Acquiring, Processing, and Displaying JWST NIRCam Imaging Data

Maxwell Oakes
Portland State University
1900 SW 4th Ave, Portland, OR 97201
maxoakes@pdx.edu

Abstract

Images coming back from the James Webb Space Telescope (JWST) present a fascinating and awe-inspiring look at distant celestial objects and provide a glimpse at the early history of the universe. This paper provides a summary of steps taken to replicate the work done to process raw astronomical imaging data into the vivid RGB images that have been released by The National Aeronautics and Space Administration (NASA). Methods to acquire the image files via download will be addressed, as well as an overview of the FITS file format in which they are contained. The steps required to contextualize and realign the images will also be reviewed. A short description of any post-processing steps will be provided before a discussion of the results that were obtained. The processed images that have been released from JWST were placed into an already-existing processing pipeline managed by developers and graphics artists that have already spent much time in the field, so the results of this project will not be at the expert-level quality of those images, but the general process described may shed light on how steps of that process is done.

1. Introduction

The James Webb Space Telescope (JWST) was launched December 25th, 2021 as a successor to Hubble Space Telescope. The mission for JWST is to examine every phase of cosmic history; ranging from the first stars after the Big Bang to the formation of galaxies, stars and planets. Of the goals for JWST, there are four themes: (1) peer back in time with infrared imaging to see the formation of the first stars and galaxies, (2) use highly-sensitive infrared technology to view faint and early galaxies to help us understand how they are formed, (3) see through clouds of dust and gas to observe how planetary systems are born (4) study atmospheres of extrasolar planets in hopes of finding the building blocks of life elsewhere in the universe [1]. Altogether, these objectives aim to give humanity a better understanding of the universe and its formation, and help us better understand our place in the universe.

Being able to meaningfully process data from JWST will prove to be immensely important in achieving these objectives. Additionally, allowing the public audience to see the fruits of JWST will drive fascination, and perhaps garner support in its ultimate objective.

Currently, the publishing of images from JWST's NIRCam imager requires a mix of computer science and art to achieve eye-catching imagery, often requiring external photo editing software and subjective judgement to make the best decisions.

This project aims to streamline the post-processing step of NIRCam data by automatically downloading imaging data from Space Telescope Science Institute's (STSI) MAST public archive, select the best wavelength-filtered exposures and combine them into a single correctly-aligned, clean RGB color image.

2. Background Information

The following offers a short description of JWST and its instruments and capabilities, and gives a summary of the processing pipeline of imaging data, and concludes with the basic data structure that this imaging data comes in.

2.1. Onboard Instruments

In order to achieve these objectives, JWST is armed with several imaging and spectroscopy tools described below, and shown in Fig. 1:

MIRI Imager: A camera sensor able to capture between 5.6 and 25.5μm. It offers 9 broadband filters allowing the capture of smaller portions of this light bandwidth. The camera itself has a field of view of $74'' \times 113''$. It also features several dither patterns that improve sampling at the shorter wavelengths, and remove artifacts from the detector or cosmic ray hits

