Applied Science and Innovative Research
ISSN 2474-4972 (Print) ISSN 2474-4980 (Online)
Vol. 6, No. 3, 2022
www.scholink.org/ojs/index.php/asir

Original Paper

Understanding Black Hole Imaging Based on Very Large Baseline Interferometry (VLBI)

Yufan (Jerry) Chen[1*]

[1] Brooks School, North Andover, USA

[*] Yufan Chen, Brooks School, North Andover, USA

Received: May 28, 2022 Accepted: June 9, 2022 Online Published: June 15, 2022

doi:10.22158/asir.v6n3p21 URL: http://doi.org/10.22158/asir.v6n3p21

Abstract

In 2019, the Event Horizon Telescope made history when it captured the first known image of a blackhole. The black hole, situated in the center of the Messier 87 galaxy, is more than 55 million light-years away from earth, and was only able to be captured using Very Large Baseline Interferometry (VLBI) technology and the computational imaging technologies integrated into the Event Horizon Telescope array (The Event Horizon Telescope Collaboration, 2019). Astronomical image processing and interferometry requires the collection of radio waves using a radio telescope and analyzing that data with software. VLBI collects signals with multiple telescopes simultaneously, and the resulting data can be reduced and analyzed as data collected by a telescope with the diameter equal to the largest distance between the telescopes, thus is capable of producing an image with higher angular resolution and capturing objects further away from earth. However, due to the distance between the VLBI telescopes, algorithms are needed to fill in the hole within the collected data and reduce atmospheric noise and delays in signals (Very Long Baseline Array (VLBA)). We conduct an in depth review of the algorithm and VLBI as a whole in this paper, and hope to use our findings to further push the development of this great technology.

Keywords

astrophysics, imaging, interferometry, astrophotography

1. Introduction

Very-long-baseline interferometry is a method used in astronomical imaging using radio waves. To create an image of astronomical objects that are of significant distance and maintain angular resolution, it is necessary to have a radio telescope with a large diameter, since the diffraction limit of smaller telescopes means there would be less angular resolution. Thus, when graphing incredibly far away objects like a

black hole, the size of the telescope required becomes unrealistic to maintain angular resolution. VLBI can be used to avoid this issue. VLBI collects data using multiple radio telescopes located at different regions of the planet. Due to the difference in location between the radio telescopes, there is a difference in time delay in receiving the signals. In extended emissions, the time difference is equivalent to the sinusoidal variation on the emission's intensity distribution, as theorized by the Cittert-Zernike Theorem (Bouman, Katherine, Johnson, Michael, Zoran, Daniel, Fish, Vincent, Doeleman, Sheperd, & Freeman, William, 2016), and we can calculate using equation

$$\Gamma_{i,j}(u,v) \approx \int_{\ell} \int_{m} e^{-i2\pi(u\ell+vm)} I_{\lambda}(\ell,m) d\ell dm$$

Where $I_{\lambda}(\ell,m)$ is the emission of wavelength from direction λ and coordinates $(\ell,m,\sqrt{1-\ell^2-m^2})$

Figure 1. Telescope Array at the Smithsonian Submillimeter Array, Hawaii, which Used VLBI Technology to help Capture a Picture of the Black Hole with the Event Horizon Project

After VLBI data is received, the collected data, either on a magnetic film or via fiber-optic cables, is synchronized at a central location. Due to the characteristics of VLBI, every antenna is a different distance away from the radio source, and each will have a different time delay in receiving the signal. After the synchronisation, 2 error factors are added to account for the time delay. The first error is a Gaussian noise that corrects the true visibility of the image, utilizing the noise model of radio interferometry given by Cornell and Wilkinson, $V_{jk} = V\hat{}_{jk}(1+a_j)(1+a_k)\,exp\,[i(\varphi_j-\varphi_k)] + \epsilon_{jk}$, where a_j is the zero-mean Gaussian amplitude error, φ_j is the phase error, ϵ_{jk} is the thermal noise (Bouman, Katherine, Johnson, Michael, Zoran, Daniel, Fish, Vincent, Doeleman, Sheperd, & Freeman, William, 2016). Ignoring atmospheric noise and focusing solely on the phase and thermal noise, getting the equation $V_{jk} = V\hat{}_{jk}\,exp\,[i(\varphi_j-\varphi_k)] + \epsilon_{jk}$. The second error is a bispectrum noise that takes the

