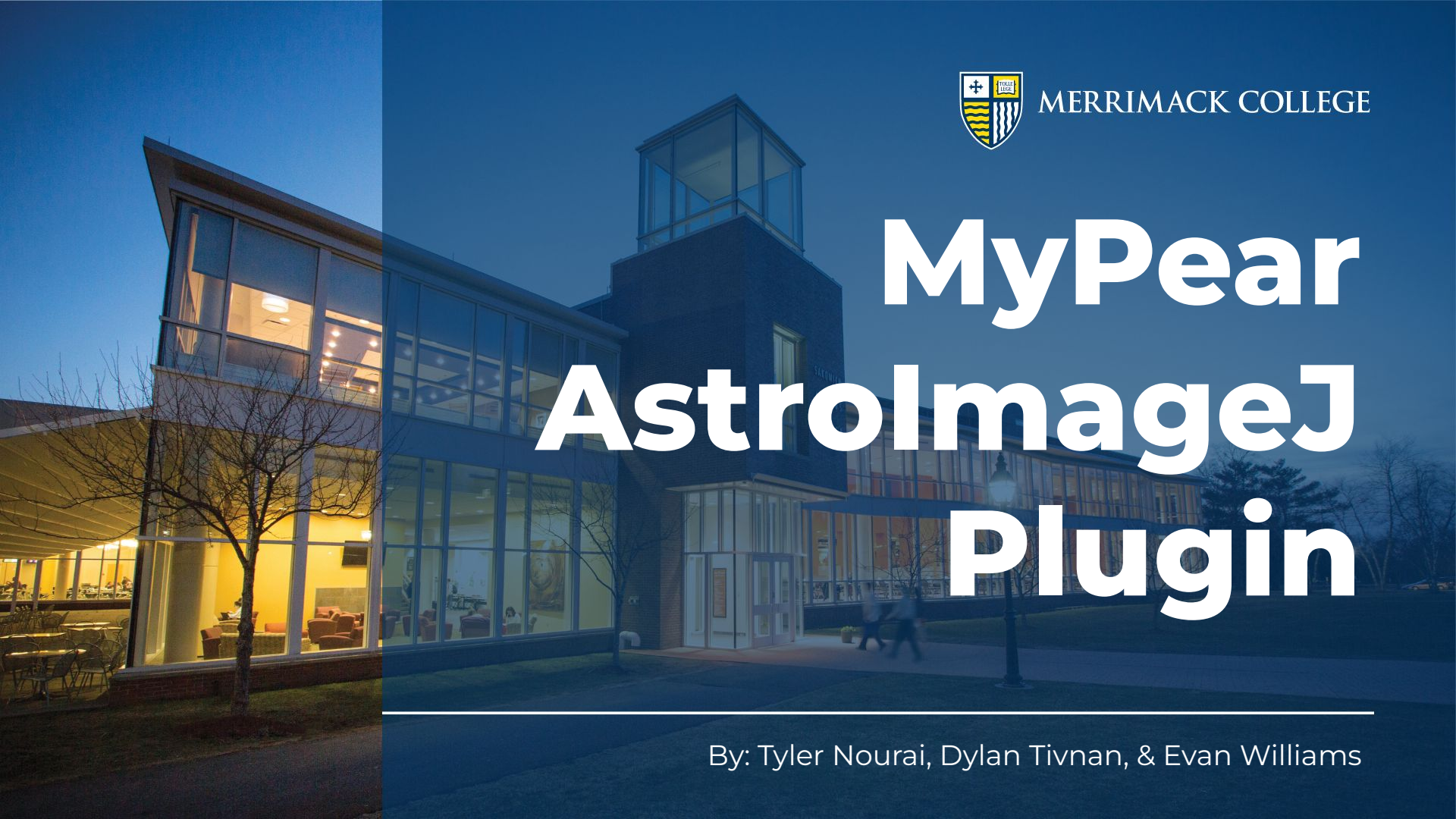


MERRIMACK COLLEGE

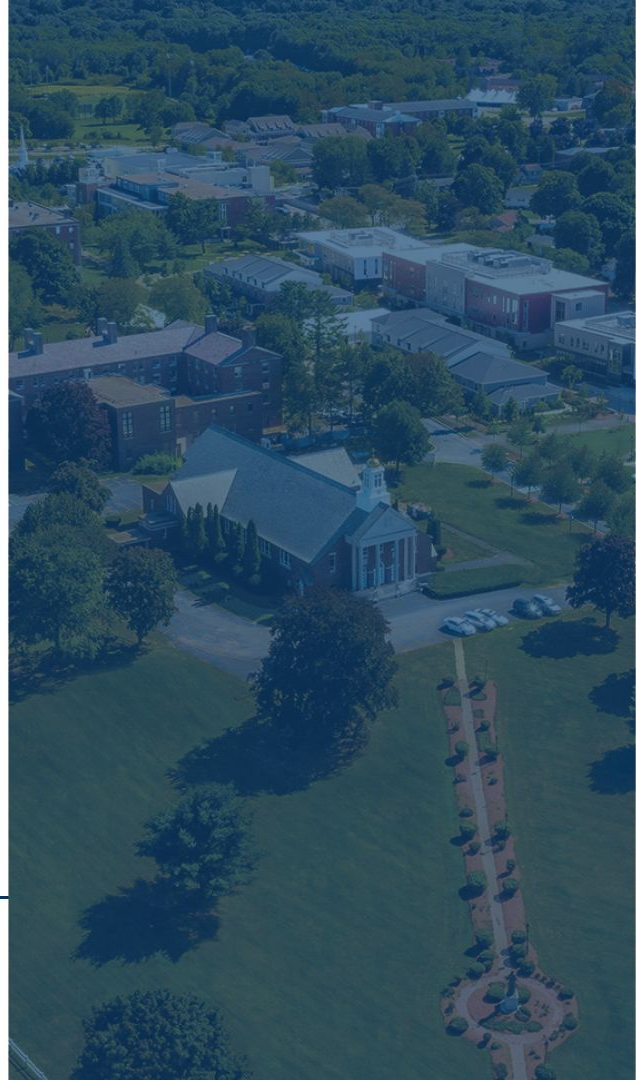
MyPear AstroImageJ Plugin

By: Tyler Nourai, Dylan Tivnan, & Evan Williams

Objectives

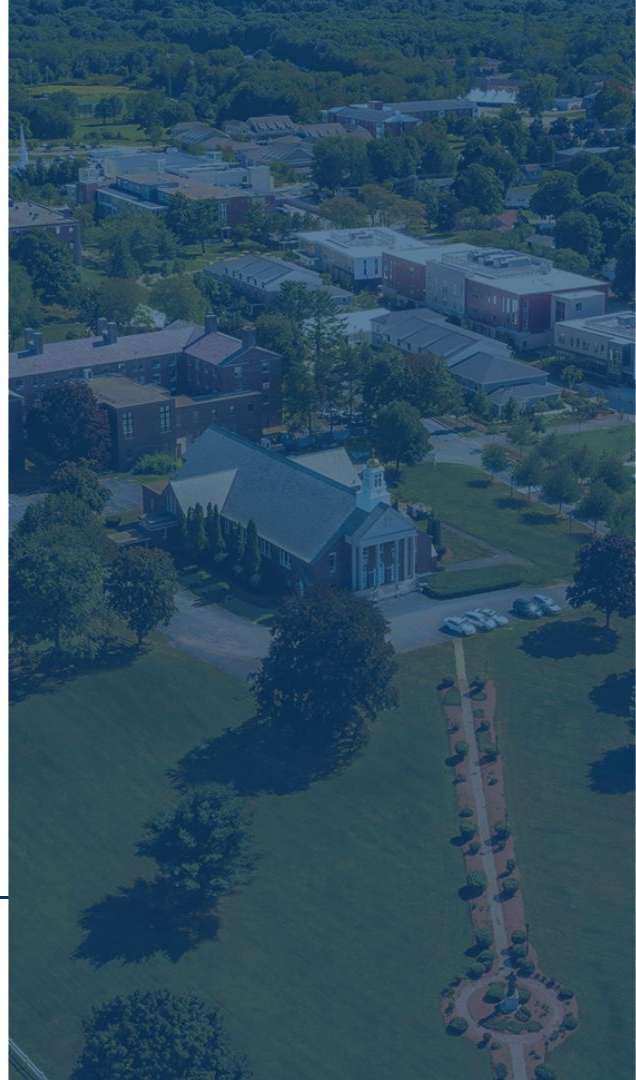
- Introducing MyPear Team
- Introducing the Client
- Team Qualifications
- Project Proposal
- Background on AstroImageJ and contour lines
- Other Available Solutions
- Project Management

Continued on next slide



Objectives (Continued)

- Design Choices
- Algorithm Implementation
- Design Diagram
- Initial and Technical Challenges
- Live Demo
- Download the plugin
- Future Work
- Conclusion
- Questions



Introducing MyPear Team

MyPear is a group of senior computer science students participating in the Computer Science Capstone course with **Professor Stueztle.**

The team consists of:

- Tyler Nourai - Team Manager, Spokesman, Developer
- Dylan Tivnan - Developer
- Evan Williams - Developer

