp` program is an image comparison utility. It measures the difference between two images using one of numerous distortion metrics (such as peak signal-to-noise ratio, mean squared error, root mean squared error, peak absolute error, and mean absolute error). The `imginfo` program provides basic information about an image, such as its geometry (i.e., number of components, width and height of components, and so on). Although the above-mentioned programs are intended mainly for demonstration purposes, they have also proven quite useful in their own right, especially the `jasper` and `jiv` programs.

7. EVOLUTION OF JASPER

Extensive changes have been made to JasPer since its early days (i.e., circa version 0.016.0 [4]). In what follows, we highlight some of these changes. First, the build process for the software has been completely reworked. As a result, the software builds reliably (and without modification) on a much wider range of systems. The documentation for the software has been very substantially improved, and now includes a manual [3] and detailed technical tutorial on JPEG 2000 [16]. The library interface has changed considerably over time. This was necessary in order to accommodate arbitrary image formats. Support for a number of new image formats has been added (e.g., JPEG and PGX), and support for previously existing formats has been enhanced. The JPEG-2000 codec driver is now complete, whereas in earlier versions some functionality was missing (e.g., region-of-interest decoding, precincts, and the JP2 file format). Some new applications have been included in the JasPer distribution (e.g., the `jiv` image viewer) and more functionality has been added to some of the previously existing appli-

[1]The maximum allowable precision is platform dependent. Most common platforms, however, should be able to accommodate at least 16 bits/sample.

cations. Some small test images are also included in the distribution for testing purposes. A testing framework for the software has also been developed. This framework, although available only to those developing JasPer, helps to ensure the stability of the software from release to release.

8. JASPER USER COMMUNITY

Since the time of its first public release, the JasPer software has acquired a relatively large user base, and this user base continues to grow steadily over time. At the time of this writing, the server hosting the JasPer Project web pages averages about 3,500 downloads of the software per month.

The JasPer library is potentially useful for many image processing applications (such as image coding, enhancement, restoration, and analysis). For this reason, the JasPer user community is quite diverse, consisting of academics, researchers, software developers, and others. Many users are undoubtedly interested in the JPEG-2000 codec implementation provided by JasPer. Some are employing JasPer for research relating to JPEG 2000 (e.g., [17]). Many software projects are known to be using JasPer, including: 1) the K Desktop Environment (KDE) [18], a powerful open-source graphical desktop environment for UNIX workstations, 2) the Netpbm utilities [6], 3) the Checkmark software [19], and 4) the Geospatial Data Abstraction Library (GDAL) [20], a translator library for raster geospatial data formats.

9. CONCLUSIONS AND FUTURE WORK

In this paper, we have described JasPer, a software tool kit for handling image data. The functionality and structure of the software were examined. In particular, the library and application programs associated with JasPer were introduced. We also briefly characterized some of the uses of JasPer. Judging by the size of the JasPer user community, one can convincingly argue that this software is proving itself to be a useful tool for many.

Although JasPer already embodies a very substantial amount of work, further development of the software will undoubtedly continue in the future. Such development might include: 1) improving the performance of the JPEG-2000 codec implementation, 2) adding support for additional formats such as JPEG LS, PNG, TIFF, and PAM (a new addition to the PNM family of formats), 3) improving the software documentation, 4) adding support for multi-threaded execution, and 5) extending the functionality provided by the color management engine.

As JasPer development continues, we hope that this software will become an even more useful tool for the academic/research community, commercial organizations, and others. To assist in achieving this goal, we both welcome and encourage feedback from the user community. Through such feedback, we can further improve the utility of the software to the benefit of all.

A. OBTAINING THE SOFTWARE

The JasPer software is available online from the JasPer Project home page (i.e., `http://www.ece.uvic.ca/~mdadams/jasper`) and the JPEG software home page (i.e., `http://www.jpeg.org/software`). At the time of this writing, the most recent version of the software is 1.700.5.

ACKNOWLEDGMENT

The authors would like to thank Image Power, Inc. for their past support of JasPer software development. In addition, the authors would like to thank the many devoted users of JasPer for their feedback which has helped to greatly improve the quality of the software.

REFERENCES

[1] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 15444-1:2000, Information technology—JPEG 2000 image coding system—Part 1: Core coding system*.

[2] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 10918-1:1994, Information technology—Digital compression and coding of continuous-tone still images: Requirements and guidelines*.

[3] M. D. Adams, "JasPer software reference manual," ISO/IEC JTC 1/SC 29/WG 1 N 2415, Dec. 2002.

[4] 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, vol. 2, pp. 53–56.

[5] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 15444-5:2002 Information technology—JPEG 2000 image coding system—Part 5: Reference software*.

[6] "Netpbm home page," `http://netpbm.sourceforge.net`, 2003.

[7] M. Luse, "The BMP file format," *Dr. Dobb's Journal*, vol. 9, no. 10, pp. 18–22, Sept. 1994.

[8] Sun Microsystems, Inc., *OpenWindows Reference Manual (Version 3.4)*, 1994.

[9] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 15444-4:2002, Information technology—JPEG 2000 image coding system—Part 4: Compliance testing*.

[10] International Color Consortium, *ICC.1:2001-12, File Format for Color Profiles (Version 4.0.0)*, Available from `http://www.color.org`.

[11] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 9899:1999, Programming languages—C*.

[12] International Organization for Standardization and International Electrotechnical Commission, *ISO/IEC 9945-1:1990, Information technology—Portable operating system interface (POSIX)—Part 1: System application program interface (API) [C language]*.

[13] Independent JPEG Group, "JPEG library (version 6b)," Available online from `http://www.ijg.org`, 2000.

[14] M. Segal and K. Akeley, *The OpenGL Graphics System: A Specification (Version 1.3)*, Silicon Graphics, Inc., 2001, Available online from `http://www.opengl.org`.

[15] M. J. Kilgard, *The OpenGL Utility Toolkit (GLUT) Programming Interface, API Version 3*, Silicon Graphics, Inc., 1996, Available online from `http://www.opengl.org`.

[16] M. D. Adams, "The JPEG-2000 still image compression standard," ISO/IEC JTC 1/SC 29/WG 1 N 2412, Dec. 2002, Available online from `http://www.ece.uvic.ca/~mdadams`.

[17] S. Chatterjee and C. D. Brooks, "Cache-efficient wavelet lifting in JPEG 2000," in *Proc. of IEEE International Conference on Multimedia and Expo*, Lausanne, Switzerland, Aug. 2002, vol. 1, pp. 797–800.

[18] "K desktop environment home," `http://www.kde.org`, 2003.

[19] "Checkmark benchmarking," `http://watermarking.unige.ch/Checkmark`, 2003.

[20] "GDAL — geospatial data abstraction library," `http://www.remotesensing.org/gdal`, 2003.