triple product of the 3 ideal visibilities, which are ideal fourier components of the source image at the respective coordinates calculated by their respective measured fourier components with $\Gamma_{i,j}^{meas}\Gamma_{j,k}^{meas}\Gamma_{k,i}^{meas} = e^{i(\varphi i-\varphi j)}\Gamma_{i,j}^{ideal}e^{i(\varphi j-\varphi k)}\Gamma_{j,k}^{ideal}e^{i(\varphi k-\varphi i)}\Gamma_{k,i}^{ideal} = \Gamma_{i,j}^{ideal}\Gamma_{j,k}^{ideak}\Gamma_{k,i}^{ideal}$, and then multiplies the noise on one bispectrum (Bouman, Katherine, Johnson, Michael, Zoran, Daniel, Fish, Vincent, Doeleman, Sheperd, & Freeman, William, 2016). Together, these 2 error factors approximate the gain error distribution of the image prior. With these 2 error factors added and synchronised, the data produced can now be used and calculated as data collected with a single telescope. Using data created with this method, we can simply use astronomical interferometry, superimposing the radio waves to extract visual signals from the data, to picture far away stars and other astronomical objects, such as the black hole. The Event Horizon Telescope, which uses VLBI data collected from 8 radio telescope arrays (Figure 1) around the globe, successfully captured an image of the black hole in 2019 (Figure 2). In creating this image, the EHT team employed a special algorithm to combat the issues such as delays caused by atmospheric noise, which significantly decreases accuracy of the images. This algorithm is one that we will examine further in the following sections, and try to improve upon to retain even more information.

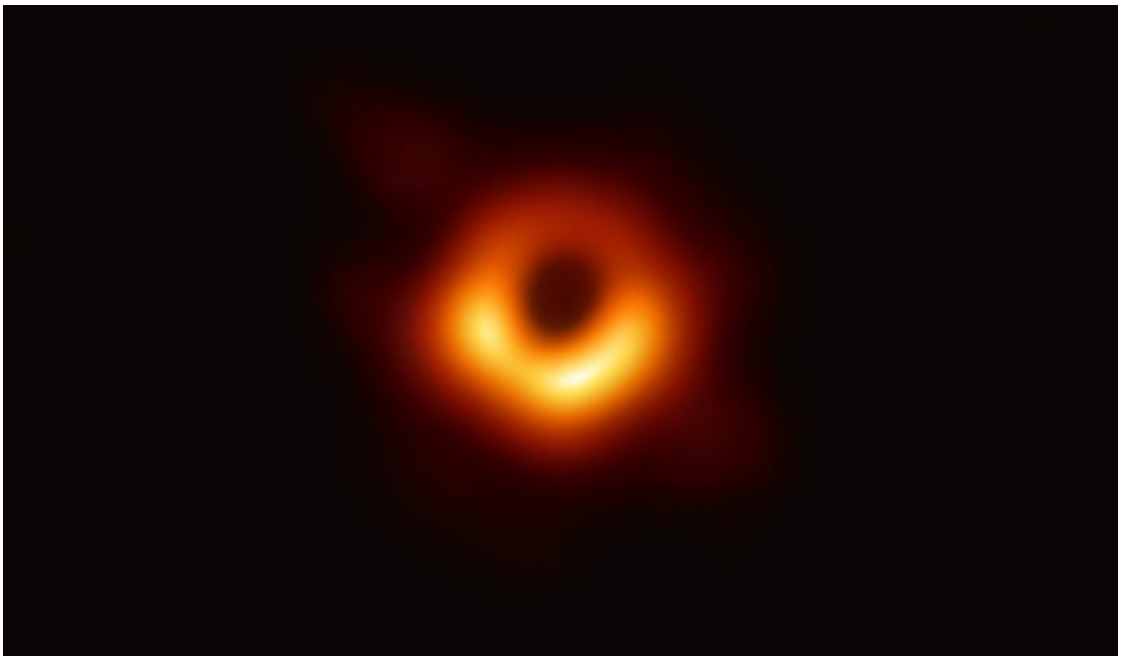

Figure 2. Image of the Messier 87 Blackhole Captured by the Event Horizon Telescope with VLBI Technology and CHIRP Algorithm

2. Method

To better understand the way EHT creates images with VLBI and its algorithms, we examine some sample data and simulate the image creation with Python and the EHTIM package (Bouman, Katherine, Johnson, Michael, Zoran, Daniel, Fish, Vincent, Doeleman, Sheperd, & Freeman, William, 2016), which

allows us to simulate and manipulate VLBI data. The EHTIM plug-in utilizes the pyNFFT module to complete Fourier transformations and deconstruct the frequencies of light emitted. The EHTIM package synthesizes the VLBI data given custom targets and reconstructs images of the targets from the VLBI data.

