School of Electrical Engineering, Computing and Mathematical Sciences

High-Time Resolution GPU Imager for Low-Frequency Radio Telescopes

Gayatri Aniruddha

This thesis is presented for the Degree of
Master of Philosophy (Physics)
of
Curtin University

December 2023

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgement has been made. This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Gayatri Aniruddha

“*Take care of Mathematics and Literature. They will take care of you*”

— G.S.S Sharma (my maternal grandfather)

Acknowledgements

So many thanks are in order :)

First and foremost, thank you, Marcin, for your time, support and guidance over the last two years. Thank you for re-scoping this project and saving my life from what it would have been otherwise (crazy!). Thank you for responding to all my e-mails so quickly and in detail and making this journey effortless for me. Thank you for always being available and being just a 'knock' away at all times of the day.

Thank you, Randall, Nicole and Chris, for all the milestone discussions.

Thank you to the entire student and staff cohort at CIRA, whose journey overlapped with mine: Steve, Ravi, Jishnu, Ajay, Aishwarya, Brandon, Jun, Kashan, Manoj, Praveen, Garvit, Silvia, Flora, Susmita, Callan, Himanshu, Tyrone, Jaiden, Mawson, Ferry, Jordan, AJ, Tom, Scott, Cristian, Ben Burridge, Chris, Mike, Luca, Karuiki, Manasvee, Leike and Lucia. To Steve, thank you for being the most ~~'helpful'~~ - 'funny' and 'cool' guy. Thank you for spending all those late-night hours with me at CIRA, teaching me astronomy and life. You are the best friend and 'life supervisor' I could ever ask for! To Ravi, thank you for being the best desk friend and gym trainer I could have asked for and for making me feel like the funniest person ever! I can see you becoming Vegan one day. To Ajay, thank you for making the city life fun and entertaining, and making me feel so welcome and included at ICRAR! To Jishnu, thank you for suggesting that I develop a Python imager and thereby save my life. To Aish, for being so kind, caring, affectionate and inclusive! To Brandon, thank you for buying code names, exploding kittens, and giving us all some of the best board game sessions! To Praveen, for laughing at all my jokes wholeheartedly and making gym so much fun! To Garvit, thank you for your 10 am and 3 pm coffee break initiatives and 'kaafi offended' times!

I would also like to thank Clancy, Adrian, Cath, Ali, Marcin G, Bradley, Sammy, Apurba, Ridhima, Dev, Maria and Kirtsy for all the fun conversations over the last two years.

Thank you to my old housemates from Yugo, Janika, Beckie and Suneha for making 2209 feel like home!

Thank you to Vishwa Bhaiya for all your encouragement and help during my difficult times in Perth.

Thank you to my extended SSRVM(Sohu, Neha and Vishu), PES(Vinith, Karthik, Dhiraj, Kiruthika, Banu, Chakith, Debarka, Gautham and Amshu et. al) and Monash fam (Vaibhs et al.) for all your long-distance support, plethora of reels, late-night catch-ups and encouragement, and legit for being a call away!

Additionally, I would like to also thank some of the new friends that I made at ICRAR, whose journey crossed paths with mine as I was working on my thesis corrections: Truman, Nick, Sriram, Kavin, Lucia, Mellisa, Ana Maria, Seona, Angel, Kris, Matt Frost, Scott, Andy, Jordan, Juno, Shinna, Marta, Connor, Robin, Nicole, Ben, Matt, Katy, Austin, Nadia, Callan, Ryan, Dave, Kevin and Mark. Thank you for making the last 6 months some of the best few months of my work life by making me feel so welcome and included at ICRAR!

I would also like to thank Andreas Wicenec, my current supervisor in the DIA team, for giving me the flexibility to work on my thesis corrections.

Thank you to my current housemates from Twin Dolphin, Gabi and Liz, for making 504 feel like home. Gabi, I miss your Sagittarius energy and our weeknight dinner and Saturday morning deep conversations on you-know-who and you-know-what.

A special thanks to Rhea, Varun, Krits and Shaunak for the much-needed amazing road trip and vacation in June and for making my home so vibrant and full. Your visit, presence and laughter gave me the much-needed fuel to wrap my thesis corrections up.

And, finally,

Thank you to Amma and Appa for everything and for answering all my cries and calls for help. Thank you, Amma, for doing all the research for me since 1997 and encouraging me to join academia, as this was the career path I never knew I needed! Thank you, Appa, for providing for me and for being so cool and chill in the literal sense. Thank you to my grandparents for spoiling me with the best food and making me feel that I was capable of doing anything and everything in this world! Thank you, Kannan Thatha, for being available for me whenever I wanted to talk. Thank you, Ammama, for constantly reminding me that everything eventually works out and falls in place, and only good things are going to come my way! Thank you, Delhi Thatha, for making my math foundation so strong, for still preserving my math notes and for making me feel like I could take on any amount of load. Thank you to my baby sister, Achu, for making me feel like the funniest and most-sorted person in the room and giving me all the ego boost I need.

And finally, I want to thank the entire Universe for helping me experience this wonderful Astronomical simulation and introducing me to this field of Astronomy! :)

Abstract

Fast Radio Bursts (FRBs) are one of the most thrilling and bizarre phenomena discovered only 15 years ago in the archival data from Murriyang, the Parkes radio telescope in Australia. They are very bright (reaching tens of Janskys) and short (millisecond) duration radio pulses originating from even the very distant Universe (redshifts up to 1). Though they were initially discovered at GHz frequencies, in the last five years, hundreds of FRBs have been detected down to 400 MHz by the CHIME telescope in Canada. So far, only a handful of FRBs have been detected by various other telescopes worldwide at frequencies below 350 MHz. Furthermore, only one FRB has been detected down to even 110 MHz, and it was detected by the LOFAR telescope in the Netherlands. However, based on the FRB daily rates measured by CHIME and extrapolated to lower frequencies, there can be even a few FRBs per day over the entire sky. Nevertheless, no FRB has been yet detected in the Southern Hemisphere below 400 MHz. The current methods of low-frequency FRB searches are computationally inefficient, and there is a need for better software. Hence, the main aim of this work was to develop a high-time-resolution imager executed on Graphical Processing Units (GPUs), which would be a part of the larger GPU-based processing pipeline searching for low-frequency FRBs. This work describes an implementation of a GPU imager for radio astronomy data and its benchmarks on the Topaz and Setonix supercomputers at the Pawsey Supercomputing Centre Finally, this work also showcases the results of a pilot FRB survey using a few hours of 100 ms data from the SKA-Low prototype station (EDA2).

Contents

List of Figures

Chapter 1

Introduction and Background

1.1 Introduction

In this chapter, a brief overview of the FRB field and FRB surveys at low frequencies are described in Section 1.2. In Section 1.3 and 1.4, the traditional beamforming-based search methods and correlation-based imaging methods, respectively, are described along with the associated computational costs. The target instruments and the previous FRB searches with these instruments are discussed in Section 1.5. The chapter concludes with the goal of the thesis in Section 1.6 and the thesis layout in Section 1.7.

1.2 Fast Radio Bursts

FRBs are bright, short-duration (milliseconds) signals typically originating from outside our galaxy, the Milky Way. The first FRB, (FRB 20010724A[1]), also known as the Lorimer burst (depicted in Figure 1.1), was discovered in 2007 while processing archival pulsar data from Murriyang, the Parkes radio telescope in Australia (Lorimer et al., 2007). This particular FRB was discovered even

[1] The naming convention followed by the FRB community is 'FRB YYYYMMDDX'. This includes the year, month, day, and an additional letter (X) to distinguish events reported on the same day.

before the term 'fast radio burst' was coined by Thornton et al. (2013), and for many years after its discovery, this burst was the only event of this kind.

When FRBs travel from their source, they undergo dispersion, a mechanism where the group velocity of the wave changes as a function of frequency. In the context of FRBs, this is due to the free electrons along the line of sight. As a result, the signals are delayed, i.e. the signals at a higher frequency arrive earlier, whereas signals at a lower frequency arrive later. For reference, see Figure 1.1, which shows the dynamic spectrum of the Lorimer burst. This delay in their arrival times *(Δt)* in ms is a function of observing frequency *(ν^{-2})*, and can be calculated according to the equation:

$$\Delta t = 4.15 \times 10^{6} (ms) \times DM \times (\nu_l^{-2} - \nu_h^{-2}), \tag{1.1}$$

where Δt is the dispersive delay in ms, DM is the dispersion measure in pc cm^{-3}, ν_l and ν_h are the lower frequency and higher frequency respectively (both expressed in MHz) (Lorimer & Kramer, 2012). The DM represents the integrated column density of the free electrons along the propagation path length *(l)* of the FRB, given by the following equation:

$$DM = \int_0^D n_e \, dl \, (\mathrm{pc}\, cm^{-3}), \tag{1.2}$$

where l is the path along the line of sight, n_e is the electron number density along the line of sight and D is the distance to the FRB. Hence, by assuming that the distribution of electrons in the intervening medium is uniform, the DM of an FRB can be used as a proxy for distance to the FRB to estimate the propagation path length.

The total observed DM of an FRB can be expressed as the sum of three components, given by the following equation:

$$DM_{total} = DM_{MW} + DM_{IGM} + DM_{host}, \tag{1.3}$$

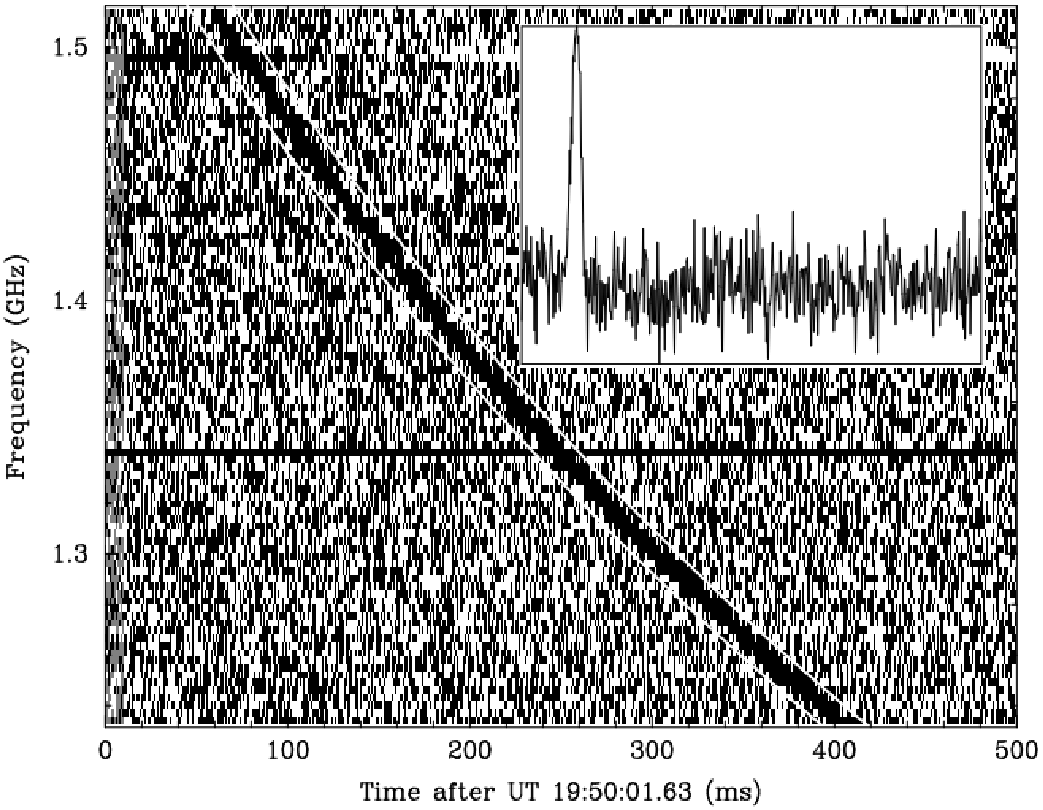


Figure 1.1: The dynamic spectrum of Lorimer burst. In this dynamic spectrum plot, the horizontal axis represents time-step (in ms), and the vertical axis represents frequency (in GHz). It is seen that the high-frequency components arrive earlier than the low-frequency components. The dispersive delay is seen as a sweep across frequency. The horizontal line at around 1.34 GHz is due to a malfunctioning frequency channel. Inset: The 'pulse profile' summed over all the frequency channels after de-dispersion, where suitable time delays are added, over different DM trials to align the signals from different frequency components in time.
Image credits: (Lorimer et al., 2007).

where DM_{total} is the total measured DM of the FRB, and DM_{MW}, DM_{host} and DM_{IGM} are DM contributions of the Milky Way galaxy, host galaxy of the FRB and the intergalactic medium (IGM) respectively. The DM$_{MW}$ can be estimated from the galactic electron density models such as NE2001 (Cordes & Lazio, 2002) and YMW16 (Yao et al., 2017). Similarly, the contribution from DM_{host} includes the interstellar medium (ISM) within the FRB host galaxy along with its halo measurements and the contribution of the local environment surrounding the

FRB. Finally, the DM contribution of the IGM is related to the redshift of the FRB (Inoue, 2004; Ioka, 2003). The DMs of all FRBs mostly exceed what the Milky Way contributes, further pointing to its extragalactic origins. This has also been confirmed by identifying the host galaxies of a few subarcsecond localised FRBs (e.g. Chatterjee et al. (2017); Macquart et al. (2020)). In general, most FRBs have high DMs predominantly spread in the range of 200 - 1700 pc cm^{-3} (Li et al., 2018), illustrated in Figure 1.2.

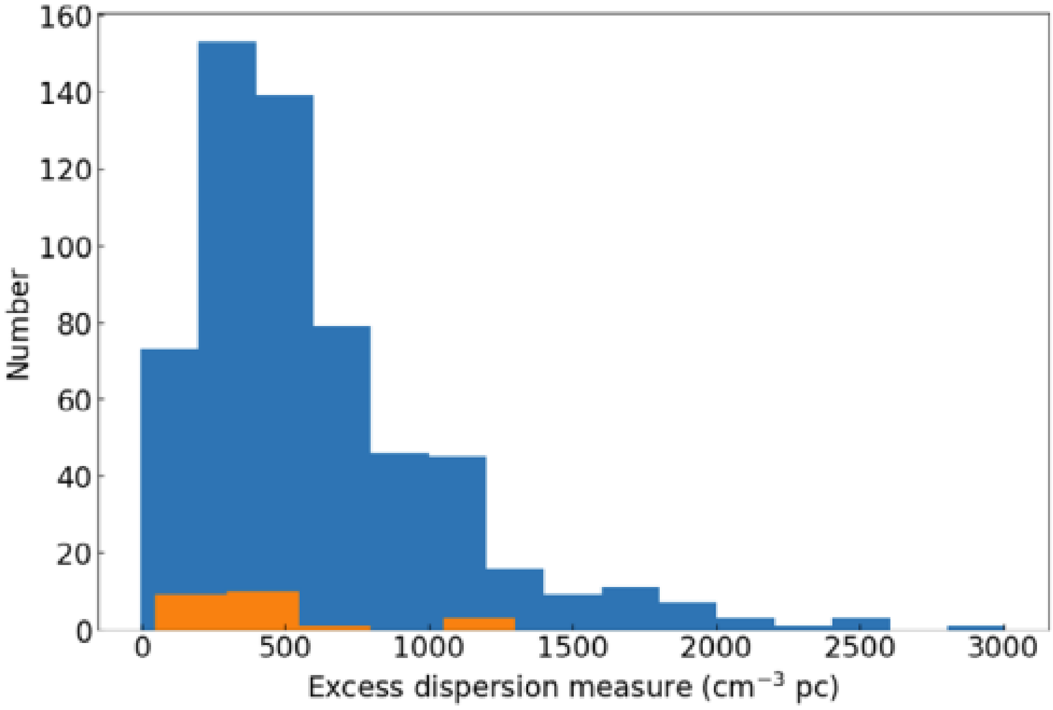


Figure 1.2: The distribution of reported DMs of the presently known sample of FRBs for one-off bursts (blue histogram) and the first-detected bursts from observed repeaters (orange histogram). Here, the excess dispersion measure refers to the DM_{total} - DM_{MW}, in terms of Equation 1.3.
Image credits: (Petroff et al., 2022).

1.2.1 FRB population

Since their initial discovery in 2007, the field of FRBs has emerged as one of the most dynamic and vigorously studied areas in astronomy. The FRB population is growing incredibly, so our knowledge of their origins is evolving rapidly. The total number of FRB detections made as of November 2023 is 757 (Petroff & Chatterjee, 2023). A distribution of known FRBs as of Jan 2022 is shown in Figure 1.3. Currently, most of the detections (around 500 FRBs, including 20 repeaters) have been made by the Canadian Hydrogen Intensity Mapping Experiment (CHIME/FRB Collaboration et al., 2018), which was constructed for real-time FRB detections between 400 and 800 MHz (CHIME/FRB Collaboration et al., 2021), making CHIME a top performing 'FRB machine'. Most FRBs seem to be one-off (or non-repeating) events, i.e., detected only once at a given position. However, some FRBs are known to repeat, where bursts are seen from the FRB source at irregular intervals. Presently, there are 51 such sources known to repeat, and there is evidence for periodic activity from: FRB 20121102A (Cruces et al., 2021; Rajwade et al., 2020) and FRB 20180916B (CHIME/FRB Collaboration et al., 2020).

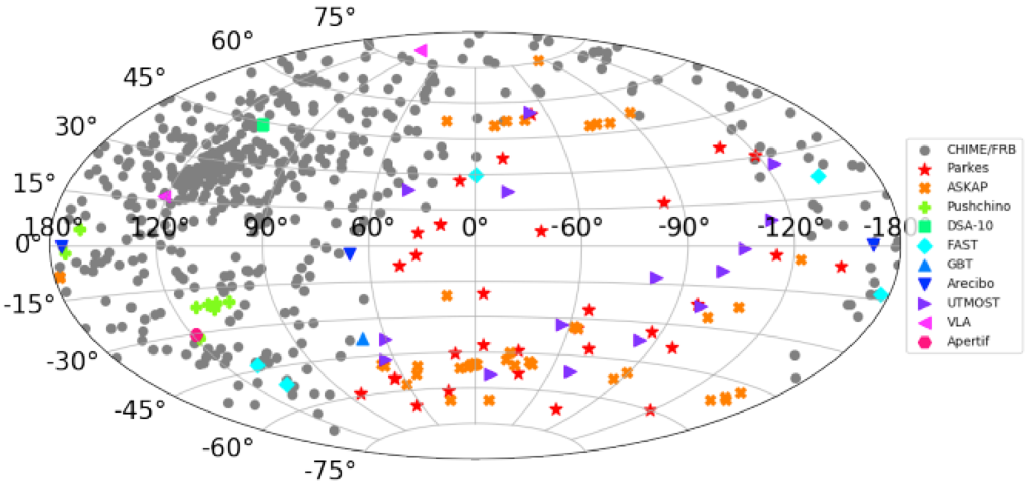


Figure 1.3: Distribution of known FRBs as of Jan 2022. The distribution is in Galactic coordinates.
Image credits: (Petroff et al., 2022).

1.2.2 FRB searches below < 400 MHz

To date, FRBs have been detected at frequencies from 300 MHz (Chawla et al., 2020; Pilia et al., 2020; Parent et al., 2020) up to 8 GHz (Gajjar et al., 2018).

However, though searches for FRB emission at low frequencies (< 400 MHz) are ongoing using Low-Frequency Array (LOFAR, van Haarlem et al., 2013), MWA and the Long Wavelength Array (LWA, Ellingson et al., 2009), as of now, the only detection has been of the repeating FRB 20180916B (CHIME/FRB Collaboration et al., 2019). Although many FRBs have been observed to repeat at irregular intervals, this FRB has been observed to repeat regularly, approximately every 16.35 days. Bursts are observed for a period of four days, followed by an inactive period of about twelve days (CHIME/FRB Collaboration et al., 2020). The Green Bank Telescope (GBT, Jewell, 2000) in the United States detected bursts of the repeating FRB 20180916B at 350 MHz on 19 December 2019 and 20 December 2020 (Chawla et al., 2020). Similarly, the Sardinia Radio Telescope (SRT, Prandoni et al., 2017; Bolli et al., 2021) in Italy, pointed in the direction of FRB 20180916B, also detected bursts at 328 MHz during the observations performed from 20 - 24 February 2020 (Pilia et al., 2020). Furthermore, bursts were detected by the upgraded Giant Meterwave Radio Telescope (uGMRT) in India in the frequency band from 300 to 500 MHz on 23rd and 24th March 2020 (Sand et al., 2020). The first low-frequency detections of the bursts from this FRB were achieved with LOFAR in the Netherlands between 6 June 2019 and 26 August 2020. Around 18 bursts were observed at 110–188 MHz (see Figure 1.4), which were, to date, the lowest frequency detections of any FRB (Pleunis et al., 2021). Thus, FRB 20180916B was a milestone in the history of FRB detections as this was the first FRB to be observed at low frequencies (even down at 110 MHz), and this detection confirmed the presence of low-frequency emission from at least some FRBs and gave renewed hope for low-frequency FRB detections.

1.3 Traditional FRB search methods

Traditional FRB searches are similar to single pulse searches from pulsar observations, where data are prepared for searches through tied-array beamforming. A detailed description of tied-array beamforming implemented with the MWA can

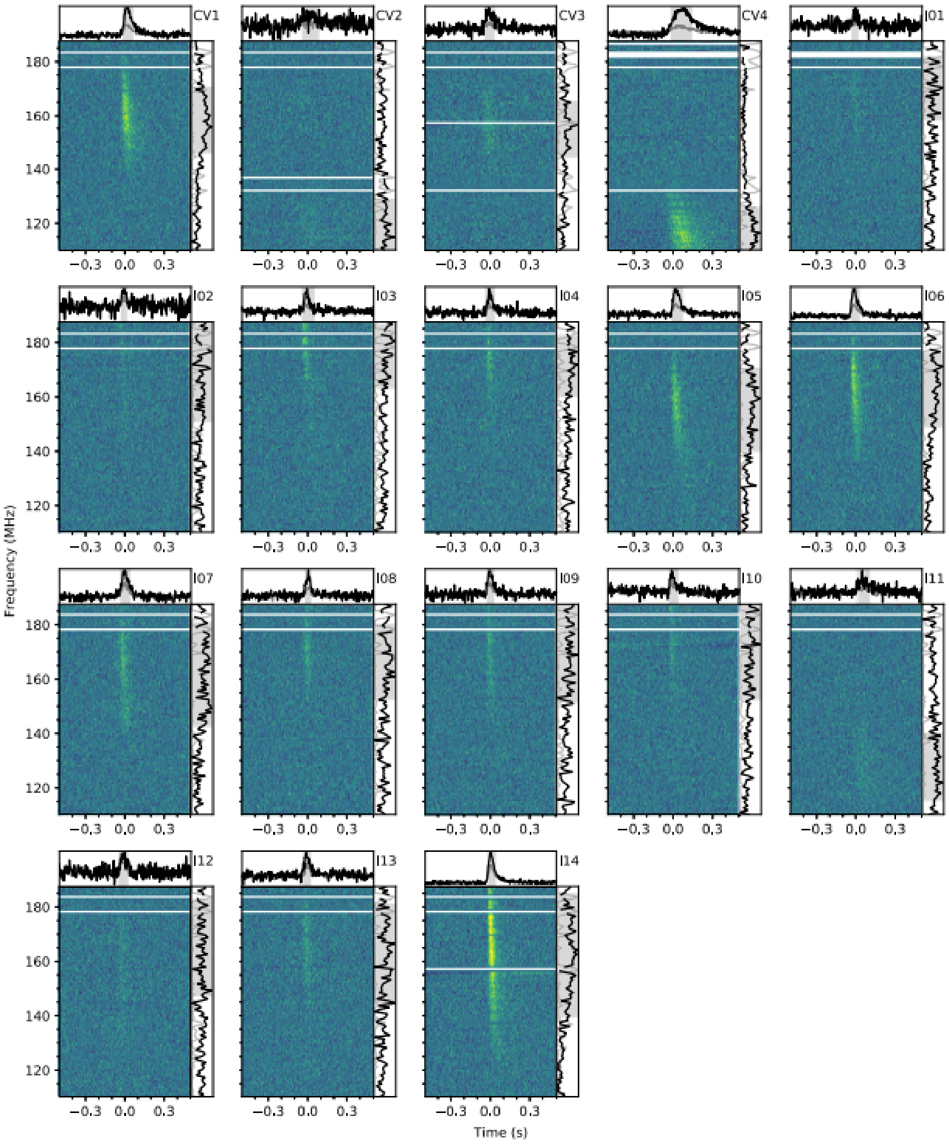


Figure 1.4: The dynamic spectra of the 18 bursts detected by LOFAR. The colour scale is uniform across all the panels. The horizontal white bands denote parts of the spectra where RFI was masked. On the right-hand side of each panel, the time-averaged spectra are presented in black, and the portion of the averaged frequency and time points that were masked to eliminate RFI are shown in light grey. On the top of these panels, the frequency-averaged pulse profile, spanning the spectral envelope of the burst, is represented in black, and the entire band (110 - 188 MHz) is presented in grey. The events labelled 'CV_n' and 'I_n' are those from the complex voltage (CV) data and total intensity data, respectively.
Image credits: (Pleunis et al., 2021).

be found in Ord et al. (2015) and Ord et al. (2019). In this section, an overview of these algorithms is given.

1.3.1 Tied-array beamforming

In beamforming, an attempt is made to estimate the sky brightness in a particular direction. To achieve this, we perform a sum of voltages over all the antennas to produce a data product similar to a single pixel of an interferometric image. Here, raw voltages recorded by the telescope are calibrated and combined into a single (dual-polarisation voltage) tied-array beam (also called a *pencil beam*) by performing a coherent (in phase) addition of voltages over all the antenna elements by applying suitable delay and phase corrections. The phases of voltages recorded by individual antennas are modified to correct for different path lengths to antennas, assuming arrival from a specific direction in the sky. This tied-array beam ($e^{'}$) can be calculated as:

$$e' = \sum_{j=0}^{N_a} J_j^{-1} v_j e^{-i\phi_j}, \tag{1.4}$$

where v_j is the complex voltage from antenna j, J_j^{-1} is the inverse of the complex gain from the direction-independent calibration (for example, in the case of the MWA, it is calculated by the Real Time System (RTS, Mitchell et al., 2008)), $e^{-i\phi_j}$ represents the geometric delay, and N_a is the number of antennas in the telescope.

To search for FRBs, complex voltages from individual antennas are beamformed in a particular direction in the sky, and the resulting time series of complex voltages are searched for FRBs. To cover the entire field-of-view (FoV) of the interferometer, multiple tied-array beams are formed, and this number depends on the angular size of the tied-array beam. For the 'beamformed image' of size $N_{pixels} \times N_{pixels}$, the number of pixels in 1D can be calculated as $\approx$ FoV$/(\lambda/B_{max})$, where (λ/B_{max}) is the spatial resolution of the interferometer at an observing wavelength λ and B_{max} is the maximum distance between the antennas of the interferometer. Hence, to produce an $N_{pixels} \times N_{pixels}$ image from multiple tied-array beams using N_a antennas, the associated computational can be estimated as $O(N_a N_{pixels}^2)$. Additionally, there is significant I/O time to read data from the disk. This further led to the development of a multi-pixel beamformer for the MWA, described in Swainston et al. (2022), which runs on GPUs, can produce many tied-array beams at the same time and works almost ten times faster than the original single-pixel beamformer. As described in Swainston et al. (2022), the approximate time estimated to process a 10-minute observation (with MWA Phase II compact array configuration) with tied-array beams for the entire FoV, using OzSTAR[2] is around 14 weeks with the single-pixel beamformer and 22 days with the multi-pixel beamformer, assuming 16 CPUs are available on average. Thus, though this GPU-based multi-pixel beamforming software is efficient, it is still not sufficiently fast to support real-time processing. Therefore, to explore possible speed-ups, this work explores an alternative approach by forming high-time resolution time series in multiple directions using sky images obtained with interferometric imaging, almost the same as forming multiple tied-array beams in the sky. The theoretical computational costs associated with the main steps of imaging are described further in Section 1.4.

1.4 Correlation-based imaging

This is a different approach to forming sky images, and the different steps involved in correlation-based imaging (explained further in Section 2.2) are correlation, gridding, and Fourier transform. In this section, the focus is on the approximate computational costs of these three processes.

For a given telescope with N_a antennas, there are a total of $N_a(N_a - 1)$ baselines, resulting in $N_a(N_a - 1)$ correlation products, ignoring polarisation. Hence, the computational cost associated with correlation can be represented as

[2] Swinburne University's OzSTAR supercomputer.

$O(N_a(N_a - 1))$. Here, $O(N)$ represents the execution time required for the algorithm or process, based on the input size N, and 'O' here stands for 'order of magnitude'. In the next step, these correlation products must be re-sampled or gridded on a regularly spaced grid before performing a Fourier transform to get the sky image. Hence, the computational cost associated with gridding will be the same as that of correlation, i.e. $O(N_a(N_a - 1))$. This suggests that the run times of correlation and gridding increase quadratically with an increase in the number of antennas.

Similarly, the computational cost associated with performing a 2D Fast Fourier transform (FFT)[3] to generate an output image of dimensions: $N_{pixels} \times N_{pixels}$, can be represented as: $O(N_{pixels}^2 log_2(N_{pixels}^2))$. Thus, this suggests that the execution times of the Fourier transform operation increase quadratically with an increase in the image size.

Thus, the total computational cost associated with correlation-based imaging can be calculated as the sum of the costs associated with correlation, gridding and imaging: $O(N_a(N_a - 1)) + O(N_a(N_a - 1)) + O(N_{pixels}^2 log_2(N_{pixels}^2))$. In both beamforming and imaging, the total computational cost is dominated by the component $\approx \alpha N_{pixels}^2$, where $\alpha = N_a$ for beamforming and $\alpha = log_2(N_{pixels}^2)$ for imaging. Thus, in general, for a given image size N_{pixels}^2, these calculations will result in imaging being faster than beamforming since $\mathrm{O}(N_a.N_{pixels}^2) > \mathrm{O}({N_a}^2 + N_{pixels}^2\ \log_2(N_{pixels}^2))$ when $N_a \geq \log_2(N_{pixels}^2)$ (i.e. for any reasonable size of N_a and N_{pixels}^2). Also, it can be seen that the FFT expenses primarily influence the cost of imaging. Hence, in this work, a GPU imager will implemented, starting from these correlation products as inputs, and the main goal of the presented GPU imager is to become a part of a larger GPU-based processing pipeline searching for FRBs. This pipeline will read the complex voltages (i.e. input) directly from the telescope or archive, and all processing will be done completely inside GPU memory to reduce the number of I/O operations. When the image data cubes

[3] https://en.wikipedia.org/wiki/Fast_Fourier_transform

are created, dynamic spectra for all pixels (i.e. time series in all directions in the sky) will be formed, which will be searched for FRBs or other short-duration transients. FRB searches will be performed either using existing software packages (e.g., Anderson et al. (2021), Magro et al. (2011), Ransom (2011), Zackay & Ofek (2017), etc.) or new algorithms will be developed. In the next Section 1.5, the two main target instruments for this GPU imager will be discussed.

1.5 Target Instruments

The Engineering Development Array 2 (EDA2, Wayth et al., 2022) (see Figure 1.5) and the Murchison Widefield Array MWA (MWA, Tingay et al., 2013) (see Figure 1.8) are the two target instruments for the GPU imager developed in this presented work. They are both located at the Inyarrimanha Ilgari Bundara, the CSIRO Murchison Radio-astronomy Observatory in the mid-west of Western Australia, a region with extremely low radio frequency interference (RFI) levels. The parameters of these two instruments are summarised in Table 1.1. This section provides a brief overview of these target instruments and a summary of previous FRB searches with these instruments.

Parameter	**MWA**	**SKA-Low station**
Number of antennas (N_{ant})	128[1]	256
Maximum baseline (B_{max})	3 km (MWA Phase 1) 5.3 km (MWA Phase 2)	35 m
Frequency range	70 - 300 MHz	50 - 350 MHz
FoV at 200 MHz	≈(20° x 20°)	≈(12000 deg^2) [2]
Spatial Resolution at 200 MHz (arcmin)[3]	1.7 (MWA Phase 1) 1.0 (MWA Phase 2)	150

[1] MWA was recently upgraded to 144 tiles, with the intent of future upgrade to 256 tiles.
[2] Full hemisphere at elevation $\geq 20°$.
[3] Calculated as $\approx (\lambda/B_{max})$, where λ is the observing wavelength and B_{max} is the maximum baseline length.

Table 1.1: Summary of parameters of the MWA and the SKA-Low stations.

1.5.1 Engineering Development Array 2

The main target telescope for the imager presented in this thesis is one of the low-frequency Square Kilometre Array (SKA-Low) prototype stations, the EDA2. Figures 1.5 and 1.6 show 256 dual-polarisation dipole antennas, aligned in North-South and East-West directions, that are pseudo-randomly spread over a diameter of 35 m. The block diagram of EDA2 is shown in Figure 1.7. The dipoles are connected to Small Modular Aggregation & RFoF Trunk (SMART) boxes via a KSR100 coaxial cable of length 7 m, and each SMART box clusters 16 dipoles. It converts the electrical signals to optical signals and sends signals downstream as RF-over-Fibre (RFoF) optical fibre cables. The Field Node Distribution Hub (FNDH) provides and controls power to the SMART boxes, aggregates signal from each SMART box and has optical fibres connected to long-distance (roughly 5 km) optical fibre connections to the control room at the observatory. In the control room, 16 Tile Processing Modules (TPMs) (Naldi et al., 2017) receive and process the optical signals (Wayth et al., 2022).

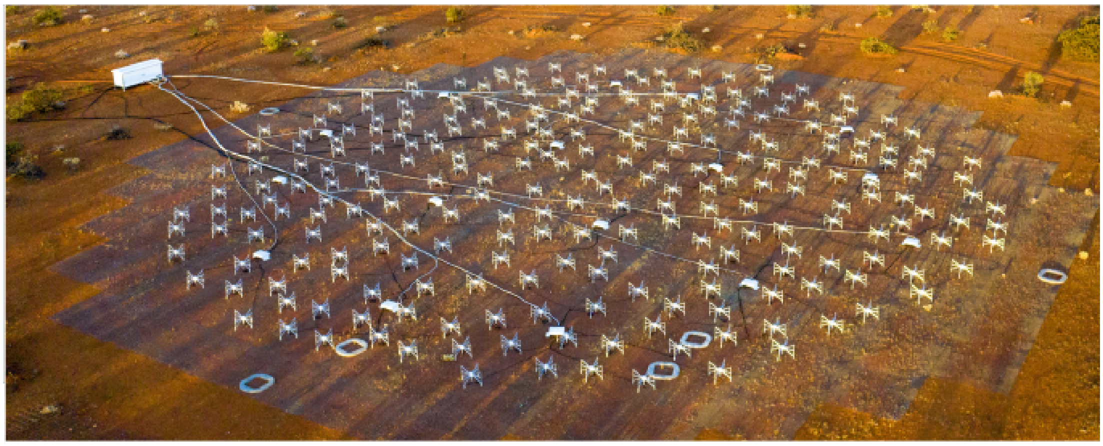


Figure 1.5: The aerial view of the EDA2. The SMART boxes are seen as the small white boxes, and the FNDH is the large white box outside the array of dipoles.
Image credits: (Wayth et al., 2022).

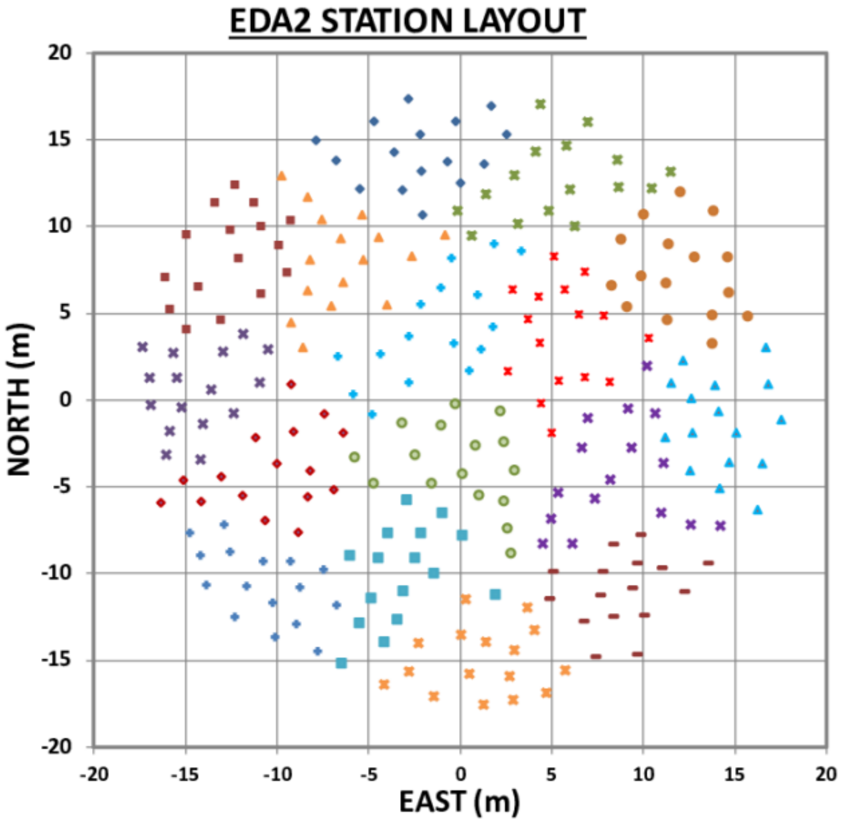


Figure 1.6: The antenna layout of the EDA2. The dipoles with the same colour and symbol are connected to the same SMART box.
Image credits: (Wayth et al., 2022).

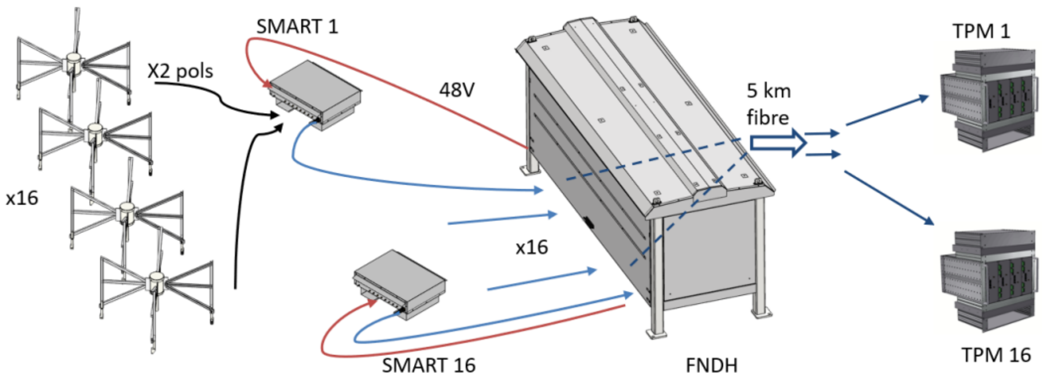


Figure 1.7: The main components of the EDA2. The antennas are connected to SMART boxes in groups of 16. Each SMART box is connected to the FNDH.
Image credits: ()2022JATIS...8a1010W.

