institut für astrophysik
UNIVERSITÄTSSTERNWARTE WIEN

universität wien

RADTEL

Control Software for Small Radio Telescopes



Control Software for Small Radio Telescopes

Reference: RADTEL-UVIE-SW-001

Version: Issue 1.1, December 20, 2021

Prepared by: Armin Luntzer[1]

Checked by: Verena Baumgartner[1]

Approved by: Franz Kerschbaum[1]

[1] Department of Astrophysics, University of Vienna

Copyright ©2022

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Front-Cover, no Logos of the University of Vienna.

Abstract

This document formally describes software designed for the operation of small (<10m) radio telescopes, which are intended for the purpose of education and teaching. The software is built upon a client-server, multi-user model which supports the operation of different hardware backends via a unified callback interface and network protocol design. The graphical user interface was created with ease of use in mind, offering basic analysis tools, instrument status information, and automated observation programmes. A configurable radio telescope simulator is included for additional flexibility in lectures, or in case an actual instrument is not available. Also provided is a complementary selection of projects and exercises, which convey basic concepts of radio astronomy and can be realised with the software.

Contents

Prologue 1

1 Introduction 3
- 1.1 The Radio Window . . . 4
  - 1.1.1 Ionospheric Reflection . . . 4
- 1.2 Astronomical Sources of Radio Signals . . . 5
  - 1.2.1 Black Body Radiation . . . 6
  - 1.2.2 Neutral Hydrogen . . . 7
- 1.3 Receiving Systems . . . 7
  - 1.3.1 Terminology . . . 8
  - 1.3.2 Receiver Noise . . . 9
  - 1.3.3 System Temperature . . . 10
  - 1.3.4 Image Rejection . . . 10
  - 1.3.5 Angular Resolution . . . 11
  - 1.3.6 Antenna Gain . . . 11
  - 1.3.7 Surface Error Losses . . . 11
  - 1.3.8 Illumination Losses . . . 11
  - 1.3.9 Antenna Beam Pattern . . . 12
  - 1.3.10 Antenna Performance . . . 13
  - 1.3.11 Aperture Efficiency . . . 14
- 1.4 Absolute Amplitude Calibration . . . 16
  - 1.4.1 Closed Two-Load Calibration . . . 16
  - 1.4.2 Semi-Open Single-Load Calibration . . . 16
  - 1.4.3 Open Calibration . . . 16
  - 1.4.4 Calibration Factor . . . 17
- 1.5 Observational and Measurement Techniques . . . 17

2 Control Software 19
- 2.1 Requirements . . . 20
  - 2.1.1 Overview . . . 20
  - 2.1.2 User Requirements . . . 21
  - 2.1.3 Software Requirements . . . 23
- 2.2 Software Design . . . 24
  - 2.2.1 Overview . . . 24
  - 2.2.2 Server Design . . . 26
  - 2.2.3 Client Design . . . 28
  - 2.2.4 Requirements to Design Components Traceability . . . 32
- 2.3 Service Commands . . . 33
  - 2.3.1 Execution profiles . . . 33
- 2.4 Command and Data Exchange Packet Format . . . 35
  - 2.4.1 Packet Structure Members . . . 36
- 2.5 Service Identifiers . . . 37
  - 2.5.1 Invalid Packet . . . 38
  - 2.5.2 Capabilities . . . 38
  - 2.5.3 Control . . . 39

2.5.4 Move to Azimuth/Elevation . . . 39
2.5.5 Command Succeeded . . . 40
2.5.6 Command Failed . . . 40
2.5.7 Recalibrate Pointing . . . 40
2.5.8 Park Telescope . . . 41
2.5.9 Spectrometer Acquisition Configuration . . . 41
2.5.10 Spectral Data . . . 42
2.5.11 Get Current Azimuth/Elevation . . . 43
2.5.12 Enable Spectrum Acquisition . . . 43
2.5.13 Disable Spectrum Acquisition . . . 44
2.5.14 Spectrometer Acquisition Configuration Request . . . 44
2.5.15 Spectrum Acquisition Status . . . 45
2.5.16 Drive Slew Status . . . 45
2.5.17 Drive Move Status . . . 46
2.5.18 Spectrometer Recording Status . . . 46
2.5.19 Lack Of Privilege . . . 47
2.5.20 Text Message . . . 47
2.5.21 User List . . . 48
2.5.22 User Nick Name . . . 48
2.5.23 Capabilities: Hot Load Extension . . . 49
2.5.24 Enable Hot Load . . . 50
2.5.25 Disable Hot Load . . . 50
2.5.26 Drive and Spectrometer Characterisation . . . 51
2.6 Backend Plugin Calls . . . 52
2.7 Implemented Backends: Haystack SRT (CASSI Kit) . . . 54
2.7.1 Overview . . . 54
2.7.2 Antenna and Mount . . . 54
2.7.3 Receiving System . . . 55
2.7.4 Calibration Device . . . 56
2.7.5 Hardware Interface . . . 57
2.7.6 Drive Unit Operation . . . 58
2.7.7 Spectrometer Unit Operation . . . 61
2.7.8 Implementation . . . 61
2.8 Implemented Backends: Simulator . . . 65
2.8.1 Motivation . . . 65
2.8.2 Characteristics . . . 65
2.8.3 Telescope pointing . . . 66
2.8.4 HI Data Set . . . 66
2.8.5 Hydrogen Sky . . . 68
2.8.6 Convolution Kernel . . . 68
2.8.7 Hydrogen Spectrum Generation . . . 70
2.8.8 Hydrogen Map Generation . . . 70
2.8.9 Moon . . . 71
2.8.10 Sun . . . 72
2.8.11 Cosmic Microwave Background . . . 72
2.8.12 System Temperature and Noise . . . 72
2.8.13 Calibration Devices . . . 72
2.9 User Interface . . . 73
2.9.1 Chat & Log . . . 73
2.9.2 Sky View . . . 74

2.9.3 Spectrum . . . 75
2.9.4 Telescope . . . 76
2.9.5 Spectrometer . . . 77
2.9.6 History . . . 78
2.9.7 Observation . . . 78
2.10 Fast Map Drawing . . . 80

3 Lab Projects and Exercises **83**
3.1 Antenna Beamwidth . . . 84
3.1.1 Procedure . . . 84
3.1.2 Data Evaluation . . . 85
3.2 Aperture Efficiency via Flux Comparison . . . 87
3.2.1 Procedure . . . 87
3.2.2 Data Evaluation . . . 88
3.3 Aperture Efficiency via Beam Dilution . . . 90
3.3.1 Procedure . . . 90
3.3.2 Data Evaluation . . . 90
3.4 Spillover . . . 92
3.4.1 Procedure . . . 92
3.4.2 Data Evaluation . . . 92
3.5 Galactic Plane Velocity Diagram . . . 93
3.5.1 Procedure . . . 93
3.5.2 Data Evaluation . . . 94
3.6 Galactic Rotation Curve . . . 95
3.6.1 Procedure . . . 96
3.6.2 Data Evaluation . . . 96
3.7 Spiral Arm Structure . . . 97
3.7.1 Procedure . . . 97
3.7.2 Data Evaluation . . . 97
3.8 Map-Making . . . 99
3.8.1 Procedure . . . 99
3.8.2 Data Evaluation . . . 100
3.9 Other Projects . . . 101
3.9.1 All-Sky Drift Scan Map . . . 101
3.9.2 Solar Radio Bursts . . . 101
3.9.3 Radio Environment Map . . . 101
3.9.4 2D PSF reconstruction . . . 102
3.9.5 Instrument Spectral Response . . . 102

4 Outlook **105**

Bibliography **106**

Appendix **107**
Acronyms . . . 108
Glossary . . . 108

Prologue

The software described hereafter is named *radtel*. It is published under the GNU General Public License, version 2. Use a search engine of your choice to find it on the Web. Links are not provided here, as they may change over time or become invalid. This text, or updated versions thereof, will be included as part of the documentation package.

It is important to note that this is *not* a user manual. For help on building, installing, or configuration, please refer to other documentation available in the source code repository.

This document is laid out for viewing in electronic form. If you view this in printed form, you will notice coloured elements within the text. These are hyperlinks to locations within the document.

1. Introduction

Unlike most other sciences, astronomy cannot study its objects of interest in place or in a laboratory. Except for the data and material collected from within our solar system by manned missions and unmanned space probes, most of the knowledge we have on the universe comes from observations of the electromagnetic radiation emitted by celestial objects.

For a very long time in the history of astronomy, observations were restricted to the visual spectrum. Other wavelength bands were hidden from mankind because of the unavailability of detectors, except for the human eye, and also the lack of knowledge of their existence. In 1800, the visual window expanded into the near infrared, when *William Herschel* measured the temperatures of different colours of sunlight split by a prism. Noticing a temperature increase from the blue to the red, he found that the heating effect was even larger in an apparently unlit area beyond the red colour. When *James Clerk Maxwell* integrated the phenomena of electricity and magnetism into a combined theory in the 1860s, he showed that his equations implied the propagation of light as an electromagnetic wave. It was *Heinrich Hertz* twenty years later, who found the first conclusive experimental proof of the existence of these electromagnetic waves beyond the visual spectrum.

In the late 1890s, largely based on inventions by *Nikola Tesla*, *Guglielmo Marconi* developed a first practical application in the form of a wireless telegraph, which was still a purely mechanical device, consisting of a switch-operated spark-gap transmitter for transmission and a *coherer* based detector. After the turn of the century, the first powerful, continuous-wave transmitters were constructed. The *arc converter* was an undamped variant of the spark transmitter and the *Alexanderson alternator* was a device similar to a high-frequency, alternating-current electrical generator. These devices allowed speech transmission for the first time by modulating the amplitude of their output with a *magnetic amplifier*. The first viable continuously operating radio detector was created by *G. W. Pickard*, who used the rectifying effect of semiconductor crystals, touched by thin wire, to demodulate a signal. In 1904, *J. A. Fleming* developed a vacuum tube rectifier, which was modified by *L. d. Forest* two years later into the *triode*, the first electronic amplifier. This marked the beginning of the electronics age. In the 1910s the *superheterodyne receiver* was invented, which was a milestone in radio technology. By converting the carrier wave, the resulting signal could be treated at a fixed, lower frequency in the output stage, allowing not only an expansion of the usable wavelength range, but also improved selectivity and sensitivity compared to simpler, directly-rectifying designs. It was not until the early 1930s, that this still new technology opened a new and wide window of the electromagnetic spectrum for astronomers, when radio engineer *Karl G. Jansky* conducted observations on static noise with a directional antenna and discovered sources of radiation of extraterrestrial origin other than the Sun. In 1937, his work was taken up and improved upon by radio engineer and amateur astronomer *Grote Reber*, who built the first dedicated radio telescope, a 9.5 m parabolic dish antenna and became the first person to conduct a survey of the radio sky. Although his observations were published in professional astronomical journals, his work did not receive much attention until after the conclusion of World War II. Radio technology received significant development during this time, due to the need of sensitive and efficient military radar systems that saw the adaption and development of *cavity magnetrons* as microwave sources, and *klystrons* and *travelling-wave tubes* as amplifiers. Soon after the war, the first institutional observatories were founded and radio astronomy became an important field of study.

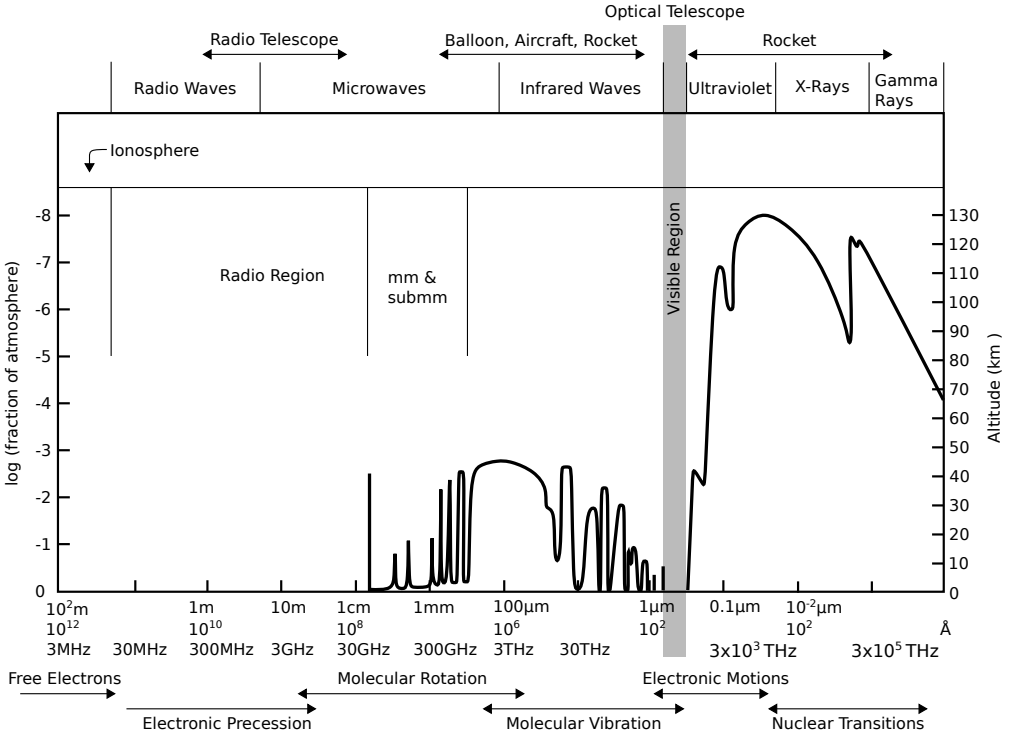


Figure 1.1: The transmission of the Earth's atmosphere for electromagnetic radiation. The diagram gives the height above sea level at which the radiation is attenuated by half. The absorption bands below 1 cm come mostly from H_2O and O_2. Absorption above 300 GHz is caused by other abundant atmospheric gasses such as N_2 and CO_2 (*fig.* recreated from [1]).

1.1 The Radio Window

Whether electromagnetic radiation can be observed from the ground, depends on the opacity of the Earth's atmosphere (*fig.* 1.1) at a given wavelength. Similarly to the atmospheric window of visible light, the Earth's atmosphere is transparent to a large portion of the radio spectrum, ranging from about 10 MHz to 1 THz. At the lower end of the spectrum, the radio window is limited by ionospheric reflection, as solar radiation creates electron plasma by ionization of gas in this atmospheric layer. The cutoff at the upper end is caused due to resonant absorption in the rotation bands of molecules in the lower atmosphere, primarily water vapour and oxygen.[1]

Since the millimetre and sub-millimetre range of the radio spectrum is rich on molecular spectral lines, there is great interest to extend the upper frequency limit as much as possible. This can be achieved by satellites, airborne observatories or high-altitude ground-based installations in a very dry climate, such as the Submillimeter Telescope (SMT) on Mount Graham, Arizona, or the Atacama Large Millimeter/submillimeter Array (ALMA) located in the Atacama Desert of northern Chile.

1.1.1 Ionospheric Reflection

When a radio wave encounters the ionospheric layer, the electric field forces the free electrons into oscillations of the same frequency, which then re-radiate the incoming wave energy. If however the frequency of the wave is greater than the oscillation frequency of the electrons, it is not reflected, but passes through.

The refractive index n in a plasma medium for a collision frequency ν is given by

$$n^2 = 1 - \frac{n_e e^2}{m_e \varepsilon_0 (\omega^2 + \nu^2)} ,$$

where n_e is the density, e the charge, m_e the mass of the electron, and ε_0 the electric constant. [2]

For normal incidence, reflectance is a special case of the Fresnel equations:

$$R = \left| \frac{n_1 - n_2}{n_1 + n_2} \right|^2 \quad .$$

Total reflectance is achieved when the refractive index falls to zero, so for an angular frequency of the wave $\omega \gg \nu$ with $\omega = 2\pi f$, the critical frequency at or below which ionospheric reflection occurs at vertical incident is given by

$$f_{\text{crit}} = \frac{1}{2\pi} \sqrt{\frac{n_e e^2}{m_e \varepsilon_0}} \approx 9 \cdot 10^{-3} \sqrt{n_e} \ \ [\text{MHz}]$$

for n_e in electrons per cm^3. This is also referred to as the *plasma oscillation frequency*.

Since the critical frequency is dependent on the angle of attack

$$f_\alpha \approx \frac{f_{\text{crit}}}{\sin \alpha} ,$$

where α is the angle above the horizon, the observing frequency must be chosen well above the critical frequency to avoid ionospheric effects.

Depending on altitude, time of day, and solar activity, typical values for n_e range from 10^5 to 10^6 e$^-$/cm^3 in the ionisation region of the Ionosphere (called the F-layer), causing a variation of the lowest observable frequency between about 3 MHz to 10 MHz. [3]

1.2 Astronomical Sources of Radio Signals

While extended objects observed in the visual spectrum usually radiate because of their inherent temperature (1.2.1), many radio sources are of non-thermal origin. The processes behind these emissions are diverse: MASERs coming from molecular clouds, molecular rotation and vibration, synchrotron radiation of charged particles in the outflows of astrophysical jets, *bremsstrahlung*, and hyperfine (spin-flip) transitions are the most common. Of the latter, the most relevant and well known is the 21cm line of neutral hydrogen. Still, thermal sources also emit a radio continuum as part of their black body spectrum and can thus be observed with radio telescopes.

Astrophysical MASERs are typically found in the environment around highly evolved late-type stars and in clouds in which molecular species undergo an excitation from their ground state, being forced into a non-equilibrium state of population inversion by a process called *pumping*.

Rotational spectral lines come from molecules that are characterised by a dipole moment and a shift between their centre of mass and their charge. Incident radiation can excite slow rotating molecules by exerting a torque. When the molecule returns to its de-excited state, the difference in energy is emitted as an electromagnetic wave. Molecular vibration occurs when the atoms of a constantly rotating or translating molecule are in periodic motion. The vibrational state can be excited by incident photons, similar to the rotational mechanism.

As vibrational transitions usually occur in conjunction with rotational transitions, they are commonly referred to as *rovibrational coupling* and occur in particular when then rotational frequency of a molecule is close to its natural vibrational frequency. Examples for abundant molecules which show spectral rotation and vibration bands are CO, CO_2 and H_2O.

Synchrotron radiation is generated by the gyrations of charged particles along magnetic fields. Prominent sources of this type of emission are outflows of ionised matter in the direction of the spin axis of astrophysical objects, so called *jets*, which are often found in the vicinity of black holes. The interaction with the magnetic field forces the charges to travel along the field lines in curved paths at relativistic speeds, thereby undergoing an acceleration perpendicular to their velocity vector, resulting in the emission of electromagnetic waves. The emitted wavelength continuum depends on the energy of the particle and may span over large parts of the electromagnetic spectrum.

Thermal *Bremsstrahlung* is another source which produces continuum emissions and is for example found in ionised interstellar clouds of HII. This process is also called *free-free* radiation, as the electrons are usually free before and after interactions with the ions.

The low levels of incident power and the angular resolutions required to resolve most celestial objects typically restrict small radio telescopes (in the low GHz range) to the observation of bright continuum emitters, such as the Sun or the Moon. MASER observations, for example of the OH species just above 1.6 GHz, are very challenging due to the point-like nature of the sources, considering a typical resolution of several degrees of the collecting dishes used in these types of systems. Neutral hydrogen is however very easily observed due to its ubiquitous presence within the Milky Way, spanning the entire sky.

1.2.1 Black Body Radiation

All (dense) objects emit electromagnetic waves because of the movement of charged particles in their atoms due to oscillations linked to their temperature. An idealized model of this property is the *black body*. Its radiation spectrum is described by *Planck's law*:

$$B(\nu, T) = \frac{2h\nu^3}{c^2} \frac{1}{e^{h\nu/kT-1}} \ [\mathrm{Wm^{-2}\,Hz^{-1}}] \quad .$$

Differentiation results in *Wien's displacement law*, which gives the frequency of the intensity maximum of a black body at a given temperature:

$$\nu_{\max} = 2.821\frac{kT}{h} \approx 5.879 \cdot 10^{10}\ T\ [\mathrm{Hz}] \quad .$$

The total power radiated per unit area is given by the *Stefan-Boltzmann law*, which can be derived by integration of Planck's law over all frequencies and the hemisphere into which the power is radiated:

$$j^* = \sigma T^4\ [\mathrm{Wm^{-2}}] \quad .$$

For the purpose of determining the radiated power emitted by thermal radio sources within a spectral range, it is prudent to use the approximation of spectral radiance as per the *Rayleigh–Jeans law*, which, unlike Planck's law, can be directly integrated. It describes the radiation for the long-wavelength tail of the black body spectrum and is given by

$$B(\nu, T) = \frac{2\nu^2 k_B T}{c^2}\ [\mathrm{Wm^{-2}\,Hz^{-1}}] \quad .$$

1.2.2 Neutral Hydrogen

Spin-flip or *hyperfine structure* transitions occur in atoms due to the differences in field energy levels between the spin states of electrons which interact with nuclear multi-pole moments. Depending on its configuration, an atom can be in a state of slightly higher energy. As the state changes to the opposite spin, the difference in energy is emitted as electromagnetic radiation. Although the transition process has a very low probability and is not detectable under laboratory conditions on Earth (and therefore also called *forbidden*), the sheer number of atoms in the interstellar medium (ISM) make observations with radio telescopes possible.

The most well-known transition is the 21 cm line of neutral hydrogen. It was predicted by *Hendrik van de Hulst* in 1944 and first detected by *Ewen*, *Purcell* and *Westerhout* in 1951. This transition splits the *1s* ground state of atomic hydrogen by only $5.9 \cdot 10^{-6}$ eV between the parallel and antiparallel spin states and is extremely small compared to the ground state energy of -13.6 eV.

However, as neutral HI makes up about 20 % of the ISM, 21 cm line emissions are easily detected. Observations show that the ISM is present practically everywhere, but its structure is quite irregular. There are large regions of extremely low density, but also very large accumulations in clouds. Unlike the optical spectrum, 21 cm radiation is not impeded by interstellar dust, making it an important tracer for revealing the structure and dynamics of the Milky Way. By determining the *Doppler shift* of the spectral line, the radial velocity of a source region can be calculated, and in combination with a model of galactic disc rotation, its distance can be inferred.

1.3 Receiving Systems

In order to measure the incident electromagnetic energy, one of two detection principles can be applied. *Coherent* receivers can preserve the amplitude and phase information of radio waves by directly coupling them to the receiver electronics, while *incoherent* receivers destroy phase information. The latter are usually a type of *bolometer*, which are high-bandwidth, highly sensitive thermometers that measure temperature increase by the absorption of the wave energy. These detectors are typically used in millimetre and sub-millimetre frequency ranges. By design, they can only determine the amount of incident energy, so they must be combined with devices such as *Fourier transform spectrometers* (FTS) when used for spectral line observation. [4] The achievable spectral resolutions are usually limited by the construction sizes of these instruments. For example, a two-sided FTS with a maximum (optical) path difference of 1 meter will have a resolution of $2\Delta_{\text{opd}}\lambda^{-1} = 2000$ [5] for a wavelength of 1 mm. Polarization measurements with bolometers are possible, but require specially designed mounting grids (e.g. [6]).

Coherent receivers are typically based on the *superheterodyne principle*, in which the frequency observed at the *frontend* is mixed with a slightly different frequency produced by a tuned *local oscillator*. This produces the *intermediate frequency*, which is the beat frequency that is used in the sampling process of the *backend* to convert the signal to a digital data representation. The intermediate frequency output band corresponds to the passband of the original signal, but is shifted down along the frequency axis. A major advantage of this technique is the ability to operate on a fixed centre frequency and passband in the backend, so filters and data sampling equipment are not only independent from the frontend receiver, but can also be operated in always the same configuration. This is especially useful at higher frequencies,

where the digital electronics in analysing equipment becomes increasingly harder to implement due to technological constraints. As an additional benefit, digital data rates become lower by several magnitudes. Since the sampling frequency (as per the *Nyquist–Shannon sampling theorem*) must be at least twice as high as the highest input frequency, processing of the mixed - rather than the original - signal can be done at much lower rates. Coherent receiving systems can measure linear polarisation angles by the orientation of the pickup antenna in the feed. Circular polarisation is not very common in most astronomical sources, with the exception of OH masers in galactic star forming regions, which often show a high degree of circular polarisation (e.g. [7]). Left- or right-handed helical antenna designs are very suitable for measuring these types of sources.

1.3.1 Terminology

Although radio astronomy relies solely on passive reception of radiated energy, the terminology in the context of radio systems is often used from the perspective of a transmitting, rather than a receiving configuration. This is because most terms and definitions in the field were coined during the development of radio telecommunication technology, long before it was applied to astronomical research. For example, the term *beam* refers to the radiation power pattern of an antenna, but can equally be used for the sensitivity response pattern in a receiving antenna, which is an identical, but inverse, property. Similarly, *illumination* is used to describe the efficiency of a radiation pattern with regard to signal amplitude variations on a (secondary) aperture, but can also be used in the characterisation of receiver gain losses over the capture area of an antenna.

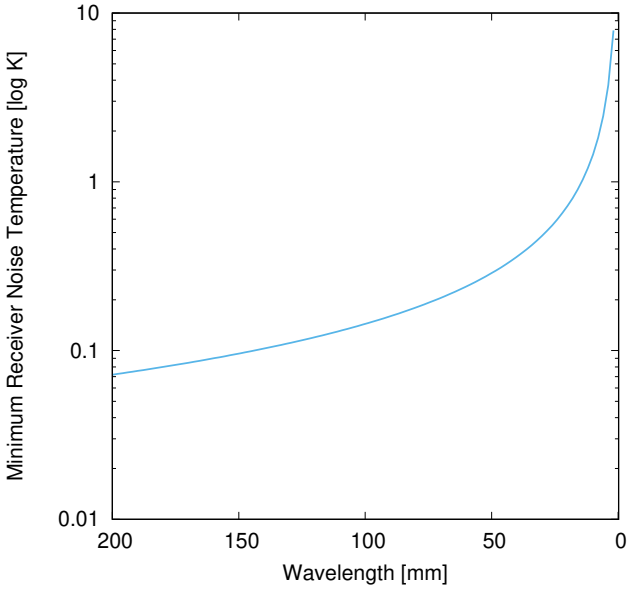


Figure 1.2: Minimum noise for a coherent receiver. The noise contribution is small for longer wavelengths, but quickly increases in the sub-millimetre range.

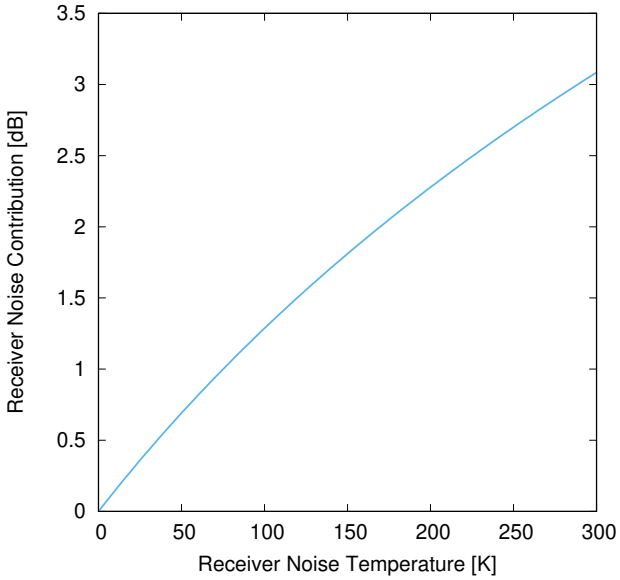


Figure 1.3: Noise figure for $T_0 = 290$K. It corresponds to the signal-to-noise ratio of a receiver, so an improvement of 1 dB will improve the SNR by about 1 dB.

1.3.2 Receiver Noise

The minimum noise in a coherent receiver is caused by the uncertainty in input photon number and input phase and is derived by applying the *Heisenberg uncertainty principle*:

$$T_{\mathrm{Rec}}(\mathrm{min}) = h\nu/k \quad .$$

While the effect is mostly negligible down to wavelengths of a few millimetres, e.g. 3 K at 5 mm, it quickly increases to 30 K at 0.5 mm. Incoherent detectors, such as bolometers, are not affected by this fundamental limit, as phase information is not preserved. [8]

Unless active cooling is applied, the most dominant noise source is *thermal noise*. It is caused by the random thermal motion of electrons in a conductor (*Johnson–Nyquist noise*). Since thermal noise is additive white Gaussian noise, the total noise power detected over the *operating bandwidth* Δf is given by

$$P_{T_{\mathrm{rec}}} = k_B T_{\mathrm{Rec}} \Delta f \quad .$$

The receiver noise added to an input noise temperature T_0 (usually assumed to be 290 K for terrestrial applications) is quantified by the *noise factor*:

$$F = 1 + \frac{T_{\mathrm{rec}}}{T_0} \quad .$$

The noise factor can also be expressed in decibels and is then called the *noise figure*:

$$F_{\mathrm{dB}} = 10 \log(F) \quad .$$

Another way of expressing noise is the *noise temperature*:

$$T_N = T_0(10^{F_{\mathrm{dB}}/10} - 1) \quad .$$

The noise factor contributions of each stage in a cascade of components can be calculated using the *Friis formula for noise*:

$$F_{\mathrm{total}} = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + \cdots + \frac{F_n - 1}{G_1 G_2 \ldots G_{n-1}} \quad .$$

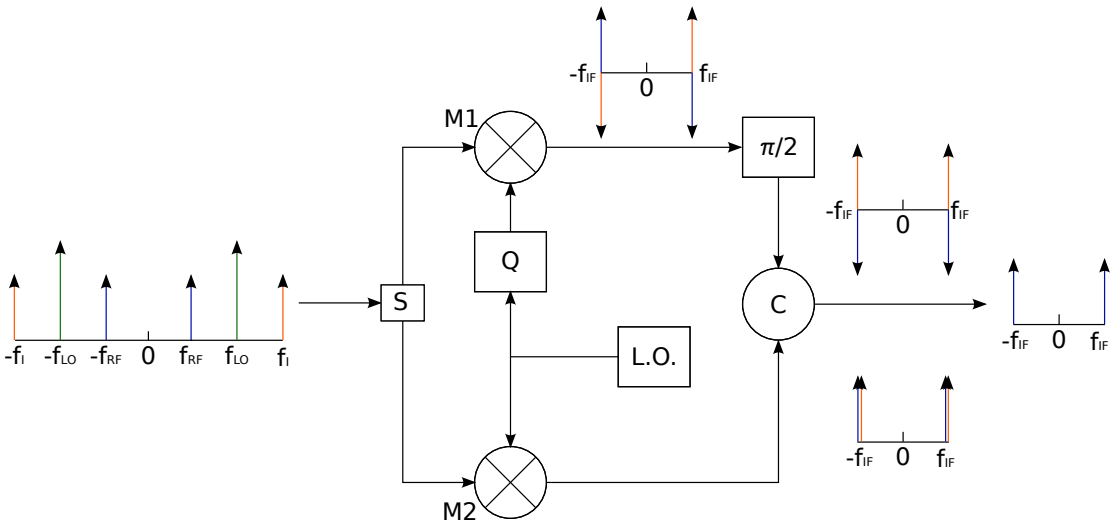


Figure 1.4: Working principle of an image reject mixer. The input signal (left) is divided in half by a splitter (S) and fed to the mixer inputs (M1, M2). The inputs are mixed with the signals from the local oscillator (L.O.), of which one is in *in quadrature* (Q) (i.e. an orthogonal sinusoid). The 90° phase-shifted signals are then recombined (C), resulting in the suppression of the unwanted sideband.

Except for the unlikely case of a very noisy component in a later stage, it is obvious that the first two terms define the significant part of the noise factor, thus making the choice of preamplifier critical for radio astronomy applications.

1.3.3 System Temperature

In addition to receiver noise, the thermal environment contributes radiative energy through spillover and side lobes (1.3.8) of the feed and antenna. The total noise temperature of all effects combined is called the *system temperature*:

$$T_{\mathrm{sys}} = T_{\mathrm{amb}} + T_{\mathrm{rec}} + T_{\mathrm{sky}} \,,$$

where T_{sky} contains emissions from the atmosphere T_{atm} and $T_{\mathrm{cmb}} \approx 3K$ of the isotropic cosmic background radiation. T_{amb} is the noise picked up from the environment, such as from spillover, and T_{rec} is the total receiver noise temperature including the contributions from the amplification stages.