The function in EHTIM that visualizes the imaged target is called im_display. In the im_display function, the program reads the .txt file data using the load_text and load_array_text function, the EHT algorithm visualizes a rough image of the photographed object (Figure 3).

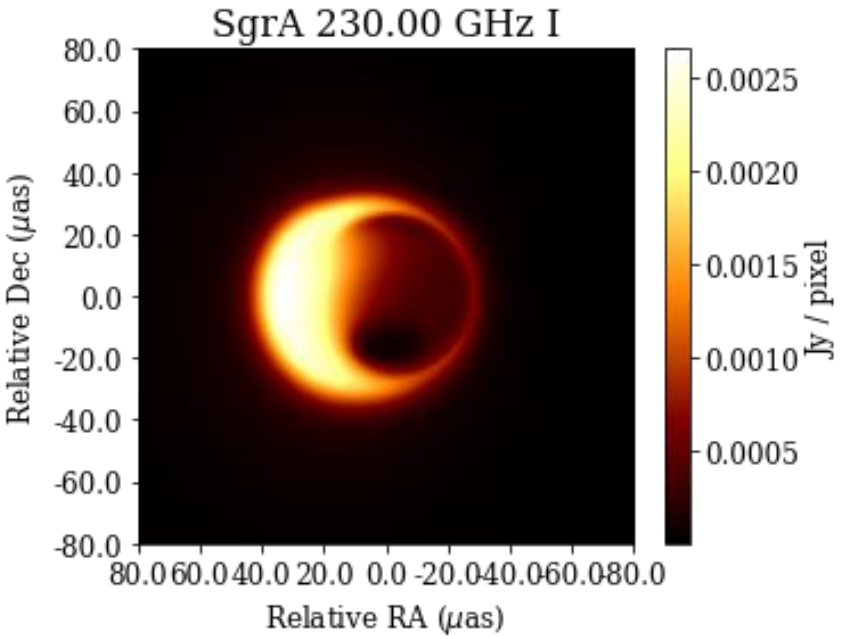

Figure 3. Initial Image Generated by the Im_display Function, in Terms of Right Ascension and Declinition. The Color Bar on the Right is on a Scale of Jansky/pixel, or Unit Radiation Density per Pixel

The function im_observe realistically synthesizes the VLBI measurements. There are several functions and parameters that we can manipulate with the simulated data, which will change different aspects of the resulting image.

The im_observe function allows us to observe the image given some key observation specifications, such as integration time, advance time and bandwidth. Changing parameters sgrscat (Figure 4), phasecal (Figure 5) and ampcal (Figure 6) will alter the image by adding blur with the Sgr A* kernel, gaussian errors and station-based random phases. These can help with the accuracy of the image based on the consideration of image frequency, gain variations, and phase errors. In addition, we can use function obs.deblur (Figure 1) to reduce the blur through the dividing blur by scattering kernels (Bouman, Katherine, Johnson, Michael, Zoran, Daniel, Fish, Vincent, Doeleman, Sheperd, & Freeman, William, 2016).