1.5.2 Murchison Widefield Array

The imager will also be applied to other low-frequency telescopes, particularly the MWA, the Australian low-frequency precursor of the SKA-Low. The MWA started science operations in 2013 (i.e. phase 1). Its operational frequency range is between 80 and 300 MHz. It originally had 128 aperture array 'tiles', consisting of 16 dual-polarisation dipole antennas in every tile with a minimum baseline of 7.7 m and a maximum baseline of ≈ 3 km, having a field of view ranging from 300 deg^2 to 1000 deg^2 (for a detailed description, see Tingay et al. (2013)).

The phase II upgrade of MWA in 2018 (Wayth et al., 2018) has an additional 128 tiles, which can be configured either as a compact array or as an extended array with baselines within ≈300 m and ≈6 km respectively. An illustration of MWA's compact and extended configurations is shown in Figure 1.9.

The Voltage Capture System (VCS, Tremblay et al., 2015) of the MWA provides high-time and frequency resolution capabilities for the MWA. MWA VCS observations have a bandwidth of 30.72 MHz in 24 1.28 MHz coarse channels, which can be arbitrarily distributed over between 70 and 300 MHz as needed, and have 10 kHz frequency channels and 100 µs time resolution. In addition to the VCS mode, MWA has a large FoV ≈610 deg^2 at 155 MHz, making it highly suitable and effective for large-scale surveys of the sky for FRBs.

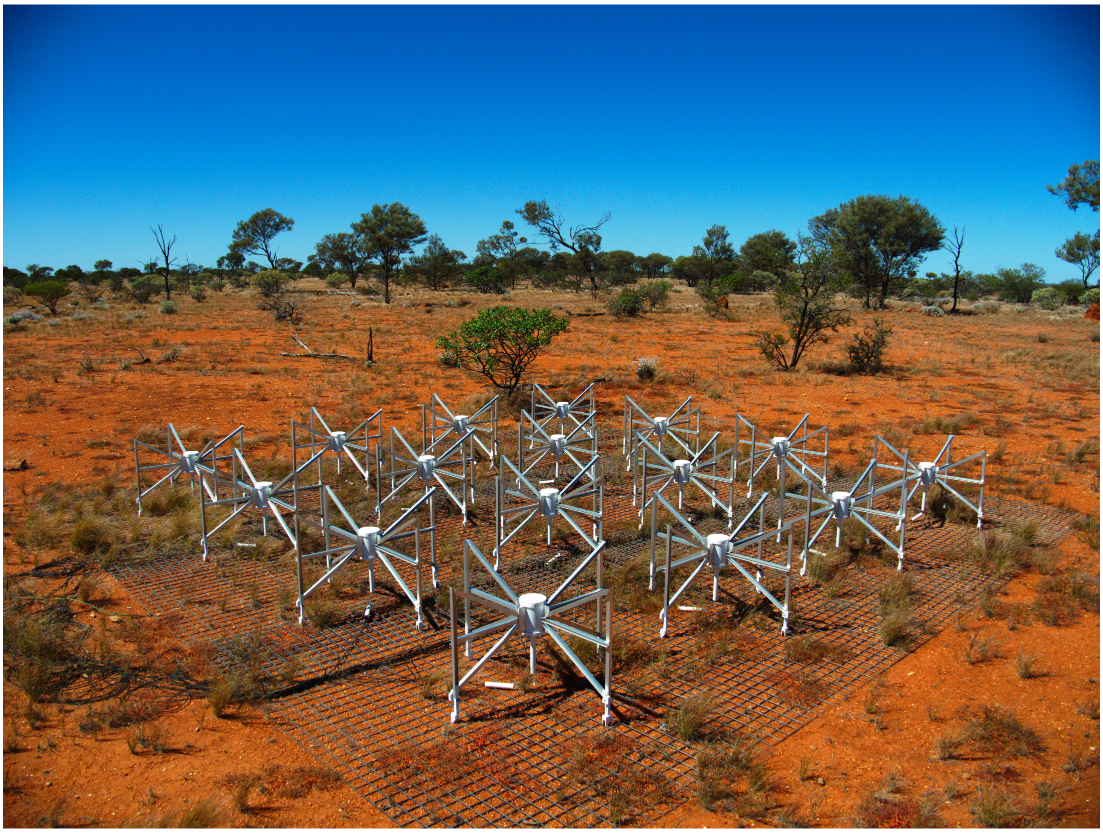


Figure 1.8: Single tile of MWA.
Image credits: https://www.mwatelescope.org/telescope/site/

1.5.3 Previous FRB searches with the MWA

The MWA has been formerly utilised to search for FRBs in the image domain (Tingay et al., 2015; Rowlinson et al., 2016; Sokolowski et al., 2018) as well as using MWA VCS observations (Tian et al., 2023). In this section, an overview of these searches, followed by their results, is given.

In Tingay et al. (2015), a search for FRBs in the frequency range between 139-170 MHz was performed using 10.5 hours of observations recorded in the imaging mode with a time resolution of 2s, frequency resolution of 1.28 MHz over a sky area of 400 deg^2. FRBs are not expected to be generally strong enough to dominate these sample integration times. However, 2s integration times were used because no better capability was available on the software front at that time. Here, the data set targeted a specific field initially obtained for Epoch of

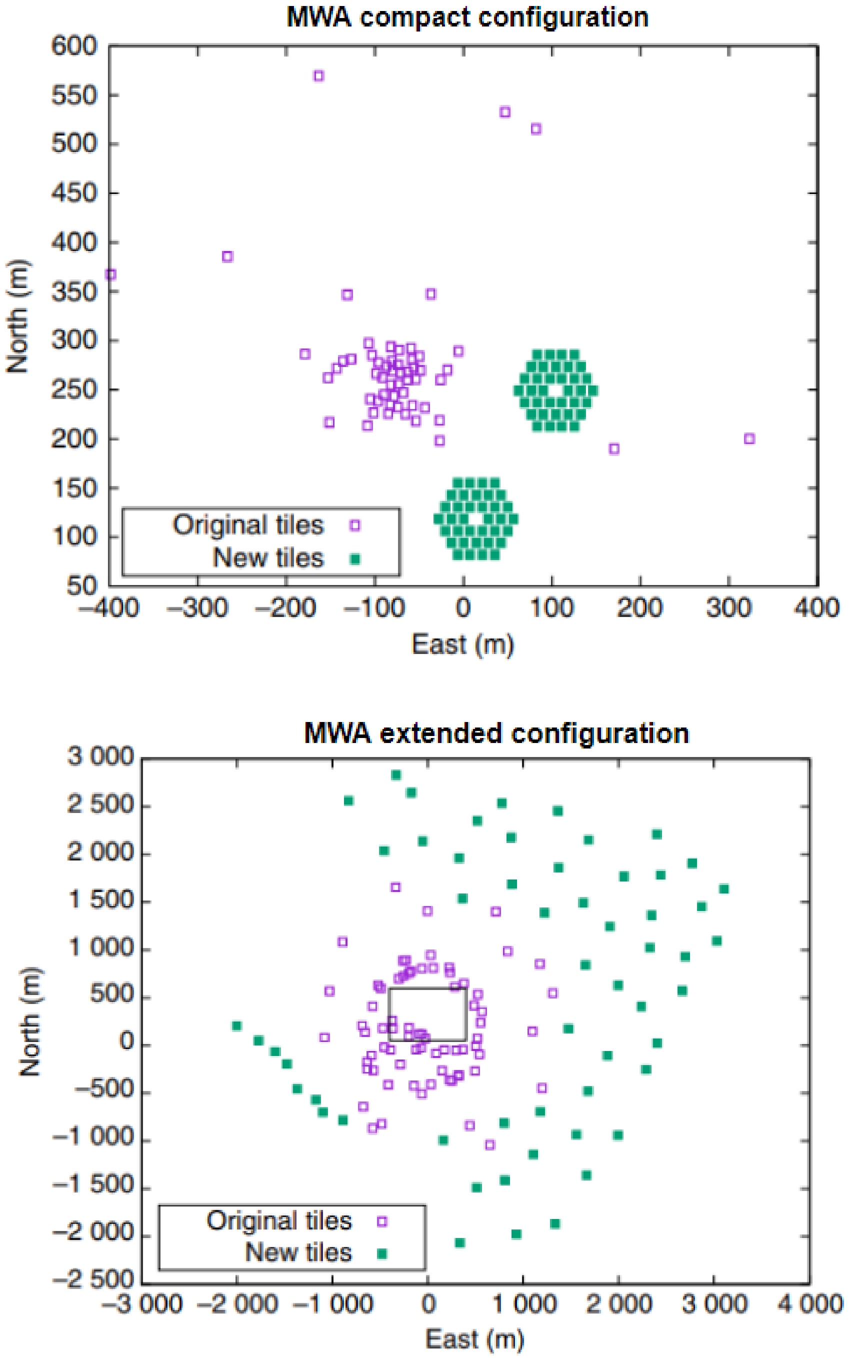


Figure 1.9: The green squares show the tile layouts for MWA's phase II compact and extended configurations in the top and bottom panels, respectively. The figure was obtained from Wayth et al. (2018).

Re-ionisation (EoR) observations. The data was processed using '`COTTER`' (Offringa et al., 2015), followed by the `MIRIAD` package to generate images, and

DART[4] (Wayth et al., 2011) was used to perform dedispersion over different DM trials. Thus, existing software packages and non-parallelized computing were used, which needed some manual intervention, and it took around three days to process around two hours' worth of data. No FRB candidates were found above a threshold of 700 Jy ms, which placed an event rate limit of < 700 per sky per day, above a fluence F > 700 Jy ms.

In Rowlinson et al. (2016), a survey was performed with a frequency resolution of 30.72 MHz and a time resolution of 28s, using 100 hours of MWA observations at 182 MHz. The same field used in Tingay et al. (2015) was imaged here, with a resulting FoV of 452 deg^2. The data were processed using COTTER, images were created using WSCLEAN, and images were processed using the LOFAR TRANSIENTS PIPELINE (TraP, Swinbank et al., 2015). No candidates above a threshold of 7980 Jy ms were found, placing an event rate of < 82 FRBs per day per sky for dispersion measures < 700 pc cm^{-3}, above a fluence F > 7980 Jy ms.

In Sokolowski et al. (2018), a coordinated targeted search campaign was conducted with the MWA to shadow the FoV of (Macquart et al., 2019; Shannon et al., 2018) the Commensal Realtime ASKAP Fast Transients (CRAFT, Macquart et al., 2010) survey, using Australia SKA Pathfinder (ASKAP, DeBoer et al., 2009) at 1.4 GHz. In this campaign, the MWA co-tracked the pointing directions of the ASKAP antennas to observe the same area of the sky. During these co-observations, ASKAP detected seven FRBs at 1.4 GHz, and the MWA data were used to look for their counterparts at 170 - 200 MHz. This enabled to accurately determine the DM, arrival time and the approximate pointing position (usually within a 10' x 10' region) for each known burst, greatly reducing the search volume of parameter space compared to a blind survey. The CASA software calibrated the MWA data, and dirty images were created using WSCLEAN (Offringa et al., 2014), and a transient detection algorithm was used to generate a list of reference sources above a 5σ threshold in these images. A dedispersion

[4] Dedisperser of Autocorrelations for Radio Transients

search was conducted over simultaneous MWA observations of the seven ASKAP FRBs with a temporal and spectral resolution of 0.5s and 1.28 MHz across the 170-200 MHz band. No candidates were found to exceed a 5σ threshold, corresponding to a fluence upper limit of 450 - 6500 Jy ms (differed between FRBs). Thus, this campaign determined that this could be due to a spectral turn-over above 200 MHz or a dense absorbing medium near the ASKAP FRBs.

In Tian et al. (2023), a targeted search was performed to search for low-frequency FRB emission in the frequency range from 144-215 MHz from five known repeaters using 23.3 hours of archival MWA VCS data recorded between September 2014 and May 2020. The VCS data offers significantly greater time resolution than the imaging mode, making it more sensitive to short-duration (ms) FRB emission. The data was processed using the VCS data processing pipeline initially developed for pulsar detections (e.g. Bhat et al. (2016), McSweeney et al. (2017), Meyers et al. (2017)), and the single-pulse search was done using the `PRESTO` (Ransom, 2001) software. The search was performed using a time resolution of 400 µs, a frequency resolution of 10 kHz, over a 100 pc cm^{-3} DM range around each repeater's known DM. No FRBs above a 6σ threshold were detected in the MWA observations, and fluence limits in the range 32 - 1175 Jy were established from the observations.

In conclusion, a comparison of the above four searches with the MWA has been summarised in Table 1.2. It can be seen that the VCS observations used in Tian et al. (2023) exhibit the finest sensitivity. Though the previous searches with the MWA successfully established limits on the sensitivity of the searches and upper limits on the FRB rates, no FRB has been discovered with the MWA (or with EDA2).

1.6 Motivation for this Thesis

In summary, the motivations for developing a high-time resolution GPU imager as a part of this thesis are diverse. Firstly, as shown in Sections 1.3 and 1.4,

Table 1.2: Comparison between the earlier FRB searches with the MWA.

Work done	Obs. (hours)	Frequency (MHz)	Time, Frequency resolution	Sensitivity (Jy ms)
Tingay et al. (2015)[1]	10.5	139-170	2s, 1.28 MHz	700
Rowlinson et al. (2016)[2]	100	182	28s, 30.72 MHz	7980
Sokolowski et al. (2018)[3]	3.5	185	0.5s, 1.28 MHz	450-6500
Tian et al. (2023)[4]	24.1	144-215	400 µs, 10 kHz	32-1175

[1] It took around three days to process two hours' worth of data.

[2] Processing times not available.

[3] Processing time: N/A as this was a targeted search with known FRB positions, and the scripts were mostly written in Python.

[4] Processing times not available.

due to the larger computational costs associated with beamforming-based FRB searches, in this work, alternate image-based approaches to FRB searches have been explored. In this approach, high-time resolution time series in multiple directions in the sky are obtained with interferometric imaging, which is almost the same as forming multiple tied-array beams in the sky to cover the FoV of the telescope (Ord et al., 2019). Depending on the number of antennas (N_a) of the telescope and the image size ($N_{pixels} \times N_{pixels}$), imaging can be computationally more efficient than forming multiple tied-array beams when $N_a \geq \log_2(N^2_{pixels})$. For example, for image size 180 × 180, imaging is faster than beamforming when $\mathrm{N}_a \geq 15$. Similarly, for image size 1024 × 1024, imaging is faster than beamforming when $\mathrm{N}_a \geq 20$. In Table 1.3, the approximate calculations of theoretical computational costs for the two target instruments of this GPU imager, the EDA2 and the MWA, to produce output images of sizes 180 × 180 and 1024 × 1024, respectively are shown. MWA has a much higher spatial resolution than the EDA2 (see Table 1.1), so the number of independent beams within FoV is larger. Hence, a larger image size was used for the MWA. For 128 and 256 antennas (MWA and EDA2, respectively), the number of operations in beamforming is around 6 and 13 times larger than in imaging. Thus, for these specifications, imaging is more efficient than beamforming.

Secondly, although existing imaging packages such as Common Astronomy

Table 1.3: The approximate theoretical total computational costs associated with beamforming and image-based data preparation for both MWA and EDA2.

Number of antennas (N_a)	**Image-size**[1] ($N_{pixels} \times N_{pixels}$)	**Beamforming cost**[2] (n_{bf})	**Imaging cost**[3] (n_{im})	**Ratio** (n_{bf}/n_{im})
256 (EDA2)	180 × 180	8.29×10^6	6.16×10^5	13.5
128 (MWA)	1024 × 1024	1.34×10^8	2×10^7	6.4

[1] For the 'beamformed image' of size $N_{pixels} \times N_{pixels}$, the number of pixels in 1D can be calculated as $\approx$ FoV$/(\lambda/B_{max})$, where (λ/B_{max}) is the spatial resolution of the interferometer at an observing wavelength λ and B_{max} is the maximum baseline length. Thus, image size i.e. the number of pixels in 1D in the image, has been calculated after substituting the following values: (FoV, B_{max}) for MWA: (20°, 3000 m), for EDA2: (180°, 35 m), at an observing frequency of 200 MHz (i.e. $\lambda = 1.5$ m).

[2] Calculated as: $O(N_a N^2_{pixels})$. See Section 1.3.

[3] Calculated as: $O(N_a(N_a - 1)) + O(N_a(N_a - 1))) + O(N^2_{pixels} log_2(N^2_{pixels}))$. See Section 1.4.

Software Applications (CASA, CASA Team et al., 2022), Multi-channel Image Reconstruction, Image Analysis, and Display (MIRIAD, Sault et al., 2011) and WSCLEAN are suitable for FRB searches, they require input data to be present in particular formats (for e.g, CASA measurement sets or `UVFITS` files), which demands an additional conversion cost, slowing down the entire process. Additionally, forming high-time resolution images in real-time requires efficient parallel software that can only be realised on GPUs. Other than the image-domain-gridding mode (IDG, van der Tol et al., 2018) in WSCLEAN, these imagers do not fully utilise the computational power of modern GPUs. Hence, this presented work involves the development of a new imager, which will process the data starting from correlation products (i.e. visibilities) and eliminate these additional I/O conversion operations. Furthermore, this imager will be operational on GPUs to speed up the imaging process by fully utilising the computational power of GPUs.

Lastly, as shown in Section 1.5.3, since the very beginning, efforts have been taken by low-frequency interferometers such as the MWA to try and detect FRBs at frequencies below 300 MHz. Though these searches successfully established limits on the FRB rates, no FRB has been discovered with the MWA (or EDA2).

Nevertheless, since 2018, CHIME has successfully detected hundreds of FRBs in the frequency band 400 to 800 MHz (see Section 1.2.1). Furthermore, signals from many of these FRBs have been observed down to 400 MHz, indicating that some FRBs can be observed even at lower frequencies.

Additionally, the detections of FRB emissions of the repeating FRB 20190916B at 110 - 188 MHz with LOFAR, making it the lowest frequency detections of any FRB to date, has further confirmed the presence of low-frequency emission from at least some FRBs and gave renewed hope for low-frequency FRB detections. This small number of low-frequency FRB detections could be likely due to fewer FRB events at low frequencies, lesser on-sky time or higher computational complexity of beamforming-based searches. Furthermore, these searches were software-limited and more efficient software for high-time resolution data from wide-field radio interferometers is needed. For example, the pilot search for FRBs using MWA in Tingay et al. (2015) took around three days to process two hours' worth of data.

Thus, the main aim of the GPU imager presented in this work is to become a part of a larger GPU-based processing pipeline searching (Di Pietrantonio et al., in preparation) for low-frequency FRBs and is a step towards making image-based searches with these low-frequency interferometers computationally feasible. Initially, the main target instruments for this imager will be the low-frequency interferometers in Western Australia, the EDA2 and the MWA, and the future versions of the imager will be applicable to data from any radio interferometer.

1.7 Structure of the Thesis

The remainder of this thesis is arranged as follows: The different steps involved in aperture synthesis and imaging in radio astronomy are discussed in Chapter 2. In Chapter 3, the initial version of the imager implemented for CPUs is described, followed by the implementation of the GPU imager for a single time-step, single-channel and single time-step having multiple frequency channels. Additionally, this chapter provides benchmarks obtained with the improvement in data processing of the GPU imager on Topaz and Setonix supercomputers. The comparisons with the initial CPU version of the imager are also provided. Chapter 4 describes a pilot FRB search and analysis using EDA2 data. The imager (starting CPU version) discussed in Chapter 3 was used to image a few hours of EDA2 data in 100 ms time resolution and approximately 0.94 MHz bandwidth (fine channelised into 32 frequency channels) to search for FRB candidates in the images. This was a pilot test search to verify the imager. The two projects presented in this work are summarised in Chapter 5.

Chapter 2

Methodology

This thesis describes a high-time resolution GPU imager for data from radio interferometers (mainly SKA-Low prototype station EDA2). Therefore, this chapter discusses the basis of radio astronomy imaging. The fundamentals of radio interferometry and imaging are explained in detail in many texts on the subject (e.g., Marr et al. (2015), Thompson et al. (1991)). In this chapter, aperture synthesis and the functioning of the two-element interferometer are discussed in Section 2.1. Section 2.2 provides an overview of this project's processing steps, such as correlation, gridding, and imaging.

2.1 Radio Interferometry

Radio telescopes are employed to detect and study the radio emission from astronomical objects. At an observing wavelength of λ, the angular resolution of a single-dish telescope having an aperture diameter D is given by λ/D. Thus, to achieve angular resolutions of a few arc-seconds, we would have to construct radio telescopes a few kilometres in diameter. The largest known single-dish telescope in the world is the Five-hundred-meter Aperture Spherical Radio Telescope (FAST). It is located in China and has a diameter of 500 metres and a standard observing frequency range between 70 MHz - 3 GHz. However, making a dish larger than FAST is hard and nearly impossible. Additionally, though FAST is valuable in performing scientific endeavours, the demand and need for an improved resolution nonetheless exists.

To attain a resolution higher than a single dish can provide, astronomers use a technique called aperture synthesis. In aperture synthesis, an array of antennas, which could be either single dishes or other types, are spread over a large area, and each pair of antennas functions as an interferometer. Thus, instead of constructing one big dish with an extremely large diameter, a set of smaller antennas acting as one big telescope is used. Their effective diameter is equivalent to the maximum distance between antennas. This is illustrated in Figure 2.1, and in the next Section 2.1.1, the functioning of the basic unit of an aperture synthesis telescope, the two-element interferometer, which consists of two antennas, is explained.

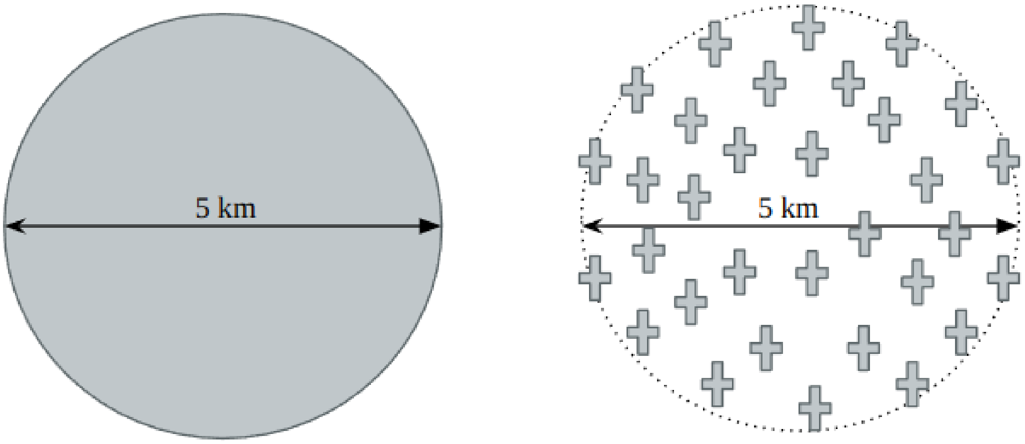


Figure 2.1: The figure shows a single dish with a diameter of 5 km (left panel) and a synthesised dish utilising 128 antennas (right panel). The maximum distance between any antennas in the imitated dish is 5 km. The area of the single dish can be calculated as $\pi R^2 = 1.963 \times 10^7 \ m^2$, and the effective area of the synthesised dish using antennas can be approximated as (128 × area of 1 antenna), which is approximately $128 \times 16 m^2 = 2048 \ m^2$, for an array like the MWA. Thus, the imitated dish still provides the same angular resolution (i.e. sensitivity) as the single dish but with a decreased effective area.

2.1.1 Two-element interferometer

Radio interferometers study the radio sky by correlating electric signals from multiple antennas (or dishes). These antennas or dishes are spread across an area and work together as a single telescope. The response of an individual antenna (or sensitivity as a function of direction in the sky) is called a *primary beam* and is typically determined analytically or using sophisticated electromagnetic simulations. In this section, the principles of a two-element interferometer (i.e. consisting of two antennas) are discussed.

A two-element interferometer, as shown Figure 2.2, consists of Antenna 1 and Antenna 2, separated by a distance, denoted by the vector $\vec{B}$ (also known as the baseline), pointed towards a source ($\vec{s}$) located at zenith angle θ. Since the source is, effectively, infinitely far away, the signal from the source approaches both antennas along parallel paths. As depicted in Figure 2.2, the radiation entering

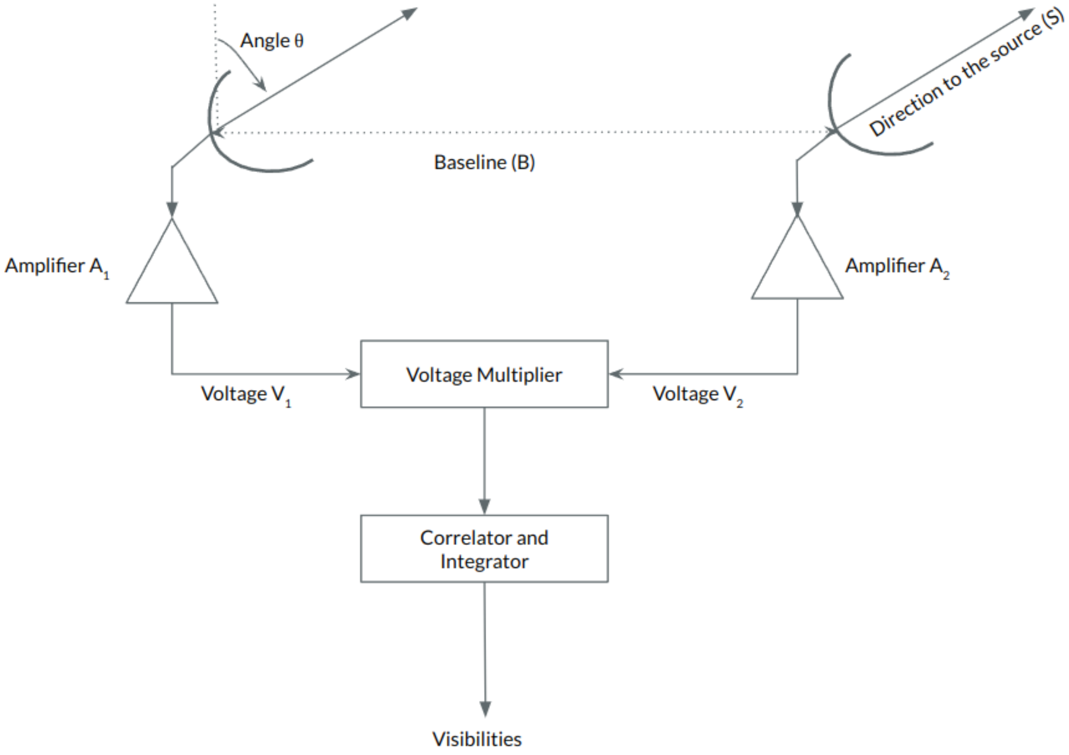


Figure 2.2: An interferometer, consisting of two antennas, separated by a distance B, observing a source at zenith angle θ.

Antenna 1 travels a longer distance than Antenna 2. Thus, the time delay (τ) corresponding to the additional light travel time can be calculated according to the following equation:

$$\tau = \frac{B \times sin(\theta)}{c} = \frac{\vec{B}.\vec{s}}{c}, \tag{2.1}$$

where B is the length of the baseline vector and c is the speed of light.

The electric fields due to the source, at any time (t), are converted to voltages (V_1, V_2) and measured at the output of the antenna terminals as:

$$\begin{aligned} V_1 &\propto E_1(t) = Ee^{i2\pi\nu t} \\ V_2 &\propto E_2(t) = Ee^{i2\pi\nu(t-\tau)}, \end{aligned} \tag{2.2}$$

where ν is the observing frequency, E is the amplitude of the electric field, $E_1(t)$ is the electric field incident at the antenna 1, $E_2(t)$ is the electric field incident at the antenna 2 and t is time.

The signals are then multiplied and time-averaged (represented as '<>' in the below Equation 2.3) by the correlator. The following equation describes the output of the correlator:

$$< V_1.V_2^* >= \frac{1}{t_{int}} \int_0^{t_{int}} E_1(t)E_2^*(t)dt \quad , \tag{2.3}$$

where $E_2^*(t)$ is the complex conjugate of $E_2(t)$.

Equation 2.3 can be further simplified as:

$$\begin{aligned} < V_1.V_2^* > &= \frac{1}{t_{int}} \int_0^{t_{int}} Ee^{i2\pi\nu t}Ee^{-i2\pi\nu(t-\tau)}dt \\ < V_1.V_2^* > &= \frac{1}{t_{int}} \int_0^{t_{int}} E^2e^{i2\pi\nu\tau}dt \\ < V_1.V_2^* > &= \frac{1}{t_{int}} E^2e^{i2\pi\nu\tau} \int_0^{t_{int}} dt \\ < V_1.V_2^* > &= E^2e^{i2\pi\nu\tau} \end{aligned} \tag{2.4}$$

The expression denoted as Equation 2.4 represents the complex correlation between the signals, as observed by a single baseline. Similarly, signals from all the other baselines are multiplied and integrated over a certain interval to generate correlation products, or '*visibilities'*. Thus, Equation 2.4, can be also expressed as:

$$V_{12}(\nu) = E^2 e^{i2\pi\nu\tau}. \tag{2.5}$$

As shown in Figure 2.3, visibility contribution from a small solid angle of the sky is given by:

$$dV_{12}(\nu) = A(\vec{s}) I_\nu(\vec{s}) e^{i2\pi\nu\tau}\, d\Omega\, d\nu, \tag{2.6}$$

where $\vec{s}$ points to a specific part of the source, $d\Omega$ is the solid angle within the source emitting the radio waves, $A(\vec{s})$ is the effective area of the antenna, $I_\nu(\vec{s})$ is the spectral brightness emitted from the solid angle $d\Omega$ per unit area per unit of frequency ν and τ is a function of $\vec{s}$ (see Equation 2.1). Using this visibility for a single source, we can further integrate contributions from the entire sky. When we integrate over the entire source, we obtain visibility V_{12}, given by:

$$V_{12}(\nu) = \int_{Source} A(\vec{s}) I_\nu(\vec{s}) e^{i2\pi\nu\tau}\, d\Omega\, d\nu. \tag{2.7}$$

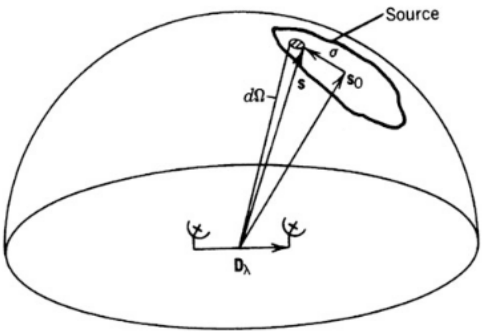


Figure 2.3: The interferometer's baseline and the source's position vectors are shown. The outline of the celestial sphere denotes the source.
Image credits: (Thompson et al., 1991).

Using Equation 2.1, visibility $V_{12}(\nu)$ can expressed in terms of baseline coordinates $V_{12}(\vec{D_\lambda})$ as:

$$V_{12}(\vec{D_\lambda}) = \int_{Source} A(\vec{s}) I_\nu(\vec{s}) e^{i2\pi(\vec{D_\lambda}.\vec{s})} d\Omega \; d\nu, \tag{2.8}$$

Expressing $\vec{s} = \vec{s_0} + \sigma$, where $\vec{s_0}$ points to the centre region (also known as the phase centre) as shown in Figure 2.3, Equation 2.8 can be reduced to:

$$\begin{aligned} V_{12}(\vec{D_\lambda}) &= \textstyle\int_{Source} A(\vec{\sigma}) I_\nu(\vec{\sigma}) e^{i2\pi(\vec{D_\lambda}.(\vec{s_0}+\vec{\sigma}))} d\Omega \; d\nu, \\ V_{12}(\vec{D_\lambda}) &= e^{i2\pi\vec{D_\lambda}.\vec{s_0}} \textstyle\int_{Source} A(\vec{\sigma}) I(\vec{\sigma}) e^{i2\pi(\vec{D_\lambda}.\vec{\sigma}))} d\Omega \; d\nu, \end{aligned} \tag{2.9}$$

To track the phase centre, the visibilities are phase-shifted (i.e. multiplied by the term $\mathrm{e}^{-i2\pi\vec{D_\lambda}.\vec{s_0}}$), which cancels the plane wavefront term. Thus, visibility can be expressed as:

$$V_{12}(\vec{D_\lambda}) = \int_{Source} A(\vec{\sigma}) I(\vec{\sigma}) e^{i2\pi(\vec{D_\lambda}.\vec{\sigma})} d\sigma \; . \tag{2.10}$$

Equation 2.10 can be identified as a Fourier transform relation between the measured visibilities (V_{12}) and the brightness distribution of the source (I_ν). From this Equation 2.10, we can see that Visibility corresponds to a given baseline. This Fourier transform relation is key to imaging, further explained in Section 2.2. It is important to note that shorter baselines of a telescope are more sensitive to the finer structures of the image, while longer baselines are more sensitive to the larger structures, and a discussion is further done in Section 2.2.2.

2.1.2 The *uvw* plane

Applying the Fourier transform relation between visibilities and the brightness distribution as in Equation 2.10 to create images from the radio telescopes requires a different coordinate system. In this new coordinate system, i.e. the *(uvw)* coordinate system, u and v are calculated in a plane normal to the direction of the phase reference position ($\vec{w}$ axis). Here, u points in the East, v points in the direction of the North celestial pole, and w points in the direction of the source's phase centre (s_o). Here, u is measured towards the East, v is measured towards the North as defined by the plane through the origin, the source and the pole and w is measured in the direction of the source's phase centre (s_o). Figure 2.4 illustrates the geometry of this *uvw* coordinate system.

The position of the source under observation is represented in *(l,m,n)*, which are the direction cosines measured from the u, v and w axes, respectively, and are given by:

$$\begin{aligned} l &= \cos\alpha \\ m &= \cos\beta \\ n &= \cos\theta = \sqrt{1 - l^2 - m^2}, \end{aligned} \tag{2.11}$$

where the angles α, β and θ are between $\vec{s}$ and the u, v and w axes respectively. This is illustrated in Figure 2.5.

Using *(u,v,w)* and *(l,m,n)* coordinates, we find:

$$\begin{aligned} &\vec{D_\lambda}.\vec{s_\circ} = w, \\ &\vec{D_\lambda}.\vec{\sigma} = ul + vm + w\sqrt{1 - l^2 - m^2}, \\ &d\Omega = \frac{dldm}{\sqrt{1 - l^2 - m^2}}, \end{aligned} \tag{2.12}$$

where $\sqrt{1 - l^2 - m^2}$ is equal to n, the third direction cosine measured with respect to the w axis. Thus, Equation 2.10 can be re-written in terms of *(u,v,w)* and *(l,m,n)* coordinates as:

$$V(u, v, w) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} A(l, m) I(l, m) e^{i2\pi(ul+vm+wn)} \frac{dl\,dm}{n}, \tag{2.13}$$

Equation 2.13 is a practical Fourier transform formula for the visibilities in terms of baselines and wavelengths where every point in the *(l,m)* plane has a

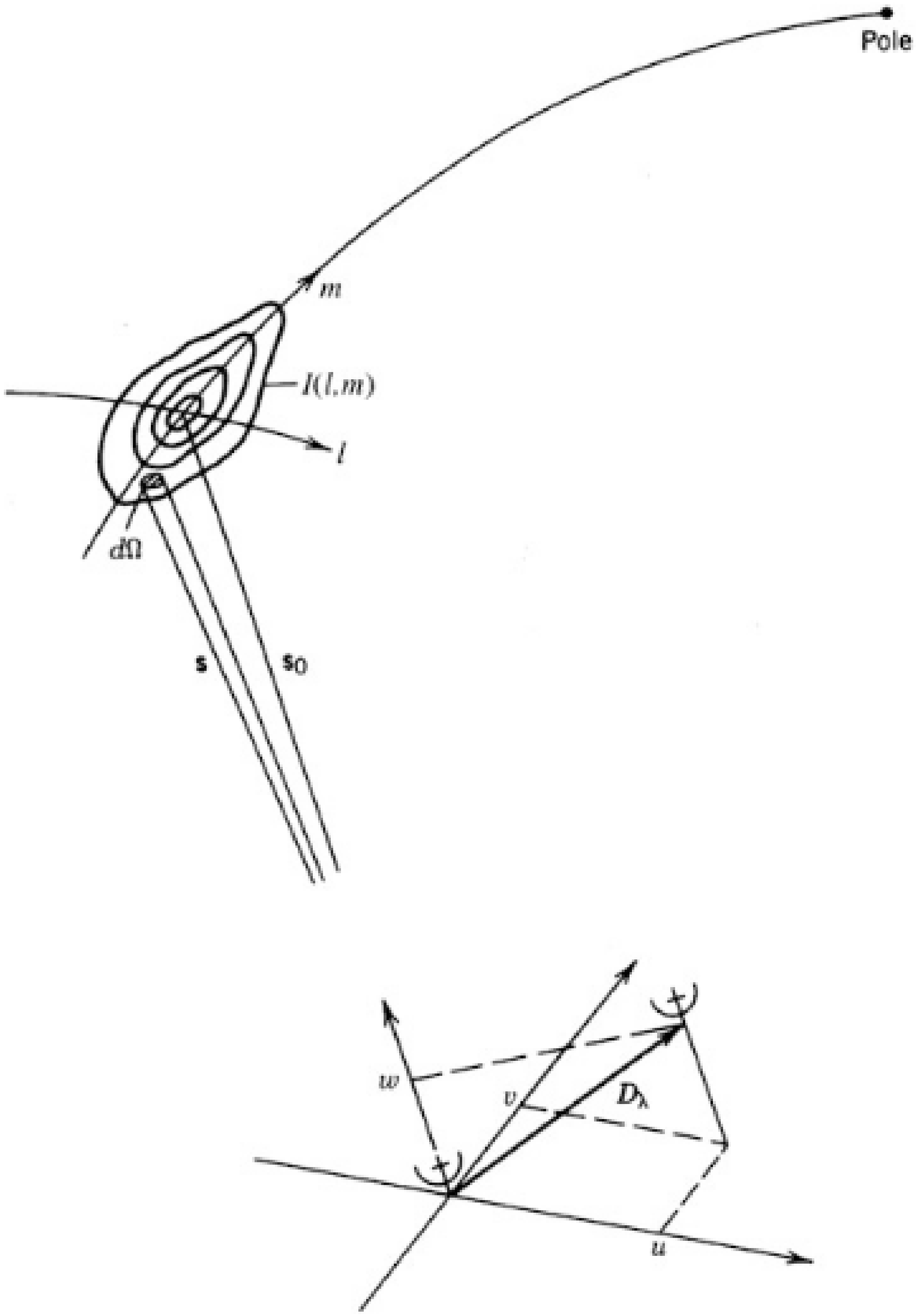


Figure 2.4: Figure shows the *(u,v,w)* coordinate system. An antenna pair observe the source with a baseline $\vec{D_\lambda}$. Here, $\vec{s_0}$ is in the direction of the w axis pointing towards the phase centre.
Image credits: (Thompson et al., 1991).