1.3.4 Image Rejection

A problem in superheterodyning receivers is *image frequency*. The mixing process is a mathematical operation, which produces intermediate frequencies for both sides of the input centre (carrier) frequency, i.e. lower and upper side bands, which are the signal components above and below the carrier. This results in unwanted noise and interference in the sampling electronics, if both side bands are combined on the same output line. One approach to solve this problem is to use an *image reject mixer* (*fig.* 1.4). Here, the carrier signal is split in-phase and one part is mixed with the local oscillator signal, while the other part is mixed with the signal shifted by a quadrature generator. This process gives the same target frequency in both outputs. At this stage, the image components are in quadrature. The intermediate frequency signals are then subtracted in a 90° phase shifter, effectively cancelling out the undesired image frequencies.

1.3.5 Angular Resolution

The angular diameter of the *main lobe* (*fig.* 1.6) at half-power (-3 dB) down from the peak power of the main lobe is called the Half-Power Beam Width (HPBW) or Full Width at Half Maximum (FWHM). It corresponds to the spatial resolution of the telescope and is approximated by the *Rayleigh criterion* for the minimum resolvable detail:

$$\angle\theta = \arcsin\left(1.22\frac{\lambda}{D}\right) \quad .$$

1.3.6 Antenna Gain

The antenna gain is the intensity ratio of an antenna in a given direction (its *directivity*) compared to an ideal antenna which radiates isotropically, i.e. equally in all directions.

The maximum gain for a circular aperture parabola can be estimated from:

$$G \approx \pi^2 \frac{D^2}{\lambda^2} \quad .$$

This value is usually written as *decibel gain over isotropic*:

$$G_{\mathrm{dBi}} = 10\log_{10} G \quad .$$

1.3.7 Surface Error Losses

Under the assumption that surface errors follow a random Gaussian-shaped distribution with a mean of zero and a standard deviation equal to the *root mean square*, an approximation of the efficiency of a reflector due to surface imperfections is given by Ruze's equation [9]:

$$\eta_R = \exp\left[-\left(4\pi\frac{\sigma}{\lambda}\right)^2\right] \quad .$$

The loss can also be expressed in decibel gain:

$$\Delta G = 10\log_{10}(\eta_R) \quad .$$

1.3.8 Illumination Losses

Illumination (or signal) losses have a variety of causes. The most obvious in dish-based arrangements (*fig.* 1.5) is blockage by the prime focus feed horn or reflector, which shades part of the dish from the incident radio energy. The radiation pattern of a feed or secondary reflector usually does not perfectly match the geometrical shape of the dish, so less signal is being picked up towards the edges. This is referred to as the *aperture illumination* or *illumination taper* of the antenna. It comes from a trade-off between taper efficiency and the ratios of the radio power received in the *main lobe* (the beam component with the highest field strength) to the (undesired) signals received in the higher-order *side lobes*. This unwanted signal component is called *spillover*. Since it is not possible to cut off the radiation pattern exactly at the edge of the dish, some noise from the ground will be picked up by the feed. Increasing the taper will also decrease the amount of spillover, so typically a more tapered distribution is desirable over uniform illumination, as it improves the main beam efficiency and reduces unwanted ingress of environmental noise, although it comes at a loss of gain and overall total efficiency of the system.

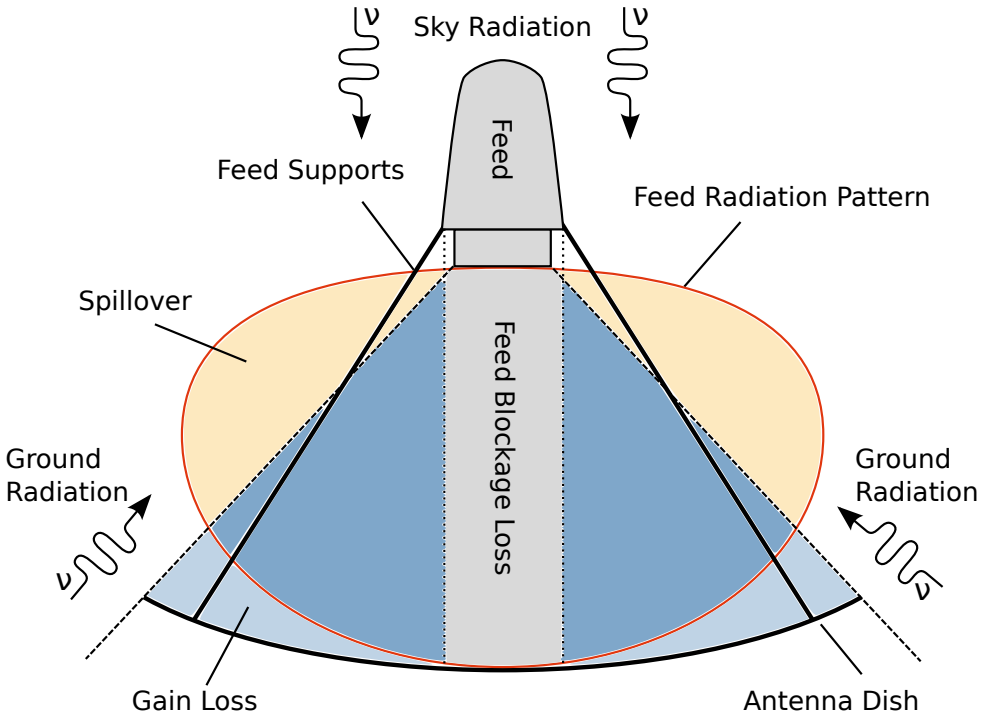


Figure 1.5: Signal losses in dish antenna configurations stem from the area shaded by the feed horn (or reflector in Cassegrain-type arrangements) and the loss of sensitivity due to the feed's radiation pattern, which results in imperfect illumination of the dish. A major ingress of unwanted thermal noise also comes from the feed radiation pattern, which partially looks at the ground behind the dish itself (*spillover*). The first sidelobe of the dish's radiation pattern also contributes spillover, when at low elevations (not illustrated).

1.3.9 Antenna Beam Pattern

The idealised beam pattern of a dish-type antenna corresponds to that of a Fraunhofer diffraction pattern of a circular aperture (*Airy disk*):

$$P(\theta) = A_0 + A_1 \left(\frac{2J_1(\tau)}{\tau} \right)^2$$

with

$$\tau = \frac{\pi(\theta - \theta_0)}{\lambda / D},$$

where θ is the angular offset from the centre of the beam, λ the wavelength, D the aperture diameter and J_1 the spherical Bessel function of order one. The peak shift from zero-point θ_0 is equal to the *pointing error*.

Side lobes are the parts of the far field radiation pattern (*fig.* 1.6) of an antenna, which are not the main beam. The first sidelobe is usually the most significant in dish antennas, with a typical peak sensitivity that can be expected to be in the range of -20 to -15 dB (or 1–3 %) relative to the peak gain of the main lobe, given sensibly efficient configurations (*fig.* 1.8, 1.9). Other losses that are related to illumination come from reflector surface deviations (1.3.7), improper focus positioning, angular misalignment of the feed or secondary reflector, and scattering on the feed support struts. In feed horns, side lobes contribute part of the spillover signal fraction. For suppression, the horn can be corrugated (e.g. [10]) by manufacturing concentric grooves or steps along the inner wall of the horn flare.

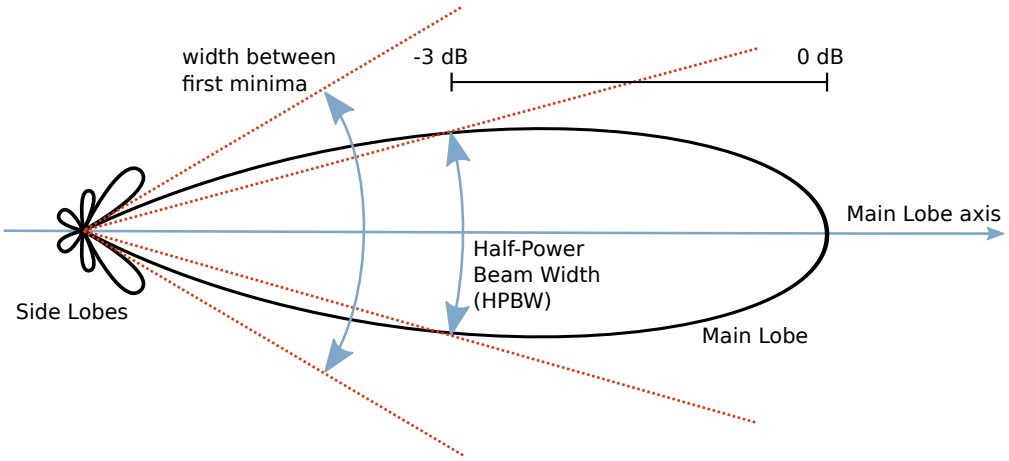


Figure 1.6: The idealised antenna beam consists of the main lobe, which contains the wanted signal components and the side lobes that contribute unwanted signals from other directions. The antenna resolution is considered to be the angular diameter at half the peak sensitivity (-3 dB) of the main lobe.

1.3.10 Antenna Performance

In order to achieve optimum performance with dish antennas, the feed radiation pattern (i.e. the directional sensitivity) would have to match the shape of the reflector in a way that the illumination is constant over the whole dish. Since the distance from the focus increases towards the edge of the reflector, more incident energy (or sensitivity) is required compared to the centre of the curve, while undesirable spillover must be kept low at the same time. As the required radiation pattern cannot be realized with real antenna feeds, perfectly uniform dish illumination is impossible to achieve. This results in illumination loss, causing the effectiveness to decrease towards the edge (*fig.* 1.7) and is referred to as *edge taper*.

The illumination function for a circular aperture of unity radius is given by *Hansen's one-parameter function* [11], where the singular parameter h determines the shape of the taper:

$$f(r) = I_0\left(h\sqrt{1-r^2}\right) \quad .$$

As h becomes larger, the edge taper increases and the side lobe levels (*fig.* 1.9) go down, along with the aperture efficiency. Additional performance degradation is caused by feed blockage. The linear blocking fraction β is

$$\beta = \frac{d_{\mathrm{Feedhorn}}}{D_{\mathrm{Reflector}}} \quad .$$

The upper limit to the overall efficiency of a dish antenna, combining illumination and blocking, is given by

$$\eta = \eta_i\eta_b = \frac{4\left(1-\beta^2\right)}{h^2}\frac{I_1^2\left(h\sqrt{1-\beta^2}\right)}{I_0^2\left(h\right)-I_1^2\left(h\right)} \, ,$$

where I_n is the modified Bessel function of order n. To achieve high efficiency and gain, a low edge taper is desirable and a minimal blocking fraction is needed, which should not exceed more than 10 % of the aperture size to avoid performance degradation (*fig.* 1.8). This also implies a minimum size that should be chosen for the diameter of a dish reflector antenna, as the dimensions of the feed antenna are determined by the observed wavelength. The choice of edge taper and blockage also have significant impact on the side lobe levels and thus control the environmental noise component in the recorded signal.

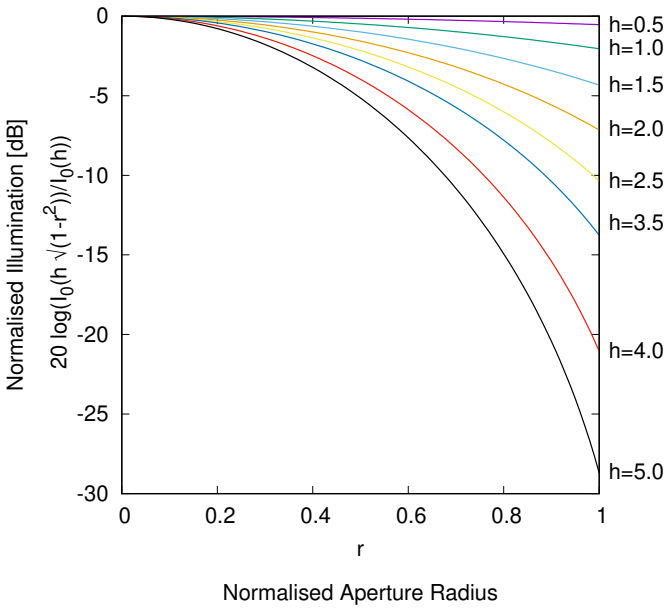


Figure 1.7: The efficiency of dish illumination is dependent on the choice of edge taper. While low tapers offer good consistency in signal power towards the edge of the aperture, it is also clear that a significant amount is lost past the boundaries of the dish.

The level of the first side lobe including blockage is given by [12]:

$$\text{SLL} = \frac{-0.1323 - \beta^2 \bar{I}_0 \Lambda_1 \left(\beta\sqrt{h^2 + 5.14^2}\right)}{2h^{-1} I_1(h) - \beta^2 I_0},$$

with

$$\bar{I}_0 = \frac{1}{2}\left[I_0(h) + I_0\left(h\sqrt{1-\beta^2}\right)\right]$$

and

$$\Lambda_n(x) = 2^n n!\, x^{-n} J_n(x)$$

where J_n is the (ordinary) Bessel function of order n.

Low side lobe levels result in low antenna noise temperatures, so low-noise designs should aim to keep the blocking fraction small (*fig.* 1.9). Edge taper is a trade-off between signal gain and spillover, so a compromise between illumination loss and spillover loss has to be found to optimise a configuration for a particular application.

1.3.11 Aperture Efficiency

The efficiency of an antenna is defined as the ratio of the effective aperture to the true physical aperture [13]:

$$\eta_{\text{ap}} = \frac{A_e}{A_p} \quad .$$

The effective aperture A_e is considered to be the remaining aperture after feed and support blockage (1.3.10), spillover (1.3.8), and under-illumination (1.3.9) due to the radiation pattern, as well as surface structure irregularities (1.3.7) and ohmic losses in the conductive surface material are taken into account.

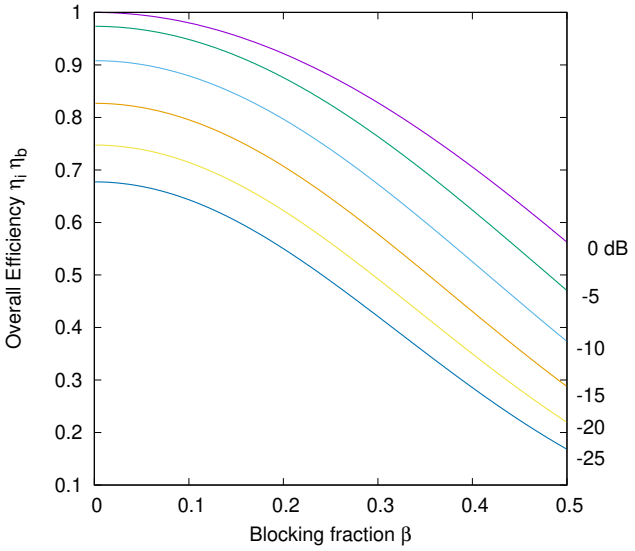


Figure 1.8: Overall efficiency is generally good for medium edge tapers and low feed blockages, which should not be larger than about 10 % of the main dish, otherwise performance degrades quickly. Note: edge taper curves are listed in terms of normalised illumination instead of h (see *fig.* 1.7).

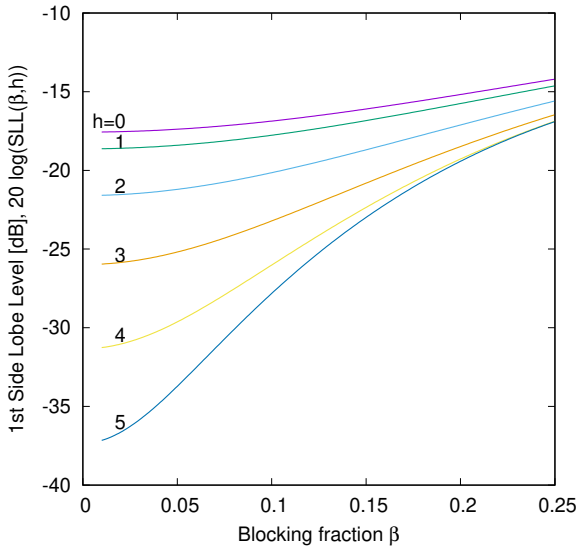


Figure 1.9: For low blocking fractions, most edge taper configurations generally give a reasonable side lobe level, but efficiency decreases significantly if the feed or secondary reflector becomes large compared against the reflector aperture.

The relationship between the measured incremental antenna temperature ΔT_a and the source flux density S for an aperture A_e ultimately derives from the *Johnson–Nyquist noise*

$$p = k_B T$$

in which p is the noise power density generated by the thermal movement of charge carriers inside a conductive element for a given temperature T.

For a uniformly illuminated antenna, the received power is a simple relation between collecting area and source flux density:

$$p = S A_e \quad .$$

The effective aperture of an antenna can therefore be written as [14]:

$$A_e = \frac{2 k_B \Delta T_a}{S} \, ,$$

or in terms of aperture efficiency as

$$\eta_{ap} = \frac{2 k_B \Delta T_a}{S A_p} \quad .$$

1.4 Absolute Amplitude Calibration

The output of a receiver is typically in arbitrary units of count C, which must be converted to a flux density scale or equivalent temperature for meaningful quantities of measured values. The calibration factor g is either determined for the receiver noise temperature T_{rec} in a closed-off configuration for a two-load comparison, or in a hybrid semi-open or fully open configuration to get the composite system temperature T_{sys}.

1.4.1 Closed Two-Load Calibration

The two-load calibration procedure uses the so-called *Y-factor* method, which uses hot and cold loads to determine the conversion factor g. The most straightforward approach is to terminate the receiver system input at the preamplifier with a matched resistor that is kept at two sufficiently different temperatures T_{hot} and T_{cold}, and thus acts as a thermal noise source of known magnitude (1.3.2). The Y-factor is determined from the receiver counts measured at both temperatures:

$$Y = \frac{C_{\mathrm{hot}}}{C_{\mathrm{cold}}}$$

and the receiver noise temperature:

$$T_{\mathrm{rec}} = \frac{T_{\mathrm{hot}} - Y\,T_{\mathrm{cold}}}{Y - 1} \quad .$$

The main advantage of this method is that it gives a more accurate calibration, as it excludes the unknown external noise contributions to the total system temperature T_{sys} (1.3.3), because any noise in the system originates from the temperature variance of the terminated input and the constant noise generated by the receiving system itself. A major drawback is that the cold load configuration requires cooling, e.g. by liquid nitrogen, so that a reasonable temperature differential can be achieved.

1.4.2 Semi-Open Single-Load Calibration

A variant of the closed two-load calibration is a hybrid method which uses an absorber with the characteristics of a black body (1.2.1), or a matched resistor as the hot load (at ambient temperature) to block off the receiver input. Instead of a cold load, this approach uses the blank sky temperature as second reference, resulting in what is referred to as the *effective system temperature*, which is equivalent to T_{sys}, but excludes the blank sky noise contribution:

$$T_{\mathrm{sys,eff}} = T_{\mathrm{hot}} \frac{C_{\mathrm{sky}}}{C_{\mathrm{hot}} - C_{\mathrm{sky}}} \quad .$$

A problem which can arise with this calibration method, is that the absorber material can retain moisture from the environment, which may change its black body properties. If this is the case, the matched resistor method is preferable.

1.4.3 Open Calibration

In this calibration method, the system temperature is determined by a direct comparison of the signal counts with a known external source, to the measured blank sky counts. In the most simple case, the external source can be an object of known temperature, e.g. a building or a tree, or an object on the sky of known temperature. A more reliable source for reference signals are noise diodes, which radiate power at a known equivalent temperature T_{inj}.

Their output is preferably directly injected into the signal path, but they may also be positioned externally, so that they radiate directly into the prime feed antenna.

The system temperature is determined from:

$$T_{\mathrm{sys}} = T_{\mathrm{inj}} \frac{C_{\mathrm{sky}}}{C_{\mathrm{inj,on}} - C_{\mathrm{sky}}} \quad .$$

1.4.4 Calibration Factor

The conversion constant g from electronic unit counts to temperature is calculated from

$$g = \frac{T_{\mathrm{hot}} - T_{\mathrm{cold}}}{C_{\mathrm{cold}}} \quad .$$

Here T_{hot}, T_{cold}, and C_{cold} represent the equivalent values for the particular method used. Where the blank sky is used, T_{cold} is at least the temperature of the cosmic microwave background $T_{\mathrm{cmb}} \approx 3K$, depending on the opacity of the atmosphere at the given frequency.

1.5 Observational and Measurement Techniques

Radio astronomy makes use of a variety of observation strategies, which depend on the source type, scientific goals, as well as the available technology. One of the oldest methods is *total-power radiometry*, which measures the integrated noise power received from a source. This is done with power meters, which operate similarly to bolometers and measure the total signal contained within their characteristic bandwidth (1.3.2). This makes this type of measurement most suited for continuum sources and is also the reason why source signal amplitudes are traditionally given in units of temperature.

For *spectroscopy*, the available full bandwidth must be divided into sub-channels of smaller passbands. In analog setups, this is done by banks of narrow bandpass filters, where the signal is measured at each filter output. In the recent decades, powerful digital sampling electronics have become available, which allow the waveform to be recorded at full bandwidth so it can later be Fourier-transformed to obtain the spectrum numerically. This method is not only much cheaper to implement than custom, high-quality analog filters, but also allows for easy re-analysis of the recorded data and provides flexibility in the selection of bandwidth resolution.

Systematic effects in the recorded signal originate in the receiving system or the local environment of the telescope. To overcome and remove those, techniques such as *position switching* (also known as *beam switching*) or *frequency switching* can be used. In position switching, the source signal is compared to the signal recorded at an angular offset on the sky. The switch between a source and the background can be done by moving the whole telescope, or by using a movable secondary reflector. Frequency switching is based on the fact that most line emissions are usually narrow-band. Given that the level of sky and environmental background noise does not usually change for nearby frequencies, the local oscillator frequency is repeatedly shifted by a few bandwidths to acquire a reference signal. A variation of this method is in-band switching, where a smaller shift is applied, so the desired line appears in both the source and the reference spectrum, resulting in improved observation time efficiency.

Radio telescopes are typically single-sensor instruments and cannot be used for imaging in the traditional sense. Instead, maps of extended sources are acquired by scanning a grid of points. If the shape of the beam is well characterised and if the grid is sufficiently oversampled, i.e. the coordinate step size is much smaller than the HPBW, de-convolution techniques may be used to improve the resolution beyond the instrument's baseline.

2. Control Software

The main aim of this work is the creation of software for use with small radio telescopes in order to provide ease of access to the instrument and to offer a low threshold of entry into the field of radio astronomy. The complexity and amount of data generated by modern instruments require a considerable amount of detailed knowledge in the tools and methods needed for observation planning, processing, reduction, and interpretation of the results. This is a hurdle when introducing novices to the field, since it requires a significant amount of motivation and effort on their part. An obstacle to many is the lack of experience in handling radio telescopes. This is contrary to optical instruments, which are available widely and in many sizes. *Small* radio telescopes can fill this niche, although they are only small relative to the dimensions of comparable institutional scientific facilities. They are often self-designed and built, or assembled from publicly available plans and instructions, using commercially available components, but may also be available as kits or turn-key solutions from commercial vendors.

The software described hereafter intends to provide a modern, unified interface which is capable of remote access to a radio telescope via a network connection. With its primary use case being a teaching utility, it allows for the participation of multiple passive users, with a single user in control of the instrument. This enables a tutor to demonstrate or assist students with observation tasks, while everyone else can run an instance of the software user interface on their own machine at the same time. Although radio telescopes are not usually affected by time of day, it may be necessary to perform observations as part of homework assignments or outside of course hours, particularly if a target is below the horizon or at an elevation not yet accessible to the instrument. To facilitate communication in such situations, and also to show a list of current users of the telescope, a messaging system is integrated into the software.

In order to support different configurations of instruments, the networked endpoint (i.e. the server) implements an Application Programming Interface (API) for software plugins, which are linked into the main server code at runtime. These plugins must be provided by the operator of the telescope. They must implement pre-defined call interfaces so that the particular hardware characteristics are mapped to the behaviour expected by the backend API, in order to be controlled properly by the server and therefore, the user. This is done so that the user interface can present parameters like telescope pointing and spectrum acquisition settings in a unified way and independently from the connected instrument's particular properties.

For easy navigation, to provide an overview of observable objects, and to show the current telescope pointing, a map of the visible sky at the geographical location of the telescope is presented to the user. If the local horizon blocks visibility of parts of the sky, a horizon profile can be added to indicate unobstructed areas. The sky map also accepts coordinate input for slewing the telescope. The user can command this by clicking a desired target pointing on the map. Furthermore, the map can be shifted in time, so a past or future view of the sky is displayed. For more explicit control, input fields are available, which allow the selection of the desired celestial reference system, specification of coordinates, and tracking of a selected target.

The primary data display is the spectrum viewer, which is capable of concurrently plotting a configurable number of spectra received from the telescope and a configurable moving average thereof. To indicate the historical order of the received data, the individual spectra are displayed with an *alpha channel* and *fade* as they age, creating a *digital phosphor* effect. This gives the user a more intuitive insight over any temporal effects present in the data, either due to external unwanted interference, or changes in the observation, such as drift or explicit position change. To aid in the examination of the data, the spectrum data view can be zoomed and shifted. Besides the frequency axis, an alternative radial velocity axis is displayed. The reference rest frequency is configurable, with the reference system being the Velocity of the Local Standard of Rest (VLSR). For further analysis of spectral lines, an area can be selected by drawing a box, which marks the set of points that become the input to a fitting routine. This routine attempts to fit a Gaussian function and displays results such as peak frequency, radial velocity, and amplitude of a line. Like for position control, input fields are available for spectrometer settings. These include selectable input modes, either by frequency or velocity equivalent (given a reference rest frequency) and other settings of the receiver, such as spectral resolution. To auto-adjust the receiver frequency for a given radial velocity equivalent, the *Doppler tracking* function can be enabled, which compensates for the VLSR when the telescope is used to scan across the sky. This is particularly useful when creating position-velocity maps of objects like the Milky Way, where the centre of the spectrum should correspond to the same radial velocity for every position observed. If available on the instrument, a hot load can be toggled for calibration purposes.

In addition to the spectrum display, the average amplitude of a dataset is displayed as a function of time. This is useful in the observation of continuum sources such as the Sun and for tracking drifts in the sensitivity of the receiver due to environmental temperature variations. To illustrate the temporal change in the spectral data, a spectrogram (*waterfall* diagram) is rendered, where the horizontal axis corresponds to the frequency and the vertical axis represents time. The spectral bin amplitudes are rendered as a heat map. This mode of display is well suited for identification of spurious signals and the variability present in the radio background in the local environment of a telescope.

As it is more comfortable to execute complex or time-consuming observations in automated procedures, rather than manually, a set of pre-programmed routines are available to the user. These include, among others, map-making, scans along the galactic plane, and beam switching operations. The user typically only needs to set up the spectrometer in a configuration suitable to the selected observation mode and then configure parameters such as coordinate ranges, number of spectral acquisitions per measured position, and number of scan repetitions. Results are presented in a pre-reduced fashion, i.e. as spectral data, graphs, or heat maps, depending on the chosen mode.

2.1 Requirements

2.1.1 Overview

This section formally lists a set of fundamental requirements of the software. They are divided into two categories: *user requirements*, i.e. the features and properties of the software required by a user or operator of an instrument, and *software requirements*, which concern the way the software and its functionality are to be implemented. A unique identifier is assigned to each requirement, which is used to reference and trace it to the software component design.

2.1.2 User Requirements

R-USR-001	Short Text	Software Requirement
	Backend Support	The software shall be adaptable to a variety of radio telescope mounts and instruments.

R-USR-002	Short Text	Software Requirement
	Networking	The software shall provide access to an instrument via a network connection, so it may be operated remotely.

R-USR-003	Short Text	Software Requirement
	Multiple Clients	The software shall implement network access such that multiple clients may connect to the instrument, but only a single user may be in control at a time.

R-USR-004	Short Text	Software Requirement
	Client Privilege Levels	The software shall implement control privilege levels to enable command and control override and a maintenance mode.

R-USR-005	Short Text	Software Requirement
	User Communication	The software shall implement a form of inter-user communication infrastructure. This infrastructure shall also be responsible to track the current number of connected clients, as well as the active privilege level of each client.

Comment This can be implemented in a style reminiscent of internet relay chat (IRC) client software, providing a text entry interface, line-by-line message display and a list of usernames.

R-USR-006	Short Text	Software Requirement
	Common Control Interface	The software shall implement all parameter controls in a unified fashion so it is applicable to most backend instrumentation.

Comment This concerns the way frequency, data rates and telescope pointing are configured by the user. The intent of this requirement is to provide a unified interface which is usable independently and without in-depth knowledge of the particular characteristics of the attached backend.

R-USR-007	Short Text	Software Requirement
	Graphical User Interface	The software shall implement a graphical user interface which provides control and display of telescope parameters and data through a point-and-click interface.

R-USR-008	Short Text	Software Requirement
	Observation Modes	The software shall implement predefined observation programmes, as well as a means to freely define such programmes.

R-USR-009	Short Text	Software Requirement
	Data Display and Reduction	The software shall, as part of its GUI, provide basic functionality for displaying recorded data and to perform simple data reduction steps.

2.1.3 Software Requirements

R-SWA-001	Short Text	Software Requirement
	Programming Language	The software shall be written in the C programming language.

R-SWA-002	Short Text	Software Requirement
	External Library Dependencies	The use of external libraries of the software shall be limited to *GTK+ 3* or later and the dependencies thereof.

R-SWA-003	Short Text	Software Requirement
	Networked Model	The software shall implement the hardware control component (*server*) and the user interface component (*client*) as standalone programmes. Communication between these components shall be implemented via a network layer.

R-SWA-004	Short Text	Software Requirement
	Communication Protocol	The software shall implement a data exchange protocol to communicate commands, acknowledgements and data.

R-SWA-005	Short Text	Software Requirement
	Backend Abstraction Layer	The software shall implement an abstraction layer for backend calls in the hardware control component. This abstraction layer shall provide a generic way of interaction with an instrument.

R-SWA-006	Short Text	Software Requirement
	Hardware Driver Plugins	The software shall implement a plugin-system in the hardware control component, which is able to load hardware drivers modules provided as object code and map calls of the backend abstraction layer to their corresponding implementations in the hardware driver module.

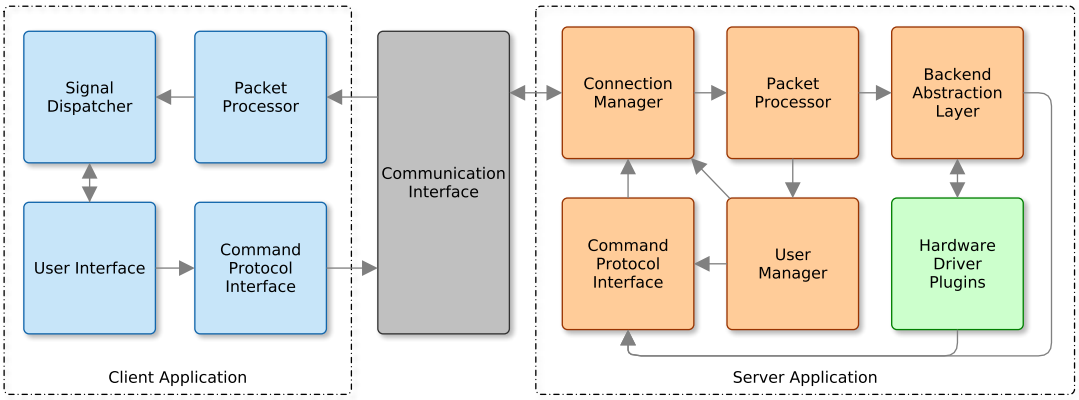


Figure 2.1: The top-level architecture of the software. It consists of a client and a server component which communicate via a network interface. Data is exchanged via command packets, which trigger functions in other parts of the software.

2.2 Software Design

2.2.1 Overview

This section describes the architectural design of the software shown in *fig.* 2.1. It is divided into two components: the server application and the client application. They transfer data and commands via a packet exchange protocol across a network connection.