Introducing the Client

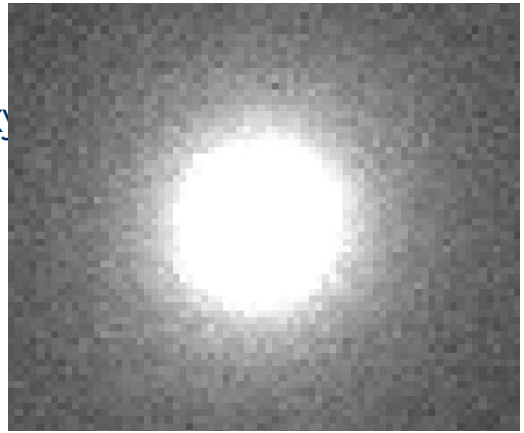
- The software client for MyPear Project is **Professor Christopher Duston.**
- Prof. Duston is a faculty member in the Physics department at Merrimack College.
- Member of the Merrimack College Astronomical Research Group (MCARG).
- The proposed plugin is of an immediate need to the research group.

Team Qualifications

- 3 ½ years of software development experience as Merrimack College Computer Science students.
- Experience with java programming.
- Experience with bitbucket and source code control.
- Experience working in teams, in CS courses such as Computer Graphics, Network Security, and Web Development.

Project Proposal

- Our Client and the MCARG have been taking pictures of the active galaxy M87.
- They want to measure how the shape of the galaxy has changed over time.
- The proposed project will help making the measurements of the objects.