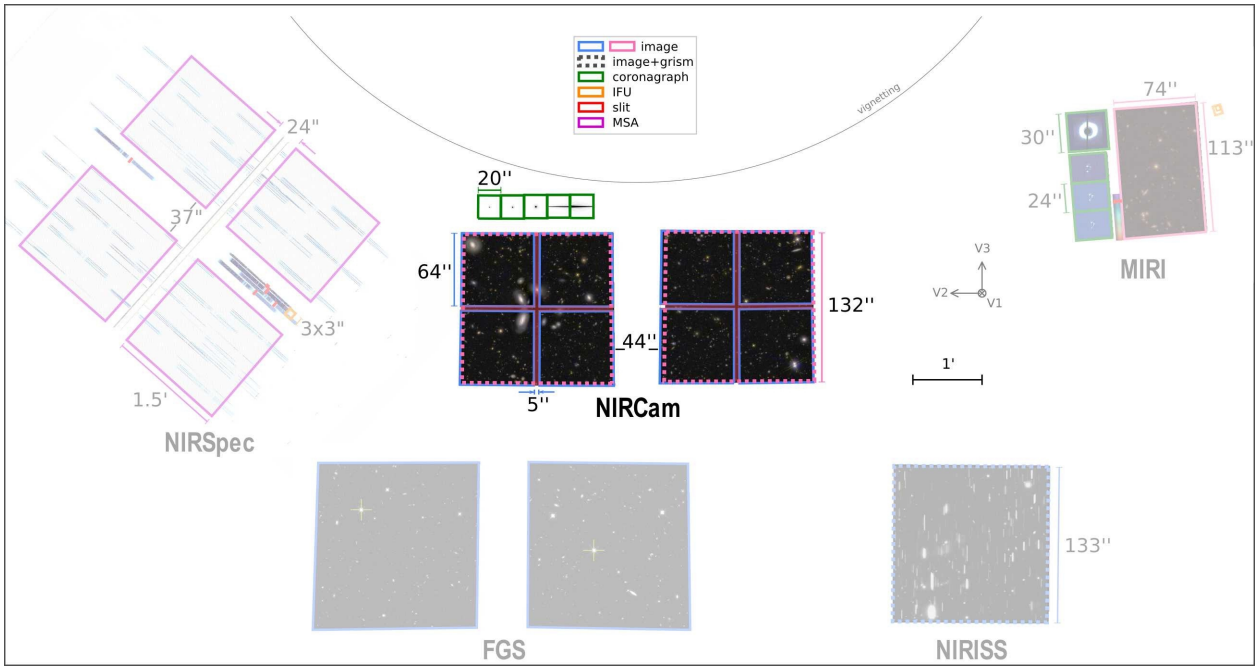

Figure 1: Placement of instrument array onboard JWST [4]. The instrument that is focused is NIRCam, the primary tool that will be used in the project presented in this paper.

[3].

NIRCam Infrared Camera: A tool that offers many observation modes from regular imaging to coronagraphic imaging, to wide field slitless spectroscopy. For this project, NIRCam will be the primary instrument that is studied and utilized. For imaging, NIRCam has two $2.2' \times 2.2'$ fields that cover 9.7 arcmin^2 with a total of 29 available wavelength filters. The coronagraphic imaging mode offers occlusion masks offer round bar-shaped occulting masks allowing for the capture of a star's corona without capturing the intense light of the star's center [4]. In some of the pre-processed images of this project, some of the larger stars will have occlusion masks featured in the image outputs.

NIRISS Imaging: An imager that enables capture of wavelengths between 0.8 and 5.0μm in a $2.2' \times 2.2'$ field of view. Like NIRCam, NIRISS offers spectroscopy, but it is more sensitive to low surface brightness between 0.8 and 2.5μm. NIRISS imaging is also an alternative to NIRCam in cases where the position of a target is not known with great accuracy [5].

NIRSPEC Spectroscopy: Provides near-IR spectroscopy between 0.6 and 5.3μm in a 3.4×3.6 arcmin field of view. It is designed to be especially powerful for multiplexing spectroscopy and high contrast high throughput single-object spectroscopy [6].

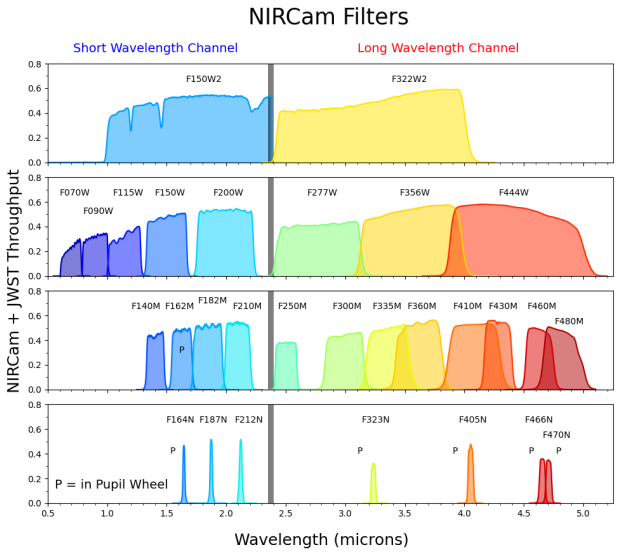

Figure 2: Available filters for the NIRCam instrument. Those marked with (P) are on the pupil wheel which require transmission through a second filter on the filter wheel [7].