The client application is the principle user interface to the software. It connects to a single remote server during a session. When the connection is established, it requests information which describe the characteristics and operational parameters of the radio telescope. These include geographical location and the capabilities of the drive and mount, such as range of motion and axis step size resolution, which are needed for the setup of the telescope pointing interface and the projected sky map. To operate the receiver, the settings ranges of the spectrometer are requested and the corresponding user interface is configured accordingly. Finally, the client application registers a user identifier for the purpose of message exchange to other clients via the server. As the software setup is complete, the client awaits input, which may arrive on the network interface, sent by the server, or come from user input via the Graphical User Interface (GUI).

Command and data packets received from the server are processed in the *packet processor*, which delegates the indicated action to the *signal dispatcher*. The dispatcher operates a callback system that implements signals with accompanying data structures. Software components of the client (such as GUI widgets) can register functions to this signal, which are executed by the dispatcher whenever the particular signal is emitted. The data are passed to the component as a function argument.

An example for a signalling event is the *spectral data* signal, which is triggered by the packet processor when spectral data are received from the server via the network interface. The dispatcher accepts the data as input and executes all registered callbacks. If one or more plotting widgets are active in the client application, and if they have registered their input callback functions to the dispatcher, they will receive a reference to the data, which they may then display in the *user interface*.

Conversely, the signal dispatcher may be used by interface components to emit signals in order to inform other components of updates. The *tracking* signal, for example, can be emitted by the sky map display, if the user selects one of the catalogue objects. In such a case, the control switches in the telescope pointing control interface would toggle their state to indicate the active tracking mode.

For sending requests to the server, interactive user inputs or pre-programmed procedures can call functions in the *command protocol interface*, which generates the packets to send via the network connection. The server accepts multiple client connections in parallel, the handling of which is the responsibility of the *connection manager*. This component is responsible to feed command packets received from connected clients to the packet processor and to distribute broadcast messages sent by other server components back to the clients.

Depending on their type, the *packet processor* executes commands on either the backend abstraction layer or the *user manager*. The latter is responsible for passing messages between clients, to manage user privilege levels that enable instrument control, and is also tasked with disconnecting timed-out or slow clients, which would block the network output queues, if they are unable to process or receive data packets within a certain amount of time.

The *backend abstraction layer* is based on a generic call interface model, which maps functions implemented by telescope hardware driver plugins. If certain calls are not implemented or a matching plugin is not loaded, the abstraction layer can respond with an error message indicating the failed outcome of a command packet request.

The *hardware driver plugins* implement the actual functionality of the instrument. They are responsible for the acceptance and conversion of requests in a manner which is suitable to the hardware component, and to condition the signals and measurement data received from a drive, spectrometer or other device in a way, which ensures the interface requirements of the backend abstraction layer are fulfilled and can be interpreted properly.

Software modules can send messages to connected clients via the functions provided by the *command protocol interface*, which constructs network packets and hands them off to the connection manager for distribution.

2.2.2 Server Design

D-SAD-001	Identifier	Function
	Backend Support	The server application component implements a plugin system that is capable of loading selected software extensions which are specified in a configuration file. These extensions implement hardware device functionality in backend calls, which the command processor of the server expects to be present. As the server loads a plugin module, it inspects it for specially marked functions, which must match the function names and API specified in the backend abstraction layer. The server then links these functions to the calls in the abstraction layer and executes special initialisation functions provided by the plugin, which are used to set up the prerequisites needed in the handling of the hardware device, such as threads, or loading of configuration or calibration files.

Purpose R-USR-001, R-SWA-005, R-SWA-006

Comment Extensions are typically part of the same source tree as the software itself and are built along with the software, but may be supplied from external sources in binary form, as long as they obey the API call specification.

D-SAD-002	Identifier	Function
	Networking and Commands	The server component implements a network interface which is tasked with connection and user privilege management. Packets received by the interface are inspected and verified for their integrity by calculating a checksum. If packets are valid, they are passed to the service command processor, which evaluates the protocol command and executes the requested action. If a packet is invalid, the server drops the network input buffer and ignores the request. If the requested service is not part of the protocol specification, the server returns a packet to the client, indicating the request failed. If the client does not currently hold the required minimum privilege level for a command, the server returns a packet to the client, indicating the lack of permission.

Purpose R-USR-002, R-USR-005, R-SWA-003, R-SWA-004

D-SAD-003	Identifier	Function
	Connection Management	The server component implements a network socket listening service, which accepts and sets up newly incoming connections. The total number of connections is determined by a configurable limit. Packets can be sent either to individual clients, or as broadcast messages. In order to prevent blocking, a configuration dependent, finite number of threads are allocated to each connected client. If a connected client exhausts their available threads, they are actively disconnected by the server. This is done so clients which experience network interruptions, or are on very slow connections, do not impair the operation of the server.
Purpose	R-USR-002, R-USR-003, R-USR-004, R-SWA-003, R-SWA-004	

D-SAD-004	Identifier	Function
	User Management	The server component implements three user privilege levels: *watch*, *control*, and *configure*. Of the *control* and *configure* levels, only one can be active at a time, with *configure* overriding *control*. A client can raise their privilege level by sending the proper password. If the equal or lower privilege level is currently active for any other connected client, that client's level will be lowered to *watch*, otherwise the level change request will fail with a message to the requesting client. Passwords are transmitted as a cryptographically hashed message. If there are no active client connections, or no client is in level *control* or above, a newly connecting client will be assigned *control* level by default, otherwise they will be assigned *watch* level.
Purpose	R-USR-004, R-USR-005	

D-SAD-005	Identifier	Function
	Exchange Protocol	An exchange protocol is defined for command and data exchange between server and clients. The detailed protocol specification, description and design of command processing is given later in this document (2.3, 2.4).
Purpose	R-SWA-004	

D-SAD-006	Identifier	Function
	Abstraction Layer	A software layer is made available as an abstraction of the underlying hardware devices by providing a generic API of hardware control functions, which must be implemented in software plugins for the particular hardware backend. The design and specification of the API is given later in this document.
Purpose	R-SWA-005	

2.2.3 Client Design

D-SAD-007	Identifier	Function
	Networking	The client component implements a network interface for communication with a remote server. It operates on a single connection to the server and attempts to automatically reconnect, if the connection is interrupted by external events, such as loss of signal, or disconnect of the underlying physical network connection. The user is also allowed to explicitly command a reconnection event. Packets received via network are inspected and verified for their integrity by calculating a checksum. If packets are valid, they are passed to the packet processor, which evaluates the protocol command and executes the requested action. If a packet is invalid, the client drops the network input buffer and ignores the request. If the requested service is not part of the protocol specification, the client takes no further action and ignores the request.
Purpose	R-USR-002	

D-SAD-008	Identifier	Function
	Privilege Levels	The client component implements an override password input menu, which allows the user to request a change in their control privilege level on the server. The success status of the requested privilege change is evaluated by the user via the server response in the text-based communication interface.
Purpose	R-USR-004	

D-SAD-009	Identifier	Function
	User Communication Interface	The client component provides a text-based communication interface as part of the inter-user communication infrastructure.
Purpose	R-USR-004, R-USR-005	

D-SAD-010	Identifier	Function
	Graphical User Interface	The client component implements a GUI, which provides a selector for subgroups of control and display elements, such as spectral displays and telescope pointing controls. The activation of a selector changes the displayed subgroup. To enable parallel views of the subgroups, the selectors can be used to freely detach, reattach, or reorder the subgroups by user-initiated drag-and-drop action via the input pointer device.
Purpose	R-USR-007	

D-SAD-011	Identifier	Function
	Chat and Log	The client component implements a GUI subgroup which allows display and input of text-based inter-user communication and a log display for showing informative or warning messages emitted by client components relevant to the user. The text-based communications display is implemented in a style reminiscent of internet relay chat (IRC) client software, providing a text entry interface, line-by-line message display and a list of names of connected users.
Purpose	R-USR-007	

D-SAD-012	Identifier	Function
	Sky View	The client component implements a GUI subgroup which displays catalogue objects present in the currently visible sky at the location of the telescope in azimuthal equidistant projection with the zenith as its centre point. The user can interact with the map by selecting objects to be tracked by the telescope and by shifting the projected time of day back and forward. The current pointing and the last requested pointing adjustment of the telescope are marked on the map.
Purpose	R-USR-007	

D-SAD-013	Identifier	Function
	Spectrum Display	The client component implements a GUI subgroup which displays spectral data received from the telescope. The user can select the number of spectra to be displayed concurrently in FIFO order. The user can also set the number of spectra to be displayed as a moving average of FIFO order. The plotted data viewport can be zoomed and moved by the user. To allow basic analysis, a range of data can be selected visually by defining a box within which to fit a Gaussian profile to the contained data.
Purpose	R-USR-007, R-USR-009	

D-SAD-014	Identifier	Function
	Telescope Control	The client component implements a GUI subgroup which exposes an interface to the user that allows them to get and set the desired telescope pointing in a selectable celestial coordinate system. The interface provides further options to interrupt the telescope's movement, recalibrate the zero position reference and to move the telescope to a parking position. Active sky tracking at sidereal speeds can be enabled or disabled by means of a switch.
Purpose	R-USR-006, R-USR-007	

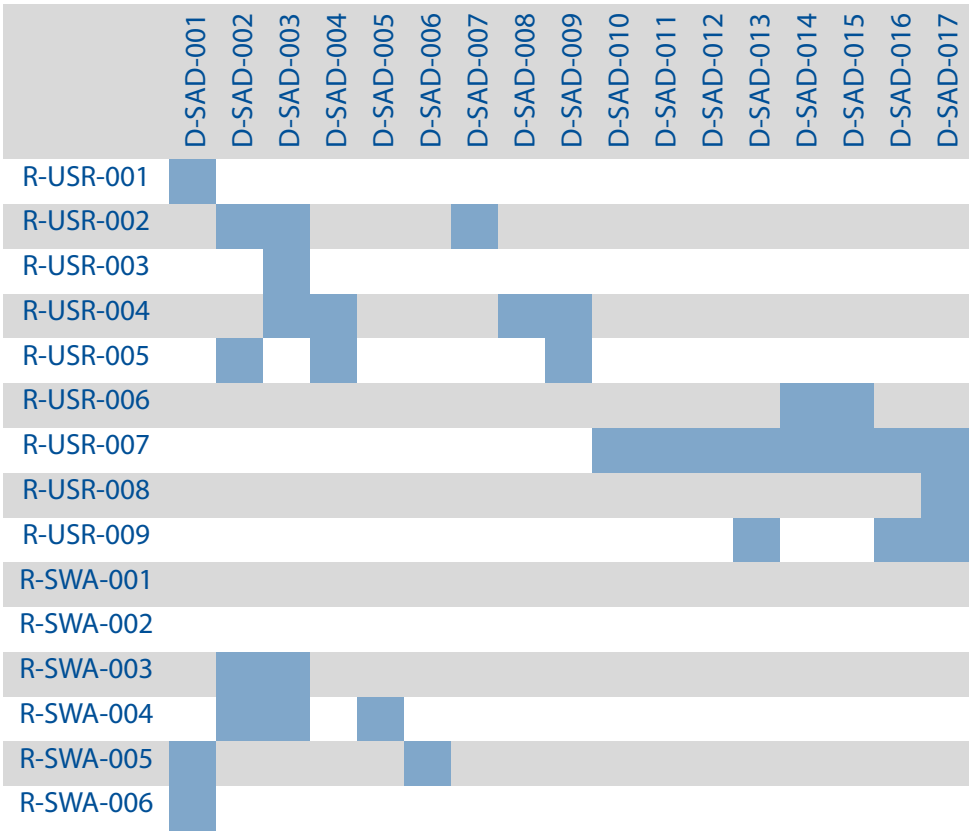
D-SAD-015	Identifier	Function
	Spectrometer Control	The client component implements a GUI subgroup which exposes an interface to the user that allows them to get and set the desired spectrometer settings. The configured spectral range can be set in frequency or in equivalent radial velocity for a configurable reference rest frequency. Ranges can be set by defining a passband by either lower and upper stops, or by centre and bandwidth. The resolution bandwidth of the receiver can be configured if supported by the remote instrument. To compensate for radial velocity effects of the observer, Doppler tracking can be enabled to auto-adjust the centre frequency of the configured bandwidth, as the telescope's pointing changes with reference to a standard of rest. The active acquisition of spectral data by the remote instrument can be controlled by means of a switch. If present in the instrument, hot load can be toggled by means of a switch.
Purpose	R-USR-006, R-USR-007	

D-SAD-016	Identifier	Function
	History Display	The client component implements a GUI subgroup which displays historical spectral data received from the telescope. The integrated value of each of the spectra received is plotted to indicate the continuum flux. A waterfall display of the received spectra is shown, with the oldest spectrum on the bottom of the graph and the newest on top. The user can select the number of samples to be displayed concurrently in FIFO order.
Purpose	R-USR-007, R-USR-009	

D-SAD-017	Identifier	Function
	Observation Programs	The client component implements a GUI subgroup which allows the user to select a series of pre-programmed observation procedures with configurable parameters. When activated, these task the client software with an automated sequence of commands, which collect and display data in a pre-defined manner.
Purpose	R-USR-007, R-USR-008, R-USR-009	

2.2.4 Requirements to Design Components Traceability

Functional requirements are always referenced to their design components, others only as needed, or for clarification. This is reflected in the traceability matrix.



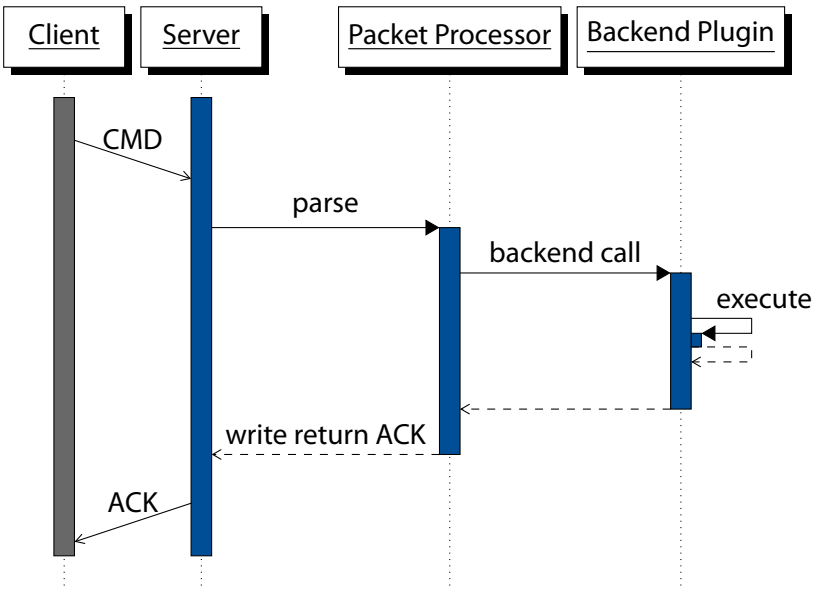


Figure 2.2: The execution profile of synchronous command processing. A service command packet sent by a client application is handed over to the packet processor, which parses the request and immediately executes the requested action on the backend plugin which has implemented and registered that action. A response packet is immediately returned to the client.

2.3 Service Commands

The packet based command and data exchange protocol used in the software for instrument control and monitoring is based on the operational concept of *services*. These services are identified by a unique number and represent self-contained control operations, information requests, or responses.

Examples of control actions are: pointing of the instrument, configuration of the receiver, and execution control of the data acquisition process. These types of commands trigger responses, which either indicate success, failure, or rejection due to lack of control privilege. Information requests include: geographical location of the instrument, current pointing and receiver settings. Responses to these commands are always accompanied by payload data which convey the actual information.

A special type of information request are services which transmit updated states or data from the instrument. These are typically sent in the form of unsolicited responses, which do not require (and may be ignored if used in) active queries. Such services include spectral data transmission, when the receiver is in acquisition mode, or drive status and pointing updates, when the telescope is slewing.

2.3.1 Execution profiles

Service commands fall into two conceptual categories of execution profiles: *synchronous* and *asynchronous* operations. This differentiation is needed in order to avoid stalls in the command processing of the server component, which, as part of its function, must operate attached hardware backends such as telescope drives, spectrometers, or calibration sources. These devices may have long response times, usually in timescales of minutes, and could lead to the apparent unresponsiveness of the server component, particularly when multiple clients were connected simultaneously.

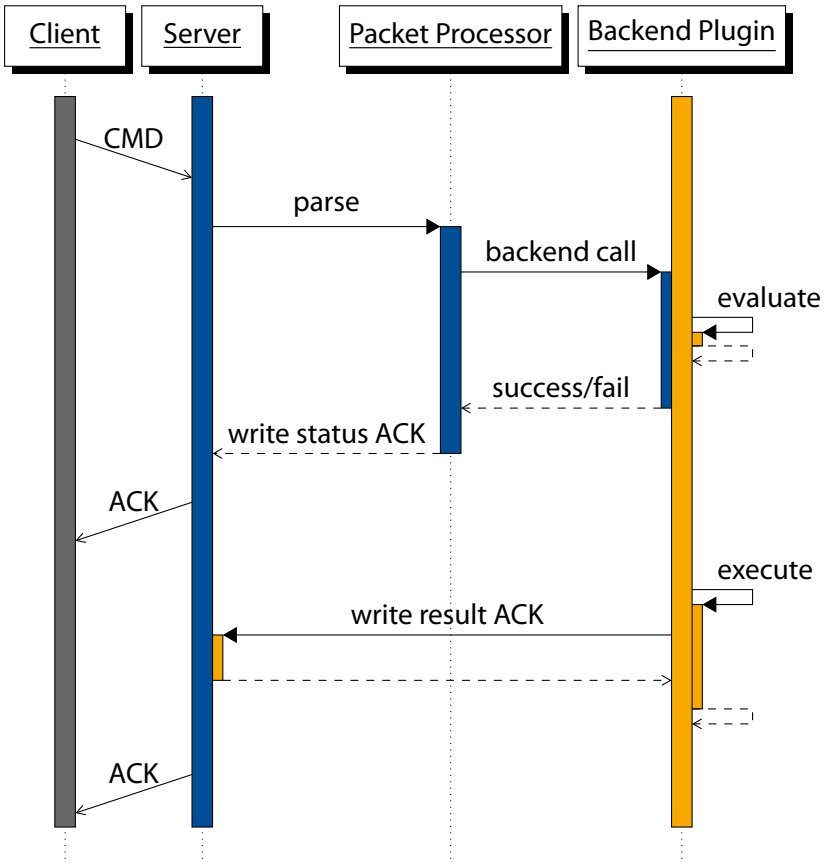


Figure 2.3: The execution profile of an asynchronous command. A service command packet sent by a client application is handed over to the packet processor, which parses the request and initiates the requested action on the backend plugin. The packet processor then immediately responds to the client with the outcome of the request, while the backend starts the process of executing the requested procedure. Once the procedure has completed, the backend independently transmits a second response packet to the client.

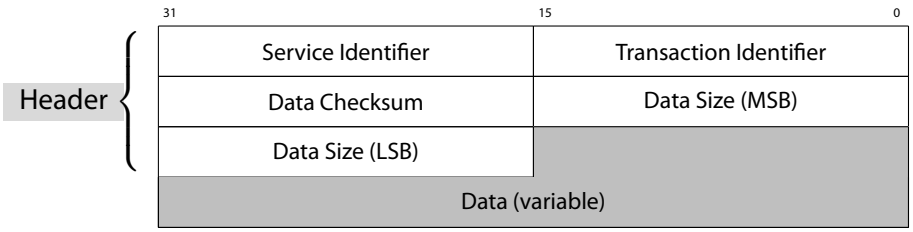


Figure 2.4: The structure of a protocol exchange packet. All fields with the exception of the *Data* field are mandatory. If a packet does not have an associated data payload, the data size field is set to 0 and the CRC16 is set to *0xFFFF*, as demanded by the verification code generator used (see [15]).

Synchronous operations (*fig.* 2.2) are characterised by short execution durations which typically have execution times of at most 1 second, but typically significantly less than that. Purely informative services which request known states, such as pointing information, fall in this group. *Asynchronous* commands (*fig.* 2.3) can take arbitrary lengths of time to complete and thus should not be waited on to finish within an execution cycle of the packet processor. Instead, the selected action is initiated and only the success or failure of the initiating process is returned, with the actual result sent later, when finished. An example for this kind of action is a command to rotate the telescope to another point on the sky.

It is important to note that there is no inherent categorisation of particular service commands. Rather, this lies in the responsibility of the design of a backend plugin. Since asynchronicity typically requires logic which requires significant more effort to design, it is reasonable to implement any action synchronously, as long as the response is acceptably fast, i.e. available in near real-time.

2.4 Command and Data Exchange Packet Format

The command and data exchange protocol used in the network layer employs a packet structure (*fig.* 2.4) with a fixed size header, followed by an arbitrary sized data block, which may hold a payload structure that is defined by the service type of the packet. Fields in the header shall be in network (*big-endian*) byte order, while fields in the data payload shall be in *little-endian* order. This choice is made deliberately, as virtually all target platforms for the software are little-endian based, or are capable of operating in both modes, thus avoiding potentially expensive byte-swapping operations of long data blocks for the bulk of users.

2.4.1 Packet Structure Members

Service Identifier (16 bits)

The service identifier field holds the numeric protocol ID of a service command.

Transaction Identifier (16 bits)

The transaction identifier may be set to an arbitrary value by the client component to track the status of requests and responses by the server component. The server shall always copy the transaction identifier from the request packet to the response packet.

If transactions are not tracked in the client, this field shall have all bits set (*0xFFFF*) to mark the transaction identifier as undefined.

Unsolicited packets sent by the server component to connected clients shall always mark the transaction identifier as undefined.

Data Checksum (16 bits)

The data checksum field shall be calculated in accordance with the specification of the cyclical redundancy code (CRC) in *ECSS-E-70-41A* Annex *A.1* (see [15]).

Data Size (32 bits)

The data size field shall specify the number of octets of payload data attached to the packet header.

Data (variable)

The data field is optional and may be used to transport payload data by a service command. The size of the data in the data field shall be signed or unsigned integer types and an integer multiple of one octet wide.

2.5 Service Identifiers

The services used in the software are assigned a unique arbitrary identification number. The numerical identifiers reference the type of action and any optional associated payload data. If a payload structure of a service command is added or changed in later versions of the protocol, it shall be assigned a new and previously unused unique numerical identifier. This is done so backwards compatibility with older protocol version can be maintained. An example is PR_CAPABILITIES_LOAD, which is an extension added to PR_CAPABILITIES, so the use of hot loads could be integrated into the software while maintaining backwards compatibility for older client versions.

Purpose and payload structure of services are described on the following pages in numerical order of their service identifiers. Note that the terms *request* and *response* are subsequently used to refer to *any* packets sent by a client for the former, and packets sent by the server for the latter. A list of services, assigned numerical identifiers, and their function is given below.

Service Short Name	Service ID	Service Function
PR_INVALID_PKT	0xA001	invalid packet signal
PR_CAPABILITIES	0xA002	capabilities of the telescope
PR_CONTROL	0xA004	request control of the telescope
PR_MOVETO_AZEL	0xA005	move to azimuth/elevation
PR_SUCCESS	0xA006	last command succeeded
PR_FAIL	0xA007	last command failed
PR_RECAL_POINTING	0xA008	recalibrate telescope pointing
PR_PARK_TELESCOPE	0xA009	park telescope
PR_SPEC_ACQ_CFG	0xA00A	set spectrum acquisition configuration
PR_SPEC_DATA	0xA00B	spectral data
PR_GETPOS_AZEL	0xA00C	get azimuth/elevation
PR_SPEC_ACQ_ENABLE	0xA00D	enable spectrum acquisition
PR_SPEC_ACQ_DISABLE	0xA00E	disable spectrum acquisition
PR_SPEC_ACQ_CFG_GET	0xA00F	get spectrum acquisition configuration
PR_STATUS_ACQ	0xA010	spectrum acquisition status
PR_STATUS_SLEW	0xA011	drive slew status
PR_STATUS_MOVE	0xA012	drive movement (to target) status
PR_STATUS_REC	0xA013	(full) spectrum recording status
PR_NOPRIV	0xA014	lack of privilege
PR_MESSAGE	0xA015	arbitrary text message
PR_USERLIST	0xA016	list of connected users
PR_NICK	0xA017	set the user's nickname
PR_CAPABILITIES_LOAD	0xA018	PR_CAPABILITIES, extended for hot loads
PR_HOT_LOAD_ENABLE	0xA019	enable hot load
PR_HOT_LOAD_DISABLE	0xA01A	disable hot load

2.5.1 Invalid Packet

Short Name

PR_INVALID_PKT

Purpose

Sent in response to any malformed request. This can be triggered by events such as invalid header size or format, invalid data checksum, or invalid service command.

Payload

None.

Privileges

Not applicable.

2.5.2 Capabilities

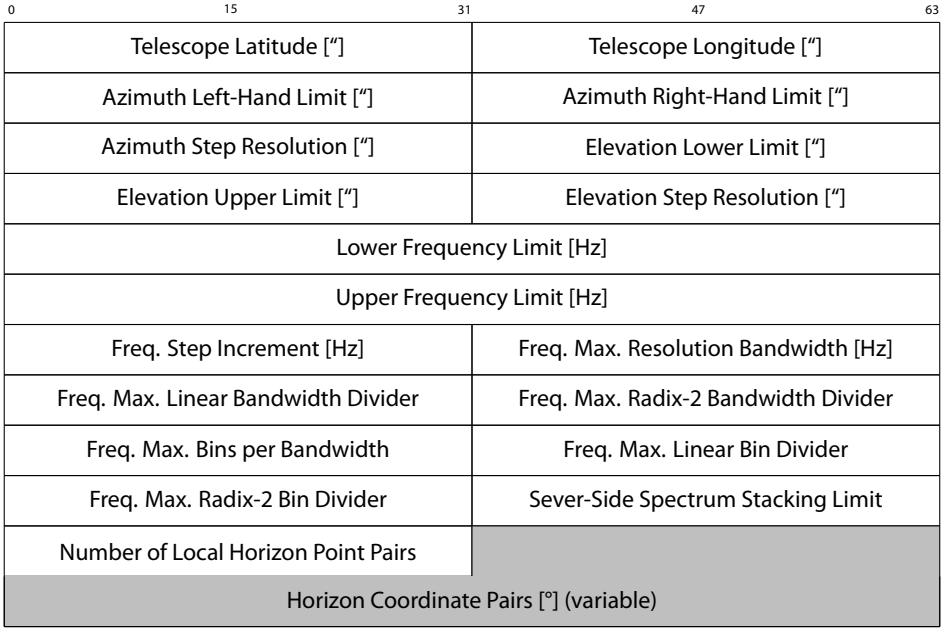


Figure 2.5: The Capabilities payload structure. The frequency limits of the spectrometer are 64 bit integer values, all others are 32 bits wide. The optional profile of the local horizon of the telescope is given in a series of point pairs in azimuth and elevation as 32 bit integers.

Short Name

PR_CAPABILITIES

Purpose

This service conveys parameters which characterise the instrument. These include geographical location, drive and spectrometer parameters, such as configurable movement limits and the tunable frequency range and resolution.

Payload

Requests: None.

Responses: see PR_CAPABILITIES_LOAD for a description of the payload. That extension supersedes PR_CAPABILITIES, which is maintained for backwards compatibility only.

Privileges

Not applicable.

2.5.3 Control

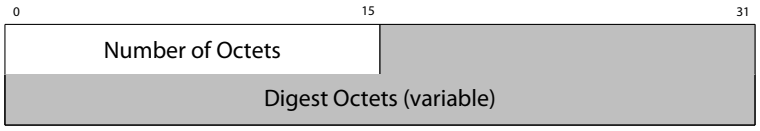


Figure 2.6: The structure of the Control request packet.

Short Name
PR_CONTROL

Purpose
Requests a privilege escalation of an unprivileged client connection for control of the instrument's drive and spectrometer settings.

Payload
Requests: The structure and contents of the response payload is shown in *fig.* 2.6. The transported digest is an HMAC-256.

Responses: None, service is never used for responses.

Privileges
None required.

2.5.4 Move to Azimuth/Elevation

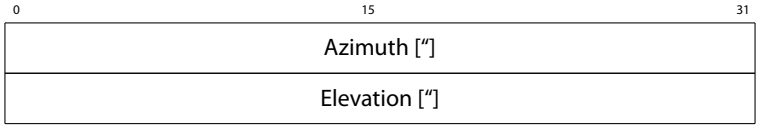


Figure 2.7: The structure of the Move to Azimuth/Elevation packet payload.

Short Name
PR_MOVETO_AZEL

Purpose
This service conveys a rotation request to the instrument to angular coordinates in the horizon system.

Payload
Requests: The structure and contents of the response payload is shown in *fig.* 2.7. Azimuth and elevation are signed integers in units of *arcseconds*.

Responses: The structure is the same as in the request form. In reply form, the packet is broadcast by the server to all connected clients without solicitation to indicate the next pointing target.

Privileges
Client connection must be in control mode.

2.5.5 Command Succeeded

Short Name
PR_SUCCESS

Purpose
Sent in response to any request packet which does not have a specific, immediate reply in another format, to indicate the successful execution of the request. This type of packet responds with the transaction identifier specified in the request.

Payload
None.

Privileges
Not applicable.

2.5.6 Command Failed

Short Name
PR_FAIL

Purpose
Sent in response to any request packet which does not have a specific, immediate reply in another format, to indicate the failed execution of the request. This type of packet responds with the transaction identifier specified in the request.

Payload
None.

Privileges
Not applicable.

2.5.7 Recalibrate Pointing

Short Name
PR_RECAL_POINTING

Purpose
Requests a recalibration procedure to be executed, if the telescope's drive is suspected to be in misalignment. The telescope rotates back to the last requested target pointing when the procedure is finished.

Payload
None.

Privileges
Client connection must be in control mode.

2.5.8 Park Telescope

Short Name

PR_PARK_TELESCOPE

Purpose

Requests that the telescope is moved to its parking position.

Payload

None.

Privileges

Client connection must be in control mode.

2.5.9 Spectrometer Acquisition Configuration

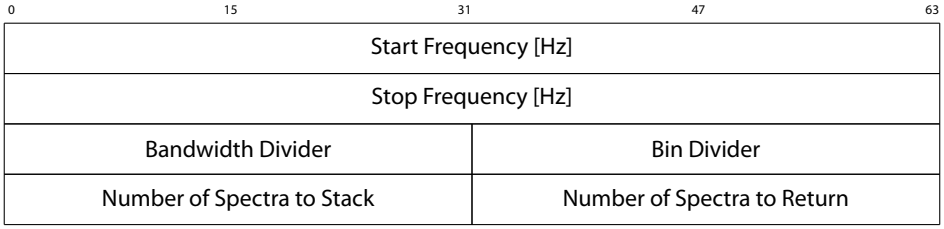


Figure 2.8: The structure of the Spectrometer Acquisition Configuration packet payload. All values are treated as unsigned integers.

Short Name

PR_SPEC_ACQ_CFG

Purpose

This service conveys the spectrometer acquisition configuration.

Payload

Requests: The structure and contents of the response payload is shown in *fig.* 2.8. Start and stop frequencies are unsigned integers in units of *Hz*. If set to a value greater than 0, acquisition will pause once the configured number of spectra have been transmitted to the clients, otherwise acquisition will continue indefinitely. The server-side stacking of spectra can be set to the maximum value allowed by the configuration (2.5.23). If the number of spectra to stack is set to either 0 or 1, no stacking is done. For details on how the spectrometer configuration is generalised, refer to 2.5.26.

Responses: The structure is the same as in the request form. In reply form, the packet is broadcast by the server to all connected clients without solicitation, unless a request is made via 2.5.14, in which case only the requesting client is informed. In such a case, this type of packet responds with the transaction identifier specified in the request.

Privileges

Client connection must be in control mode.

