ImageX: New and improved Image Explorer for astronomical images and beyond

Soichi Hayashi[a], Arvind Gopu[a], Ralf Kotulla[b], and Michael D. Young[a]

[a]Indiana University, 2709 E 10th St., Bloomington, IN 47408, USA
[b]Department of Astronomy, University of Wisconsin - Madison, 475 N Charter St, Madison, WI 53706, USA

ABSTRACT

The One Degree Imager - Portal, Pipeline, and Archive (ODI-PPA) has included the Image Explorer interactive image visualization tool since it went operational. Portal users were able to quickly open up several ODI images within any HTML5 capable web browser, adjust the scaling, apply color maps, and perform other basic image visualization steps typically done on a desktop client like DS9. However, the original design of the Image Explorer required lossless PNG tiles to be generated and stored for all raw and reduced ODI images thereby taking up tens of TB of spinning disk space even though a small fraction of those images were being accessed by portal users at any given time. It also caused significant overhead on the portal web application and the Apache webserver used by ODI-PPA. We found it hard to merge in improvements made to a similar deployment in another project's portal. To address these concerns, we re-architected Image Explorer from scratch and came up with ImageX, a set of microservices that are part of the IU Trident project software suite, with rapid interactive visualization capabilities useful for ODI data and beyond. We generate a full resolution JPEG image for each raw and reduced ODI FITS image before producing a JPG tileset, one that can be rendered using the ImageX frontend code at various locations as appropriate within a web portal (for example: on tabular image listings, views allowing quick perusal of a set of thumbnails or other image sifting activities). The new design has decreased spinning disk requirements, uses AngularJS for the client side Model/View code (instead of depending on backend PHP Model/View/Controller code previously used), OpenSeaDragon to render the tile images, and uses nginx and a lightweight NodeJS application to serve tile images thereby significantly decreasing the Time To First Byte latency by a few orders of magnitude. We plan to extend ImageX for non-FITS images including electron microscopy and radiology scan images, and its featureset to include basic functions like image overlay and colormaps. Users needing more advanced visualization and analysis capabilities could use a desktop tool like DS9+IRAF on another IU Trident project called StarDock, without having to download Gigabytes of FITS image data.

Keywords: Interactive Image Visualization, NodeJS, OpenSeaDragon, Javascript

1. BACKGROUND & DESIGN PHILOSOPHY

The One Degree Imager - Portal, Pipeline, and Archive (ODI-PPA) [1] has included the Image Explorer interactive image visualization tool since it went operational. Users of ODI [2] were able to quickly open up several ODI images within any HTML5 capable web browser, adjust the scaling, apply color maps, and perform other basic image visualization steps typically done on a desktop client like DS9 [3].

However, the original design of the Image Explorer [4] required lossless PNG tiles to be generated and stored for all raw and reduced ODI images thereby taking up tens of TB of spinning disk space. We use a spinning disk server with 24 TB of disk to store these tile files, and it was filled up completely with 2 years worth of ODI raw & reduced data. We researched our logs and found that only a small fraction of those images ($< 5\%$) were being accessed by portal users at any given time. The original Image Explorer design also required a complex web application to be included within a web portal like ODI-PPA. Any improvements made to a similar deployment in another projects portal, for example on our Electron Microscopy Centers portal could not be easily merged

Further author information, contact agopu@iu.edu

Software and Cyberinfrastructure for Astronomy III, edited by Gianluca Chiozzi, Juan C. Guzman, Proc. of SPIE Vol. 9913, 99134E · © 2016 SPIE
CCC code: 0277-786X/16/$18 · doi: 10.1117/12.2233425

into the ODI-PPA Image Explorer. Finally we were serving the image tiles through the same web application responsible for the portal itself (PHP/Zend Framework) creating a large overhead for the application and the webserver (Apache).

