JHelioviewer: Visualizing Large Sets of Solar Images Using JPEG 2000

All disciplines that work with image data—from astrophysics to medical research and historic preservation—increasingly require efficient ways to browse and inspect large sets of high-resolution images. Based on the JPEG 2000 image-compression standard, the JHelioviewer solar image visualization tool lets users browse petabyte-scale image archives as well as locate and manipulate specific data sets.

The Sun exhibits phenomena on all observable time and length scales, from seconds to tens of years, and from tens to hundreds of millions of kilometers. Over the last decade, the amount of data returned from space and ground-based solar telescopes has increased by several orders of magnitude. Space missions and ground-based observatories have been taking advantage of better optics, higher network capacities, and greater storage capabilities to produce and deliver an ever-growing volume of solar data.

Today, the Solar and Heliospheric Observatory (SOHO; http://soho.nascom.nasa.gov), launched in 1995, transmits approximately 200 Mbytes of imagery per day. Its lineal descendant, the Solar Dynamics Observatory (SDO; http://sdo.gsfc.nasa.gov), to be launched at the end of 2009, will send 1.4 Tbytes of images per day. Among other data products, SDO will provide full-disk images of the Sun taken every 10 seconds in eight different ultraviolet spectral bands with a resolution of 16 megapixels (MPs) per image. This translates to a 4,096 × 4,096 pixel resolution—that is, a single full image that no monitor or LCD on the market today is large enough to display. These data volumes make downloading and locally browsing and analyzing significant fractions of the data impossible, simply because such activity exceeds the existing Internet and network infrastructure.

With such staggering volume, the data is bound to be accessible from only a few repositories, and users will have to deal with data sets effectively immobile and practically difficult to download. From a scientist's perspective, this poses three problems: accessing, browsing, and finding interesting data while avoiding the proverbial search for a needle in a haystack.

1521-9615/09/$26.00 © 2009 IEEE
Copublished by the IEEE CS and the AIP

Daniel Müller and Bernhard Fleck
European Space Agency
George Dimitoglou, Benjamin W. Caplins, and Desmond E. Amadigwe
Hood College
Juan Pablo García Ortiz
University of Almería
Benjamin Wamsler
University of Applied Sciences, Ulm
Alen Alexanderian
University of Maryland, Baltimore County
V. Keith Hughitt and Jack Ireland
ADNET Systems

The current number of data-browsing tools is limited, and each offers only a specific functionality. For example, SOHO's Web-accessible Near Real-Time Image Browser is serving over 12 years of heliophysics data as JPEG images in two sizes: 1,024 × 1,024 and 512 × 512 pixels (see http://sohodata.nascom.nasa.gov/cgi-bin/data_query). The SOHO Movie Theater uses the same data to create on-screen animations with basic movie control functionality (see http://sohodata.nascom.nasa.gov/cgi-bin/soho_movie_theater). The Solar Weather Browser allows quick image browsing of highly compressed data and can handle up to two overlays.[1] All these widely used tools work well with current data volumes and meet many of the current browsing needs, but they will be severely challenged in the immediate future.

To address this problem, we developed the JHelioviewer (www.jhelioviewer.org) visualization software, which lets users browse large data volumes both as still images and movies. We did so by deploying an efficient image encoding, storage, and dissemination solution using the JPEG 2000 standard. This solution enables users to access remote images at different resolution levels, without requiring the storage of multiple image resolution files. Users can view, manipulate, pan, zoom, and overlay JPEG 2000 compressed data quickly, without severe network bandwidth penalties. Besides viewing data, the browser provides third-party metadata and event catalog integration to quickly locate data of interest.

Our goal is to help scientists discover new phenomena and link related data sets from various instruments that are often analyzed in isolation. In addition, we will make a huge amount of information available to the general public by visualizing it in intuitive and appealing ways. While the impetus for this article is handling solar physics data, similar requirements for accessing, browsing, and searching image data exist for applications in other areas, such as the earth sciences and medical research.[2–4]

Tiles, Pyramids, and Transforms

A popular technique to handle image rendering and visualization of large data sets over networks is *image tiling*. For this method, each original image is divided into subimages, or tiles, at various resolution (zoom) levels, thus creating a pyramid of image tiles for each image (Figure 1). Used by many geospatial-data providers (such as Google and MapQuest), this approach has the advantage of transferring only data for the chosen region of interest (ROI) and zoom level as image tiles from the server to the client. Although this method works well for data repositories with only a few files, it has several distinct disadvantages when the number of files increases.

Resolution level	Number of files
5	1
4	4
3	16
2	64
1	256
Total:	341 files

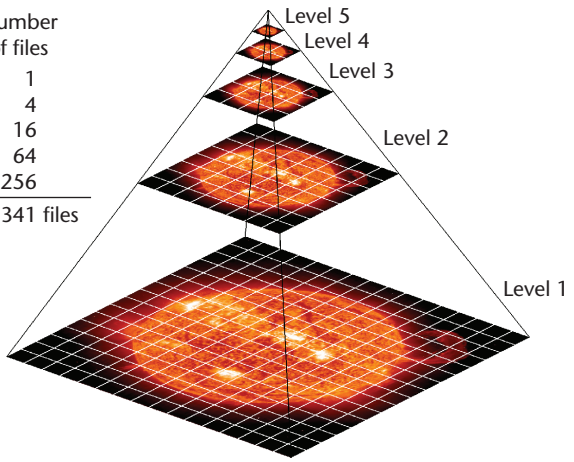


Figure 1. Image tile pyramid with five levels. In this example, a 16-megapixel image is tiled into 341 subimages that are 256 × 256 pixels each.

