Scalable Computing: Practice and Experience
Volume 7, Number 1, pp. 59–75. http://www.scpe.org
ISSN 1895-1767

YOURSKYG: LARGE-SCALE ASTRONOMICAL IMAGE MOSAICKING ON THE INFORMATION POWER GRID

JOSEPH C. JACOB*, JAMES B. COLLIER , LORING G. CRAYMER , AND DAVID W. CURKENDALL

Abstract. The yourSkyG custom astronomical image mosaicking software has a web portal interface that allows custom access via ordinary desktop computers with low bandwidth network connections to high performance mosaicking software deployed on a computational grid, such as NASA's Information Power Grid (IPG). In this context, custom access refers to on-the-fly mosaicking to meet user-specified criteria for region of the sky to be mosaicked, data sets to be used, resolution, coordinate system, projection, data type and image format. The portal uses pipelines and data caches to construct multiple mosaics on the grid with high throughput.

Key words. Astronomy, virtual observatory, image, mosaic, web, grid, portal

1. Introduction. In recent years the Astronomy community has witnessed rapid growth in the size and complexity of astronomical data sets due to rapid advances in remote sensing technology. The massive data sets that now exist collectively contain tens of terabytes of imagery and catalogs in wavelengths spanning the entire electromagnetic spectrum. Although this rich data store represents a significant opportunity for new scientific discoveries, it also represents a serious challenge to the community: How does one effectively and efficiently extract information from such a large and complex collection of data? The National Virtual Observatory (NVO) [1, 2, 3] is addressing this question in the United States and similar efforts exist elsewhere in the world [4, 5, 6, 7]. Many of the these virtual observatory projects are cooperating to ensure that they remain integrated and interoperable via the International Virtual Observatory Alliance (IVOA) [8].

As a community effort, the virtual observatory necessarily exhibits a loosely coupled, distributed architecture, with an emphasis on interoperability between components developed and deployed by domain experts in various areas. Since many of these components require an enormous amount of computation and data movement, the NVO needs to be deployed in a distributed, high performance, scalable computing environment. However, a significant fraction of astronomical research is conducted by scientists and students with limited resources, ordinary desktop computers and low bandwidth network connections. Therefore, to be effective the NVO also needs to provide portals to its high performance infrastructure that will make it usable by researchers anywhere.

1.1. Grid Computing in Astronomy. The emergence of the virtual observatory concept coincides with the maturation of computational grids as a viable architecture for high-performance or data-intensive, distributed computations. The fundamental computational grid infrastructure includes both hardware-distributed, possibly heterogeneous processors interconnected by networks—and software to launch remote computations and to transport data to the processes that require them. The fundamental software infrastructure, provided by the Globus Toolkit, is what makes a collection of distributed computational resources into a functional computational grid, providing users with a single point of authentication for simultaneous access to all of the grid resources. In the grid development community, research is ongoing to bring into production more sophisticated grid software, layered on top of Globus, to provide additional functionality such as job monitoring, checkpointing, stop and restart, error recovery, planning, and scheduling.

The research described in this paper was conducted in 2002-2003, at which time the NVO was in its early stages and application of grid technology to astronomical research was very limited. At this time, a number of important web-based systems were instantiated, which serve as a model for the grid-based and web-based architectures prevalent today.

A number of projects used grid computing to allow science users around the world to access computational software over the Internet. The main advantage is that deploying these algorithms as grid and web services makes them accessible to science users with limited resources and only lightweight computers and network connections. One example of this is the Hera architecture [9], which makes it possible to run the High Energy Astrophysics Science Archive Research Center (HEASARC) data analysis software at NASA's Goddard Space Flight Center (GSFC) on a remote server via a simple graphical interface. Astrocomp [10] is a web portal

*Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109-8099, Joseph.Jacob@jpl.nasa.gov

for access to software for N-body simulations. Similarly, the yourSky web portal [11], described in this paper, provides remote access to JPL's parallel astronomical image mosaicking software via a simple web form interface.

The eSTAR Project [12] is an intelligent robotic telescope network that connects intelligent agents to telescopes and databases through grid and web services. This is an example of how grid computing is applied to science activities like Gamma-Ray Burst followup observations and the hunt for planets outside our solar system.

The Sloan Digital Sky Survey (SDSS) was an early adopter of web services technologies to facilitate access to large astronomical datasets. SDSS used many of the same technologies that are prevalent today in web-based commerce, including SOAP (Simple Object Access Protocol), XML (Extensible Markup Language), and WSDL (Web Service Description Language), to build services for resource discovery, data mining, visualization, and statistical analysis. The SkyQuery [13, 14] portal was implemented for SDSS using this web services architecture.

A number of projects related to virtual observatories are focused on using grid and web services technology to access multiple datasets (images and catalogs) from different archive centers and merge them to provide richer information content than is available from any of the datasets alone. This type of interoperability concept is demonstrated in the Aladin/GLU system [15], the European Space Agency (ESA) science archives [16], the Smithsonian Astrophysical Observatory (SAO) spectral archives [17], the Astrophysical Virtual Observatory (AVO) interoperability prototype [18], the On-Line Archive Science Information Services (OASIS) [19], and Grist (Grid Services for Astronomy) projects [20]. The Unified Content Descriptor (UCD) [21] prescribes a naming scheme for astronomical catalogs in order to facilitate this type of interoperability.

NASA's Information Power Grid (IPG) provided the grid infrastructure used in this work.[1] The IPG connected SGI Origin servers and Linux clusters distributed at NASA centers nationwide. The National Science Foundation (NSF) also sponsors a computational grid called the TeraGrid, which, at the time the research reported in this paper was conducted, linked together large Linux clusters at five sites, California Institute of Technology, San Diego Supercomputer Center, Argonne National Laboratory, National Center for Supercomputing Applications, and Pittsburgh Supercomputing Center. By September 2004, additional TeraGrid centers were added, including Indiana University, Oak Ridge National Laboratory, Purdue University, and Texas Advanced Computing Center, for an aggregate processing power of 40 teraflops, with 2 petabytes of disk storage, all interconnected with a 10-30 gigabits per second dedicated national network [22].

1.2. Image Mosaicking in Astronomy. The mosaicking described in this paper involves reprojecting input image plates to a common coordinate system, projection, equinox, and epoch, and combining the resulting plates to produce a single output image. There are strong science drivers for mosaicking. The most obvious is that large image mosaics enable analysis of celestial objects that either do not fit on a single image plate in the native image partitioning scheme used by a survey, or fall at the boundary between two or more neighboring plates. Also, mosaics enable analysis of the large-scale structure of the universe. In addition, mosaicking data sets in different wavelengths or from different surveys to the same coordinate grid enables multi-spectral analysis, which could be essential for identifying new, previously unknown, types of objects, or for identifying new objects that are so faint in a single wavelength that they are overlooked until combined with the signals from other wavelengths.

A number of software packages exist that can be used to construct astronomical image mosaics. This paper describes the yourSky software and its usage on the IPG. The yourSky software is the baseline for Montage [23, 24], a general science-grade astronomical image mosaicking toolkit that preserves both astrometry (object positions) and photometry (brightnesses) in the images. The Montage software was deployed as a service on the TeraGrid using a scheduler called Pegasus [25, 26] and Condor DAGMAN [27] to launch the computations on the grid in a manner that preserves all of the dependencies. Other notable astronomical mosaicking projects include SWarp [28] from the French TERAPIX center and MOPEX [29] from the Spitzer Science Center at Caltech.

1.3. From yourSky to yourSkyG. In this paper, we describe yourSky and yourSkyG, portals for high-performance, on-demand, astronomical image mosaicking. Both yourSky and yourSkyG can perform their high performance computations and data movement on conventional supercomputers, but yourSky requires use of a local multiprocessor system, while yourSkyG is capable of launching its computations on remote computers organized in a computational grid such as the IPG. A key characteristic of the portal architecture is that the

[1]NASA decommissioned the IPG in 2004.

data movements required to construct a requested mosaic and the actual location where the computations are carried out are transparent to the user who simply orders his mosaic by specifying the parameters that describe the mosaic. Regardless of where the computations are performed, these portals are accessible via lightweight client software—the ubiquitous web browser. The architecture of yourSkyG is motivated by the loosely-coupled, distributed nature of both the NVO and IPG infrastructures.

The yourSkyG portal is optimized for efficient processing of mosaic *ensembles*, multiple mosaic requests to be processed together. This mode of processing can dramatically improve throughput by (*i*) reducing the amount of data communication in cases where multiple mosaics require the same input image plates, and (*ii*) performing computation and communication for different mosaics in parallel where possible. Furthermore, we introduce the concept of *data reservoirs*, carefully managed data caches maintained at each stage of the data flow pipeline that have the effect of smoothing out variations in throughput as the availability of and load on shared grid resources change over time.

The architecture of yourSky is summarized in Section 2. Enhancements required to produce yourSkyG, running on the IPG, are described in Section 3. Optimizations for processing multiple mosaic requests are described in Section 4. Performance results are provided in Section 5. Finally, a summary is provided in Section 6.