To address these concerns, we have re-architected Image Explorer and come up with ImageX, a set of microservices that are part of the IU Trident project software suite [5], with rapid interactive visualization features useful for ODI data and beyond. By default we use simple application modules to convert proprietary or domain specific file formats (for example, FITS in astronomy, DM3 in Electron Microscopy) to a standard format (JPG or PNG) before producing JPG (default, or PNG) tile images. We expect to expand the current functionality to include simple features like image overlay and colormaps in the near future but users interested in further advanced image visualization and analysis can use DS9+IRAF (or similar set of applications) on the StarDock system.

ImageX's design allows updates to the tile generation algorithms to be easily applied to regenerate tiles while also having vastly decreased spinning disk requirements. Other improvements include use of Docker containers to deploy backend services, use of AngularJS [6] for the client side Model/View code (instead of depending on backend PHP Model/View/Controller code previously used), the widely used OpenSeaDragon Javascript library [7] to render the tile images, use of nginx [8], and a lightweight NodeJS [9] application to serve tile files thereby significantly improving the user experience and responsiveness of our image preview capability

2. IMAGEX DESIGN DESCRIPTION

Shown in Figure 1 is a simple design schematic of the redesigned ImageX suite of microservices. It includes the server side backend elements described below as well as the frontend/user interface elements described in 2.2.

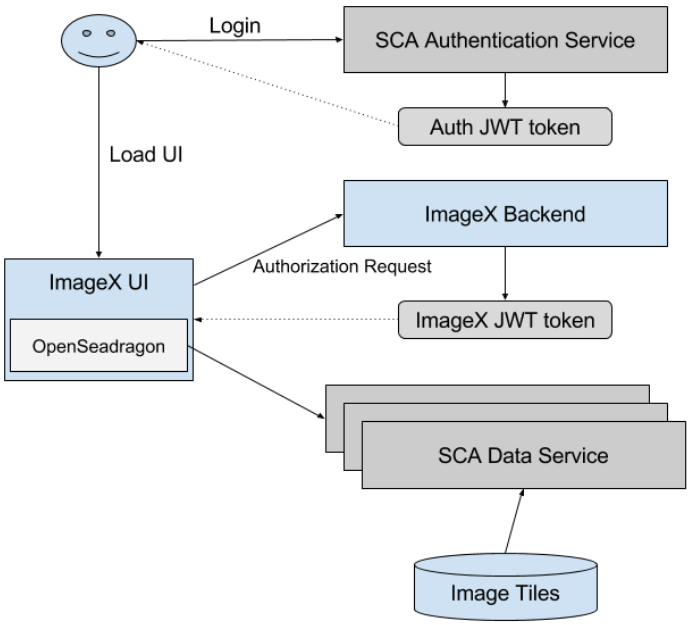

Figure 1. ImageX Design Schematic

2.1 ImageX Server Side

2.1.1 SCA Authentication Service

A shared authentication microservice, that is part of the Trident Scalable Compute Archive project microservice suite, provides Single-Sign-On capabilities. It handles the actual user authentication and generation of authentication token (JWT). JWT tokens contains basic information about the user, and roles that user is authorized,

and it is signed by SCA authentication service's private key. All other service, that are configured with SCA authentication service's public key can verify this token and trust that user has a valid token issued by the SCA authentication service. By using JWT, we can provide stateless authorization model and easily scale our application horizontally without having to provide a shared database among different services.

2.1.2 ImageX backend Service

Once the user is authenticated, the ImageX backend service receives requests to access ODI tile images. It looks up information from its database to determine if the user has access to that image. If user can access to the image, it issues a secondary JWT token (using a private key stored for ImageX backend) which allows access to that specific image. This lightweight application is written in NodeJS.

2.1.3 Tile Server / SCA Data Service

The previous version of Image Explorer relied on the ODI-PPA portal's PHP application to authorize the user by PHP session and DB lookup, and stream individual tiles using the Apache webserver. This proved to be very inefficient with high Time-To-First-Byte (TTFB) values and low bandwidth.