First, the number of tiles increases as a power of the number of resolution levels. It is given by the finite geometric series

$$\sum_{k=0}^{n-1} z^k = (1 - z^n)/(1 - z),$$

where z is the number of tiles each subimage is divided into to create the next resolution level and n is the number of resolution levels. Even for a modest 16-MP image divided into 256 × 256 pixel tiles at five resolution levels, the tiling approach increases the number of files to be stored by a factor of 341 and the total data volume by at least a factor of two. The total file-size overhead depends on the compression scheme used and the image content. SDO's Atmospheric Imaging Assembly (AIA; http://aia.lmsal.com/) instrument will take 16-MP images of the Sun in eight spectral channels at least every 10 seconds—on the order of 70,000 images per day or 30 million files per year. For data sets of this magnitude, the number of tiles is staggering. Even for a modest fraction of the data, generating tiles becomes prohibitive.

Second, typical use cases for image browsing involve repeated zooming in and out of different ROIs. For each zoom level, a new set of image tiles must be transferred from the server to the client. This method uses significantly more network bandwidth than necessary because it fails to exploit the fact that part of the information

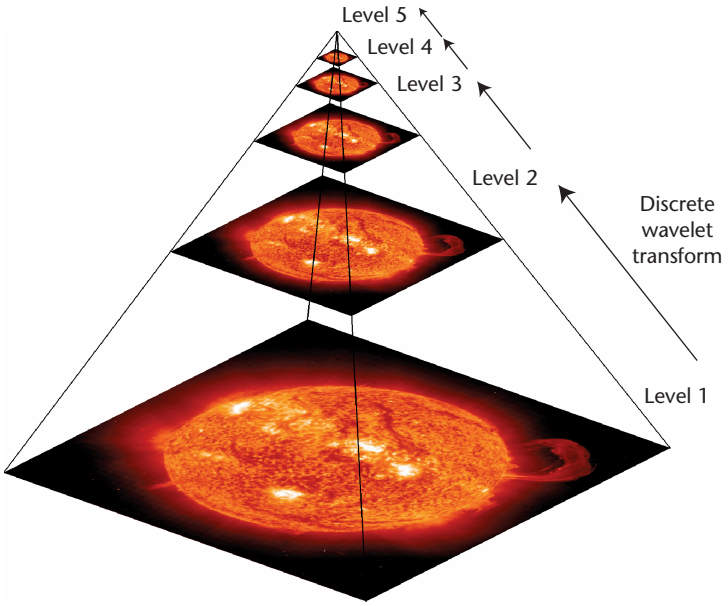


Figure 2. JPEG 2000 pyramid of image representations. Starting from the original image, each resolution level is constructed by applying a discrete wavelet transform to the level below.

contained in the image has already been transferred at a different zoom level.

Clearly, image tiling is not a sufficient solution. A *discrete wavelet transform* (DWT), on the other hand, eliminates the need for tiling and forms the basis of the JPEG 2000 image-compression scheme.

Principles of JPEG 2000

The ISO JPEG 2000 standard[5] was created by the Joint Photographic Experts Group (JPEG) with the intention of improving on, and superseding, the successful JPEG standard[6] that has been in use for almost 20 years. JPEG 2000 is a novel image-compression standard that offers both a lossless and a lossy compression mode and provides many new features, making it a promising format for handling massive amounts of image data and associated metadata. Images need only be encoded once in the highest desired quality and can subsequently be decoded in many ways to extract subfield images with a chosen spatial resolution, level of detail, and ROI. This offers significant advantages compared to storing multiple versions of images or tiles for different resolution levels and drastically reduces the size and complexity of storage and network transmission requirements. Figure 2 shows the JPEG 2000 equivalent of the image tile pyramid in Figure 1. Level 1 represents the image encoded at the full resolution, and DWTs are iteratively applied to each level, creating a hierarchy of image representations at descending resolution. Unlike the tile-based pyramid, these representations are stored in a single JPEG 2000 compressed file.

Table 1 compares the sizes and number of files required for an image tile pyramid composed of losslessly compressed Portable Network Graphics (PNG) subimages (8 bit, single channel) to the size of an equivalent losslessly encoded JPEG 2000 file. JPEG 2000's compression efficiency advantage increases significantly when compressing lossily. However, because a comparison between different lossy compression algorithms requires the adoption of a specific, nonunique quality metric, Table 1 gives a comparison of losslessly compressed data.

How JPEG 2000 Works

The traditional JPEG standard is based on the discrete cosine transform. By comparison, JPEG 2000 uses DWT and therefore offers resolution scalability—that is, image representations at different resolution levels are automatically created during the encoding process. It also offers

Table 1. Comparison of Portable Network Graphics compressed subimages (8 bit, single channel) to an equivalent, losslessly encoded JPEG 2000 file.