Project Proposal (Continued)

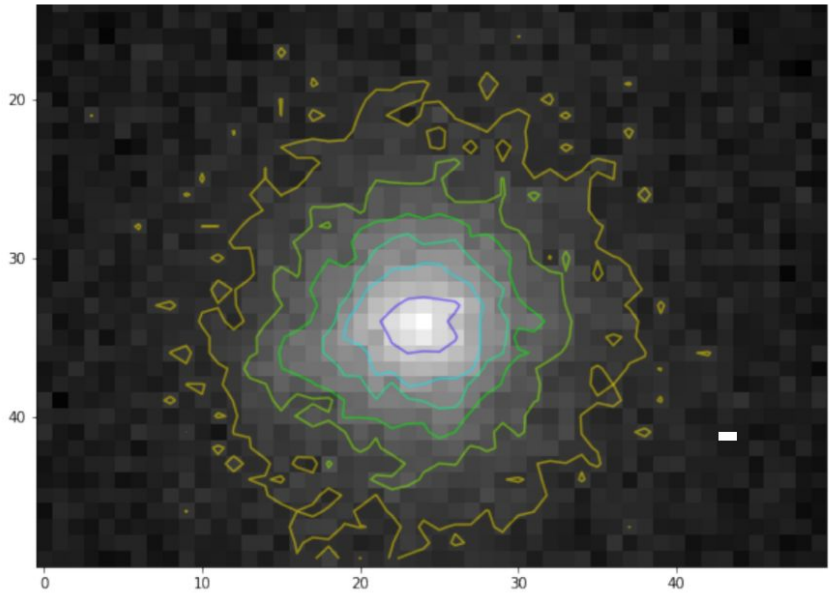
- The proposal was to develop an add-on plugin for an existing software called AstroImageJ.
- The plugin would be called Contour Lines and Area Measurement.
  - It would add contour lines to a selected star.
  - Then, it measures the area of each contour region.
  - And displays the resulting data.
  - User can save the data.

Background on AstroImageJ

- AstroImageJ, is built with ImageJ as its framework.
- ImageJ is an image processor application.
- It is a research-grade image analysis software.
- Built utilizing the Java programming language with GUI interface.
- It is a primary tool used by our client and MCARG.

Background on Contour Lines

- An outline, one representing or bounding the shape or form, of equal heights, in our project, pixels.
- In our clients case, contour lines and areas of its region would show a change in shape.

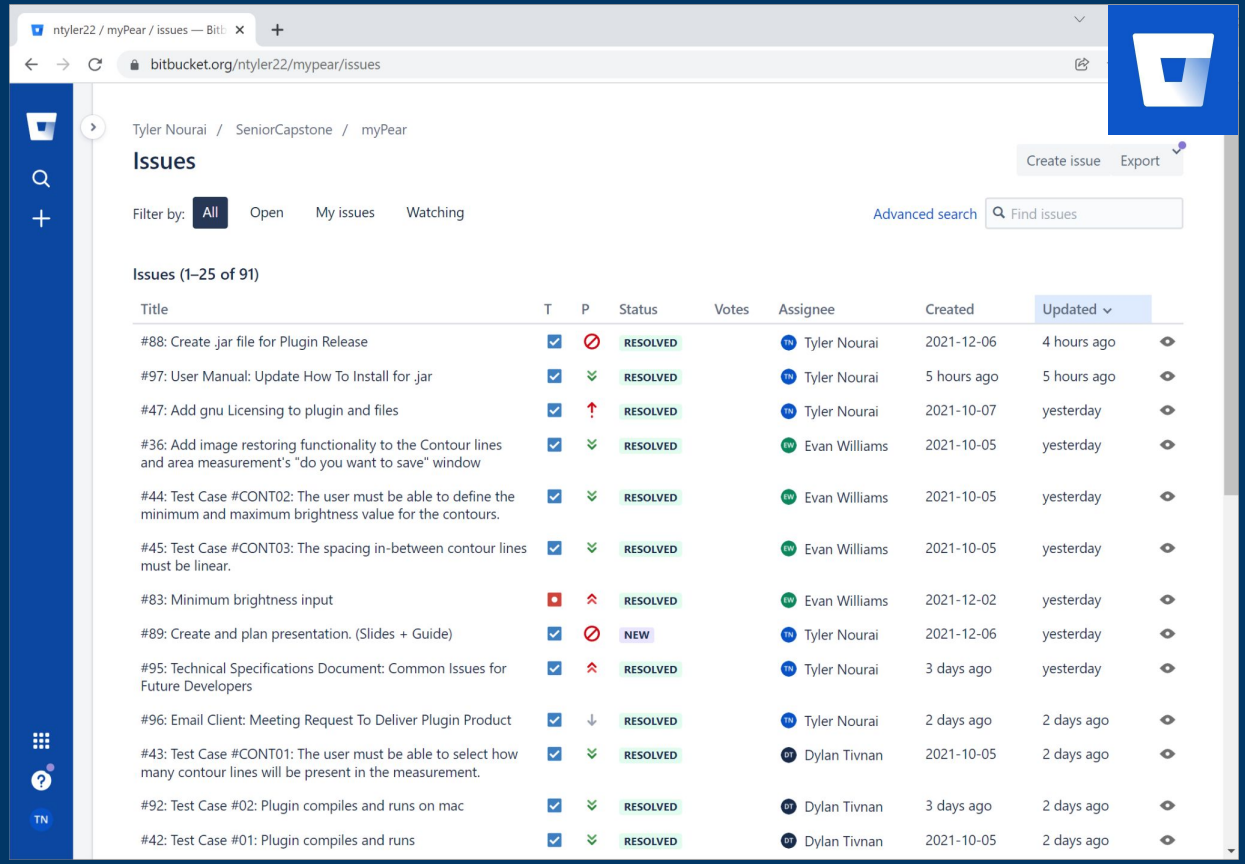


Other Solutions Available

- There is a plugin called Contour Plotter.
- And another plugin called Contour Lines.
- Both have stability issues, such as crashing.
- Neither had functionality to calculate areas of the region.

Project Management

- We used Bitbucket for managing, tracking changes in source code, and documenting the project as a whole.
- Including utilizing its task tracker.