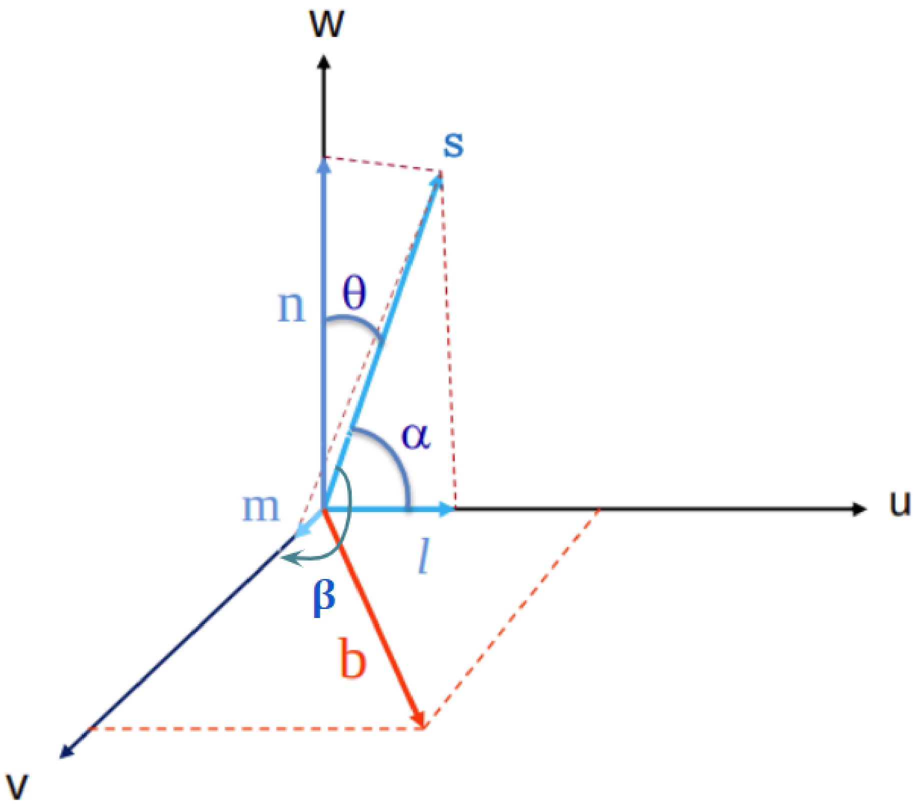


Figure 2.5: The vector $\vec{s}$ is defined by its projections *(l.m.n)* on the *(u,v,w)* axes. The *(l.m.n)* components are called the 'Direction Cosines'. The angles α, β and θ are between $\vec{s}$ and the u, v and w axes respectively. Here, $\vec{b}$ is the baseline vector.
Image credits: Lectures taken from ATNF Radio School, Narrabri, October 2014.

corresponding data point in the (u,v) plane. The w term can be neglected for small FoVs, and n = $\cos\theta$ can be approximated to 1. Similarly, for a co-planar array (like the EDA2) and an all-sky image with phase centres at the zenith, where the w terms are nearly exactly = 0, resulting in $e^{i2\pi w} \approx 1$. Under these approximations, the integral in Equation 2.13 becomes a two-dimensional inverse Fourier transform of the measured visibilities. This means that in this case, it is the inverse Fourier transform of the measured visibilities, expressed as a function of baseline coordinates (u,v) measured in wavelengths, that gives the sky intensity distribution, i.e. the image of the source under observation. Thus, Equation 2.13

can be approximated as:

$$V(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} A(l,m)I(l,m) \times e^{i2\pi(ul+vm)}\, dl\, dm. \tag{2.14}$$

This relation is also known as the van Cittert-Zernike theorem. This theorem is based on a study published by van Cittert in 1943, followed by a simpler derivation by Zernike a few years later. A detailed description of the results established by van Cittert and Zernike is given in Born & Wolf (1959). This approximation holds for all-sky imaging with the SKA-Low stations. This makes the all-sky imaging using 2D Fourier transform with SKA-Low stations exact without any need for approximation as w terms are zero within a very small fraction of a wavelength (≈ 0.1 or better) for a small co-planar array like EDA2 (35m diameter).

An important point to note here is that, in the case of wide-field imaging, like that with the MWA, which has a wide FoV, the curvature of the sky and the non-coplanar baselines will result in a non-vanishing w-term. Hence, the w-term cannot be approximated to 0 in these cases. Thus, applying a standard 2D Fourier transform to visibility data obtained from non-coplanar radio interferometers having a wide FoV (such as the MWA) will radially smear the sources that are away from the phase centre (image centre), resulting in a reduced signal-to-noise (S/N) detection of the source. Hence, the imaging algorithm needs to account for the w-term to prevent these imaging artefacts. There are several ways to account for and correct the w-term effect during imaging, such as: faceting (Cornwell & Perley, 1992), a 3D Fourier transform (Taylor et al., 1999), w-stacking (Humphreys & Cornwell, 2011), warped snapshots (Taylor et al., 1999), w-snapshots (Cornwell et al., 2012), and w-snapshot imaging (Offringa et al., 2014) and detailed descriptions of these methods are outside the scope of this thesis. However, at this stage, in the GPU imager described in this work, the w-term has not been taken into account. Although this will affect parts of the image away from the phase centre, leading to some S/N loss, it is not yet known if this is acceptable or not. This will be assessed and adding suitable corrections will be considered against the additional computing requirements of this step. Additionally, at this stage, the impact of not considering the w-terms on the sensitivity and transient/FRB searches away from the phase centre is not been investigated. This assessment will be considered in future upgrades, along with accounting for the w-term in the imaging algorithm. WSCLEAN is most commonly used for imaging with MWA data, and it will be the main reference imaging software for MWA images. Future plans include comparing the sensitivity of images obtained with WSCLEAN to those produced by the GPU imager and comparing the sensitivity and quality of images away from the phase centre. This comparison will help to understand the impact on transient and FRB searches. Additionally, as outlined in Section 1.6, the ultimate goal of the GPU imager discussed in this work is to process data from MWA and the SKA-Low stations for the purpose of searching for FRBs. Ideally, the best-case scenario is to develop imaging software that is as fast as possible and is capable of processing large volumes of data, thereby increasing the likelihood of detecting FRBs, which are typically one-off events. The goal is to produce images fast, and this involves a trade-off between speed and image quality.

Next, in Section 2.2, using Equation 2.14, an overview of imaging implementations in a computer program starting from the visibilities obtained from a radio interferometer is provided.

2.2 Imaging in Radio Astronomy

Imaging of radio data is the process of creating an image of the sky at radio frequencies, starting from voltages recorded by multiple antennas in a radio telescope. The first step involves multiplying and integrating signals (over a particular time interval) from all the antenna pairs. This step generates ‘correlation products’ (i.e. visibilities). For a telescope with N_a antennas, only considering single polarisation, there are N_a auto-correlation (i.e. products obtained by the correlation of a signal with itself) and $N_a(N_a - 1)$ cross-correlation products (i.e. products obtained by the correlation of different signals), respectively, which form the inputs for imaging. An illustration of a correlation matrix is shown in Figure 2.6. In practice, only cross-correlation products are used in the standard imaging process. Auto-correlation products may be used, but only in some special-purpose imagers. The imager described in this thesis has an option of including auto-correlation products.

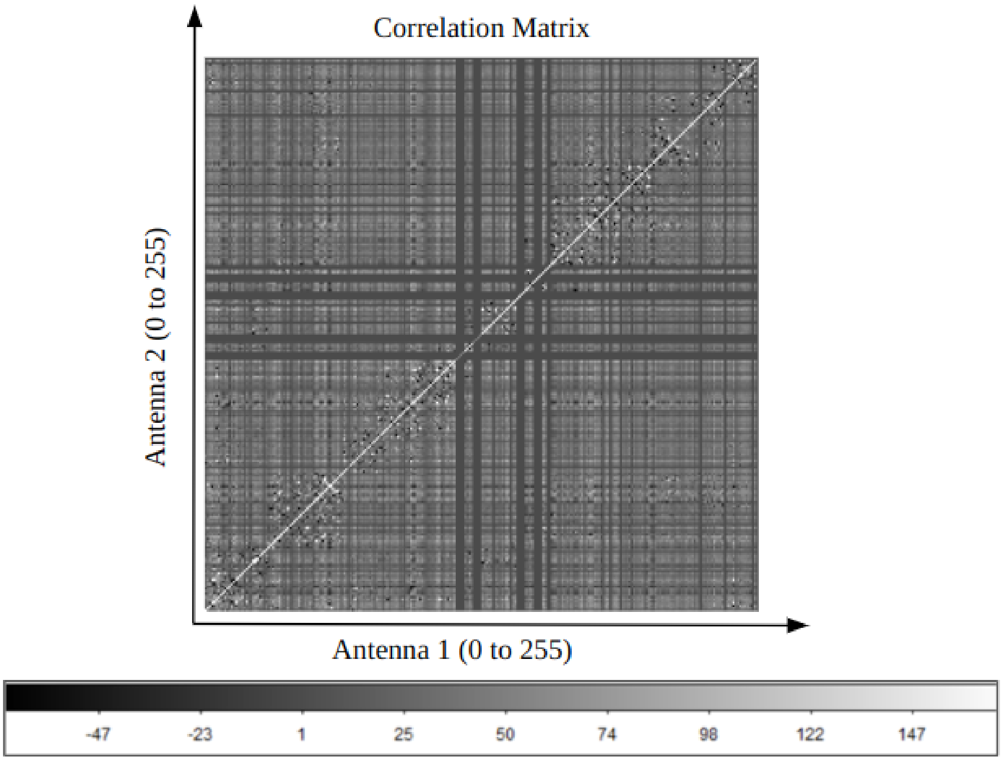


Figure 2.6: The real value of visibilities is shown in a correlation matrix, which forms the input for imaging, where the auto-correlation products are visible along the diagonal. The correlation matrix is Hermitian, and hence the real part of the matrix is symmetric. The visibilities are obtained from EDA2 data at approximately 166 MHz. The x and y axis are the 256 EDA2 antennas numbered from 0 to 255.

An image is created for a given frequency and a given point in time from the visibilities obtained from all the baselines (i.e. unique antenna-pair combinations), excluding the bad antennas (i.e. flagged antennas). To be able to apply Equation 2.14 and obtain an image of the sky from the visibilities, they have to be expressed as a function of baseline coordinates (u,v,w), present in a right-handed coordinate system measured in wavelengths (see Section 2.1.2). From Equation 2.14, we know that each visibility is related to the sky brightness distribution, i.e. the image of the sky through the van Cittert-Zernike theorem. To calculate what is called a *'dirty image'* of the sky, a 2D Fourier transform is applied to the visibilities to calculate $I_{dirty}(l, m)$ from $V(u, v)$ which transforms data in the (u,v) plane to sky brightness in the image plane. In practice, this is done by the computer using the Fast Fourier Transform (FFT) algorithm (Cooley & Tukey, 1965), which is the most efficient way to do this. It has a computational complexity of $O(NMlog(NM))$, where N and M are the dimensions of the (u,v) grid. The FFT algorithm, however, requires the input data (in this case, the complex visibilities) to be placed on a regularly spaced grid in the (u,v) plane. Therefore, the input visibilities must be 'gridded' to such a regular grid on the (u,v) plane, which is further explained in Section 2.2.1. After visibilities are gridded on the (u,v) plane, 2D FFT is applied, resulting in the sky image (i.e., 'dirty image').

2.2.1 Gridding

Gridding is the process of re-sampling data (i.e. here visibilities) for FFT by placing the visibilities on a grid of regularly spaced cells of dimensions Δu × Δv (see Figure 2.7), which are dependent on the size and dimensions of the desired output sky image. In the simplest version of gridding, visibilities are added to the closest cell in the (u,v) grid. This section will discuss the guidelines that were followed to choose the gridding parameters and image dimensions for gridding.

The output sky image is often an $N_x \times N_y$ square array of pixels (i.e. $N_x = N_y$), with each pixel having an angular width and height, of Δx and Δy respectively, and often being a square (i.e. Δx = Δy). Thus, the angular dimensions of the entire sky image are $N_x \Delta x \times N_y \Delta y$. The longest baselines provide the finest angular resolution, i.e. enable the ability to recover the finest structures of the radio sources. Hence, Δx and Δy should be chosen such that they satisfy the

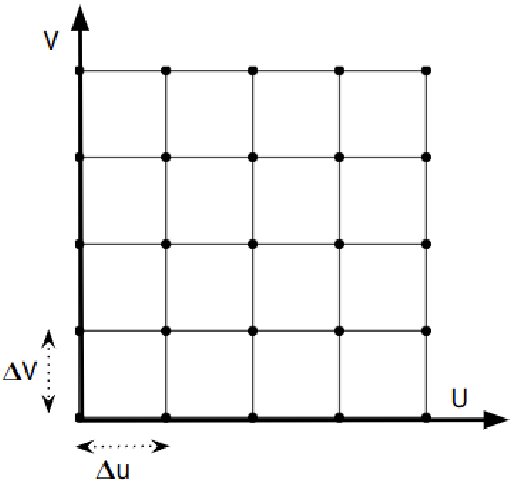


Figure 2.7: Figure depicts the dimensions of the (u,v) grid, Δu and Δv, as calculated using Equation 2.16. They are the reciprocals of the angular dimensions of the sky image: N_xΔx, N_yΔy, respectively.

Nyquist sampling theorem (Shannon, 1949) for the longest baselines (i.e. u_{max} and v_{max}). Nyquist sampling theorem states that to reconstruct a signal from its sampled version correctly, the sampling rate must be a minimum of twice the highest frequency present in the signal. Hence, dictated by the Nyquist sampling theorem, we need at least two pixels across a single-resolution element i.e. *the synthesized beam*. The *synthesized beam pattern* represents the measure of the spatial resolution of an interferometer. This synthesized beam is smaller than the primary beam, which is the response of an individual antenna (see Section 2.1.1). The angular width between the points where the received power falls to half the peak value is called the full width at half maximum (FWHM) of the synthesized beam. Hence, in order to satisfy the Nyquist theorem, we must sample at least 2 pixels across the FWHM of the synthesized beam so that Δx and Δy satisfy the following criteria:

$$\Delta x < \frac{1}{2u_{max}} \quad \text{and} \quad \Delta y < \frac{1}{2v_{max}}. \tag{2.15}$$

However, in practice, 3-5 pixels across the FWHM of the beam are used as over-sampling is useful in the mapping process. Likewise, the dimensions of a single cell in the (u,v) grid will be determined by the maximum angular dimensions of the sky image given by:

$$\Delta u = \frac{1}{N_x \Delta x} \quad \text{and} \quad \Delta v = \frac{1}{N_y \Delta y}. \tag{2.16}$$

In summary, the main steps leading from voltages recorded by individual antennas to images of the sky are summarised in Figure 2.8.

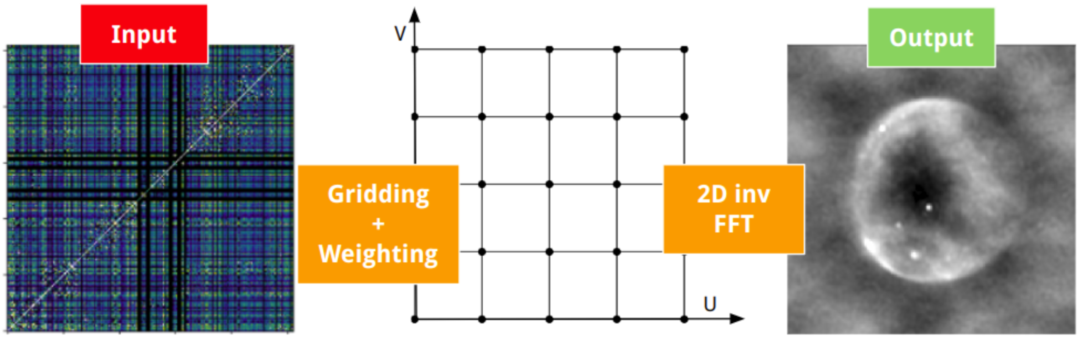


Figure 2.8: Figure shows the three main steps involved in imaging, namely: (i)correlation, (ii)gridding, and (iii)Fourier transform. The visibilities, in the form of a correlation matrix, form the input for imaging. These visibilities are gridded, and a 2D inverse FFT is performed on the gridded visibilities to produce the output sky image (i.e. 'BLINK' image generated by the CPU imager; see Section 3.1.1).

2.2.2 Weighting

Visibilities are gridded such that each cell in the (u,v) grid satisfies one of the three conditions: (i) if there is one visibility corresponding to a cell, the cell will have that value (altered if weighting other than natural is applied); (ii) if no visibilities correspond to a cell, the cell will have a zero weight; (iii) if multiple visibilities are assigned to a cell, the visibilities are summed with some weighting and assigned to the corresponding cell. The choice of the weighting scheme (i.e. natural, uniform etc) is in the user's control. The GPU imager described in this work has both natural and uniform weighting implemented.

In natural weighting, all visibility values receive the same weight, implying that cells with more than one visibility point will contribute more to the final output image than cells with a single visibility. Interferometers typically have more short than long baselines, resulting in more visibility data from smaller (u,v) distances. Since all data receive the same weight, natural weighting tends to favour this shorter baseline data, which can decrease the image resolution. On the other hand, since all visibility values receive the same weight, natural weighting minimises system noise[1] and generates images with minimal system noise. Hence, this weighting scheme improves the image sensitivity. This makes it particularly useful for imaging faint radio sources.

In uniform weighting, the same weight is applied to all cells in the (u,v) grid irrespective of the number of visibilities they contain. If there are multiple values per cell, they are averaged to make the cell weights the same. This prevents any bias towards specific data and avoids the domination of particular regions of the (u,v) plane over others. This results in increased image resolution. However, the disadvantage of uniform weighting is that it leads to higher system noise. In this approach, a (u,v) cell containing a single visibility is given the same weight as a (u,v) cell with multiple visibilities. Hence, uniform weighting is particularly advantageous when creating maps of strong sources where the S/N does not pose a limiting constraint. Furthermore, typically, there will always be more short baselines than longer baselines, which is why uniform weighting will always give higher-resolution images, and natural weighting will always correspond to lower-resolution images.

While performing an inverse 2D FFT, not all the (u,v) cells contain visibility data. This leads to incomplete sampling in the image domain; hence, the imaging algorithm needs to interpolate to fill these gaps. This interpolation involves estimating the value of missing pixels based on surrounding data, which results

[1] Though the contribution from receiver noise is minimised, confusion noise and sidelobes from confusing sources are usually higher at low frequencies. Thus, this has caveats that confusion noise is higher in this weighting.

in an imaging error. These obtained images are called 'dirty images' because they are also affected by the interferometer's 2D response to the sky. In the following Section 2.2.3, an overview of the (u,v) coverage concept and the process of reconstructing the '*true image*' of the sky, using the dirty images as a starting point will be provided.

2.2.3 The Dirty beam

The interferometer as a whole has a beam pattern on the sky, which is determined by the combination of all the baselines together, called the *synthesized beam*. If the (u,v) sampling function of the interferometer is defined as $S(u, v)$, then the sampled visibilities $V_s(u, v)$ can be defined as:

$$V_s(u, v) = V(u, v)S(u, v). \tag{2.17}$$

Thus, from Equations 2.14 and 2.17, the dirty image I_{dirty}*(l,m)* can be reduced to:

$$I_{dirty}(l, m) = \mathcal{F}\{V(u, v)S(u, v)\}, \tag{2.18}$$

where the Fourier transform operation is denoted by $\mathcal{F}$. Using the convolution theorem: ($\mathcal{F}\{AB\} = \mathcal{F}\{A\} \circledast \mathcal{F}\{B\}$), Equation 2.18, can be re-written as:

$$I_{dirty}(l, m) = \mathcal{F}\{V(u, v)\} \circledast \mathcal{F}\{S(u, v)\}. \tag{2.19}$$

Equation 2.19 can be further reduced to:

$$I_{dirty}(l, m) = I_{true}(l, m) \circledast s(l, m). \tag{2.20}$$

Here, $I_{true}(l, m)$ is the true sky image and $s(l, m)$ is the Fourier transform of the sampling function (i.e. the data points in the (u,v) plane where visibility measurements have been taken), also known as the *dirty beam* or the Point Spread Function (PSF). The PSF is thus the instrument's 2D response to the sky. In

Figure 2.9, three examples of the (u,v) coverage and the corresponding PSFs obtained with Phase 1, Phase 2 compact and extended configurations of the MWA are shown.

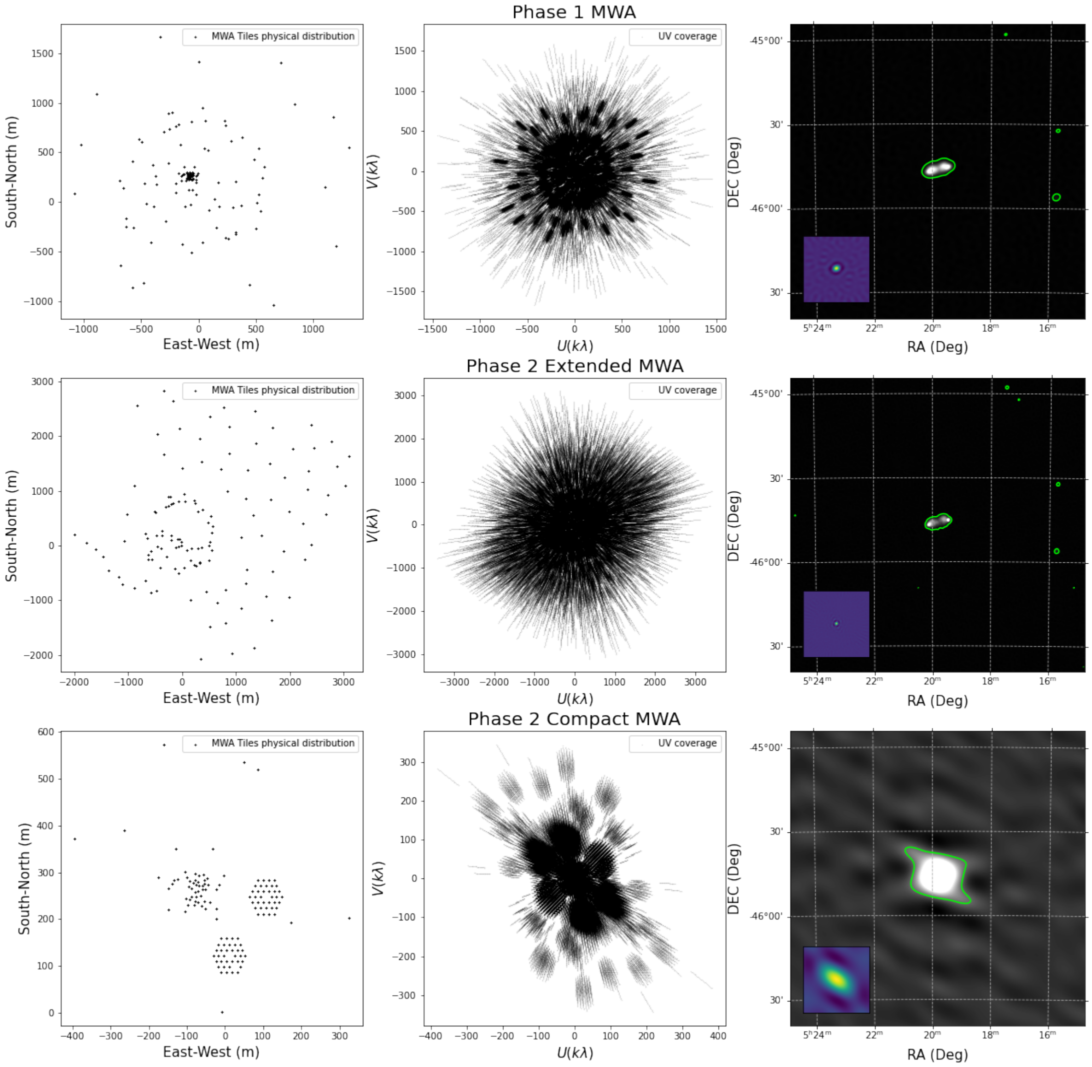


Figure 2.9: The layout and the instantaneous (u,v) coverage of MWA Phase 1 (top panel), MWA Phase 2 extended (middle panel) and MWA Phase 2 compact (bottom panel) configurations. The corresponding PSFs are in the insert panel of the reconstructed image.
Image credits: (Prabu, 2021).

From Equation 2.19, it is also evident that the dirty image is the convolution of the sky intensity (i.e. true sky image) with the PSF of the telescope. Thus, de-convolution has to be performed in order to *deconvolve* the PSF pattern from the dirty image and obtain the true image of the sky. This procedure of deconvolution using CLEANing algorithms has been discussed in Section 2.2.4. However, in the GPU imager discussed throughout this thesis, only dirty images were generated by the GPU imager, and no CLEANing process was applied. The decision to implement deconvolution (e.g. CLEAN algorithm) will be made after more tests of the imager for transient/FRB searches are performed. Deconvolution may be introduced in future versions if it is deemed required by the transient/FRB searches and/or to make the imager useful for a wider audience. This is also discussed in Section 2.2.4.

2.2.4 De-convolution

The process of reconstructing/recovering the true sky image from the dirty image is called de-convolution. Various approaches are available; in this section, the most popular CLEAN algorithm (Högbom, 1974), which assumes the dirty image to only have point sources, is discussed. It is an iterative process, and the main steps are:

1. The brightest pixel (delta function) in the dirty image is identified, and its amplitude and location are stored in a list of *clean components*.

2. The visibility model of this identified delta function is then subtracted from the observed visibilities.

3. This procedure is repeated many times in the image until there are no longer any pixels exceeding the specified threshold. Having subtracted the model of every detected delta function, the resulting image ideally only contains noise-like structures and is called the *residual map*.

4. The list of clean components is then convolved with a 2D Gaussian model of the PSF (also called the restoring beam) and then added to the residual noise image. This is called the *clean image*.

CLEANing is generally performed when there is more than one source in a given image to reduce the noise from other sources. In this case, using the GPU imager described in this work (see Chapter 3), images will be generated in 100 ms time resolution. At this time resolution, we do not expect to see anything except transient events/FRBs (which are point sources) and not more than a few point sources in an image. Hence, imaging diffuse or extended sources was not a priority, as the primary goal of the GPU imager was to find transients and FRBs, which are point sources. Moreover, CLEANing is computationally expensive and a time-consuming process. Also, when looking for transients such as FRBs, the amplitude of the event is of primary concern rather than its shape. Hence, artefacts from the PSF may not matter that much. They may only matter when they produce spurious (false) sources. Additionally, in such short, noise-dominated images, difference imaging may be sufficient to remove the artefacts. This will be explored and tested on real data, and if it is not sufficient, deconvolution will also be implemented on GPU. The current approach is to avoid implementing deconvolution/CLEANing as this requires more compute time and may not be absolutely necessary to identify FRBs and other transients. Hence, at this stage, the high-time resolution GPU imager described in this work is fairly simple and forms only the so-called dirty images. However, it may be implemented later depending on future requirements (e.g. search for longer timescale transients) to make the imager a more general tool useful for a wider range of projects. If alternate methods (e.g. difference imaging) do not work in future versions of the GPU imager, we may consider implementing a different, faster version of CLEAN (on GPU).

2.3 Summary

In summary, this chapter discussed the basics of radio interferometry and the functioning of a two-element interferometer. The different steps involved in radio astronomy imaging, namely correlation, gridding, weighting and imaging, were also explained. The chapter concluded with an overview of CLEANing and an explanation of why only dirty images were generated and why cleaning was not yet implemented in the GPU imager developed in this work.

Chapter 3

‘BLINK’ GPU Imager for low-frequency radio telescopes

The main goal of this work is to develop a preliminary version of a high-time resolution imager that is operational completely on GPUs. This version will eventually evolve and be incorporated into an FRB searching pipeline. Generally, FRB searches involve beamformed data, similar to single pulse searches from pulsar observations. However, applying this approach to data from low-frequency radio telescopes can incur substantial computational costs. Therefore, employing image-based searches could be a better option. Additionally, though existing imagers such as CASA, MIRIAD and WSCLEAN are suitable for FRB searches, they require their inputs to be present in particular formats, which demands an additional conversion cost. Hence, in this work, a new high-time resolution imaging software was developed.

A detailed description of the FRB field and the basics of radio astronomy imaging are given in Chapter 1 and 2, respectively. In this chapter, the test GPU implementations of the ‘BLINK’ imager (i.e. test GPU imager for a single time-step with a single frequency channel and test GPU imager for a single time-step with multiple frequency channels), generating all-sky images from EDA2 data starting from my primary supervisor, Dr Marcin Sokolowski’s ‘BLINK’ CPU imager as a starting point, are discussed. Section 3.1 describes the starting CPU version of the imager for a single time and frequency channel. The description of the starting CPU and the latest GPU versions of the imager are given in Sokolowski et al. (2024a). Starting with this CPU imager, in this work, I converted it to a GPU version for NVIDIA GPU hardware and Computer Unified Device Architecture (CUDA[1]) programming environment, where the code in CUDA was written in `C++`. The GPU imager was implemented, tested, and benchmarked on Topaz[2](now decommissioned) and Setonix[3], supercomputers at the Pawsey Supercomputing Centre in Perth, Western Australia In Section 3.2, the implementation of the GPU imager for a single time step and frequency channel is given. The description of the multi-channel, multi time-step version of the GPU imager is given in Section 3.2.3. The chapter ends with the benchmarks of the GPU imager on Topaz and Setonix in Sections 3.3.2, 3.3.3 and 3.3.4.

3.1 Original CPU Imager

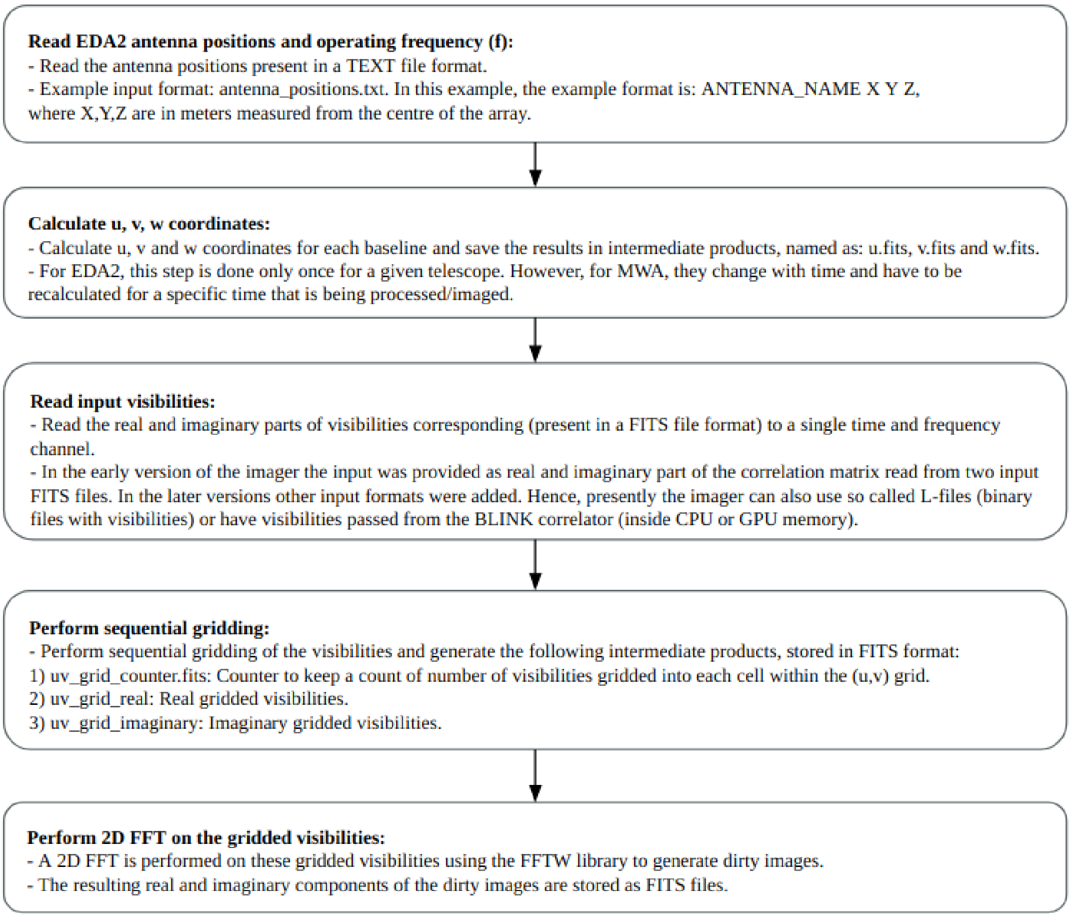
3.1.1 CPU imager

The first version of the imager was designed to create images of the entire visible hemisphere using visibilities from SKA-Low stations (mostly EDA2). The inputs to this initial version were: (i) the EDA2 visibilities corresponding to a single timestep and a single frequency channel present in a 'Flexible Image Transport System (FITS)' file format and (ii) EDA2 antenna positions present in a TEXT file format. The outputs of this imager were the real and imaginary parts of the dirty image, present in a FITS file format. The intermediate products were the (u,v) grid counter and the real and imaginary gridded visibilities. The whole block diagram showing the main steps in this CPU version of the imager is shown in Figure 3.1. Initially, this imager used CPU for both gridding and imaging, and the

[1] http://nvidia.com/cuda
[2] https://pawsey.org.au/topaz/
[3] https://pawsey.org.au/systems/setonix/

Fourier transform was performed using the `fftw`[4] library. This imager generated all-sky images for a single time step and frequency channel. A detailed description of how imaging is done in radio astronomy is detailed in Chapter 2. The imager was also verified and tested on real and simulated data from the EDA2 and MWA telescopes. This validation was carried out by the software author. Different imagers were used to validate the BLINK CPU imager with real and simulated data. CASA and MIRIAD were used to validate the BLINK CPU imager using real data and WSCLEAN was used for validation with simulated data. Initially, real EDA2 data was used for testing and nearly all EDA2 imaging was done with MIRIAD. Hence, MIRIAD was a natural choice of reference software for validation comparison for the real EDA2 data. Similarly, WSCLEAN is widely used to image MWA data; hence, this was a choice of reference software for validation comparison for the simulated MWA data. A detailed description of the validation of the CPU imager on real and simulated data is found in Sokolowski et al. (2024a) and is briefly described in Sections 3.1.2 and 3.1.3.

[4] https://www.fftw.org/



Read EDA2 antenna positions and operating frequency (f):
- Read the antenna positions present in a TEXT file format.
- Example input format: antenna_positions.txt. In this example, the example format is: ANTENNA_NAME X Y Z, where X,Y,Z are in meters measured from the centre of the array.

Calculate u, v, w coordinates:
- Calculate u, v and w coordinates for each baseline and save the results in intermediate products, named as: u.fits, v.fits and w.fits.
- For EDA2, this step is done only once for a given telescope. However, for MWA, they change with time and have to be recalculated for a specific time that is being processed/imaged.

Read input visibilities:
- Read the real and imaginary parts of visibilities corresponding (present in a FITS file format) to a single time and frequency channel.
- In the early version of the imager the input was provided as real and imaginary part of the correlation matrix read from two input FITS files. In the later versions other input formats were added. Hence, presently the imager can also use so called L-files (binary files with visibilities) or have visibilities passed from the BLINK correlator (inside CPU or GPU memory).

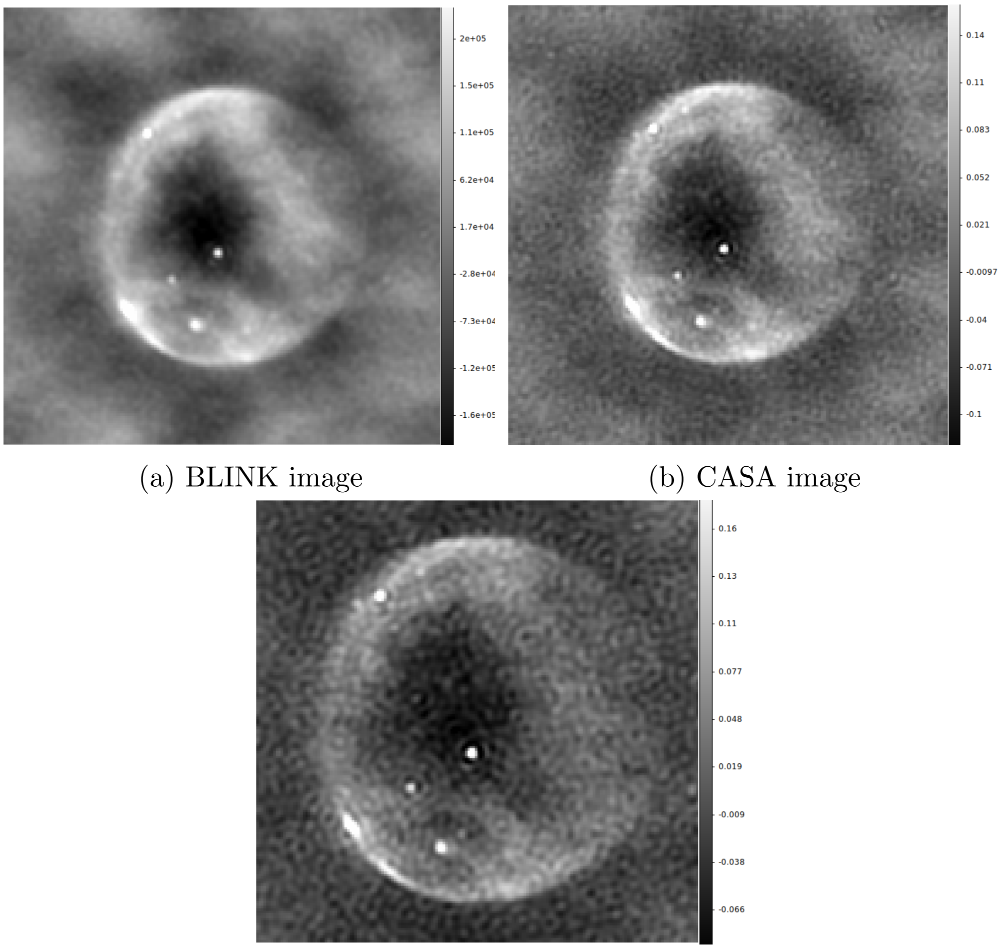
Perform sequential gridding:
- Perform sequential gridding of the visibilities and generate the following intermediate products, stored in FITS format:
1) uv_grid_counter.fits: Counter to keep a count of number of visibilities gridded into each cell within the (u,v) grid.
2) uv_grid_real: Real gridded visibilities.
3) uv_grid_imaginary: Imaginary gridded visibilities.

Perform 2D FFT on the gridded visibilities:
- A 2D FFT is performed on these gridded visibilities using the FFTW library to generate dirty images.
- The resulting real and imaginary components of the dirty images are stored as FITS files.