2. The yourSky Portal. The only client software required to use the yourSky custom astronomical image mosaic server is the ubiquitous web browser. By filling out and submitting the request form, users have custom access on their desktop to all of the publicly released data from the member surveys. In this context, "custom access" refers to new technology that enables on-the-fly astronomical image mosaicking to meet user-specified criteria for region of the sky to be mosaicked, data set to be used, resolution, coordinate system, projection, data type, and image format. All mosaic requests are fulfilled from the original archive data so that the domain experts maintain control and responsibility for their data and data corruption due to resampling is minimized because only one reprojection is done from the raw input data. Currently the data archives that are accessible with yourSky are the Digitized Palomar Observatory Sky Survey (DPOSS) [30] and the Two Micron All Sky Survey (2MASS) [31]. DPOSS has captured the entire northern Sky at 1 arc second resolution in three visible wavelengths. 2MASS has captured the entire sky at 1 arc second resolution in three infrared wavelengths. The yourSky architecture supports expansion to include other surveys, without regard to the native image partitioning scheme used by a particular survey.

2.1. Architecture. The architecture for yourSky is illustrated in Fig. 2.1. In the figure, the numbered descriptions on some of the arrows give the steps taken to fulfill a typical mosaic request. The procedure is as follows. The clients at the top left of the illustration are the web browsers that may be used to submit requests to yourSky. A simple HTML form interface, shown in Fig. 2.2, is used to specify the parameters that are to be passed to the custom astronomical image mosaicking software. The mosaicking software and the mosaic parameters are described in detail in Section 2.2. The yourSky Mosaic Request Manager running on the yourSky server checks for mosaic requests and hands them off to the Mosaic Request Handler, using the user priority scheme described in Section 2.3. The Mosaic Request Handler queries the Plate Coverage Database, described in Section 2.4, to determine which input image plates from DPOSS or 2MASS are required to fulfill the mosaic request. A fixed size data cache is maintained on the yourSky server to store the input image plates required to build recent mosaic requests. If all of the required input image plates are already present locally in the data cache, the mosaic is constructed immediately using the custom astronomical image mosaicking software. If some of the required input image plates are not already cached locally, they need to be retrieved from their respective archives. Therefore an "archive request" is issued. The yourSky Archive Request Manager checks for archive requests and hands them off to the Archive Request Handler, which retrieves the required input image plates from the appropriate remote archive. Once all of the input image plates for a request have been cached on a local disk, the custom astronomical image mosaicking software is launched to construct the mosaic. When the mosaic, built precisely to match the user's request parameters, is ready an email is sent back to the user with the URL where the image mosaic can be downloaded.

2.2. Custom Astronomical Image Mosaicking Software. The heart of the yourSky server is the custom astronomical image mosaicking software that is used to construct an image mosaic precisely matching user-specified parameters. The inputs to the mosaicking software are a list of input images to be mosaicked and the custom parameters that determine the properties of the mosaic to be constructed. The only requirements on the input images are the following. First, they must comply with the standard dictated by the Flexible Image

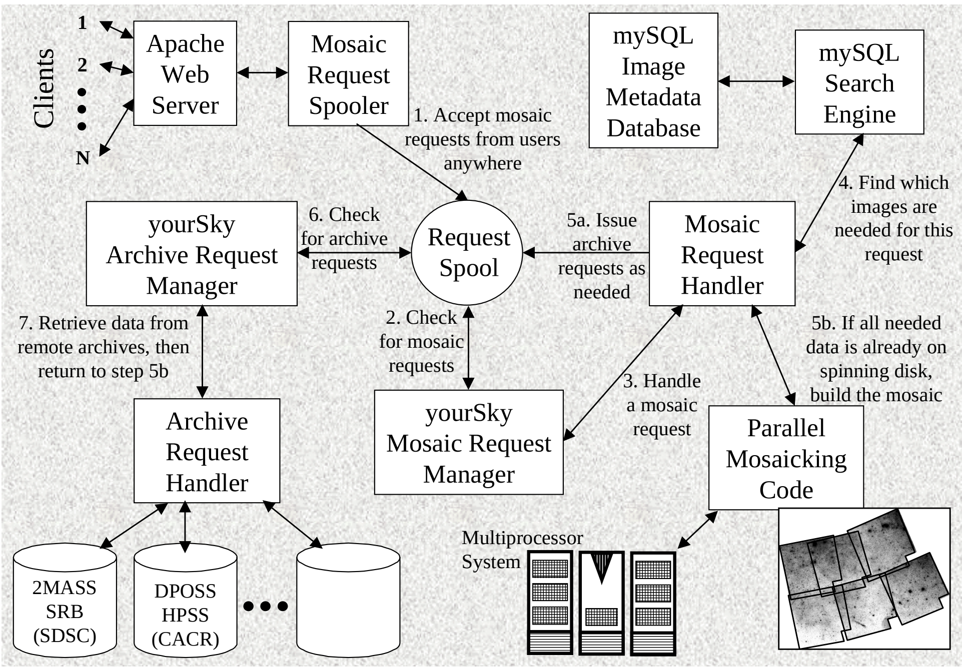


Fig. 2.1. *The architecture of yourSky supports fully automated mosaicking, including retrieval of the input image plates from the remote survey archives.*

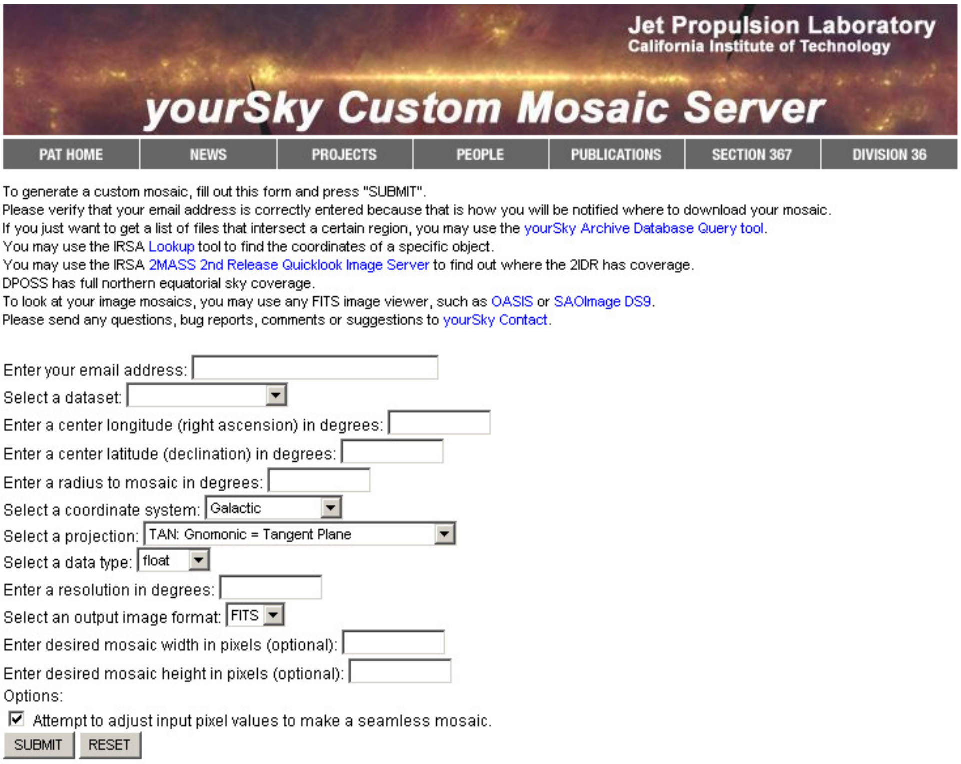


Fig. 2.2. *The yourSky custom mosaic web form interface.*

Transport System (FITS), a data format that is well understood by the astronomy community and has long been used as the de facto method for sharing data within the community [32]. FITS format images encapsulate the image data with keyword-value pairs that give additional information about how the data values in the image map to locations on the sky. The second requirement for input images to the mosaicking software is that the FITS header must contain valid World Coordinate System (WCS) information. The WCS defines pixel-to-sky and sky-to-pixel coordinate transformations for a variety of coordinate systems and projections commonly used by the astronomy community [33].