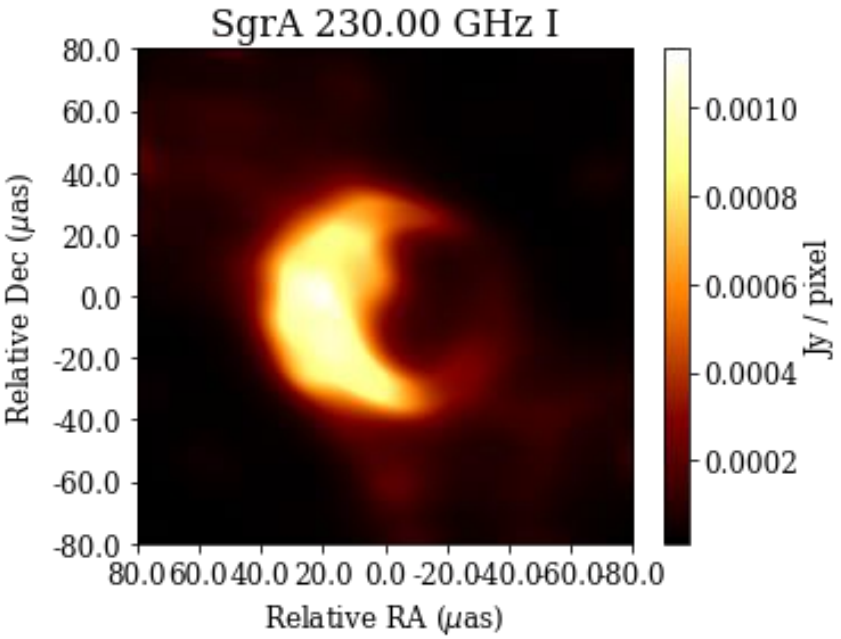

Figure 4. Image Generated after Altering the Sgrscat Parameter to False, which Adds a Blur to the Image Based on the Radio Wavelength of Sagittarius A (Sgr A)

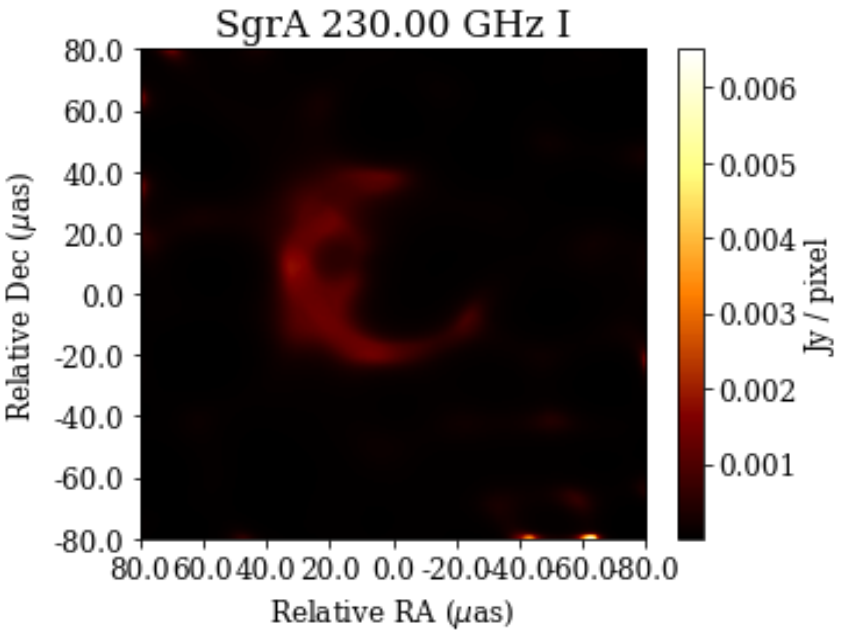

Figure 5. Image Generated after Altering the Phasecal Parameter to False, which Removes the Station-based Random Phases and Removes Correction towards Atmospherical Interference