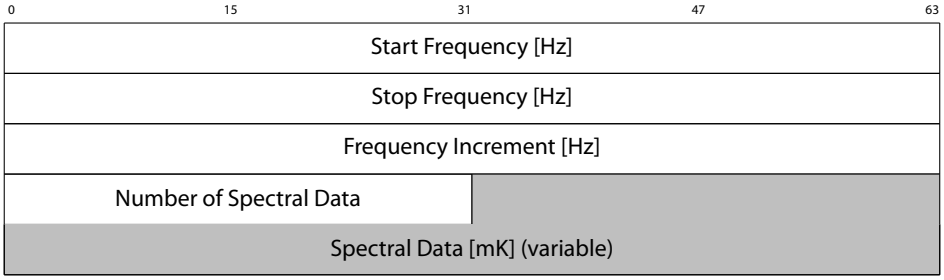


Figure 2.9: The structure of the Spectral Data payload packet. Frequencies are in units of *Hz* and treated as unsigned integers, spectral data are transported as equivalent temperature in *mK* as 32 bit integers.

2.5.10 Spectral Data

Short Name

PR_SPEC_DATA

Purpose

This service transfers acquired spectral data to clients.

Payload

Requests: None, service is never used for requests.

Responses: The structure and contents of the payload are shown in *fig.* 2.9. The frequency range bounds and the step increment are unsigned integers in units of *Hz*. The spectral data are transported as calibrated equivalent temperature in units of *mK*.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients if the spectrometer is enabled.

Privileges

Not applicable.

2.5.11 Get Current Azimuth/Elevation

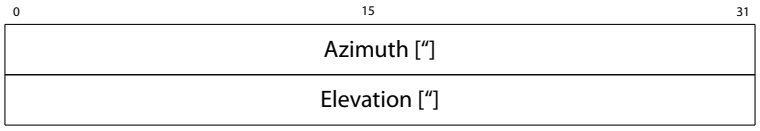


Figure 2.10: The structure of the Get Current Azimuth/Elevation packet payload.

Short Name

PR_GETPOS_AZEL

Purpose

This service conveys the last known angles of the instrument in the horizon system.

Payload

Requests: None.

Responses: The structure and contents of the payload are shown in *fig.* 2.10. Azimuth and elevation are signed integers in units of *arcseconds*.

The packet is broadcast by the server to all connected clients without solicitation during any position updates following a preceding move request (2.5.4) and sent directly to a client if prompted by a request packet. In such a case, this type of packet responds with the transaction identifier specified in the request.

Privileges

None required.

2.5.12 Enable Spectrum Acquisition

Short Name

PR_SPEC_ACQ_ENABLE

Purpose

Requests: enables spectrum acquisition on the instrument.

Responses: informs clients that spectrum acquisition has been enabled.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current status.

Payload

None.

Privileges

Client connection must be in control mode.

2.5.13 Disable Spectrum Acquisition

Short Name

PR_SPEC_ACQ_DISABLE

Purpose

Requests: disables spectrum acquisition on the instrument.

Responses: informs clients that spectrum acquisition has been disabled.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current status.

Payload

None.

Privileges

Client connection must be in control mode.

2.5.14 Spectrometer Acquisition Configuration Request

Short Name

PR_SPEC_ACQ_CFG_GET

Purpose

Get the current spectrum acquisition configuration.

Payload

Requests: None.

Responses: None, service is never used for responses.

Note: a service request triggers a response of type PR_SPEC_ACQ_CFG, see 2.5.9.

Privileges

None required.

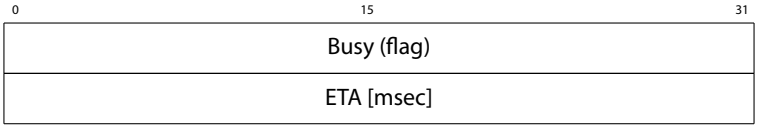


Figure 2.11: The structure of a Status payload packet. The busy field is a flag to convey information for the particular action. If set to any value other than *0*, activity is indicated. If busy is set, the ETA field holds the estimated remaining time in *msec* until the activity is finished.

2.5.15 Spectrum Acquisition Status

Short Name

PR_STATUS_ACQ

Purpose

This service indicates the current status of spectral acquisition activity. Note that the purpose of this service is to convey information about busy status within a full spectrum recording cycle (2.5.18).

Payload

Requests: None, service is never used for requests.

Responses: The structure, contents, and function of the payload are shown in *fig.* 2.11.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current activity status.

Privileges

Not applicable.

2.5.16 Drive Slew Status

Short Name

PR_STATUS_SLEW

Purpose

This service indicates the current status of drive slewing activity. Note that the purpose of this service is to convey information about busy status within a drive movement cycle (2.5.17).

Payload

Requests: None, service is never used for requests.

Responses: The structure, contents, and function of the payload are shown in *fig.* 2.11.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current activity status.

Privileges

Not applicable.

2.5.17 Drive Move Status

Short Name

PR_STATUS_MOVE

Purpose

This service indicates the status of a drive movement cycle. Note that the purpose of this service is to convey information about a single move to a new pointing, as commanded by a PR_MOVETO_AZEL request (2.5.4). If the movement cycle is performed in partial steps, these shall be indicated by PR_STATUS_SLEW messages (2.5.16).

Payload

Requests: None, service is never used for requests.

Responses: The structure, contents, and function of the payload are shown in *fig.* 2.11.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current activity status.

Privileges

Not applicable.

2.5.18 Spectrometer Recording Status

Short Name

PR_STATUS_REC

Purpose

This service indicates the status of a spectrum recording cycle. Note that the purpose of this service is to convey information of the recording of a full spectrum that is the transmitted to the clients. If the recording cycle is performed in partial steps, these shall be indicated by PR_STATUS_ACQ messages (2.5.15).

Payload

Requests: None, service is never used for requests.

Responses: The structure, contents, and function of the payload are shown in *fig.* 2.11.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current activity status.

Privileges

Not applicable.

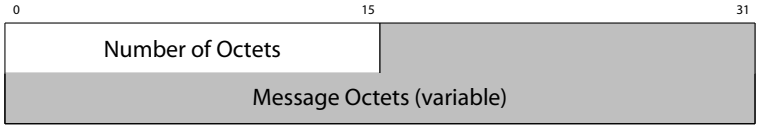


Figure 2.12: The structure of a string transport packet.

2.5.19 Lack Of Privilege

Short Name

PR_NORPIV

Purpose

Sent in response to any request packet which requires a form of elevated privilege of the client connection, if that condition is not fulfilled.

Payload

Requests: None, service is never used for requests.

Responses: None.

Privileges

Not applicable.

2.5.20 Text Message

Short Name

PR_MESSAGE

Purpose

Transports arbitrary message strings to be displayed in the text-based client inter-user communication and logging interface.

Payload

Requests: The structure and contents of the payload are shown in *fig.* 2.12.

Responses: Same as request.

Privileges

None required.

2.5.21 User List

Short Name
PR_USERLIST

Purpose
Transports a list of nicknames of currently connected users.

Payload
Requests: None, service is never used for requests.

Responses: The structure and contents of the payload are shown in *fig.* 2.12.

Privileges
None required.

2.5.22 User Nick Name

Short Name
PR_NICK

Purpose
Transports a user's nickname to be associated with a client connection.

Payload
Requests: The structure and contents of the payload are shown in *fig.* 2.12.

Responses: None, service is never used for responses.

Privileges
None required.

2.5.23 Capabilities: Hot Load Extension

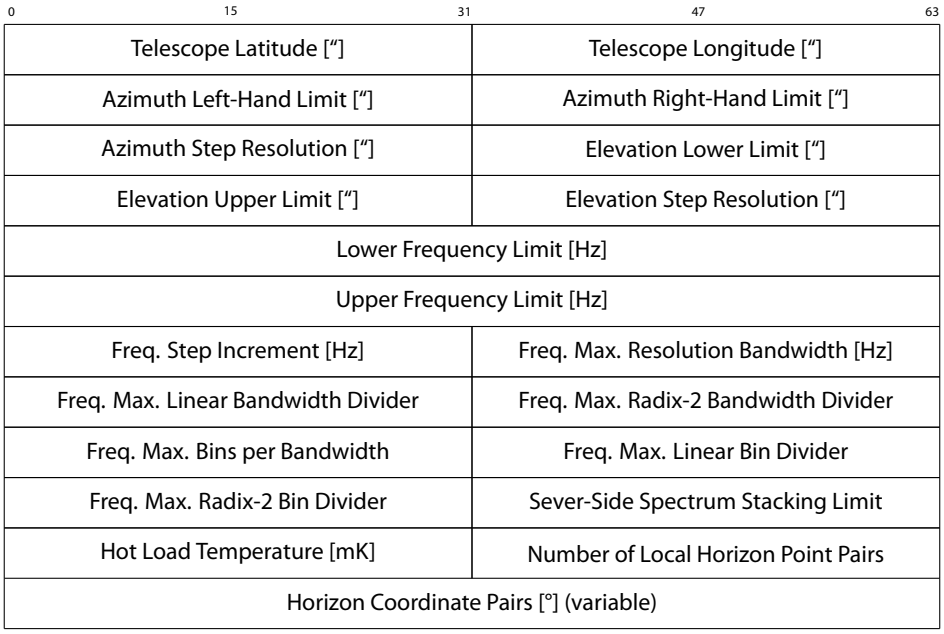


Figure 2.13: The Capabilities payload structure, extended for hot loads. The frequency limits of the spectrometer are 64 bit integer values, all others are 32 bits wide. The optional profile of the local horizon of the telescope is given in a series of point pairs in azimuth and elevation as 32 bit integers.

Short Name

PR_CAPABILITIES_LOAD

Purpose

This service conveys parameters which characterise the instrument. These include geographical location, drive and spectrometer parameters, such as configurable movement limits, the tunable frequency range and resolution, and the temperature of a switchable hot load. This is an extension of PR_CAPABILITIES_LOAD.

Payload

Requests: None.

Responses: The structure and contents of the response payload are shown in *fig.* 2.13. All fields holding values in angular units are given in *arcseconds*, with the exception being the value pairs describing the local horizon of the telescope, which are in degrees and part of the dynamic size of the payload structure. All frequency values regarding the spectrometer are given in units of *Hz*. The temperature value of the hot load is given in units of *mK*. A hot load temperature of 0 mK indicates the absence of such a device. All angular values are treated as signed integers, all frequency related values are treated as unsigned integers. All remaining fields are treated as unsigned integers.

For details on how the spectrometer is characterised, refer to 2.5.26.

Privileges

Not applicable.

2.5.24 Enable Hot Load

Short Name

PR_HOT_LOAD_ENABLE

Purpose

Requests: enable a hot load on the instrument.

Responses: informs clients that a hot load has been enabled.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current status.

Payload

None.

Privileges

Client connection must be in control mode.

2.5.25 Disable Hot Load

Short Name

PR_HOT_LOAD_DISABLE

Purpose

Requests: disable a hot load on the instrument.

Responses: informs clients that a hot load has been disabled.

This type of packet is always transmitted in an unsolicited broadcast to all connected clients to indicate current status.

Payload

None.

Privileges

Client connection must be in control mode.

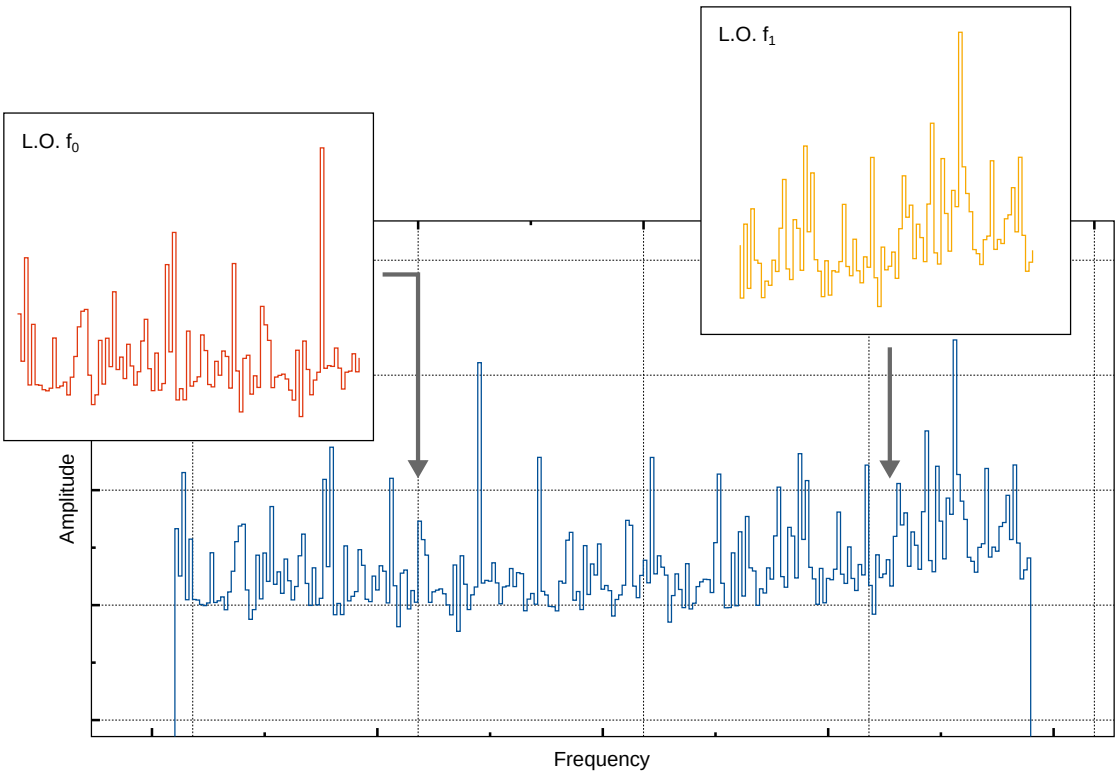


Figure 2.14: Spectra delivered to clients can be cut out of a fixed-width passband, if a smaller section is selected. Should the selected frequency range be larger than the passband, the delivered spectrum can be constructed from sequentially recorded and concatenated spectra, if the local oscillator frequency can be adjusted.

2.5.26 Drive and Spectrometer Characterisation

Telescope drive mechanics and radio spectrometers come in many varieties. In order to generalise the unique properties of the various configurations that may be used with radio telescopes, a set of characteristic parameters (2.5.2) was chosen which should be sufficient to describe most setups that are likely encountered in the type of device targeted by this software. This is needed so that a consistent, unified control interface can be presented to the user, regardless of the specifics of the connected instrument.

While equatorial mounts may be used with small radio telescopes, this type is not given consideration, as azimuth/elevation drives are much more likely due to their easier construction and setup. For this type of mount, the *angular limits* and *step resolution* is a sufficient description. Should the need arise to create a backend plugin for an equatorial mount, it must be described in the horizon coordinate system. It has to be noted however, that in this situation, the configuration of the step resolution parameter must be carefully considered, given the rotation between the coordinate systems. Alternatively, an extension to the existing protocol and backend API can be defined.

The characterisation of a radio spectrometer is more complex. For the purpose of the software it is however sufficient to consider the intermediate frequency output band of the receiver, regardless of the technical implementation (which will typically be based on the superheterodyne principle). It is immediately obvious that the total output band will be primarily defined by an *upper* and *lower frequency stop* and a *frequency increment*, if a sub-band range can be selected from the output band. Furthermore, a receiver will have a limit to the *bandwidth*, given its computational or physical input filters. Since the spectrum is encoded as a sequence of digital samples, the maximum number of *data bins (samples) per bandwidth segment* is also needed. In conjunction with the bandwidth, this gives the frequency resolution

per spectral sample. If the bandwidth can be adjusted, a range for *linear* or *radix-2* based dividers can be specified. The latter covers the needs of fast Fourier transform based instruments, while the former is added to provide support for receivers with selectable physical filter banks. Identical fields are reserved for adjustable bins-per-bandwidth settings.

Note that the actual implementation of the receiver and recording of spectra is up to the creator of a backend plugin. If, for example, a system is based on a fixed frequency local oscillator with a wide bandpass, only the configured segments of the spectrum would be transferred to the clients. On the other hand, if a spectrometer is capable of adjusting the local oscillator frequency, but only has a small frequency bandpass, the local oscillator frequency can be shifted and a wider spectrum can be constructed from a series of concatenated subspectra, which were recorded sequentially (*fig.* 2.14).

2.6 Backend Plugin Calls

In order to operate the underlying hardware, the server software component implements a plugin module system which allows object code to be linked into the executable at runtime. This requires a set of predefined abstract interface calls, which must be implemented in the plugin, so the server can interact with hardware functions from the command protocol interface.

A backend call can only be registered to a single implementation, with the latest registration given precedence. This means that if multiple plugins are loaded in sequence and more than one have implemented a certain call, the call implemented in the last loaded plugin will be the call used, and all others are ignored. If a backend call is not implemented in any plugin, or no plugins are loaded, the functionality is not available and an attempt to call the backend will always result in an error code, which in turn will lead to a notification packet of *Command Failed* by the command processor to the requesting client. The list of calls with their symbolic names and purpose is given below. The actual definition of the API is left to the implementation.

Get Drive Capabilities
: Retrieves the parameter set describing the instrument's drive mount in a manner compatible with the payload description given in 2.5.23.

Get Spectrometer Capabilities
: Retrieves the parameter set describing the instrument's receiver system and spectrometer in a manner compatible with the payload description given in 2.5.23.

Move to Azimuth/Elevation
: Commands the drive to rotate the instrument to the specified horizon system coordinates.

Get Azimuth/Elevation
: Reads back the instruments last known rotation in horizon system coordinates.

Recalibrate Pointing
: Initiates a recalibration procedure on the telescope's drive.

Park Telescope
: Moves the instrument to its parking position.

Spectrum Acquisition Configuration
: Sets the acquisition configuration of the receiver. The parameters accepted by the plugin must be compatible with the payload structure given in 2.5.9.

Get Spectrum Acquisition Configuration
Gets the current acquisition configuration of the receiver. The returned parameter set corresponds to the payload structure given in 2.5.9.

Spectrum Acquisition Enable
Enables or disables spectrum acquisition. Accepts a boolean flag (0: disable, 1: enable).

Shared Comlink Acquire
This function can be implemented if the hardware control unit shares a common link to the host computer, which must guarantee mutually exclusive access, for example on serial links with a shared hardware command processor.

This function actively waits until the link is acquired and locks access to it, so other software components must also actively wait until it is released. This call must not be used by software components other than backend plugins.

Shared Comlink Release
This function returns control of a previous acquired shared communication link. It has no effect if no lock on a link is held.

Shared Comlink Write
Writes a series of octets to a communication link. The link must be acquired beforehand, otherwise the behaviour of this function is undefined.

Shared Comlink Read
Reads a series of octets from a communication link. The link must be acquired beforehand, otherwise the behaviour of this function is undefined.

Hot Load Enable
Enables or disables a hot load, if present. Accepts a boolean flag (0: disable, 1: enable).

2.7 Implemented Backends: Haystack SRT (CASSI Kit)



Figure 2.15: The Small Radio Telescope as operated by the Department of Astrophysics at the Vienna University Observatory. (Photos courtesy of H. Chmela)

2.7.1 Overview

From about 1998 onwards (see [16]), MIT's Haystack Observatory developed the Small Radio Telescope (SRT) as a teaching and training tool for the introduction of undergraduate students to the field of radio astronomy. In its default configuration, it is capable of spectral line and continuum observation in the L-Band between 1400 and 1440 MHz, making it suitable for the observation of galactic neutral hydrogen.

The SRT was commercially available as a kit from CASSI Corporation (now defunct). According to the Haystack's SRT website, different configurations of the SRT existed over the years. This implementation only covers the later model using the CASSI mount equipped with the digital receiver model.

2.7.2 Antenna and Mount

The antenna is made of a standard television dish with a diameter of 2.3 meters (7.5 feet) and an f/D of 0.4, constructed from subsections of C/Ku-band grade expanded aluminium mesh frames, supported by an AZ/EL mount (*fig.* 2.15). Given the diameter and operating frequency band, the angular resolution (1.3.5) of the dish is about 6.3 degrees with a power gain (1.3.6) of about 28 dBi. The holes in the surface mesh are approximately 2 mm wide, which gives a very low surface error loss (1.3.7) of no worse than 0.7 dBi, thus providing a suitably smooth surface for the intended application. The diameter of the (upgraded) feed is about 25 cm (including the mounting ring), giving the instrument a blocking fraction (1.3.10) of $\beta \approx 0.1$. The feed has been measured to have an edge taper of -25 dB [17], which corresponds to a Hansen parameter h=4.5 and thus to a theoretical maximum aperture efficiency (1.3.11) of $\eta_{\mathrm{ap}} \approx 0.65$. The CASSI mount allows for a maximum pointing accuracy of about 0.25° in azimuth and about 0.1° in elevation. These values are defined by the reed-switch sensor pulses per degree driven for a given motor axis.

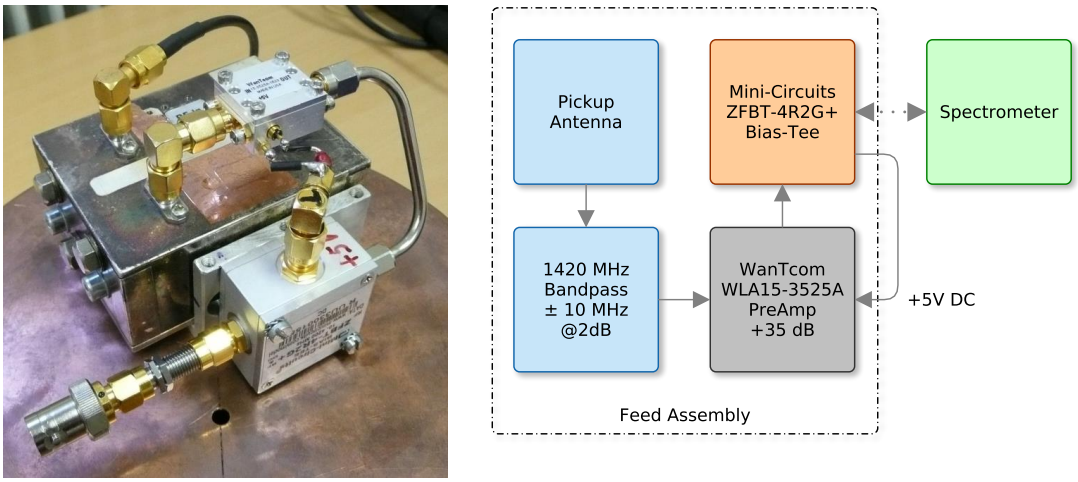


Figure 2.16: The signal from the feed's pickup antenna is filtered by a two-cavity bandpass and amplified in the WanTcom WLA15-3525A. Power is supplied by a Mini-Circuits ZFBT-4R2G+ bias-tee. The block diagram of the setup is on the right, the upgraded frontend assembly is on the left. (Photo courtesy of H. Chmela)

2.7.3 Receiving System

The prime feed antenna supplied with the original CASSI kit was a modified commercial C-band feed, as used for satellite TV reception. Its pickup antenna was a quarter-wavelength sized exposed core of a semi-rigid coaxial cable that was not optimally impedance-matched to the target frequency, thus resulting in unnecessary power loss. Intermodulation products in the preamplifier from local out-of-band emissions were a problem due to improper input filter design. While this feed was used successfully for many years, unwanted interference was becoming an issue with the increasing prevalence of electronic devices operating in high-frequency bands since the early 2010s. It was eventually replaced with a purpose-built circular waveguide antenna [17], which also introduced a two-cavity bandpass filter in the signal path between the pickup antenna and the upgraded preamp stage (a WanTcom WLA15-3525A), which efficiently suppresses all undesired out-of-band signals (*fig.* 2.16). The very low-noise 35 dB preamplifier is specified for a noise figure of 0.3 dB (0.45 dB max) in the 1.42 GHz band. For the Mini-Circuits MAR-6SM amplifier at the spectrometer input, the noise figure is 2.2 dB for a 19 dB gain at 1.42 GHz. This amounts to a cascaded baseline noise figure (1.3.2) in the updated configuration of only 0.3 dB (0.45 dB max), equivalent to a contribution of less than 22 K (32 K max) of noise temperature by the amplifier chain.

In the spectrometer assembly (*fig.* 2.17), the input signal is down-converted in an image reject mixer which is driven by a voltage-controlled local oscillator that can be digitally programmed in 40 kHz increments. The intermediate frequency, with the centre at 800 kHz, is filtered by a 500 kHz bandpass to suppress aliasing effects of high-frequency components present after the the 8-bit Analog to Digital Converter (ADC) sampling process. The digital samples are down-converted by the GC1011A digital receiver and accumulated in a Digital Signal Processor (DSP) for a total of 4096 64-byte sized blocks, which are Fourier-transformed and squared to get the power spectrum. The result is then transferred to a computer via a serial interface as 64 unsigned 16-bit integers.

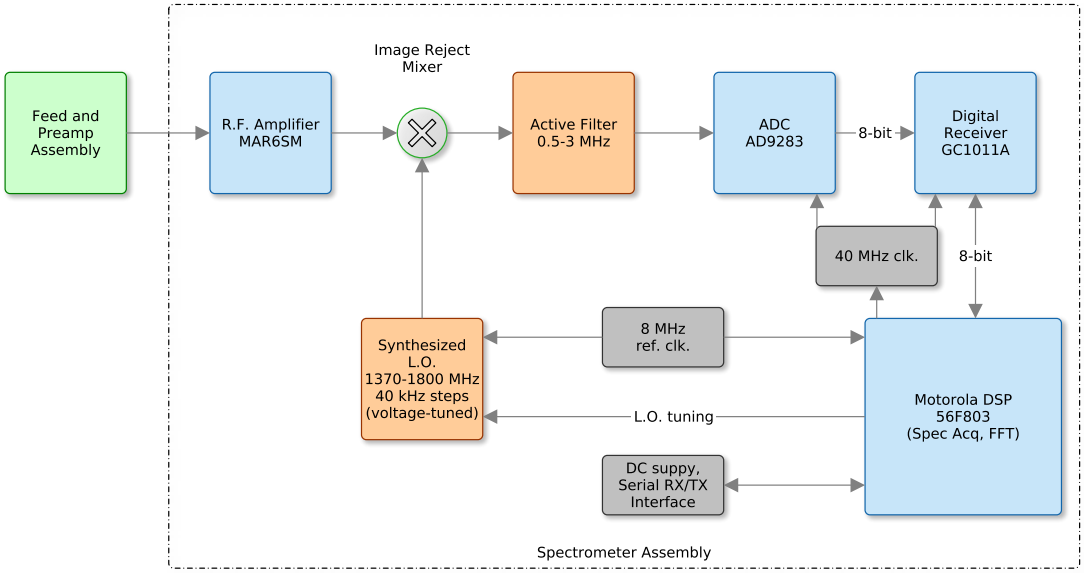


Figure 2.17: Block diagram of the spectrometer assembly

2.7.4 Calibration Device

The electronic noise calibrator of the SRT was not implemented in the control software in a self-contained calibration procedure, but as a hot load which can be turned on arbitrarily to verify the server-configured temperature calibration. This is justified by the observation that spurious, strong, narrow-band interference may still occasionally be present in the filtered signal due to unpredictable variations in the modern local radio environment. An automated, quick calibration procedure would be difficult to implement, and also hard to verify by an untrained user. This is because the originally used calibration procedure (see below) relies on a comparison of continuum flux, but the noise source may not reliably provide a dominant enough signal to consistently overpower the interference. Instead, the software uses a configurable calibration factor, which can be determined by observation of a source with a sufficiently strong, known radio flux (typically the Sun) or by using a matched termination (1.4.2) at the prime feed filter input. The updated feed setup of the SRT shows to be very stable with temperature, so once calibrated, the observed long-term daily drift is typically less than the variation in local radio noise.

The following description of the original calibration procedure using the electronic noise calibrator is given for reference only.

Older models of the SRT used a motorized ambient temperature vane calibrator to determine the system temperature. It was made of an absorber material, radiating at a blackbody equivalent temperature of $\sim$ 300K. Its stability however suffered from environmental conditions (water, dirt) and reflection of LNA noise within the cavity formed, when the vane was in front of the feed. [18] To overcome these problems, an active calibration source was designed based on a Noise/Com NC302L broadband noise diode. This diode is characterised by symmetric white Gaussian noise with a flat output power vs. frequency in a 10 Hz to 3 GHz range through a controlled *avalanche effect*.

Noise Diode Calibration Procedure

The calibration procedure using the electronic noise generator relies on the ratios of power measurements with the noise calibrator enabled and disabled. Both measurements contain signal from the amplifier electronics, the sky, and antenna spillover, while the power (or temperature equivalent) of the diode is a known constant.

The ratio of the measured signals is given by

$$\frac{P_{\mathrm{on}}}{P_{\mathrm{off}}} = \frac{T_{\mathrm{rec}} + T_{diode} + T_{\mathrm{spill}} + T_{\mathrm{sky}}}{T_{\mathrm{rec}} + T_{\mathrm{spill}} + T_{\mathrm{sky}}} ,$$

with the receiver temperature

$$T_{\mathrm{rec}} = \left[\left(\frac{P_{\mathrm{on}}}{P_{\mathrm{off}}} - 1 \right) (T_{\mathrm{spill}} + T_{\mathrm{sky}}) - T_{diode} \right] \left(1 - \frac{P_{\mathrm{on}}}{P_{\mathrm{off}}} \right)^{-1}$$

and the system temperature

$$T_{\mathrm{sys}} = T_{\mathrm{rec}} + T_{\mathrm{spill}} + T_{\mathrm{sky}} ,$$

where $T_{diode} = 200K$, $T_{\mathrm{sky}} = 3K$, and an assumed $T_{\mathrm{spill}} = 20K$.

Finally, the conversion constant g from electronic units (as reported by the spectrometer) to temperature is calculated from

$$g = \frac{T_{\mathrm{rec}} + T_{\mathrm{spill}}}{P_{\mathrm{off}}} \quad .$$

2.7.5 Hardware Interface

The hardware spectrometer and the drive controller share the same input-output pins of the LT1181ACSW RS232 driver/receiver, which interfaces to both the BASIC Stamp 1 controller responsible for the drive unit, and the 56F803 processor of the spectrometer hardware. The digital signal is transported to the latter via modulation of its DC supply voltage. The RS232 connection operates at 2400 baud with 8-N-1 framing. The mode of operation is selected by a keyword, followed by a series of octets containing the instruction. Once the instruction is executing, the selected unit will ignore any other requests until the operation has finished.

ID	Function
0	move clockwise azimuth (right)
1	move counter-clockwise azimuth (left)
2	move clockwise elevation (down)
3	move counter-clockwise elevation (up)
4	move calibration vane out
5	move calibration vane in
6	switch noise diode off
7	switch noise diode on

Table 2.2: List of *move* commands and responses. Note that command IDs 4 and 5 are intended for the calibration vane present in models without a noise diode. They use the same electrical outputs as 6 and 7, but in a different timed sequence, in order to rotate the vane motor.

2.7.6 Drive Unit Operation

The drive unit is addressed via the keyword *move*, which controls the motors and the calibrator, by sending the ASCII-encoded string

move <*ID*> <*CNTS*>

over the serial connection.

The *ID* field specifies the desired operation (*tbl.* 2.2), while *CNTS* is the number of pulses to be counted on the selected drive's axis rotation sensor. The operation of the calibration device requires this field to be transmitted, but it may be set to *any value*.

When driving an axis motor, the controller keeps the selected motor powered on, counting impulses from a reed-switch sensor, until the specified number of turns has been reached, or an end switch or pulse timeout is triggered. For the calibration device operation, the controller will either respond immediately, if a noise diode command is issued for later SRT models, or after a fixed delay needed to drive the calibration vane in earlier models.

For successful motor commands, the controller will respond with a string

M <*CD*> <*CFR*> <*ID*>,

where *CD* denotes the number of counts driven. For older firmware, *CFR* gives the remaining counts (should always be 0) or the number of fractional counts for the most recent firmware (see [19]). The legal values for the latest version are in range [0, 2] for the number of half-counts indicating motor coast, as determined by the state of the reed switch at the beginning and end of a drive movement. The *ID* field references the last issued command.

If a *pulse timeout* occurs for any reason, indicating that an axis did not (or unexpectedly stopped to) turn, the controller responds with a string

T <*CD*> <*ID*> <*CNTS*>,

where *CD* denotes the number of counts driven, while *ID* and *CNTS* show the parameters of the last *move* command.