Figure 3.1: Block diagram of the main steps in the CPU version of the imager, for a single timestep, having a single frequency channel. At every stage of the GPU imager, writing all these intermediate FITS files is only required for diagnostics and testing and can be turned off. The final version is expected to minimise I/O operations and just read the input and write the final output (images and FRB/transient candidates)

3.1.2 Validation on real data

The resulting CPU image (referred to as the BLINK image, shown in Figure 3.2a), which was generated using real EDA2 data, was compared to images produced by the widely used imagers such as CASA and MIRIAD using options corresponding to our imager (natural weighting, no cleaning etc.). These images are shown in Figure 3.2. More details about the validation process can be found in Section 5.1.1 of Sokolowski et al. (2024a). The next step was to validate this imager using



(a) BLINK image

(b) CASA image

(c) MIRIAD image

Figure 3.2: CPU imager validation using real EDA2 data: Comparison of all-sky images from EDA2 visibilities generated by the CPU ‘BLINK’ imager, CASA and MIRIAD. The image shows the entire visible hemisphere at 160 MHz. All the images are of size 180 × 180 pixels, dirty images in natural weighting.

simulated data, described in Section 3.1.3.

3.1.3 Validation on simulated data

In the next step, the imager was validated on simulated data. EDA2 visibilities were simulated using MIRIAD, which were imaged using both MIRIAD and the BLINK CPU imager. MWA visibilities for the Hydra-A radio galaxy were simulated using CASA tasks `simobserve` starting with MWA’s antenna positions and the model image of the Hydra-A radio galaxy from the VLA Low-frequency

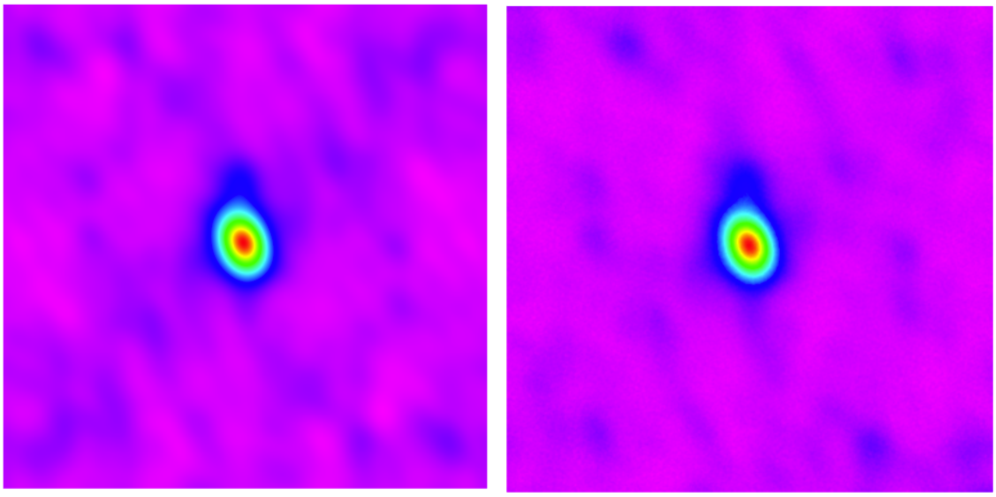


Figure 3.3: CPU imager validation using simulated MWA visibilities: A comparison of the images produced by the BLINK CPU imager (left panel) and WSCLEAN (right panel) from simulated visibilities generated by CASA starting from MWA antenna positions and a model image of Hydra A radio galaxy.

Sky Survey (VLSS, Lane et al., 2005) survey. These simulated visibilities were used to produce images using the CPU imager, WSCLEAN, and CASA (all dirty images, in natural weighting). The images produced by CASA and WSCLEAN were identical. Hence, given this and given that WSCLEAN is commonly used for MWA data, a comparison of the BLINK image and the WSCLEAN image is shown in Figure 3.3. It can be seen that the two images were nearly identical, which further validated that the CPU imager indeed produced the correct images.

The verification of the CPU version of the imager using both real and simulated data provided confidence that this test imager generated correct images. Hence, the images generated by the CPU imager were used as reference images, and this CPU imager was used as a starting point for the GPU imager, further explained in Section 3.2. This GPU version of the imager was developed and tested using the same test cases with reference input datasets and output images.

3.2 GPU imager

3.2.1 Single time-step, single-channel GPU Imager

The first version of the GPU imager was implemented based on an all-sky single time-step, single-channel test CPU imager described earlier in Section 3.1.1. In this imager, the operation of 2D FFT was implemented using the `cuFFT`[5] library instead of the `fftw` library, and the gridding procedure was implemented as a CUDA 'gridding' kernel, described below. Specific to the single time-step, multi-channel GPU imager, the `cufftPlan2d` function was implemented, as shown in Listing 3.1.

```
cufftHandle plan=0;
cufftPlan2d(&plan, (u,v)_grid_counter_xSize, (u,v)
    _grid_counter_ySize, CUFFT_C2C);
cufftExecC2C(plan, m_in_buffer_gpu, m_out_buffer_gpu,
    CUFFT_FORWARD);
cufftDestroy(plan);
```

Listing 3.1: Implementation of cuFFT.

The GPU gridding kernel was written to map the visibility data as a function of (u,v) coordinates on the (u,v) plane to (x,y) coordinates in the (u,v) grid and parallelise this process over all the (u,v) points (i.e. visibilities). The kernel was launched with 1024 threads (NTHREADS) per block and an adequate number of blocks (i.e. nBlocks = (xySize + NTHREADS -1)/NTHREADS, where 'xySize' refers to the size of the full correlation matrix) to process all the input visibilities. Different CUDA threads processed different visibility samples, and each thread was tasked with calculating the (x,y) index corresponding to one (u,v) point and adding complex visibilities to (u,v) grid cells at these (x,y) indices (see Section 2.2.2 for discussion of weighting).

[5] https://docs.nvidia.com/cuda/cufft/

The real and imaginary visibilities assigned to the same cell in the (u,v) grid were added using an atomic function[6]. This function is guaranteed to be executed without interrupting other threads, ensuring data consistency when multiple threads access and modify shared memory locations concurrently. Atomic functions are commonly used in parallel programming, especially in GPU programming using CUDA. They are crucial when multiple threads in a GPU kernel (in this case, the gridding kernel) may simultaneously try to update a shared variable, such as the same cell in the (u,v) grid. This can lead to data race conditions, where the value of the shared variable can become unpredictable. To prevent this, atomic operations ensure that only one thread updates the variable at a time, even when multiple threads try to update it simultaneously. In particular, atomicAdd[7] is a specific atomic operation that performs an atomic addition on a shared variable and ensures that the addition is performed in a mutually exclusive manner.

The next step was to validate the performance of this single time-step, single-channel GPU imager and compare its outputs to the original CPU version. This validation was done using real EDA2 data and is explained next in Section 3.2.2.

3.2.2 Validation of the single time-step, single-channel GPU Imager

To validate the functionality of the GPU imager, a comparison of its two functions, gridding and Fourier transform, was done. Several metrics were used to validate gridding. Firstly, the number of visibilities added to each cell of the (u,v) grid ('uv counters') was compared against the original CPU version. Similarly, real and imaginary parts of gridded visibilities were also compared. Likewise, to validate the Fourier transform function, a good measure was to compare the final resulting dirty images generated by the CPU and GPU imagers. The input source

[6] https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#atomic-functions

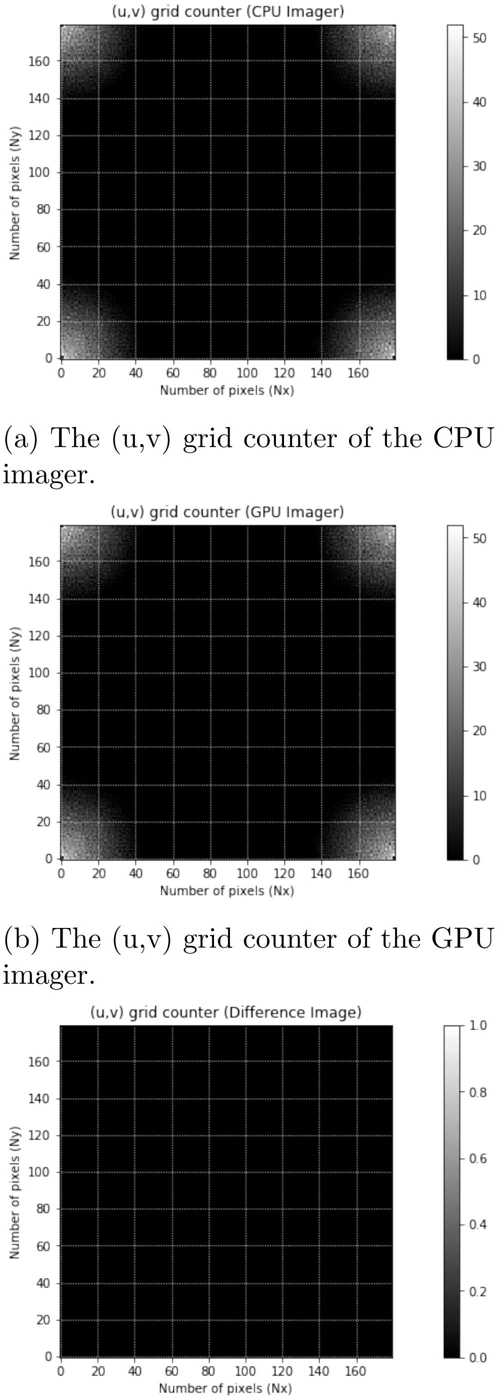
[7] https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#atomicadd

data for testing and validating the GPU imager was from the EDA2 telescope. The details and findings of these comparisons are described below:

1. *Comparing the (u,v) grid counters:* The (u,v) grid counter serves as a 'count' for how many visibilities have been added to a specific cell within the (u,v) grid. The comparison of these (u,v) grid counters, generated by the CPU and GPU imagers, respectively, is represented in Figures 3.4a and 3.4b. Furthermore, the difference image is shown in Figure 3.4c, demonstrating that the GPU imager's (u,v) grid counter perfectly matches the CPU imager's (u,v) grid counter. This signifies that an equal number of visibilities have been placed into each cell within the (u,v) grid.

2. *Comparing the correctness of the aggregated values within each grid:* The next point of comparison pertains to the actual (u,v) grids, where the visibilities corresponding to specific cells are summed. This comparison looks at the correctness of the aggregated values within each grid. This comparison compares the values in cells of the real and imaginary (u,v) grids. The comparison of these (u,v) grids (real) is visually depicted in Figure 3.5. Although the (u,v) grids appear remarkably similar at first glance, small differences of the order of approximately ≤ 0.25 (i.e. ≤ 0.0001)%[8] in the gridded visibilities were identified upon closer examination. These disparities arise as a natural consequence of utilising GPUs, resulting in a random order of additions in every execution and the inherent limitations of floating-point arithmetic in numerical computations. These differences are small and are still being examined. Thus, these two checks and comparisons confirmed the proper functionality of the GPU gridding kernel.

[8] 0.25: Maximum actual difference, 0.0001%: Maximum fractional difference.

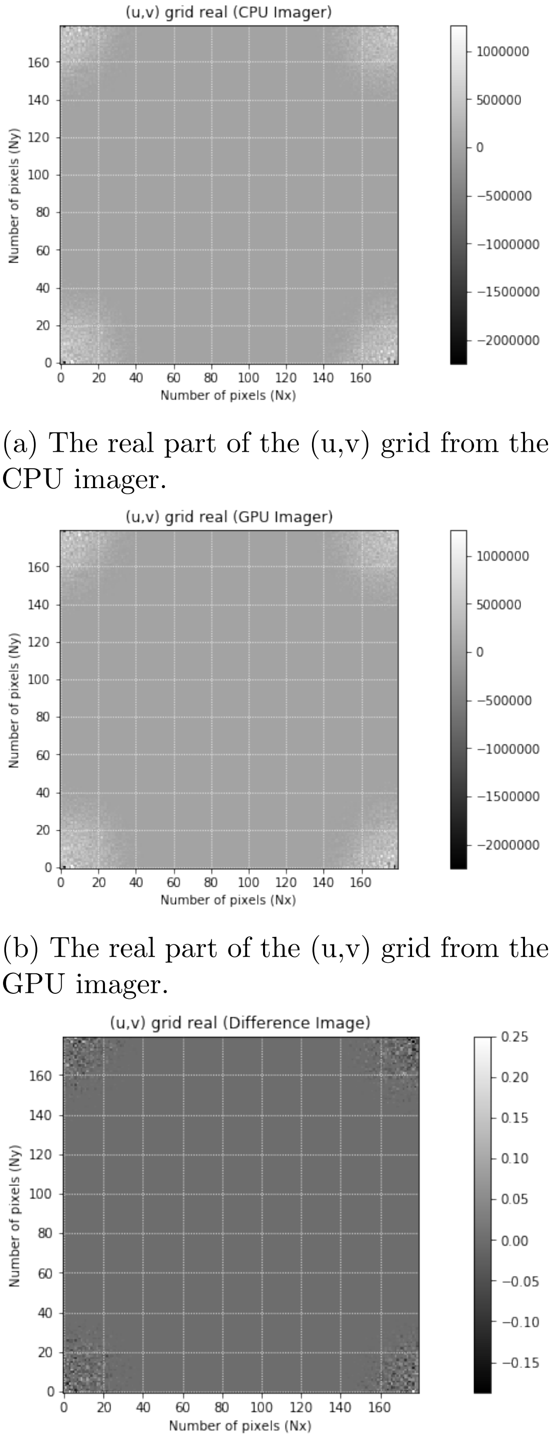


(a) The (u,v) grid counter of the CPU imager.

(b) The (u,v) grid counter of the GPU imager.

(c) The difference image of the CPU and GPU (u,v) grid counters.

Figure 3.4: Comparison of (u,v) cell counters for CPU imager (top plot a), GPU imager (middle plot b), and the difference of the two (bottom plot c). It is seen that the GPU imager's (u,v) grid counter perfectly matches the CPU imager's (u,v) grid counter, indicating that an equal number of visibilities have been placed into each comparison cell within the (u,v) grids.



(a) The real part of the (u,v) grid from the CPU imager.

(b) The real part of the (u,v) grid from the GPU imager.

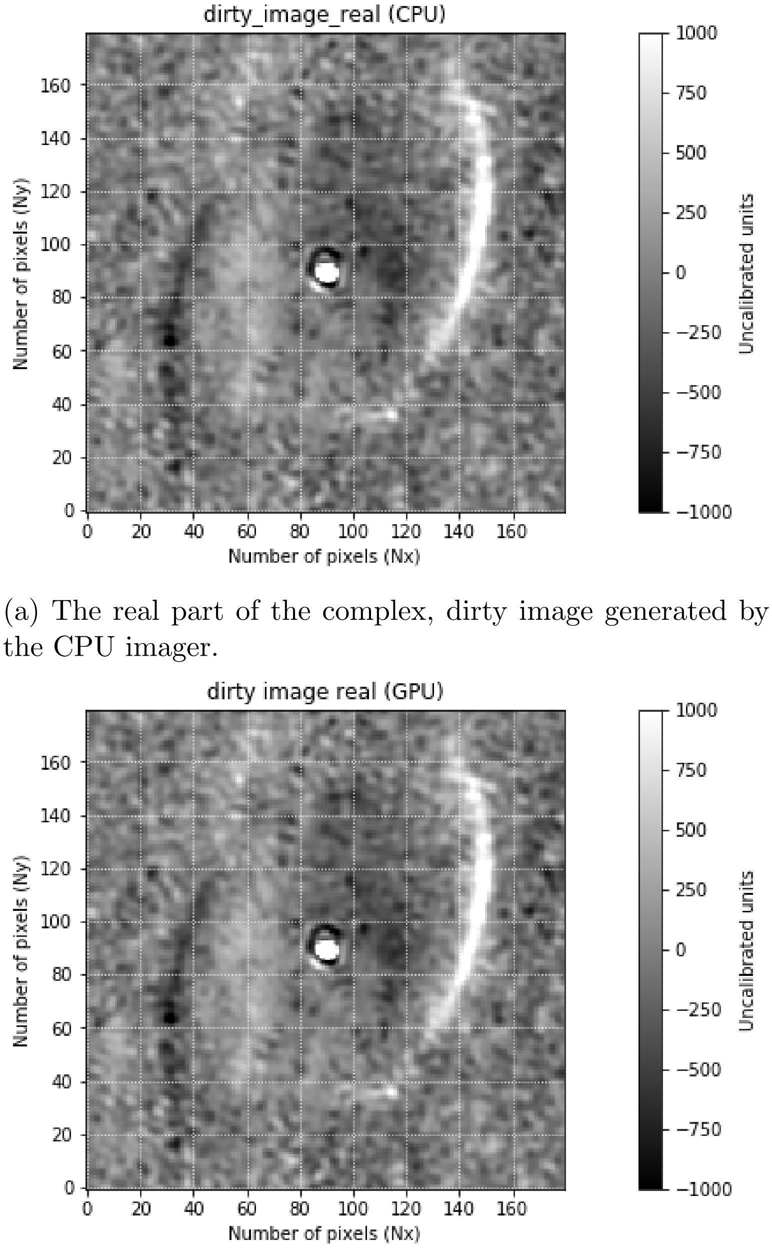
(c) The difference image of the real part of the (u,v) grid from the CPU and GPU imagers.

Figure 3.5: Comparison of the real part of the (u,v) grid for CPU imager (top plot a), GPU imager (middle plot b), and the difference of the two (bottom plot c). It is seen that the gridded visibilities of the CPU imager mostly match with the gridded visibilities of the GPU imager. Small differences of the order of approximately ≤ 0.25 ($\leq 0.0001\%$) were found in the gridded visibilities (for explanation, see Section 3.2.2).

3. *Comparing the dirty images (i.e. the real part of the complex image):* The final step involved comparing the generated dirty images (i.e. the real part of the complex image) obtained from both the CPU and GPU imager versions to validate the accuracy of the GPU-generated image. This comparison of the real part of the complex, dirty images is presented in Figure 3.6 below. The resulting flux density values in the GPU-generated image closely aligned with those in the CPU-generated image. Minor discrepancies, typically 0.001 (i.e. $\leq 0.38\%$)[9], were observed in the dirty images' final real and imaginary components (see Figure 3.7). No units have been mentioned, as these images were not flux-calibrated.

These differences between the CPU and GPU images are still being examined. The current understanding is that they result from the small fractional differences observed in the gridded visibilities, a natural outcome of utilising GPUs for parallel addition. Using GPU threads for addition results in combining data randomly, leading to different round-off errors with each execution. This is a small limitation of the current test version of the GPU imager, as the results are not strictly reproducible. This also affects and limits the results' predictability, not precision. Consequently, atomicAdd may not be ideal for implementing gridding. Alternative methods like parallel reduction to achieve more predictable outcomes are being considered. This will eliminate these differences between the CPU and GPU versions. It will also result in code optimisation by removing atomicAdd, a rather costly operation because it is implemented as an active wait loop. However, replacing atomicAdd is not currently a high priority. Its impact on performance is minimal compared to other parts of the code (FFT, memory copies, etc), and the resulting errors in the final dirty images are negligible for the FRB/transient search applications. Therefore, these modifications have been deferred for future implementation.

[9] 0.001: Maximum actual difference, 0.38%: Maximum fractional difference.



(a) The real part of the complex, dirty image generated by the CPU imager.

(b) The real part of the complex, dirty image generated by the GPU imager.

Figure 3.6: Comparison of the dirty image (real) for both the CPU and GPU versions. Images of size: 180 × 180 were generated.

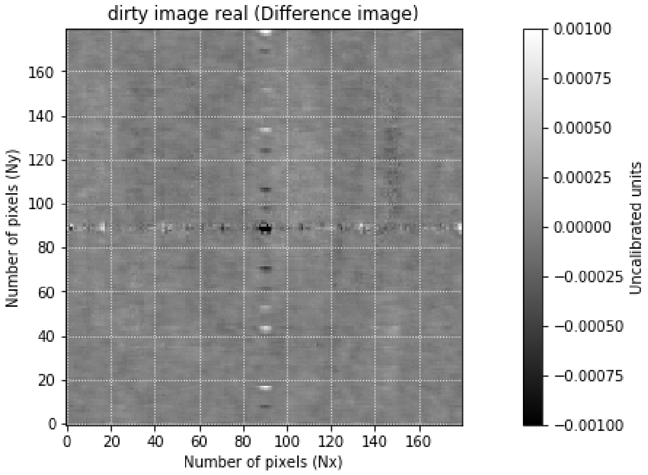


Figure 3.7: The difference between the real part of the CPU and GPU dirty images. The maximum difference observed in the real part CPU and GPU dirty images is 0.001 ($\leq$ 0.38%). For an explanation, see Section 3.2.2.

3.2.3 Single time-step multi-channel GPU Imager

In the next step, the GPU imager for a single time step with numerous frequency channels ($n_{channels}$) was implemented. In future imager versions, the number of channels will be similar to the number in the MWA. For example, N will be equal to 768 × 40 kHz MWA channels for 30.72 MHz bandwidth. Initially, 768 channels may be too much to process, and we may need to average down a few channels. However, the number of channels is expected to be 100 or so. In the end, the number of channels will be optimised for the sensitivity of FRB searches to minimise dispersion smearing.

An important thing to note here is that the high-level goal of the project was to make the imager applicable to any GPU architecture (both NVIDIA and AMD GPUs). I developed and tested the initial CUDA version of the GPU imager. Hence, in this chapter, only the CUDA implementation of the GPU code is discussed. This was later converted to the Heterogeneous-Compute Interface for Portability (HIP)[10] version of the GPU imager, by my primary supervisor, Dr Marcin Sokolowski. This task of making the GPU imager general was completed

[10] https://rocm.docs.amd.com/projects/HIP/en/latest/

by Dr Marcin Sokolowski and a PhD student after the submission of this thesis. Currently, a single version of the code is applicable for both CUDA and HIP (Sokolowski et al., 2024a). After the HIP version was developed, I benchmarked two different versions (i.e. CUDA and HIP versions) on Topaz and Setonix and generated the profiled reports. The motivation for the Topaz vs Setonix comparison was to see how well the GPU imager performed on Setonix and to verify if more work on the software side using the HIP programming environment was required to make the imager more efficient on Setonix (as eventually, all data processing will be performed on Setonix).

In this test version of the GPU imager, $n_{channels}$ was a variable input, allowing us to test the implementation with different values. Specifically, the imager was tested with $n_{channels} = 50$, 500 and 5000 channels. Gridding was done in a parallel manner, either using 2D-grid of CUDA layers or CUDA streams[11], independently, which has been discussed below.

CUDA streams provide a way to manage concurrent execution of multiple tasks on a GPU, enhancing parallelism and improving overall GPU utilisation. The two main advantages of using CUDA streams are:

1. **Concurrency:** CUDA streams enable concurrent execution of GPU kernels on a GPU. Instead of running one task at a time and waiting for it to finish, multiple tasks are initiated and executed simultaneously within separate streams, resulting in significant performance gains. The number of tasks that can run in parallel depends on the number of multiprocessors available. The number of tasks that can run parallelly within a single stream depends on the number of available CUDA cores and the specific architecture of the GPU. Thus, a GPU with multiple multiprocessors and more cores can handle more parallel tasks.

2. **Error isolation:** Streams can help in isolating errors. If an error occurs in one stream, it typically does not affect other streams. This isolation

[11] https://developer.download.nvidia.com/CUDA/training/StreamsAndConcurrencyWebinar.pdf

makes debugging and error handling more manageable.

In summary, CUDA streams are an important component of GPU programming[12], enabling concurrent and efficient execution of tasks on GPUs. They provide a flexible mechanism for harnessing GPUs' massive parallel processing power while effectively managing concurrency, resource allocation, and synchronisation.

In the version of parallel gridding using CUDA streams, a small section of the code creates a preferred number of streams (i.e. nStreams[N]), and the gridding kernel is launched in a for-loop over the frequency channels. Here, the gridding kernel is launched with 1024 threads (NTHREADS), a sufficient number of blocks (NBLOCKS), and an array of 15 CUDA streams (i.e. nStreams[15]) to accommodate all the visibilities that need to be gridded for a given time step. For each frequency channel ($n_{channels}$) within a time-step, the gridding kernel is independently launched, resulting in one launch per channel. These launches occur concurrently, leveraging the parallel execution capability of GPUs. To ensure organised execution, each kernel launch for a different frequency channel is assigned to a distinct CUDA stream from the array of streams. This allocation is achieved through a modulo operation using the formula nStreams[b % 15], where 'b' ranges from 0 to ($n_{channels}$ - 1) corresponding to the frequency channel.

This creation of CUDA streams, though it happens only once during the entire execution of the program, has some overheads. Hence, as an alternative, another implementation launched the gridding kernel with a 2D grid of blocks. From here on, this implementation will be referred to as CUDA layers. In the version of parallel gridding using CUDA layers, the kernel is launched only once for the single time step through a 2D grid of blocks, with 1024 threads in each block. This is illustrated in Figure 3.8. Here, the dimensions of this 2D CUDA grid are: (NBLOCKS × $n_{channels}$). Thus, the first dimension of the CUDA grid (i.e. NBLOCKS) is configured to process each image, and the second dimension of the

[12] The AMD equivalent of CUDA streams are HIP streams.

CUDA grid is configured to process all the frequency channels. This eliminates the need to launch multiple gridding kernels using a for-loop.

Thus, the number of CUDA blocks that need to be launched to accommodate visibilities from a single frequency channel of a single time-step (i.e. NBLOCKS) is given by:

$$NBLOCKS = \frac{n_{\text{vis}}}{1024} = \frac{65280}{1024} = 64, \tag{3.1}$$

where n_{vis} refers to the number of visibilities for the EDA2 when only cross-correlation products are considered, corresponding to a single frequency channel of a given single time step.

Finally, the total number of CUDA blocks (n_{all_blocks}) that are needed to be launched for all frequency channels for a given time-step is given by NBLOCKS $\times n_{channels}$. Consequently, the total number of CUDA threads ($n_{all\_threads}$) that will be launched is given by $n_{all\_blocks} \times 1024$, assuming there are 1024 threads in a single CUDA block. Table 3.1 summarises the total number of n_{all_blocks} and $n_{all\_threads}$ launched, corresponding to a single time-step to accommodate gridding of visibilities resulting from different numbers of channels.

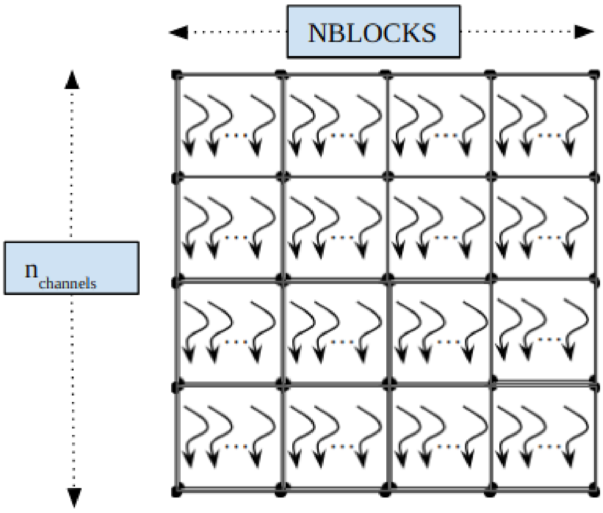


Figure 3.8: The figure shows a grid of blocks of threads. In the implementation of parallel gridding using CUDA layers, the gridding kernel is launched only once with a 2D grid of CUDA blocks. The dimensions of this 2D CUDA grid are: (NBLOCKS × $n_{channels}$), and the number of CUDA threads in each block is 1024. Here, NBLOCKS is the number of blocks launched to accommodate visibilities from a single frequency channel of a single time-step, given by $n_{vis}/1024 = 64$. And, $n_{channels}$ is a variable input and refers to the number of frequency channels processed in a single time-step of the GPU imager.

Table 3.1: The total number of CUDA blocks and CUDA threads launched to accommodate parallel gridding of visibilities resulting from different numbers of frequency channels from a single time-step.

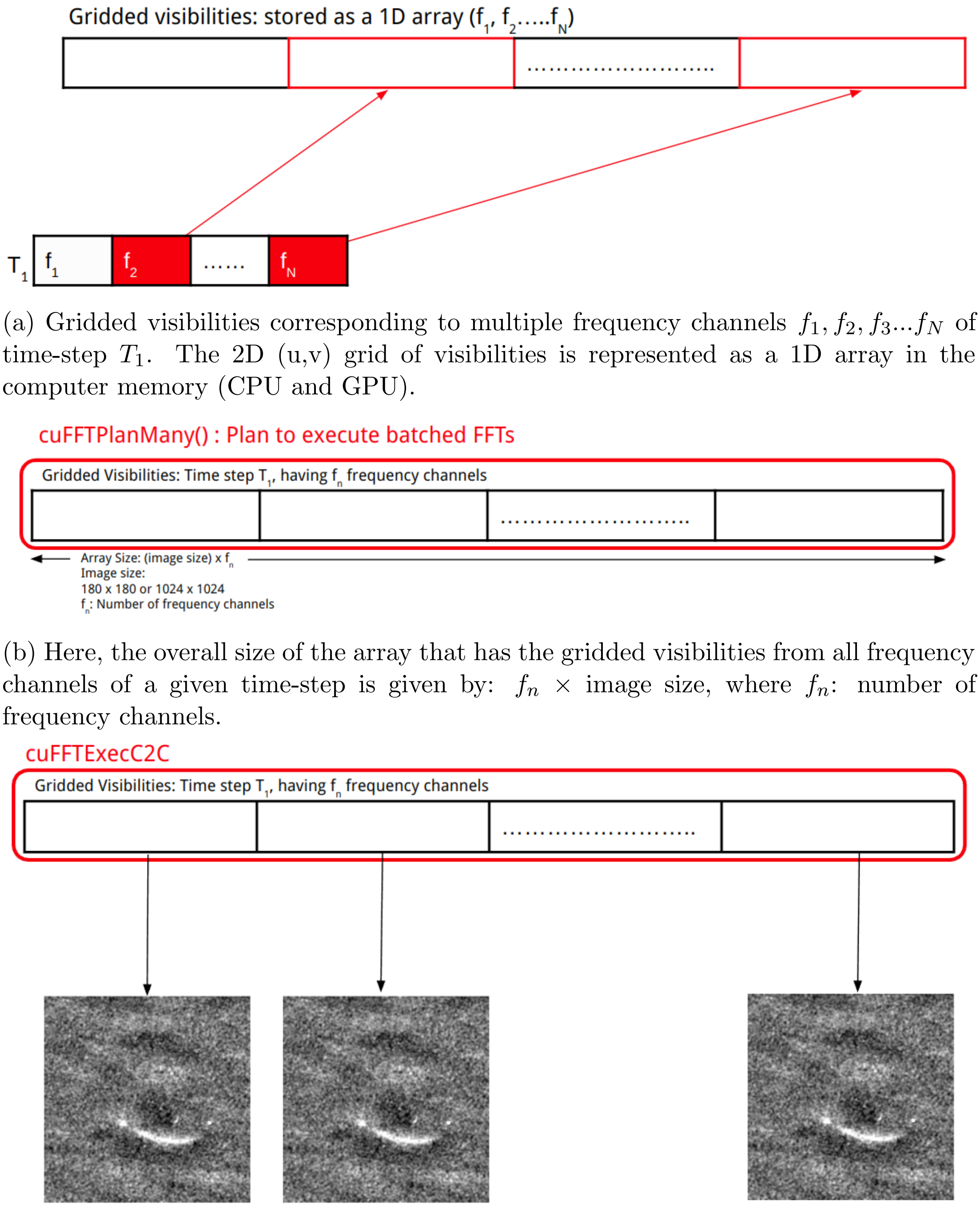
Number of channels ($n_{channels}$)	**Number of visibilities** (n_{vis})	**Total Blocks:**[1] **NBLOCKS** × $n_{channels}$ (n_{all_blocks})	**Total Threads:**[2] n_{all_blocks} × **1024** ($n_{all\_threads}$)
1	65280	(64 × 1)	65536
2	130560	(64 × 2)	131072
10	652800	(64 × 10)	655369
32	2088960	(64 × 32)	2097152

[1] Dimensions of the 2D-grid of CUDA blocks. Here, the number of channels determines the number of layers of the CUDA blocks.

[2] First, the number of blocks is decided, and since the number of threads in each block is the same, more threads than needed are launched. Hence, some threads launched in the last block are not used, and there is a guardian (if) condition in the kernel to exit the gridding function if the thread corresponds to a non-existing visibility.

After completing the gridding process, the Fourier transforms were configured using `cufftPlanMany`, which sets up batched FFTs on numerous gridded visibilities to generate batched images in a single step. Here, the multiple FFTs are executed concurrently, leveraging the parallel processing capabilities of GPUs.

In summary, in this sub-section, the implementation of the two main processes, gridding and Fourier transform, in the single time-step, multi-channel implementation of the GPU imager was explained. The two implementations of parallel gridding, one using CUDA streams and another using a 2D grid of CUDA layers, were also discussed. The implementation of the FFTs using `cuFFTPlanMany` to generate multiple batched images in a single step was also explained. An illustration of parallel gridding and `cuFFTPlanMany` creation and execution is shown in Figure 3.9. Next, in Section 3.3, an overview of the performance of the two versions of the GPU imager and its benchmarks obtained on the Topaz and Setonix supercomputers at Pawsey will be provided.


Gridded visibilities: stored as a 1D array (f_1, f_2.....f_N)
....
T_1
f_1
f_2
......
f_N
cuFFTPlanMany() : Plan to execute batched FFTs
Gridded Visibilities: Time step T_1, having f_n frequency channels
....
Array Size: (image size) x f_n
Image size:
180 x 180 or 1024 x 1024
f_n: Number of frequency channels
cuFFTExecC2C
Gridded Visibilities: Time step T_1, having f_n frequency channels
....

(a) Gridded visibilities corresponding to multiple frequency channels $f_1, f_2, f_3...f_N$ of time-step T_1. The 2D (u,v) grid of visibilities is represented as a 1D array in the computer memory (CPU and GPU).

(b) Here, the overall size of the array that has the gridded visibilities from all frequency channels of a given time-step is given by: $f_n \times$ image size, where f_n: number of frequency channels.

(c) Batched FFTs are performed using `cuFFTExecC2C`, to generate batched images.

Figure 3.9: The Fourier transforms are configured using `cufftPlanMany`, which sets up batched FFTs on numerous gridded visibilities (panel a) to generate batched images in a single step. A `cuFFTPlanMany` is created (panel b) and executed (panel c) on these gridded visibilities to generate output images as illustrated.

3.3 GPU imager performance

This section gives an overview of the performance of the GPU imager and its benchmarks obtained on the Topaz and Setonix supercomputers at Pawsey. In Section 3.3.1, the performance comparison of the CPU and GPU versions of the test imager for a single channel and a single time step is provided. Section 3.3.2 discusses the performance of the single time-step, multi-channel version of the imager. Section 3.3.3 compares the two versions of the single time-step multi-channel GPU imager, one that uses CUDA streams and another that uses a 2D grid of CUDA blocks. Section 3.3.4 compares the performance of gridding and FFT for the single time-step multi-channel version of the GPU imager implemented on Topaz and Setonix. In all comparisons, the 180×180 and 1024×1024 images exemplify the images generated from EDA2 and MWA, respectively.

3.3.1 Performance of the single-channel, single time-step GPU imager

Table 3.2 shows the performance of the CPU and GPU versions of the imager on a single image for one time-step and frequency channel. There is a noticeable improvement, up to $\sim$3 times in the total execution time in the GPU imager for higher image sizes (4096 × 4096). However, as expected, there is no improvement for smaller image sizes (180 × 180 and 1024 × 1024). Thus, the speed-up is only seen on image sizes outside the EDA2/MWA image sizes. It also shows that the gains in speed-up are the highest for the largest image sizes due to the dominant contribution from 2D FFT.

An important thing to note is that, in this case, only one image is being dealt with. For a single image, the time is dominated by memory copies, and therefore, the gain from moving to GPUs is quite small or non-existent. Hence, it is important to copy from CPU to GPU memory in bulk (i.e. multiple input visibilities at once), process them in bulk in the GPU memory, and copy them back from GPU memory to CPU memory in bulk (i.e. multiple output images at once).

Thus, at first look, though it does look like the GPU imager is slower than the CPU imager, this is due to the memory operations (copy of data to and from GPUs) dominating the execution time. However, this changes in the practical setting as there will not be just one time step and one channel. There will be many time steps with multiple frequency channels; hence, these memory copy operations will not dominate the benchmarks. This single-channel, single-time-step GPU imager was just a starting point in the overall development of the GPU imager, and hence, a benchmark of its run-time was done for validation purposes. The single time-step, multi-channel GPU imager was developed from this version, where parallel gridding and batched Fourier transform were performed. Next, in Section 3.3.2, the performance of this single-time-step, multi-channel GPU imager is discussed.

Table 3.2: A comparison of the total time of gridding and imaging for both CPU and GPU implementations of the imager for a single time-step and frequency channel, tested on EDA2 data.

Image size	**180 × 180 (ms)**	**1024 × 1024 (ms)**	**4096 × 4096 (ms)**
CPU Total Execution Time	0.02	0.36	43.48
GPU Total Execution Time	0.42	0.58	13.2

3.3.2 Performance of the single time-step multi-channel GPU imager

This section discusses the performance of the single time-step, multi-channel GPU imager on the Topaz (using NVIDIA V100 GPUs) and Setonix (using AMD Instinct MI250X GPUs) supercomputers (see Table 3.3) at the Pawsey Supercomputing Centre, in Perth, Western Australia.

System	**GPU Model**	**CUDA/HIP Version**	**Memory Bandwidth (GB/s)**
Topaz	NVIDIA V100 GPUs	CUDA 10.1	900
Setonix	AMD MI250X GPUs	ROCM 5.4.3	3200

Table 3.3: The specifications of the GPU architectures used for testing and benchmarking the imager presented in this work.

The creation times for FFT PlanMany (implemented using `cuFFT` and `hipFFT`[13] libraries on Topaz and Setonix, respectively) remain constant irrespective of image size and the number of images generated simultaneously. They are around 360 ms and 9 ms on Topaz and Setonix, respectively. Thus, PlanMany creation is nearly 40 times faster on Setonix, i.e. using the AMD HIP framework. It's crucial to note that while this creation time is not negligible, it occurs only once during execution, specifically before processing all time steps. The run-times of PlanMany 'execution' on Topaz and Setonix, as shown in Figures 3.10 and 3.11 respectively, are more important and occur for every time step.

From Figure 3.10, as expected, on Topaz, the run-time of FFT PlanMany executions increases with an increasing number of images generated per execution. This means that as more images are processed in a single batch, the run-time for FFT PlanMany execution also increases.

Similarly, from Figure 3.11, on Setonix, the FFT PlanMany execution time slightly increases with the number of images generated per execution. In summary, there seems to be a constant execution time of about $\approx$ 210 ms and then an additional $\approx$ 1.3 ms per image for 180 × 180 images and $\approx$ 37.6 ms per image for 1024 × 1024 images. This slight increase in PlanMany execution times is also shown in Table 3.4.

Furthermore, this similar execution time for the AMD cards (on Setonix), as shown in Figure 3.11, is of the same order as the extra creation time for NVIDIA cards (on Topaz). This consistency in execution times suggests that Setonix's GPU architecture handles FFT PlanMany operations differently from

[13] https://rocm.docs.amd.com/projects/hipFFT/en/latest/

Table 3.4: FFT PlanMany average execution times on Setonix for two image sizes as a function of the number of images generated in one execution.