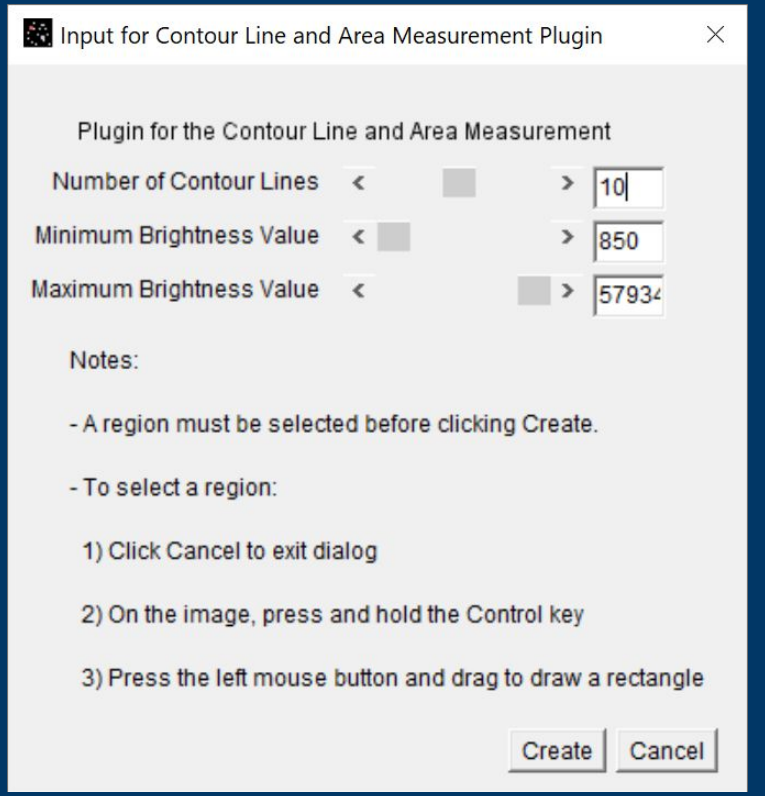


MERRIMACK COLLEGE

Design Choices

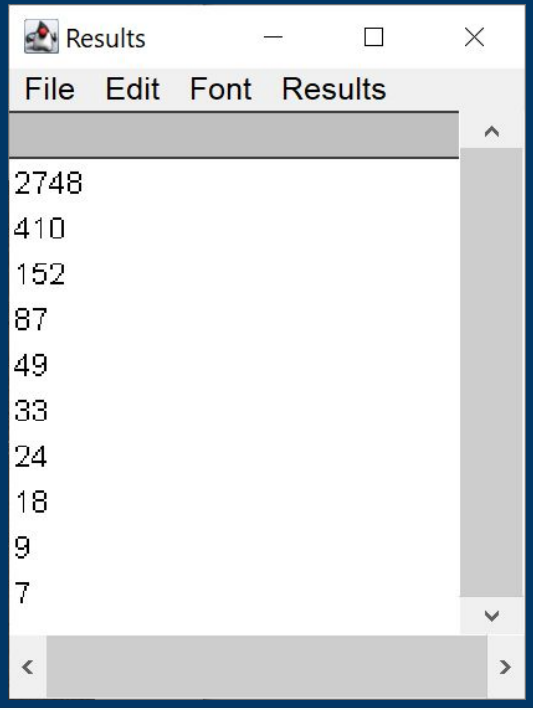
We designed our plugin software to seamlessly incorporate into AstroImageJ.

It consists of several GUI components for customization of settings to enhance the algorithm.



Design Choices (Continued)

- An additional feature that the client asked during our development, was to display the area data.
- Hence, we implemented the results functionality.

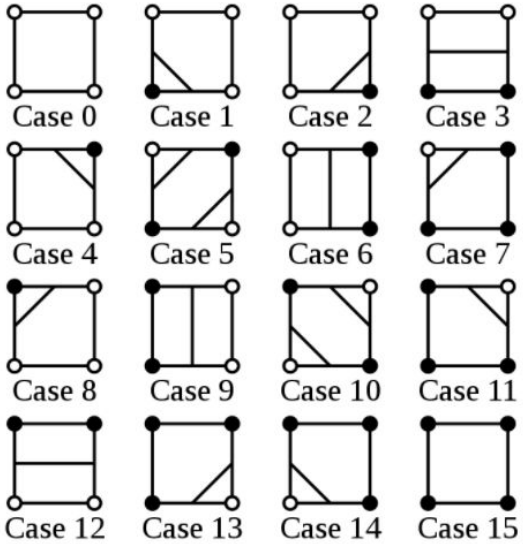


Algorithm Implemented

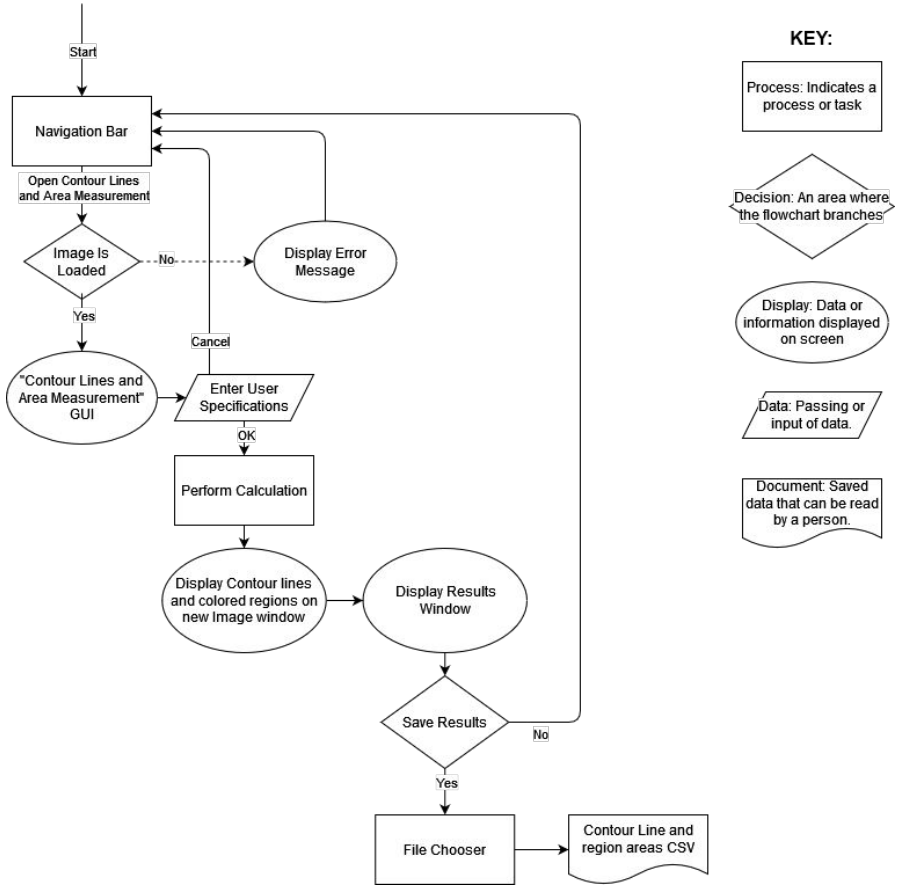
Marching squares algorithm was implemented to draw the contour lines on the image.