			Image tile pyramid		JPEG 2000	
Levels	Zoom (%)	Image size (pixels)	Number of files	Mosaic size (kilobyte)	Number of files	Image size (kilobyte)
5	6.25	256^2	1	112		
4	12.5	512^2	4	452		
3	25	$1{,}024^2$	16	1,844	1	13,324
2	50	$2{,}048^2$	64	7,180		
1	100	$4{,}096^2$	256	16,220		
Total size			341	25,808	1	13,324

progressive refinement, or quality scalability. The image consists of a hierarchy of quality layers that can be selectively decoded to provide the desired level of detail. JPEG 2000 improves functionality through its spatial random access: the standard specifies the JPEG 2000 interactive protocol (JPIP), which can be used to access large images remotely without having to download the entire file to the client. In addition, JPEG 2000 files can contain multiple frames and an arbitrary number of image components. For example, this feature can be used to store microscopy scans with different focus positions, multispectral astrophysical data, or time series of solar images. The JPEG 2000 standard also specifies a movie format, Motion JPEG 2000, that relies on intra-frame encoding rather than the MPEG compression scheme's inter-frame, or temporal encoding. It is therefore ideally suited for applications requiring high-quality individual frames, such as time-dependent solar physics data.

JPEG 2000's DWT is dyadic and can be performed with either the reversible Le Gall 5/3 taps filter[7] for lossless encoding or the irreversible Daubechies 9/7 taps biorthogonal filter,[8] which provides a higher, yet lossy compression ratio. Figure 3 illustrates the 2D wavelet decomposition of a composite image of the Sun and the solar corona (SOHO Extreme ultraviolet Imaging Telescope [EIT] and Large Angle Spectrometric Coronagraph [LASCO] images) using the Daubechies 9/7 filter.

For lossless compression, JPEG 2000 performs on average better than the JPEG standard's lossless mode (L-JPEG) and almost as well as the lossless JPEG-LS scheme, while offering greatly enhanced functionality. For lossy compression, it performs better than the traditional JPEG algorithm, especially at low bit rates.[9] (See related research for a general overview of the JPEG 2000 standard.[10])

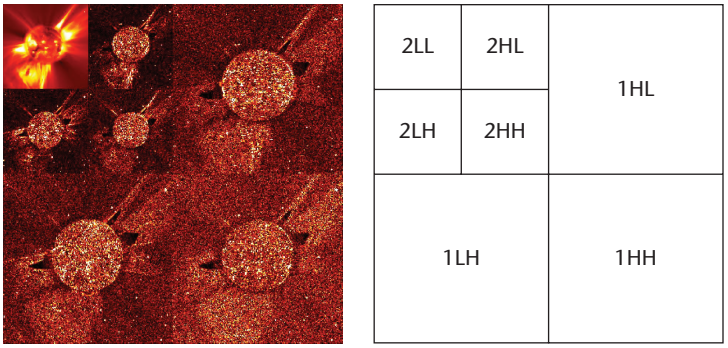


Figure 3. Wavelet decomposition using the Daubechies 9/7 taps filter. (a) A 2D two-level wavelet decomposition of a composite Solar and Heliospheric Observatory (SOHO) image. (b) The application of high-pass (H) and low-pass (L) filters in the horizontal and vertical directions, as indicated by the first and second index, respectively. For example, 1HH indicates high-pass filtering of the first level in both horizontal and vertical directions, while 2LH denotes low-pass filtering of the second level in the horizontal direction and high-pass filtering in the vertical direction.

Remote Image Access With JPIP

JPIP is a data-streaming protocol that enables the remote access of images from a server to a client. It allows full or selected ROI access of JPEG 2000 images[11] and supports both stateful and stateless operation. It also provides sophisticated data-caching capabilities that eliminate redundant data transmission. With JPIP, the client does not access the compressed file stored on the server directly. Instead, the client sends requests that identify the client's current focus window, spatial ROI, resolution, and quality level. This lets the server determine the most appropriate response and return an optimal sequence of selected parts.

All these JPIP characteristics make the protocol especially useful for browsing large remote data sets. For example, a user could interactively browse a day's worth of ultraviolet images of the Sun (Figure 4) from the upcoming NASA SDO's Atmospheric Imaging Assembly at medium spatial resolution (1,000 × 1,000 pixel) and temporal resolution (5 minutes) in three spectral channels, identify an interesting feature, and then browse this ROI at full spatial and temporal resolution. With a compression rate of 0.5 bits/pixel—which offers good visual quality for a quick-look data product—the total data volume transferred for browsing this data set is 66 Mbytes. This is a small fraction of the uncompressed data volume of more than 600 Gbytes for full spatial and temporal resolution at 4,096 × 4,096 pixel image size, 10-second cadence, and 12 bits/pixel image depth.

JHelioviewer Architecture

To leverage the benefits of the JPEG 2000 image format and JPIP, we developed JHelioviewer, a solar image visualization tool that provides image and solar-event browsing capabilities. Users can access local or remote image data streams and play animations. It also offers the option of applying basic image-processing filters such as color tables, γ-correction, and image sharpening. Animations can be created easily on demand both on the client and server side.