2.2. Filters

JWST is able to capture between $0.6 - -28.8\mu m$ between all available tools. On each instrument, there are several filters that allow for capturing a smaller

bandwidth of that total. On NIRCam alone, there are 29 filters available: some for short wavelengths ($0.6 - 2.3\mu m$) and some for long wavelengths ($2.4 - -5.0\mu m$). Many of these filters can be used in combination of each other by means of a filter wheel and pupil wheel [7]. For this project, the pupil wheel will either be clear (i.e, no filter), or filtering for a wavelength, and the filter wheel will always be set to some wavelength.

2.3. JWST Data Processing

The processing of JWST goes through 3 stages. Stage 1 consists of detector-level corrections that are performed on a group-by-group basis. The output of this stage is a count rate image measuring the collection of photons. Stage 2 includes additional instrument-level corrections to produce fully calibrated image exposures. There are different pipelines for imaging and spectroscopic exposures.

Stage 3 consists of working of with multiple exposures and often consists of combining exposures in some way [8]. This is the stage that will primarily be used in the project described by this paper. Stage 3 processing adds corrections for astrometric alignment, background matching of the different mosaics, and outlier rejection [9]. The output of this stage, for NIRCam imaging specifically, are clean and calibrated images with metadata detailing photometry, physical positioning, timing, exposure, and much more.

2.4. Post-Processing by Astronomers

When the end goal is to out a color image from NIRCam, it is common that, after stage 3 data processing as previously described, several of the filter exposure images of a mission are compiled in image processing software like Photoshop [10]. At this point, it is up to the graphics artist to interpret the exposures and try to output the most visually striking combination of filters using only three color channels.

2.5. FITS File Format

FITS files are the only format that house JWST (and most astronomical) data. All FITS files are composed of header + data units (HDUs) as described below. A header unit consists of many records that make up something that resembles a standard dictionary data structure, but instead contains a key, value *and* a comment.

Alongside the header unit, an HDU can also contain an N-dimensional array, frequently a 1D spectrum, 2D image, or 3D data cube. The values in this array can be integers (signed, or unsigned for 8-bit), or floats of varying sizes [11]. This is the data unit of the HDU. Each FITS file can contain one or more of these HDUs. In the context of JWST stage 3 data used in this project, it is very common that a FITS file contains a primary HDU that contains no data unit and a header unit with upwards of 300+ entries with information about the mission, target object, and previous stage's processing steps. Alongside the primary HDU exists the "science" (SCI) HDU that contains the raw exposure data for the captured filtered wavelength. The accompanying table of cards is a lot shorter and contains information regarding the applied filter(s), exposure time, spacecraft orientation, spatial extent, as well as other metadata pertinent to the capture of the image. There are several other HDUs in the FITS file, most of which measure uncertainty and variance for each pixel in the SCI HDU data unit [12]. The remaining HDUs are not relevant to processing of the image data in this project.

2.6. MAST

The Mikulski Archive for Space Telescopes (MAST) is the sole repository that this project uses, and likely the only option to reliably retrieve JWST imaging data. Alongside JWST, MAST includes a wealth of datasets organized by "collections", from many missions, including "HST, Kepler, GALEX, EUVE, FUSE, IUE, SWIFT, and several other missions." [13]. For each entry in MAST, there are several dozen metadata entries describing the image/data target, associated mission, time of capture, and more. For image and spectroscopy data, most data also comes with a preview image that comes in a widely viewable format like JPG.

2.7. Astropy and Python

For this project, Python was used with several libraries including math and image processing libraries like NumPy, scikit-image, SciPy, and PIL. For astronomy related information processing, AstroPy, AstroQuery, and AstroAlign were also used.

3. Methodology

The general goal of this project was to acquire JWST imaging data automatically via script, process it in some way so that it becomes usable and recognizable to the viewer, then combine the all associated data in such a way that the output is a full color RGB image that is of the same quality as those images released by the JWST team. This section will describe each step in my script that ultimately completes that goal.

3.1. Data Acquisition

The script that I created has two options when starting the script. Choosing the argument *query* will start the Query mode to allow the user to find a mission, and choosing the *run* argument will allow the user to input a unique ID to automatically start the entire acquisition and processing pipeline.