We have created an independent microservice called SCA Data service which is responsible for providing access to arbitrary data (including tile images for ImageX) stored on disk mounted on the host running the data service. The SCA data service allows us to use a JWT public key to verify if user has access to the requested data.

Unlike the ODI-PPA portal, we can easily instantiate as many data service instances to meet demands, and serving of tiles would not interfere with other services. With SCA Data service's stateless session management, we have reduced our image loading latency from 200-300ms TTFB down to 80-100ms TTFB for each tile image. The overall throughput of the data transfer for each images has also decreased from about 500ms to 100ms. These are show in figures 2 and 3

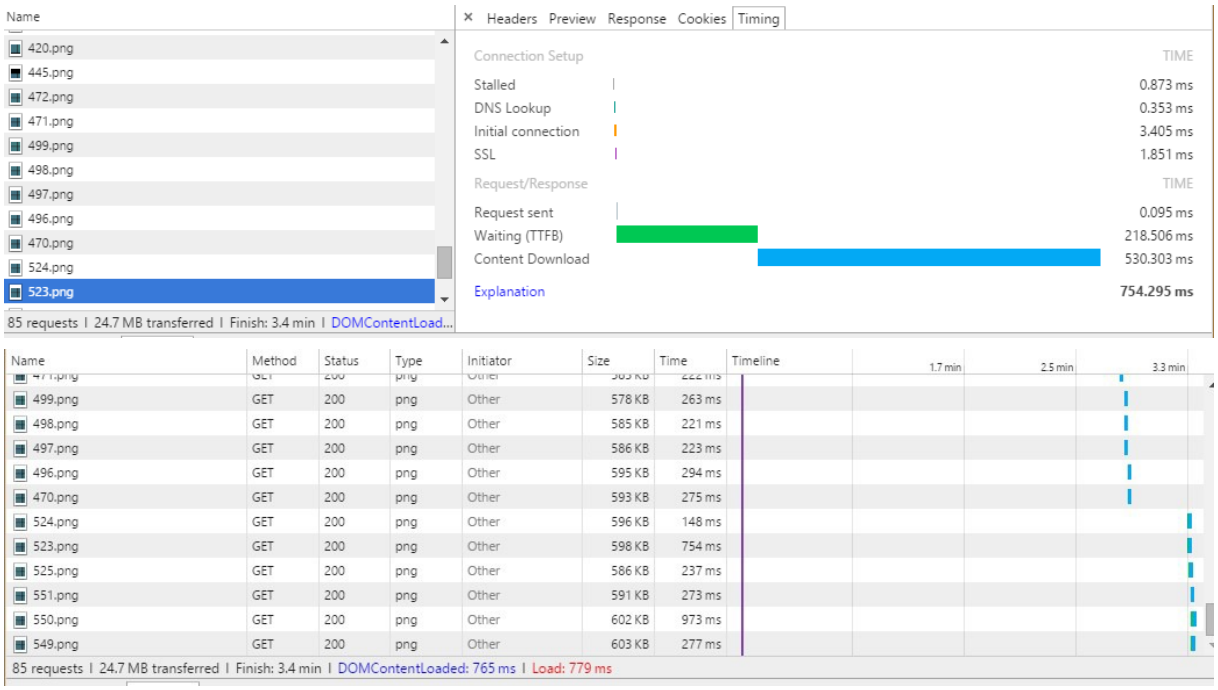

Figure 2. Serving Image Explorer tile(s) from previous Apache+PHP based ODI portal

When hundreds of these tiles are served to the user, there is a significant improvement to the usability / responsiveness of our image viewer. Apart from improving the performance of our application, splitting of our ODI-PPA application allows us to reuse our SCA data service to serve other data types for other services.