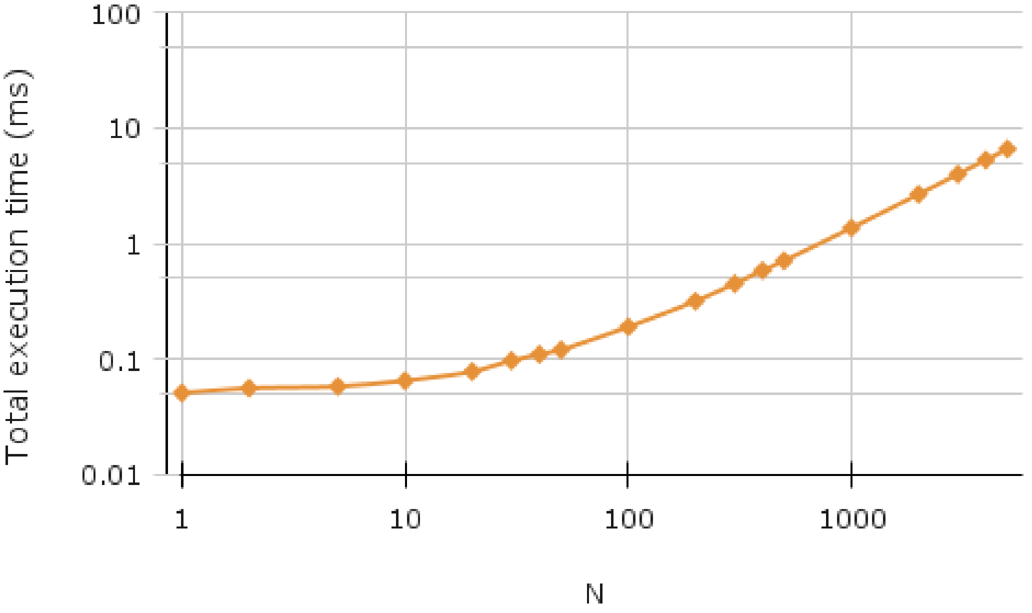
Image Size	**Number of images in one execution (N)**	**PlanMany Execution Time (Average)**
180 x 180	1 to 500	208 ms
	1000 to 9000	216 ms
	10000 to 30000	237 ms
	30000 to 50000	262 ms
1024 x 1024	1 to 500	217 ms
	750 to 2000	260 ms
	2000 to 2500	300 ms

Topaz. This could be because some calculations are probably moved from PlanMany creation to PlanMany execution in the HIP programming framework used to write GPU code on AMD GPUs.

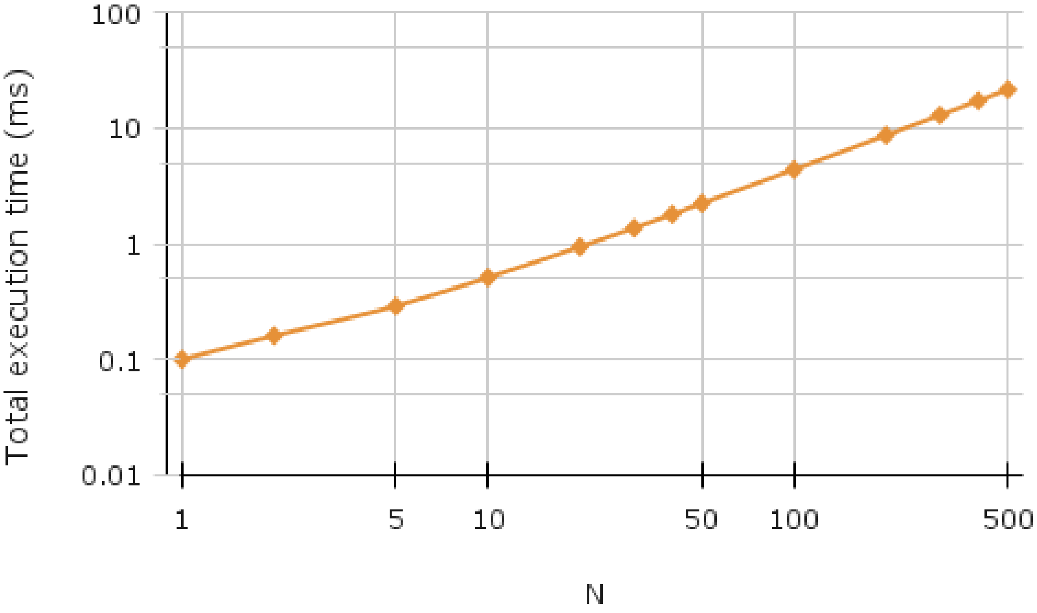
Table 3.5 summarises the total run times of FFT PlanMany (which includes plan creation and execution) to generate 500 images in a single step. It can be seen that performing the operation of FFT per image utilising GPUs is extremely quick, as generating a single image takes less than 1 ms on both Topaz and Setonix.

Table 3.5: The total time taken by FFT PlanMany (i.e. including plan creation and execution) to generate 500 images in a single batch on Topaz and Setonix. The run times are shown for generating two image sizes: 180 × 180 and 1024 × 1024. The benchmarks were performed on Topaz and Setonix supercomputers.

Supercomputer	**Size of the image**	**Total run-time (ms)**	**Execution time per image (ms)**
Topaz	180 × 180	360.491	0.72
	1024 × 1024	377.139	0.75
Setonix	180 × 180	230.726	0.46
	1024 × 1024	253.719	0.49

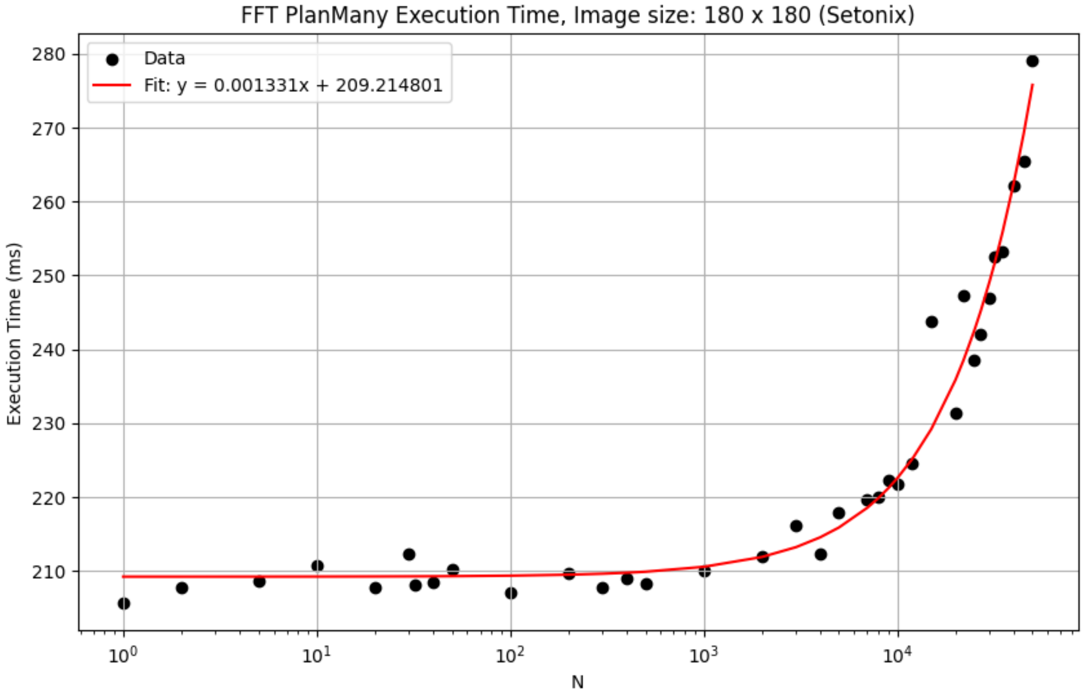


(a) Image size: 180 × 180.

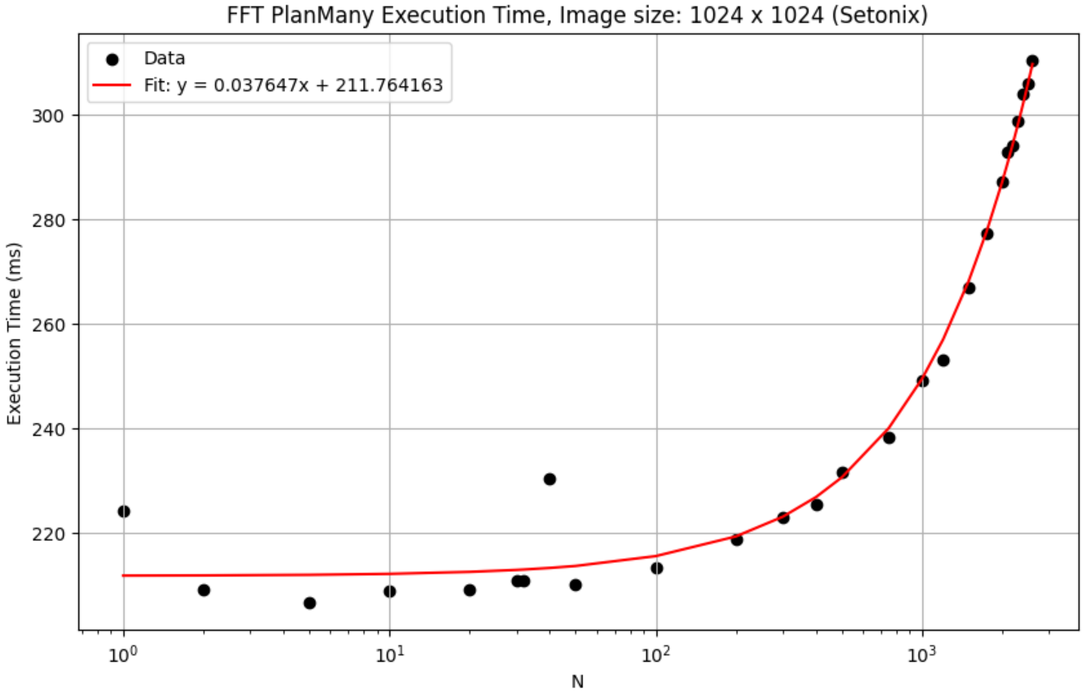


(b) Image size: 1024 × 1024.

Figure 3.10: The run-times in (ms) of FFT PlanMany (execution) on Topaz (orange curves) for image sizes: 180 × 180 (top panel) and 1024 × 1024 (bottom panel).



(a)



(b)

Figure 3.11: The run-times in (ms) of FFT PlanMany (execution) on Setonix for image sizes: 180 × 180 (top panel) and 1024 × 1024 (bottom panel). There seems to be a constant execution time of about ≈ 210 ms and then an additional ≈ 1.3 ms per image for 180 × 180 images and ≈ 37.6 ms per image for 1024 × 1024 images.

3.3.3 Parallel Gridding: Streams vs layers

In this section, the performance of the two versions of the GPU imager, one that uses GPU streams [14] and another that uses a 2D grid of layers on both Topaz and Setonix, is compared.

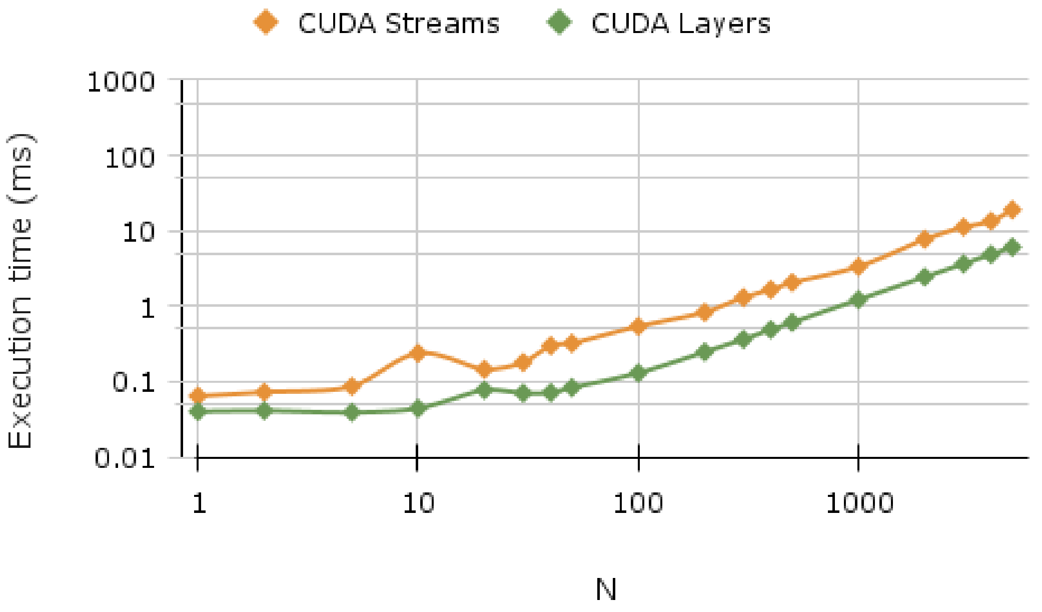
The run-times of the two versions of the GPU implementations, one using CUDA streams and another using 2D-grid of CUDA layers on Topaz, obtained using a high-resolution clock is shown in Figure 3.12. The following observations can be made from Figure 3.12 for the benchmarking results on Topaz:

- The gridding execution times remain nearly identical regardless of image size, as the number of visibilities being gridded is constant in each case. Therefore, these results are in line with expectations.

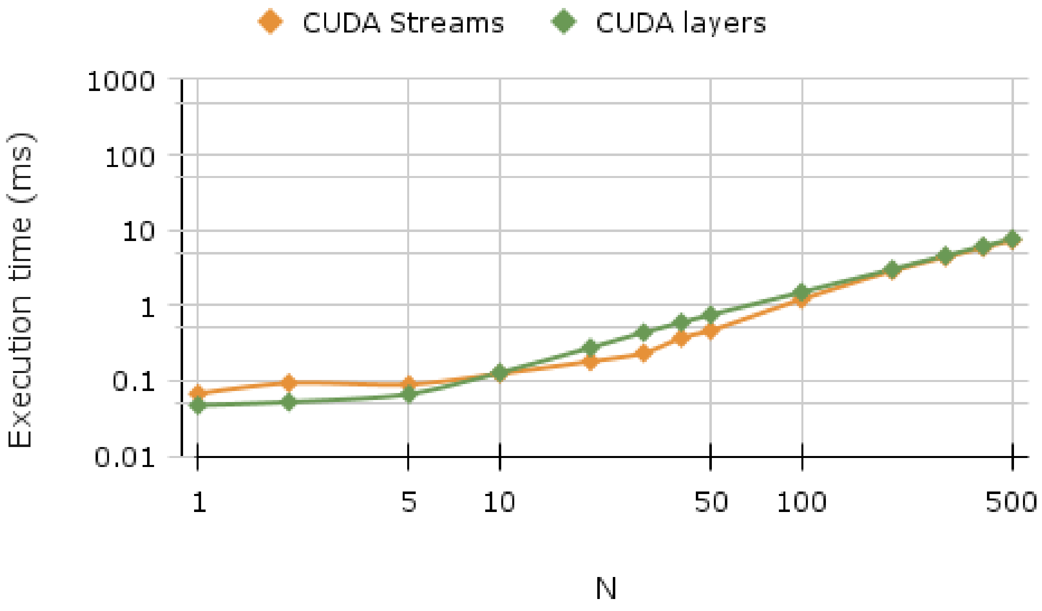
- For smaller image sizes (i.e. 180 × 180), launching a single kernel with a 2D grid of CUDA layers seems to have slightly lower run times (i.e. slightly improved performance) when compared to launching multiple kernels with CUDA streams. Launching a CUDA kernel has some overheads, including setting it up, launching it, and synchronising threads. Thus, when multiple kernels are launched with CUDA streams, this overhead accumulates for each launch. On the other hand, a single kernel launch with a 2D grid of CUDA layers incurs this overhead only once, which can be more efficient for smaller tasks (i.e. in this case, smaller image sizes).

- For higher image sizes (i.e. 1024 × 1024), the run times of the two versions are nearly the same and result in comparable performance. Launching multiple kernels with CUDA streams introduces some overhead related to stream management. For smaller image sizes, this overhead may become a significant fraction of the total execution time. In contrast, the relative overhead becomes smaller for larger image sizes, making the performance of CUDA streams more similar to the performance of the CUDA layers.

[14] where GPU refers to CUDA or HIP respectively for NVIDIA and AMD GPUs.



(a) Image size: 180 × 180

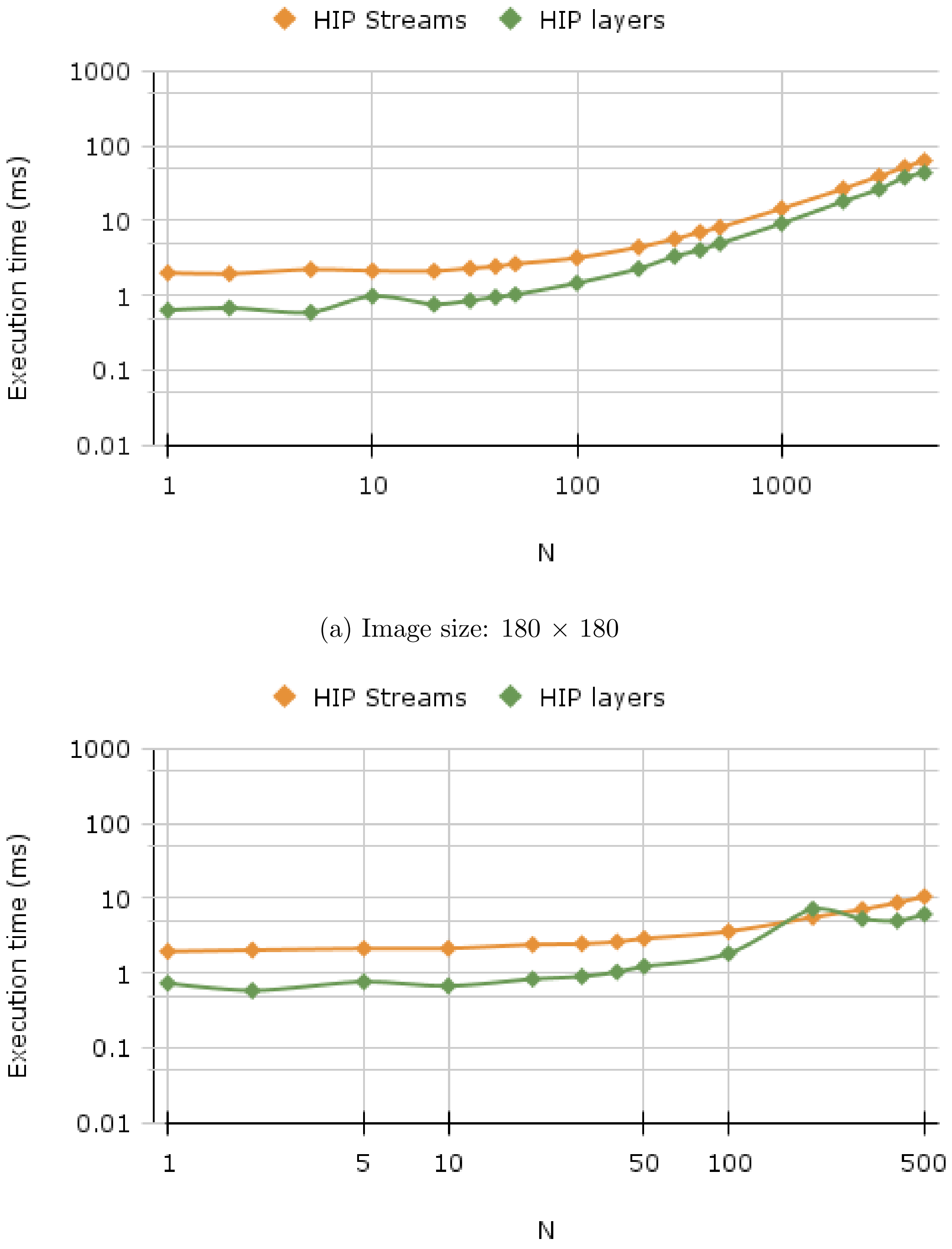


(b) Image size: 1024 × 1024

Figure 3.12: CUDA streams vs CUDA layers on Topaz: Total run-times for the gridding section of the code that generates N images of two different sizes (i.e. top panel: 180 × 180 and bottom panel: 1024 × 1024), executed on Topaz. The orange curves correspond to the code that uses CUDA Streams to launch multiple CUDA kernels, and the green curves refer to the code that uses a 2D grid of CUDA layers to launch a single CUDA kernel. The gridding execution times remain nearly identical regardless of image size, as the number of visibilities being gridded is constant in each case. Therefore, these results are in line with expectations.

Similarly, the run-times of the two versions of the GPU implementations, using HIP streams and HIP layers on Setonix (i.e. the HIP equivalents of CUDA streams and CUDA layers), obtained using the high-resolution clock are shown in Figure 3.13. From Figure 3.13, it is evident that for smaller (i.e. 180 × 180) and higher (i.e. 1024 × 1024) image sizes, the run times of HIP layers are slightly lower (i.e. indicating slightly improved performance) in comparison to using HIP streams.

This observed performance variation while using CUDA/HIP streams and layers on Topaz and Setonix could also be due to the difference in GPU architectures. A performance comparison of Topaz vs Setonix is further done in Section 3.3.4.



(a) Image size: 180 × 180

(b) Image size: 1024 × 1024

Figure 3.13: HIP streams vs HIP layers on Setonix: Total execution times for the gridding section of the code that generates N images of two different sizes (i.e. top panel: 180 × 180 and bottom panel: 1024 × 1024), executed on Setonix. The orange curves correspond to the code that uses HIP Streams to launch multiple HIP kernels, and the green curves refer to the code that uses a 2D grid of HIP layers to launch a single HIP kernel.

3.3.4 Parallel Gridding: Topaz vs Setonix

In this section, the performance of parallel gridding using streams[15] and layers[16] on Topaz vs Setonix are compared.

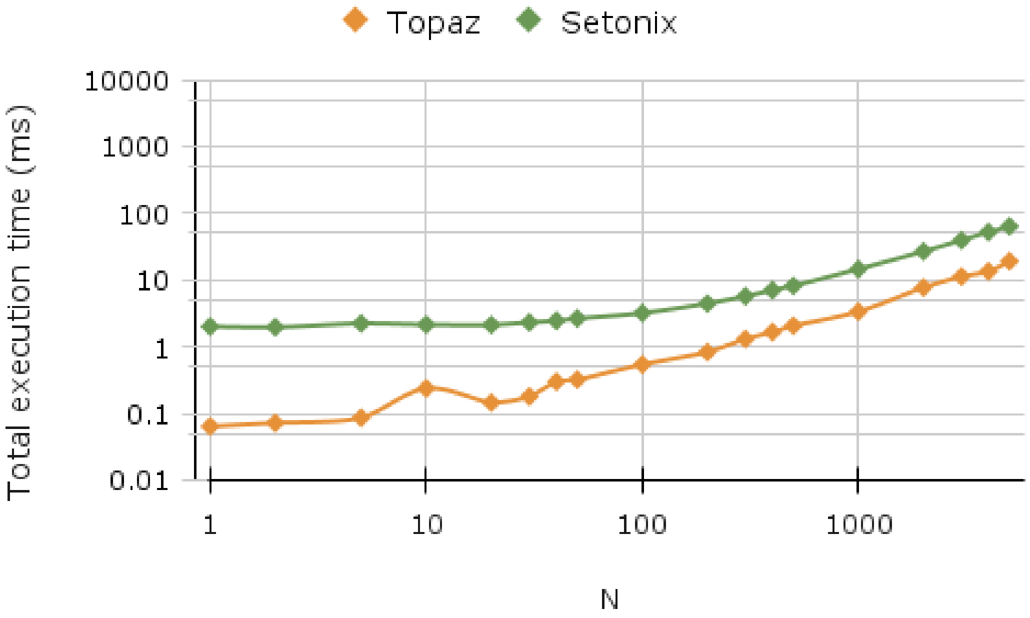
The comparison of the performance of parallel gridding using streams on Topaz vs Setonix is shown in Figure 3.14. From Figure 3.14, it is seen that for parallel gridding using streams, the gridding execution times are, on average, ten times faster on Topaz than Setonix for both 180 × 180 and 1024 × 1024 images.

A similar trend was observed for parallel gridding using layers on Topaz and Setonix. For parallel gridding using layers for generating 180 × 180 images, the gridding execution times are smaller on Topaz than on Setonix. They are, on average, ten times faster on Topaz than on Setonix. Likewise, for parallel gridding using layers for generating 1024 × 1024 images, Topaz is, on average, four to five times faster than Setonix for smaller values of N. However, for N $>$ 300, the execution times on Topaz and Setonix become comparable.

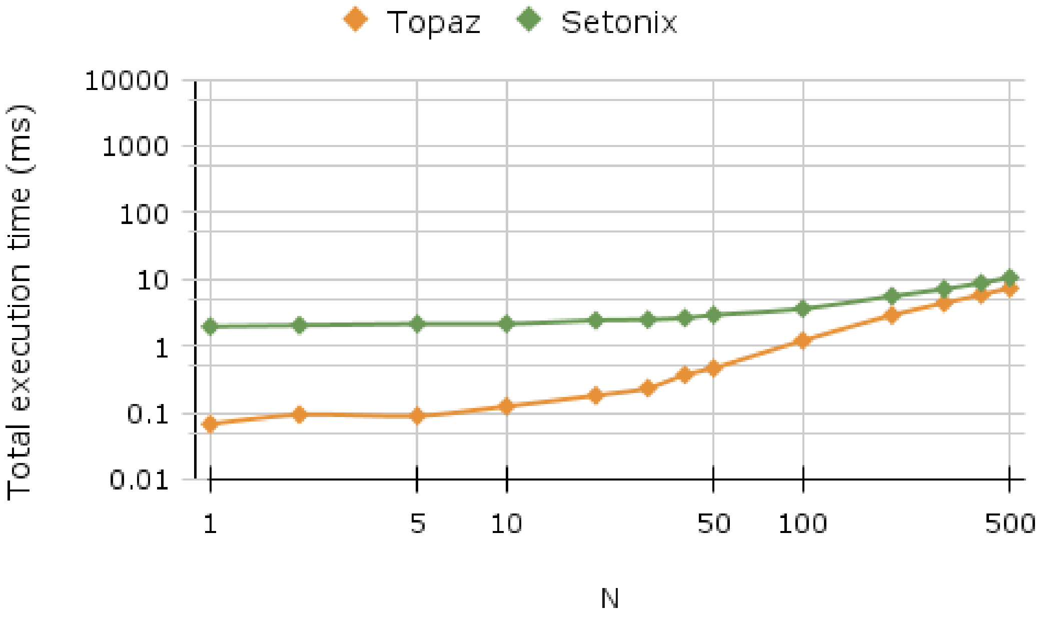
The most important thing to note here is that the GPU gridding performs extremely fast on both Topaz and Setonix. Tables 3.6 and 3.7 show the performance of the GPU imager on Topaz and Setonix that uses the streams and layers versions, respectively, of gridding. From these values, it is clear that it takes only a few microseconds to perform gridding for a single image of any image size while performing gridding on a batch of 500 or 5000 images.

[15] CUDA streams on Topaz and HIP streams on Setonix.

[16] CUDA layers on Topaz and HIP layers on Setonix



(a) Image size: 180 × 180



(b) Image size: 1024 × 1024

Figure 3.14: Comparison of execution times of parallel gridding, using streams (CUDA streams on Topaz, HIP streams on Setonix) to generate N images, for two different image sizes: 180 × 180 (top panel) and 1024 × 1024 (bottom panel), executed on Topaz and Setonix denoted by orange and green curves, respectively.

Table 3.6: Gridding execution time of the GPU Imager for image sizes 180 × 180 and 1024 × 1024 using CUDA streams on Topaz and HIP streams on Setonix, respectively.

Supercomputer	**Image Size**	**Number of images**	**Total processing time (ms)**	**Execution time per image (µs)**
Topaz	180 × 180	5000	19.03	3.8
	1024 × 1024	500	7.32	14.6
Setonix	180 × 180	5000	63.38	12.7
	1024 × 1024	500	10.56	21.1

Table 3.7: Gridding execution time of the GPU Imager for image sizes 180 × 180 and 1024 × 1024 using CUDA layers on Topaz and HIP layers Setonix, respectively.

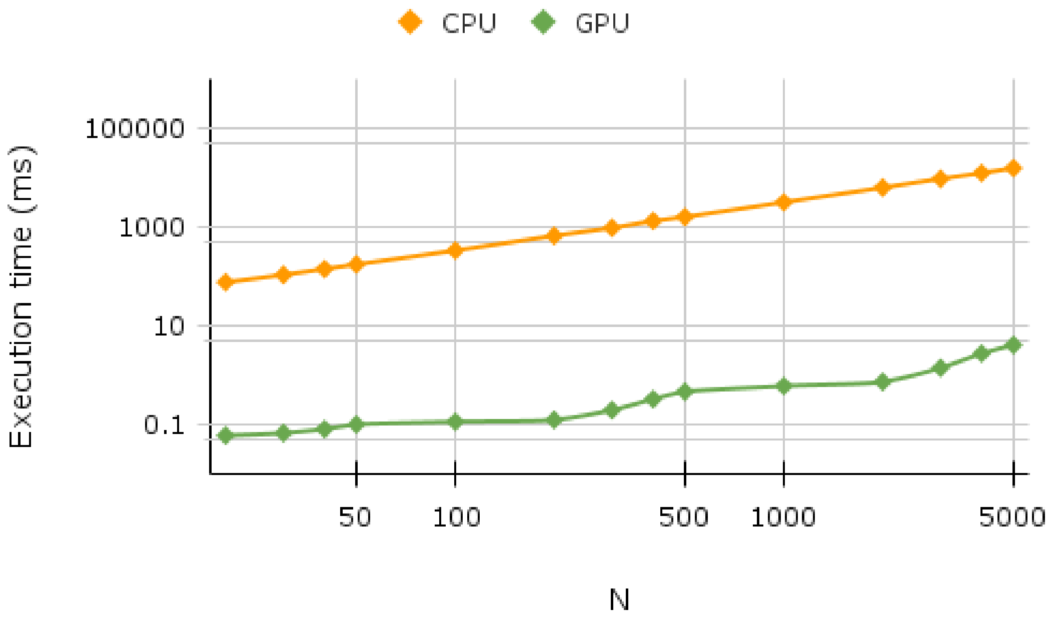
Supercomputer	**Image Size**	**Number of images**	**Total processing time (ms)**	**Execution time per image (µs)**
Topaz	180 × 180	5000	6.06	1.2
	1024 × 1024	500	7.61	15.2
Setonix	180 × 180	5000	43.61	8.7
	1024 × 1024	500	6.11	12.2

3.3.5 Performance Comparison with the baseline CPU imager

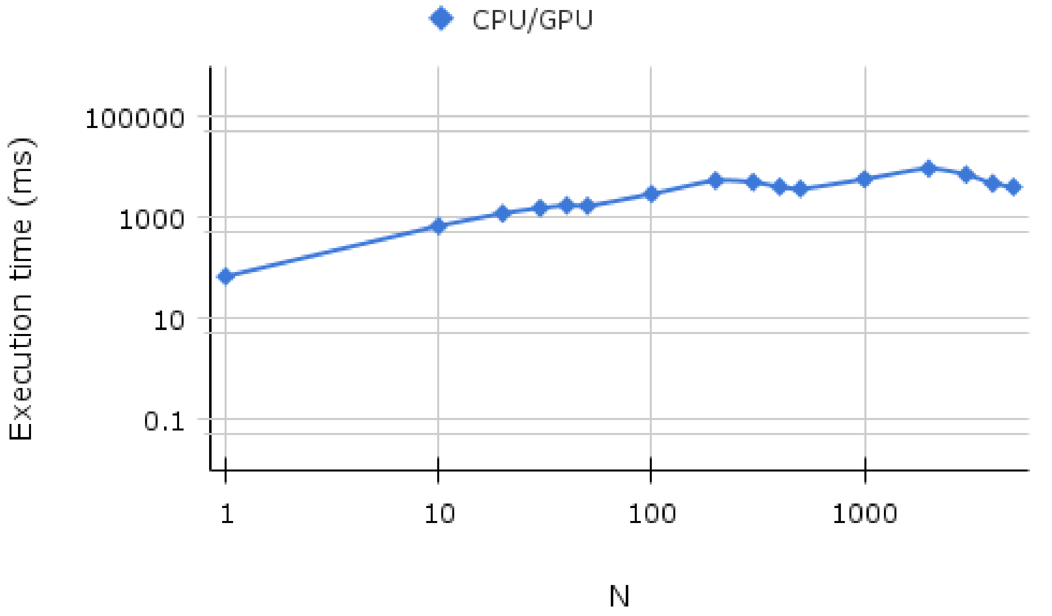
In this section, the performance of the single time-step, multi-channel GPU imager is compared to the baseline single time-step multi-channel CPU imager. In the baseline single time-step, multi-channel CPU version, FFT was implemented using `fftw_plan_many_dft()`[17] from the `fftw` library, used to perform FFT on multiple (u,v) grids, producing multiple sky images in a single batch. A comparison of the run times corresponding to the CPU and GPU implementations

[17] https://www.fftw.org/fftw3_doc/Advanced-Complex-DFTs.html

of the imager is shown in Figures 3.15 and 3.16 below. Thus, in conclusion, the comparisons show that the imager's single time-step multi-channel GPU version is up to thousands of times faster than the baseline single time-step multi-channel CPU version.

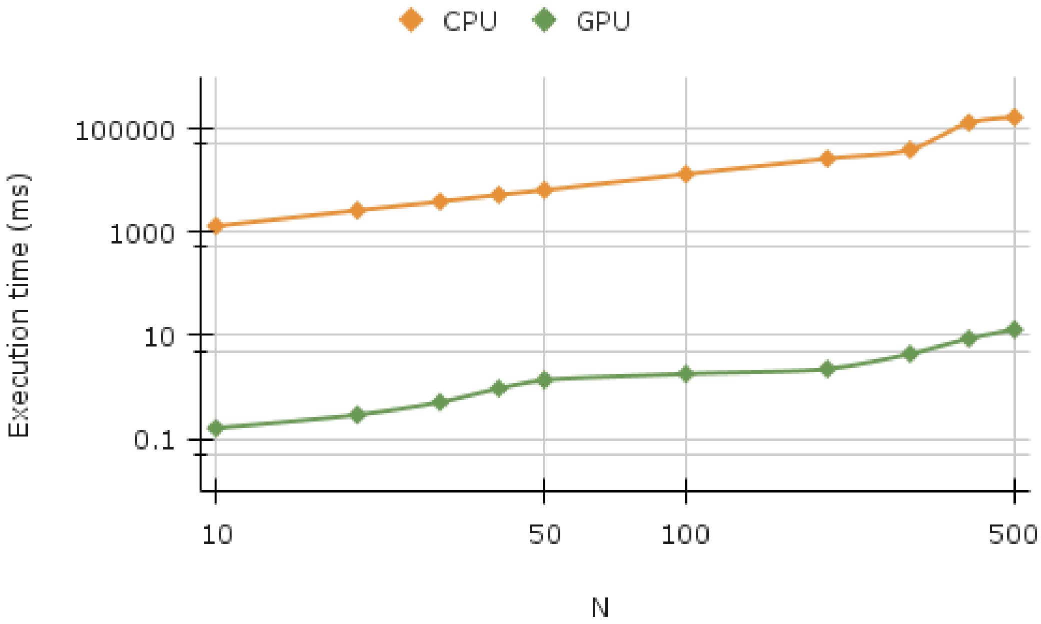


(a) FFT PlanMany execution run-time for image size: 180 × 180.

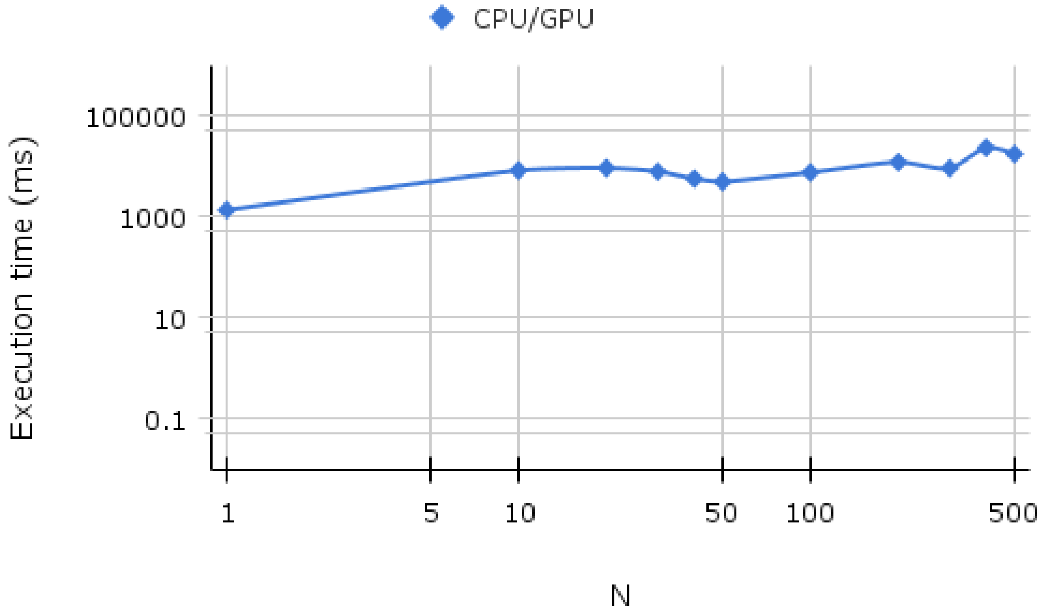


(b) Ratio of FFT PlanMany execution times corresponding to the CPU and GPU implementations of the imager for image size: 180 × 180.

Figure 3.15: Performance comparison of FFT execution times of the CPU (orange curves) and GPU (green curves) imagers to generate images of size: 180 × 180, for various values of $n_{channels}$, denoted as N in the above figures on Topaz. FFT is implemented using `fftw_plan_many_dft()` and `cufftPlanMany` in the CPU and GPU imager implementations, respectively. Notably, the imager's GPU version (CUDA implementation) is up to thousands of times faster than the starting CPU version of the imager.



(a) FFT PlanMany execution run-time for image size: 1024 × 1024.



(b) Ratio of FFT PlanMany execution times corresponding to the CPU and GPU implementations of the imager for image size: 1024 × 1024.

Figure 3.16: Performance comparison of FFT execution times of the CPU (orange curves) and GPU (green curves) imagers to generate images of size: 1024 × 1024, for various values of $n_{channels}$, denoted as N in the above figures on Topaz. FFT is implemented using `fftw_plan_many_dft()` and `cufftPlanMany` in the CPU and GPU imager implementations, respectively. Notably, the imager's GPU version (CUDA implementation) is up to thousands of times faster than the starting CPU version of the imager.