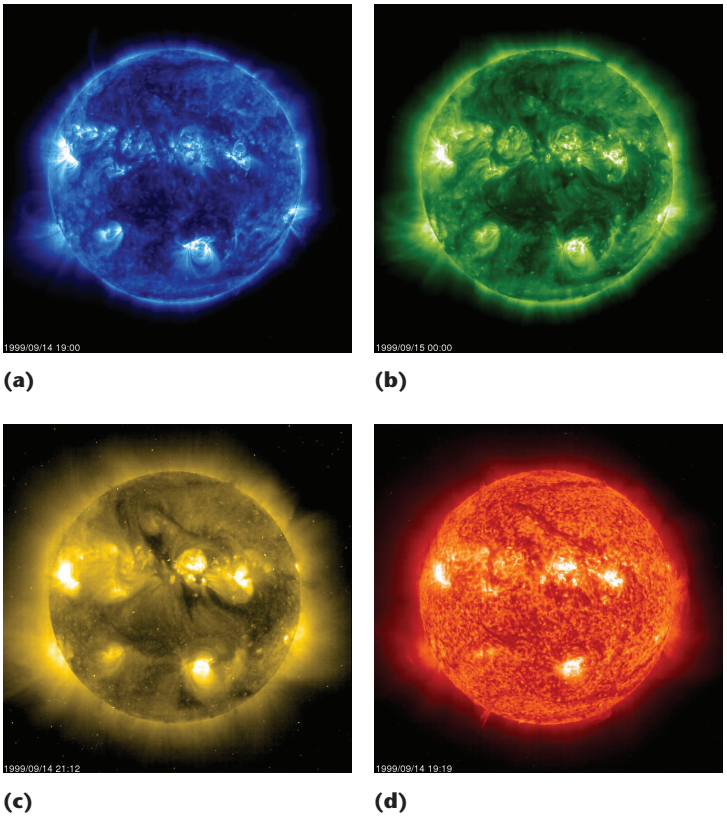


Figure 4. Images of the Sun taken by the Solar and Heliospheric Observatory (SOHO) in four different ultraviolet passbands. The wavelengths are (a) 17.1, (b) 19.5, (c) 28.4, and (d) 30.4 nanometers. The upcoming Solar Dynamics Observatory (SDO) mission will provide similar images, but in more spectral channels and with higher temporal and spatial resolution.

As Figure 5 shows, JHelioviewer makes local and remote JPEG 2000 data sets easily accessible via a cross-platform Java Web Start client application (http://java.sun.com/javase/technologies/desktop/javawebstart/index.jsp) at www.jhelioviewer.org (Figure 5).

Figure 6 shows a diagram of JHelioviewer's client-server architecture. JHelioviewer is based on a client-server architecture. The communication between the client-side browser and the JPIP server is based on request and response messaging using JPIP on top of HTTP. The target JPEG 2000 images are stored in an image repository, while metadata extracted from header information is ingested in a metadata repository, so users can search the metadata to locate data of interest.

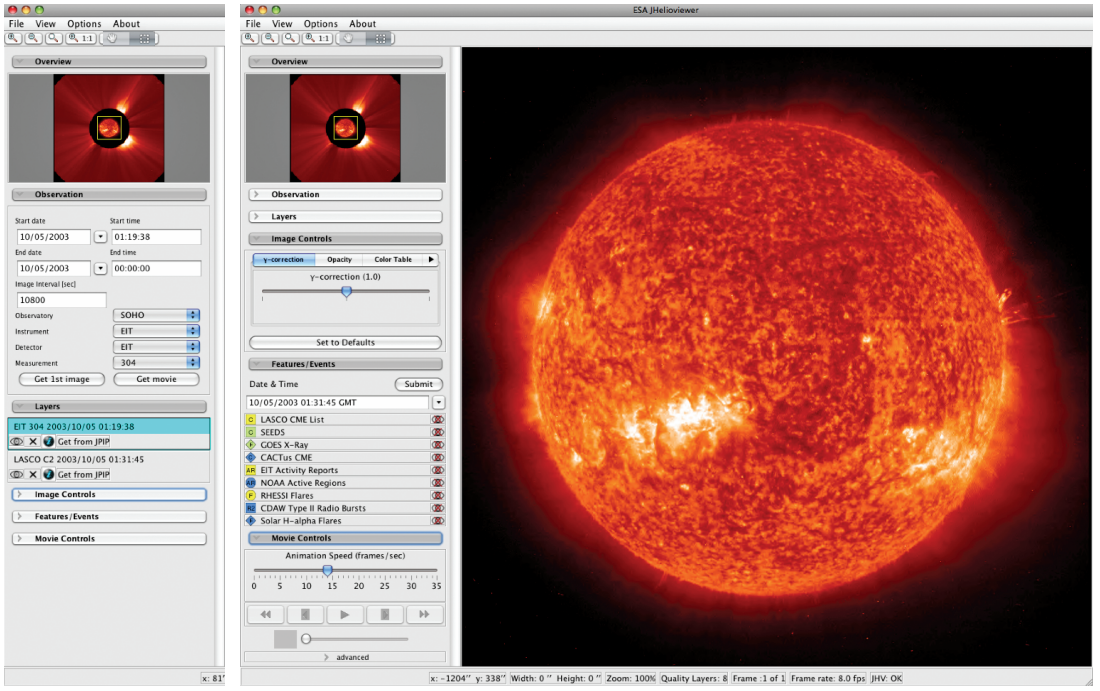
Each part of the JHelioviewer architecture includes several components. The browser includes a user interface, an image renderer, a layer and event manager, a metadata handler, and a local image cache. The server contains a JPEG 2000 parser/reformatter and a JPEG 2000 image repository. A metadata repository may be integrated in the server or operated independently. This repository contains observation descriptions of the data stored in the JPEG 2000 image repository along with location information and access methods.

