Imaging Pipeline Software

Bobbie Ware 17962233

Math705 Research Project

Abstract

Aperture synthesis is the process of taking data from interferometry telescope arrays and producing an image of the sky. This research project covers the gathering of knowledge on the required topics, and then using this knowledge to create software capable of performing these processes. The software will be tested against ideal models, changed and improved upon, based on those results.

Please note, this was submitted for a mid-project review and so the report is not finished. Also, the pipeline hasn’t been finished yet, so exact details for the methods of deconvolution are not included

Introduction

An imaging pipeline for Image Synthesis is designed to take data either gathered by radio interferometry telescopes, or generated to simulate those telescopes, and form an image of an area of the sky. This software will be similar to that used for the Square Kilometer Array – a project AUT is involved in.

This project will follow a Design Science methodology where software will be the generated artefact used to experimentally investigate image synthesis. The project will start with the gathering of knowledge on the techniques used in an imaging pipeline. It will also involve gaining knowledge on solutions dealing with complications such as concurrency control and the mapping of visibilities to a grid. The project will involve implementing the techniques in the form of a Java program, and then will be tested using visibility input data available in the High Performance Computing Research Laboratory. The software will be tested against other pipelines and changes will be made to try to improve its performance.

The output from the developed software, using visibility data as input, will be an image of the sky. The images produced by the pipeline will be analysed to compare it against the known sky images for the data sets to validate whether the techniques are implemented properly and potentially look at its performance.

It is expected that an imaging pipeline will be developed with the capability of image synthesis. Also expected is to gain knowledge in the three main steps involved in the pipeline, namely gridding, (inverse) Fourier transform, and deconvolution. As well as some techniques for algorithm optimization.

Table of Contents

[Abstract 2](#_Toc6159291)

[Introduction 3](#_Toc6159292)

[Literature Review 5](#_Toc6159293)

[Synthesis Imaging 5](#_Toc6159294)

[Fast Fourier Transform (FFT) 6](#_Toc6159295)

[Gridding 7](#_Toc6159296)

[Deconvolution 7](#_Toc6159297)

[Methods 9](#_Toc6159298)

[Design of pipeline 9](#_Toc6159299)

[Implementing 10](#_Toc6159300)

[Testing and Improving 10](#_Toc6159301)

[Software Implementation 12](#_Toc6159302)

[Gridder 12](#_Toc6159303)

[Inverse Fast Fourier Transform 13](#_Toc6159304)

[Deconvolution 13](#_Toc6159305)

[Discussion of Results 15](#_Toc6159306)

[Conclusion 16](#_Toc6159307)

[**References** 17](#_Toc6159308)

Literature Review

## Synthesis Imaging

The resolution of radio telescopes can be increased by using pairs of telescopes (baselines) and taking the product of the received signals. This resolution can be changed by increasing the separation of the baseline, rather then increasing the size of the individual telescopes. This method popularized by the work of (Ryle & Hewish, 1960) states that using these baselines the telescopes produces; “exactly the same result as that obtained by using the complete large aperture”. This technique allowed for cheaper production of much larger apertures and the eventual development of the techniques used now.

These techniques gather Fourier domain data in the form of a visibility. However, the way in which they are sampled is non-uniform, so it must be placed on a rectangular grid. This process is known as gridding and the methods used now are based on the work by (Brouw, 1975). These visibilities V(u, v) fall upon the plane in which the baselines are setup. For a wider coverage of this plane, more baselines can be added and could also be moved around.

With more modern telescopes being developed, moving them around became a substantial task and instead the rotation of the earth can be used to move these points around the plane. An image of these points on the V(u, v) plane can be seen on the left of Figure 1.

As there are gaps within the V(u, v) plane the image is a “dirty image”, this can be seen on the right of Figure 1.

Synthesis imaging is a large research field with many discoveries being made, a recent image produced of a black hole received a large amount of public attention. Reading this paper (The Event Horizon Telescope Collaboration et al, 2018), showed insight into the process of developing an imaging pipeline.

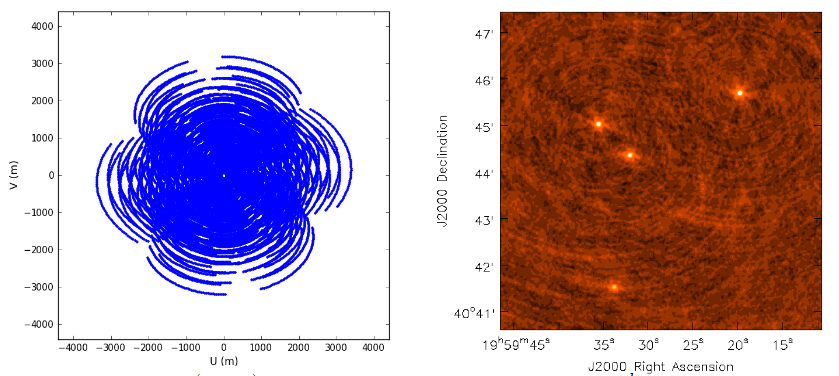


Figure 1. V(u,v) plane showing data points and a dirty image. (Rau, 2012) Retrieved from Australian Telescope National Facility, from https://www.atnf.csiro.au/research/radio- school/2012/lectures/tue/RVU\_ImagingDeconvolution.pdf