3.4 Summary

In summary, in this chapter, two standalone test versions of the GPU imager were described and benchmarked (starting from my supervisor's CPU imager), which generated all-sky images from EDA2 data. The main highlights in this development and benchmarking have been summarised below:

1. In the single time-step, single-channel version of the GPU imager, gridding was implemented using a 'GPU gridding kernel', and Fourier transform was performed using the `cuFFT` library.

2. To validate gridding, the (u,v) grid counters and the gridded visibilities of the GPU imager were compared with those of the CPU imager (validated earlier by my primary supervisor against well-established imagers). The (u,v) grid counter of the GPU imager matched exactly with that of the (u,v) grid counter of the CPU imager, signifying that an equal number of visibilities had been placed into each cell within the (u,v) grid, thereby validating that the gridding kernel worked correctly. However, there were subtle fractional differences of approximately ≤ 0.25 ($\leq 0.0001\%$) in the real and imaginary gridded visibilities upon closer examination. Since these differences were small, they could be discarded.

3. Similarly, the final real part of the dirty images produced by the CPU and GPU imagers were compared to validate the Fourier transform function. The resulting flux density values in the GPU images are closely aligned with those in the CPU images. There were minor discrepancies in the order of $\leq$ 0.001 uncalibrated units ($\leq 0.38\%$) observed in the final dirty images of the CPU and GPU imager, which resulted from the small fractional differences observed in the gridded visibilities. These differences were negligible and could be easily discarded. In conclusion, the single channel, single time-step GPU imager generated images equal to the CPU version within acceptable limits.

4. In the single-time-step, multi-channel version of the GPU imager, gridding was done parallelly across all the channels, either through CUDA streams or a 2D grid of CUDA blocks, and `cuFFTPlanMany` was used to configure batched FFTs on multiple gridded visibilities corresponding to different frequency channels of the same time-step.

5. The two test versions of the GPU imager were initially implemented and benchmarked on Topaz (with NVIDIA GPUs). A HIP version of the GPU imager was then developed by my supervisor to test the imager on Setonix (with AMD GPUs). The benchmarks of the CUDA version of the GPU imager were then compared with the benchmarks of the HIP version of the GPU imager. It was seen that the GPU gridding kernel and cuFFTPlanMany performed extremely fast on both Topaz and Setonix. It took only a few microseconds to perform gridding and Fourier transform for a single image.

6. In the end, the performance of the GPU imager's single time-step, multi-channel CUDA implementation was compared to the baseline single time-step multi-channel CPU imager. In the single time-step, multi-channel baseline CPU version, FFT was implemented using `fftw_plan_many_dft()` from the `fftw` library, used to perform FFT on multiple (u,v) grids, producing multiple sky images in a single batch. Thus, in conclusion, the comparisons showed that the single time-step multi-channel GPU version of the imager was up to thousands of times faster than the baseline single time-step multi-channel CPU version.

Chapter 4

Pilot FRB searches on EDA2 data

The goal of the project discussed in this chapter was to execute the imager discussed in Chapter 3 to image a few hours of EDA2 data in 100 ms time resolution and approximately 0.94 MHz bandwidth (fine channelised into 32 frequency channels) and to search for FRB candidates in these images. Since EDA2 was used to record visibilities, all-sky images of 180 × 180 were created. This was a pilot test search to verify the baseline CPU imager, see how the imager performed on a larger data set, devise a way of looking for FRBs and other transients with these data, and potentially even detect some interesting astrophysical events. Furthermore, since only a small amount of data (a few hours) was being processed, the system (i.e. supercomputer) used for processing this data was not that significant. Additionally, the CPU imager version was still more mature then, and the GPU version was not yet ready to be used on Setonix in production mode. Though the imager performed faster on Topaz than on Setonix, Topaz was decommissioned when processing this data. Hence, all the data processing done in this project was carried out on Setonix (also because Setonix is the ultimate target platform for this project), and the CPU imager was chosen to perform this pilot search for FRBs using EDA2 data.

In this chapter, Section 4.1 summarises the data used for performing this search, along with the different steps followed to prepare this data. Section 4.2 explains how this data was analysed. Section 4.3 and 4.4 details the candidates obtained after shortlisting and removing possible candidates due to RFI to check for FRBs. Section 4.5 concludes with the summary of the results.

4.1 Data Preparation

EDA2 visibilities were recorded in 100 ms time resolution and bandwidth of about 0.94 MHz fine channelised into 32 frequency channels (centre frequency of 230 MHz). These visibilities were imaged into 180 × 180 all-sky images. The observations were recorded in June and July 2023, each one minute long. Thus, the total number of images obtained from all the observations was:

- 6 June 2023 (June data): 38 observations (≈ 38 minutes duration), which resulted in 38min × 60sec/min × 10 images/sec × 32 channels = 729,600 images.

- 9 July 2023 (July data): 108 observations (≈ 108 minutes duration), which resulted in 108min × 60sec/min × 10 images/sec × 32 channels = 2,073, 600 images.

Figures 4.1 and 4.2 show an example of eight such all-sky images corresponding to four channels of a single time step.

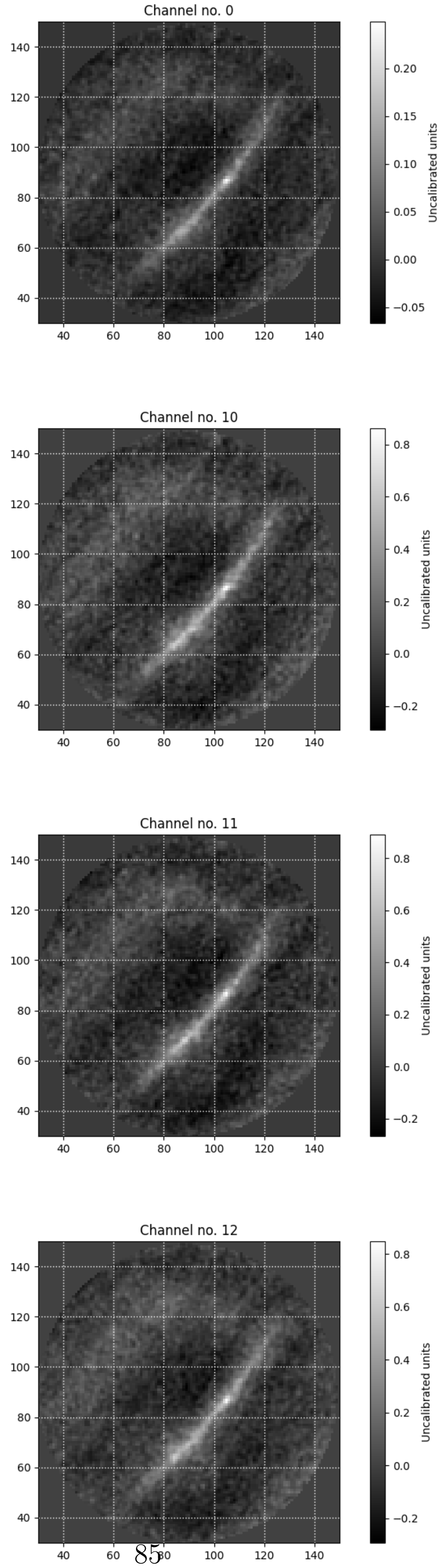


Figure 4.1: An example of four all-sky images, generated using data set 2 (2023-07-09), at UTC 13:43:23, from four channels, with a centre frequency = 230 MHz. The size of each image is 180 × 180, and the non-sky pixels have been masked.

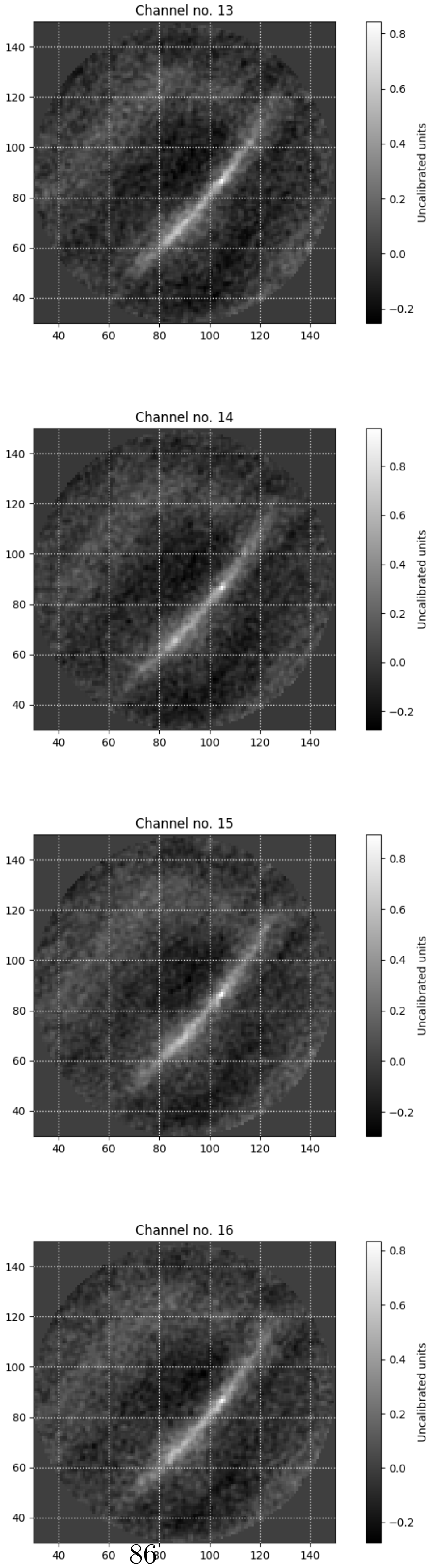


Figure 4.2: An example of four all-sky images, generated using data set 2 (2023-07-09), at UTC 13:43:23, from four channels, with a centre frequency = 230 MHz. The size of each image is 180 × 180, and the non-sky pixels have been masked.

Using existing software described in Anderson et al. (2021)[1], for each pixel (i.e. for every direction in the sky), the following steps were followed using these images as input to prepare data for FRB searches:

1. For each of the on-sky pixels present in each of 180 × 180 images, a dynamic spectrum plot was generated using all the images obtained across all the time steps and frequency channels. An example dynamic spectrum is shown in Figure 4.3. In this dynamic spectrum plot, the horizontal axis (x-axis) represents the time steps, and the vertical axis (y-axis) represents the frequency. The different time steps increase from left to right, and frequency increases from bottom to top.

2. For every on-sky pixel, a de-dispersed time series was generated for different DM trials, ranging from 150 to 1000 pc cm^{-3}, in steps of 150 pc cm^{-3}. This DM range was chosen since FRBs have their DMs predominantly spread in the 200 to 1700 pc cm^{-3}. Additionally, this DM range was also chosen as the sensitivity of EDA2 limits the FRB searches to low-DM events (Sokolowski et al., 2024b). Also, from Equation 1.1, it can be seen that with this search design, the relative dispersion across a 0.94 MHz band at 230 MHz will increase by about one 100 ms time step (i.e. time resolution of the dataset) with each 150 pc cm^{-3} increment. Hence, this value of DM increment was chosen.

3. The last step was to iterate through all values of these time-series files, which was generated for every pixel to look for pixels (i.e. candidate events) that had S/N $>$ 5 and store the results in a text file corresponding to the June and July data respectively. This threshold of S/N $>$ 5 was chosen as normally while searching for FRBs and transients, this threshold of SNR is chosen as the starting point. This is done (i) to isolate the best possible candidates from the remaining candidates and (ii) to reduce the number of candidates for inspection.

[1] Link to the software that was used: https://github.com/marcinsokolowski/frb_search

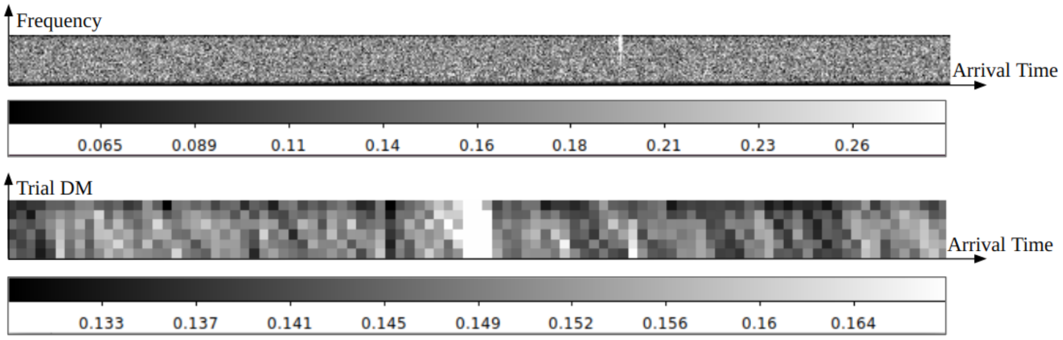


Figure 4.3: The dynamic spectrum (top-panel) and the heat map of the DM/arrival time phase space (bottom-panel) at pixel (88, 59), with (RA, DEC):(14:03:06, −59:00:44)

These text files (one for each June and July data) formed the input for further analysis, explained in Section 4.2 below.

4.2 Data Analysis

The histograms in Figure 4.4 visualise the distribution of S/N values in the June and July data sets. The vertical red line is added at the S/N = 10 on the x-axis. The number of candidates kept dropping until S/N = 10, and then they stayed more or less constant from S/N = 10 onwards. From Figure 4.4, it can be seen that the S/N distributions of the June and July data are similar, if not the same, and this difference between these distributions did not affect the analysis of this data.

4.2.1 Properties of the Image Noise

To further understand the nature of the distribution of candidates and noise in the images, the noise/candidate distribution both in the non-sky pixels and the on-sky pixels were analysed. Figures 4.5a and 4.5b illustrate two examples of the histogram of noise/candidate distribution in the non-sky and on-sky pixels from an image in the June data. After inspecting the two distributions visually, it is seen that the noise/candidate distribution for both the off-sky and on-sky

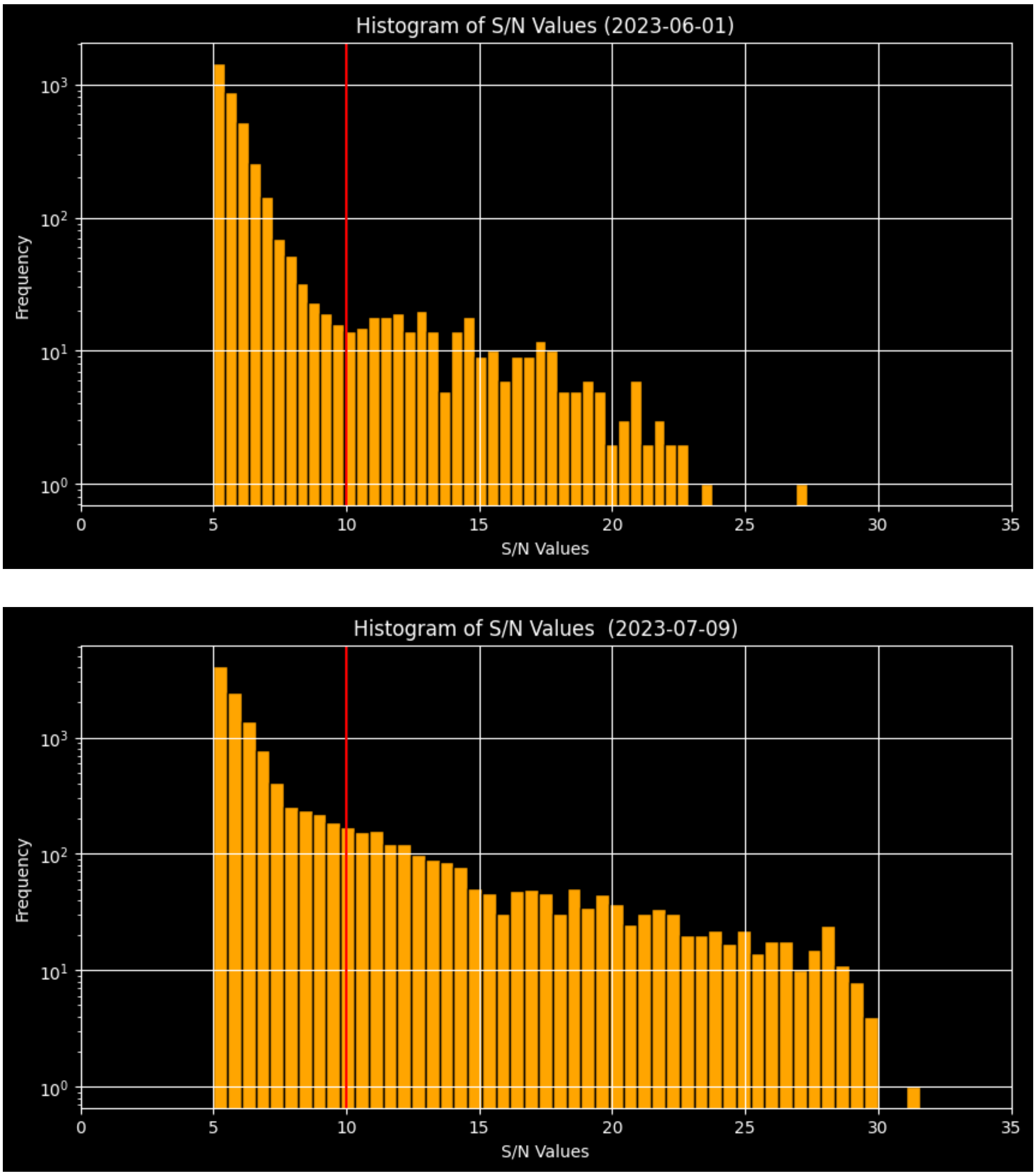


Figure 4.4: The histogram of S/N values of all the candidates from the June 2023-06-01 data (top panel) and July 2023-07-09 data (bottom panel).

pixels is similar and follows a Normal distribution. The standard deviation for a few images from the June and July data was also calculated for the off-sky and on-sky pixels. The standard deviation for off-sky pixels was ≈0.008 in June data and ≈0.012 in July data. The standard deviation for on-sky pixels was ≈0.011 in June data and ≈0.018 in July data. Units have not been included, as these images were not flux-calibrated. Thus, these findings indicate that the noise/candidate distributions between off-sky and on-sky pixels in both datasets from June and July are similar. However, the on-sky pixels have slightly higher noise, which may be caused by noise due to astronomical sources and RFI. However, it is still correct to analyse the data assuming that on-sky pixels are normally distributed.

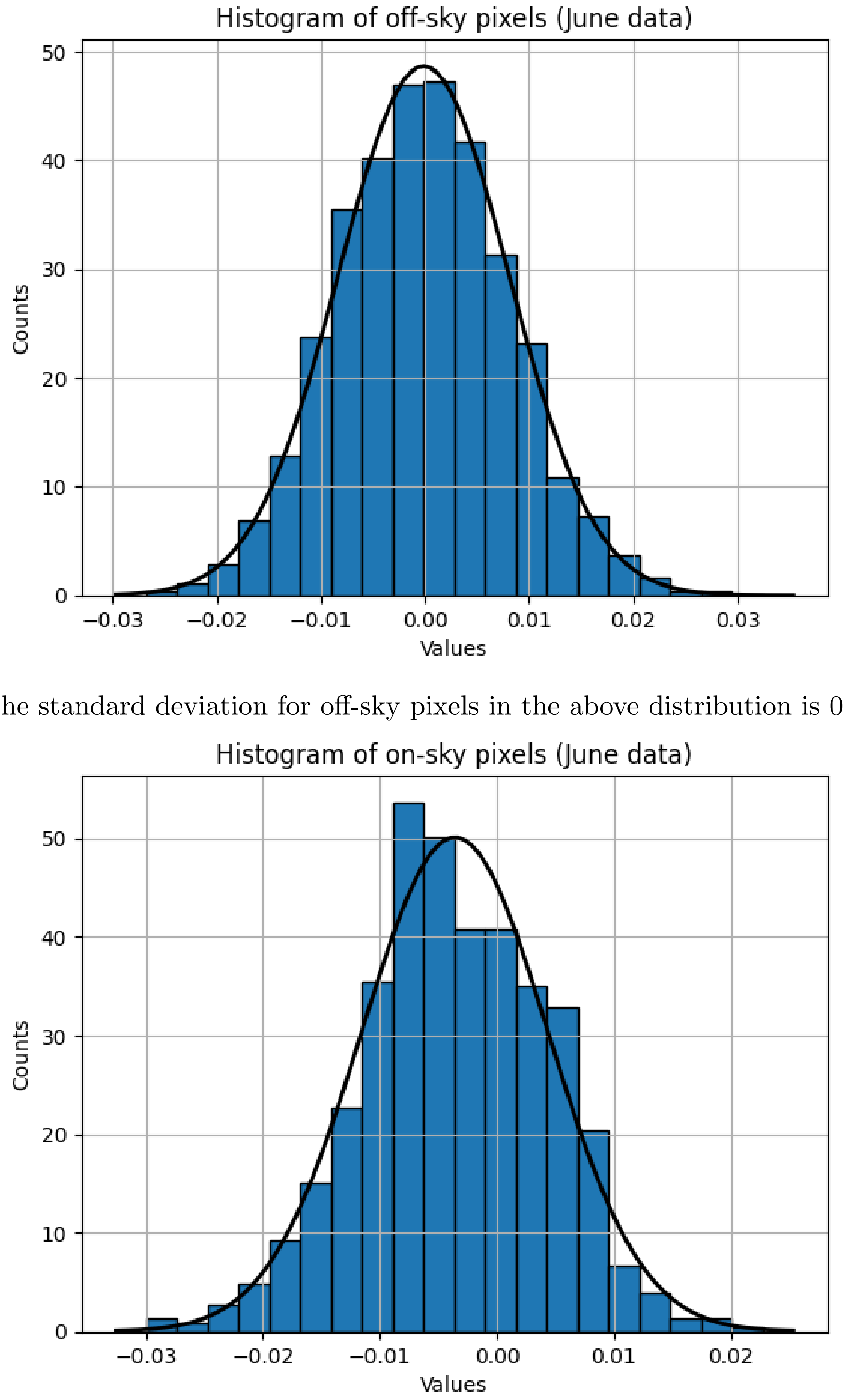
Thus, assuming a Normal (Gaussian) distribution of the noise, the expected number of false positives with S/N > 5 can be calculated as the probability multiplied by the total number of trials. The total number of trials (N_{trials}) is given by:

$$N_{trials} = N_{obs} \times N_{totalpixels} \times N_{DM} \times N_t, \tag{4.1}$$

where N_{obs} is the number of observations, $N_{totalpixels}$ is the number of pixels in the image(only on-sky pixels, i.e. (50, 50) to (50, 150) have been considered in this calculation), N_{DM} is the number of DM trials and N_t is the number of time-steps in each observation. These parameters for the June and July data are summarised in Table 4.1.

Substituting the parameter values in Table 4.1 for the above parameters results in 1368000000 and 3888000000 trials for the June and July data, respectively. The probability for noise to be detected as a candidate in a 5σ detection is 0.000000573303/2[2], resulting in around 392 and 1115 false positive candidates from the June and July data respectively, for this S/N threshold. Thus, the number of noise-caused candidates we expect to observe for this S/N threshold are 392 and 1115 from the June and July data, respectively. These calculations have also been summarised in Table 4.1. These were the theoretical estimations,

[2] https://en.wikipedia.org/wiki/Normal_distribution



(a) The standard deviation for off-sky pixels in the above distribution is 0.008.

(b) The standard deviation for on-sky pixels in the above distribution is 0.011.

Figure 4.5: The histogram of noise/candidate distribution in the off-sky pixels(top panel) and on-sky pixels(bottom panel) from an image from the June data.

and the next step was to see how many candidates were observed due to noise in the June and July data.

Parameters	**June data**	**July data**
Number of observations (N_{obs})	38	108
Number of images (N_{images})	729,600	2,073,600
Number of pixels ($N_{totalpixels}$)[1]	10000	10000
Number of DM Trials (N_{DM})[2]	6	6
Number of time-steps in each observation (N_t)[3]	600	600
Number of trials (N_{trials})[4]	1368000000	3888000000
Probability (5σ detection)	0.000000573303	0.000000573303
Candidates with S/N > 5 (Expected)[5]	392	1115
Average expected candidates per single image	0.000537281	0.000537712

[1] Only on-sky pixels, i.e., (50, 50) to (50, 150) have been considered, i.e., ≈ 100 × 100.

[2] From 0 to 900 pc cm^{-3}, in steps of 150 pc cm^{-3}.

[3] Calculated as: 1 minute/(100 ms) = 600 time-steps in each observation.

[4] Calculated according to Equation 4.1.

[5] Candidates expected due to noise, calculated as: N_{trials} × probability.

Table 4.1: The theoretical estimations of the number of candidates caused due to noise for an S/N threshold of 5σ for the June and July data.

On inspection, in the June data, there were 3,726 candidates with S/N > 5, spread across 30 time steps. Similarly, in the July data, there were 11,920 candidates spread across 70 time steps with S/N > 5. Thus, the observed number of candidates was nearly 10 times higher than expected! Some time steps had a lot of candidates, and in some cases, even up to 300 candidates from a single time-step. These time steps could have been affected by a strong RFI causing side-lobes in the images, which resulted in false positives. Upon visual inspection of each image corresponding to time steps with more than 20+ candidates, it was seen that none of these candidates looked like astrophysical signals and had been caused by RFI. Consequently, these time-steps with more than 20 candidates were excluded from further analysis. Thus, after excluding these candidates that could be caused by RFI (i.e. 2945 candidates from June data and 11346 candidates from July data), the total number of observed possible candidates was 781 (389 more than calculated) and 574 (541 less than calculated) from the June and July data, respectively. This discrepancy was not fully understood, but the expected and observed numbers of candidates matched within an order of magnitude. The estimates and RFI exclusion only agree at the order of magnitude level, as the candidates were searched in the de-dispersed time series, complicating this comparison. Therefore, this order of magnitude agreement is taken as an indication only. Based on these calculations, out of the 3,726 and 11,920 candidates from the June and July data, 2,945 and 11,346 candidates could have been caused by RFI. This analysis is also summarised in Table 4.2.

Number of candidates	**June data**	**July data**
Total Candidates with S/N $>$ 5 (Expected)	392	1115
Total Candidates with S/N $>$ 5 (Observed)	3726	11920
Potential Candidates (Observed)[1]	781	574
Candidates due to RFI [2]	2945	11346

[1] After eliminating candidates from observations that had 20+ candidates from a single observation.

[2] Number of false positive candidates caused due to RFI (i.e. found in images with >20 candidates).

Table 4.2: Summary of the number of candidates with S/N $>$ 5, caused by noise and RFI for June and July.

It was impossible to visually inspect over 3000 candidates; therefore, a higher threshold of S/N $>$ 10 was applied to limit the number of candidates for inspection. This narrowed down the candidates to 272 and 1829 candidates for inspection from the June and July data, respectively.

4.2.2 Potential RFI Sources

The next step was to check if these candidates overlapped with the Digital Television (DTV) or FM radio bands. The FM radio broadcast band in Australia spans 87.5 to 108 MHz[3]. Thus, none of these candidates (corresponding to time-steps with 20+ candidates) overlapped with the FM radio bands. Additionally, in Australia, the DTV signals are typically broadcast[4] across both the Very High Frequency (VHF band III), from 170 to 230 MHz and the Ultra High Frequency (UHF) band, from 520 to 820 MHz. Thus, some of these candidates (which were near the horizon) could be due to the DTV signals coming from the direction of towns near the MRO. These candidates (which were not on the horizon) could also be unintended transmissions from satellites[5]. They could also be side lobes from very strong RFI signals on the horizon or elsewhere. Consequently, these observations with more than 20+ candidates were excluded from further analysis. This further narrowed down the candidates to 12 and 8 from the June and July data, respectively, for further inspection. These final short-listed candidates are summarised in Table 4.3 and 4.4.

[3] https://www.acma.gov.au/broadcasting-transmitter-apparatus-licences#where-you-can-broadcast

[4] https://nulifetv.com.au/what-are-the-digital-tv-frequencies-used-in-australia/

[5] In Section 4.4.2, a further check has been done to see if any of the final short-listed candidates, i.e. 12 from June data and 8 from the July data matched with the positions of known satellites during the observations.

Table 4.3: The description of the candidates short-listed from the June data.

Candidate Number	**UTC**	**RA**	**DEC**	**S/N**	**DM (pc cm^{-3})**	**Near Horizon**[1]
1A	10:37:21	14:24:09	-38:00:46	11.3	0 - 150	No
1B	10:37:21	15:00:43	-18:14:40	11.9	0 - 150	No
1D	10:37:21	16:06:22	-32:54:45	27.4	0 - 150	Yes
3A	10:36:22	16:02:36	-35:00:22	19.6	150 - 300	Yes
3B	10:36:22	16:05:01	-35:05:11	18.5	0 - 150	Yes
4A	10:33:22	15:49:07	-40:55:57	23.6	0 - 150	Yes
6A	10:38:21	13:40:49	-16:26:25	10	0 - 150	No
6B	10:38:21	14:29:26	-32:08:32	11.6	0 - 150	No
6C	10:38:21	14:51:28	-14:05:20	11	0 - 150	No
6D	10:38:21	16:07:35	-30:36:49	17.7	0 - 150	Yes
6E	10:38:21	15:47:32	-11:13:21	11.5	0 - 150	No
7A	10:03:08	13:46:50	7:38:19	16.6	0 - 150	No

[1] An elevation limit of 30 degrees was used to classify whether a candidate was on the horizon.

Table 4.4: The description of the candidates short-listed from the July data.

Candidate Number	**UTC**	**RA**	**DEC**	**S/N**	**DM (pc cm^{-3})**	**Near Horizon**[1]
2A	11:26:33	14:03:06	-59:00:44	19.8	0 - 150	No
4A	11:25:32	14:03:20	-44:58:10	21.4	0 - 150	No
5A	13:01:44	13:06:36	-33:45:26	10.4	750 - 900	No
7A	11:15:54	13:06:28	30:35:51	21.8	0 - 150	No
7B	11:15:54	13:06:28	30:35:51	10.8	450 - 600	No
8A	11:14:54	13:01:04	6:43:20	23	0 - 150	Yes
8B	11:14:54	13:01:04	6:43:20	13.3	450 - 600	Yes
9A	12:18:02	09:31:15	-46:57:56	21.3	0 - 150	Yes

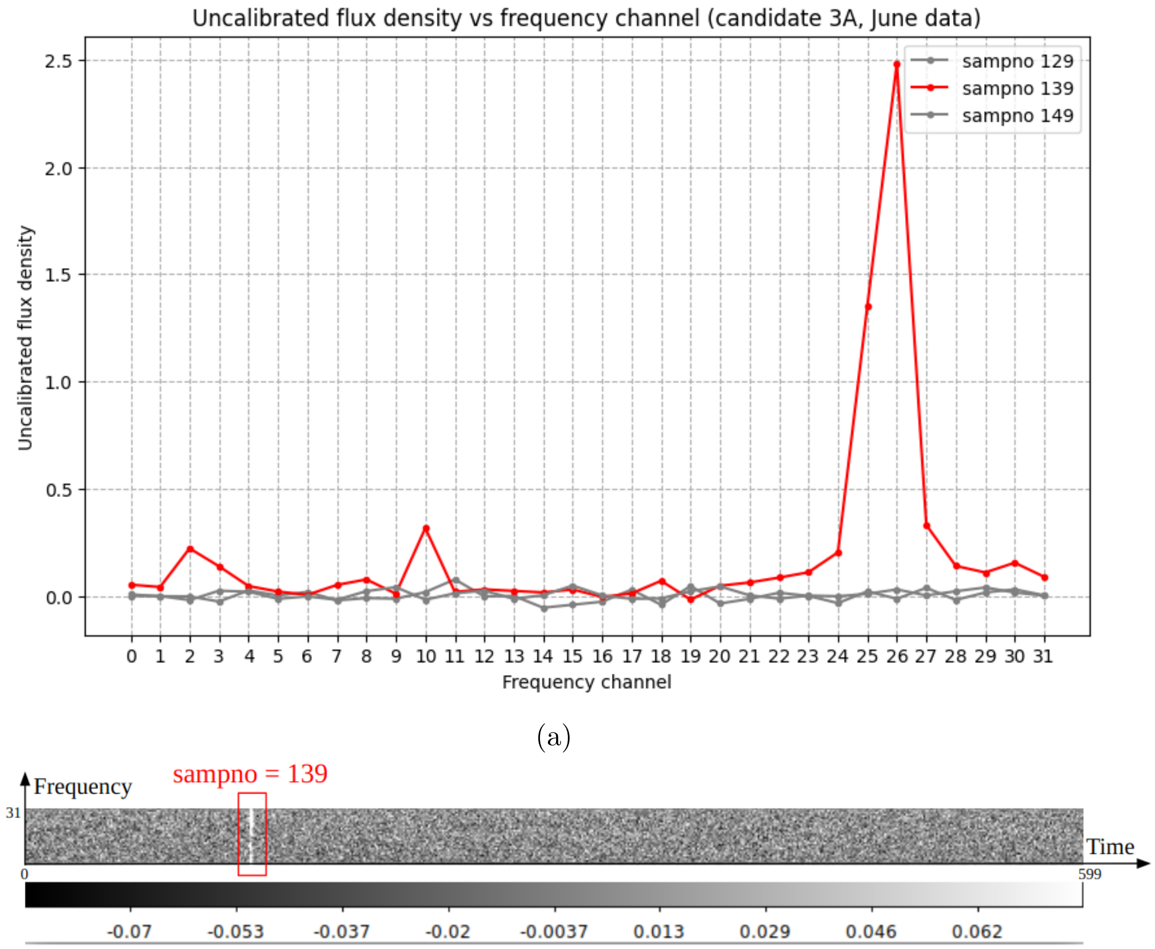
[1] An elevation limit of 30 degrees was used to classify whether a candidate was on the horizon.

4.2.3 Analysing Spectral Properties

To analyse the spectral properties of the candidates, graphs of the uncalibrated flux densities as a function of frequency channel for the bright candidates from both the June and July data were created. These are the candidates that were clearly visible in the images. The candidates clearly visible in the images in the June data were: 1A, 1B, 1D, 3A, 3B, 4A, and 6D. Similarly, the candidates clearly visible in the images in the July data were 2A and 4A. In Figures 4.6 and 4.7, two such graphs corresponding to candidates 3A and 4A from the June data are shown. In Figures 4.8 and 4.9, two such graphs corresponding to candidates 2A and 4A from the July data are shown.

From Figures 4.6 and 4.7, it can be seen that though the signal seems to be observed across all 32 channels in the dynamic spectra, there is always a single channel with very high power compared to other channels, and it is channel 26 for candidates 3A and 4A from the June data. Thus, these candidates are most likely RFI and not astrophysical signals, as they were only observed at a single time-step in a single frequency channel(very narrow band), while astrophysical sources typically have much smoother and continuous spectra. In Section 4.4, further analysis has been done to see if these candidates could be signals coming from near the horizon.

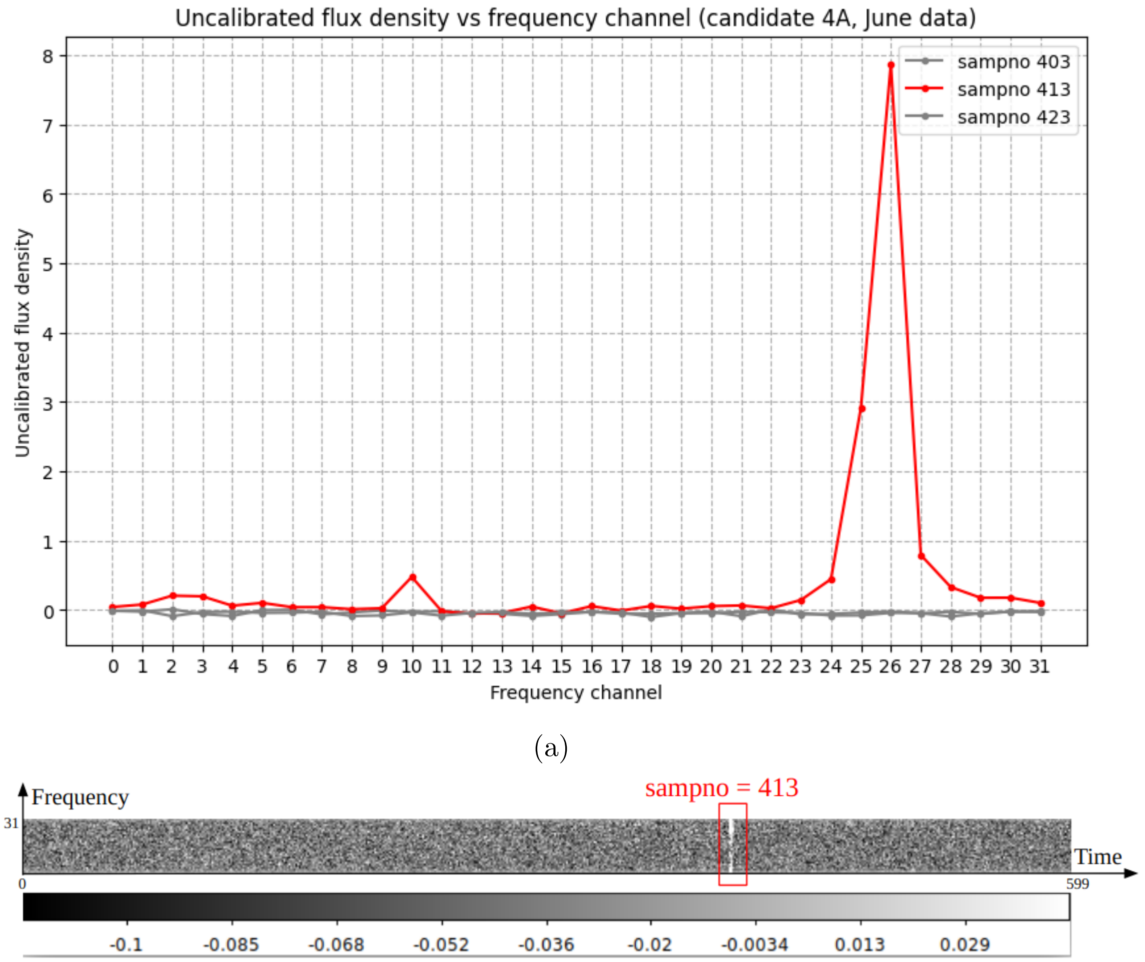
Similarly, from Figures 4.8 and 4.9, it can be seen that though the signal seems to be observed across all 32 channels, there are 10-15 channels with very high power compared to other channels. The signals seem to be comparatively weaker in channels 0-20, and then there is a steady increase in the power from channel 20 onwards and a steady decrease in power after channel 29. The signal has higher power across channels 26-30, with peak power observed at channel 29. Thus, candidates 2A and 4A, from the July data, are slightly more likely to be of astrophysical origin, but they still have a very narrow band ($\approx$ 290 kHz), which still makes them more likely due to RFI. However, in Section 4.4, further analysis has been done to see if these candidates could be due to astrophysical sources.