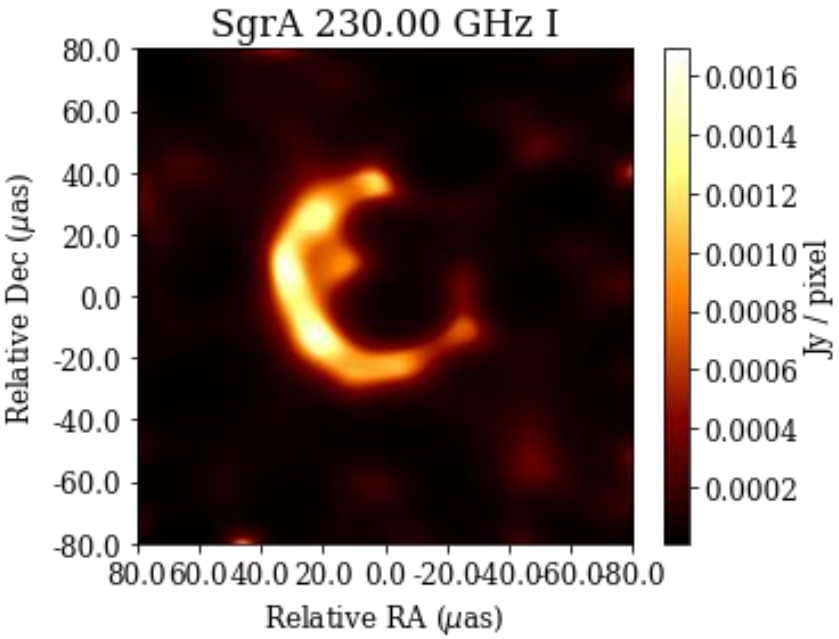

Figure 6. Image Generated after Altering the Ampcal Parameter to False, which Removes the Station-based Gaussian Error and Removes the Correction towards Gain Corruption

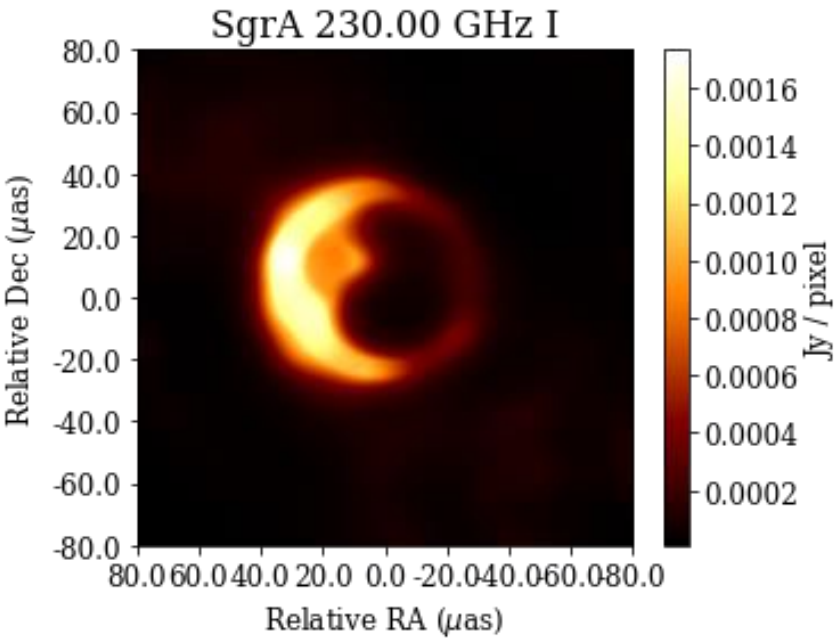

Figure 7. Image Generated after the Obs.deblur Function was Applied

In the next functions, nominal array resolution and beam parameters are specified. Nominal array resolution is the desired angular resolution in terms of the maximum achievable resolution. Beam parameters denote the gaussian beam's properties, based on wavelength, radius and refracture. This is defined by parameters pixel size (npix) and field of view (fov) (Figure 10).

Using functions obs.dirtyimage, obs.dirtybeam, and obs.cleanbeam, combined with the adjustments made with parameters npix and fov, we can observe the image in different states: as a dirty image, with a dirty beam or a clean beam. A clean beam image (Figure 7) is one that has been run through the CLEAN restoring beam algorithm which suppresses high spatial frequencies, and the dirty image (Figure 8) is one that has not been filtered, causing measurements to appear inaccurate. at different fields of view. These images can then be displayed to examine the different results.