Query Mode: On start, the user will be prompted with three search parameters: Name of the object, title of the mission, and proposal ID. Each of these prompts allow the user to narrow down search results of the future query output. If nothing is entered, all data will be returned.

However, by default, there are several parameters that are applied that cannot be changed so as to use only certain data: JWST data, publicly available data, science (as opposed to measurement or calibration), NIRCam images, and stage 3 data. Any additional parameters that the user entered are appended to this default list of parameters.

A query is then made to MAST using the aforementioned search parameters and a resulting CSV is written to the disk in an appropriate sub directory.

The user will then find the data that they seek and identify the proposal ID. The proposal ID is unique to each mission, and is short.

Run Mode: Upon starting run mode, the user will be prompted to enter a proposal ID. Upon entering the ID, MAST will be queried again, but this time to download all files relevant to that proposal's mission, and that conform to the aforementioned default search parameters.

The files will be downloaded to their own dedicated "mission" subdirectory. The FITS files for the mission will be sent to *./missions/¡target name¿/data* and the preview of each FITS file will be sent to *./missions/¡target name¿/preview*. Any future script executions of this same proposal ID will first check that the data already exists, and use existing data if possible. For the FITS files of the dataset used for this project, the total size on the disk was 2.27 GB.

Internally, after the files are either downloaded or read from the disk, the SCI HDU data unit image will be extracted and placed into an object that will also store metadata that is highly relevant to the image such as pixel size, spacecraft location and orientation, exposure time, and filters used. In case there needs to be any processing of the image, this data can be readily accessed rather than opening the associated FITS file each time the information is needed. At this point, the images are ready to be processed.

3.2. Managing the Pixel Values

The resulting image data has a very high dynamic range: from just below zero (due to stage 1 or 2 calibration) all the way to 25,000 in one image. If these images were to be scaled down to between 0 and 255, only a spec of a star's corona would show up in the center of a black image. A solution to allow detail to show up when scaled down would be to clip the values and ignore those that are simply too bright. I found that taking the 99.9% highest value pixel as the new upper limit suffices (as well as taking the lower limit at 0). For all images in this project, the resulting highest values were in the double digits. When scaled to between 0 and 255, the detail is now very clear, and the clipping of star's extreme brightness is not noticeable. This clipping is applied to all images.

Now that each image is of reasonable dynamic range, the next obvious issue is that each image has a different resolution. Additionally, not all images capture the same exact spatial extent in space. This is because each image is taken with a different filter, and each image is exposed for several minutes. Because there is movement onboard the telescope to swap filters with the filter wheel, the spacecraft's orientation is not consistent across all captures.

3.3. Image Alignment

My initial approach was to assume the images do indeed capture the same spatial extent. Therefore, the thing needed was to identify a "center" (available in the header unit of each image), scale the images so each nominal pixel area is the same across all images, then pad each image to they are indeed the same exact resolution. The result of this was was a failure on two fronts. First, while each image does come with a center X and center Y value, these values do not correspond to the same point in space, or are they the exact center of the image. So, for the purposes of alignment, that statistic is useless. Second, each image does not cover the exact same spatial area, as explained previously. A new method was required to align the images in order to proceed.

I found the AstroAlign Python library does exactly what I needed. The primary alignment function required a "target image" and a "source image", where the source would be aligned to the target image. This function finds stars in the image and attempts to match the stars of the target image with the stars of the source image and derive a transformation that is then applied to the source image. The source image is then automatically cropped and padded to match the size of the target image.

After arbitrarily choosing a target image, every

other image is aligned with the target. However, sometimes alignment with this method fails. If there is a failure, sensitivity and iteration values are adjusted and the alignment is retried. At some point, each image will either find an alignment, or fail once it is determined that alignment can be found. For the mission used in this project using only one target image, there are a resulting 4 images that share the same alignment: one target, and three successful alignments.

3.4. Adding Color

Now that the images are aligned, the coloring process can start. My initial approach was to assign each filter to a color. This mapping can be seen as shifting infrared wavelengths downward to the visible light spectrum such that the longest infrared will be mapped to red, and the shortest infrared wavelengths will be mapped to magenta.