2.2.1. Custom Access. With yourSky, the emphasis is on custom access to astronomical image mosaics. The following parameters may be used to specify the mosaic to be constructed:

1. Center right ascension and declination: Required parameters, analogous to the CRVAL1 and CRVAL2 FITS keywords, which specify the location on the celestial sphere of the tangent point for the image projection plane. By default, this center of projection is placed at the center pixel in the mosaic, analogous to the CRPIX1 and CRPIX2 FITS keywords.
2. Resolution: Required parameters, analogous to the CDELT1 and CDELT2 FITS keywords, which specify the pixel size in degrees in each of the two image dimensions at the mosaic center of projection.
3. Radius in degrees: Optional parameter that limits the mosaic size using degrees from the mosaic center. If not specified, the radius is determined automatically from the region of coverage of the input image plates.
4. Width and height in pixels: Optional parameters, analogous to the NAXIS1 and NAXIS2 FITS keywords, that limit the mosaic size using a specific number of pixels. These parameters supersede the radius in degrees if that is given as well. If not specified, the size is determined automatically from the region of coverage of the input image plates.
5. Coordinate system: Required parameter, analogous to the first half of the CTYPE1 and CTYPE2 FITS keyword values, that specifies the alignment of the mosaic axes in 3-D space. Four coordinate systems are supported: galactic, ecliptic, J2000 equatorial, and B1950 equatorial.
6. Projection: Required parameter, analogous to the second half of the CTYPE1 and CTYPE2 FITS keyword values, that specifies how locations on the celestial sphere are mapped to the image projection plane. All of the projections specified by WCS are supported: Linear (LIN), Gnomonic (TAN), Orthographic (SIN), Stereographic (STG), Zenithal/Azimuthal Perspective (AZP), Zenithal/Azimuthal Equidistant (ARC), Zenithal/Azimuthal Polynomial (ZPN), Zenithal/Azimuthal Equal Area (ZEA), Airy (AIR), Cylindrical Perspective (CYP), Cartesian (CAR), Mercator (MER), Cylindrical Equal Area (CEA), Conic Perspective (COP), Conic Equidistant (COD), Conic Equal Area (COE), Conic Orthomorphic (COO), Bonne (BON), Polyconic (PCO), Sanson-Flamsteed Sinusoidal (SFL), Parabolic (PAR), Hammer-Aitoff (AIT), Mollweide (MOL), COBE Quadrilateralized Spherical Cube (CSC), Quadrilateralized Spherical Cube (QSC), Tangential Spherical Cube (TSC), Digitized Sky Survey Plate Solution (DSS), and Plate fit polynomials (PLT).
7. Image Format: Required parameter that specifies the output mosaic image format (FITS is recommended). The following image formats are currently supported: FITS, JPEG, PGM, PNG, TIFF, and Raw Data.
8. Data Type: Required parameter that specifies the data type of the mosaic pixels. This is analogous to the BITPIX FITS keyword. The data types currently supported are 8-, 16-, and 32-bit signed and unsigned integer, and single and double precision floating point.
9. Quantization Extrema: Optional parameters that specify the minimum and maximum over which to stretch the input pixel values for those data types that require quantization to a limited number of output bits per pixel (especially, 8-bit and 16-bit integers). The user can specify these values to control how many gray levels in the output mosaic are assigned to low or high intensity regions of the sky.
10. Pixel Masks: Optional masks may be specified to discard pixels around the outer perimeter or from particular rectangular regions in each input image.
11. Background Matching: Logical parameter that specifies whether or not yourSky should attempt to match the background intensities among the input images that comprise a mosaic in an attempt to produce a mosaic that is as seamless as possible.

2.2.2. Parallel Mosaicking Algorithm. The yourSky mosaicking algorithm is designed to be able to handle arbitrarily sized mosaic requests from typical small requests covering a single celestial object to all-sky mosaics at full resolution. Also, the algorithm is efficient in the face of arbitrarily sized input image plates, so that yourSky can be extended to support other image archives without consideration of the native image partitioning scheme used by the archive. For example, the two surveys currently accessible by yourSky have drastically different native image partitioning schemes, from millions of small 2 MB image plates in the case of 2MASS to thousands of much larger 1 GB plates in the case of DPOSS. In addition, the mosaicking algorithm is designed to support arbitrary mappings from input image pixels to output mosaic pixels.

The mosaicking proceeds in two phases, Analysis and Build. During Analysis, the following is accomplished. First, the mosaic width and height are determined if they are not provided explicitly as part of the user-specified parameter set. Second, the pixel coordinates that intersect the mosaic are determined for each input image along with the corresponding intersection coordinates from the mosaic. These coordinates are used to set loop bounds and buffer sizes during the Build phase. Third, in cases where the data type requires quantization to a limited number of output bits per pixel in the output mosaic (e.g., 8-bit and 16-bit integers), the minimum and maximum over which the pixel values should be quantized are determined if these extrema are not specified explicitly as part of the user-specified parameter set. Fourth, if background matching is to be done, the intensity correction for each input image plate is determined.

During the Build phase the information gathered during Analysis is used to construct the custom mosaic. If the mosaic is to be lower resolution than the input image plates, the outer loop is over the input image pixels and the mosaic pixel values are calculated as the average of the input pixels for which the pixel center falls within the mosaic pixel region of coverage. If the mosaic is to be roughly the same or higher resolution than the input image plates, the outer loop is over the mosaic pixels and the mosaic pixel values are computed to be the result of sampling from the input images using bilinear interpolation. In either case, mapping from input pixel coordinates to output pixel coordinates is done by first mapping from input pixels to a location on the sky, then mapping from the sky coordinates to the output pixel coordinates, as illustrated in Fig. 2.3.

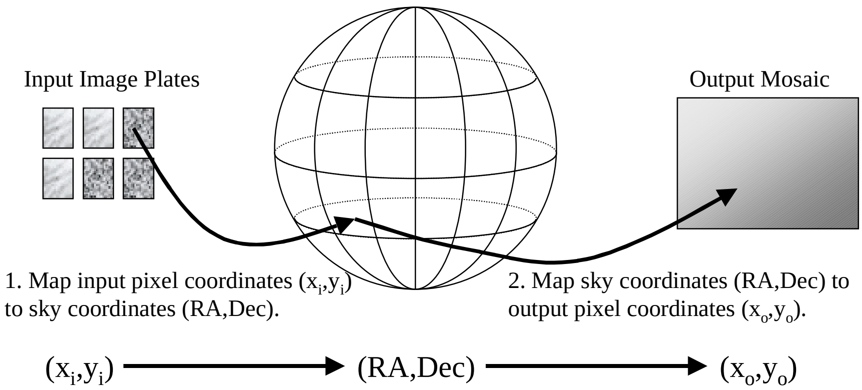


Fig. 2.3. *Mapping from input pixel coordinates to output pixel coordinates is done in two steps. First, the input coordinates are mapped to a position on the sky, then that position on the sky is mapped to the output mosaic coordinates.*

The mosaicking proceeds in parallel during both Analysis and Build, with each processor being assigned a subset of the input image pixels. By default the input images are assigned to processors in a round robin fashion, with one processor per image, but the user can reconfigure this at run-time by specifying the number of processors to be assigned to each input image. Assigning multiple processors to each input image plate dramatically improves efficiency for archives, such as DPOSS, that have such large image plates that only a single or a few input image plates are required for a typical mosaic request. If multiple processors are assigned to each input image, a group synchronization among the processors assigned to the same image is required for each image so that Analysis results can be accumulated and shared. Also, in all cases, a global synchronization is required between the Analysis and Build phases so that Analysis results that relate to the entire mosaic, such as pixel value distributions required to calculate the appropriate quantization extrema, can be accumulated and shared. The software should be portable because it is written in ANSI C and all inter-processor communication and synchronization is done using Message Passing Interface (MPI), which has been implemented on many platforms [34].

2.2.3. Sample Mosaics. Some sample image mosaics, constructed with the yourSky custom image mosaicking software, are shown in Figs. 2.4, 2.5 and 2.6.

Fig. 2.4 shows a full 90 arc second resolution, all-sky, $14,400 \times 7,200$ pixel mosaic constructed from 430 Infrared Astronomical Satellite (IRAS) [35] image plates in each of 4 wavelengths, 12, 25, 60, and 100 μm. High performance exploration of this and other large data sets is possible using visualization software developed previously at JPL [36], [37].

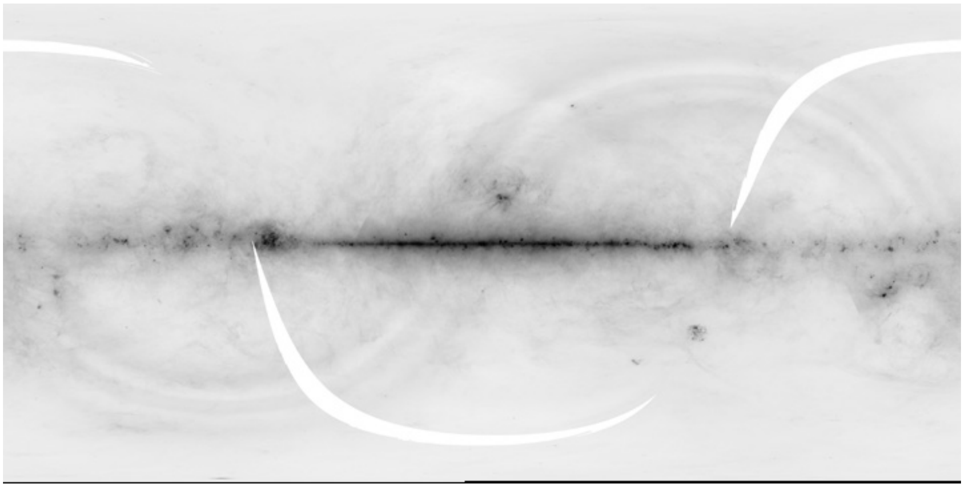
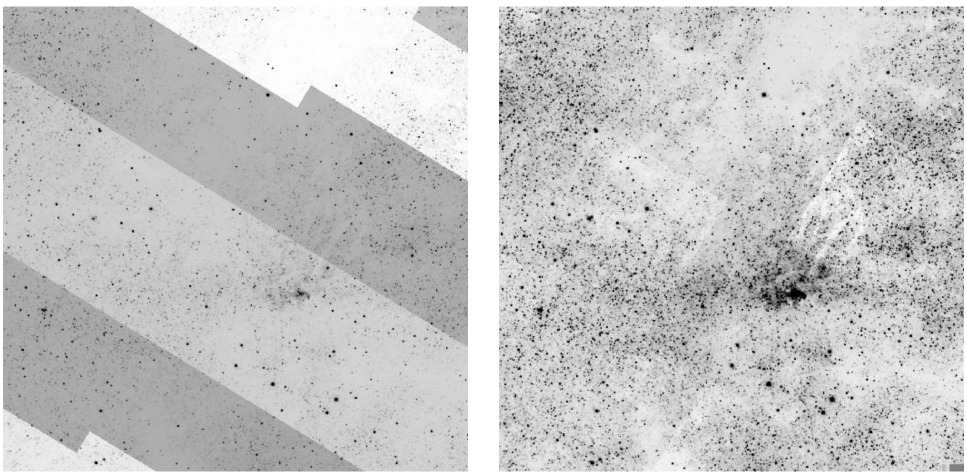