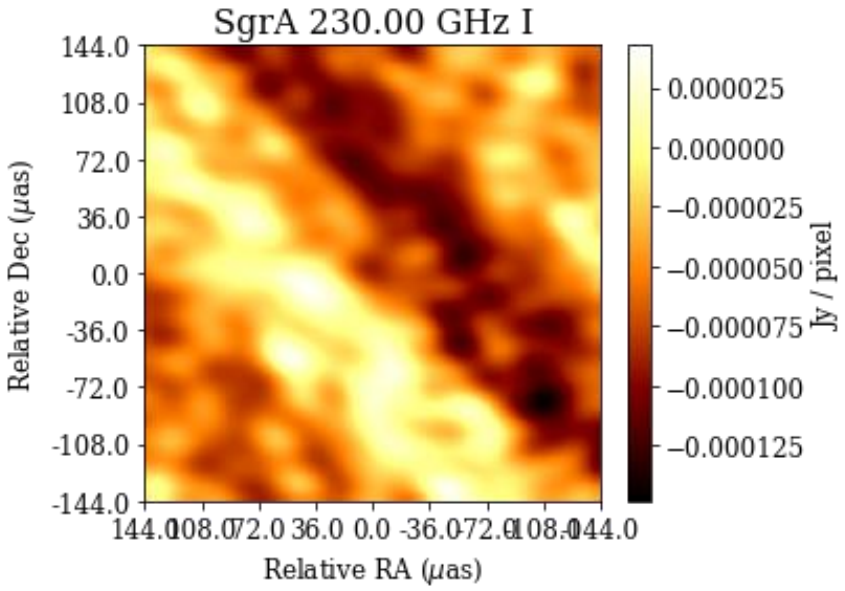

Figure 8. Image Generated under the Dirty Image Conditions and a Widened FOV

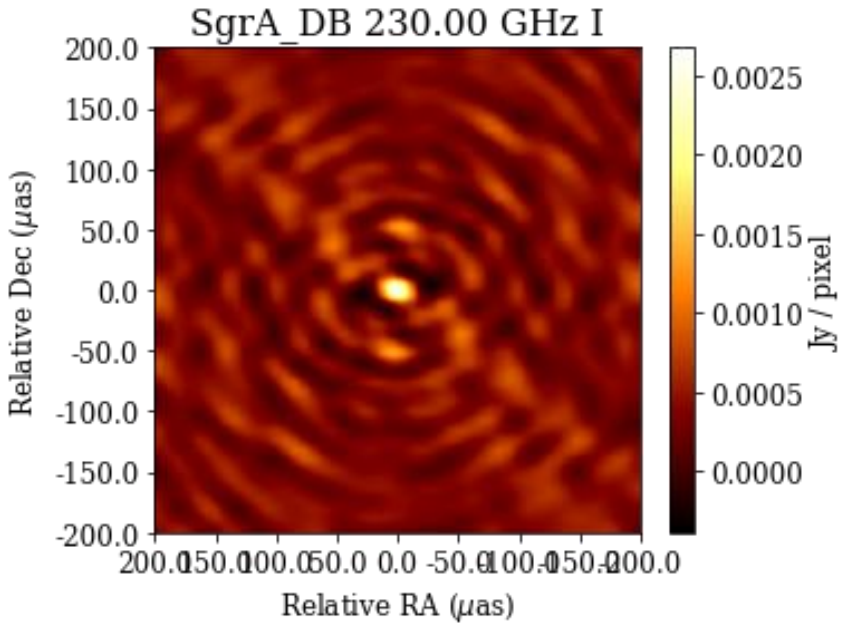

Figure 9. Image Generated under the Dirty Beam Conditions and a Widened FOV

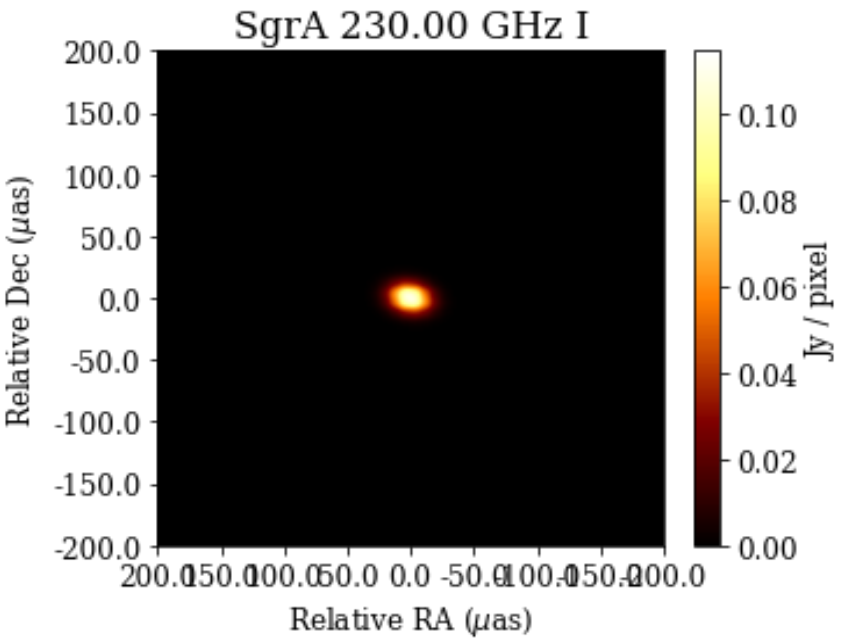

Figure 10. Image Generated under the Clean Beam Conditions and a Widened FOV

There are a few more functions that each tweak specific features of the photo. eh.image.make_square clears all preceding inputs, emptyprior.add_flat flattens (or combines) all prior inputs, and emptyprior.add_gauss adds the appropriate gaussian blur to the prior inputs.