I created an array structure that would hold RGB values from 0 to 1 for several general bold colors sorted by wavelength: violet (1, 0, 0.5), magenta (1, 0, 1), blue (0, 0, 1) green (0, 1, 0), yellow (1, 1, 0), orange (1, 0.5, 0), red (1, 0, 0) to name a few. I then sorted the images based on their filter name (which is also in a format that matches their wavelength).

An RGB image is now made from each mono-color image where each channel is multiplied by the weights in the previously described color list. There are now 4 images that are colored from black to a color hue.

A second approach is to utilize the color assignment that the JWST team used. As it turns out, the color coding of filters in Fig. 2 was carefully chosen, as it matches the idea that I originally had, but each filter has a corresponding color. I transcribed the colors chosen in the chart to a dictionary of filter names and RGB values, then simply created a color-hue image using those colors for each filter.

3.5. Displaying the Image

The final step was to combine the images in some way. After some experimenting with different methods, such add, blend, overlay, I found that the lighten operation yields the best results. Starting with a black image, the following algorithm is run: for each channel of each image, take the maximum of the current image channel and the existing image's channel.

Finally, the resulting image is displayed, as well as saved to disk in 8-bit RGB format. Each major step's image result is also written to the disk in their own directories.

Figure 3: Final unedited output of my script. The color-to-filter mappings are: Blue = F090W; Cyan = F187N; Yellow = F356W; Red = F405N.

4. Experiments

Initial results of the script are promising, although there is certainly more that can be done, which will be addressed in the Future Works section.

4.1. Success

The final output of the script using the AstroAlign alignment method and the JWST team's color assignments yields an image (Fig. 3) that is similar to the one that was publicly released of the same nebula target object (Fig. 5). Immediately, it is easy to see some key differences. First, the outward orangish explosion is not visible in my image. This is because most of that color was in an image that was thrown out due to lack of usable alignment. Second, the blue color of the center of the nebula is not as vivid. There are two reasons for this: the green color image was also thrown out due to lack of usable alignment, and second, the publicly released image also goes through some post-processing to bring out color, simply for aesthetics.

4.2. Failures

Before the final result in (Fig. 3), there were some failed attempts at alignment and coloring. As shown in Fig. 4, the shift in each color filter color channel is obvious. Since there are more color channels in use, the overall color looks different than my final correctly

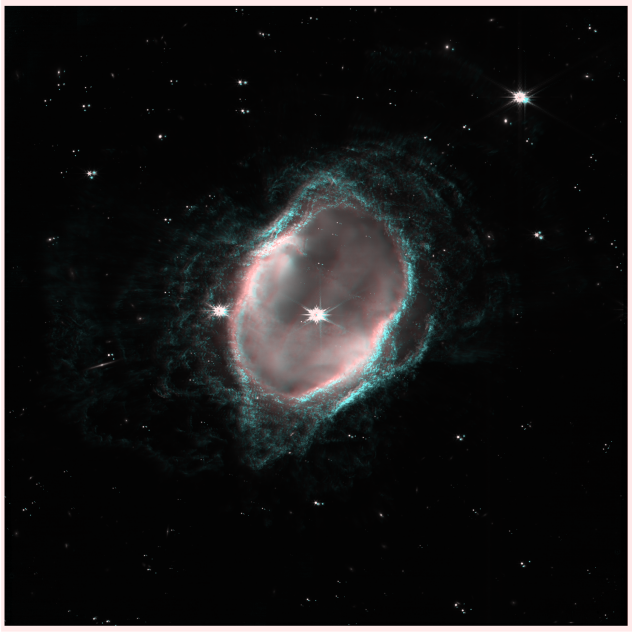

Figure 4: Misaligned image with all available color filter images of my script. The color-to-filter mappings are: Blue = F090W; Cyan = F187N; Green = F212N; Yellow = F356W; Red = F405N; Red = F470N.

Figure 5: Full color release image of NGC 3132 provided by NASA. The color-to-filter mappings are: Blue = F090W; Cyan = F187N; Green = F212N; Yellow = F356W; Red = F405N; Red = F470N.[14]

aligned one, despite using the same color assignment.

5. Discussion

Overall, this script and project was successful to an extent, but there is still a lot more that can be added to this script, and that will be the majority of the discussion section

5.1. Future Work

Each aspect of the script could be improved. If I had more time on this project, I would be able to attempt or complete some of the improvements.