Fig. 2.4. *IRAS all-sky mosaic in the Cartesian (CAR) projection at 90 arc second resolution, constructed from 430 IRAS image plates in each of four wavelengths. The full resolution mosaic is* 14400×7200 *pixels.*

Fig. 2.5 shows a center of the galaxy mosaic from the 2MASS H band (1.65μm wavelength) before and after background matching is performed. The striped appearance without background matching is primarily due to atmospheric effects that become more pronounced as the path length through the atmosphere gets longer at different look angles. The background matching algorithm used by yourSky results in a more seamless mosaic, but edge effects are still visible. The Montage algorithms, described earlier, can be used to improve this background matching algorithm further.



(a) (b)

Fig. 2.5. *2MASS H band (1.65μm wavelength) center of the galaxy mosaic constructed from 16 2MASS image plates at 1 arc second resolution (a) without and (b) with background matching.*

Fig. 2.6 shows a DPOSS F band (650 nm wavelength) mosaic of M31 in a Galactic Tangent Plane projection. The mosaic shown in the figure is the center part of a larger $34,816 \times 36,352$ single precision floating point mosaic constructed from 9 DPOSS plates at full 1 arc second resolution.

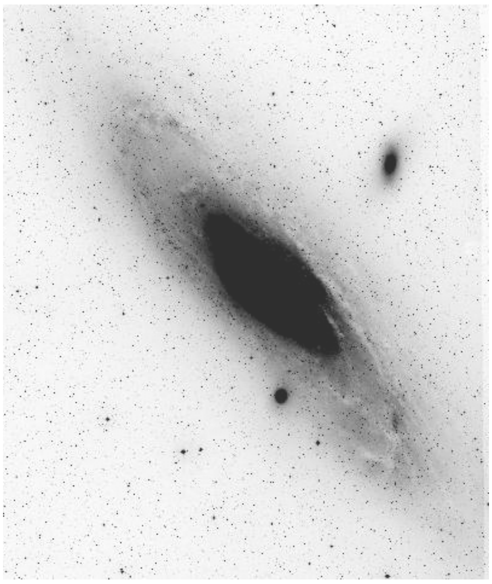


FIG. 2.6. *DPOSS F band (650 nm wavelength) mosaic of Andromeda (M31) at 1 arc second resolution. The image shown here is the center of a much larger mosaic constructed from 9 DPOSS plates.*

2.3. Request Management. Simultaneous mosaic requests are accepted from a simple HTML form submitted from the yourSky mosaic request web page, and queued on the yourSky server by a Common Gateway Interface (CGI) program interfacing with the Apache web server. The mosaic parameters for each request are stored on the yourSky server along with the identity of the user that submitted the request. The yourSky Mosaic Request Manager, shown in the architecture diagram in Fig. 2.1, needs to locate these mosaic requests and assign them one at a time to the Mosaic Request Handler. A user priority scheme is in place that starts off all users with equal priority. As requests are processed the user priorities change based on the number of mosaic pixels produced by each user in the past period referred to as the "priority window", currently set to 1 week. Users with the least number of mosaic pixels produced in the priority window period have highest priority for future mosaic requests. Furthermore, a mosaic request in progress that has had to wait for input image plate retrieval from a remote archive gets the highest priority to run next once all of the required input images have been retrieved. This ensures that all users get a chance to have their mosaic constructed and no single user will dominate all the available resources.

2.4. Plate Coverage Database. In order to be accessible by yourSky, all member surveys have to be included in the Plate Coverage Database that contains the minima and maxima of the longitudes (right ascensions) and latitudes (declinations) in each of the supported coordinate systems for all of the input image plates. The yourSky Mosaic Request Handler queries this database to determine which input image plates are needed to fulfill each mosaic request. The open source database, MySQL, is used to store this plate coverage information [38], [39]. The result of the query to the Plate Coverage Database is a list of the input image plates that are required to fulfill the mosaic request. These input image plates are retrieved from the appropriate remote archives, staged in a local data cache, and provided as an input to the image mosaicking software described in Section 2.2. The plate coverage database is also available as a stand-alone service, called the yourSky Archive Database Query.

2.5. Data Management. A data management scheme is implemented on the yourSky server to manage both a data cache for the input image plates, used to fulfill recent mosaic requests, and a work area, used to store recently constructed mosaics until they are downloaded.

The input data cache is maintained at a fixed size with image plates discarded on a least recently used basis. This enables mosaics to be recomputed with some changes to the custom request parameters without having to repeat the input image plate retrieval from the remote archives if the new request is resubmitted before the input images are purged from the cache. Also, mosaics of popular regions of the sky are likely to have their input image plates already cached on the yourSky server from previous requests, so they can be constructed more quickly.

Mosaics that have been completed are stored in a work area from which they may be downloaded by the appropriate users. Currently mosaics are purged after a 1 week period expires.

2.6. Data Archive Access. All of the publicly released data from two archives are currently accessible by yourSky, the Digitized Palomar Observatory Sky Survey (DPOSS) and the Two Micron All Sky Survey (2MASS) Second Incremental Data Release (2IDR).

DPOSS has captured nearly the entire northern sky at 1 arc second resolution in three wavelengths, 480 nm (J Band—blue), 650 nm (F Band—red), and 850 nm (N Band—near-infrared). The survey data were captured on photographic plates by the 48-inch Oschin Telescope at the Palomar Observatory in California [30]. The total size of the DPOSS data accessible by yourSky is roughly 3 TB, stored in over 2,600 overlapping image plates on the High Performance Storage System (HPSS) [40] at the Center for Advanced Computing Research (CACR) at the California Institute of Technology. The DPOSS plates are each about 1 GB in size and contain $23,552 \times 23,552$ pixels covering a roughly 6.5×6.5 degree region of the sky. The yourSky server uses a client program called the Hierarchical Storage Interface (HSI) to retrieve selected DPOSS plates in batch mode from the HPSS [41].

2MASS has captured nearly the entire sky at 1 arc second resolution in three near-infrared wavelengths, 1.25μm (J Band), 1.65μm (H Band), and 2.17μm (K_S Band). The survey data were captured using two 1.3 meter telescopes, one at Mt. Hopkins, AZ and one at the Cerro Tololo Inter-American Observatory (CTIO) in Chile [31]. The 2MASS archives contain roughly 10 TB of images and the subset that was released as part of the 2MASS Second Incremental Release (2IDR), nearly 4 TB, is fully accessible by yourSky. This 4 TB of data is stored in about 1.8 million overlapping plates managed by the Storage Resource Broker (SRB) at the San Diego Supercomputer Center (SDSC). Each 2MASS plate is about 2 MB in size and contains $512 \times 1,024$ pixels covering a roughly 0.15×0.30 degree region of the sky. The SRB is a scalable client-server system that provides a uniform interface for connecting to heterogeneous data resources, transparently manages replicas of data collections, and organizes data into "containers" for efficient access [42]. The yourSky server uses a set of client programs called SRB Tools to access selected 2MASS plates in batch mode from the SRB.

3. From yourSky to yourSkyG. In Section 2 we described how yourSky enables custom astronomical mosaic construction on a local multiprocessor system. Here, "local" refers to the fact that the mosaic computations are performed on the same machine that hosts the web server. The objectives of yourSkyG are (i) to provide the same kind of custom, web-accessible mosaic service as yourSky, but to perform the compute intensive portions remotely on a grid, and (ii) to make much larger mosaicking jobs feasible by leveraging the computational power of the grid to full advantage.