From the browser side, the components' functionality is as follows:

- The *user interface* displays images and animations and provides search, retrieve, pan, zoom, and overlay capabilities. It also lets users create animations and apply basic image-processing techniques such as changing color tables, image sharpening, and applying a γ-correction filter. Most importantly, it allows the user to search and overlay images, animations, and data markers from event catalogs that identify solar events.
- The *image renderer* receives a JPIP stream along with image headers and renders the user's request and ROI at the requested resolution of any subset of the original compressed data. Image metadata is stored in an XML box inside the JPEG 2000 file. This makes the data self-contained and permits consistent scaling when overlaying images from different telescopes with different image scales. As users manipulate the image, the image renderer queries the local cache to improve image quality and avoid constant data retransmission from the server.
- The *local image cache* contains a cache of previously transmitted data from the server.
- The *layer manager* provides image and animation overlay functionality of multiple JPEG 2000 data streams. It lets users dynamically add and remove data layers or superimpose display markers of solar-event metadata such as the location of an observed sunspot or solar flare. This is a critical functionality that connects image data to metadata event catalogs, letting users search for data based on solar events, not just observation times.
- The *solar-event manager* component interfaces internally with the layer manager and externally with event catalog repositories accessible as Web services.

The server-side component functionality includes the following:

- The *parser/reformatter* receives an image request with parameters (such as the window size, location offset, quality, or resolution), queries the image repository, and returns the result to the client in the form of a JPIP data stream.



(a) **(b)**

Figure 5. Screenshot of the JHelioviewer application. The left part of the application window has several expandable sections, which are shown in two panels for clarity. (a) The overview, database search interface, and layer manager. (b) The image controls, list of solar-event catalogs, and movie controls.

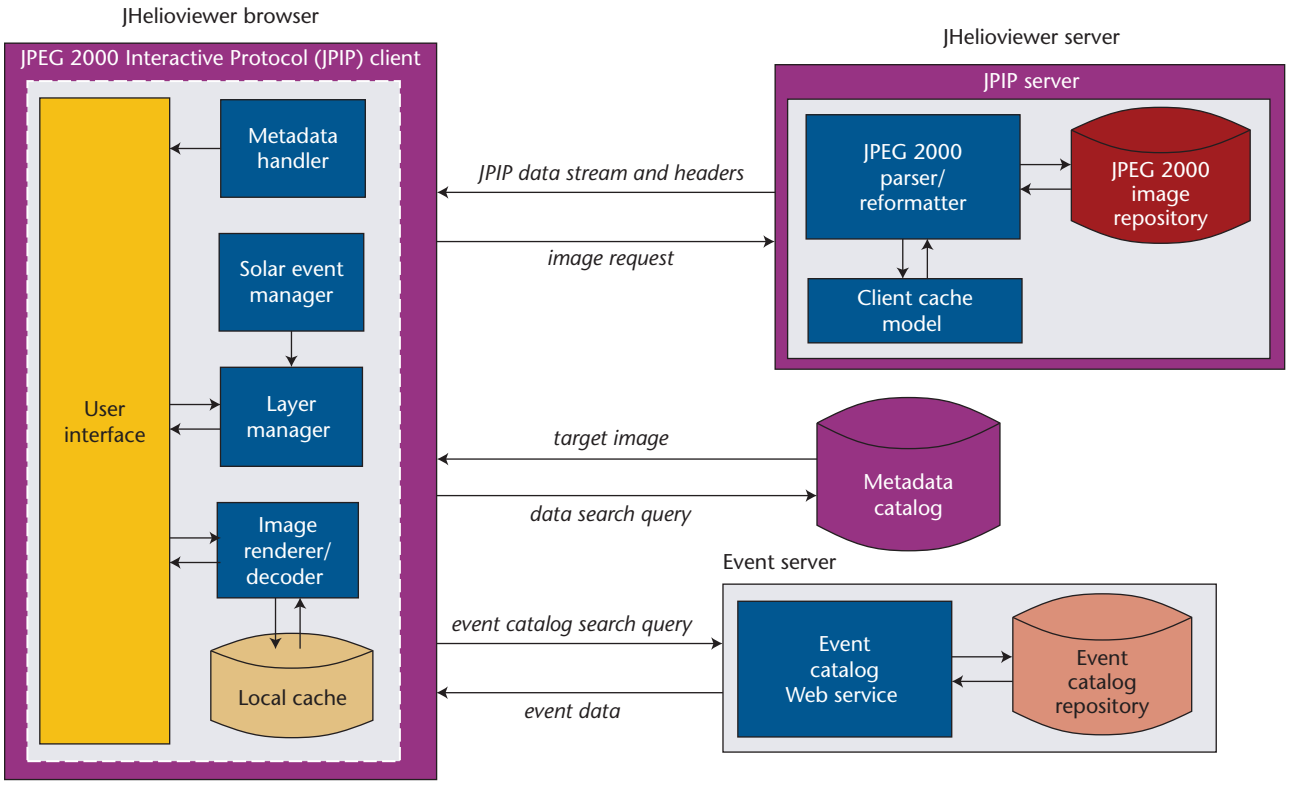


Figure 6. The JHelioviewer architecture. The architecture includes three basic parts—the browser (client), server, and solar-event server—that each include several key components.