(a)

(b)

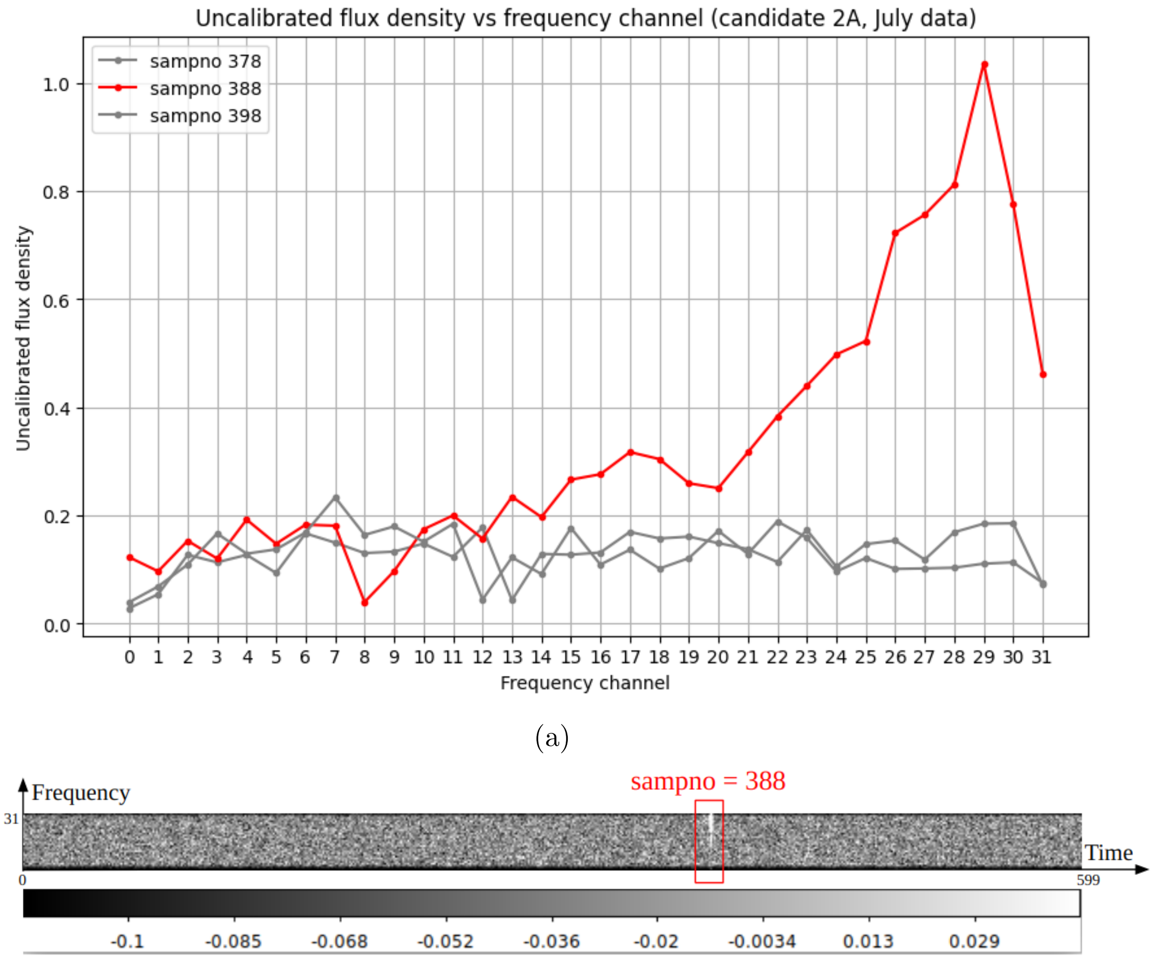
Figure 4.6: The uncalibrated flux densities as a function of frequency channel for candidate 3A (top panel) and the corresponding dynamic spectrum (bottom panel) from the June data. The red curve shows the signal seen across the 32 different channels at the specific time-step 139 (i.e. sampno in the image). As a reference, the grey curves (Figure a) depict the noise observed across the same 32 channels in the time steps before and after time-step 139.



(a)

(b)

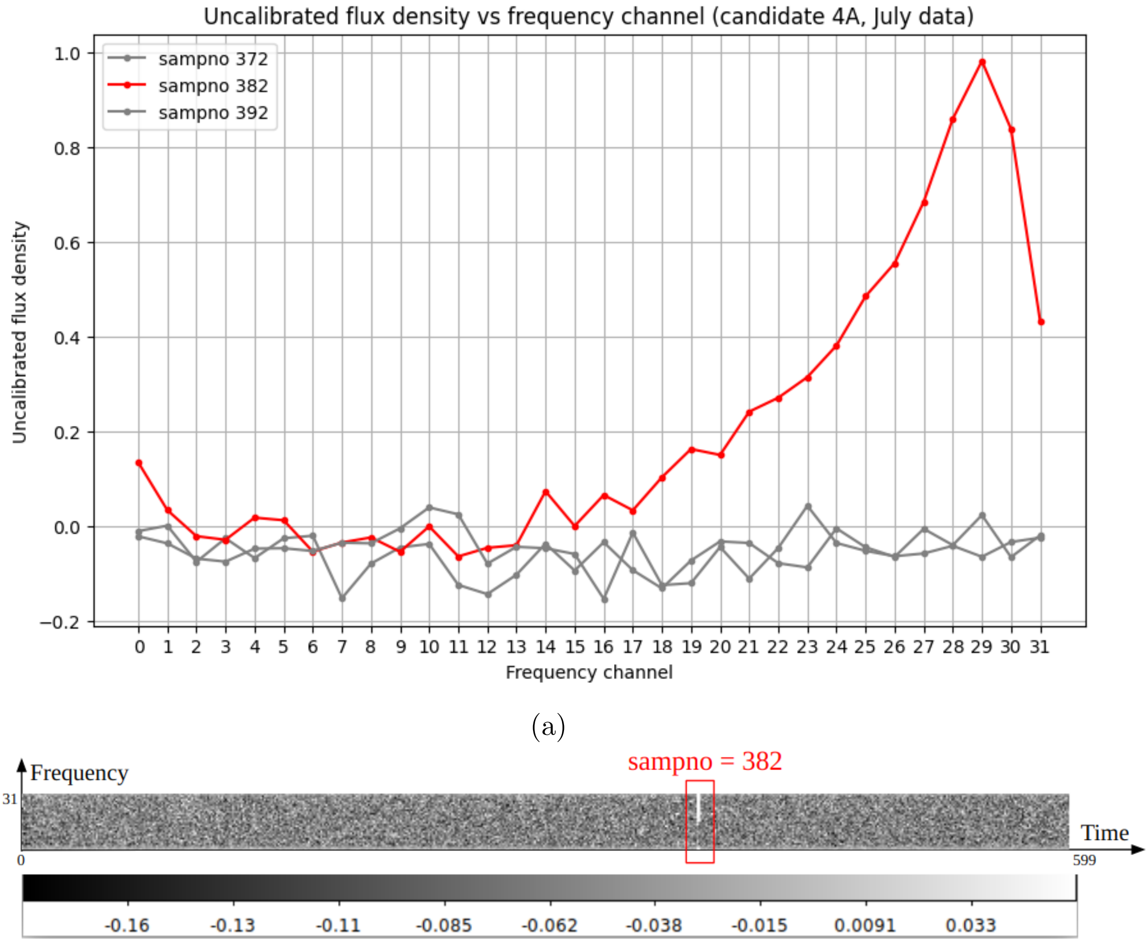
Figure 4.7: The uncalibrated flux densities as a function of frequency channel for candidate 4A (top panel) and the corresponding dynamic spectrum (bottom panel) from the June data. The red curve shows the signal seen across the 32 different channels at the specific time-step 413 (i.e. sampno in the image). As a reference, the grey curves (Figure a) depict the noise observed across the same 32 channels in the time steps before and after time-step 413.



(a)

(b)

Figure 4.8: The uncalibrated flux densities as a function of frequency channel for candidate 2A (top panel) and the corresponding dynamic spectrum (bottom panel) from the July data. The red curve shows the signal seen across the 32 different channels at the specific time-step 388 (i.e. sampno in the image). As a reference, the grey curves (Figure a) depict the noise observed across the same 32 channels in the time steps before and after time-step 388.



(a)

(b)

Figure 4.9: The uncalibrated flux densities as a function of frequency channel for candidate 4A (top panel) and the corresponding dynamic spectrum (bottom panel) from the July data. The red curve shows the signal seen across the 32 different channels at the specific time-step 382 (i.e. sampno in the image). As a reference, the grey curves (Figure a) depict the noise observed across the same 32 channels in the time steps before and after time-step 382.

4.3 Near horizon candidates

For each potential candidate (i.e., 12 from the June data and 8 from the July data), the images at the corresponding time steps were inspected. A check was done to see if the candidates in these images were on the horizon, which meant they were likely RFI from one of the nearby towns surrounding MRO. An elevation limit of 30 degrees was used to classify whether a candidate was on the horizon. Thus, all candidates with an elevation lesser than 30 degrees were considered as 'near horizon' candidates, and candidates above this elevation limit were considered as 'above horizon' candidates.

Figure 4.10 shows the five 'near horizon' candidates from the June data. These candidates were observed within a few minutes of each other. All five candidates were very well visible in the images, and the location of the candidates in the images is marked with white circles. For these candidates, their Azimuth and Elevations were calculated (see Table 4.5) to see where these radio signals could be coming from. These calculations show that these radio signals could be coming from towns near MRO, in the directions of Cue and Leonora Airport. This is illustrated in Figure 4.11.

Candidate Number	Azimuth (Degrees)	Elevation (Degrees)
4A	124.01	29.46
3A	118.02	26.1
3B	118.26	25.64
1D	115.97	24.96
6D	113.57	24.24

Table 4.5: The summary of on-horizon candidates from the June data.

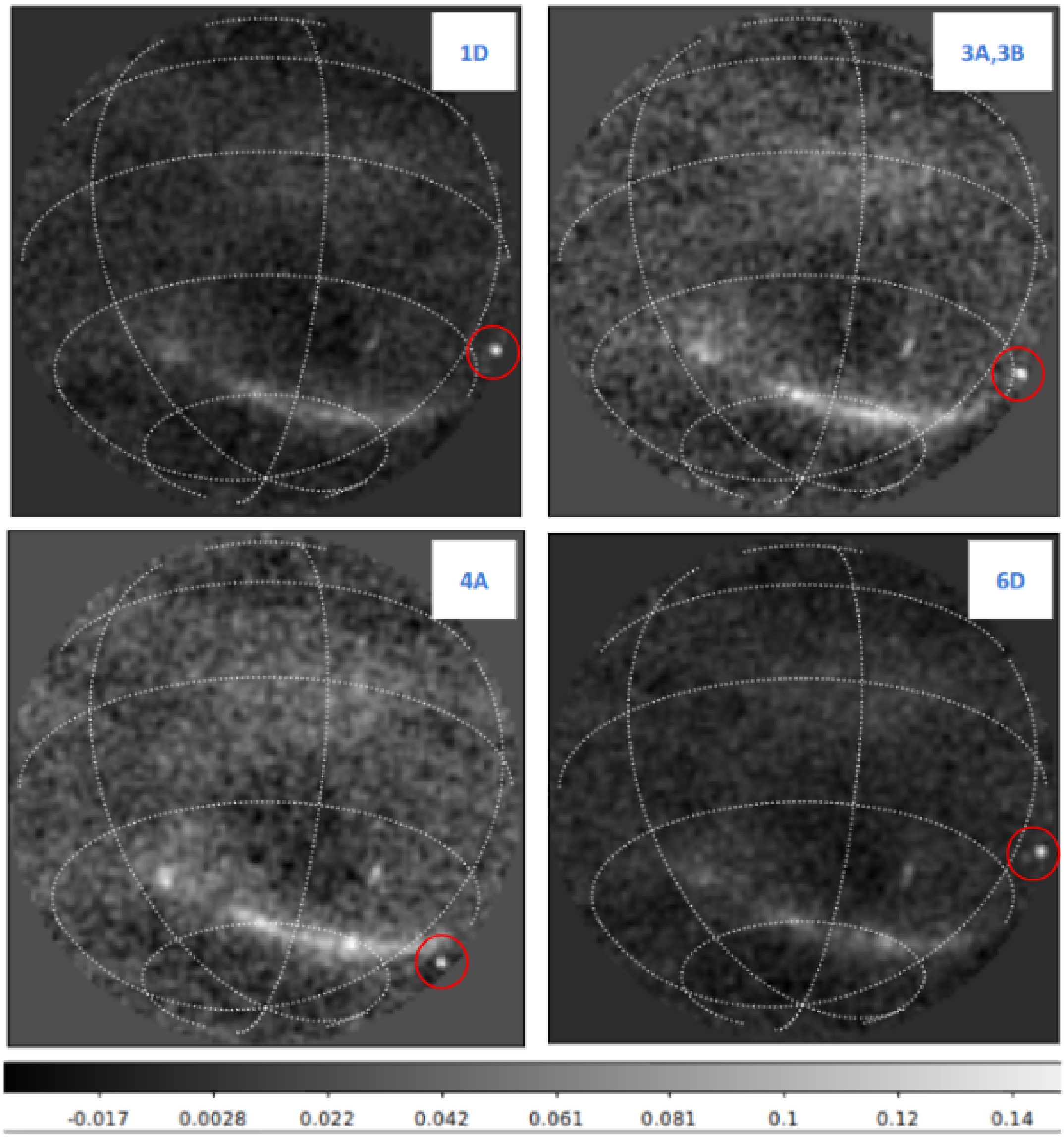


Figure 4.10: The candidates near the horizon from the June data. In each image, the candidates are circled in red and appear as bright white spheres at the corners of the large circle, separating the off-sky pixels.

4.4 Above horizon candidates

Each of the remaining (above the horizon) candidates, i.e. eight from the June data, had their DM within 150 pc cm^{-3}. All except one candidate (5A) from the July data had their DM within 150 pc cm^{-3}. The images containing 5A were inspected, and this candidate was not visible in the images. Furthermore, no

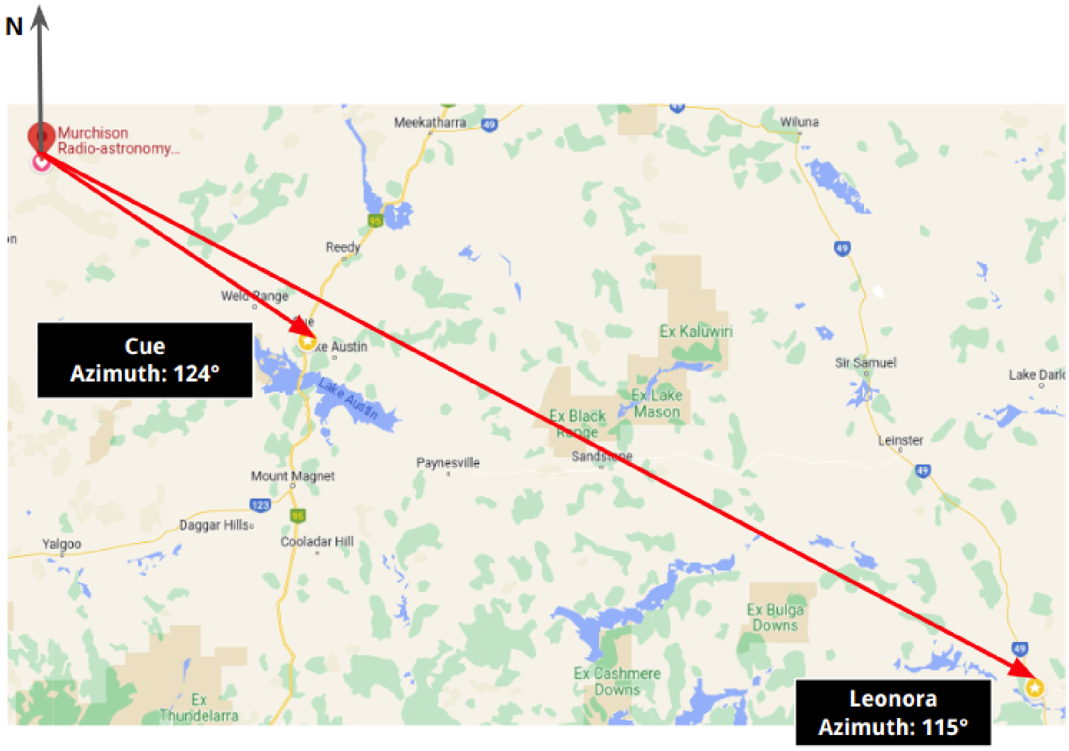


Figure 4.11: The almost on-horizon candidates obtained from the June data. *Image credits: Google Maps.*

vertical sweep was seen in the dynamic spectrum as well. Figures 4.12 and 4.13 show examples of two candidates each from the June and July data, respectively. Additionally, a scatter plot of the distribution of DM vs S/N values of all candidates for both June and July data is shown in Figure 4.14. It is seen that most of the high S/N candidates, with S/N $>$ 10, have a DM in the range of 0 - 150 pc cm^{-3}. Very few candidates with S/N $>$ 10 have high DMs. This DM check eliminated the possibility of any of these candidates being FRBs, as most FRBs have their DMs spread in the range of 200 - 1700 pc cm^{-3}. So, the next check was to see if any of these candidates were due to pulsars or any nearby satellites, described in Section 4.4.1 and 4.4.2, respectively.

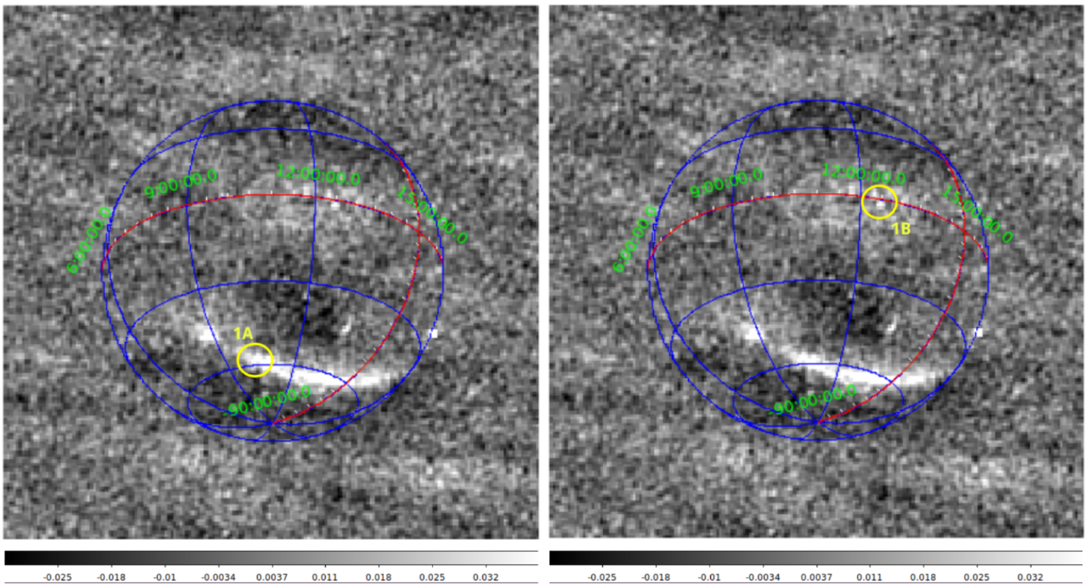


Figure 4.12: Candidate 1A (left) and 1B obtained (right) from the June data set.

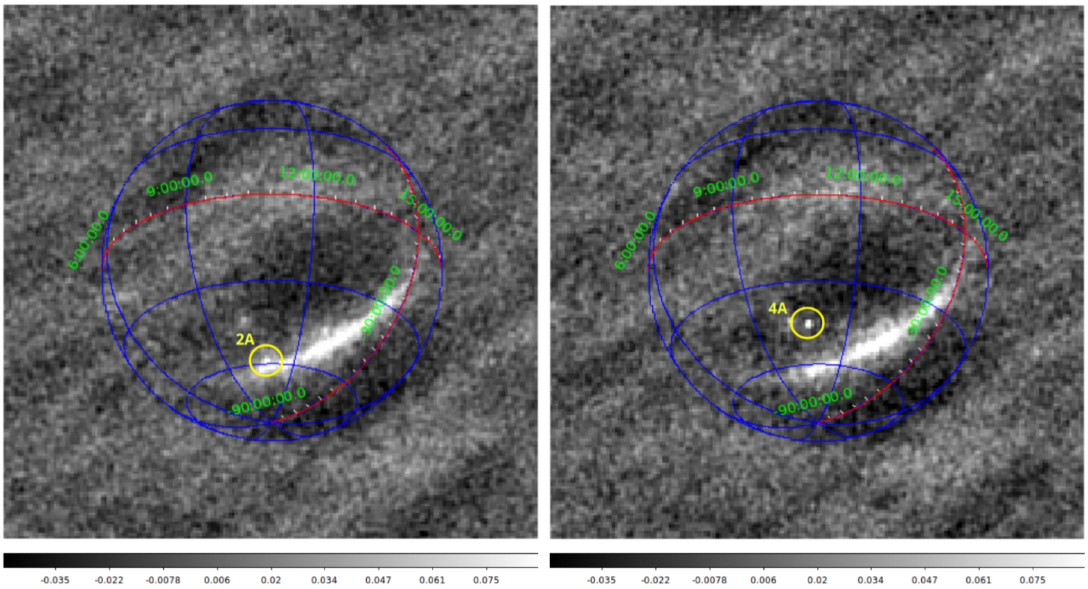
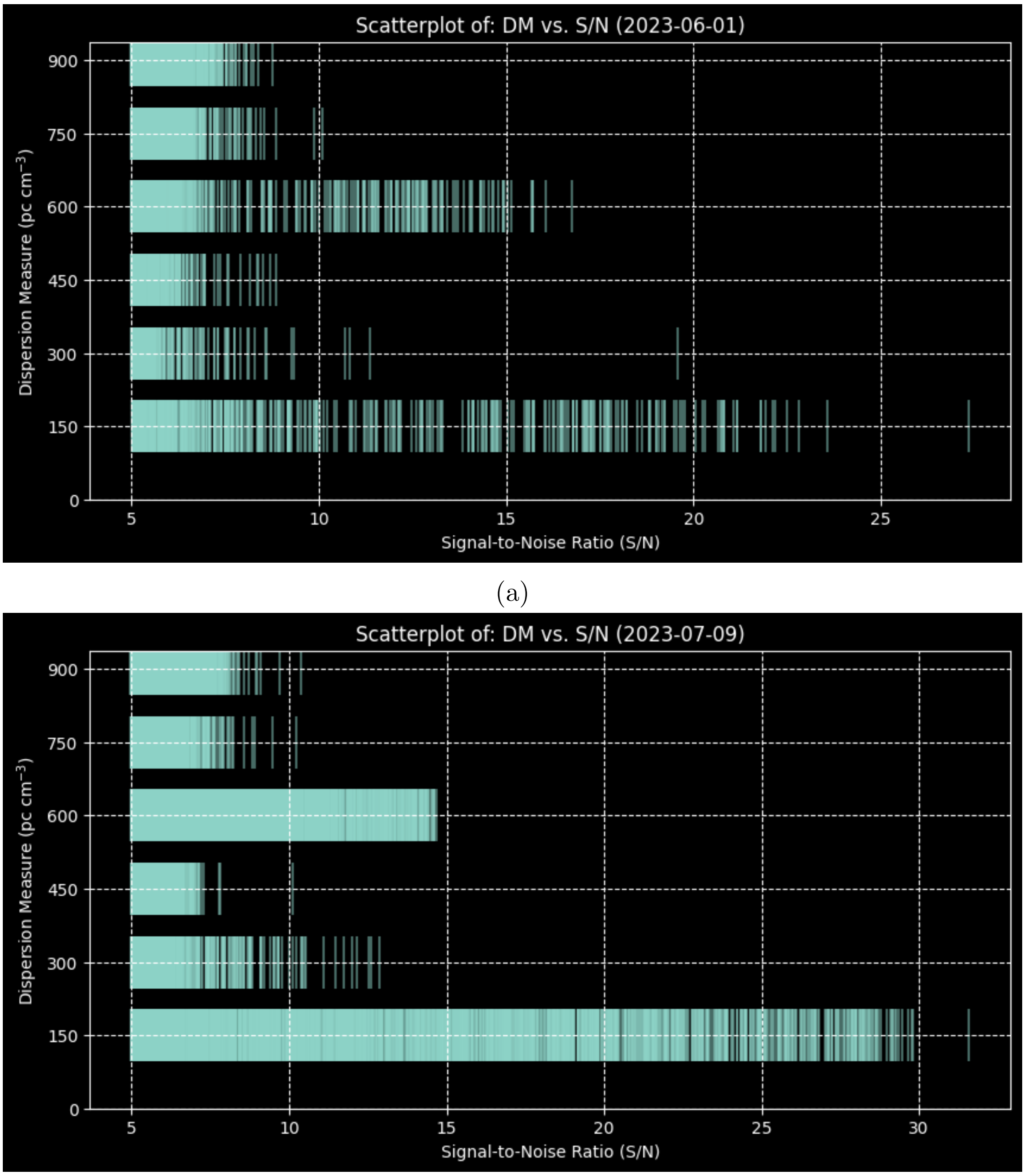


Figure 4.13: Candidate 2A (left) and 4A obtained (right) from the June data set.

4.4.1 Cross-matching with known pulsars

For the above-the-horizon candidates, their RA and DEC were checked if they matched with those of known pulsars from the Australian Telescope National Facility (ATNF) pulsar catalogue[6]. For the pulsar candidates, the following two

[6] https://www.atnf.csiro.au/research/pulsar/psrcat/



(a)

(b)

Figure 4.14: Scatter-plot of DM vs S/N values of all the candidates from the June data (top panel) and the July data (bottom panel).

checks were performed:

- **DM limit check:** Only vertical sweeps were observed in the dynamic spectra of all the short-listed candidates, suggesting that the maximum dispersive delay observed was less than 100 ms (i.e. Δ t $\leq$ 100 (ms)).

Hence, the maximum DM for any of these candidates could be calculated using the following equations:

$$\Delta t = 4.15 \times 10^6 \times DM \times ((v_1)^{-2} - (v_2)^2), \tag{4.2}$$

$$\Delta t \leq 100(ms). \tag{4.3}$$

Since the bandwidth used to record this data was known, to get the maximum DM value any of these candidates could have, we can substitute the following values for $v_1 = 230$ MHz and $v_2 = 231$ MHz.
Thus, Equation 4.2 can be reduced to:

$$\Delta t = 0.67645 \times DM. \tag{4.4}$$

Now, substituting Equation 4.4 in Equation 4.3, we get:

$$\begin{aligned} &0.67645 \times DM \leq 100 \\ &DM \leq 147.83. \end{aligned} \tag{4.5}$$

Thus, the maximum DM any of these candidates could have should not exceed 147.83 (i.e. approximately 150 pc cm^{-3}). This check eliminated pulsars above this DM limit.

- **Flux density check:** While processing this data, no amplitude and primary beam corrections were performed. Consequently, the flux densities in the images are not accurate, and the current estimates are only approximate. Indeed, this would have changed the results, especially for candidates away from zenith, but at that stage, it was not the primary goal of the project to apply beam correction to these images. This lack of beam correction implied uniform noise across the image, which was confirmed and verified by plotting graphs of sensitivity vs elevation for a few sample images. This analysis showed that the images did have nearly consistent sensitivity from zenith to horizon. This was also verified for a few images from the June data, and a similar trend was observed in all of them. Figure 4.15 shows a plot of standard deviation versus the X coordinate of the pixel (corresponds to moving from low elevation in the East, through zenith, to low at the West). Here, the standard deviation was calculated for all values within a radius of 5 pixels centred at the X coordinate of the pixel, and the Y coordinate was the centre of the image (i.e. here, Y = 90).

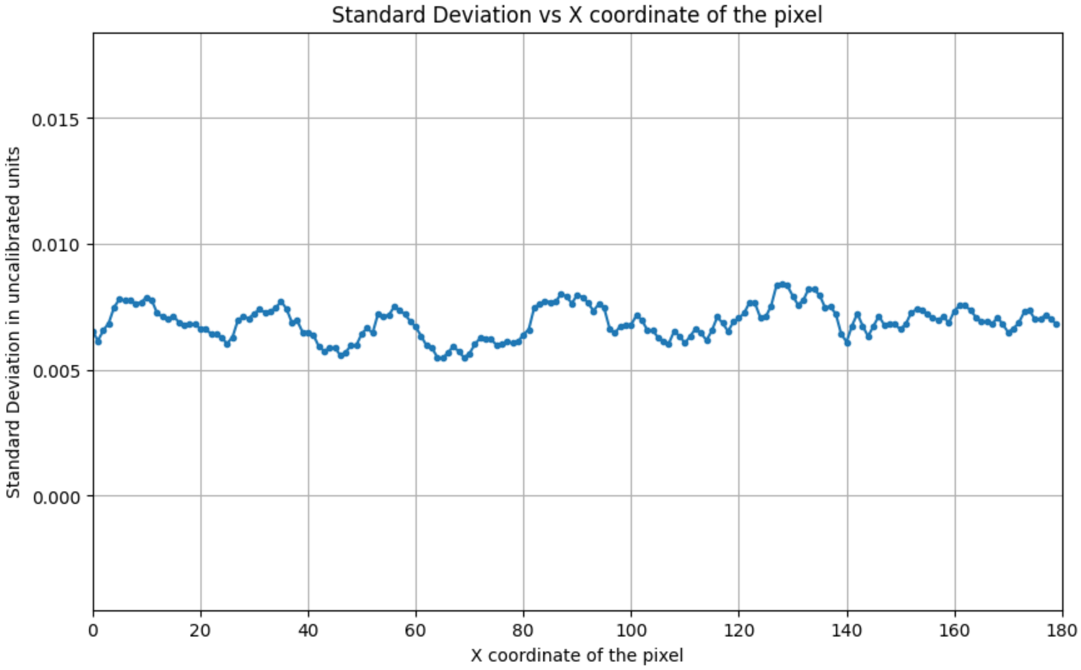
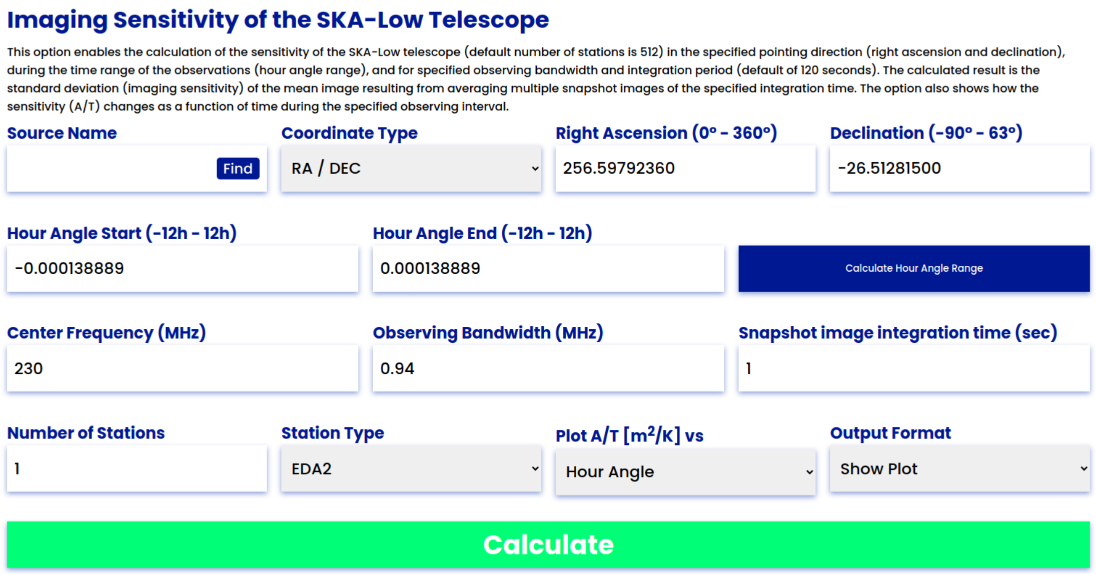


Figure 4.15: The standard deviation vs X coordinate of the pixel for an image from June data at time UTC 10:36:22 on 2023-06-01. Along the y-axis, the standard deviation is plotted in uncalibrated units of all values centres around the X coordinate of the pixel within a circle of radius 5 pixels. The x-axis represents the X-coordinate of the pixel around which the standard deviation calculations are centred. The standard deviation was calculated for all values within a radius of 5 pixels centred at the X coordinate of the pixel, and the Y coordinate was the centre of the image (i.e. here, Y = 90). It can be seen that the standard deviation distribution is nearly uniform, with all values ranging between 0.006 and 0.008.

These images' sensitivity (σ) was approximately 100 Jy. This was only a theoretical sensitivity of the individual 100 ms images, and it was calculated using the sensitivity calculator for SKA-low[7] (Sokolowski et al., 2022b). Using this calculator, as shown in Figures 4.16a and 4.17a, the imaging sensitivity can be calculated in a specified pointing direction (i.e. RA and DEC) during the time range of the observations for a specified bandwidth and integration period. The calculated result, as shown in Figures 4.16b and 4.17b, is the sensitivity of the image resulting from averaging multiple snapshot images of the specified integration time.

[7] https://sensitivity.skalow.link/

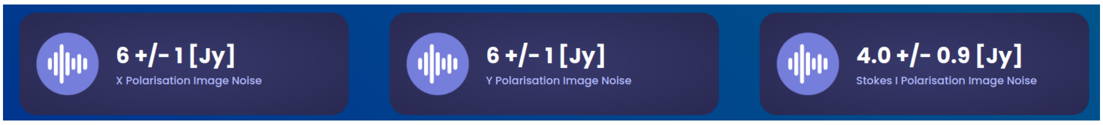


Imaging Sensitivity of the SKA-Low Telescope

This option enables the calculation of the sensitivity of the SKA-Low telescope (default number of stations is 512) in the specified pointing direction (right ascension and declination), during the time range of the observations (hour angle range), and for specified observing bandwidth and integration period (default of 120 seconds). The calculated result is the standard deviation (imaging sensitivity) of the mean image resulting from averaging multiple snapshot images of the specified integration time. The option also shows how the sensitivity (A/T) changes as a function of time during the specified observing interval.

Source Name	
Coordinate Type	RA / DEC
Right Ascension (0° - 360°)	256.59792360
Declination (-90° - 63°)	-26.51281500
Hour Angle Start (-12h - 12h)	-0.000138889
Hour Angle End (-12h - 12h)	0.000138889
Center Frequency (MHz)	230
Observing Bandwidth (MHz)	0.94
Snapshot image integration time (sec)	1
Number of Stations	1
Station Type	EDA2
Plot A/T [m^2/K] vs	Hour Angle
Output Format	Show Plot

(a) The imaging sensitivity calculator for the SKA-Low telescope, with the specified options provided. Here, the RA and DEC are approximate coordinates of the zenith in the middle time of the June (2023-06-01) observations. The centre frequency of observation is 230 MHz, the observing bandwidth is 0.94 MHz, and the image integration time is given as 1s. The Hour Angle range (±0.5s) is also consistent with the integration time.



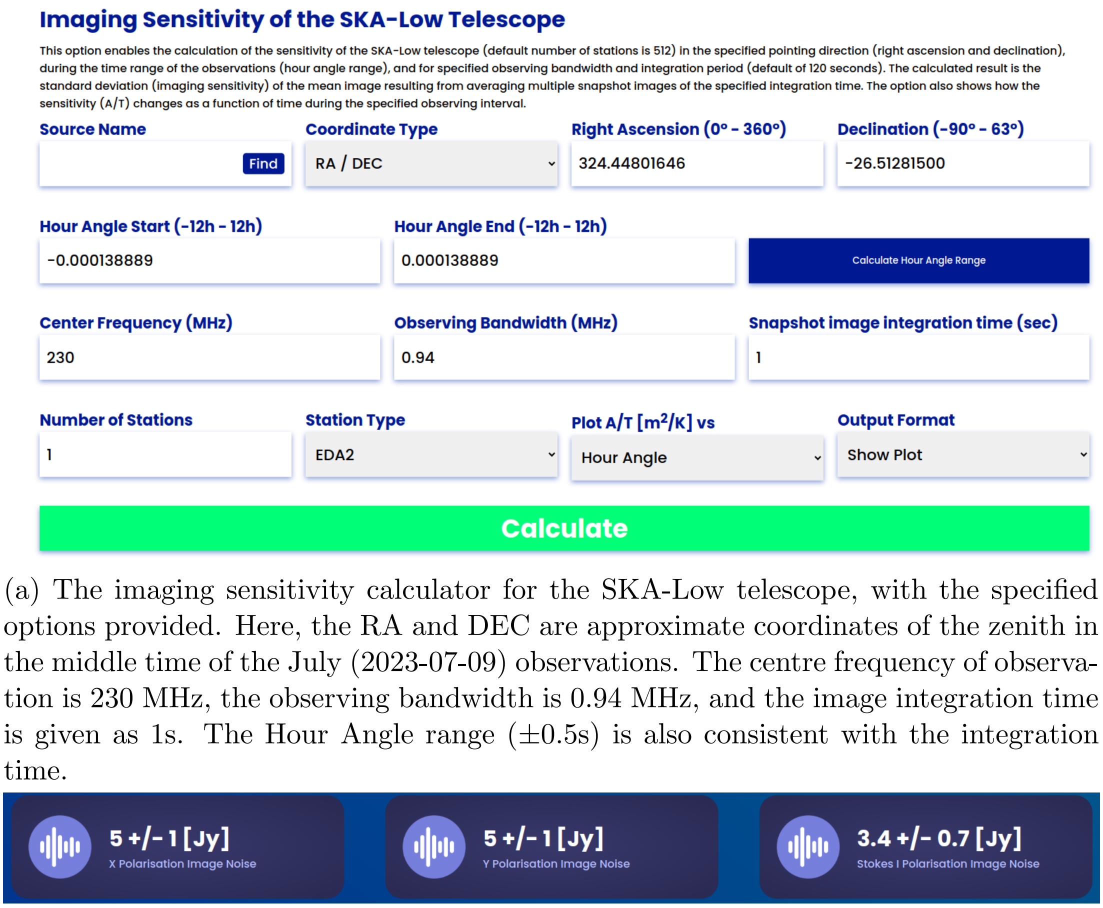
6 +/- 1 [Jy]	X Polarisation Image Noise
6 +/- 1 [Jy]	Y Polarisation Image Noise
4.0 +/- 0.9 [Jy]	Stokes I Polarisation Image Noise

(b) The calculated image sensitivity of the mean image, resulting from averaging multiple snapshot images of the specified integration time (i.e. here, 1s) at an observing bandwidth of 0.94 MHz. In this case, X Polarisation, Y Polarisation and Stokes I Polarisation Image Noise have been calculated for the parameters specified in Figure 4.16a.

Figure 4.16: The imaging sensitivity calculator for the SKA-Low telescope, with the specified options provided (top panel) and the calculated image sensitivity of the mean image (bottom panel).

In the imager used for processing this data, we have considered only the X polarisation, and hence, only the X Polarisation Image Noise has been considered, which is approximately 6 Jy and 5 Jy, for the June and July images, respectively, for the entire observing bandwidth of 0.94 MHz and a snapshot integration time of 1s.