To calibrate the *zero-point reference* for any axis, this condition must be created intentionally by issuing a rotation with a number of counts high enough, so that the drive will trigger a motor end-stop by activating a limit switch. The pointing reference of the telescope in this state is then used as the zero-point for the conversion between the drive and horizon coordinate systems.

When commanding the calibration device, the controller will respond with

C *0 <ID <CNTS>*

when the state change sequence of the device has finished. Here, *ID* and *CNTS* show the parameters of the last *move* command.

Note that all strings transmitted by the controller are terminated with ASCII-value *0x13*, i.e. *Carriage Return* or '\r' and not *0x10* (*New Line Feed* or '\n').

Horizon to Drive Coordinate Conversion

Since the telescope is mounted in an AZ/EL setup, the horizon system is the reference coordinate of choice to control the pointing. In case the axes are not perfectly aligned to the ground reference plane, the first step in the conversion process is to compute the tilt corrections [20] to compensate for the residual errors due to misalignment.

For the desired target coordinates Az and El, the corrections can be calculated from

$$\begin{aligned}\mathrm{corr}_{\mathrm{Az}} = &\arctan\left(T_{\mathrm{Ex}} + T_{\mathrm{Ax}}\sin\left(Az - T_{\mathrm{Az}}\right)\right)\tan\left(El\right)\\ &-\arctan\left(T_{Ex} + T_{\mathrm{Ax}}\sin\left(Az_0 - T_{Az}\right)\right)\,,\end{aligned}$$

$$\mathrm{corr}_{\mathrm{El}} = T_{\mathrm{Ax}}\cos\left(Az - T_{\mathrm{Az}}\right) - T_{\mathrm{Ax}}\cos\left(Az_0 - T_{\mathrm{Az}}\right)$$

where T_{Ax} is the angular tilt of the azimuth axis from the vertical, T_{Az} is the azimuth of the rotation axis of the *tilted reference plane* as projected onto the sky. T_{Ex} is the equivalent tilt in the elevation axis from the horizontal and Az_0 is the zero-point azimuth reference angle of the telescope at the stow (end-stop) position.

The corrections are applied to the target coordinates:

$$\begin{aligned}Az &= Az + \mathrm{corr}_{\mathrm{Az}}\,,\\ El &= El + \mathrm{corr}_{\mathrm{El}}\end{aligned}$$

which are then corrected for the zero-reference angles in azimuth and elevation:

$$\begin{aligned}Az &= Az - Az_0\,,\\ El &= El + El_0\end{aligned}$$

The distance to drive in azimuth from the zero-point to the given target is given by

$$Az_{\mathrm{cnt}} = Az \cdot Az_{\mathrm{cdeg}}\,,$$

where Az_{cdeg} is the number of sensor counts per degree of azimuth driven, which is ≈ 4.74 for the CASSI mount.

The calculation for elevation is more complex, as the pushrod (*fig.* 2.18) converts a linear distance relation to a nonlinear circular motion. The base sensor count (where the pushrod is extended to the maximum) for the elevation axis, given the zero-point reference to the horizontal, is calculated from

$$L = P_L^2 + P_J^2 - 2P_L P_J\cos\left(P_\theta - E_0\right) - P_{Col}^2 \quad .$$

If $L \geq 0$ then

$$\hat{L} = \sqrt{L}\,,$$

otherwise

$$\hat{L} = 0$$

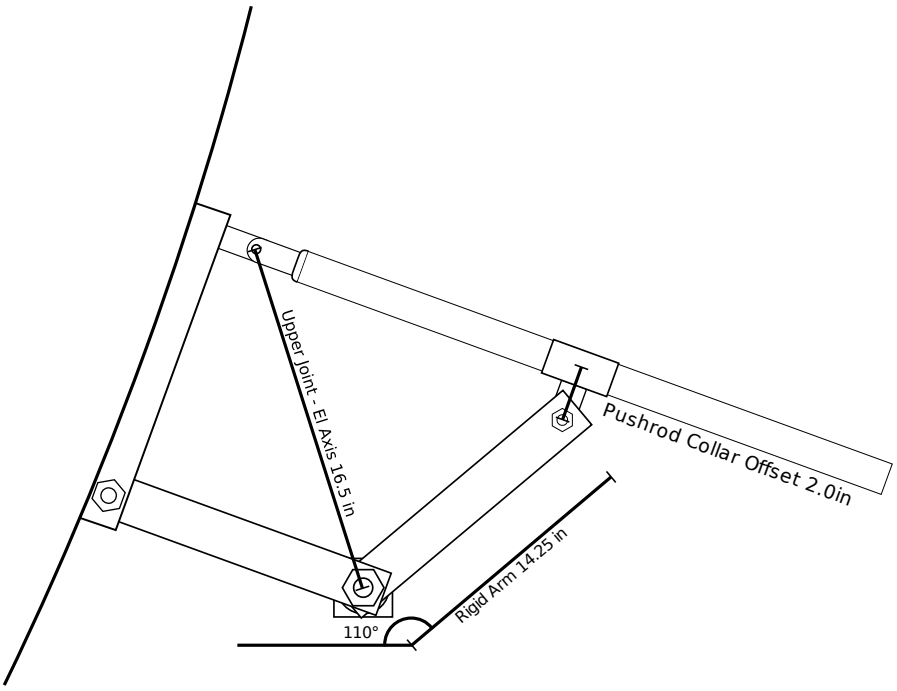


Figure 2.18: Schematic diagram of the CASSI elevation mount. All length values are given in U.S. customary units as per the original specification.

The value of $\hat{L}$ does not change unless the mechanical configuration or position of the telescope is modified. The constants for the setup of the CASSI kit are listed below. Note that all length values are given in *U.S. customary units* as per the original specification.

Variable	Value	Description
P_L	14.25 in	length of the rigid arm (pushrod)
P_J	16.5 in	distance from the pushrod's upper joint to the elevation axis
P_{Col}	2.0 in	pushrod collar offset
P_θ	110°	angle at horizon
P_{cin}	30	sensor counts per inch of pushrod movement
E_0	—	minimum elevation angle (at stow/end stop)

The sensor count is then calculated for the target coordinate

$$El_{\mathrm{Cnts}} = P_L^2 + P_J^2 - 2P_L P_J \cos\left(P_\theta - El\right) - P_{\mathrm{Col}}^2 \,,$$

and if $El_{\mathrm{Cnts}} \geq 0$

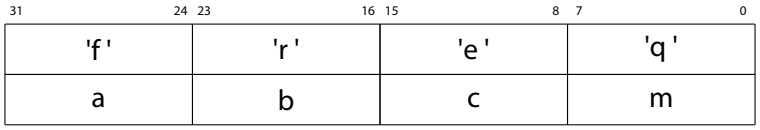
$$\hat{El}_{\mathrm{Cnts}} = \left(\hat{L} - \sqrt{El_{\mathrm{Cnts}}}\right) P_{\mathrm{cin}} \,,$$

else

$$\hat{El}_{\mathrm{Cnts}} = 0 \quad .$$

2.7.7 Spectrometer Unit Operation

The spectrometer unit (*fig.* 2.17) is addressed via the keyword *freq*, followed by a series of octets specifying the reference divider registers for controlling the centre frequency of the local oscillator and the selected bandwidth resolution [21]:



The reference divider for the desired centre frequency in MHz is calculated from:

$$\texttt{refdiv} = \lfloor \texttt{frequency}/0.04 + 0.5 \rfloor$$

Note that the 40 kHz divisor corresponds to the tuning step size of the local oscillator's Phase-Locked Loop (PLL).

The register values *a*, *b*, and *c* are extracted from the reference divider:

```
a = (refdiv >> 14) & 0xff
b = (refdiv >> 6)  & 0xff
c =  refdiv        & 0x3f
```

The bandwidth mode *m* can be set to:

```
0: 7.81 kHz
1: 3.90 kHz
2: 1.95 kHz
```

The controller returns the spectrum as a series of 64 unsigned 16-bit integer words, where the first word is the centre component followed by 31 words of the upper and 32 words of the lower half of the spectrum. The integer values are in internal digital units and can be converted into temperature by multiplication with the calibrated conversion factor.

2.7.8 Implementation

Since the drive and spectrometer controllers share a single data link, it is necessary to enforce exclusive access by serialising communication requests to the link. The design of the plugin backend interface API readily allows for such an implementation (2.6). To facilitate adaptation of the hardware to different configurations, e.g. when only the drive or the spectrometer are to be used in other hardware setups, the spectrometer, drive and link control plugins were implemented as independent separate units and can be loaded individually in the server configuration file.

Access to the serial link (*fig.* 2.19) is implemented via the *shared comlink* group of backend calls. The *acquire* function actively waits to gain a lock on the *link mutex* in order to ensure exclusive access to the link. The link is then accessed via successive pairs of *read-write* commands. The caller must ensure the correct execution of the sequence, otherwise the link may appear stuck, requiring reset. None of the SRT controllers actively transmit data unless first commanded via either the *move* or *freq* keywords. Any issue of a command results in only a single response, provided the prerequisites of valid keyword and required parameter fields are satisfied. The caller must then *release* control of the lock, in order to allow its use by another plugin.

The drive plugin (*fig.* 2.20) implements telescope pointing controls as threaded functions, which are triggered by *move*, *recalibrate* or *park* commands issued by the user via the server

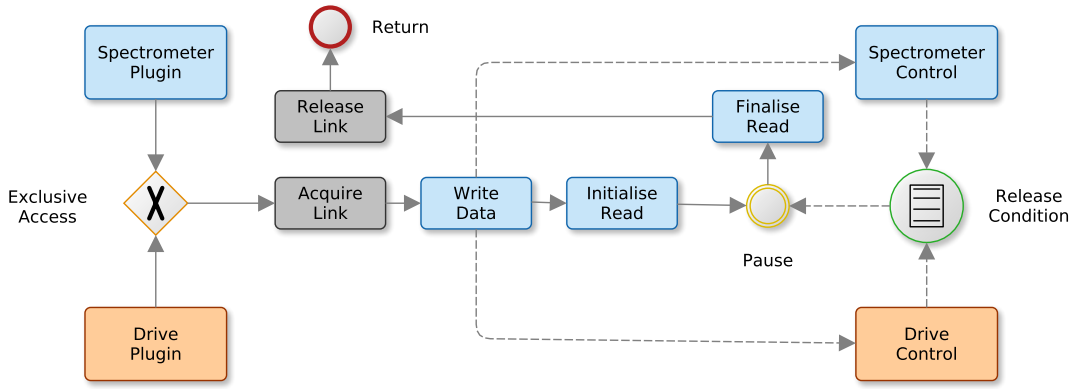


Figure 2.19: The SRT spectrometer and drive hardware cannot be controlled concurrently, as both components electrically share the same link interface. To ensure serialised access to the individual controllers, an intermediate software layer provides a unified interface to the spectrometer and drive control components and ensures mutually exclusive access.

network interface. The mount does not provide absolute position encoding information, therefore the telescope must be rotated to the parking position once per server restart, so the drive coordinate reference is reset to its zero-point.

When issued, the AZ/EL parameters of a *move* command are converted to the drive reference and set internally as the new target pointing. The main control thread then acquires the serial link and executes a *move* command for an axis at a time. On completion, the updated current and commanded target coordinates are compared. If they match, the control logic will pause until a new pointing is commanded.

Since drive operations are uninterruptible when executed by the hardware controller, an option in the plugin's configuration file allows the user to specify an upper limit to the number of counts driven per *move* command. As a rotation over wide angles can take several minutes, this does not only greatly improve interactivity in the system, but is also essential to be able to abort or correct the pointing, e.g. when the user accidentally specified an incorrect target. If the drive controller reports a *drive timeout* during normal operation, a recovery procedure is attempted by rotating the telescope to the park position, which is considered to be a safe state. When this happens, a text message is broadcast to all connected users to inform them of the issue.

The spectrometer plugin (*fig.* 2.21) implements spectral data acquisition. The SRT spectrometer hardware cannot deliver single spectra wider than 500 kHz and uses a fixed number of 64 data bins. Of these, a total of 17 (8 low, 9 high) bins must be discarded due to the prominent I.F. filter stop bands of the GC1011A's decimation filter, which introduce a significant degradation of the Signal-to-Noise Ratio (SNR) in these sections of the raw spectrum, rendering them unusable.

In order to satisfy the requirements for the generalised characterisation of spectrometer hardware (2.5.26), the plugin enables acquisition of the full configurable spectral range of the hardware by concatenation of sequentially observed (raw) spectra, which are acquired by shifting the centre frequency of the receiver's local oscillator through the range requested by the user. The needed strategy is determined every time the user updates the acquisition configuration. When acquisition is enabled, the main control loop locks the configuration for the duration of a cycle. It then proceeds to request the raw spectra from the hardware by repeatedly taking control of the serial link and issuing a *freq* command.

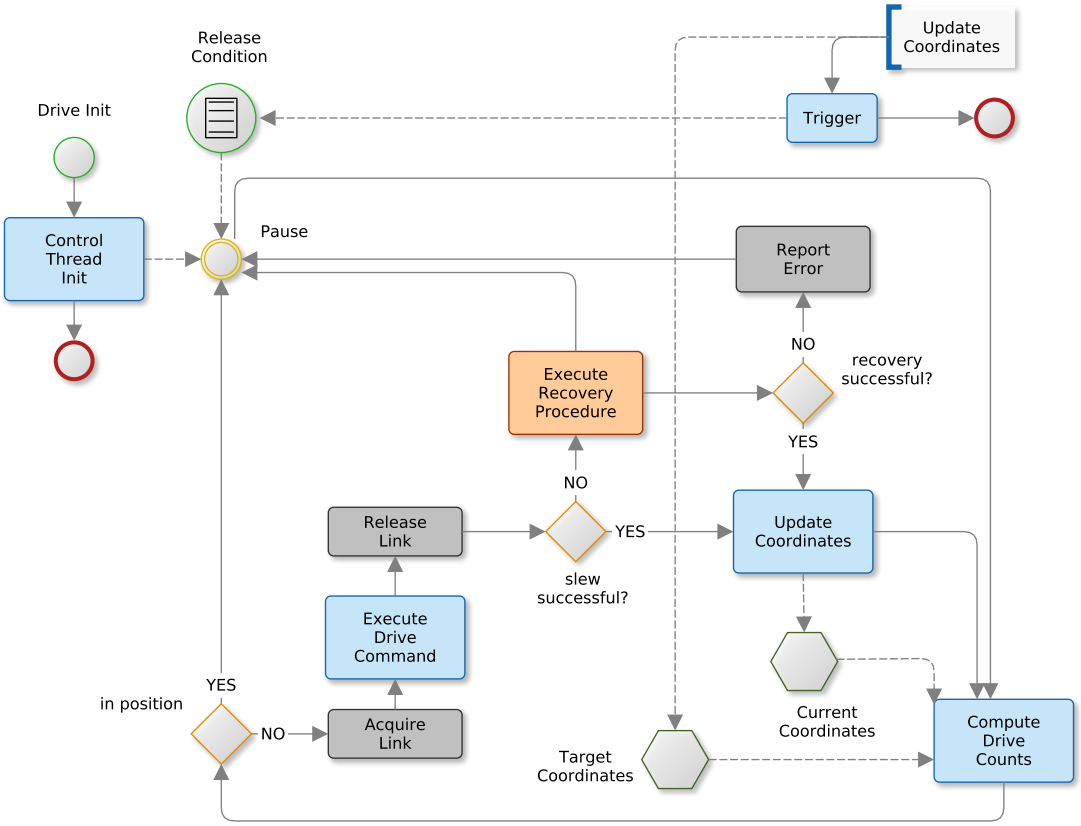


Figure 2.20: The operating principle of the drive plugin. On initialisation, a thread is created which waits until triggered, then executes the move to the new pointing. If an error (timeout) of the drive occurs, a recovery procedure is attempted. Note that this diagram only shows the primary control logic of the plugin.

The raw spectra are pre-processed and stored until the full range has been accumulated. The usable sections of the raw spectra are then extracted and assembled in a continuous sequence (*fig.* 2.14) along with the frequency information. As a last step, the configured calibration is applied and the final spectrum is transferred to any clients connected via the server's network interface.

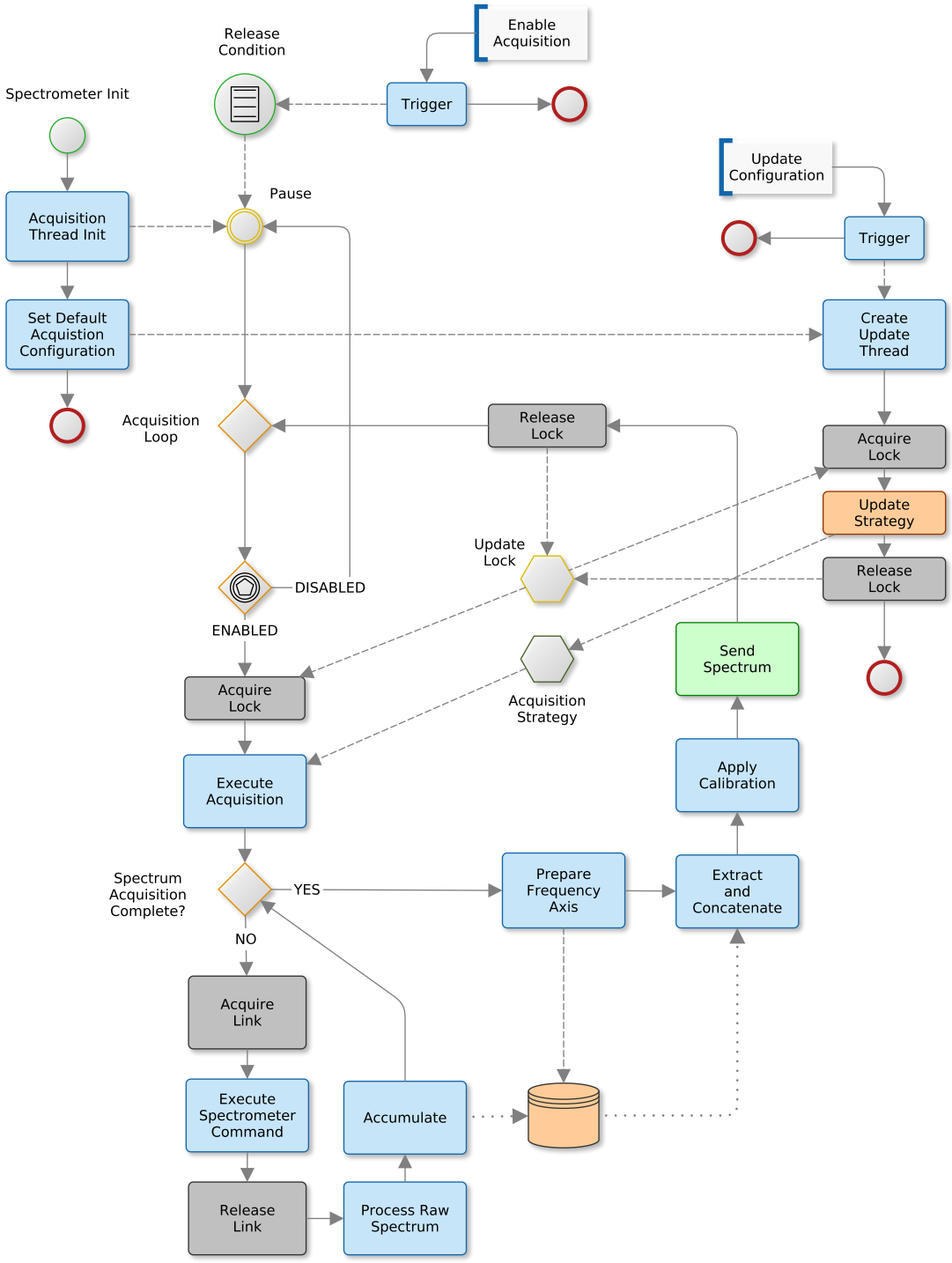


Figure 2.21: The operating principle of the spectrometer plugin. On initialisation, an acquisition control thread is created and a default configuration is set up. The acquisition is then paused until enabled by a user. When enabled, the acquisition control loop locks the configuration updater, so that the computed observation strategy cannot change unexpectedly while an acquisition cycle is being executed. When all needed raw spectra have been acquired, the frequency axis is determined and the raw spectral data are extracted, concatenated and the temperature calibration is applied to form the requested spectrum. The data are then transmitted to all connected users via the network interface. Note that this diagram only shows the primary control logic of the plugin.

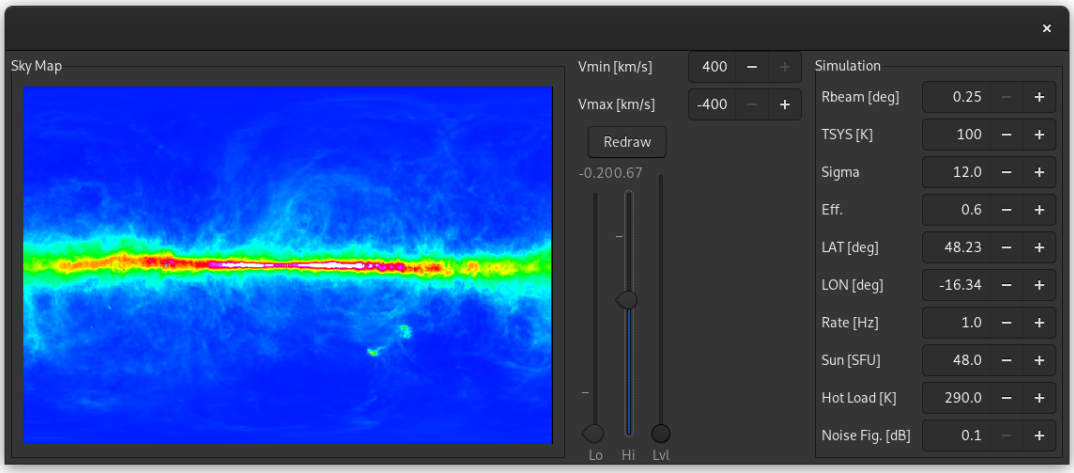


Figure 2.22: The user interface of the simulator plugin allows the user to configure various instrument parameters, such as beam size, geographical location and acquisition data rates. A rendered preview of the sky for the given beam size is also shown.

2.8 Implemented Backends: Simulator

2.8.1 Motivation

Radio telescopes are typically shared instruments. This is relevant in science, but even more so in the context of teaching. Typically, tens of students will share a single instrument to do projects and exercises in courses on radio astronomy. It is important for students to use actual hardware for exercises, so they may develop a sense for their usage, operation, and problems which present themselves in real devices and can affect the data in the acquired measurements. Still, even with the most user-friendly control interfaces, some time is needed to understand concepts, functions and configuration options. It is therefore sensible to offer all participants a resource in the form of a *simulated radio telescope*, which they may use to train and practice their skills, without the need to wait for a slot to become available, just to be pressed for time while using the actual instrument. Furthermore, this adds the option to change the simulation parameters in order to demonstrate the different characteristics of other radio telescope configurations (*fig.* 2.22), such as beam size, system noise contributions and data rates. Lastly, it represents a safe test bed and sand box for unexperienced users, allowing them to run and verify their observation strategy before executing it on an actual telescope.

2.8.2 Characteristics

The minimum configurable beam size is 0.5°, with the upper limit set to 10°. While the latter is more or less chosen arbitrarily, the selection of the minimum size actually simplifies a lot of assumptions needed in the simulation of objects such as the Sun and the Moon, as their apparent diameter corresponds to approximately 30′. This means that the simulated emission of these objects for any beamsize greater than the minimum can be modelled simply by scaling their equivalent temperature to the centre bin of an N×N sized grid of a Gaussian-shaped beam (normalised to unity integral), as any lower resolution telescope would just detect them as a point source. The chosen minimum beam is also a sensible lower limit in the choice of grid resolution for the included hydrogen database (2.8.4).

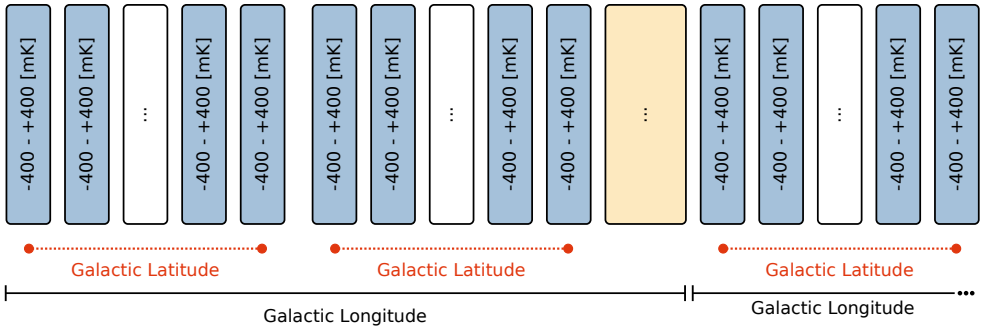


Figure 2.23: The binary layout of the simulator's hydrogen data file. Starting at galactic longitude 0°, blocks of 801 16-bit integers hold the temperature values (in centi-Kelvins) for the radial velocity bins from -400 to +400 km/s for all latitudes from -90° to +90°.

2.8.3 Telescope pointing

Since the minimum configurable beam size is 0.5° and also the resolution of the hydrogen data, the pointing precision of the simulated telescope is set to the same lower limit, i.e. any coordinate γ is snapped to the underlying grid of resolution δ by rounding:

$$\gamma = \delta \left\lfloor \delta^{-1} \gamma \right\rceil \quad .$$

This greatly simplifies beam convolution, as no extra steps must be taken to interpolate the data onto a higher resolution grid.

2.8.4 HI Data Set

To simulate spectra for the 21cm line of neutral hydrogen, a dataset of HI profiles has been retrieved from the *HI Surveys Data Server* run by the *Argelander-Institut für Astronomie* (AIfA) of the Department of Physics and Astronomy at the University of Bonn, Germany.

The spectra were extracted from the HI profile collection of the HI4PI [22–24], EBHIS [25], and GASS [26–28] databases for the whole sky in a grid pattern at a beam resolution of 30′, which is a reasonable choice, given the beams of the original data sources, which are 36′ and less. A bash shell script was used to automatically retrieve the data. It makes use of the interpolation feature provided by the web interface to select a beam size of one half degree. The script is provided as part of the source code repository of the software.

The downloaded uncompressed data size amounts to about 9.3 GiB in the form of text files, formatted in two columns, which are the VLSR and the corresponding temperature:

```
(...)
%%EBHIS      0  datapoints: v_lsr [km/s] and T_B [K]
%%GASS       0  datapoints: v_lsr [km/s] and T_B [K]
%%LAB      777  datapoints: v_lsr [km/s] and T_B [K]
    -399.83     0.004
    -398.80    -0.026
    -397.77    -0.022
    -396.74     0.015
    -395.71    -0.002
(...)
```

Since in this shape, both storage size and format are highly inefficient, the data was converted to a binary format. The storage format was designed based on the following considerations:

- Per each position, the spectrum is composed of 801 samples, as the velocity axis ranges from -400 to +400 km/s at about 1 km/s per bin, including the zero-point.
- The galactic latitude axis is 362 positions high, ranging from -90° to +90°, (including the zero-point), in steps of 0.5°.
- The galactic longitude axis is 722 positions wide, ranging from 0° to 360° in steps of 0.5°.
- The provided temperature resolution is in milli-Kelvin and much higher than needed. The last digit can therefore be discarded, which allows the values to be stored in 16-bit integers in units of centi-Kelvin.

Note that the choice to include the data redundancies at the 0°/360° and $\pm$90° overlaps was made deliberately to simplify index offset calculation in the software. Also note that close to the Galactic Centre ($\sim$1°), the temperature data around 0 km/s is corrupted by what appears to be a numerical error in the original data in the HI survey database. No attempt was made to correct these data.

The storage format for the binary database is shown in *fig.* 2.23. The original data are extracted from their text representation and re-formatted by the following procedure:

1. For each galactic longitude from 0° to 360°, iterate over all corresponding galactic latitudes in 0.5° steps.
2. Open the downloaded text file for the given position and read the spectral data line by line.
3. Round the velocity bins to the nearest integer and count their occurrence.
4. Accumulate the temperature value for the given velocity bin.
5. Iterate over the acquired data array.
6. If a velocity bin shows an occurrence of zero, perform a linear interpolation of the temperature between adjacent bins and set the occurrence to one.
7. Divide the accumulated temperatures per bin by their occurrence.
8. Multiply by 100 to convert to centi-Kelvin and cast to integer.
9. Append the resulting array of integer temperatures to the binary output file in little-endian format.

A conversion utility implementing the procedure is provided in the source code repository of the software.

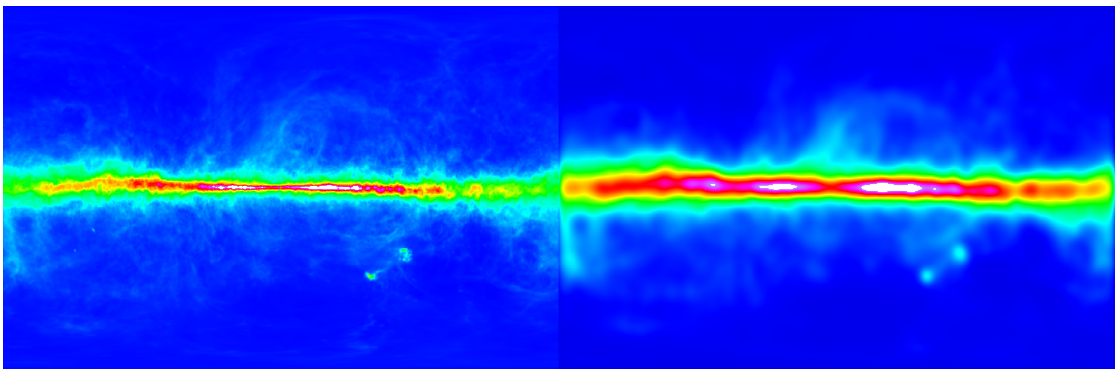


Figure 2.24: The hydrogen sky in resolutions of 0.5° (left) and 6.0° (right) as shown in the simulator's user interface. The spectral data are integrated along the radial velocity axis within the user configured interval (here: $\pm$400km/s) and convolved by a Gaussian kernel appropriate for the selected simulated beam diameter.

2.8.5 Hydrogen Sky

In order to simulate radio telescope beams larger than the 30′ baseline resolution, convolution is applied to generate hydrogen data for Point Spread Functions (PSFs) of different sizes. For simplification, and given the contribution of higher diffraction orders to the signal are negligible, especially in the context of the solely educational purpose of the simulator, the PSFs are assumed to be of symmetric two dimensional *Gaussian* shape.

Two distinct approaches are used to generate the convolved data. The first method employs discrete convolution of the spectral data. This is used for the simulated spectra which are sent to the client software. The other method applies the *Fourier convolution theorem* to compute an image of the hydrogen sky (*fig.* 2.24) for display in the user interface of the simulator software.

Both methods show significant performance advantages given their use case. This is due to the way data are handled to achieve the desired outcome. The discrete approach is very suitable for the simulation of a convolved spectrum for a given on-sky position, as the convolution kernel is typically much smaller than the whole-sky grid and is applied to the velocity bins of the spectra for a point-wise output, while the Fourier method is very fast when applied to a whole-sky image, which is derived from the integration of the spectral bins along the velocity axis.

2.8.6 Convolution Kernel

The PSF is modelled as a symmetric two-dimensional Gaussian for a given cutoff σ, a beam radius r, and a baseline resolution θ in degrees.