- The *image repository* is a file system that stores the JPEG 2000 images using a hierarchical directory structure. Each image's full directory path information is stored in the metadata catalog and is associated with image metadata such as the observation date, instrument, detector, and observatory.
- The *client cache model* contains information about the client's local cache. The parser/reformatter uses this information to avoid retransmitting data that is already stored in the client's cache.

The metadata catalog is stored in a relational database and is accessible as a Web service. We chose to physically decouple the catalog from the JHelioviewer server and host it on a different system to avoid throughput and processing contention among the image data stream rendering, transmissions, and catalog searches. Our choice was merely precautionary; we have no empirical evidence that would indicate server performance degradation if we hosted the metadata catalog together with the JPIP server.

JHelioviewer Features

The current method of observational solar physics research involves downloading large portions of data and then using data-analysis techniques to measure and correlate physical features and events. This method will become effectively obsolete when the data volume is so large that downloading any meaningful portion of data will be infeasible due to network bandwidth and storage constraints.

We designed JHelioviewer to allow solar physicists to quickly browse large volumes of data. Our approach lets them refine and limit the data volume by focusing only on date ranges and ROIs. Consequently, downloading and locally analyzing smaller but highly targeted data sets using existing tools is still feasible. We accomplished this with the following features:

- *Data search and browsing.* JHelioviewer users can search catalogs from multiple distributed data repositories. Search criteria include observation time, observation type, events, and a multitude of other options. Search results are browsable as single images or image series (movies) and include any available metadata and file-header information.
- *Layering and image operations*. Users can overlay an unlimited number of images or movies and adjust layer transparency levels to extenuate features and simultaneously compare those to other data sets. We complement layering with basic image manipulation operations such as image sharpening, applying γ-correction filters, and changing color tables and image opacity.
- *Event catalog integration*. Augmenting the data search functionality, JHelioviewer lets users execute event metadata catalog searches and display solar events as markers on observed data.

The integration feature is a particularly powerful mechanism because it allows highly focused, solar-event-driven searches based on already observed, identified, and catalogued events. The sources of these events can be existing catalogs—such as the US National Oceanic and Atmospheric Administration's active regions and Geostationary Operational Environmental Satellites solar x-ray flux catalogs—and metadata repositories populated with the output of automated feature detection algorithms on raw data.

Current Use and Practical Experience

To test JHelioviewer's usability and performance, we created a database containing a full year of images from multiple imaging telescopes onboard SOHO. This data set is suited to test JHelioviewer's performance because it contains images with different spatial scales, temporal resolution, and wavelengths. Metadata is stored in the JPEG 2000 file headers that allows the overlay and nesting of images with the correct scale and positioning (Figure 7). Movies with arbitrary time cadence are created on demand on the server. Movies with different time cadence can be played simultaneously, and users can even apply image processing functions (such as sharpening and color table changes) while a movie is playing. In addition, data markers from multiple event catalogs can be added to identify solar events such as flares or coronal mass ejections. This constitutes major progress over previous tools because it fully accounts for the data's time-dependent and multiscale nature.

Without any code optimization, we have already displayed movies of the Sun and its extended atmosphere, the corona, at a frame rate of more than 25 frames per second (fps) for a 512 $\times$ 512 pixel ROI of a movie with 4,096 $\times$ 4,096 pixels frames, and more than 15 fps for a ROI of 1,024 $\times$ 1,024 pixels while applying a color table in real time. The individual movie frames are 8-bit grayscale images compressed at 0.5 bits/pixel. By utilizing the computer's graphics processing unit (GPU) using, for example, OpenGL