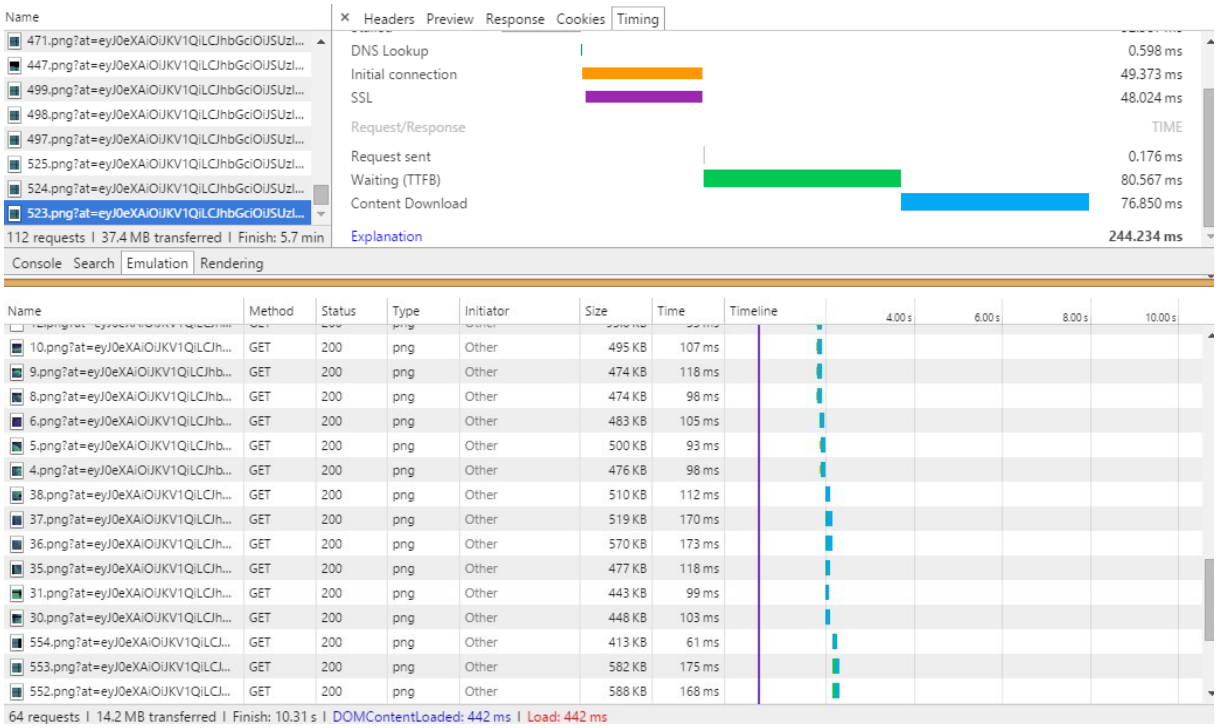

Figure 3. Serving ImageX tile(s) from Nginx + NodeJS + SCA Data Service ImageX backend

2.2 ImageX Frontend User Interface

The original Image Explorer used jquery-tileviewer (a home grown library) to render image tiles. ImageX uses an alternative widely used & well supported Javascript library called Open Seadragon as the image tile rendering framework – this decreases the maintenance effort needed on our side and to makes it easier for other projects to reuse our components. Since ImageX itself is a collection of lightweight microservices, it can be deployed or accessed from several locations where images may be displayed including:

- A search results page or a collections page with thumbnails
- A workflow output page displaying data products produced a pipeline
- An operator vetting page used to approve pipeline data products
- A users laptop or desktop to visualize images offline

The redesigned ImageX User Interface allows a user to choose

- A collection view as shown in figure 4, or
- A sequence/film strip view as shown in figure 5.

The Collection view is particularly helpful in sifting through a large number of images to look for scientifically useful data products. In the example set of images shown in 4, it becomes quickly evident that 2nd image from the bottom right is a stack, and that its scaling needs to be tweaked.