The FWHM of the width-normalised Gaussian kernel between $\pm\sigma$ is given by

$$\mathit{FWHM} = r\sigma\sqrt{2\log 2} \quad .$$

The width and height of the number of grid elements n needed for a discrete representation of the kernel is calculated by rounding the FWHM towards the nearest integer, given the baseline resolution:

$$n = \lfloor 2\,\mathit{FWHM} \rceil \sigma^{-1} \quad .$$

The dimensions of the kernel must be an integer no more than n:

$$\hat{n} = \lfloor n \rfloor \quad .$$

The number of elements must also be uneven in order to accommodate a single central bin, so the convolution kernel can be centrally placed on the data subgrid, without the need for interpolation:

$$\hat{n} = \begin{cases} \hat{n}, & \{\hat{n} = 2k+1 \mid k \in \mathbb{N}\} \\ \hat{n}+1, & \text{otherwise} \end{cases}$$

To compute the convolution kernel K, it is necessary to determine the distance s from the central bin for each point via the *Euclidean norm*, scaled to the resolution of the underlying grid:

$$s_K = \frac{1}{2}\theta\|x_K\| \quad .$$

The distance s_K corresponds to σ, but must be rescaled to $\hat{s_K}$ to get to the width-normalised Gaussian for a given beam radius (half the FWHM with $\sigma = 1$):

$$\hat{s_K} = r^{-1} s_K \, 2\sqrt{2\log 2} \quad .$$

The value of the Gaussian at the given distance is then calculated from:

$$g(\hat{s_K}) = \left[\sqrt{2\pi}\exp\left(-\frac{\hat{s_K}^2}{2}\right)\right]^{-1} \quad .$$

Finally, the convolution kernel is normalised to unity integral:

$$\hat{K} = K\left[\sum_{i=1}^{\hat{n}^2} K_i\right]^{-1} \quad .$$

This is needed so the total signal amplitudes of the convolved spectra are preserved.

Note that the employed method of generating the convolution kernel is only an approximation of the Gaussian, as there is no integration between the integer coordinate boundaries of the sampling grid and thus no consideration of the inter-grid gradient, which would (in a practical implementation) require expensive oversampling and resampling of the Gaussian for little benefit, given the application.

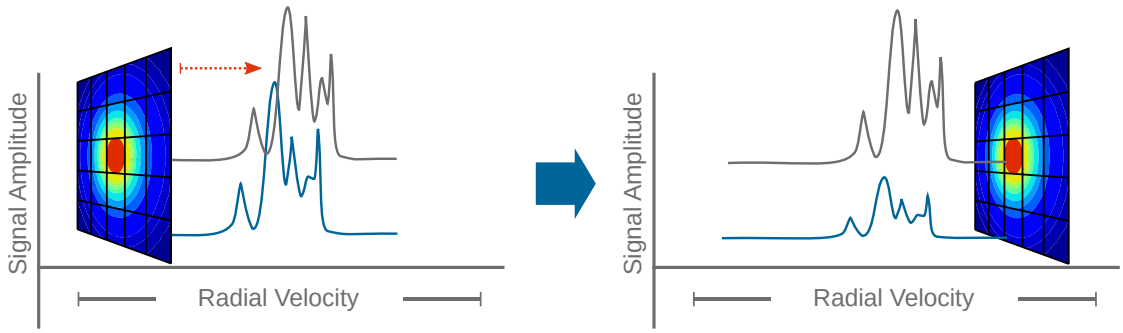


Figure 2.25: In the discrete convolution approach, the spectra within range of the target coordinate (given the beam size) are convolved by multiplication along the radial velocity axis, so their amplitude is scaled to appropriately reflect their contribution to the total signal contained in the beam. The processed spectra are then co-added, resulting in the final spectrum (not shown).

2.8.7 Hydrogen Spectrum Generation

Simulated spectra for transmission to the client software are generated by applying pointwise multiplication of the normalised convolution kernel $\hat{K}$ with the spectral data S (*fig.* 2.25) around the selected telescope pointing. The distance limit is set by the selected beam size and σ-cutoff parameter of the Gaussian and is directly derived from the dimensions of the convolution kernel, as one grid unit corresponds to the base resolution of 30′. The convolved data are accumulated for each bin in the velocity range $[v_0, v_1]$ to form the output spectrum $\hat{S}$:

$$\hat{S} = \sum_{i=1}^{n^2} \left(\hat{K}_i S_{ij} \right), \; v_0 \leq j \leq v_1 \quad .$$

The radial velocity interval is determined by the client-controlled (virtual) spectrometer frequency range parameter, given the VLSR in the direction of the current telescope pointing. Note that the selection of input spectra for a sky coordinate is done on a square, rather than a circular grid, for simplicity of computation.

The selected method is superior to the Fourier approach in that spectra can be recomputed quickly when the beam size is changed in the simulation parameters, rather than having to pre-compute several hundred convolved subframes along the radial velocity axis. It must be mentioned that (long-term) the cumulative computation time will be greater than what would be consumed by the application of the convolution theorem, particularly when the size of the beam is changed infrequently. The choice to select this method was however made with the user experience in mind, as it allows very quick, just-in-time data generation with no perceived lag in responsiveness, while introducing a constant (albeit minor) background computational load.

2.8.8 Hydrogen Map Generation

In order to provide insight to the user on the current configuration of the simulated telescope, a map of the generated sky is displayed as part of the simulator user interface. While objects such as the Sun or the Moon can be displayed with little effort, a map of the neutral hydrogen is computationally expensive. The method described in the previous section (2.8.7) is not suited for this task, as it would necessitate the computation and integration of a convolved spectrum for each point on the map with every configuration change. While this is somewhat acceptable for small beams, it is unfeasible for larger beam sizes, given that the shape of the convolution kernel (2.8.6) implies a computational complexity of at least $\mathcal{O}(n^2)$.

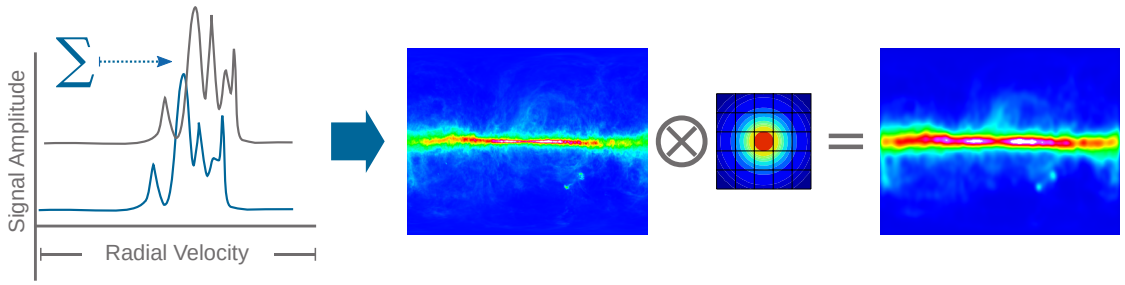


Figure 2.26: In the Fourier convolution approach, a map of the sky is first computed by integrating the spectra along their velocity axis for each position. Convolution with the beam is then performed on the flattened map.

To provide a satisfying user experience, multiplication in Fourier space was implemented via the circular convolution theorem. Here, a map of the sky is first generated by integrating the spectra for all on-sky positions within the simulator-configured radial velocity interval $[v_0, v_1]$. The map M is then convolved with a convolution kernel $\hat{K}$, resulting in $\hat{M}$ (*fig.* 2.26):

$$\hat{M} = \mathscr{F}^{-1}\{\mathscr{F}\{M\} \cdot \mathscr{F}\{\hat{K}\}\} \quad .$$

A compact radix-2 fast Fourier transform, similar to the method of *Cooley–Tukey* [29] was implemented and made available as part of the source tree. This was done in order to minimise dependencies to external libraries, as demanded by requirement R-SWA-002. The code is reasonably fast when compiled with modern OpenMP optimisation directives and within one order of magnitude of highly optimised implementations, such as provided by the well-known *FFTW* libraries. The details of the methodology (e.g. [30]) or implementation are not discussed here.

2.8.9 Moon

The Moon is modelled by the approximation of its reflected solar energy based on the solar constant at about 1 AU distance from the Sun, reduced by the angle between the surface of the Moon facing the Earth and the Sun.

First, the position of the Moon M is calculated for the current date and snapped to a 30′ grid point (2.8.3). Next, the position of the Sun S is calculated and the angle between the Moon and the Sun towards the Earth is computed to determine the fraction of reflected solar energy:

$$\cos\theta = \frac{|S|}{|S - M|} \quad .$$

The average annual local solar irradiance averaged over the Moon's surface is then

$$E = 1361\,\pi^{-1}\cos\theta\,[\mathrm{Wm}^{-2}] \quad .$$

Under the assumption that the Moon's emissivity equals its reflectance, its temperature can be calculated by applying the Stefan-Boltzmann (1.2.1) equation:

$$T = \sqrt[4]{E\sigma^{-1}}\,[\mathrm{K}] \quad .$$

Finally, the temperature is scaled by the ratio of the Moon's apparent diameter of 30′ to the configured Gaussian beam and stacked on top of the simulated hydrogen background.

The incident radio flux can then be calculated from [31]:

$$S = 7.349 f^2 T d^2\,[\mathrm{Jy}]$$

where f is the observed frequency in GHz, T is the average lunar brightness temperature, and d is the angular diameter of the Moon.

2.8.10 Sun

For the simulation of the Sun, its current position is computed and the user-configurable solar flux in *solar flux units* (1 *sfu* = 10^4 Jansky) is converted to equivalent antenna temperature (1.3.11) for the given beam size, under the assumption of an apparent angular solar diameter of 30′. The resulting temperature is then scaled to fit the Gaussian beam grid and stacked on top of the simulated hydrogen sky.

While solar limb brightening can be quite prominent depending on the radio band, no relation between frequency, radius, and brightness (e.g. [32, 33]) is simulated, given the choice of base resolution and hence the achievable simulated telescope resolution.

2.8.11 Cosmic Microwave Background

The cosmic microwave background (CMB) is assumed to be uniform in all directions and is added to the signal for a contribution of 2.725 Kelvin [34].

2.8.12 System Temperature and Noise

A user-configured system temperature (1.3.3), excluding the CMB contribution, is added to the signal.

The noise is modelled to be Gaussian-like and scaled to unity, with a configurable σ parameter and an amplitude which is dependent on the square root of the signal level S:

$$S' = S + n\sigma\sqrt{S} \quad .$$

The noise parameter n is approximated by applying the *Box-Muller* [35] transform to two independent samples u, v chosen by the *POSIX rand()* function and scaled to unit interval. These samples are then transformed to

$$\begin{aligned} n_1 &= \sqrt{-2\log u}\cos\left(2\pi\, v\right), \\ n_2 &= \sqrt{-2\log u}\sin\left(2\pi\, v\right) \end{aligned}$$

in sequence, before a new pair of samples u, v is chosen. The transformed values n_1 and n_2 are independent random variables following a standard normal distribution.

The noise temperature (1.3.2) contributed by the amplifier chain is computed from a user-configurable noise figure. The theoretical root mean square of the noise amplitude σ is calculated from recording bandwith $\Delta\nu$ and the integration time τ for the given system temperature $\mathrm{T_{sys}}$, which is considered the temperature of the amplifier:

$$\sigma = \frac{\mathrm{T_{sys}}}{\sqrt{\nu\,\Delta\tau}} \quad .$$

2.8.13 Calibration Devices

A hot load can be activated in the simulator. The hot load temperature is configurable in the user interface and its value is simply added to the output spectrum, when activated.

2.9 User Interface

This section describes the considerations which went into the visual design, layout and logical flow of the user interface and, if applicable, justifies the choices made. For the formal software design of the GUI client see subsection 2.2.3, which identifies and constrains all essential components, their purpose and functionality.

The principle design is based on a tabbed layout, which presents a row of buttons to the user that switch between the various displays. These displays may be detached from the main screen arrangement by a drag operation on any of the selectors and will then be shown in a separate window when released outside the selector row, or reattached at the pointer position, when released within a selector row group. A collapsible indicator field is displayed at the bottom of all windows, showing essential information regarding the current status of the telescope. The top menubar allows access to a configuration menu, where parameters such as the server internet address and the username may be set. In addition, buttons are provided, which allow the user to quickly distribute and collect the tabbed displays to and from separate windows.

2.9.1 Chat & Log

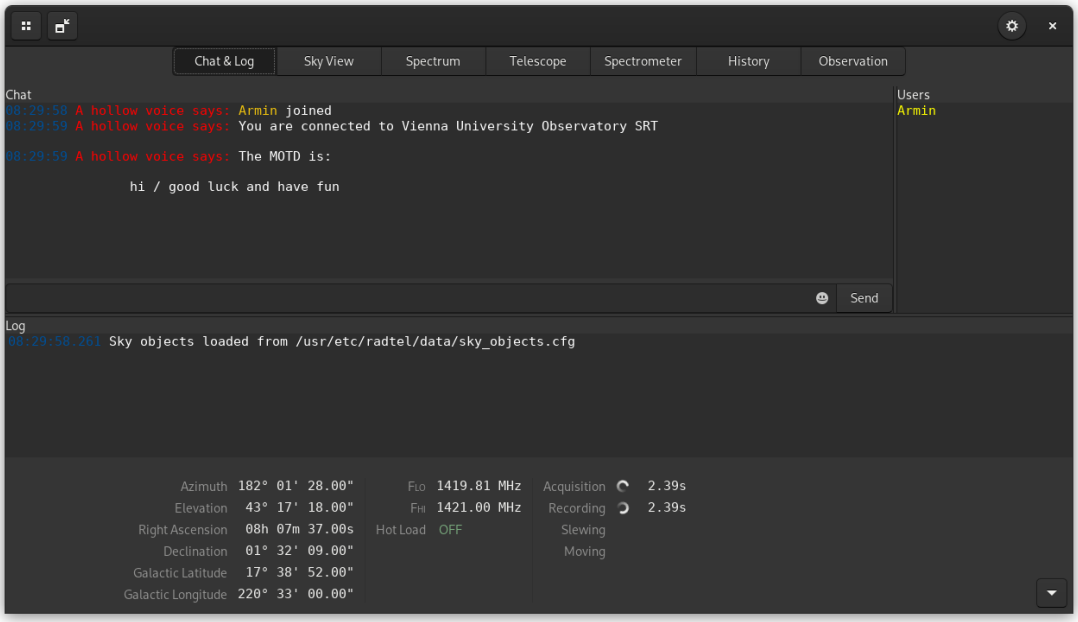


Figure 2.27: The Chat & Log tab is split horizontally. Server messages are rendered in the top section, while client messages are displayed on the bottom. The user may type messages which are broadcast to all other connected users via the server, reminiscent of Internet Relay Chat.

Multi-user capability and communication are requirements of the software (R-USR-004, R-USR-005). This is reflected in the design of the interface (D-SAD-011) and was realised as depicted in *fig.* 2.27. The screen is split horizontally to contain the *chat* portion at the top, which serves as a means of inter-user communication and for displaying broadcast status messages sent by the server, along with the list of currently connected users to the right. The bottom half shows a *log* of content generated by the client software, such as informative messages or error indicators.

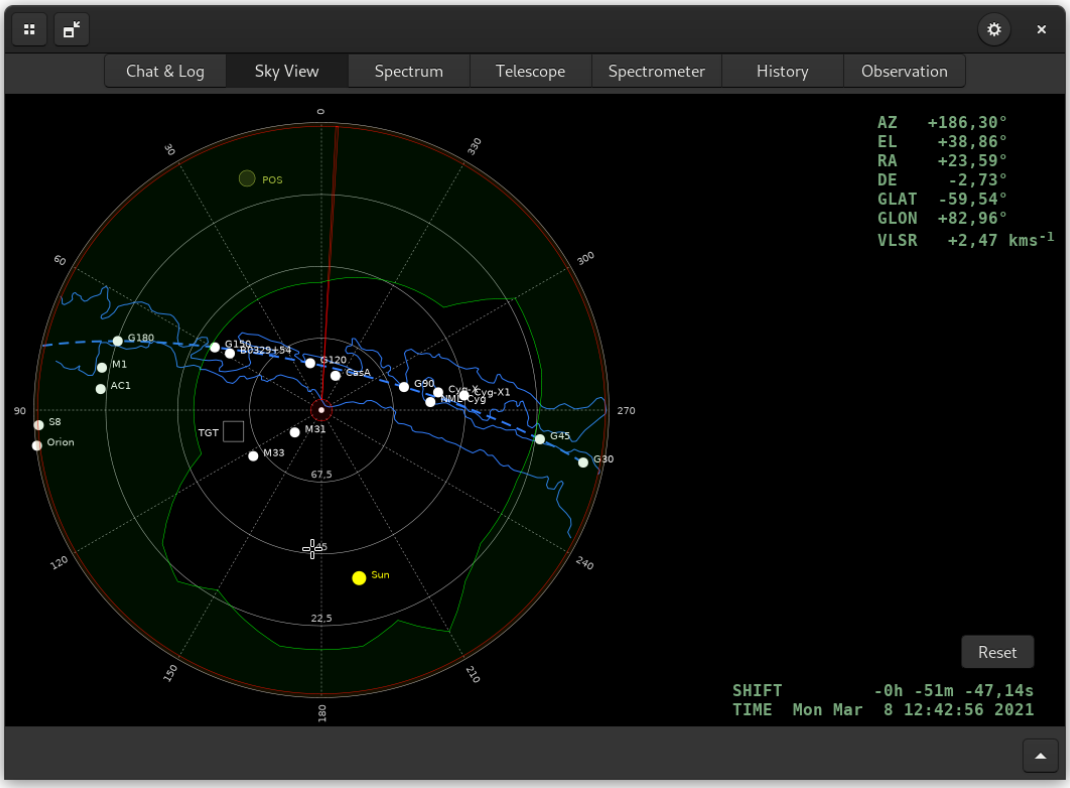


Figure 2.28: The Sky View renders a view of the sky at the geographical location of the telescope in an azimuthal projection with the zenith at the centre. Mechanical constraints of the drive are indicated in red, while the local horizon profile is depicted in green. Catalogue objects and the Milky Way are shown, along with the current and requested target pointing of the telescope. Coordinates under the cursor are rendered in the top right corner. If time shift is enabled, the parameters are rendered in the bottom right corner.

2.9.2 Sky View

The sky view (*fig.* 2.28) implements D-SAD-012 and shows the current sky at the geographical location of the radio telescope in an *azimuthal equidistant projection*. The zenith is located in the centre of the map, north is rendered on top and east to the left. This represents the view of an observer lying on their back, oriented north-south.

The mechanical movement limits of the telescope are indicated by red lines. The rendered lower elevation limit is fixed to the mathematical horizon of zero degrees, as there is no practical reason to go below this limit during regular operation, even if the telescope's mount is capable of rotating to negative elevations.

If set up in the server configuration file, a profile of the real local horizon around the telescope is shown as a green line and the area below the line is shaded. This serves as a quick-look visual aid for recognising the unobstructed (and hence observable) regions of the sky.

Catalogue objects are represented as white dots with labels. These may be edited in a configuration file at the user's discretion, the exception being the Sun and the Moon, which are algorithmically computed. The plane of the Milky Way (along the galactic equator) is indicated by a blue dashed line, with approximate contours of its outline drawn for reference.

If an object is selected by the user, tracking of the coordinate will be enabled until it is deselected. Objects with variable equatorial coordinates, particularly the Moon, which changes its position quickly compared to the stars, are tracked at the appropriate speed when enabled

via this interface. The user can also issue a drive command to the telescope to go to any position within the mechanical movement constraints, by clicking the target area and holding the proper command key.

The current position of the telescope's main beam is indicated by a circle, while the last requested pointing target is indicated by a square. These positions are automatically broadcast by the server and updated on the map whenever a coordinate update is commanded.

In the top right corner of the map, the sky coordinates under the cursor and the corresponding VLSR are displayed. A drag operation on the map can shift the rendered sky arbitrarily in time to help with observation planning. If this is the case, any coordinates displayed will be computed according to the absolute time and date set by the shift.

2.9.3 Spectrum

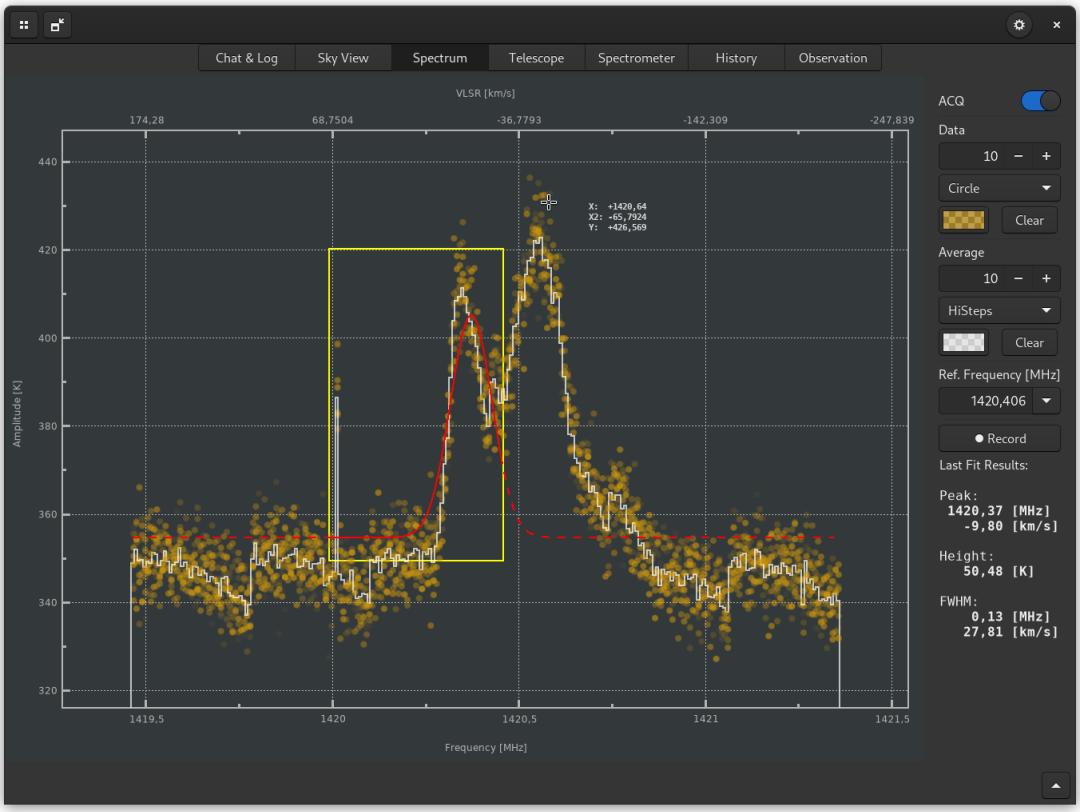


Figure 2.29: The Spectrum display shows individual spectra as a point cloud, while their moving average is drawn as a line. The depth of the stored data buffer, as well as the length of the moving average is user-configurable. Users may zoom and move the graph. Basic fitting of spectral lines is built-in. Other functions can be reached via a pop-up menu and keyboard commands.

The spectrum display (*fig.* 2.29) implements D-SAD-013. By default, the individual spectra are represented as a point cloud, while the accumulated spectrum is shown as a line. The user may configure the maximum number of spectra displayed concurrently. These are buffered in FIFO order and rendered with an alpha channel according to their age. This means that the display renders historic data with configurable *persistence* analogous to phosphorus screens, with old data fading according to the selected buffer depth. The accumulated data are similarly updated, with a moving average computed from the stored spectra for the configured window length.

The display is capable of zooming in both axes, either by selecting a region of interest by rectangle, or by scaling the axes individually. Axes can also be shifted arbitrarily and be re-scaled automatically for best fit. The lower x-axis is shown in frequency, while the upper axis is in VLSR. The reference rest frequency for this axis can be configured by the user. The y-axis amplitude is given in temperature. Coordinate values under the pointer are displayed for all three axes.

For basic analysis of spectral lines, a region of interest in the point cloud can be selected, for which a Gaussian profile is fitted. Fitting is done by a *Levenberg-Marquardt* algorithm implemented as part of the software, but is not discussed here. The fit parameters show FWHM, position and height of the peak above baseline for the Gaussian.

The spectrum display allows the user to enable recording of the raw spectral data in the form of a text file, if they desire further processing in other software. The file format is described in the header of the recorded data files.

The display also allows the export of a snapshot of the rendered plot or of individual graphs as xy-formatted text files in a right-click pop-up menu. These, or any similarly formatted files, can be imported and displayed via the same menu. It is also possible to import three-column data. In this case, the data will be displayed as a map (2.10), with the third column encoded as a colour value.

2.9.4 Telescope

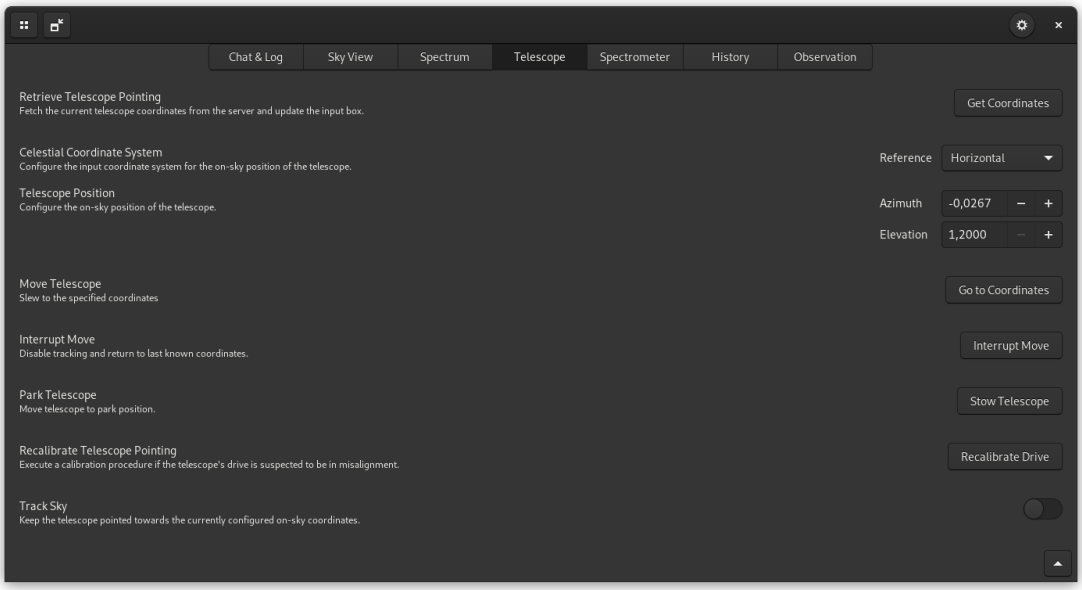


Figure 2.30: The Telescope pointing controls are arranged in top-to-bottom order, which allow the user to fetch, set and command the telescope drive unit parameters.

The telescope tab (*fig.* 2.30) implements D-SAD-014. It presents the user with a logical flow which is oriented top to bottom. The entry fields in this tab are not automatically set to the current coordinates of the telescope sent by server updates, so there is no interference with user input.

While not strictly necessary, the user may fetch the current telescope pointing to the position entry field by clicking the appropriate button. If this is of no concern, they may skip ahead and select the coordinate reference system, which may be chosen from either *horizontal*, *equatorial* or *galactic*. When changing the reference, the position entry fields are re-labelled

accordingly and the coordinates are transformed to the selected system. The user may then edit the fields and issue the *go-to* command. This concludes the nominal procedure for moving the telescope.

Slewing the telescope can take significant amounts of time, so in case an incorrect target has been commanded, a function to interrupt the movement is available. This will stop the telescope at its last known position.

In order to rotate the telescope to its default parking position, a move to *stow* can be commanded. This procedure may not be interruptible, depending on the backend plugin's implementation in use on the server.

If the telescope is suspected to be out of alignment, a recalibration procedure of the drive can be commanded. This will drive the telescope to a hardware-referenced position to reset the zero-point of the telescope's coordinate frame, followed by the re-pointing to the last commanded target.

The sky tracking feature may also be toggled via a switch in this tab. This is interconnected with the sky view (2.9.2), as well as the automatic observation procedures (2.9.7). Deactivating this feature may cause unexpected results, if done while using one of these automated programmes. If tracking was enabled via this switch, the equatorial coordinates at the current telescope pointing are used as the reference target.

2.9.5 Spectrometer

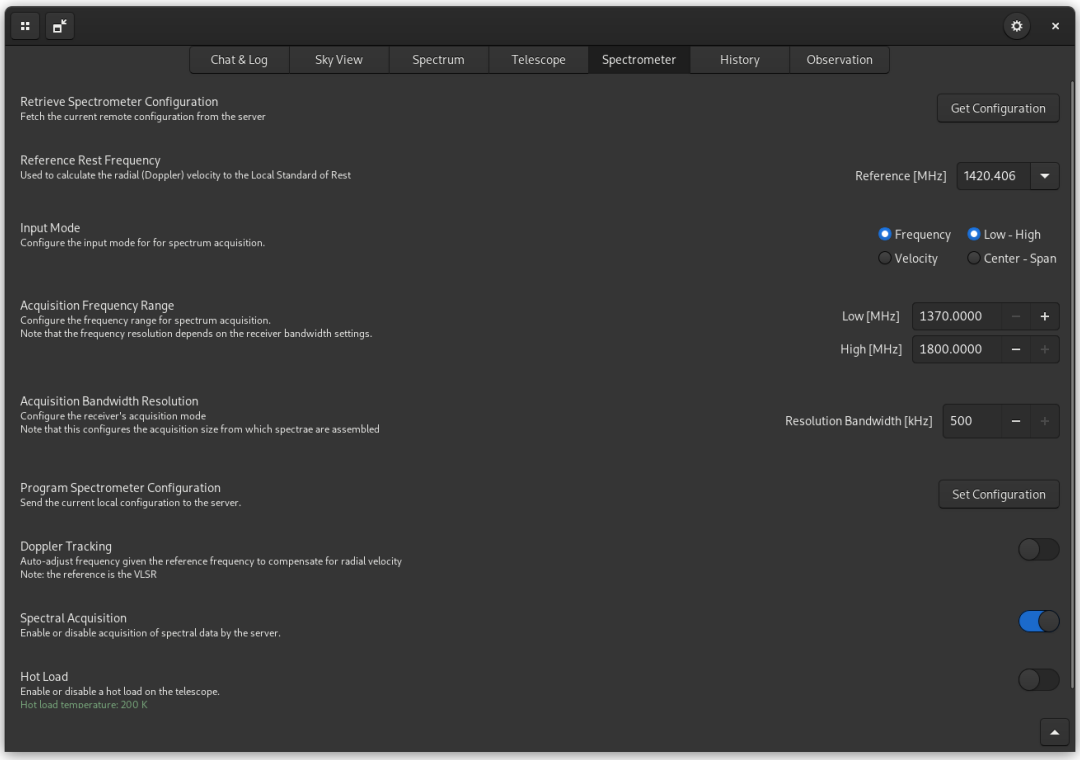


Figure 2.31: The Spectrometer settings follow a top-to-bottom workflow. The user sets up the configuration items step by step, before issuing a command to the server.

Similarly to the Telescope control tab, interaction with the spectrometer settings (*fig.* 2.31) follows a top-to-bottom sequence, implementing D-SAD-015. Here, the user may retrieve the current settings from the server before changing the configuration. Next, they can change

the reference rest frequency for the radial velocity calculation. This defaults to the 21cm line of neutral hydrogen, but other presets are available for selection as well.

Next is the input mode configuration, which can change the entry fields between the following configurations:

- `Frequency, Low - High:`
  the spectral range is defined by a lower and upper frequency bound
- `Frequency, Centre - Span:`
  the spectral range is defined by a centre frequency and total bandwidth span
- `Velocity, Low - High:`
  the spectral range is defined by a lower (*red*) and upper (*blue*) radial velocity
- `Velocity, Centre - Span:`
  the spectral range is defined by a central radial velocity and a radial velocity span

Changes to the input method trigger a conversion of the values in the entry fields to the new setting. The radial velocities are computed according to the reference rest frequency and internally converted to a frequency range, whenever a configuration command is transmitted to the server.

The acquisition bandwidth setting changes a resolution setting which is specific to the implemented spectrometer plugin (2.5.26, 2.7.7) and can only be changed if supported by the backend.

Finally, the user can command a configuration parameter update to the server by clicking the corresponding button.

The *Doppler tracking* slider enables auto-adjustment of the central frequency given the last configuration update, when the telescope pointing is changed on the sky. This allows for the compensation of the changing VLSR in the viewing direction, relative to the configured rest frequency reference, so the line position in the spectrum is only affected by the proper motion of the source, rather than the relative velocity of the observer.

2.9.6 History

The history tab (*fig.* 2.32) implements D-SAD-016 and serves as a signal trend display. In the upper half, the integrated (continuum) value of recent spectra is shown, indicating the change in measured temperature over time. The lower half shows the data in a top-to-bottom oriented waterfall diagram, with the frequency axis shown in the horizontal direction and the spectral amplitudes encoded in colour.