## Fast Fourier Transform (FFT)

A Fourier Transform is a process for signal-processing and analysis. (Brigham, 1988) states that the extent of the use the process is as follows; “biomedical engineering, imaging, analysis of stock market data, spectroscopy, metallurgical analysis, nonlinear systems analysis, mechanical analysis, geophysical analysis, simulation, music synthesis”. It is widely regarded as one of the most important algorithms based on its impact in so many areas. Simply put a Fourier Transform is used to show different parts of a continuous signal. However, for Interferometry an Inverse Fourier Transform is used as we are taking the data from the Fourier Plane and creating an image from it. While it is possible to perform a basic Fourier Transform using computers, the method has a run time of O(). Instead, we can perform a Fast Fourier Transform with a run time closer to O() as it is a divide and conquer method. Such a method is based upon the work of (Hogg, MacDonald, Conway, & Wade, 1969). The algorithm used was first discovered by Gauss and later rediscovered by (Cooley & Tukey, 1965) which notes that, “Wherever possible the use of N = with r = 2 or 4 offers important advantages”. This algorithm works under the assumption that the data is in an organized array, hence the visibilities must be gridded. Using butterfly operations, the data is combined in pairs using either a decimation in time or in frequency variation.

## Gridding

Gridding is the process of mapping the data collected onto a rectangular grid so that it may be processed by the iFFT, and then displayed as an image. Early techniques for placing the visibility data on a grid involved finding the closest grid point to visibility’s V(u, v) co-ordinate. Then, either adding them all together or averaging them out on that grid location. Early methods were used by (Hogg, MacDonald, Conway, & Wade, 1969). However, their method produced many artifacts in the image. Therefore, there were limited application for the process. An alternative method used by (Brouw, 1975) would take a weighted value based on the distance between local grid point and the point of the visibility. By designating a “support” area around the local grid point, the data can be added to these areas. An ideal gridding method was given by (O'Sullivan, 1985), with his gridding algorithm that used a sinc function and produced images with “arbitrarily small artifact levels”. However, this function would give infinite extent to the support. This is not ideal computationally for the gridder. Instead, convolution kernels with a set support area are used. These functions also have a quick fall off in the grid to help aliasing.

A simplified approach to gridding follows this process; for every visibility, find the closest grid point to the data on the V(u, v) plane. Then, using the convolution kernel, the data point is spread across the support region. By using a convolution kernel generated by a Prolate Spheroidal, the value of the data point is spread based on its distance to the center.

## Deconvolution

Once an image is formed from the Fourier Transform is it called a ‘dirty image’, as seen in Figure 1, this is due to the effects of having limited sampling of the V(u, v) plane. The process of Deconvolution can be used to ‘CLEAN’ the image. This method by (Högbom, 1974) uses the original V(u, v) data to form a “dirty beam”, then by taking away the dirty beam from the points of the dirty image with the greatest brightness you are left with a residual image. By iteratively carrying out this process the effects of the convolution are removed to the best extent possible.

This was expanded upon by (Clark, 1980) to make it more efficient. His method involves using more FFT’s in a major and minor cycle to subtract points away from the dirty image. The minor cycle works by performing a Högbom ‘CLEAN’ on smaller beam patches, then the major cycle applies an FFT on the points found by the minor cycle and is used to subtract from the dirty image. There is also a varient of this process where (Schwab, 1984) uses the major cycle to take away from un-gridded visibilities. This helps to remove noise from and potential errors from the gridding process.

Further examples of the ‘CLEAN’ algorithm and the Maximum Entropy technique by (Skilling & Bryan, 1984) are compared by (Cornwell & Bridle, 1996).

The image in Figure 2 shows the ‘CLEAN’ image based on the images from Figure 1.

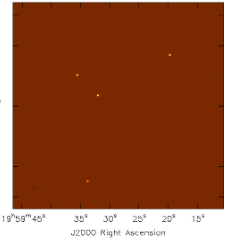
**

Figure 2. A ‘CLEAN’ image. (Rau, 2012) Retrieved from Australian Telescope National Facility, from https://www.atnf.csiro.au/research/radio- school/2012/lectures/tue/RVU\_ImagingDeconvolution.pdf

Methods

The goal of the project is to gather sufficient knowledge on the topics required to produce software capable of taking data from interferometry telescopes and produce an image of the sky. Starting with a literature review, relevant knowledge regarding the topics of Image Synthesis, gridding, Fourier Transforms, and Deconvolution were gathered. Using this knowledge, an artefact will be produced that to perform the required processes. Once the pipeline is created, it will be tested against a perfect image of the sky to ensure that it is correctly carrying out the processes involved. The pipeline will then be adapted and improved upon to increase its performance.