Divider: 200

100	200	100
200	300	200
100	200	100

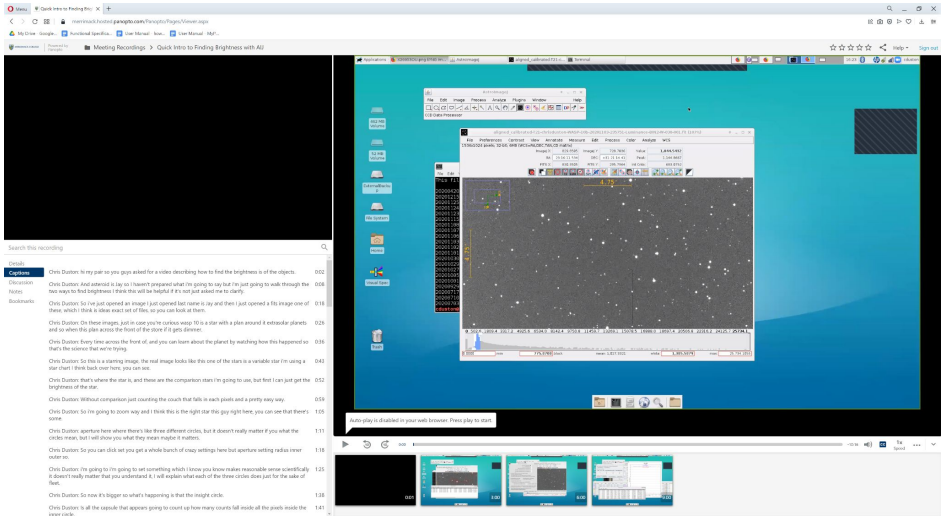


Design Diagram



Initial Challenges

- AstroImageJ software and astronomy was new to us.
- We did not know how the tool was used for the astronomy research.
- We had no sample images.
- Client helped out by providing video and test images.



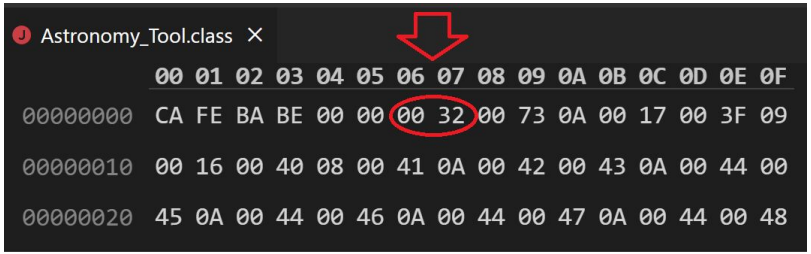
Technical Challenges

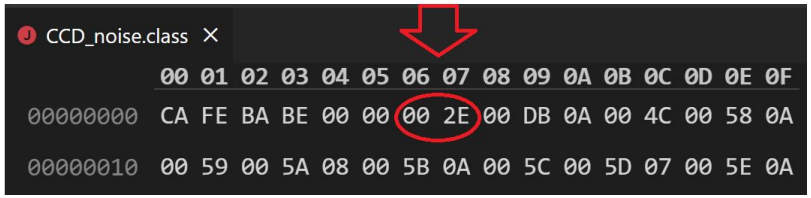
- AstroImageJ, is built off an older version of ImageJ.
- AstroImageJ and ImageJ uses various versions of Java 1.2 - Java 7.
- Current version of Java is 17.
- We developed our plugin using Java 7.

Version	Release date
JDK Beta	1995
JDK 1.0	January 1996
JDK 1.1	February 1997
J2SE 1.2	December 1998
J2SE 1.3	May 2000
J2SE 1.4	February 2002
J2SE 5.0	September 2004
Java SE 6	December 2006
Java SE 7	July 2011
⋮	
Java SE 17 (LTS)	September 2021

Technical Challenges

How we identified challenges of compiling errors:

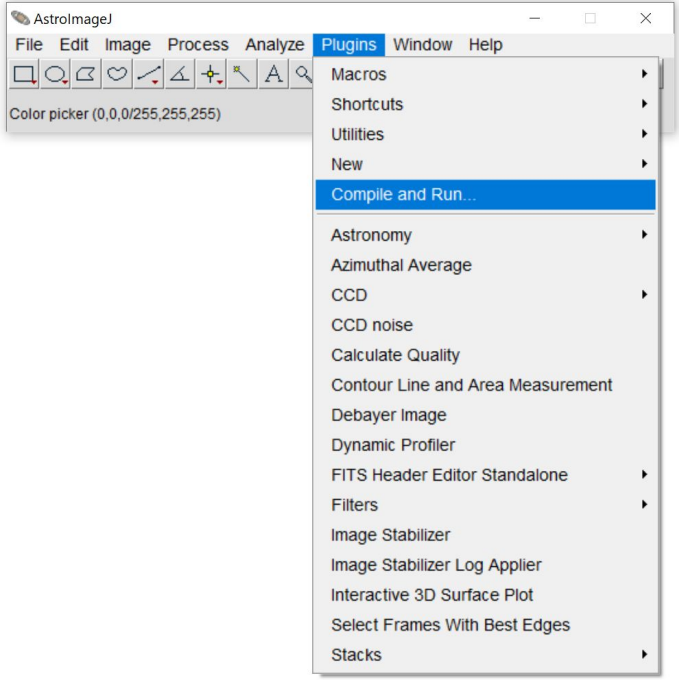




Java SE 17 = 61 (0x3D hex)
⋮
Java SE 8 = 52 (0x34 hex)
Java SE 7 = 51 (0x33 hex)
Java SE 6.0 = 50 (0x32 hex)
Java SE 5.0 = 49 (0x31 hex)
JDK 1.4 = 48 (0x30 hex)
JDK 1.3 = 47 (0x2F hex)
JDK 1.2 = 46 (0x2E hex)
JDK 1.1 = 45 (0x2D hex)

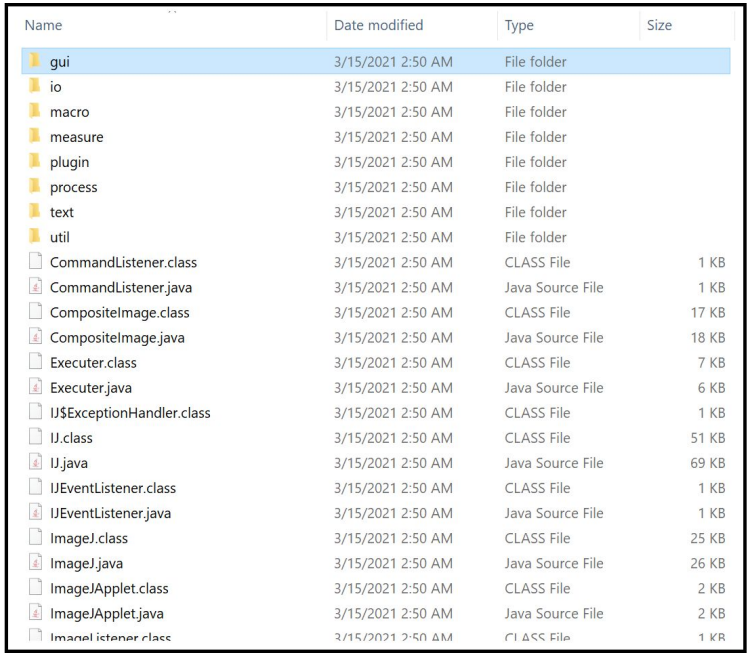
Technical Challenges

- The ImageJ built-in “Compile and Run” uses Java 1.2, and due to changes in Oracle’s licensing, it has not been updated.
- Since some AstroImageJ later components used Java 7, we decided to use that as well.
- As a result, the new plugin needed to be compiled outside of the AstroImageJ.
- This required JDK development environment to be setup with Java 7.



Technical Challenges

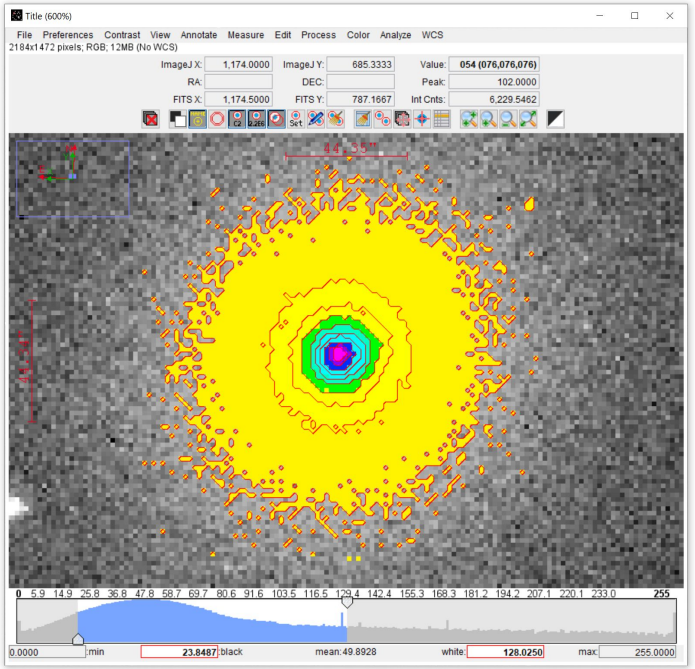
- We had to learn the existing libraries and capabilities of AstroImageJ.
- We built this into our development schedule.



Name	Date modified	Type	Size
gui	3/15/2021 2:50 AM	File folder	
io	3/15/2021 2:50 AM	File folder	
macro	3/15/2021 2:50 AM	File folder	
measure	3/15/2021 2:50 AM	File folder	
plugin	3/15/2021 2:50 AM	File folder	
process	3/15/2021 2:50 AM	File folder	
text	3/15/2021 2:50 AM	File folder	
util	3/15/2021 2:50 AM	File folder	
CommandListener.class	3/15/2021 2:50 AM	CLASS File	1 KB
CommandListener.java	3/15/2021 2:50 AM	Java Source File	1 KB
CompositeImage.class	3/15/2021 2:50 AM	CLASS File	17 KB
CompositeImage.java	3/15/2021 2:50 AM	Java Source File	18 KB
Executer.class	3/15/2021 2:50 AM	CLASS File	7 KB
Executer.java	3/15/2021 2:50 AM	Java Source File	6 KB
IJ$ExceptionHandler.class	3/15/2021 2:50 AM	CLASS File	1 KB
IJ.class	3/15/2021 2:50 AM	CLASS File	51 KB
IJ.java	3/15/2021 2:50 AM	Java Source File	69 KB
IJEventListener.class	3/15/2021 2:50 AM	CLASS File	1 KB
IJEventListener.java	3/15/2021 2:50 AM	Java Source File	1 KB
ImageJ.class	3/15/2021 2:50 AM	CLASS File	25 KB
ImageJ.java	3/15/2021 2:50 AM	Java Source File	26 KB
ImageJApplet.class	3/15/2021 2:50 AM	CLASS File	2 KB
ImageJApplet.java	3/15/2021 2:50 AM	Java Source File	2 KB
ImageListener.class	3/15/2021 2:50 AM	CLASS File	1 KB