2.9.7 Observation

The observation tab (*fig.* 2.33) implements D-SAD-017 and provides a set of observation procedures to the user. When the user selects to start a procedure, they are guided through the configuration process by a wizard menu, which informs them about its purpose and the configurable parameters available in the selected mode. When a procedure is executed, the list of selectors will be hidden and information displays applicable to the task will be shown while active. The particular modes are not discussed here, but will be demonstrated in chapter 3 where applicable.

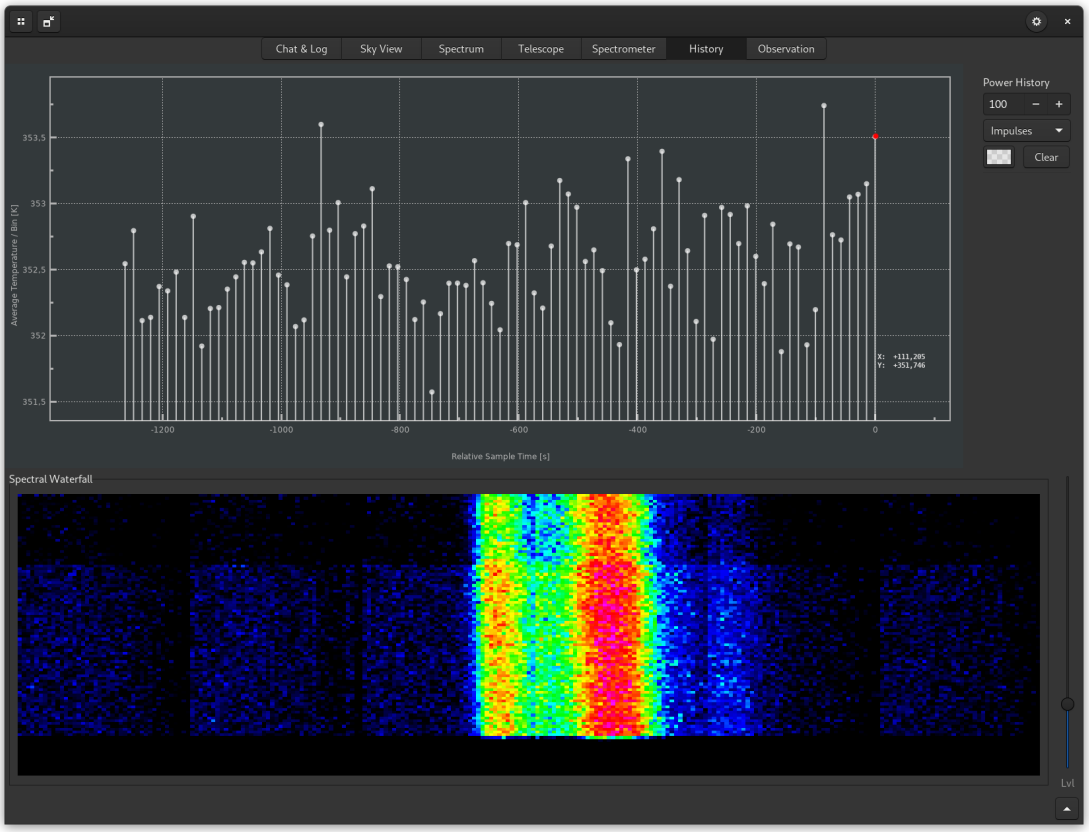


Figure 2.32: The History tab shows the continuum values of recorded spectra, with the most recent value marked in red and older data displayed towards the left. Below is a waterfall diagram of the spectra, with their amplitudes encoded in colour.

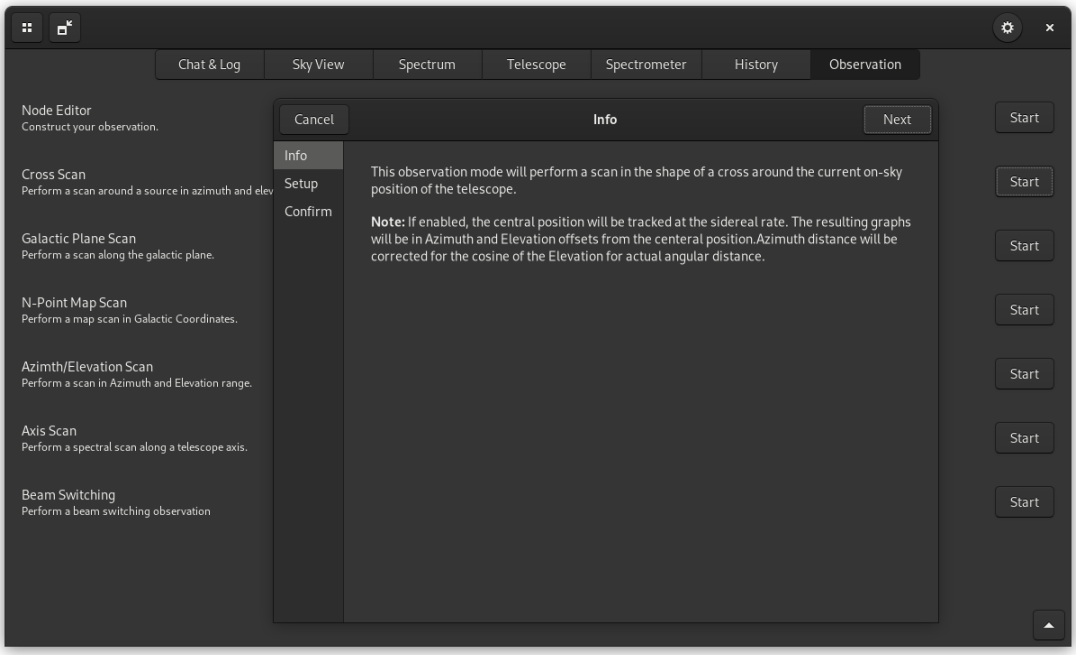


Figure 2.33: The Observation tab offers a selection of observation modes. A configuration wizard guides the user through the setup process.

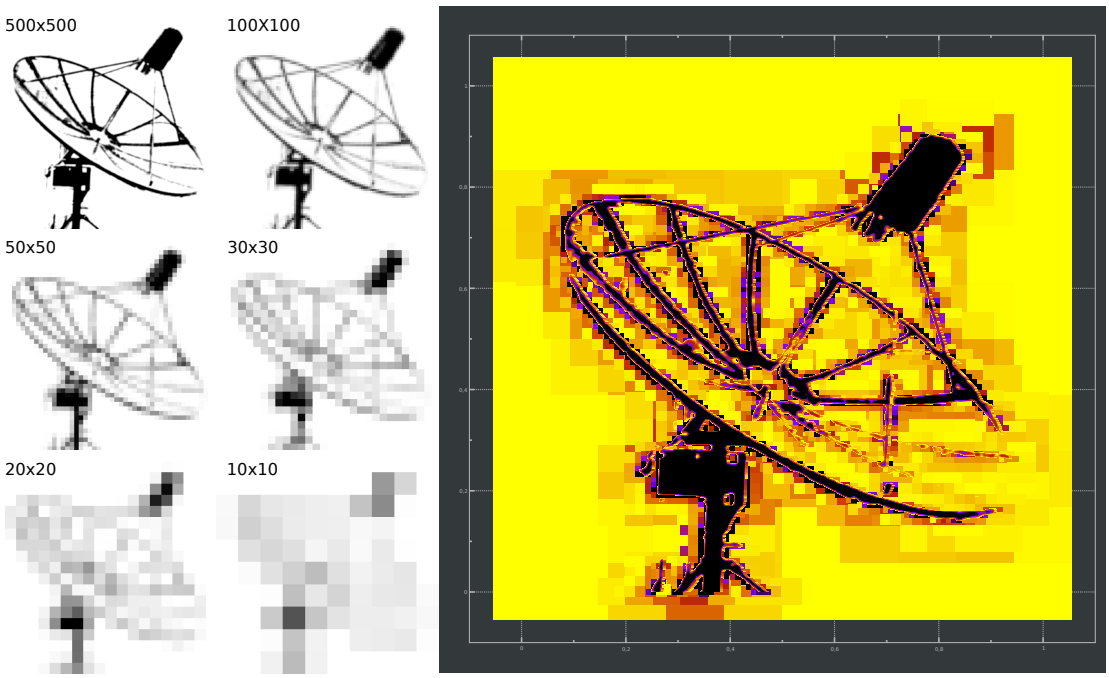


Figure 2.34: To demonstrate the functionality of the fast map drawing method, a cropped black-and-white image derived from *fig.* 2.15 was scaled to various dimensions via cubic interpolation to show different levels of detail. The pixel coordinates were mapped to unity interval and the white pixel values were discarded (i.e. considered transparent), except in the 10×10 dataset, which serves as the main background. The resulting point clouds were stacked in order of decreasing pixel scale, showing the same object at increasing levels of image resolution, rendered as differently sized squares.

2.10 Fast Map Drawing

The plotting widget implemented to display graphs that are shown in the Spectrum or History tabs of the GUI is also capable of displaying a map of three-column data, with the third column being encoded in colour.

This feature does not require the data to be pre-gridded, but accepts a sparse point cloud of any shape, which is rendered as pixel-like rectangles to form the visual representation (*fig.* 2.34). These are placed centred on the xy-coordinates of the data and coloured according to a procedurally computed lookup-table based on the range of z-coordinates.

As the main aim of this functionality is to present the user with both a visually appealing and informative representation of the data, it is desirable to draw the map in such a way that no apparent gaps occur between the points. Conventionally, this is achieved by mapping the data onto a sensibly sized grid, followed by some kind of interpolation procedure to fill in any empty sections of the map surface. This approach is however often computationally expensive, as the quality of the interpolation mechanism, the volume of data, and the dimensions of the underlying grid can require a lot of processing time.

In order to provide a near real-time experience and high refresh rates, so updates to the dataset can be shown immediately (e.g. when creating a point map of the sky), a low-overhead, fast-drawing method was conceived and implemented. A survey of related literature revealed no results on an obviously similar approach, so the algorithm is outlined hereafter.

The following goals and tradeoffs were considered in the design:

- while not required to be on a grid, the data are assumed to be structured somewhat regularly
- a continuous surface is desirable, although it is acceptable to show gaps, if it cannot be avoided
- the dimensions of the rectangles shall be the same for all data
- the space between rectangles shall be visually acceptable; no gaps must be visible, if the dataset is actually on a regular grid
- the dimensions of the rectangles should be selected in a way that they fit most data, if the point distances are non-uniform
- it is preferable to select the size of the rectangles so that gaps are avoided, even if the distances of the underlying points show a varying trend
- it is preferable that a smaller section of very dense points is overdrawn, effectively lowering the local resolution, rather than accept visible gaps for most of the rendered surface

Note that all the above points are considered on a per-dataset basis, as the plotting widget is capable of rendering multiple sets concurrently. In this case, the range of the colour axis is determined for the whole set. In addition, global values for the rectangle dimensions are computed as well. These are used when no local dimensions were determined for the individual set to draw. This means that multiple-size rectangles may be present in the rendered map surface. There is however no reordering of stacking implemented in the widget itself, i.e. overdrawing of large details with fine details or vice-versa is dependent on the order of submissions of the datasets by the user.

The drawing procedure is as follows:

- when adding a dataset, configure the initial limits for the minimum and maximum numerical values and delta steps to be DBL_MAX, -DBL_MAX and DBL_MAX respectively
- iterate the data and compare the x, y, and optional z value against the current corresponding limits per axis
- if the current value is lower than the current minimum or higher than the current maximum, adjust the limits accordingly
- if an adjustment is made for the x or y axis, record the occurrence *once per datum per axis*
- compute the delta step for the x and y axis by dividing the determined range by the number of adjustments *diminished by one*, if at least one adjustment was made
- draw the dataset point cloud as rectangles centred at the datum xy coordinates, with the rectangle sized as per the deltas and coloured according to the z value, as determined by some type of lookup table or colour mapping function

Note: DBL_MAX refers to the maximum numerical value of the used floating point type.

The procedure outlined above does not take datasets of zero delta step in an axis into account. As there is no correct answer in how to determine the dimensions of the squares used to render the point cloud, this is left to the implementation. In case of the plotting widget which implements this method, the undefined delta step (marked by DBL_MAX) is set to the value of the other axis if available, otherwise the deltas are set to unity.

3. Lab Projects and Exercises

The following sections present and demonstrate the setup and execution of tasks which fit the setting of a radio astronomy lab course. They are selected such that they illustrate the fundamental characteristics of radio telescopes along with observational techniques, basic data handling, and evaluation. While there are many more projects which can be realised even with relatively small instruments, the selection below is intentionally kept at a level which allows the experiments to be completed by students with a time investment of at most a few hours.

Please note that the descriptions below only touch the surface of the particular topic, or the scientific implications of the results, and intentionally do not go into details regarding the underlying fundamentals.

With the exception of a concise list of other possible projects, the following sections are laid out on individual pages so that they can be handed out to students individually, if necessary.

Data and observations shown below were acquired using the simulator plugin, unless marked otherwise.

3.1 Antenna Beamwidth

The beam width of a radio telescope is the solid-angle measure at the half-power point of the main lobe of the antenna pattern (the HPBW). It corresponds to the central diffraction maximum of an optical telescope and thus to the spatial resolution of the instrument.

For a circular aperture, the HPBW can be estimated from the Rayleigh criterion. For a typical small instrument operating around the spectral line of neutral hydrogen at $\lambda = 0.21$ m and an aperture of D $= 3$ m, the baseline resolution can be expected to be

$$\theta = \arcsin\left(1.22\,\frac{0.21\,\mathrm{m}}{3\,\mathrm{m}}\right) \approx 5^\circ \quad .$$

The beam of a real telescope is shaped by the influence of multiple factors, such as feed support struts, reflector imperfections, and dish illumination.

A simple method to characterise the HPBW is to move the telescope in a continuous scan across a bright radio *point source*. The data acquired by this procedure represent the beam pattern of the telescope in the given scan plane.

The recorded beam pattern also aids in finding pointing errors, indicated by the shift in the signal peak compared to the expected zero-point of the measurement. The height of the peak can also be used to determine the optimum antenna focus by observing the same source at a number of focus distance settings. As the feed position is shifted along the main axis of the dish, the measured peak power fraction in the main beam decreases, while the power in the side lobes increases, until finally the beam pattern collapses completely.

3.1.1 Procedure

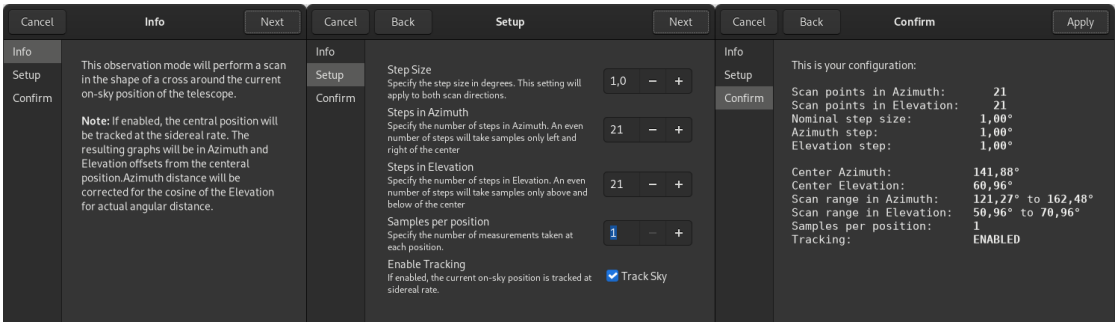


Figure 3.1: The setup wizard for the cross-scan observation guides you through the configuration process. Choose a set of parameters that suits your expected beam size.

This experiment requires a bright point source emitting *continuum radiation*. The size of smaller facilities (such as the SRT) effectively limits the available sources to the Sun. Under ideal conditions, Cygnus X [36] or certain geo-synchronous satellites could be used as well, although they require disproportionate effort to achieve a successful measurement of the beam, as the achievable SNR would be much lower.

In preparation of the measurement, wait until the target object is high enough in the sky to minimise incident radiation from the ground or other nearby objects, and to maximise the available offset for the acquisition of a stable baseline.

Since this is a continuum observation, set a frequency of at least 1 MHz *away* from the hydrogen line (1420.4 MHz) in the Spectrometer tab.

Point the telescope to the source using the Sky View. This also enables tracking of the target.

Now select the *Cross Scan* programme from the Observation tab and set up the parameters appropriately. You can use *fig.* 3.1 as a guideline, but make sure to adapt the configuration to the estimated beam size of your instrument (see above).

When the sequence is started, the software will execute the observation programme automatically by scanning the source in horizontal and vertical direction in order to characterise the beam in two directions. As the scan progresses, you will see measurements being added to the output graphs. Note that the relative azimuth angles are automatically corrected by the cosine of the elevation during the scan.

3.1.2 Data Evaluation

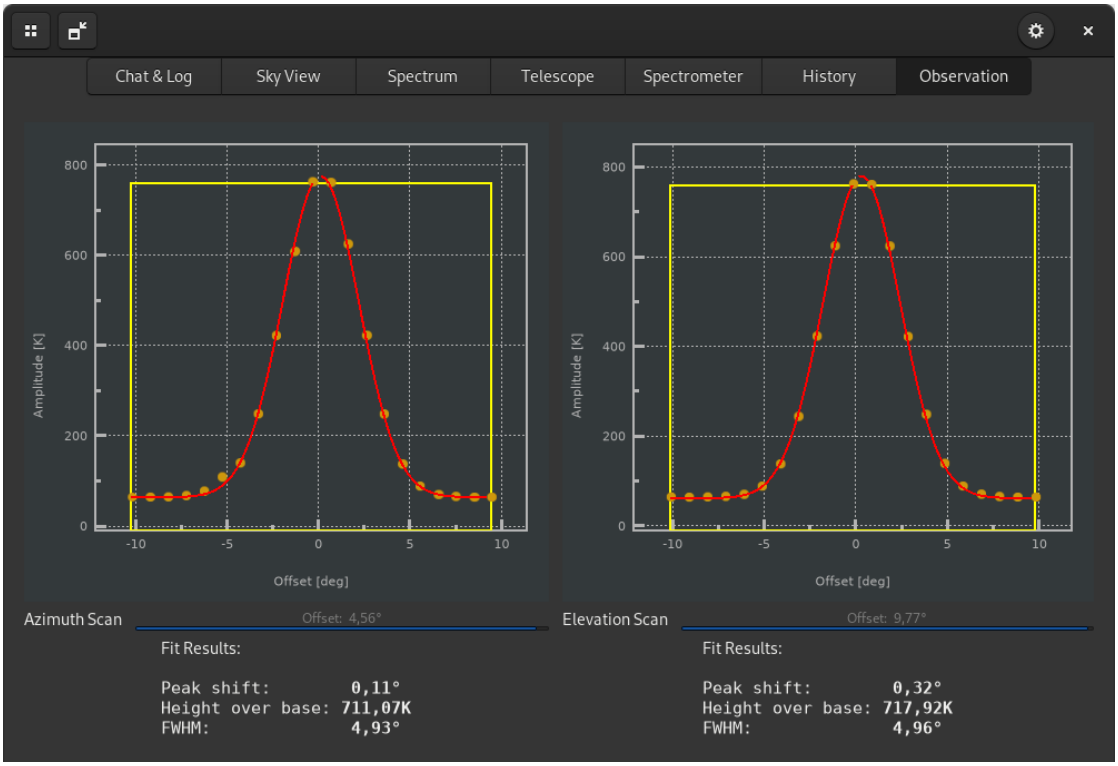


Figure 3.2: An example cross scan observation performed on the simulated Sun for a dish of D = 3 m. When completed, a Gaussian is fitted to the recorded data for both scan planes. The fitting results are shown below the graphs.

When the scan has completed, Gaussians are fitted to the datasets and the peak shifts in azimuth and elevation are determined (*fig.* 3.2), which may indicate the need for a pointing correction for the instrument. The beam width can be found in the FWHM parameter box of the plot window. In addition, you can export the data by right-clicking the plot for further evaluation. For example, one could use an external program (*fig.* 3.3) to fit a Bessel function (1.3.9), which is a more precise description of an (ideal) beam pattern.

If the Sun was used as target, the measurements can be used to determine the aperture efficiency (3.3).

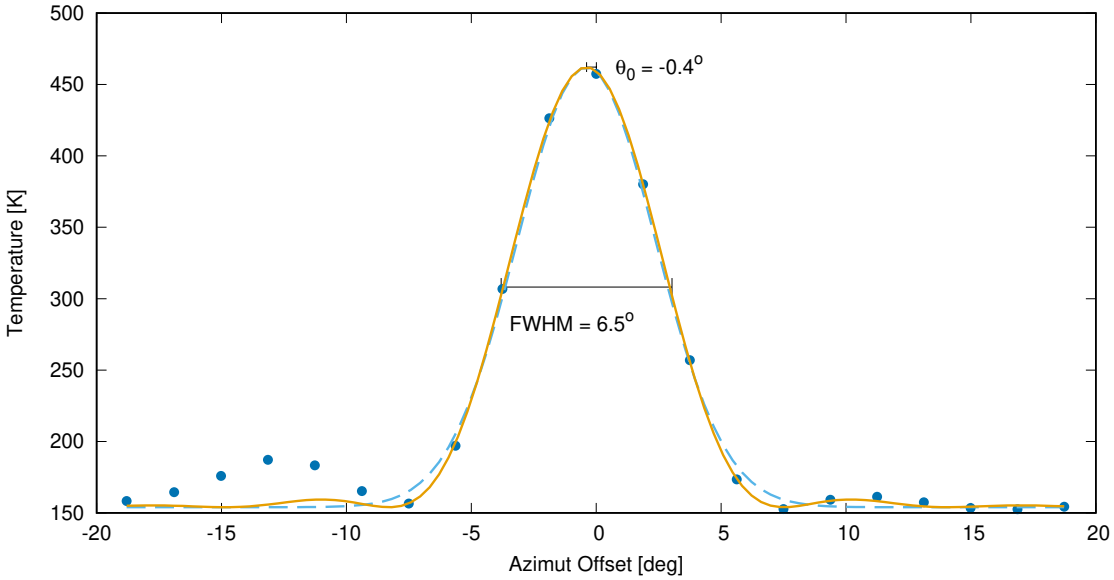


Figure 3.3: An example beam measurement of a real instrument. When compared to the radial cross-section of the Airy pattern (solid curve), it is apparent that a Gaussian (dashed curve) is a sufficient approximation of the main beam. Note that the increased signal at the negative offset is most likely not coming through a side lobe, but from a nearby object in the scan line, which is partially covered by the main beam. The slight increase at the positive offset is more likely to be a side lobe component.

3.2 Aperture Efficiency via Flux Comparison

Depending on design parameters of a radio antenna, signal losses may amount to a significant fraction of the theoretical limit (1.3.10). This means that only a fraction of the power from an observed source will reach the receiver. The goal of this experiment is to determine that fraction.

The effective aperture η_{ap} (1.3.11) is defined as the ratio of the effective aperture A_e to true aperture A_p, i.e. what remains after feed and support blockage, spillover, and under-illumination (1.3.8) due to the radiation pattern (1.3.9), as well as surface structure (e.g. mesh size) (1.3.7) and irregularities are taken into account.

One way to find the aperture efficiency is to compare a well known source flux density S to the measured incremental antenna temperature ΔT_a for that particular source.

Since this is effectively a total power measurement (1.5), a continuum source, of which the flux density is known, has to be observed.

There are several such sources available to small radio telescopes. The Sun, the Cygnus-X [36] star-forming region, Cassiopeia A [37] supernova remnant or even the Moon (2.8.9) are possible choices. Depending on the flux density of the source (*tbl.* 3.1), a lot of total integration time may be required to achieve a good value for the standard deviation T_σ.

In this procedure, measurements are acquired by executing beam switching cycles to reduce the noise contribution of the local environment.

Source	Expected T_a [K]	Flux Density [Jy]
Sun	900 - 4600	$6 \cdot 10^5 - 3 \cdot 10^6$
Cygnus X	12	8000
Cassiopeia A	3	2000
Moon	1.5	1000

Table 3.1: A list of continuum sources that are available to small instruments and their approximate fluxes. The expected antenna temperatures are calculated for a dish of D $=$ 3 m and an efficiency $\eta_{\mathrm{ap}} = 0.6$, assuming point sources.

3.2.1 Procedure

This experiment requires a point source emitting *continuum radiation*. Select a target from *tbl.* 3.1 or any other suitable object of your choice.

In preparation of the measurement, wait until the source is high enough in the sky to minimise incident radiation from the ground or other nearby objects, and to maximise the available offset for the acquisition of a stable baseline.

Since this is a continuum observation, set a frequency of at least 1 MHz *away* from the hydrogen line (1420.4 MHz) in the Spectrometer tab. If possible, use a frequency band for which reference flux data are available. If you use the Sun as source object, make sure to retrieve recent flux data, as solar activity can change significantly over time. The NOAA Space Weather Prediction Center supplies daily local noon solar radio flux values from multiple stations around the world.

Point the telescope to the source using the Sky View. This also enables tracking of the target. Now select the *Beam Switching* programme from the Observation tab and set up the parameters appropriately. Make sure to set the offset configuration to be at least one beam size.

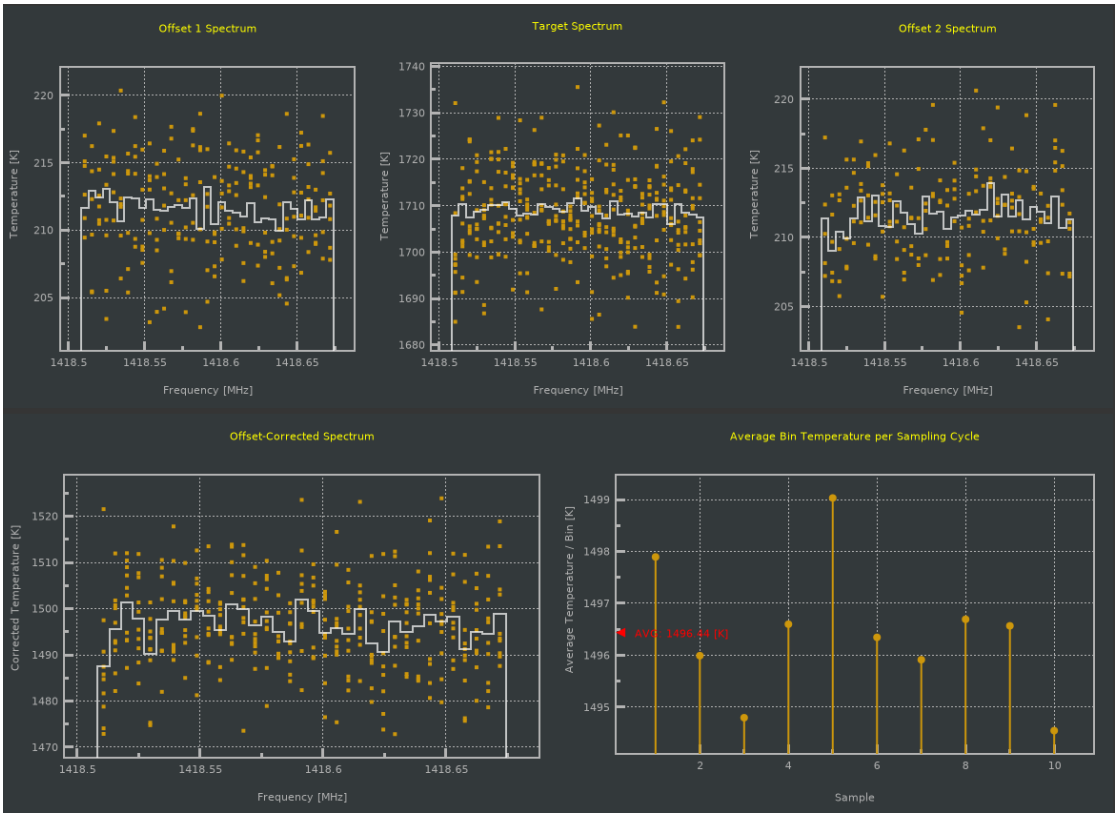


Figure 3.4: Beam switching performed on the simulated Sun (at 100 sfu) with a dish of D = 3 m shows the source signal to be about 1475 K above background. For a flux density of 10^6 Jansky, the aperture efficiency is $\eta_{ap} = 0.58$.

3.2.2 Data Evaluation

Beam switching works by comparison of the signal received at an offset position with the signal at the target position. This is done by either steering the beam to the offset (usually using a secondary steering reflector) or by pointing the telescope itself. To find the average antenna temperature T_a, the difference between the signal amplitudes of on/off source pairs are taken. The average temperature T_μ of these pairs is given as part of the observation programme in the lower right hand plot (*fig.* 3.4). To find the standard deviation T_σ, the data can optionally be exported via the right-click menu and evaluated in external software.

The aperture efficiency η_{ap} can be calculated from the formula given in 1.3.11. Using a simulated example for the Sun at 100 sfu, a dish of D = 3 m, and a measured average signal of 1475 K we get:

$$\eta_{ap} = \frac{2k_B\,1475[K]}{100 \cdot 10^4[Jy]\,1.5^2\pi[m^2]} = 0.58 \quad .$$

For an observation of the Cygnus X region (*fig.* 3.4) using the SRT of D = 2.3 m and an average signal of 7.6 K we get:

$$\eta_{ap} = 0.63 \quad .$$

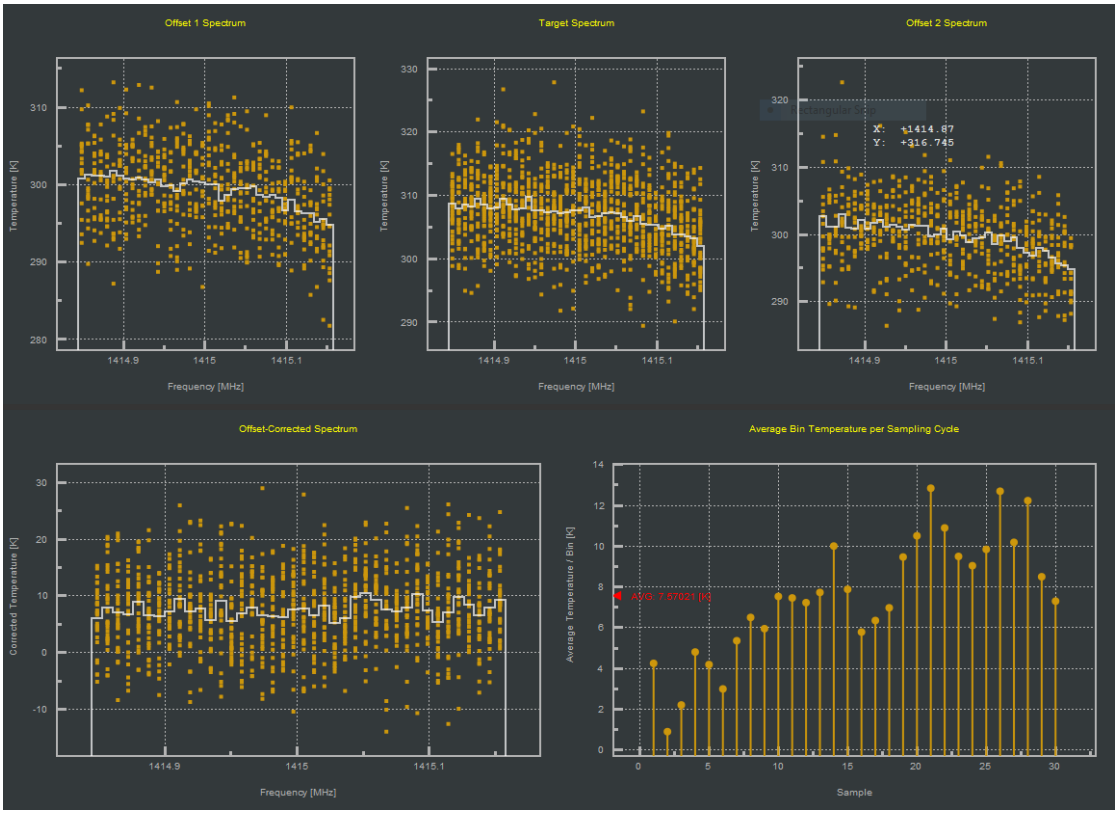


Figure 3.5: Beam switching performed on the Cygnus X star forming region with the SRT (D = 2.3 m) shows the source signal to be about 7.6 K above background. For a flux density of 8000 Jansky, the resulting aperture efficiency is $\eta_{ap} = 0.63$.