## Design of pipeline

The pipeline will consist of three main sections; the Gridder, the inverse Fourier Transform, and the Deconvolution. The programming language used to develop the pipeline was Java. Java is primarily taught in computer science papers at AUT and is capable of handling the size of the data I will be using. It also supports parallelization, a feature that will be implemented to improve performance.

The grid size to be used will be 1024x1024. Providing a high enough resolution to be able to properly test against. The grid length and height must be a power of 2 due to the (Cooley & Tukey, 1965) radix 2 FFT being used. This algorithm is more performant then a standard Fourier Transform under the condition that the data is ordered, hence why the visibilities are gridded. There are two different ways of implementing an iFFT; decimation in time, or decimation in frequency. Both methods are explained and compared in the software implementation section.

The data contains Visibilities.csv which is a file that includes around 23000 data points. Each visibility consists of its locations along the V(u, v) plane, followed by its value as a complex number. The dataset also includes the Prolate Spheroidal which is used as the convolution kernel for the data with a support size of 7. However, for better precision it has been oversampled by a factor of 4. The last thing needed from the data is the configuration for achieving the best accuracy for the gridder.

A double is used as the primitive data type, as in the Java language (Oracle, 1993) a double can store values from 4.94065645841246544e-324 to 1.79769313486231570e+308 including values with high decimal accuracy needed.

For Deconvolution, Högbom’s ‘CLEAN’ method will be implemented to deconvolve the true image out of the dirty image. To achieve this, a dirty beam must be constructed from the V(u, v) plane.

## Implementing

The processes of creating the pipeline will undergo many stages. This is due to multiple iterations of the software being produced as improvements are made. When the pipeline is complete, a fully formed Java project with unit testing and multiple output images will be produced.

When implementing the pipeline, constant reference to the literature will be required to ensure the processes are carrying out their operations correctly. The nature of the mathematical operations mean that high precision is needed for a meaningful image to be produced.

For a more in-depth explanation of the methods involved in the implementation, please see the following section.

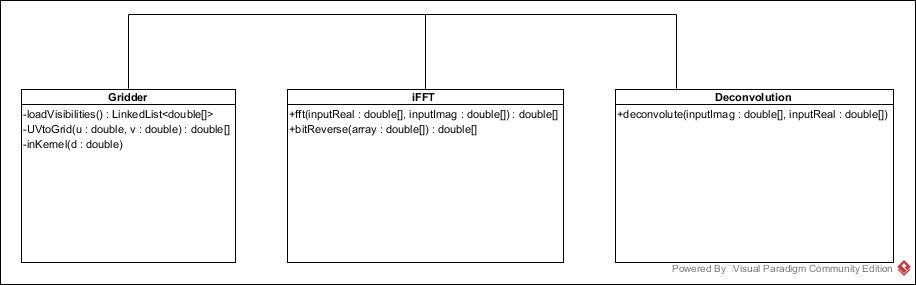
## Testing and Improving

To test the pipeline, a dataset generated by the HPCRL at AUT will be used. By using this data, the three main processes of the pipeline can be checked to see how they are performing as it can be tested against a perfect image made from the real components of the sky. This perfect image is what the pipeline would generate if working under ideal circumstances, so any inconsistences would be evidence that improvements can be made.

As the pipeline is tested, improvements will be made with respect to its runtime. An example of this is using multiple threads to perform the gridding or the Fourier Transform. However, implementing multiple threads brings issues such as concurrent updates from the threads. A concurrency issue occurs when two threads attempt to update the same values simultaneously and only one update is saved. This would arise with overlapping visibilities in the gridder and requires a solution such as a lock on each value to prevent being accessed at the same time. Another method is to have each thread manage its own grid, combining each grid after all the visibilities have been processed.

Software Implementation

The implementation will be broken into three classes. Gridder.java, which is responsible for reading the dataset and placing the points on a grid. iFFT.java, for carrying out an Inverse Fourier Transform on the data. Deconvolution.java, implementing a ‘CLEAN’ algorithm to turn the dirty image into a more representative form. Lastly, a python script will be used to display the image. This script was given by the HPCRL at AUT. Below is a UML diagram of the Java classes.



## Gridder

Implementing the gridder begins with initializing two 2d double arrays. These are used to store the real and imaginary values respectively. As the values are placed on the grid they will be stored in these arrays. Following that, the gridder loads the data from the .cvs file and stores it as a double array. The first two indexes are the u and v coordinates, after that is the real and then imaginary values. These double arrays are placed in linked list data type to ensure the size of the list is not an issue. Then going through the list, each array is transformed as per the formula mentioned before.