Technical Challenges

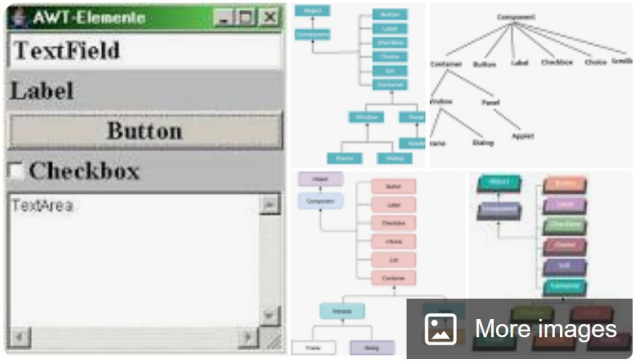
- The FITS images that we used were in greyscale, thus did not support color.
- We open a separate window with a new image that does support color.



Technical Challenges

- AstroImageJ uses Java AWT for its GUI user interface.
- Tooltips, for an example are not support.
- Java later versions moved to Swing and JavaFX, which not used in ImageJ.

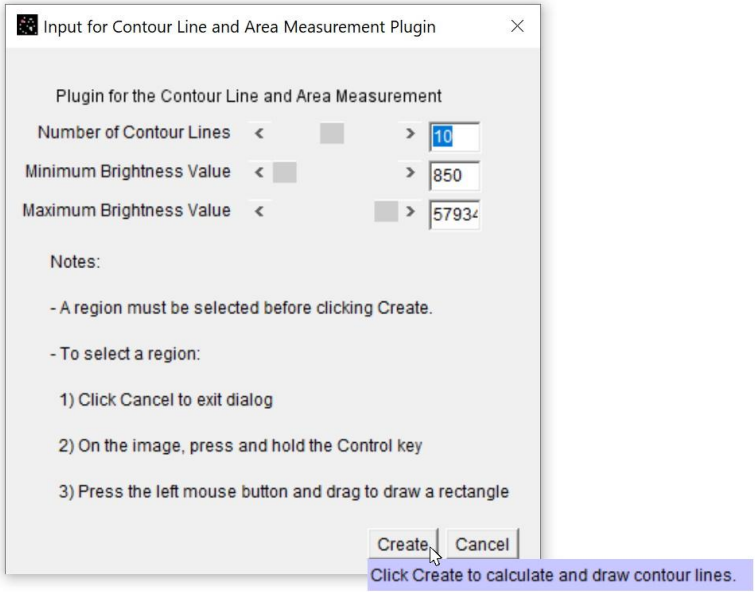
Abstract Window Toolkit



The Abstract Window Toolkit is Java's original platform-dependent windowing, graphics, and user-interface widget toolkit, preceding Swing. The AWT is part of the Java Foundation Classes — the standard API for providing a graphical user interface for a Java program. Wikipedia

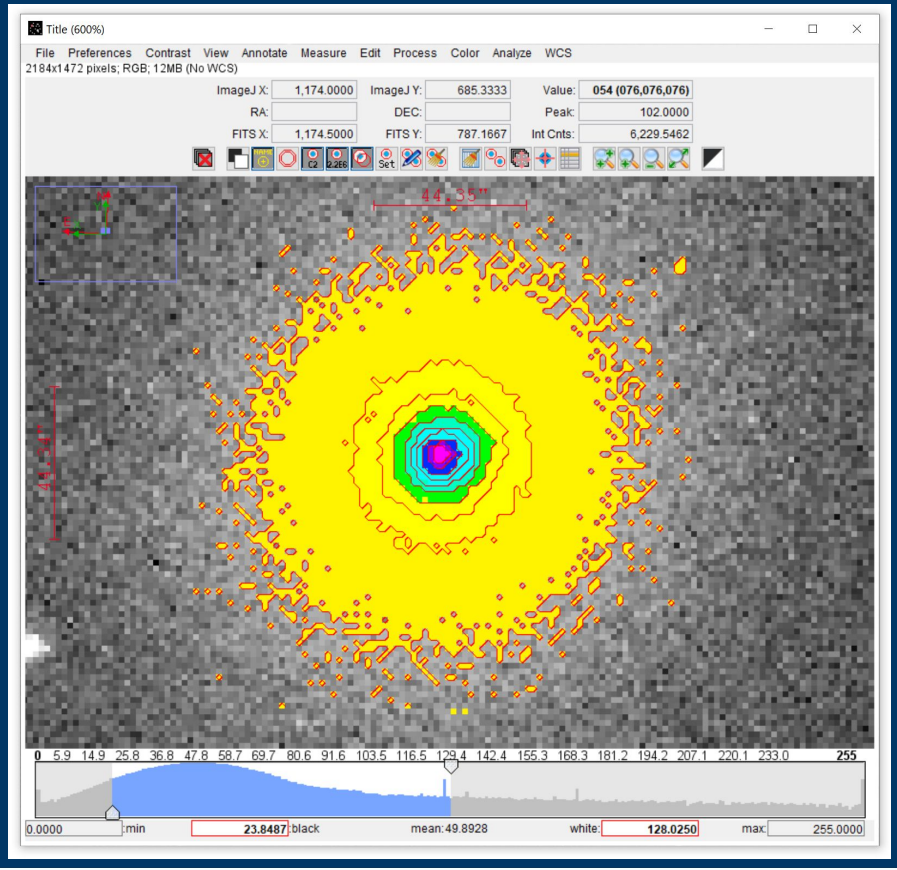
Technical Challenges

- Tooltips, is not supported for AWT.
- Researched how Tooltips can be added.
- We implemented the Tooltips functionality using an external example.