The yourSkyG portal runs on a local grid portal system and maintains compliance with grid security. It initiates data transfers to and from, and job executions on, a remote IPG computer. Local security rules dictate that the web services offered by yourSkyG and the grid portal may not be hosted on the same system. Instead, the web interface and Mosaic Request Manager are resident on a local web portal system that has no direct connection to the grid, as illustrated in Fig. 3.1. This web portal system accepts user requests and stores them in files on a local disk that is also accessible by the grid portal system. The yourSkyG job manager with IPG authentication is hosted on the grid portal system. Periodically this job manager checks for mosaic requests. When a request is found, the job manager does the following:

1. Retrieve the required input images from the remote sky survey archives
2. Get a grid proxy in order to authorize grid access
3. Upload the required input images to the target grid system
4. Generate a file in the Globus Resource Specification Language (RSL) that specifies the job to be run on the grid
5. Execute this RSL and wait for the resulting remote grid job to finish
6. Download the resulting output image
7. Remove files that are no longer needed from the remote grid system
8. Notify the user via e-mail
9. Optionally store the mosaic in an SRB archive

This architectural change successfully separates the grid portal from the less secure web portal, without requiring significant modifications to the yourSky architecture. However, this does requires coordination between the web and grid portals, which is accomplished through files stored on the shared disk.

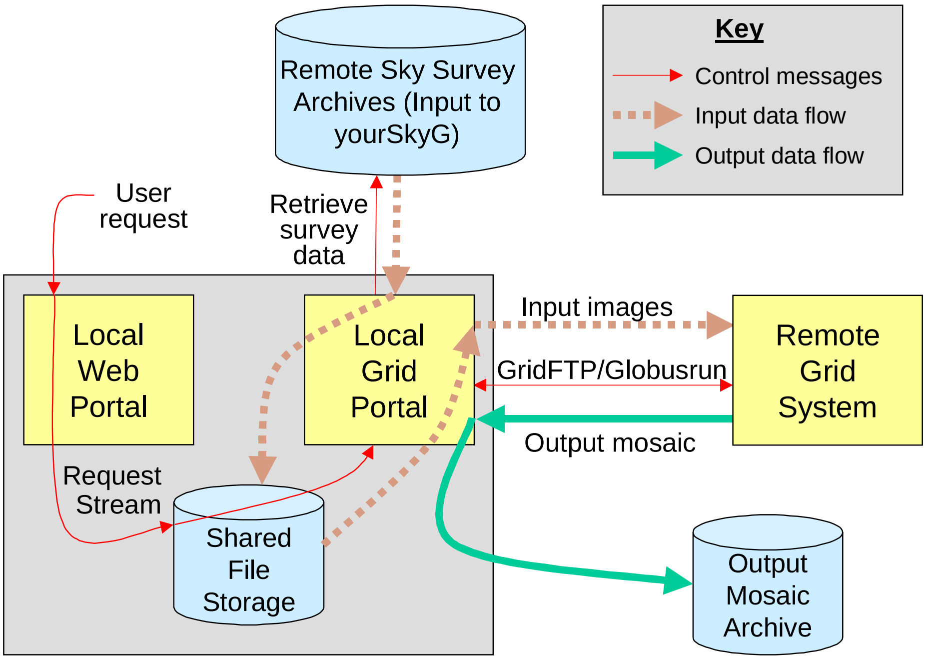


FIG. 3.1. *In the yourSkyG architecture, there is a clear separation between the local web and grid portals and the remote grid compute system where the mosaic processing is performed. The local grid portal coordinates the data flow and remote job execution.*

4. Optimizations for Large-Scale Mosaic Ensembles. The grid offers a wealth of computational, storage, and networking resources together with a rich set of tools for accessing them. The central problem is how to manage all of these resources in order to achieve an acceptable sustained throughput rate. This resource management problem is not only complex but also dynamic, with resource availability and usage changing over time. Resource management is of particular importance for the problem addressed in this paper, astronomical image mosaicking, because this is not only a compute intensive task but also a data intensive task. Not surprisingly, both computational resource scheduling and data communication are important issues. The architecture of yourSkyG addresses this resource management problem in the context of processing multiple mosaics, i. e., 100 or more mosaics at a time.

The key architectural features of yourSkyG are (*i*) state-based data flow, (*ii*) pipeline processing, and (*iii*) data reservoirs, described below. Together these produce beneficial characteristics in yourSkyG such as controlled usage of shared grid resources, improved throughput, and a degree of fault tolerance. Throughput is improved as a result of overlapping computation and communication, caching data close to the processing, and careful ordering of the mosaics to be processed to maximize the use of the data caches and minimize the amount of communication required. The use of a processing pipeline means that many different mosaics may be processed at the same time but at different phases in the processing sequence. For example, while one or more mosaics are being computed on the grid, the output from previous mosaic computations can also be downloaded and the inputs for later mosaics can be uploaded.

The following provides more detail about the key architectural features of yourSkyG, as well as some performance results summarizing our experiences in creating a set of 900 overlapping mosaics from DPOSS, collectively covering the entire northern sky in 6×6 degree patches with 1 arc second sampling. This ensemble of mosaics, totaling about 1.7 terabytes per wavelength, can be visualized using an all-sky, web-based image browsing service [43].

4.1. Ensemble Request Management. A mosaic ensemble request is a set of parameters, represented as a set of keyword-value pairs, that describe the mosaics to be constructed. Some of these parameters apply to all of the mosaics in the ensemble, for example, input survey and wavelength, coordinate system, projection, resolution, width and height in pixels. Others specify the individual mosaics, for example, right ascension and declination of the center.

Parameters are added to the mosaic ensemble request in two steps. First, the parameters that specify the input images required to compute each mosaic are added. Second, the parameters that specify the grid resources to be used in this computation are added. The final parameter set contains all the information required for computing the mosaics.

4.2. Ordering Mosaic Computations for Efficient Data Transport. Since large files must be moved to and from the remote grid system, data transport efficiency is extremely important. In our DPOSS plate ensemble example, each output DPOSS mosaic is 1.9 GB in size and on average requires processing of 7.6 input image plates each 1.1 GB for an average total transfer of over 10 GB. We have found in this example that without an adequate data transport strategy data transfer time can easily dwarf mosaic computation time.

If the requested mosaics in an ensemble are localized on the sky, some input images may be required for multiple output mosaics. The yourSkyG portal takes advantage of this by maintaining a data cache of input images on the target grid system so that these images may be reused whenever possible to reduce the volume of data transferred. Moreover, the order in which the mosaics in an ensemble are processed is selected to take maximum advantage of the available data cache. Returning to our DPOSS plate ensemble example, a single mosaic processed by itself requires on average the transfer of 8.5 GB of input images. However, if an ensemble of these mosaics are processed together, taking advantage of computation reordering and data caching may eliminate up to 85% of the input data transfers.

4.3. State-Based Processing Model. Conceptually, the processing of a mosaic job is modeled as a sequence of states and state transitions, as illustrated in Fig. 4.1. This model has been implemented so that each state is a directory and the objects in these states are files. Each state transition is then the movement of some file (object) from its current directory (current state) to a new directory (new state). The file that is moved from state to state may be an input image, an output mosaic, a mosaic job description, or a message of some kind. At any given time each possible state may be occupied by multiple objects.

An input image moves between the states of `input_awaiting_download`, `input_awaiting_upload`, and `input_cached`. The `input_awaiting_download` state means that the input image is scheduled for download from the remote survey archive. The `input_awaiting_upload` state means that the image is scheduled for upload to the remote grid system. The `input_cached` state means that the input image is preserved in a two-level cache, the primary on the remote grid system and the secondary on the local grid portal. The secondary cache is used to correct data transfer errors to the primary cache and for reuse with later mosaic ensemble requests. An output image mosaic moves between the states of `output_computed`, `output_downloaded`, `output_archived`, `output_purged_from_grid`, and `output_cached`. The transition to the `output_purged_from_grid` state means that files on the grid that are no longer needed have been removed to free grid resources for later jobs.

A mosaic ensemble request is partitioned into subsets, each of which is treated as a single batch job on the grid, referred to here as a "job". Each job moves between the states of `job_identifying_inputs_for_upload`, `job_awaiting_input_upload`, `job_ready_to_submit`, `job_queued`, `job_executing`, and `job_completed`.

Upon job completion, a message is sent back to the grid portal system that specifies the output files to be downloaded, their locations on the remote grid system, and their sizes for automated error checking.

4.4. Data Flow Model Using Concurrent Asynchronous Processes. In the yourSkyG data flow model, each state transition is implemented as a separate process and all of these processes execute asynchronously. Each state transition process executes in its own `current_state` directory, reads each file in that directory in time order, applies some operation to that file, and moves it to a `next_state` directory.

There is no direct communication between these processes and no centralized control. A state transition process executes whenever there is data available and processing occurs as rapidly as local resources allow. An output from a state transition process becomes an input for some other state transition process. The result is an efficient data flow architecture.

This design has beneficial software engineering features. Each state transition process is a small module easily constructed and modified, and easily inserted or removed from a much larger software structure. Most of the functions of a state transition process are common among all of these processes and need only be implemented once and reused.