ImageX allows the user to seamlessly and rapidly zoom in/out of any of the images displayed in either of the views. An example of this is shown in 4. The ImageX user interface also allows the user to rotate the image(s) by an arbitrary angle as shown in 6

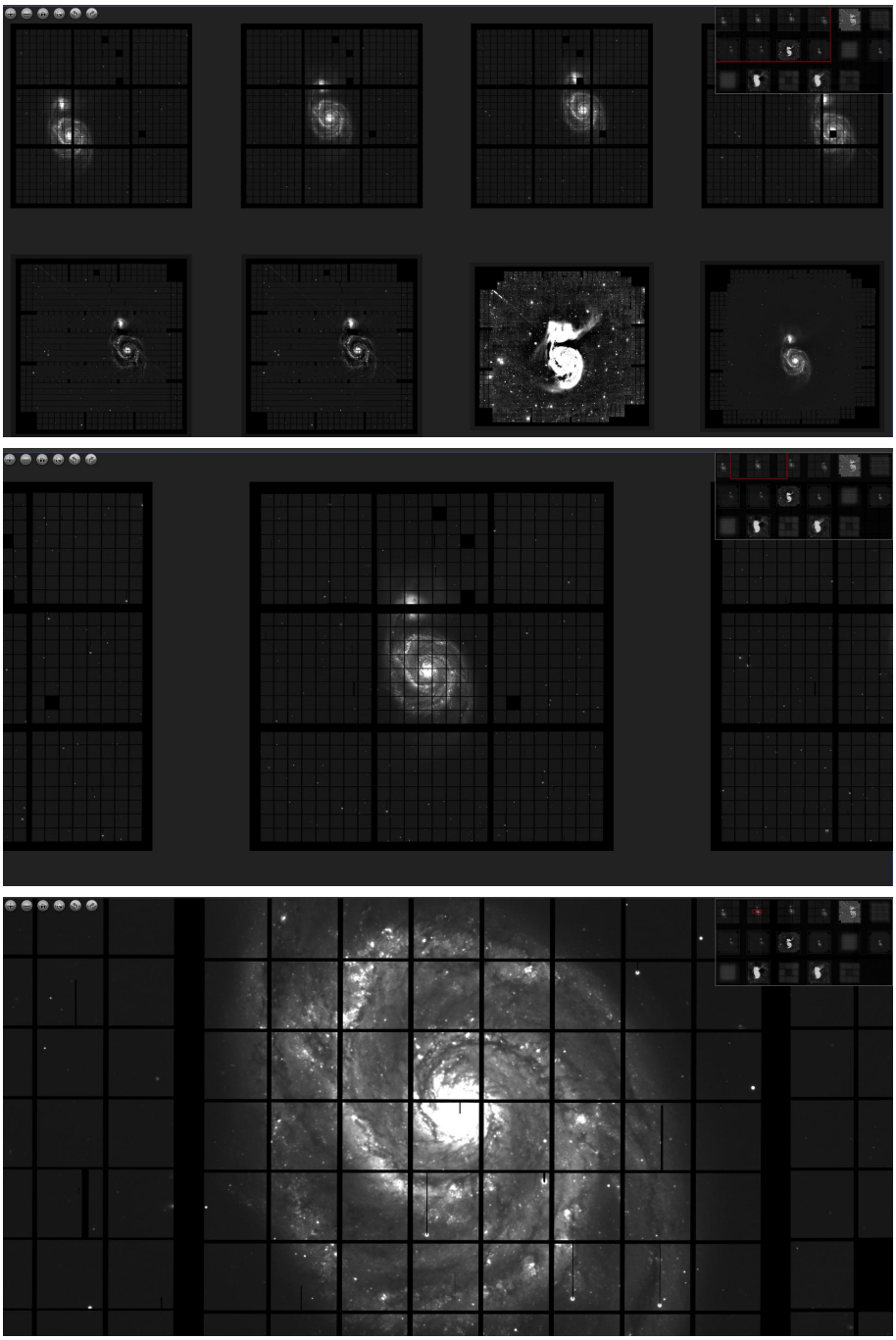

Figure 4. ImageX Collection View Example

2.3 ImageX Tile Pyramid Generation

2.3.1 JPG / PNG Tile Image Pyramid Generation

The original Image Explorer and the jquery-tileviewer library it used, depended on a proprietary image pyramid format similar to DZI (DeepZoom Image). With the switch to using OpenSeadragon as the image rendering library, we also have switched to using the standard DZI format (supported by OpenSeadragon and other similar tools). This allows us to use existing open-source tools such as deepzoom.py [10] generate our DZI tiles.