In radio astronomy, the sensitivity is dependent on the system temperature (T_{system}), integration time (t_{int}) and the observing bandwidth ($\Delta\nu$) through



Imaging Sensitivity of the SKA-Low Telescope

This option enables the calculation of the sensitivity of the SKA-Low telescope (default number of stations is 512) in the specified pointing direction (right ascension and declination), during the time range of the observations (hour angle range), and for specified observing bandwidth and integration period (default of 120 seconds). The calculated result is the standard deviation (imaging sensitivity) of the mean image resulting from averaging multiple snapshot images of the specified integration time. The option also shows how the sensitivity (A/T) changes as a function of time during the specified observing interval.

Source Name:
Coordinate Type: RA / DEC
Right Ascension (0° - 360°): 324.44801646
Declination (-90° - 63°): -26.51281500

Hour Angle Start (-12h - 12h): -0.000138889
Hour Angle End (-12h - 12h): 0.000138889

Center Frequency (MHz): 230
Observing Bandwidth (MHz): 0.94
Snapshot image integration time (sec): 1

Number of Stations: 1
Station Type: EDA2
Plot A/T [m^2/K] vs: Hour Angle
Output Format: Show Plot

(a) The imaging sensitivity calculator for the SKA-Low telescope, with the specified options provided. Here, the RA and DEC are approximate coordinates of the zenith in the middle time of the July (2023-07-09) observations. The centre frequency of observation is 230 MHz, the observing bandwidth is 0.94 MHz, and the image integration time is given as 1s. The Hour Angle range (±0.5s) is also consistent with the integration time.

5 +/- 1 [Jy] X Polarisation Image Noise

5 +/- 1 [Jy] Y Polarisation Image Noise

3.4 +/- 0.7 [Jy] Stokes I Polarisation Image Noise

(b) The figure shows the calculated image sensitivity of the mean image, resulting from averaging multiple snapshot images of the specified integration time (i.e. here, 1s) at an observing bandwidth of 0.94 MHz. In this case, X Polarisation, Y Polarisation and Stokes I Polarisation Image Noise have been calculated for the parameters specified in Figure 4.17a.

Figure 4.17: The imaging sensitivity calculator for the SKA-Low telescope, with the specified options provided (top panel) and the calculated image sensitivity of the mean image (bottom panel).

the radiometer equation, given by:

$$\sigma \approx \frac{T_{system}}{\sqrt{t_{int}\Delta\nu}} \tag{4.6}$$

Thus, in this case, for a given telescope i.e. EDA2, the sensitivity for 100 ms images over the 32 fine channels (σ_2) can be calculated as:

$$\sigma_2 = \sigma_1 \times \sqrt{\frac{t_1 \Delta\nu_1}{t_2 \Delta\nu_2}}, \tag{4.7}$$

Substituting the following values for $\sigma_1 = 5$ Jy (for June images) and 6 Jy (for July images), $t_1 = 1$s, $t_2 = 100$ ms (i.e 0.1s), $\Delta\nu_1 = 0.94$ MHz and $\Delta\nu_2$ = 0.94/32 MHz, Equation 4.7, can be reduced to the following, to calculate the sensitivity for June images:

$$\sigma_2 = 6 \times \sqrt{\frac{0.94}{0.1 \times (0.94/32)}} \tag{4.8}$$

$$= 6 \times \sqrt{\frac{1}{0.1 \times (1/32)}} \tag{4.9}$$

$$= 6 \times \sqrt{320} \tag{4.10}$$

$$\approx 107\,\mathrm{Jy} \tag{4.11}$$

Similarly, to calculate the sensitivity of the July images, Equation 4.7 can be reduced to the following:

$$\sigma_2 = 5 \times \sqrt{\frac{0.94}{0.1 \times (0.94/32)}} \tag{4.12}$$

$$= 5 \times \sqrt{\frac{1}{0.1 \times (1/32)}} \tag{4.13}$$

$$= 5 \times \sqrt{320} \tag{4.14}$$

$$\approx 89\,\mathrm{Jy} \tag{4.15}$$

Thus, to get detected in a 10σ detection, a pulsar should have a mean flux density[8] of at least 890 Jy. All the pulsar candidates we got had a few mJy

[8] The images are not beam-corrected and are not flux-calibrated, and hence the flux densities

flux densities at GHz frequencies. The next step was to extrapolate the values of their flux densities at 230 MHz. In astronomy, the flux density of a source typically has a power law dependence on frequency, given by:

$$S_\nu = \nu^\alpha, \tag{4.16}$$

where ν is frequency in Hz, S_ν is measured in Jy and α is the spectral index (Thompson et al., 1991). Using a mean spectral index value of -1.6 (Jankowski et al., 2018), the extrapolated flux densities were still of the order of mJy. This check essentially eliminated the possibility of any of these candidates being pulsars.

Table 4.6 and 4.7 summarise the pulsar check done for all the candidates.

Table 4.6: Pulsar check for the short-listed candidates from the June data set.

Candi- -date	**RA**	**DEC**	**Closest Pulsar**	**Distance (Degrees)**	**DM (pc cm^{-3})**	**Estimated Flux-Density[1] at 230 MHz (mJy)**
1A	14:24:09	-38:00:46	J1418-3921	1.7	60.49	316
1B	15:00:43	-18:14:40	-	-	-	-
6A	13:40:49	-16:26:25	-	-	-	-
6B	14:29:26	-32:08:32	-	-	-	-
6C	14:51:28	-14:05:20	-	-	-	-
6E	15:47:32	-11:13:21	J1547-0944	1.49	37.416	-
7A	13:46:50	7:38:19.29	J1334+10	3.91	24	-

1. These are only estimated flux-density values (not measured) at 230 MHz.

compared here are uncalibrated.

Table 4.7: Pulsar check for the short-listed candidates from the July data set.

Candidate	RA	DEC	Closest Pulsar	Distance (Degrees)	DM (pc cm^{-3})	Estimated Flux-Density[1] at 230 MHz (mJy)
2A	14:03:06	-59:00:44	J1406-5806	0.981	229	15
4A	14:03:20	-44.58:10.12	J1405-4656	1.89	79.7	5
5A	13:06:36	-33:45:26	J1302-3258	1.17	26.186	8
7A	13:06:28	30:35:51.38	J1303+38	2.78	19	-
7B	13:06:28	30:35:51.38	J1303+38	2.78	19	-

1. These are only estimated flux-density values (not measured) at 230 MHz.

4.4.2 Satellite check

The above-the-horizon candidates' RA and DEC were checked to determine if they matched the positions of known satellites during the observations. Satellites that could be present within a 3° radius were considered. The spatial resolution of EDA2 is around 2.6° at 230 MHz. The angular distance travelled by the satellite during the 100ms depends on the satellite's altitude and the satellite's zenithal angle, as seen by the EDA2. A satellite at $\approx$ 400 km altitude, like the International Space Station (ISS), travels at about 1 degree every second near the zenith. Hence, such a satellite would have travelled about 0.1 degrees during our 100ms image. The search radius depends on the uncertainties of the satellite's catalogued orbital elements. These catalogues tend to have about 10 km in-orbit positional uncertainty. This translates to an angular error of approximately 1.43 degrees for a satellite at 400 km altitude. Hence, the 3-degree search radius was designed to account for this in-orbit uncertainty and not the satellite's angular distance travelled within the image integration time. For the satellite candidates, the following checks were performed:

- **Space debris check:** 'Space debris', called 'space junk', are non-functional and non-operational human-made objects in space. When satellites are operational, they are designed to communicate with the ground control station during their active mission phase. However, when the satellites and rockets reach the end of their mission, they can become non-operational and hence decommissioned. Thus, their communication systems can be turned off. The satellites can sometimes get dysfunctional due to a lack of power sources to sustain communication. Thus, space debris and rocket bodies are less likely to transmit signals. Hence, all satellites that were labelled as either debris or rocket bodies were eliminated to focus on the remaining satellite candidates. This check is summarised in Tables 4.8 and 4.9.

Table 4.8: Space debris check for the short-listed candidates from the June data.

Candidate Number	**No. of all satellites within radius = 3° (S)**	**No. of space debris within radius = 3° (D)**	**No. of satellites within radius = 3° (S - D)[1]**
1A	3	3	0
1B	28	8	20
6A	3	2	1
6B	0	0	0
6C	0	0	0
6E	2	0	1
7A	11	4	7

[1] Remaining number of potential satellite candidates for inspection after excluding those marked as ‘space debris’.

Table 4.9: Space debris check for the short-listed candidates from the July data.

Candidate Number	**No. and names of all satellites within radius = 3° (S)**	**No. of space debris within radius = 3° (D)**	**No. of satellites within radius = 3° (S - D)[1]**
2A	2	2	0
4A	21	10	11
5A	3	2	1
7A, 7B	4	3	1

[1] Remaining number of potential satellite candidates for inspection after excluding those marked as ‘space debris’.

- **Activity check:** The suspected satellite candidates were checked if they were active satellites using the website: http://n2yo.com/. All satellites, except one, were active. This check is summarised in Tables 4.10, 4.12 and 4.13 for the June data and in Tables 4.11 and 4.14 for the July data, respectively.

Table 4.10: Activity check and downlink frequency check for the satellites within a 3° radius, for the candidates from the June data.

Candidate Number	**No. of all satellites within radius = 3°**	**NORAD ID**	**Active?**	**Downlink Frequencies (MHz)**
1B	20[a]	-	-	-
6A	RESOURCESAT-2	37387	Yes	2250
6E	BEIDOU_3M24	44543	Yes	2483.5 - 2500
7A	11[b]	-	-	-

[a] See Table 4.12 for name, NORAD ID, activity and downlink frequencies.
[b] See Table 4.13 for name, NORAD ID, activity and downlink frequencies.

Table 4.11: Activity check and downlink frequency check for the satellites within a 3° radius, for the candidates from the July data.

Candidate Number	**No. of all satellites within radius = 3°**	**NORAD ID**	**Active?**	**Downlink Frequencies (MHz)**
2A	0	N/A	N/A	N/A
4A	11[a]	-	-	-
5A	COSMOS_2404_(GLONASS)	28112	Yes	1200 - 1600
7A, 7B	FORMOSAT_3D	29050	Yes	400.900

[a] See Table 4.14 for NORAD ID, activity and downlink frequencies.

- **Downlink frequency check:** The downlink frequencies of the suspected satellites were checked to see if they were around 230 MHz using the following website: https://db.satnogs.org/transmitters. This check is summarised in Tables 4.10, 4.12 and 4.13 for the June data and in Tables 4.11 and 4.14 for the July data, respectively. There were no satellites that typically transmitted signals in this downlink frequency range. Thus, these candidates could be unintended transmissions from these short-listed satellites. While it is possible to have reflections of radar or DTV from non-functional satellites, it is unlikely that they caused the observed phenomena. Airport radars operate at much higher frequencies (GHz) and are unlikely to cause reflections in our data. Additionally, it is unlikely to be DTV reflections, as even more sensitive instruments like the MWA have not seen DTV reflections from satellites other than the ISS and the Hubble Space Telescope (personal communication with the Curtin University's MWA Space Situational Awareness team).

Table 4.12: Activity check and downlink frequency check for the satellites within a 3° radius for Candidate 1B (June data).

Satellite Name	**NORAD ID**	**Active?**	**Downlink Frequencies**
FLOCK_4X_27	51012	Yes	8025 - 8400 MHz
FLOCK_4X_16	51020	Yes	8025 - 8400 MHz
FLOCK_4X_32	51024	Yes	8025 - 8400 MHz
STORK-2	51088	Yes	401.1 MHz
VCUB1	56215	Yes	401.550 MHz
HADES	51080	Yes	436.888 MHz
KEPLER-7_(C3PO)	47949	Yes	400.799 MHz
DMSAT-1	47940	Yes	-
JILIN-01_GAOFEN_3D_27	52444	Yes	1615.698 MHz
STARLINK-5454	54774	Yes	1615.698 MHz
STARLINK-3113	49735	Yes	10.7 to 12.7 GHz
STARLINK-3659	52001	Yes	10.7 to 12.7 GHz
STARLINK-3252	49733	Yes	10.7 to 12.7 GHz
STARLINK-5424	54811	Yes	10.7 to 12.7 GHz
STARLINK-1740	46566	Yes	10.7 to 12.7 GHz
STARLINK-2480	48137	Yes	10.7 to 12.7 GHz
ONEWEB-0567	55147	Yes	10.7 to 12.7 GHz
ONEWEB-0710	55173	Yes	10.7 to 12.7 GHz
ONEWEB-0313	49099	Yes	10.7 to 12.7 GHz
COSMOS_2468	37154	Yes	1200 to 1600 MHz

Table 4.13: Activity check and downlink frequency check for the satellites within a 3° radius for Candidate 7A (June data).

Satellite Name	**NORAD ID**	**Active?**	**Downlink frequencies**
SPACEBEE-159	55101	Yes	437 MHz
FLOCK_4Y_12	55065	Yes	8025 - 8400 MHz
STARLINK-4620	53674	Yes	10.7 to 12.7 GHz
STARLINK-4417	53200	Yes	10.7 to 12.7 GHz
STARLINK-1433	45711	Yes	10.7 to 12.7 GHz
BEIDOU_3M8	43246	Yes	2483.5 to 2500 MHz
CARTOSAT-2A	32783	Yes	8125 MHz

Table 4.14: Activity check and downlink frequency check for the satellites within a 3° radius for Candidate 4A (July data).

Satellite Name	**NORAD ID**	**Active?**	**Downlink Frequencies**
ONEWEB-0634	56075	Yes	10.7 to 12.7 GHz
STARLINK-5416	54855	Yes	10.7 to 12.7 GHz
STARLINK-4769	53843	Yes	10.7 to 12.7 GHz
STARLINK-3585	51875	Yes	10.7 to 12.7 GHz
ONEWEB-0212	48791	Yes	10.7 to 12.7 GHz
STARLINK-1981	47579	Yes	10.7 to 12.7 GHz
STARLINK-1689	46372	Yes	10.7 to 12.7 GHz
STARLINK-1586	46137	Yes	10.7 to 12.7 GHz
STARLINK-1639	46135	Yes	10.7 to 12.7 GHz
ZHUHAI-1_01	42761	No	145.88-145.86 MHz
BRITE-PL_2	40119	Yes	2234.4 MHz

4.5 Summary

In summary, a pilot search for FRBs was conducted using a few hours of EDA2 data in a small observing bandwidth (≈ 0.94 MHz) and images in a time resolution of 100 ms, starting from visibilities recorded in June and July 2023. An earlier developed software used in Anderson et al. (2021) was used to create dynamic spectra for every direction in the sky and then de-disperse in DM range 0 to 900 pc cm^{-3} with DM steps of 150 pc cm^{-3}. A search was performed to iterate through the de-dispersed series files to look for candidate events that had S/N > 5. The steps that were followed to check for FRBs from these candidates are summarised below:

1. Since it was impossible to go through over 3000+ candidates, a higher threshold of S/N > 10 was applied to limit the number of candidates for inspection. This reduced the number of candidates to 272 and 1829 from the June and July data, respectively.

2. Some time-steps had a lot of candidates (even up to 300). These time steps could have been affected by RFI, and hence, time steps with more than 20 candidates were discarded. The images with many RFI candidates were most likely caused by side-lobes from a single strong RFI source either on the horizon or above. This further reduced the number of candidates to 12 and 8 from the June and July data, respectively, for inspection.

3. The next step was to inspect the images corresponding to these candidates to verify if any of them were coming from near the horizon. There were 5 and 3 candidates from the June and July data, respectively which were coming from near the horizon. After calculating their Azimuth and Elevation, it was seen that these radio signals could be coming from towns near MRO, in the directions of Cue and Leonora Airport. This reduced the candidates to 8 and 5 in the June and July data, respectively.

4. All the remaining candidates had their DMs within 150 pc cm^{-3}, which eliminated the possibility of any of these candidates being FRBs since the DMs of all the currently known FRBs predominantly lie in the range of 200 - 1700 pc cm^{-3}. Hence, a further check was done to see if these candidates were due to any known pulsars or satellites within a 3° radius.

5. Two checks were performed to check if any of these candidates were due to pulsars. First was the DM-limit check. Since only vertical sweeps were observed in the dynamic spectra of all the short-listed candidates, suggesting that the maximum dispersive delay was less than 100 ms, the maximum DM (i.e. DM-limit) any of these candidates could have (i.e. 150 pc cm^{-3}) was calculated. This DM-limit check eliminated all pulsar candidates above this DM limit. Second was the flux-density check. All the pulsar candidates we got had a few mJy flux densities at GHz frequencies. Using a mean spectral index value of -1.6, the flux densities at 230 MHz were extrapolated. These flux densities at 230 MHz were still of the order of a few mJy. The flux densities (of the order of mJy) of the pulsars that could be associated with these candidates were much lower than flux densities required by 10σ detections (of the order of tens of Jy). This essentially eliminated the possibility of any of these candidates being pulsars.

6. To check if any of these candidates were due to any satellites, all potential satellites within the 3 ° radius that were space debris were eliminated. Next, all inactive satellites were also discarded. Furthermore, none of the remaining potential active satellites had downlink frequencies in the range of 230 MHz. In summary, none of the candidates were likely due to FRBs and pulsars. Hence, these radio signals could be unintended transmissions from satellites.

Chapter 5

Summary and Future Work

In summary, this work has successfully demonstrated the development of two test versions of the all-sky GPU imager (with the CPU imager as a starting point) using NVIDIA GPU hardware and `CUDA` programming environment. This work has also demonstrated that the single time-step multi-channel GPU imager (with FFT implementation using `cufftPlanMany`) performed faster (≈ thousands of times faster) than the baseline single time-step multi-channel CPU imager (FFT implementation using `fftw_plan_many_dft()`). Finally, the initial all-sky CPU imager was applied offline to a few hours of EDA2 data in 100 ms time resolution to form all-sky images and search for FRBs in these images. In this chapter, the two projects that are presented in this work are summarised, and the main achievements, take-away results and lessons of the work presented in this thesis have been reviewed.

5.0.1 Project 1: High-Time Resolution 'BLINK' GPU imager for FRB searches

In the first project, two standalone test versions of the GPU imager (with my primary supervisor, Dr. Marcin Sokolowski's CPU imager as the starting point) that generated all-sky images from EDA2 data were developed using NVIDIA GPU hardware and the CUDA programming environment. In the single time-step, single-channel test version of the GPU imager, gridding was implemented using a 'GPU gridding kernel', and Fourier transform was performed using the `cuFFT` library (`cufftPlan2d` and `cufftExecC2C`). In the single-time-step, multi-channel test version of the GPU imager, gridding was done parallelly across all the channels, either through CUDA streams or a 2D grid of CUDA blocks, and `cuFFTPlanMany` was used to configure batched FFTs on multiple gridded visibilities corresponding to different frequency channels of the same time-step. The GPU imager was also compiled, tested and benchmarked[1] on the Topaz and Setonix supercomputers at the Pawsey Supercomputing Centre in Perth, Western Australia. The main achievements and take-away results of this project were:

1. The speed-ups in the single time-step, single-channel GPU version were noticed (≈ 3 times faster than the baseline CPU version) only while processing a single large image of size 4096 × 4096 (see Table 3.2). The speed-ups were not observed while processing smaller images of size: 180 × 180 and 1024 × 1024.

2. The improvements obtained due to GPU processing become even more evident while testing for large amounts of data. The GPU gridding kernel (see Tables 3.6, 3.7) and the Fourier transforms (see Table 3.5) performed extremely fast on both Topaz and Setonix.

3. While performing parallel gridding using streams (CUDA streams on Topaz, HIP streams on Setonix), it took $\approx 9.2\,\mu$s and $16.9\,\mu$s on average to grid a single image on Topaz and Setonix, respectively.

4. Similarly, while performing parallel gridding using a 2D grid of blocks (CUDA blocks on Topaz, HIP blocks on Setonix), it took around $\approx 8.2\,\mu$s

[1] The GPU imager that was tested on Setonix, was implemented using the HIP programming environment, and this HIP version of the code was written by my primary supervisor. After the HIP version was developed, I was involved in benchmarking both the CUDA and HIP versions of the GPU imager on Topaz and Setonix, respectively.

and 10.4 μs on average to grid a single image on Topaz and Setonix, respectively.

5. Likewise, performing the operation of Fourier transform per image utilising GPUs was seen to be extremely quick, and it took less than 1 ms on both Topaz and Setonix to perform Fourier transform to generate a single image.

6. While performing batched FFTs to generate 500 images in one go, it took around $\approx$ 0.74 ms and 0.48 ms on average to generate a single image on Topaz and Setonix, respectively.

7. Finally, the performance of the single time-step, multi-channel GPU version was also compared with that of the baseline single time-step, multi-channel CPU imager (see Figures 3.15 and 3.16), and it was seen that the GPU version performed up to thousands of times faster than the baseline CPU imager.

Nevertheless, there are still some areas for potential improvements. The images generated by the starting CPU imager were used as reference images, and the GPU version of the imager was validated by using these CPU reference images. Although the images generated by the GPU version aligned closely with the images from the CPU version, there were small discrepancies of the order of 0.001 (i.e. $\leq$ 0.38%) in the CPU and GPU images. These differences are still being investigated, and the current understanding is that they result from the small fractional differences observed in the gridded visibilities of the order of 0.25 (i.e. $\approx$ 0.00001%). This difference was because GPUs were used for parallel gridding, resulting in a random order of additions in each execution and the inherent limitations of floating-point arithmetic in numerical computations. This is a small limitation in the current test version of the GPU imager, and alternate methods like parallel reduction are being considered in future implementations to achieve more reproducible results.

5.0.2 Project 2: Pilot FRB searches on EDA2 data

In the second project, the initial all-sky test starting test version of the CPU imager described in Chapter 3 was used to image a few hours of EDA2 data to perform a pilot search for FRBs. Visibilities were recorded in June and July 2023, and earlier-developed software was used to create dynamic spectra for every direction in the sky and then de-disperse in DM range 0 to 900 pc cm^{-3} with DM steps of 150 pc cm^{-3}. A search was performed to iterate through the de-dispersed series files to look for candidate events that had S/N $>$ 5. In the June data, there were 3,726 candidates with S/N $>$ 5, spread across 30 time steps. In the July data, there were 11920 candidates with S/N $>$ 5, spread across 70 time steps. Since it was impossible to go through over 3000+ candidates, a threshold of S/N $>$ 10 was applied to limit the number of candidates for inspection. This brought the number of candidates down to 272 and 1829 from the June and July data, respectively. After discarding candidates due to strong RFI (with multiple candidates in a single image) or the candidates from near the horizon, the remaining candidates were inspected to see if they could be due to FRBs or other astrophysical sources. The main achievements and take-away results of this project were:

1. All the final short-listed candidates had low DMs within 150 pc cm^{-3}, which eliminated the possibility of any of these candidates being FRBs, as the DMs of all the currently known FRBs predominantly lie in the range of 200 to 1700 pc cm^{-3}.

2. Furthermore, a check was done for each potential candidate to see if these candidates were due to any known pulsars or satellites within a 3° search radius.

3. None of the candidates were due to any known pulsars, as the flux densities (of the order of mJy) of the pulsars that could be associated with these candidates were much lower than flux densities required by 10σ detections

(of the order of tens of Jy).

4. Also, none of the potential active satellites within a 3-degree radius had their downlink frequencies in the range of 230 MHz.

5. In summary, none of the potential short-listed candidates were due to FRBs, pulsars or transmissions from satellites, but these radio signals could be unintended transmissions from satellites.

6. While it is possible to have reflections of radar or DTV from non-functional satellites, it is improbable that they caused the observed phenomena. Airport radars operate at higher frequencies (GHz), making them unlikely to cause reflections in our data. Similarly, DTV reflections are also unlikely, as even sensitive instruments like the MWA have only detected such reflections from the ISS and the Hubble Space Telescope (per communication with Curtin University's MWA Space Situational Awareness team).

5.0.3 Closing Remarks and Future Work

The task of making the GPU imager general (Sokolowski et al., 2024a) was completed by my primary supervisor and a PhD student after the submission of this thesis. This code for this GPU imager is publicly available[2] and can be applied to data from any radio telescope. This all-sky GPU imager was also applied on a few hours of EDA2 data in 100 ms time resolution to form all-sky images and to search for dispersed pulses, including FRBs in these images. The results of this pilot search will be presented in an upcoming publication (Sokolowski et al., in preparation). This imaging and FRB-search pipeline for EDA2 is also currently under development.

Future work involves incorporating the production version of this GPU imager (Sokolowski et al., 2024a) into the full GPU-based processing pipeline and applying it to a large volume of data from SKA-Low stations and the MWA to search

[2] https://github.com/PaCER-BLINK-Project/imager

for FRBs and other transient phenomena. The main aim of this pipeline will be to search for these events in high-time resolution all-sky images from SKA-Low stations (Sokolowski et al., 2022a) and wide-field images from the MWA data. This complete GPU-based FRB-search pipeline is currently under development and will be described in upcoming publications (Di Pietrantonio et al., in preparation). This pipeline will perform correlation and imaging, preferably in real-time, leading to real-time FRB searches, and as estimated in Sokolowski et al. (2024b), this pipeline applied to SKA-Low stations data can yield even up to hundreds of FRB detections per year.

5.1 Acknowledgements

The work presented in this thesis is a Pawsey Centre for Extreme Scale Readiness (PaCER) project jointly funded by the Pawsey Supercomputing Centre, the International Centre for Radio Astronomy Research (ICRAR) and the Australian SKA Regional Centre (AusSRC). This work was supported by resources provided by the Pawsey Supercomputing Research Centre's Setonix Supercomputer[3], with funding from the Australian Government and the Government of Western Australia This scientific work uses data obtained from Inyarrimanha Ilgari Bundara, the CSIRO Murchison Radio-astronomy Observatory. We acknowledge the Wajarri Yamaji People as the Traditional Owners and native title holders of the Observatory site. Support for the operation of the MWA is provided by the Australian Government (NCRIS) under a contract to Curtin University administered by Astronomy Australia Limited. EDA2 is hosted by the MWA under an agreement via the MWA External Instruments Policy.

[3] https://doi.org/10.48569/18sb-8s43

Bibliography

G. E. Anderson, et al. (2021). 'Murchison Widefield Array rapid-response observations of the short GRB 180805A'. Publications of the Astronomical Society of Australia **38**:e026.

N. D. R. Bhat, et al. (2016). 'Scintillation Arcs in Low-frequency Observations of the Timing-array Millisecond Pulsar PSR J0437-4715'. The Astrophysical Journal **818**(1):86.

P. Bolli, et al. (2021). 'Status of the High-Frequency Upgrade of the Sardinia Radio Telescope'. URSI Radio Science Letters **3**:26.

M. Born & E. Wolf (1959). Principles of Optics. Pergamon Press Ltd., London, 1st edn. [many subsequent editions, e.g., 7th ed., Cambridge Univ. Press, Cambridge, UK, 1999].

CASA Team, et al. (2022). 'CASA, the Common Astronomy Software Applications for Radio Astronomy'. **134**(1041):114501.

S. Chatterjee, et al. (2017). 'A direct localization of a fast radio burst and its host'. Nature **541**(7635):58–61.

P. Chawla, et al. (2020). 'Detection of Repeating FRB 180916.J0158+65 Down to Frequencies of 300 MHz'. The Astrophysical Journal Letters **896**(2):L41.

CHIME/FRB Collaboration, et al. (2021). 'The First CHIME/FRB Fast Radio Burst Catalog'. The Astrophysical Journal Supplement Series **257**(2):59.

CHIME/FRB Collaboration, et al. (2020). 'Periodic activity from a fast radio burst source'. Nature **582**(7812):351–355.

CHIME/FRB Collaboration, et al. (2018). 'The CHIME Fast Radio Burst Project: System Overview'. The Astrophysical Journal **863**(1):48.

CHIME/FRB Collaboration, et al. (2019). 'CHIME/FRB Discovery of Eight New Repeating Fast Radio Burst Sources'. The Astrophysical Journal Letters **885**(1):L24.

J. W. Cooley & J. W. Tukey (1965). 'An algorithm for the machine calculation of complex Fourier series'. Mathematics of computation **19**(90):297–301.

J. M. Cordes & T. J. W. Lazio (2002). 'NE2001.I. A New Model for the Galactic Distribution of Free Electrons and its Fluctuations'. arXiv e-prints pp. astro–ph/0207156.

T. Cornwell & R. Perley (1992). 'Radio-interferometric imaging of very large fields-The problem of non-coplanar arrays'. Astronomy and Astrophysics (ISSN 0004-6361), vol. 261, no. 1, p. 353-364. **261**:353–364.

T. Cornwell, et al. (2012). 'Image Reconstruction from Incomplete Data VII'. In Proc. SPIE, vol. 8500, p. 85000.

M. Cruces, et al. (2021). 'Repeating behaviour of FRB 121102: Periodicity, waiting times, and energy distribution'. Monthly Notices of the Royal Astronomical Society **500**(1):448–463.

D. R. DeBoer, et al. (2009). 'Australian SKA Pathfinder: A High-Dynamic Range Wide-Field of View Survey Telescope'. IEEE Proceedings **97**(8):1507–1521.

S. W. Ellingson, et al. (2009). 'The long wavelength array'. Proceedings of the IEEE **97**(8):1421–1430.

V. Gajjar, et al. (2018). ‘Highest Frequency Detection of FRB 121102 at 4-8 GHz Using the Breakthrough Listen Digital Backend at the Green Bank Telescope’. The Astrophysical Journal **863**(1):2.

J. A. Högbom (1974). ‘Aperture Synthesis with a Non-Regular Distribution of Interferometer Baselines’. Astronomy and Astrophysics Supplement Series **15**:417.

B. Humphreys & T. Cornwell (2011). ‘Ska memo 132’. Square Kilometre Array Memo **132**.

S. Inoue (2004). ‘Probing the cosmic reionization history and local environment of gamma-ray bursts through radio dispersion’. Monthly Notices of the Royal Astronomical Society **348**(3):999–1008.

K. Ioka (2003). ‘The Cosmic Dispersion Measure from Gamma-Ray Burst Afterglows: Probing the Reionization History and the Burst Environment’. The Astrophysical Journal Letters **598**(2):L79–L82.

F. Jankowski, et al. (2018). ‘Spectral properties of 441 radio pulsars’. Monthly Notices of the Royal Astronomical Society **473**(4):4436–4458.

P. R. Jewell (2000). ‘The Green Bank Telescope’. In H. R. Butcher (ed.), Radio Telescopes, vol. 4015 of Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, pp. 136–147.

W. M. Lane, et al. (2005). ‘The VLA Low-frequency Sky Survey (VLSS)’. In American Astronomical Society Meeting Abstracts, vol. 207 of American Astronomical Society Meeting Abstracts, p. 25.10.

L.-B. Li, et al. (2018). ‘Progress in the Researches of Fast Radio Bursts’. Chinese Astronomy and Astrophysics **42**(2):165–187.

D. R. Lorimer, et al. (2007). ‘A Bright Millisecond Radio Burst of Extragalactic Origin’. Science **318**(5851):777.

D. R. Lorimer & M. Kramer (2012). Handbook of Pulsar Astronomy.

J.-P. Macquart, et al. (2010). 'The commensal real-time ASKAP fast-transients (CRAFT) survey'. Publications of the Astronomical Society of Australia **27**(3):272–282.

J. P. Macquart, et al. (2020). 'A census of baryons in the Universe from localized fast radio bursts'. Nature **581**(7809):391–395.

J. P. Macquart, et al. (2019). 'The Spectral Properties of the Bright Fast Radio Burst Population'. The Astrophysical Journal Letters **872**(2):L19.

A. Magro, et al. (2011). 'Real-time, fast radio transient searches with GPU de-dispersion'. Monthly Notices of the Royal Astronomical Society **417**(4):2642–2650.

J. M. Marr, et al. (2015). Fundamentals of radio astronomy: observational methods, vol. 13. CRC Press.

S. J. McSweeney, et al. (2017). 'Low-frequency Observations of the Subpulse Drifter PSR J0034-0721 with the Murchison Widefield Array'. The Astrophysical Journal **836**(2):224.

B. W. Meyers, et al. (2017). 'Spectral Flattening at Low Frequencies in Crab Giant Pulses'. The Astrophysical Journal **851**(1):20.

D. A. Mitchell, et al. (2008). 'Real-time calibration of the Murchison Widefield Array'. IEEE Journal of Selected Topics in Signal Processing **2**(5):707–717.

G. Naldi, et al. (2017). 'The digital signal processing platform for the low frequency aperture array: Preliminary results on the data acquisition unit'. Journal of Astronomical Instrumentation **6**(01):1641014.

A. R. Offringa, et al. (2014). 'WSCLEAN: an implementation of a fast, generic wide-field imager for radio astronomy'. Monthly Notices of the Royal Astronomical Society **444**(1):606–619.

A. R. Offringa, et al. (2015). 'The Low-Frequency Environment of the Murchison Widefield Array: Radio-Frequency Interference Analysis and Mitigation'. Publications of the Astronomical Society of Australia **32**:e008.

S. Ord, et al. (2019). 'MWA tied-array processing I: Calibration and beamformation'. Publications of the Astronomical Society of Australia **36**:e030.

S. M. Ord, et al. (2015). 'The Murchison Widefield Array Correlator'. Publications of the Astronomical Society of Australia **32**:e006.

E. Parent, et al. (2020). 'First Discovery of a Fast Radio Burst at 350 MHz by the GBNCC Survey'. The Astrophysical Journal **904**(2):92.

E. Petroff & S. Chatterjee (2023). 'FRB Newsletter Volume 04, Issue 03 — March 2023'.

E. Petroff, et al. (2022). 'Fast radio bursts at the dawn of the 2020s'. **30**(1):2.

M. Pilia, et al. (2020). 'The Lowest-frequency Fast Radio Bursts: Sardinia Radio Telescope Detection of the Periodic FRB 180916 at 328 MHz'. The Astrophysical Journal Letters **896**(2):L40.

Z. Pleunis, et al. (2021). 'LOFAR Detection of 110-188 MHz Emission and Frequency-dependent Activity from FRB 20180916B'. The Astrophysical Journal Letters **911**(1):L3.

S. R. Prabu (2021). Advantages and Limitations of using the Murchison Widefield Array for Space Surveillance: a Feasibility Study. Ph.D. thesis, Curtin University.

I. Prandoni, et al. (2017). 'The Sardinia Radio Telescope . From a technological project to a radio observatory'. Astronomy and Astrophysics **608**:A40.

K. M. Rajwade, et al. (2020). 'Possible periodic activity in the repeating FRB 121102'. Monthly Notices of the Royal Astronomical Society **495**(4):3551–3558.

S. Ransom (2011). ‘PRESTO: PulsaR Exploration and Search TOolkit’. Astrophysics Source Code Library, record ascl:1107.017.

S. M. Ransom (2001). New search techniques for binary pulsars. Ph.D. thesis, Harvard University, Massachusetts.

A. Rowlinson, et al. (2016). ‘Limits on Fast Radio Bursts and other transient sources at 182 MHz using the Murchison Widefield Array’. Monthly Notices of the Royal Astronomical Society **458**(4):3506–3522.

K. R. Sand, et al. (2020). ‘Low-frequency detection of FRB180916 with the uGMRT’. The Astronomer’s Telegram **13781**:1.

R. J. Sault, et al. (2011). ‘MIRIAD: Multi-channel Image Reconstruction, Image Analysis, and Display’. Astrophysics Source Code Library, record ascl:1106.007.

C. E. Shannon (1949). ‘Communication in the Presence of Noise’. IEEE Proceedings **37**:10–21.

R. M. Shannon, et al. (2018). ‘The dispersion-brightness relation for fast radio bursts from a wide-field survey’. Nature **562**(7727):386–390.

M. Sokolowski, et al. (2024a). ‘High-Time Resolution GPU Imager for FRB searches at low radio frequencies’. arXiv e-prints p. arXiv:2405.13478.

M. Sokolowski, et al. (2018). ‘No Low-frequency Emission from Extremely Bright Fast Radio Bursts’. The Astrophysical Journal Letters **867**(1):L12.

M. Sokolowski, et al. (2024b). ‘A commensal Fast Radio Burst search pipeline for the Murchison Widefield Array’. Publications of the Astronomical Society of Australia **41**:e011.

M. Sokolowski, et al. (2022a). ‘A High Time Resolution All-Sky Monitor for Fast Radio Bursts and Technosignatures’. In 2022 3rd URSI Atlantic and Asia Pacific Radio Science Meeting (AT-AP-RASC, p. 1.

M. Sokolowski, et al. (2022b). 'What is the SKA-Low sensitivity for your favourite radio source?'. Publications of the Astronomical Society of Australia **39**:e015.

N. A. Swainston, et al. (2022). 'MWA tied-array processing IV: A multi-pixel beamformer for pulsar surveys and ionospheric corrected localisation'. Publications of the Astronomical Society of Australia **39**:e020.

J. D. Swinbank, et al. (2015). 'The LOFAR Transients Pipeline'. Astronomy and Computing **11**:25–48.

G. Taylor, et al. (1999). 'Synthesis imaging in radio astronomy II: a collection of lectures from the Sixth NRAO/NMIMT Synthesis Imaging Summer School held at Socorro, New Mexico, USA, 17-23 June, 1998'. (No Title) .

A. Thompson, et al. (1991). Interferometry and Synthesis in Radio Astronomy, vol. -1.

D. Thornton, et al. (2013). 'A Population of Fast Radio Bursts at Cosmological Distances'. Science **341**(6141):53–56.

J. Tian, et al. (2023). 'A targeted search for repeating fast radio bursts with the MWA'. Monthly Notices of the Royal Astronomical Society **518**(3):4278–4289.

S. J. Tingay, et al. (2013). 'The Murchison Widefield Array: The Square Kilometre Array Precursor at Low Radio Frequencies'. Publications of the Astronomical Society of Australia **30**:e007.

S. J. Tingay, et al. (2015). 'A Search for Fast Radio Bursts at Low Frequencies with Murchison Widefield Array High Time Resolution Imaging'. The Astronomical Journal **150**(6):199.

S. E. Tremblay, et al. (2015). 'The High Time and Frequency Resolution Capabilities of the Murchison Widefield Array'. Publications of the Astronomical Society of Australia **32**:e005.

S. van der Tol, et al. (2018). 'Image Domain Gridding: a fast method for convolutional resampling of visibilities'. Astronomy and Astrophysics **616**:A27.

M. P. van Haarlem, et al. (2013). 'LOFAR: The LOw-Frequency ARray'. Astronomy and Astrophysics **556**:A2.

R. Wayth, et al. (2022). 'Engineering Development Array 2: design, performance, and lessons from an SKA-Low prototype station'. Journal of Astronomical Telescopes, Instruments, and Systems **8**:011010.

R. B. Wayth, et al. (2011). 'V-FASTR: The VLBA Fast Radio Transients Experiment'. The Astrophysical Journal **735**(2):97.

R. B. Wayth, et al. (2018). 'The Phase II Murchison Widefield Array: Design overview'. Publications of the Astronomical Society of Australia **35**:e033.

J. M. Yao, et al. (2017). 'A New Electron-density Model for Estimation of Pulsar and FRB Distances'. The Astrophysical Journal **835**(1):29.

B. Zackay & E. O. Ofek (2017). 'An Accurate and Efficient Algorithm for Detection of Radio Bursts with an Unknown Dispersion Measure, for Single-dish Telescopes and Interferometers'. The Astrophysical Journal **835**(1):11.