This fine-grained architecture provides the user a high degree of control over the executing system. For example, one process that appears to have a problem may be selectively halted for further study while the rest of the system continues to execute. Halting a process has no serious consequences other than the accumulation

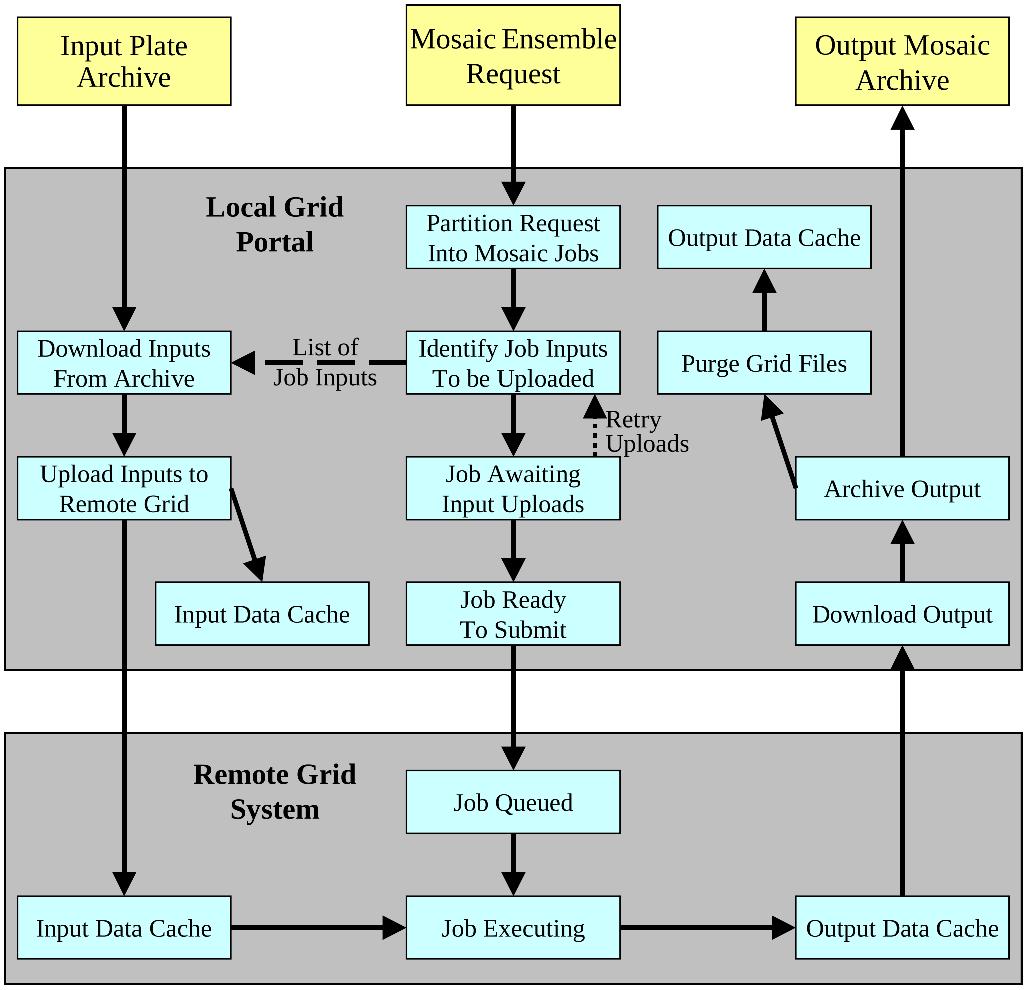


FIG. 4.1. *The yourSkyG computations to process multiple mosaics at a time on the grid can be modeled as a series of states and state transitions.*

of requests just prior to the inactive state transition. When the inactive process is restarted, processing returns to normal.

4.5. Pipeline with Reservoirs. The global architecture of yourSkyG is a pipeline created from a sequence of many independently executing state transition processes. Files move through this pipeline accumulating in different "reservoirs" and, in so doing, even out temporary fluctuations in local throughput. Given the data flow architecture, it is a straightforward extension to replicate the yourSkyG pipeline into multiple pipelines, executing asynchronously, and each targeting a different remote grid system.

A pipeline architecture by itself has the potential for greatly increased throughput. However, each of these state directories is also a data cache where files can accumulate until processing resources are available. Conceptually, each of these is a reservoir. Files accumulate in a reservoir when the state transition process following it is slower than the rest of the system and then drain out again when this state transition process speeds up again. As different resources speed up and slow down, bottlenecks move around but the reservoirs smooth out temporary variations and maintain a higher throughput rate. Only when the capacity of a resource is exceeded does that part of the pipeline shut down. For example, mosaic jobs that are ready to execute accumulate in the `job_ready_to_submit` directory but are only submitted to the batch queue when the number of mosaic jobs queued or executing on the remote grid system falls below a specific value. This prevents flooding the remote batch queue which would in turn block any other yourSkyG state process from accessing this resource; in particular, the state `job_awaiting_input_upload` would be unable to query for the current contents of the remote input data cache and would stall. However, even when one part of the pipeline shuts down for some reason, other parts do not. If the remote batch queue is full, the transfer of input files from archive to remote grid system will continue.

4.6. System Monitoring. We have found it essential to be able to monitor the functioning of the yourSkyG system in order to understand how the resources interact, determine the current system performance, and identify operational problems or implementation issues. The yourSkyG architecture makes this monitoring relatively simple.

Listing the contents of all the state directories provides a quick look at the current state of the pipeline. A bottleneck in the system is easily identified by an accumulation of files in a state directory and the directory identifies the particular resource to examine further. Inspection may include several computing systems and batch queues but these requests are easily automated.

In order to recover how the pipeline arrived in its current state, each process writes a log file to its `current_state` directory. Each log file records the name of each file processed, when it was processed, and the state transition applied. Redirecting `stdout` and `stderr` to the log file also captures any program or system error messages. Log file generation is one of the yourSkyG reusable components for state transition processes.

Generally, these few simple monitoring techniques will identify both the location and cause of a problem. If a particular problem is rare, a manual correction may be adequate. If a problem is chronic, then some form of automated error detection and correction may be necessary.

4.7. Automated Error Handling. We classify error handling into three sub-categories: detection, management, and correction.

Error handling begins with detection. In general, the supporting software used, such as the Globus tools, will report an error in a way that is automatically detectable by the calling program, usually a return value that lies within a specific range. However, there remain significant errors that are not reported this way. For example, a file transfer may fail at either end of the transfer route if one of the two host systems crashes or it may fail somewhere in the middle if a router malfunctions. The file may be nonexistent, zero length, or truncated at the destination, and that error not reported. The yourSkyG system performs an independent check on a transferred file to determine not only that the transferred file exists in its target location but also that that file has arrived having the correct size. For example, successful completion of a mosaic computation on a remote grid system is reported by the transfer of a small message file back to the local yourSkyG system containing a list of the output files to be downloaded and the file sizes that should be expected.

After an error has been detected, it must be managed in some way that protects the rest of the system so that validity of output data is not corrupted but regular processing continues with minimal disruption. This has been modeled and implemented as a state change; however, the new state cannot be the usual next state since that would indicate success. The preferred solution is to move the file responsible for this error to some preceding state and have the reprocessing correct the situation; however, careful design is required to prevent the possibility of an infinite loop. The more common alternative has been to create a fail state as a subdirectory of the state during which the error was detected and the file responsible for the error moved into this fail state directory. For example, if the transfer of a required input data file to the remote grid system fails, then that input data file is moved to the fail subdirectory of the current state directory, `input_awaiting_upload`, and the next input data file can be processed. Error correction is left for some other process.

Automatic error correction has been added to maintain reasonable processing throughput. For example, the `job_identifying_inputs_for_upload` state will check for a required input data file in the fail subdirectory for the `input_awaiting_upload` state before it attempts to download it again from the remote archive and will move this input data file back to the `input_awaiting_upload` state for another try. The function of the `job_awaiting_input_upload` state is to hold a mosaic job until it is verified that all necessary input files have been successfully transferred to the input data cache on the target grid system. Only then will the job be moved to the `job_ready_to_submit` state, meaning ready to submit to a batch queue on the target grid system. The transfer of input files is performed asynchronously by another pipeline and most mosaic job files will need to wait for these transfers to complete. However, if the `job_awaiting_input_upload` process finds that there is a required input file that not only is missing from the input data cache on the target grid system but also is missing from the `input_awaiting_upload` directory, then some error has occurred. The automated correction is to return the mosaic job file to the `job_identifying_inputs_for_upload` state for another try.

5. Performance Results. Here, we provide performance results when constructing a single mosaic on a multiprocessor system, as well as performance when constructing many of these mosaics in a batch processing mode on the grid.

5.1. Performance on a Single Mosaic. In this section we show timing results for the yourSky mosaicking software running on a SGI Origin 2000 with 300 MHz R12000 processors and 512 MB of RAM per processor. The first test mosaic is a 2×2 degree 2MASS mosaic of the galactic center at full one arc second resolution. The resulting mosaic is $7,201 \times 7,201$ single precision floating point pixels (207 MB) in size in a Galactic Tangent Plane projection, constructed from 174 2MASS plates totaling 365 MB in size. The second test mosaic is a 2×2 degree DPOSS mosaic of M31 at 10 arc seconds resolution, resulting in a 721×721 single precision floating point mosaic in the Galactic Tangent Plane projection. This 2.1 MB mosaic was constructed from 2 DPOSS plates (total size 2.2 GB). No background matching was performed for this test. Fig. 5.1 shows the wall clock time required to construct the mosaics on different numbers of processors on the Origin 2000. For the 2MASS mosaic, one processor was assigned to each input image plate, but all the processors were assigned to each input image plate for DPOSS. The plot shows the scaling curves for up to 64 processors.