Firstly, a new feature that I did not attempt so far is a way to fill in the occluded spots. As mentioned when describing NIRCam, there is a way to occlude stars during image exposure. In the final color image result, a cyan circle and small black dot are visible in the center of the brightest stars in the image. During each exposure, the brightest parts are occluded, so when the images are combined, some channels are missing the full color of stars. This can be fixed by either flood-filling the 0-value pixels inside of a very high-value "pit" of pixel values, or waiting until the final color image is created before simply assigning the center of the stars to a full white color. Both of these approaches would require a method to find each occluded area, but the first would require this happening for each filter image, even if some are thrown out.

A second improvement that can be made is the alignment. In the current script, the best alignment is done via AstroAlign. However, with this method, some of the images get dropped since no alignment can be found.

However, it might be possible to use that same function and achieve alignment with all images. If we were to choose a target image, we could build of images that align with that one. Then, we could try those images that were aligned, and attempt to align those that failed previously. This method should work if the images failed to align due to a poor selection of target image. However, the primary downside to this method is that the runtime would be $O(n^n)$, where n is the number of filter images. Since each n can already take between 2-10 seconds, this alternative approach would significantly increase the runtime.

There are also other novel algorithms for alignment that could be investigated and implemented. However, one avenue that I have not fully explored is figuring out fully what each piece of image metadata is used for. For example, in each image, there is a spatial extent value. Paired with the spacecraft orientation and other values, a way to derive a transformation matrix may exist using already-known information.

Another area that needs some exploring is the color assignment process. For the final image of this project,

Figure 6: The original aligned image plus minor color curve enhancement using paint.NET.

I used NASA's assignment. Since the source images are of wavelengths that are not visible to humans, the color selection is somewhat arbitrary. Realistically, I could develop my own equally-as-detailed color assignment that features more vivid and varied colors.

Finally, as mentioned previously, the images released by NASA go through some post-processing in Photoshop to touchup and enhance the images for aesthetic purposes. Again, since these images are outside of our perception, it would not be unrealistic to programmatically enhance the final image result. A nice addition to this script would be a simple color curve adjustment. A simple curve adjustment done manually in a photo editor is visible in Fig. 6. Even a simple operation such as this could improve the aesthetics of the image, even with two missing filter images.

LaTeX [?] is a set of macros built atop TeX [?].

References

[1] *https* : *//www.nasa.gov/mission*$_p$*ages/webb/science/index.html* 1

[2] *https* : *//archive.stsci.edu/missions* − *and* − *data/jwst*

[3] *https* : *//jwst* − *docs.stsci.edu/jwst* − *mid* − *infrared* − *instrument/miri* − *observing* − *modes/miri* − *imaging* 2

[4] *https* : *//jwst* − *docs.stsci.edu/jwst* − *near* − *infrared* − *camera* 2

[5] *https* : *//jwst* − *docs.stsci.edu/jwst* − *near* − *infrared* − *imager* − *and* − *slitless* − *spectrograph/niriss* − *observing* − *modes/niriss* − *imaging* 2

[6] *https* : *//jwst* − *docs.stsci.edu/jwst* − *near* − *infrared* − *spectrograph* 2

[7] *https* : *//jwst* − *docs.stsci.edu/jwst* − *near* − *infrared* − *camera/nircam* − *instrumentation/nircam* − *filters* 2, 3

[8] *https* : *//jwst* − *pipeline.readthedocs.io/en/stable/jwst/pipeline/main.html* 3

[9] *https* : *//jwst* − *pipeline.readthedocs.io/en/stable/jwst/pipeline/calwebb*$_i$*mage3*. *image3* 3

[10] *https* : *//www.space.com/james* − *webb* − *space* − *telescope* − *image* − *editing* 3

[11] *https* : *//www.loc.gov/preservation/digital/formats/fdd/fdd0* 3

[12] *https* : *//jwst* − *pipeline.readthedocs.io/en/stable/jwst/data*$_p$*roducts/science*$_p$*ro* 2 − *d* − *data* − *i2d* − *and* − *s2d* 3

[13] *https* : *//mast.stsci.edu/portal/Mashup/Clients/Mast/Portal.* 3

[14] *https* : *//webbtelescope.org/contents/news* − *releases/2022/news* − 2022 − 033 6