In order to correctly place the visibilities, two things must be used; a wavelength to meters ratio, and the UV scale. For the data generated, a frequency of 300000000 was used. To calculate the scale, we use the product of the grid size and the cell size. As discussed earlier, our grid size is 1024, and cell size is 4.848136811095360e-06 given by the dataset.

Once the visibilities are modified correctly, the process of placing the points on the grid begins. Firstly, we take the location modified by the UVScale and round it to the nearest integer to find the closest grid point. Then, using the support value of 7, we take a 7x7 grid around this center grid point as this is where the data point will be placed. Using our convolution kernel, we form a 2d array 27x27, due to our x4 oversampling, and finding the distance of the grid point to the true visibility point.

## Inverse Fast Fourier Transform

The iFFT will be used on a 2d array. This process involves transforming every row in the image, followed by transforming every column. A decimation in time variation will be used so the arrays will be bit reversed first. A bit-reversal is done on the inputs as the in-place operations imply a bit-reversed output. The decision between decimation in time or decimation in frequency the order in which the operation take place. For decimation in time, a bit reversal occurs first, and then progressively larger distance in the butterfly operations. Decimation in frequency is when the butterfly operations get shorter in distance and then the array is bit reversed. For this implementation, decimation in time is used. There is no advantage in using one technique over another as both carry out the same computations.

The process involves one complex multiplication, addition, and subtraction. A complex multiplication consists of two real additions and four real multiplications. Complex addition and subtractions involves two real additions. For the transform, the real and the imaginary parts are spilt into two separate arrays. This is done to avoid having to use a separate object to store them.

## Deconvolution

For this process, a Högbom ‘CLEAN’ will be implemented. This process will remove any artifacts and leave only point sources in an otherwise empty field. For this process, a dirty beam must be formed. If we were to call the true image T, the dirty image D, and the dirty beam P, then D is equal to T convolved by P. Therefore, in order to attain T we can use the dirty beam on points in the dirty image in an attempt to ‘CLEAN’ it. This works by finding points of high brightness within a number of regions in the dirty image. Then, by taking away a set ammount of this point using the dirty beam, and adding the point to a set of clean components. This process is done iteratively in each region until no more points that match the required brightness can be found. By using these components, a restored image can be formed. This image is only an estimate of the sky.

Discussion of Results

Conclusion

**References**

Brigham, R. O. (1988). *The Fast Fourier Transform and its Applications.* Prentice-Hall.

Brouw, W. N. (1975). Aperture Synthesis. In C. De Jager, & H. Nieuwenhuijzen, *Image Processing Techinques in Astronomy* (pp. 301-307). Dordrecht: Springer.

Clark, B. G. (1980). An efficient implementation of the algorithm 'CLEAN'. *Astronomy and Astrophysics*, 377-378.

Cooley, J., & Tukey, J. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 297-301.

Cornwell, T., & Bridle, A. (1996). *Deconvolution Tutorial*. Retrieved from National Radio: https://www.cv.nrao.edu/~abridle/deconvol/deconvol.html

Högbom, J. (1974). *Astronomy and Astrophysics Supplement Series*, 417.

Hogg, D. E., MacDonald, G. H., Conway, R. G., & Wade, C. M. (1969). Synthesis of Brightness Distribution in Radio Sources. *Astronomical Journal*, 1206-1213.

Oracle. (1993). *Class Double*. Retrieved from Javadocs: https://docs.oracle.com/javase/7/docs/api/java/lang/Double.html

O'Sullivan, J. D. (1985). A Fast Sinc Function Gridding Algorithm for Fourier Inversion in Computer Tomography. *IEEE Transactions on Medical Imaging*, 200-207.

Rau, U. (2012, Sept 24). *Imaging and Deconvolution.* Retrieved from Australia Telescope National Facility: https://www.atnf.csiro.au/research/radio-school/2012/lectures/tue/RVU\_ImagingDeconvolution.pdf

Ryle, M., & Hewish, A. (1960). The synthesis of large radio telescopes. *Monthly Notices of the Royal Astronomical Society, Vol. 120*, 220-230.

Schwab, F. R. (1984). Relaxing the isoplanatism assumption in self-calibration; applications to low-frequency radio interferometry. *Astronomical Journal*, 1076-1081.

Skilling, J., & Bryan, R. K. (1984). Maximum Entropy Image Reconstruction. *Monthly Notices of the Royal Astronomical Society*, 111-124.

The Event Horizon Telescope Collaboration et al. (2018). First M87 Event Horizon Telescope Results. I. The Shadow of the Supermassive Black Hole. *The Astrophysical Journal Letters*, L1.