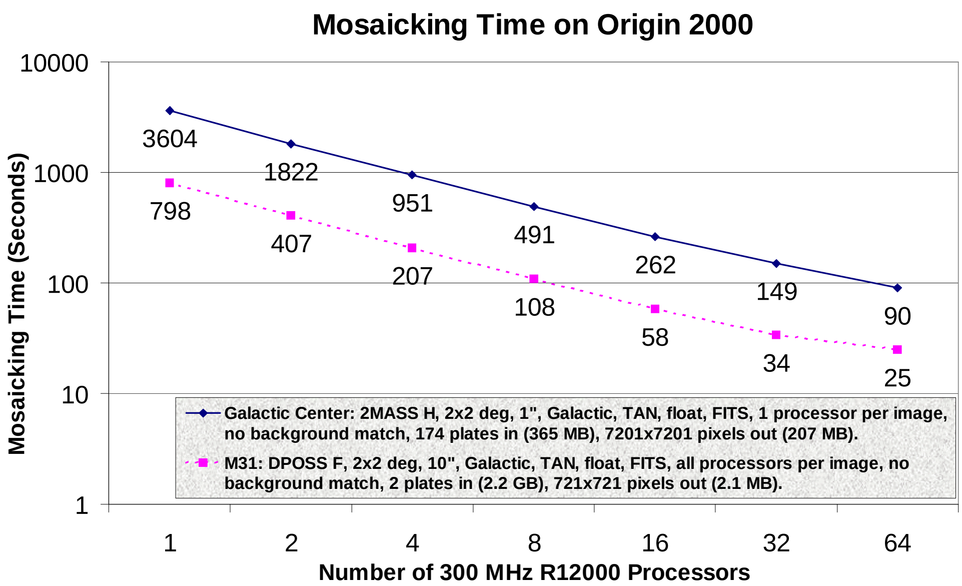


FIG. 5.1. *Mosaicking software performance on a SGI Origin 2000 for two different mosaic parameters, a 2×2 degree 1 arc second resolution mosaic of the galactic center in 2MASS with one processor assigned to each input image plate and a 2×2 degree 10 arc second resolution mosaic of M31 in DPOSS with all processors assigned to each input image plate.*

5.2. Performance on Multiple Mosaics. In this section we show typical throughput performance for yourSkyG generating an ensemble of 110 DPOSS mosaics. Each requested output mosaic is a 6×6 degree projection at 1 arc second resolution, resulting in 1.9 gigabytes per mosaic with 4 byte single precision floating point pixels. This yields a total size output of 205 gigabytes for the entire ensemble, which covers about 12% of the northern hemisphere covered by DPOSS. The input image plates are each 6.5×6.5 degrees at 1 arc second resolution, resulting in 1.1 gigabytes per input plate with 2 byte integer pixels. This yields a total size input of about 122 gigabytes for the entire ensemble.

The processing was automatically partitioned into 11 separate batch jobs, each computing 10 mosaics. The plot in Fig. 5.2 shows at any given time the percentages associated for each of the following values: total number of jobs started (on the local system), total number of input files transferred to the remote grid system cache, total number of jobs submitted to the batch queue on the remote grid system, and total number of output files transferred back to the local system. These values were extracted from the yourSkyG log files.

In order to ensure that valuable computational resources could be used for other purposes during data transfer, a 10-mosaic batch job was not scheduled until all of the input image plates it requires were transferred to a local disk on the remote grid system. The time from the start of the first job on the local grid portal to the return of the first output mosaic file is the time required to initialize the yourSkyG pipeline, which in this case

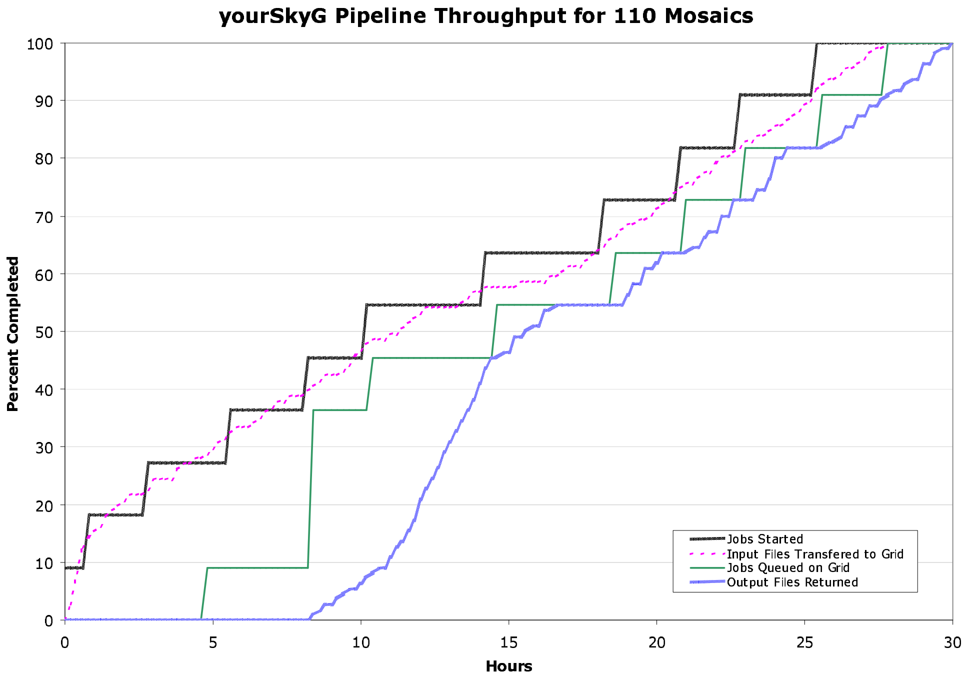


FIG. 5.2. *yourSkyG pipeline throughput for 110* 6×6 *degree DPOSS mosaics totalling about 205 GB in size.*

was 8 hours. The first half of this initialization time was spent transferring the input data required for the first 10-mosaic batch job and most of the second half was spent waiting in a batch queue on the remote grid system. However, once the pipeline was initialized, it returned computed mosaics at an average rate of 5 per hour or 120 per day. This is equivalent to a capability of mosaicking the entire DPOSS data set in about a week and the whole sky (both hemispheres) in roughly 2 weeks per wavelength at 1 arc second resolution.

This experiment also exercised the fault tolerance of the yourSkyG pipeline. Our logs indicate that during the computation of these 110 mosaics there were four failed input file transfers but each of these failures was automatically detected and corrected without user intervention. This fault tolerance is an essential feature for large scale processing on grids with distributed data archives and computational resources.

6. Summary. The yourSkyG portal uses the full computational power of the NASA Information Power Grid (IPG) to enable high-performance desktop access to custom astronomical image mosaics. The architecture of the portal allows it to exploit grid computing infrastructure, supercomputers and high bandwidth networks, on the server side. However, at the same time it is widely usable from virtually anywhere because the architecture also supports very lightweight computing resources on the client side, e.g., ordinary desktop computers with low bandwidth network connections. Since the user interface is a simple web form, the only client software required is the ubiquitous web browser, which most of the potential users probably already have and know how to use. This combination of being deployed in a high performance computing and communications environment while allowing access through simple portals running on the desktop makes yourSkyG a good match for the loosely coupled, distributed architecture of both the National Virtual Observatory (NVO) and the IPG.

The portal includes subsystems for: (i) construction of the image mosaics on multiprocessor systems or computational grids, (ii) managing simultaneous user requests, (iii) determining which image plates from member surveys are required to fulfill a given request, (iv) caching input image plates and the output mosaics between requests, and (v) retrieving input image plates from remote archives. The parallel image mosaicking software emphasizes custom access to mosaics, allowing the user to specify parameters that describe the mosaic to be built, including data sets to be used, location on the sky, size of the mosaic, resolution, coordinate system, projection, data type, and image format.

The grid work flow is optimized to achieve high-throughput processing of multiple mosaics to be constructed together as ensembles. The key architectural features for this mode of processing are a state-based data flow system, pipeline processing to overlap mosaic computations and data communications where possible, and the use of data reservoirs at various stages of the processing pipeline to provide a level of robustness in the face of varying load conditions on shared grid resources.

Acknowledgments. The yourSky portal was initially developed under sponsorship of NASA's Space Science Applications of Information Technology Program, Office of Space Science. The yourSkyG portal, a port of yourSky to the Information Power Grid, was sponsored by NASA's Computing, Networking, and Information Systems (CNIS) Program.

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

REFERENCES

[1] The National Virtual Observatory (USA), `http://us-vo.org`.

[2] National Virtual Observatory Science Definition Team Report: *Towards the National Virtual Observatory*, April 2002.

[3] NVO White Paper: *Toward a National Virtual Observatory: Science Goals, Technical Challenges, and Implementation Plan*, in Virtual Observatories of the Future, R. J. Brunner, S. G. Djorgovski, and A. S. Szalay, eds., Astronomical Society of the Pacific Conference Series, Vol. 225, pp. 353-372, 2001.