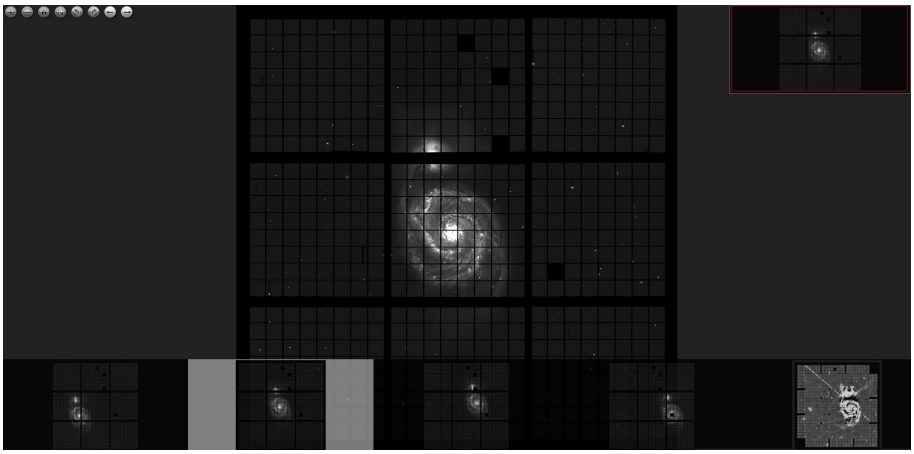

Figure 5. ImageX Sequence View Example

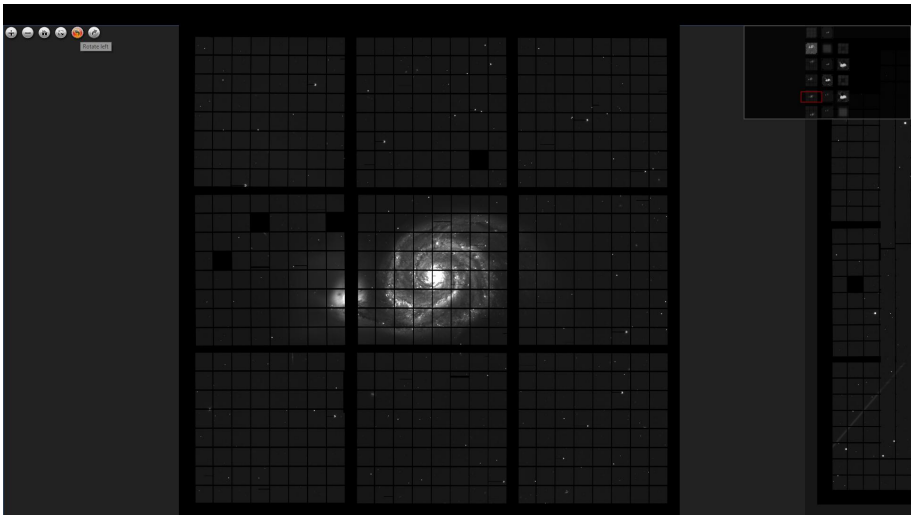

Figure 6. ImageX Rotate Image Example

Deepzoom.py allows us to generate tiles in JPG, PNG with adjustable compression rate and other parameters

2.3.2 Astronomy Data

FITS is the most common format used by Astronomy instruments and archives. In order to display images through ImageX, a FITS image has to be first converted to JPEG/PNG. Although user can do this conversion using a common image conversion tool like GraphicsMagick, it often results in non-optimal mapping from FITS's floating point value to JPEG/PNG's 32bit color depth. We have implemented the following application modules:

- fits2img.py: Analyzes FITS image and select optimal scaling parameter with configurable algorithm and generate JPEG / PNG image.
- mef2fits.py: Join multi-extension FITS into a single extension FITS.