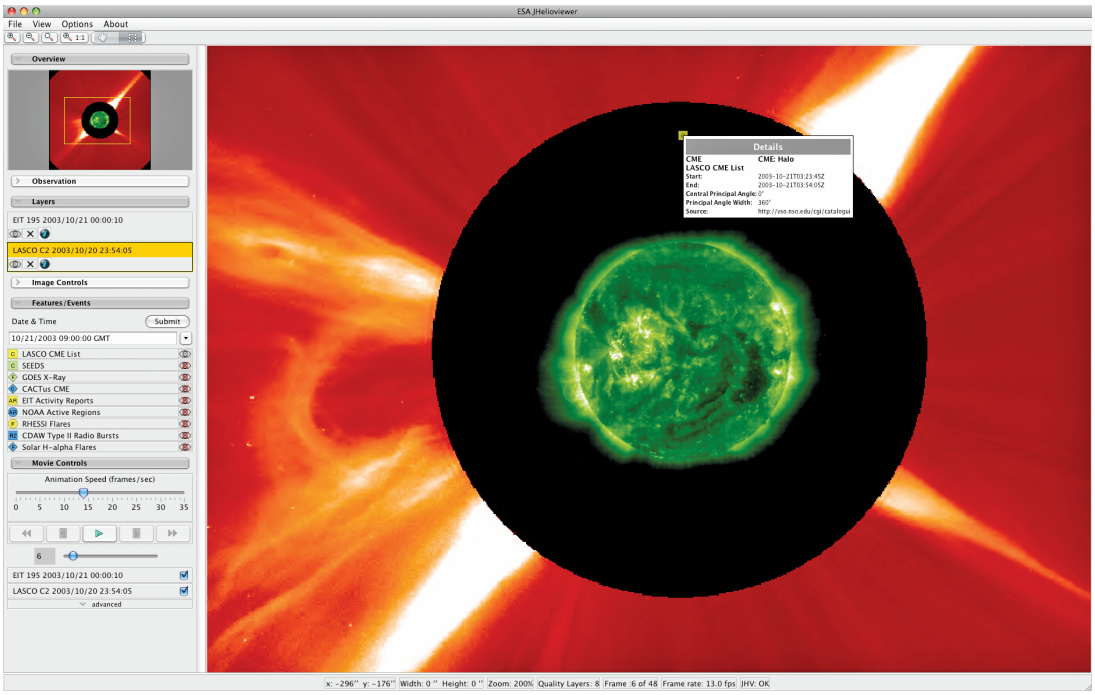


Figure 7. Screenshot of the JHelioviewer application. The center of the left panel shows a list of solar-event catalogs. Events from these catalogs can be overlaid as markers in the main panel on the right, which shows an overlay of an UV image of the Sun (SOHO EIT 19.5 nm, displayed in green) and the surrounding corona (SOHO LASCO C2, displayed in red). Clicking on an event marker displays metainformation about an event (white rectangle). The two images in the main panel are frames of two independent movies with different spatial scales and time cadence. JHelioviewer lets users play these movies simultaneously as well as pan and zoom the composite movie while it is playing.

(www.opengl.org), we hope to further increase the movie display performance.

JPEG 2000 is a relatively new standard, and like other recently popular technologies (such as MP3[12,13]), it is not immune to patent ownership claims. However, the JPEG 2000 standard's core (Part 1) was developed to be implemented and distributed without license fee or royalty obligation. Although numerous patent holders have already waived any claim rights, it is impossible to know if other nonwaived patent claims will surface. At the same time, other efforts such as OpenJPEG2000 (www.openjpeg.org) and Jasper[14] are being developed under permissive free software licenses and can provide a viable open source alternative.

Although JPEG-2000-based applications exist in various domains—ranging from medical imaging[3,4] to cultural heritage preservation[15,16]—the standard is not yet widely known and used. The traditional JPEG format is particularly successful and widely used in Internet domains and Web applications. As a consequence, not all Web browsers support some of the JPEG 2000 features. Until the standard is fully integrated into browsers, we will continue exploring alternative approaches that would let Web-based applications utilize the standard's added functionality.

A companion of the JHelioviewer project is the AJAX-based Web application Helioviewer (www.helioviewer.org) that employs a JPEG-2000-based image server to perform dynamic image tiling, which can mitigate the problem of having millions of prefabricated tiles to store and access. The server extracts and serves Web-compatible images in JPEG and PNG format to clients on demand. This solution offers full Web browser compatibility, but it lacks some of the functionality possible with JPEG 2000, such as quality scalability and minimization of data transfer. This approach is inspired by the open source project djatoka (http://african.lanl.gov/aDORe/projects/djatoka/), which also offers an open source image server based on the JPEG 2000 format. The combination of a JPEG 2000 image source and dynamic tiling (with caching) means that both

native JPEG 2000 viewers and standard Web browsers can share a single image database, which improves homogeneity and reduces the service's complexity.

A second approach is a Web plug-in, which modern browsers already support. This alternative does not require a format conversion from JPEG 2000 to a browser-compatible image format. For several Web browsers, including Mozilla Firefox, JPEG 2000 plug-ins already exist. We are currently working on a JPIP plug-in for Firefox that will make the full JPEG 2000 feature set available to Web applications. Together with JHelioviewer, this will offer a comprehensive set of interfaces to access JPEG 2000 data.

Our work has impact in two areas. First, it provides the solar physics community with a viable alternative to the current, nonscalable way of storing, accessing, and analyzing remote data. Second, we hope to enhance JPEG 2000's visibility as a realistic image-implementation standard given the inherent benefits that it brings to scientific applications. Although our implementation is focused on accessing solar physics data, our architecture and components can be easily reused in other domains with similar large data volume constraints and browsing requirements.

Our JPEG 2000 implementation for JHelioviewer is based on the Kakadu Software Development Kit (www.kakadusoftware.com), which can be inexpensively licensed for noncommercial applications. The JHelioviewer source code is available free of charge at https://code.launchpad.net/helioviewer. With the exception of Kakadu Software's JPEG 2000 libraries and Java classes, the entire JHelioviewer source code is published under the GNU Affero GPL (www.fsf.org/licensing/licenses/agpl-3.0.html).

Acknowledgments

We are grateful for the European Space Agency's Research and Scientific Support Department's financial support. J. Ireland acknowledges support from NASA Virtual Observatories for Heliophysics Data program (NASA Research Announcement NNH072DA001N [ROSES 2007] 07-VX00-0016).