[4] R. G. Mann, A. Lawrence, C. Davenhall, R. McMahon, M. Irwin, N. Walton, G. Rixon, M. Watson, J. Osborne, C. Page, P. Allan, D. Giaretta, C. Perry, D. Pike, J. Sherman, F. Murtagh, L. Harra, B. Bentley, K. Mason, and S. Garrington, *AstroGrid: the UK's Virtual Observatory Initiative*, Proceedings of Astronomical Data Analysis Software and Systems XI, ASP Conference Series, Vol. 281, 2002, D. A. Bohlender, D. Durand, and T. H. Handley, eds.

[5] P. Quinn, P. Benvenuti, P. Diamond, F. Genova, A. Lawrence, and Y. Mellier, *The AVO Project: A European VO Initiative*, Proceedings of Astronomical Data Analysis Software and Systems XI, ASP Conference Series, Vol. 281, 2002, D. A. Bohlender, D. Durand, and T. H. Handley, eds.

[6] The Australian Virtual Observatory, `http://www.aus-vo.org/`.

[7] The Virtual Observatory - India, `http://vo.iucaa.ernet.in/~voi/`.

[8] The International Virtual Observatory Alliance (IVOA), `http://www.ivoa.net`.

[9] W. D. Pence, T. McGlynn, P. Chai, and C. Heikkila, *Hera: The HEASARC Web Based Data Analysis Environment*, Proceedings of SPIE Astronomical Telescopes and Instrumentation: Virtual Observatories Conference, August 2002.

[10] P. Di Matteo, R. Capuzzo Dolcetta, P. Miocchi, V. Antonuccio-Delogu, U. Becciani, A. Costa, and V. Rosato, *Astrocomp: A Web Portal for High Performance Computing on a Grid of Supercomputers*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[11] J. C. Jacob, R. J. Brunner, D. Curkendall, S. G. Djorgovski, J. C. Good, L. Husman, G. Kremenek, and A. Mahabal, *yourSky: Rapid Desktop Access to Custom Astronomical Image Mosaics*, Proceedings of SPIE Astronomical Telescopes and Instrumentation: Virtual Observatories Conference, August 2002.

[12] A. Allen, T. Naylor, I. Steele, D. Carter, J. Etherton, and C. Motteram, *eSTAR: Building an Observational GRID*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[13] A. Szalay, T. Budavari, T. Malik, J. Gray, and A. Thakar, *Web Services for the Virtual Observatory*, Proceedings of SPIE Astronomical Telescopes and Instrumentation: Virtual Observatories Conference, August 2002.

[14] T. Budavari, T. Malik, A. Szalay, A. Thakar, and J. Gray, *SkyQuery—A Prototype Distributed Query Web Service for the Virtual Observatory*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[15] P. Fernique, A. Schaaff, F. Bonnarel, and T. Boch, *A Bit of GLUe for the VO: Aladin Experience*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[16] C. Arviset, J. Dowson, J. Hernandez, P. Osuna, and A. Venet, *Interoperability of ESA Science Archives*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[17] D. J. Mink, and M. J. Kurtz, *Federating Catalogs and Interfacing Them with Archives: A VO Prototype*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[18] M. G. Allen, F. Genova, F. Ochsenbein, S. Derriere, C. Arviset, P. Didelon, M. Dolensky, S. Garrington, R. Mann, A. Micol, A. Richards, G. Rixon, and A. Wicenec, *Toward an AVO Interoperability Prototype*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[19] J. C. Good, M. Kong, and G. B. Berriman, *OASIS: A Data Fusion System Optimized for Access to Distributed Archives*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[20] J. C. Jacob, R. Williams, J. Babu, S. G. Djorgovski, M. J. Graham, D. S. Katz, A. Mahabal, C. D. Miller, R.

Nichol, D. E. Vanden Berk, and H. Walia, *Grist: Grid Data Mining for Astronomy*, Proceedings of Astronomical Data Analysis Software and Systems XIV, ASP Conference Series, Vol. 347, 2005, P. L. Shopbell, M. C. Britton, and R. Ebert, eds.

[21] S. Derriere, F. Ochsenbein, T. Boch, and G. T. Rixon, *Metadata for the VO: The Case of UCDs*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[22] The TeraGrid, `http://www.teragrid.org`.

[23] B. Berriman, D. Curkendall, J. C. Good, L. Husman, J. C. Jacob, J. M. Mazzarella, R. Moore, T. A. Prince, and R. E. Williams, *Architecture for Access to Compute Intensive Image Mosaic and Cross-Identification Services in the NVO*, Proceedings of SPIE Astronomical Telescopes and Instrumentation: Virtual Observatories Conference, August 2002.

[24] G. B. Berriman, D. Curkendall, J. Good, J. Jacob, D. S. Katz, T. Prince, and R. Williams, *Montage: An On-Demand Image Mosaic Service for the NVO*, Proceedings of Astronomical Data Analysis Software and Systems XII, ASP Conference Series, Vol. 295, 2003, H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

[25] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, S. Patil, M.-H. Su, K. Vahi, and M. Livny, *Pegasus: Mapping Scientific Workflows Onto the Grid*, Proceedings of Across Grids Conference, 2004.

[26] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, K. Blackburn, A. Lazzarini, A. Arbree, R. Cavanaugh, and S. Koranda, *Mapping Abstract Complex Workflows Onto Grid Environments*, Journal of Grid Computing, Vol. 1, No. 1, pp. 25-39, 2003.

[27] J. Frey, T. Tannenbaum, M. Livny, and S. Tuecke, *Condor-G: A Computation Management Agent for Multi-Institutional Grids*, Proceedings of the 10th IEEE International Symposium on High-Performance Distributed Computing, 2001.

[28] E. Bertin, SWarp User's Guide.

[29] D. Makovoz and I. Khan, *Mosaicking with MOPEX*, Proceedings of Astronomical Data Analysis Software and Systems XIV, ASP Conference Series, Vol. 347, 2005, P. L. Shopbell, M. C. Britton, and R. Ebert, eds.

[30] S. G. Djorgovski, R. R. Gal, S. C. Odewahn, R. R. de Carvalho, R. Brunner, G. Longo, and R. Scaramella, *The Palomar Digital Sky Survey (DPOSS)*, in Wide Field Surveys in Cosmology, S. Colombi and Y. Mellier, eds.

[31] M. F. Skrutskie, R. M. Cutri, R. Stiening, M. D. Weinberg, S. Schneider, J. M. Carpenter, C. Beichman, R. Capps, T. Chester, J. Elias, J. Huchra, J. Liebert, C. Lonsdale, D. G. Monet, S. Price, P. Seitzer, T. Jarrett, J. D. Kirkpatrick, J. Gizis, E. Howard, T. Evans, J. Fowler, L. Fullmer, R. Hurt, R. Light, E. L. Kopan, K. A. Marsh, H. L. McCallon, R. Tam, S. Van Dyk, and S. Wheelock, *The Two Micron All Sky Survey (2MASS)*, The Astronomical Journal, Vol. 131, pp. 1163-1183, 2006.

[32] The Flexible Image Transport System (FITS), `http://fits.gsfc.nasa.gov`, `http://www.cv.nrao.edu/fits`.

[33] E. W. Greisen and M. R. Calabretta, *Representations of World Coordinates in FITS*, Astronomy & Astrophysics, Vol. 395, pp. 1061-1075, 2002.

[34] MPI: A Message Passing Interface Standard, `http://www-unix.mcs.anl.gov/mpi`.

[35] C. A. Beichman, G. Neugebauer, H. J. Habing, P. E. Clegg, T. J. Chester, and The Joint IRAS Science Working Group, *Infrared Astronomical Satellite (IRAS) Explanatory Supplement*, 1988.

[36] J. C. Jacob and L. Plesea, *Fusion, Visualization and Analysis Framework for Large, Distributed Datasets*, IEEE Aerospace Conference, 2001, ISBN 0-7803-6600-X.

[37] J. C. Jacob and L. E. Husman, *Large Scale Visualization of Digital Sky Surveys*, Virtual Observatories of the Future, R. J. Brunner, S. G. Djorgovski, and A. S. Szalay, eds., Astronomical Society of the Pacific Conference Series, Vol. 225, pp. 291-296, 2001.

[38] M. Widenius, MySQL AB, D. Axmark, *MySQL Reference Manual: Documentation from the Source*, O'Reilly, June 2002, ISBN 0-59600-265-3.

[39] MySQL, `http://www.mysql.com`.

[40] The High Performance Storage System (HPSS) at Caltech's Center for Advanced Computing Research, `http://www.cacr.caltech.edu/resources/HPSS/index.html`.

[41] Hierarchical Storage Interface (HSI) for High Performance Storage Systems (HPSS), `http://www.sdsc.edu/Storage/hsi`.

[42] The Storage Resource Broker (SRB) at San Diego Supercomputer Center (SDSC), `http://www.npaci.edu/SRB`.

[43] J. C. Jacob, G. Block, and D. W. Curkendall, *Architecture for All-Sky Browsing of Astronomical Datasets*, Proceedings of Astronomical Data Analysis Software and Systems XII, Baltimore, MD, Oct. 13-16, 2002, Astronomical Society of the Pacific Conference Series, H. Payne, R. Jedrzejewski, and R. Hook, eds.

Edited by: Marcin Papzycki
Received: April 1, 2003
Accepted: February 21, 2006