Plug-in in Action

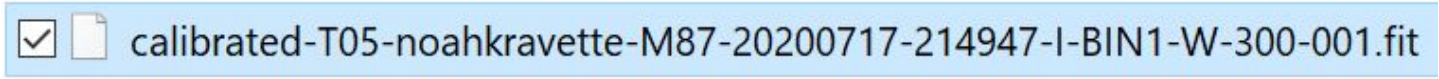
- This image is an example of a star with our plugin running to calculate and create contour lines.



MERRIMACK COLLEGE

Live Demo

Inputs:

- Image: 

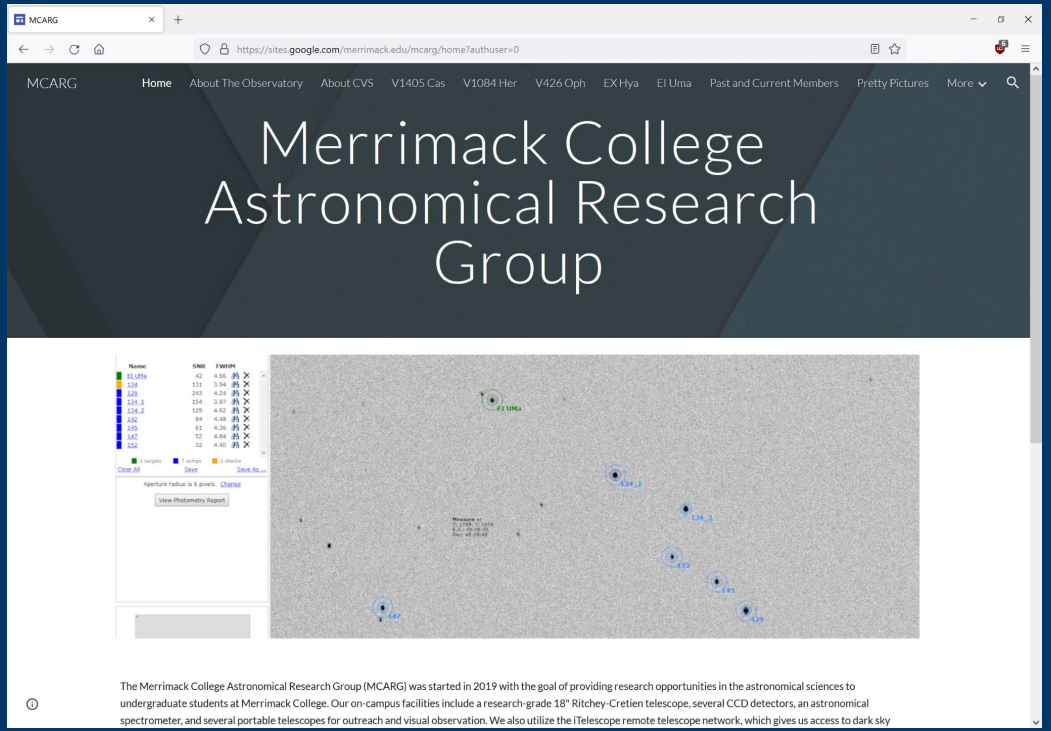
- Number of Contour Lines: **10**
- Minimum Brightness Value: **850 (Default)**
- Maximum Brightness Value: **3651**

Windows 10 laptop

MERRIMACK COLLEGE

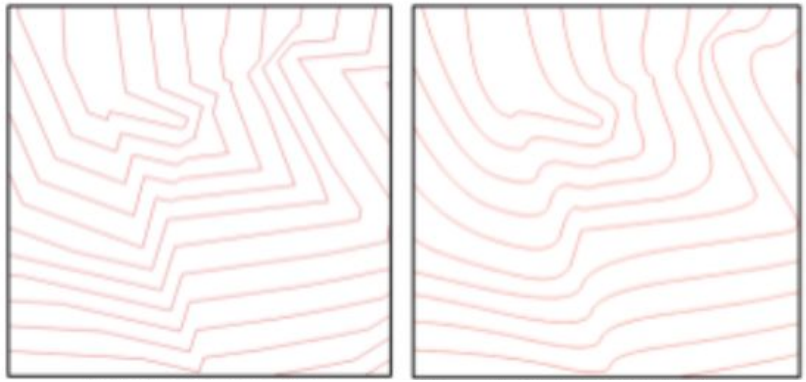
Download the plugin

- GNU Licensing
- Publicly available via download as a .jar file.
- Professor Dustons Github
- Merrimack College Astronomical Research Groups Site.



Future Work

- The contour lines are jagged and we would like to add a feature to smooth the contour lines.



Conclusion

- We overcame the challenges in this project.
- We learned how researchers work with astronomy software tools.
- We learned how to develop software with an existing software platform.

Conclusion (Continued)

- We learned the developmental process.
- We satisfied the project proposal specifications.
- We developed a functioning plugin that did not exist.
- We delivered the software to our client.

Thank You

- We would like to thank Professor Stueztle for his guidance, direction, and feedback.
- We would like to thank our client Professor Duston for the collaborative effort with MyPear.

Questions?

MERRIMACK COLLEGE