3. Result

As we are looking for the maximum a posteriori estimation for image coefficients x and y, we are essentially looking at an optimization problem of the energy function, given by

fr(x|y) = −D(y|x) − EPLLr(x),

where EPLLr is the log-likelihood of the specific patch. This then makes it similar to a Bayesian posterior probability of the prior term. D is the data term that is explained below. This is the loss function within our method, and the basis of our improvements.

Since there are an infinite amount of possibilities for the image generated from existing data, the challenge is in fact finding an explanation for the data that matches our current assumptions of the visuals. The current CHIRP algorithm parameterizes a continuous image, which allows us to consider the image as, in fact, a continuous image, instead of assuming a discretized image of point sources, which can introduce errors in the recovered image $I_\lambda(\ell, m)$. Considering the parametrization of coordinates ℓ and m as $N_\ell \times N_m$scaled pulsed functions, and applying a Discrete Time Fourier Transformation then allows us to create a more accurate representation of the data.

To optimize the image generated, we use the “half quadratic splitting” technique, which uses sets of auxiliary patches to overlay the individual patches p1x in the image. We then solve for {Zn} given x, doing so by setting {Zn} to the most likely patch under the prior given the corrupted value of PnX and

considering weighing. We then solve for x given {Zn}, under the knowledge that our energy is a 6th order polynomial as we used bispectrum measurements. We perform a 2nd order Taylor expansion around x0 to approximate the local minimum in closed-form. We will iterate between these 2 steps giving 10 weighing values, which allows us to build up a multi-scale framework. The building of the framework is essential in helping us find the best low-resolution solution of the image, starting at pulses 20x20, and over the different iterations increase the scale to 64x64.

Before the final image is generated with the out.display function, we can modify all aspects of the image with the imager_func function. The imager_func function contains most of the changeable parameters and is responsible for the generation of the final image. We input the original picture, InitIm, and the prior image generated, Prior, along with the object VLBI data, along with desired data string and regulazier, and a final image from all the data is generated.

With the function we are able to select which data string the image will be generated with, as the function allows the storage and generation of three separate sets of variable term values, and we can select them here by choosing between d1, d2, and d3. These data terms each refer to the value D within the loss function term -D(y|x), which can be further split into d1(x,y)+d2(x,y)+d3(x,y), and we can choose different statuses for these values, from 'vis', 'bs', 'amp', 'cphase', 'cphase_diag' 'camp', 'logcamp', 'logcamp_diag'. Similarly, we can also select between 3 regularizers, which are sets of data that are used to prevent overfitting of neural networks in the algorithm, mainly aimed at improving accuracy of the image generated. In our case, we incorporate a Gaussian mixture model EPLLr(x) to regularize our equation using the matrix Pn, taken from the nth patch of our coordinate x. We can also combine data strings and regularizers by changing their corresponding alpha values, which changes the weight of each string.

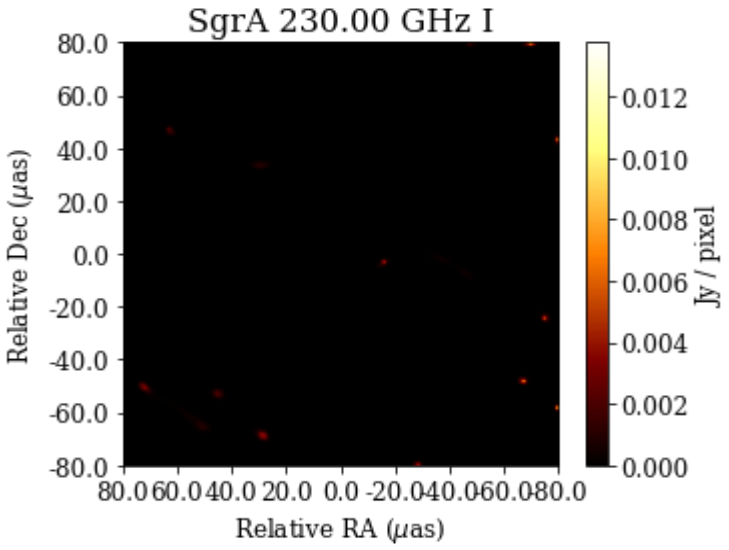

Figure 11. An Image Generated with the d2 Data String, still Set at the bs Setting