References

1. B. Nicula, C. Marqué, and D. Berghmans, "Visualization of Distributed Solar Data and Metadata with the Solar Weather Browser," *Solar Physics*, vol. 248, no. 2, 2008, pp. 225–232.
2. P. Kulkarni et al., "Compression of Earth Science Data with JPEG 2000," *Hyperspectral Data Compression*, Springer US, 2006, pp. 347–378.
3. R. Zwönitzer et al., "Digital Pathology: DICOM-Conform Draft, Testbed, and First Results," *Computer Methods and Programs in Biomedicine*, vol. 87, no. 3, 2007, pp. 181–188.
4. V. Tuominen and J. Isola, "The Application of JPEG 2000 in Virtual Microscopy," *J. Digital Imaging*, vol. 22, no. 3, 2007, pp. 250–258.
5. *ISO/IEC 15444-1/2: Information Technology, JPEG 2000 Image Coding System, Core Coding System and Extensions*, ISO, 2000.
6. *ISO/IEC 10918-1: Information Technology, Digital Compression and Coding of Continuous-Tone Still Images, Requirements and Guidelines*, ISO, 1992.
7. D. Le Gall and A. Tabatai, "Subband Coding of Digital Images Using Symmetric Short Kernel Filters and Arithmetic Coding Techniques," *Proc. IEEE Int'l Conf. Acoustic, Speech, and Signal Processing*, IEEE Press, 1988, pp. 761–765.
8. M. Antonini et al., "Image Coding Using Wavelet Transform," *IEEE Trans. Image Processing*, vol. 1, no. 2, 1992, pp. 205–220.
9. D. Santa-Cruz, R. Grosbois, and T. Ebrahimi, "JPEG 2000 Performance Evaluation and Assessment," *Signal Processing: Image Communication*, vol. 1, no. 17, 2002, pp. 113–130.
10. M. Rabbani and R. Joshi, "An Overview of the JPEG 2000 Still Image Compression Standard," *Signal Processing: Image Communication*, vol. 1, no. 17, 2002, pp. 3–48.
11. D.S. Taubman and R. Prandolini, "Architecture, Philosophy, and Performance of JPIP: Internet Protocol Standard for JPEG 2000," *Proc. Visual Communications and Image Processing* (VCIP), SPIE, 2003, pp. 791–805.
12. *Lucent Technologies, Lucent Technologies Guardian, and Multimedia Patent Trust v. Gateway, Dell, and Microsoft Corporation*, Court of Appeals for the Federal Circuit Decision (2007-1546-1580); www.cafc.uscourts.gov/opinions/07-1546.pdf.
13. M. Williams, "Apple, Samsung, Sandisk Sued over MP3," Infoworld.com, 26 Feb. 2007; www.infoworld.com/d/security-central/apple-samsung-sandisk-sued-over-mp3-686.
14. M.D. Adams and R.K. Ward, "JasPer: A Portable Flexible Open-Source Software Tool Kit for Image Coding/Processing," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing* (ICASSP 04), IEEE Press, vol. 5, 2004, pp. 241–244.
15. V. Misic et al., "MRC for Compression of Blake Archive Images," *Proc. Applications of Digital Image Processing XXV*, vol. 4790, SPIE, 2002, pp. 479–490.
16. E.A. Politou, G.P. Pavlidis, and C. Chamzas, "JPEG 2000 and Dissemination of Cultural Heritage over the Internet," *IEEE Trans. Image Processing*, vol. 13, no. 3, 2004, pp. 293–301.

Daniel Müller *is a scientist at the European Space Agency. His research interests include the dynamics of the solar atmosphere, high-performance computing, and visualization techniques for complex data sets. Müller has a PhD in physics from the University of Freiburg, Germany. Contact him at daniel.mueller@esa.int.*

George Dimitoglou *is an assistant professor of computer science at Hood College. His research interests include the design and analysis of algorithms and distributed computing. Dimitoglou has a PhD in computer science from the School of Engineering and Applied Science of the George Washington University. Contact him at dimitoglou@hood.edu.*

Benjamin W. Caplins *is a graduate student in physical chemistry at the University of California, Berkeley. His research interests include algorithm design, large data set visualization, and ultra-fast spectroscopy. Caplins has a BA in chemistry and mathematics from Hood College. Contact him at caplins@yahoo.com.*

Juan Pablo García Ortiz *is a member of the Computer Architecture and Electronics Department's Supercomputing-Algorithms Research Group at the University of Almería, Spain. He is currently working toward a PhD in remote browsing of JPEG 2000 images.*

Benjamin Wamsler *is a student at the University of Applied Sciences, Ulm, Germany. His current studies focus on mechanical, electronic, and computer engineering. Contact him at wamsler@web-grafix.eu.*

V. Keith Hughitt *is a Web developer with ADNET Systems, working at the NASA Goddard Space Flight Center, Maryland. Hughitt has a BS in bioinformatics from the University of Maryland, Baltimore County. Contact him at vincent.k.hughitt@nasa.gov.*

Alen Alexanderian *is a PhD candidate in applied mathematics at the University of Maryland, Baltimore County. His research interests include homogenization of random media, numerical methods for partial differential equations, mathematical modeling and simulation, and scientific computing. Alexanderian has an MS in applied mathematics from the University of Maryland, Baltimore County. Contact him at aa5@umbc.edu.*

Jack Ireland *is a research scientist with ADNET Systems, working at the NASA Goddard Space Flight Center, Maryland. His research interests include remote visualization of heterogeneous data, automated image analysis in solar physics, and Bayesian statistics. Ireland has a PhD in physics from the University of Glasgow, Scotland. Contact him at jack.ireland@nasa.gov.*

Desmond E. Amadigwe *is a graduate student in computer science at Hood College. His research and professional interests are focused on software engineering.*

Bernhard Fleck *is a scientist at the European Space Agency. His research interests include the dynamics of the solar atmosphere, in particular wave-propagation characteristics in the chromosphere. Fleck has a PhD in physics from the University of Würzburg, Germany. Contact him at bernhard.fleck@esa.int.*