2.3.3 ODI Data: Raw to Basic Calibrated Data

The raw instrumental data coming off the ODI instrument includes 30 Multi Extension FITS (MEF) files. We use a simple DockQR deployment – a Dockerized Quick Reduce pipeline deployed on dedicated hardware used by the ODI project – to produce basic calibrated images. Then we use the Multi Extension FITS file to Single Extension FITS file converter (mef2fits.py) previously described to produce a simple Single Extension FITS file.

3. CONCLUSION & FUTURE WORK

The totally redesigned ImageX suite of microservices provides rapid interactive visualization features useful for ODI data and beyond. It uses domain specific applications to convert proprietary or domain specific file formats (for example, FITS in astronomy, DM3 in Electron Microscopy) to a standard format (JPG, for example) before producing JPG (default, or PNG) tile images. ImageX's design allows the use of Docker containers to deploy backend services, use of AngularJS for the client side Model/View code (instead of depending on backend PHP Model/View/Controller code previously used), OpenSeaDragon to render the tile images, use of nginx and a lightweight NodeJS application to serve tile files thereby significantly improving the user experience and responsiveness of our image preview capability. We expect to add initial file format converter modules to the ImageX software stack in order to convert Electron Microscopy images in dm3 format and Radiology images in dicom/nifti formats. We also plan to explore additional client side Javascript functions within the ImageX view especially image overlay and colormaps. Finally, resources permitting, we plan to integrate advanced Astronmical Image Visualization / Analysis with StarDock based DS9+IRAF Docker containers.

REFERENCES

[1] Gopu, A., Hayashi, S., Young, M. D., Harbeck, D. R., Boroson, T., Liu, W., Kotulla, R., Shaw, R., Henschel, R., Rajagopal, J., Stobie, E., Knezek, P., Martin, R. P., and Archbold, K., "Odi - portal, pipeline, and archive (odi-ppa): a web-based astronomical compute archive, visualization, and analysis service," (2014).

[2] Harbeck, D. R., Boroson, T., Lesser, M., Rajagopal, J., Yeatts, A., Corson, C., Liu, W., Dell'Antonio, I., Kotulla, R., Ouellette, D., Hooper, E., Smith, M., Bredthauer, R., Martin, P., Muller, G., Knezek, P., and Hunten, M., "The wiyn one degree imager 2014: performance of the partially populated focal plane and instrument upgrade path," (2014).

[3] Joye, W. A. and Mandel, E., "New Features of SAOImage DS9," in [*Astronomical Data Analysis Software and Systems XII*], Payne, H. E., Jedrzejewski, R. I., and Hook, R. N., eds., *Astronomical Society of the Pacific Conference Series* **295**, 489 (2003).

[4] Gopu, A., Hayashi, S., and Young, M., "Image explorer: Astronomical image analysis on an html5-based web application.," in [*ADASS XXIII*], Friedel, D., Freemon, M., and Plante, R., eds., *ASP Conf. Ser.* **485**, 417, ASP, San Francisco (2013).

[5] Gopu, A., Hayashi, S., Young, M. D., Kotulla, R., Henschel, R., and Harbeck, D. R., "Trident: Scalable compute archive, and visualization/analysis systems," (2016).

[6] "Angularjs." http://angularjs.org.

[7] "Open seadragon." https://openseadragon.github.io/.

[8] "Nginx server." https://nginx.org.

[9] "Nodejs." https://nodejs.org.

[10] "Openzoom: Python deep zoom tools." https://github.com/openzoom/deepzoom.py.