The function will then call several parameters, both booleans and strings. grads is a boolean that applies analytics gradients when true; norm_reg and norm_init are booleans that normalizes the regularizer and initial image when true. Boolean debias apply debiasing to the amplitude of the image. There are also other string data in the function: weighing is a data string that lets you choose between natural or uniform weighing of the data, beam_size is a parameter that changes the radians for the regularizer normalization. max_set is the boolean that defines the closure quantities, and cp_uv_min then flags all baselines that are shorter than the defined value before forming the closure quantities. ttype, fft_pad_factor, and order all change the ways fourier transformation is conducted with FFT. There are also parameters that don't change the function itself, but change the way the function is displayed, such as show_updates, which display the progress of the minimizer.

The algorithm will then apply a circular gaussian blur with the out.blur_circ function, and apply the imager_func once again. Optionally, the image can be self-calibrated with the sc.self_cal function, before the imager_func is applied once again.

Finally, a final blur is applied with the out.blur_gauss function, and the final image is generated with all the parameters specified. This is done by the out.display function.

In examining the loss function, and more specifically the EPLLr(x) regularizer, we notice several interesting functions. We find that within the imager_func function, the regularizers are written as:

```
def objfunc(imvec):
        if logim:
imvec = np.exp(imvec)
datterm = alpha_d1 * (chisq1(imvec) - 1) + alpha_d2 * \
            (chisq2(imvec) - 1) + alpha_d3 * (chisq3(imvec) - 1)
regterm = alpha_s1 * reg1(imvec) + alpha_s2 * reg2(imvec) + alpha_s3 * reg3(imvec)
conterm = alpha_flux * flux_constraint(imvec) + alpha_cm * cm_constraint(imvec)
        return datterm + regterm + conterm
```

The three regularizers and their weighting is then calculated by

```
datterm = alpha_d1 * chisq1grad(imvec) + alpha_d2 * \
            chisq2grad(imvec) + alpha_d3 * chisq3grad(imvec)
regterm = alpha_s1 * reg1grad(imvec) + alpha_s2 * \
            reg2grad(imvec) + alpha_s3 * reg3grad(imvec)
conterm = alpha_flux * flux_constraint_grad(imvec) + alpha_cm * cm_constraint_grad(imvec)
        grad = datterm + regterm + conterm
```

With these regularizers in place, and combined with the steps mentioned above, a final image is generated (Bouman, Katherine, Johnson, Michael, Zoran, Daniel, Fish, Vincent, Doeleman, Sheperd, & Freeman, William, 2016).

4. Discussion

The methods used in VLBI are extremely complicated, with functions and data that can turn a txt file full of empty data into a completed image. This paper dives into the methods of VLBI, and tries to understand the workings behind it. While advanced, there are certainly possible improvements for VLBI technology. It is very possible to add more regularizer terms, selecting even more niche portions of the data that will produce clearer and more precise images.

Acknowledgement

I gratefully acknowledge the support of Qi Guo from Purdue University for his guidance with the research and writing of this essay.

References

Bouman, Katherine, L., Johnson, Michael, D., Zoran, Daniel, Fish, Vincent, L., Doeleman, Sheperd, S., & Freeman, William, T. (2016). Computational Imaging for VLBI Image Reconstruction. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Bouman, Katherine, L., Johnson, Michael, D., Zoran, Daniel, Fish, Vincent, L., Doeleman, Sheperd, S., & Freeman, William, T. (2016). Computational Imaging for VLBI Image Reconstruction Supplemental Materials. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

The Event Horizon Telescope Collaboration (2019, April 10). First M87 Event Horizon Telescope Results. I. The Shadow of the Supermassive Black Hole. *The Astrophysical Journal Letters*, *875*(1), L1.

Very Long Baseline Array (VLBA). National Radio Astronomy Observatory.