3.3 Aperture Efficiency via Beam Dilution

An alternative to the method of section 3.2 is to find the approximate value of η_{ap} by comparing the measured to the expected temperature of the Sun. For this exercise, one first needs to know the current apparent solar temperature in the observed wavelength band. This temperature depends on solar activity, the apparent size of the solar disk and the beam size. During the 11 year cycle, the radio flux in the L-band can vary up to 3 magnitudes (*fig.* 3.6). These variations are caused by changes in temperatures in the lower corona and the chromosphere, which affect the densities of free electrons. This changes the plasma oscillation frequencies (1.1.1) and thus the refractive properties in the layers above the photosphere.

3.3.1 Procedure

Find the beamwidth as described in section 3.1, record the height of the peaks above the baseline and take the average of both values. Determine the approximate angular size of the Sun at the observed wavelength from *fig.* 3.7 in units of degrees (1 solar radius corresponds to 0.25°) and estimate the expected temperature T_{Sun} from *fig.* 3.6 by making an educated guess, taking the current progression of the solar cycle into account.

3.3.2 Data Evaluation

To find the aperture efficiency, first calculate the beam dilution d, which is the ratio of the *angular size* of the Sun to the beam width:

$$d = \frac{A_{\mathrm{Sun}}}{A_{\mathrm{Beam}}} = \frac{r_{\mathrm{Sun}}^2}{r_{\mathrm{Beam}}^2} \quad .$$

Divide the mean height of the measured peaks by the dilution factor to find the true undiluted temperature for the Sun:

$$T = d^{-1} T_{\mathrm{Peak}} \quad .$$

The approximate aperture efficiency η_{ap} is then:

$$\eta_{ap} = \frac{T}{T_{\mathrm{Sun}}} \quad .$$

At 21cm, the solar radius will be about 1.2 times (*fig.* 3.7) larger than that of the visual disk, so $r_{\mathrm{Sun}} = 0.3\,\mathrm{deg}^2$. Using the values from the observation in *fig.* 3.2, $r_{\mathrm{Beam}} = 2.473\,\mathrm{deg}^2$, so $d \approx 0.015$ and thus $T = 47633\,\mathrm{K}$.

From *fig.* 3.6 one can estimate $T_{\mathrm{Sun}} = 10^5\,\mathrm{K}$, so the efficiency is

$$\eta_{ap} \approx 0.48 \quad .$$

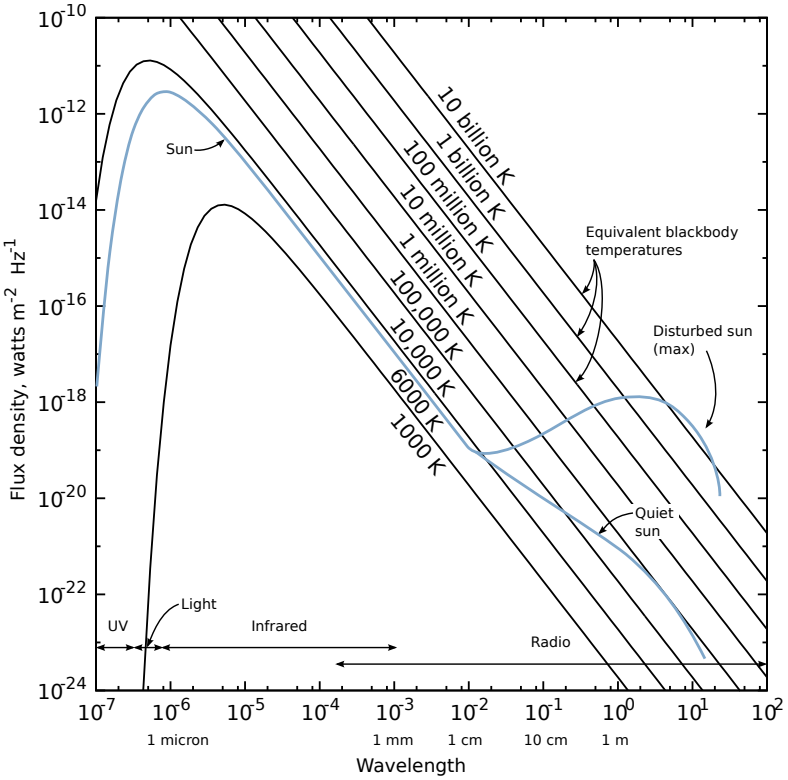


Figure 3.6: The solar spectral profile (blue curve) from the ultraviolet to radio wavelengths. The flux density is equal to that of a blackbody radiator at 6000K for wavelengths less than 1 cm, but deviates at longer wavelengths. In the radio, the observed flux varies with solar activity (*fig.* recreated from [38]).

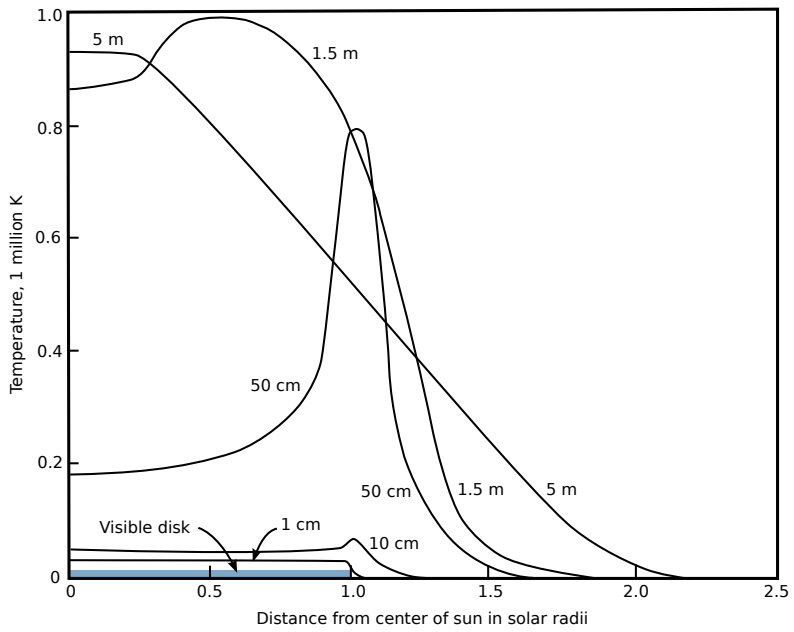


Figure 3.7: The predicted variation of the effective temperature with distance from the centre of the visual solar disk (blue) at different radio frequencies. Note that these curves are derived by ray-tracing a model of the solar corona for a given wavelength via the refractive index determined by the *plasma oscillation frequency* (1.1.1) for the given local electron densities (*fig.* recreated from [32]).

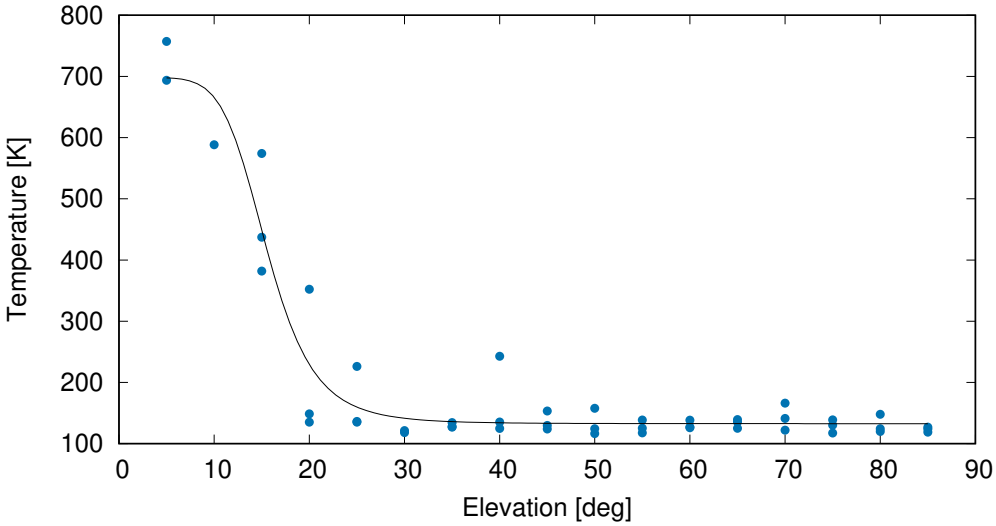


Figure 3.8: The spillover curve shown indicates significant contributions of signal coming from the ground up to elevations of 20°. This indicates partial illumination of the surroundings by the main beam, in addition to the radiation picked up by the side lobes.

3.4 Spillover

In signals recorded by radio telescopes, a certain amount of the detected flux is contributed by the local environment, primarily through the side lobes and the partial illumination of the ground by the radiation sensitivity pattern of the feed, called the *spillover* (1.3.8).

For calibration purposes of the system temperature, typically a fixed amount of spillover temperature is considered and assumed to be constant above a certain elevation. For elevations lower than the given threshold where the operating conditions of the instrument are considered optimal, the actual amount of spillover (in particular the contributions via the side lobes of the dish) is however highly variable. To determine proper angular limits suitable for calibration and observation, the following procedure is used to find the characteristic elevation profile.

3.4.1 Procedure

Since this is a continuum observation, set a frequency of at least 1 MHz *away* from the hydrogen line (1420.4 MHz) in the Spectrometer tab and avoid having any continuum sources (sky objects, buildings, trees) close to the vertical scan line.

Perform a tipping scan by recording the average temperature in steps of one beam size (or smaller) from the lowest to the highest possible elevation. Either use the History tab and write down the average signal manually or use the *record* feature of the Spectrum tab for recording and later external evaluation of the data. At each point, check the radio spectrum for any significant spikes or unusual gradients in the baseline of the spectrum, which could indicate interference from a local radio source.

3.4.2 Data Evaluation

Plot the elevation against the measured system temperature (*fig.* 3.8). The distribution of the points will most probably resemble an inverse power law. Optionally, you can fit an adequate function to the data. Determine the optimum minimum elevation for observations, which is where the measured temperature becomes approximately constant.

3.5 Galactic Plane Velocity Diagram

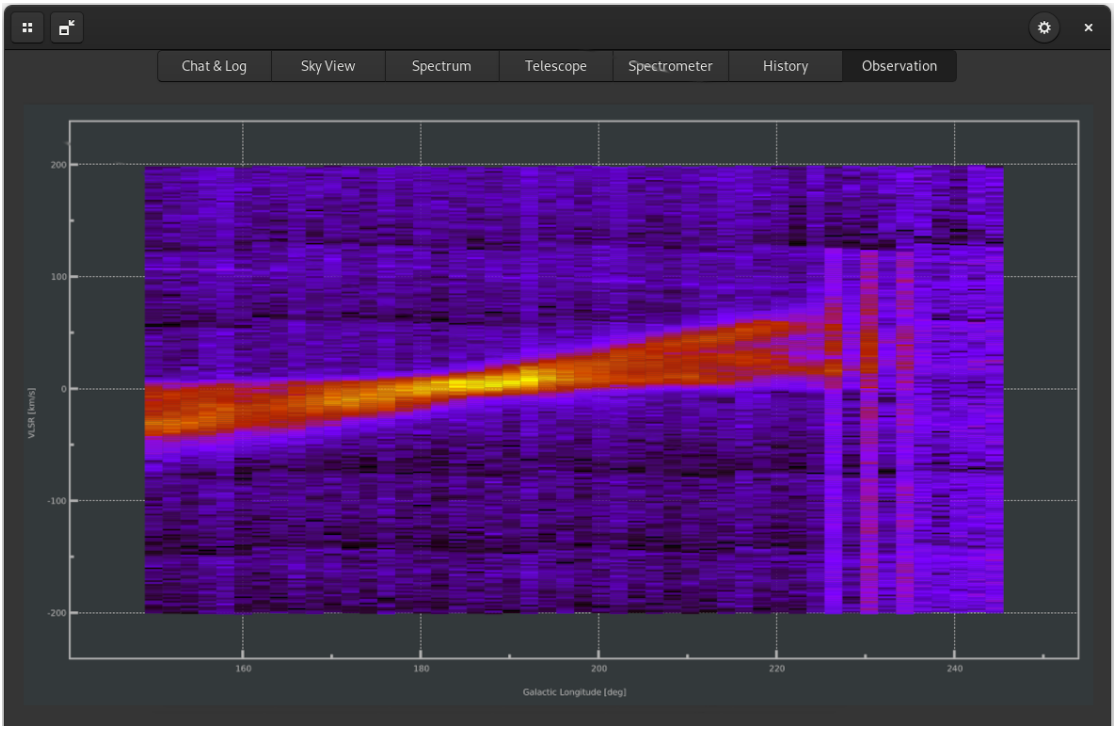


Figure 3.9: A position-velocity diagram recorded between galactic longitudes 150° to 242° with the SRT using a step size of 2°. Note that spillover severely degrades the SNR from about 246° onwards.

Due to its abundance and spatial distribution, neutral hydrogen and its 21cm line emission is an excellent source of radial velocity data and provides a lot of insight on the kinematics of the Milky Way. The aim of this experiment is to create a position-velocity diagram of the HI gas in the galactic plane, by mapping the emissions around 1420.405 MHz. The observed line profiles will present prominent Doppler shifts, which are due to two major effects. One is due to the kinematics of the Milky Way and gives the radial velocity, the other is a seasonal component which results from the motion of the Earth around the Sun.

3.5.1 Procedure

This is a line observation for the reference rest frequency of 1420.405 MHz, so it is easiest to select this reference in the Spectrometer tab and choose the *velocity* input mode rather than frequency. Select *center-span* for the bandwidth mode and use 0 km/s for the centre, with a span of at least 300 km/s. To automatically compensate for the seasonal component and other systematic effects in velocity, you should also enable *Doppler tracking*. This will automatically adjust the centre frequency during the scan, so the radial velocity of the spectra is corrected to the VLSR. Select *Galactic Plane Scan* from the Observation tab and configure the observation. Choose the galactic longitude limits according to the visible sky. You may use a step size smaller than your beam for smoother gradients in the diagram.

3.5.2 Data Evaluation

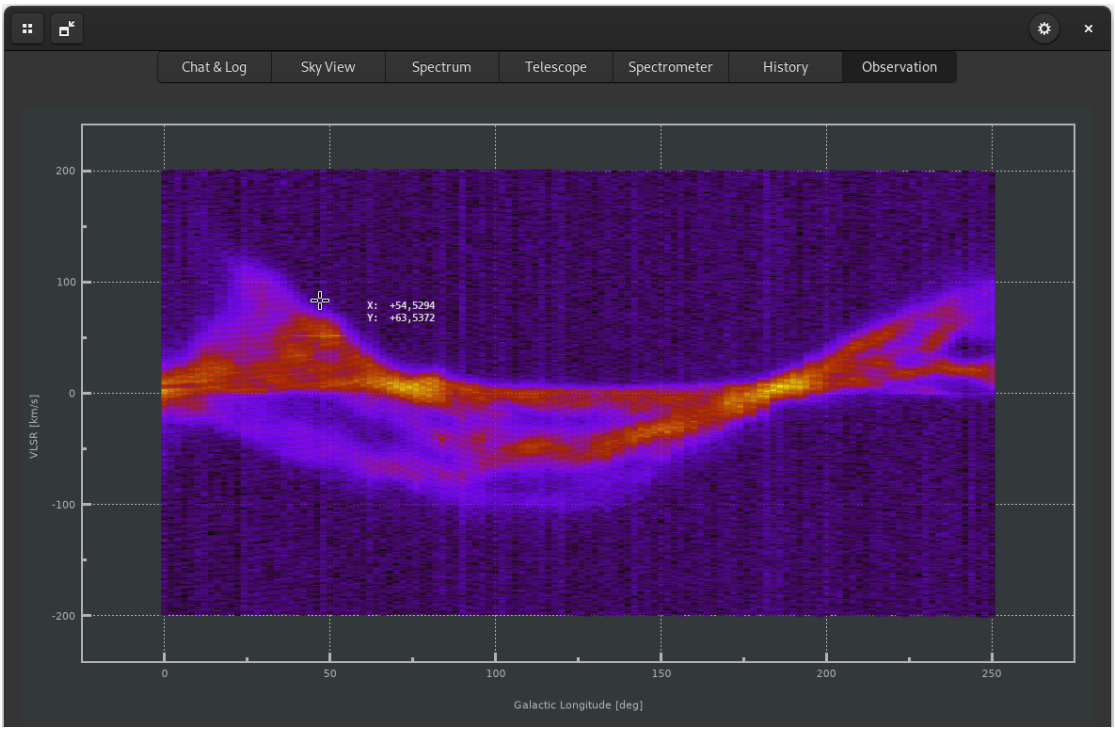


Figure 3.10: A position-velocity diagram recorded between galactic longitudes 0° to 250° for a simulated dish of D = 3 m using a step size of 2°. Note that the mouse cursor position shows the velocity and longitude for immediate basic analysis.

The observation programme will automatically create a position-velocity map for you. The spectral amplitudes are encoded in colour. The result should look similar to *fig.* 3.10 or *fig.* 3.9. You may optionally export the line data for re-use in section 3.7 via the right click menu and immediately proceed with the data evaluation in section 3.6.

3.6 Galactic Rotation Curve

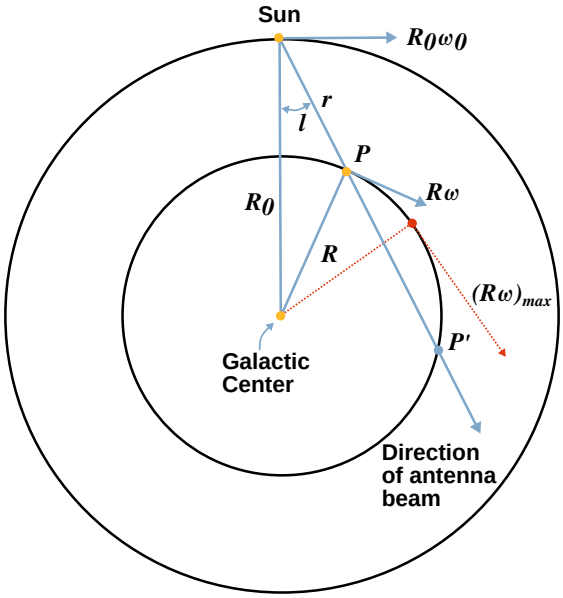


Figure 3.11: A simple model of galactic rotation as circular motion with decreasing angular rate for increasing distance from the centre. The maximum velocity of an orbit is tangential to the line of sight (*fig.* recreated and adapted from [39]).

The galactic rotation curve is the radial velocity of the components that make up a galaxy, such as gas, stars and dust. If all visible matter was concentrated near the centre, spiral galaxies would show velocity distributions corresponding to *Keplerian* rotation, i.e. matter closer to the centre would always rotate around faster than towards the outer regions. However, one finds that velocities remain about constant with increasing radius, as can be seen in the measured rotation curves. This characteristic implies that the mass is spread throughout the disk and increases in proportion to the radius:

$$v(R) = \left[\frac{G\,M(R)}{R}\right]^{\frac{1}{2}} \quad .$$

The aim of this project is the creation of a rotation curve for the Milky Way. For this purpose, a simple model of rotation (*fig.* 3.11) is used. The distance R of a point P from the centre for a given longitude l in the galactic plane, for which the maximum velocity $v_{\max}$ occurs on a circular orbit at line of sight (the direction of the antenna beam), is given by [39]:

$$R = |R_0\,\sin(l)|\ ,$$

with $R_0 = 8.5\,[\mathrm{kpc}]$ being the Sun's distance from the Galactic Centre. At location P, the tangential velocity is related to the *observed* velocity by a correction of Sun's own rotational velocity $v_{Sun} = R_0\,\omega_0 = 220\,[\mathrm{km/s}]$ as

$$v(R) = |v_{obs_{\max}}| + v_{Sun}\ |\sin(l)| \quad .$$

3.6.1 Procedure

Record a velocity diagram as described in section 3.5. Alternatively, you can set up the spectrometer the same way, but manually point the telescope to different galactic longitudes.

3.6.2 Data Evaluation

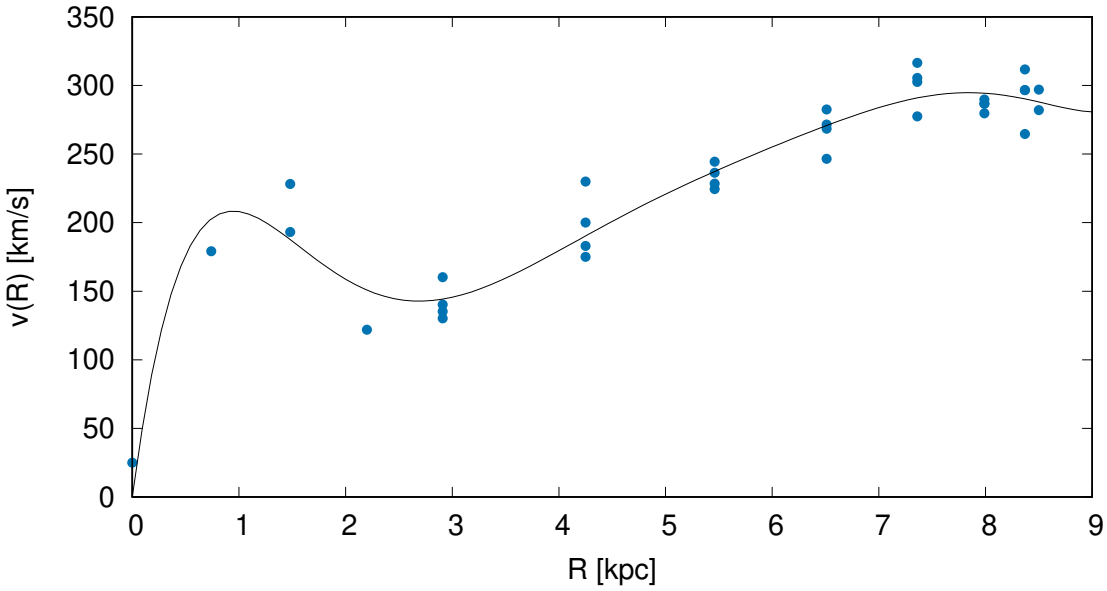


Figure 3.12: A galactic rotation curve measured for a simulated dish of D = 3 m. The data are fitted via a high-order polynomial.

To create a rotation curve profile, you can use the mouse cursor (*fig.* 3.10) to take measurements of the *maximum absolute* velocity envelope of the hydrogen spectrum at a given longitude. If you chose manual pointing, you can use the mouse cursor in the Spectrum tab to determine the greatest velocity. Note down the corresponding longitudes and velocities, calculate R and $v(R)$ and create a plot of the data (*fig.* 3.12). For an estimate of the mass of the Milky Way, you can calculate the enclosed mass $M(R)$ within a radius R.

3.7 Spiral Arm Structure

The major part of the Milky Way's hydrogen gas is concentrated in the spiral arms. Using Oort's constants, we can create a plot of the approximate spiral arm locations. This simplistic approach assumes uniform circular motion of the gas around the Galactic Centre without proper motion.

The Oort constants A and B are empirically determined constants relating to the Sun's distance and motion relative to the Galactic Centre where

$$A = \frac{1}{2}\left(\frac{V_0}{R_0} - \frac{dv}{dr}\right) = 15.3 \pm 0.4\,[\,\mathrm{km\ kpc^{-1}\ s^{-1}}]$$

and

$$B = \frac{1}{2}\left(\frac{V_0}{R_0} + \frac{dv}{dr}\right) = -11.9 \pm 0.4\,[\,\mathrm{km\ kpc^{-1}\ s^{-1}}]\,,$$

as derived from Gaia DR1 data (see [40]).

The radial velocity of a point in the solar neighbourhood at a distance r from the Sun and a galactic longitude l from the Galactic Centre (*fig.* 3.11) is derived from a triangle spanned between these points (e.g. [39]) and using Oort's constant A is

$$v_{rad} = r\,A\,\sin(2l) \quad .$$

For the purpose of this exercise we have to express r:

$$r = \left| v_{rad}\,(A\,\sin(2l))^{-1} \right| \quad .$$

When calculating distances, be aware that this function has poles at multiples of 90 degrees.

3.7.1 Procedure

Measure spectra at multiple points along the galactic plane for a spectrometer setup as described in section 3.5 or use previously recorded data from that exercise.

3.7.2 Data Evaluation

Fit Gaussians to the lines in the spectra, or simply use the velocity values of the peaks at the observed galactic longitudes. Calculate the distances using the relation above and create a plot of the locations around the location of Sun (*fig.* 3.13). Compare with the Milky Way spiral arm structures proposed in literature.

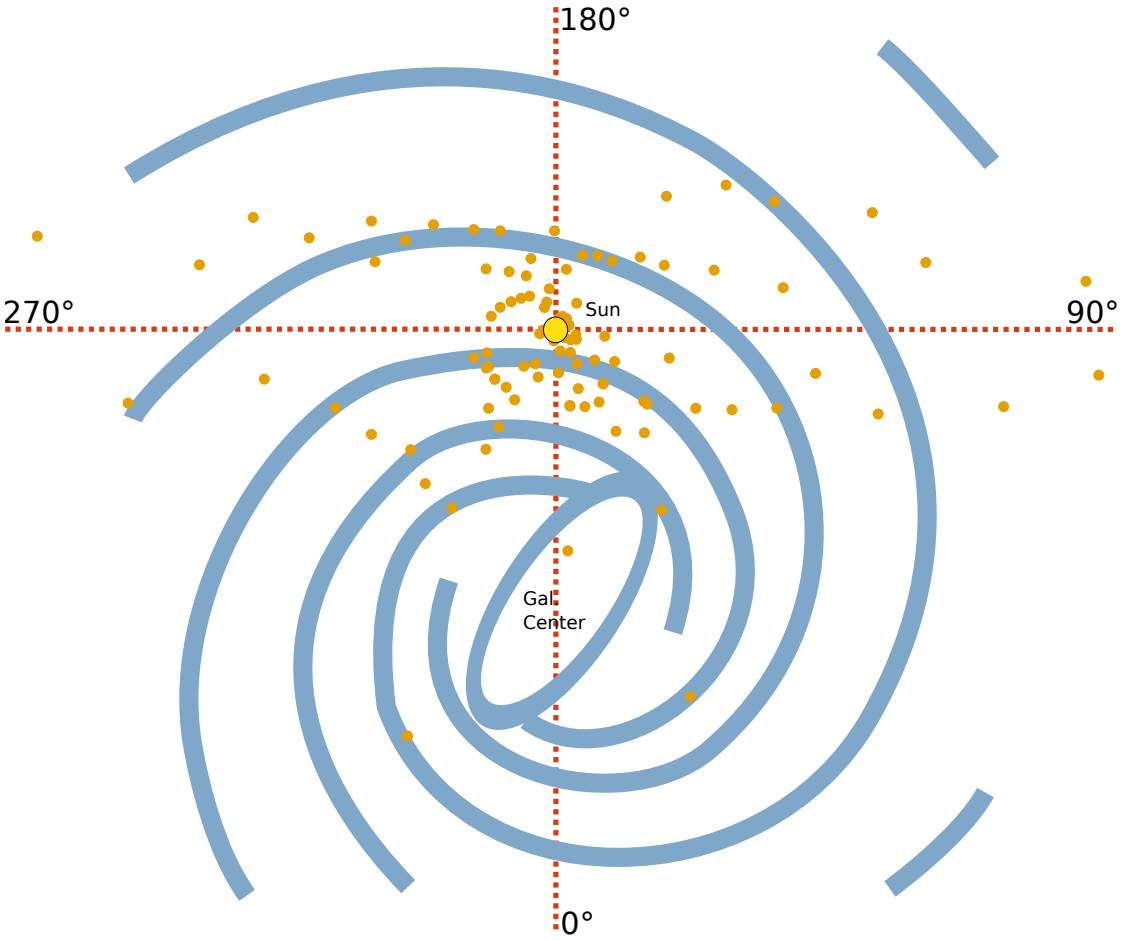


Figure 3.13: The measured distribution of spiral arms (dots) as measured with a simulated dish of D = 3 m. Proposed locations of spiral arms (solid lines) were taken from [41].

3.8 Map-Making

Radio telescopes, in particular those which operate at lower frequencies, are essentially single-pixel instruments, as they usually cannot employ arrays of feed antennae for practical reasons. The only method to create spatial images is therefore the piece-wise scanning of a source region and the reconstruction of the scan pattern by mapping it onto an image grid.

For small, low resolution instruments, there are essentially no extended sources available for mapping except the Milky Way, but even that may not give a satisfactory image with fine structured details.

Instead of using an actual instrument, a much higher-resolution instrument can be emulated by the simulator plugin, which is available for the server component of the software. The base resolution of the available dataset is in fact detailed enough to allow the imaging of the *Magellanic Clouds*, which will be the objective of this exercise.

3.8.1 Procedure

First, make sure the client is set up to use a server which is configured with the simulator plugin. Look up information on the Magellanic Clouds in an astronomical database such as *SIMBAD*. Determine their location on the sky and consider the size of the grid needed to image both galaxies at the same time. Next, think about the geographical location that needs to be configured for the target region to be visible to the simulated telescope and set it up in the server user interface (*fig.* 2.22).

The *N-Point Map Scan* programme available in the Observation tab maps the level of the integrated signal. In order to remove the HI line from our own galaxy, take a look at the radial velocities of the LMC and SMC.

Configure the frequency and bandwidth settings in the Spectrometer tab so that the signal component of the Milky Way is not present in the spectrum. Since you can switch between frequency and relative velocity input, you can also set up the spectrometer by selecting the appropriate velocity limits for the HI rest frequency reference.

Select a beam size in the simulator configuration which allows the resolution of details, given the angular sizes of the target objects.

Finally, select the *N-Point Map Scan*, enter your desired observation parameters and start the program. Consider repeating it with different beams and step sizes.

3.8.2 Data Evaluation

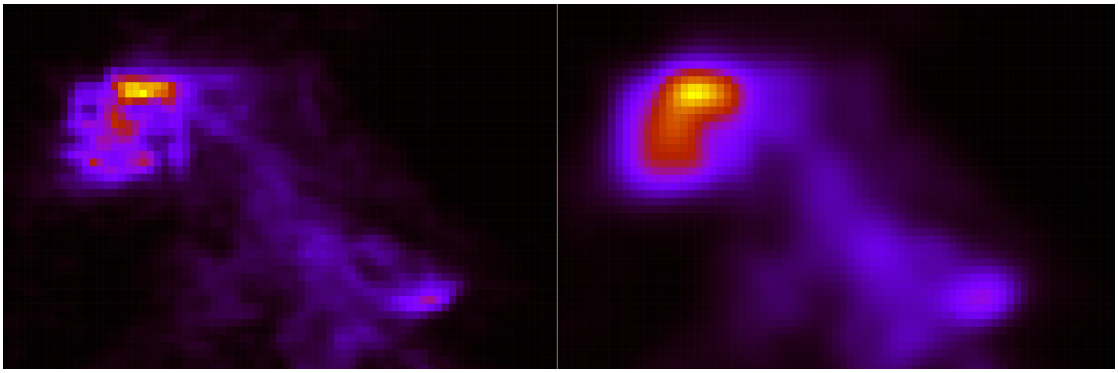


Figure 3.14: The Magellanic Clouds observed with simulated beams of 0.5° (left) and 2° (right) with a grid step size for both scans of 0.5°, recorded between galactic latitudes -50° to -25° and longitudes 270° to 310°.

The observation programme will automatically create a map (*fig.* 3.14) for you. The signal amplitudes are encoded in colour. If you repeated the scan with different settings of beams and step sizes, compare the resulting maps and consider the differences, in particular the effect on the image when large beam widths are used together with small step sizes, i.e. the effect of *supersampling* the beam.

3.9 Other Projects

This section briefly lists several projects which are less suited for a typical lab course, as they require much more time to execute, or are more detailed and thus less instructive with regard to more general concepts of radio astronomy.

3.9.1 All-Sky Drift Scan Map

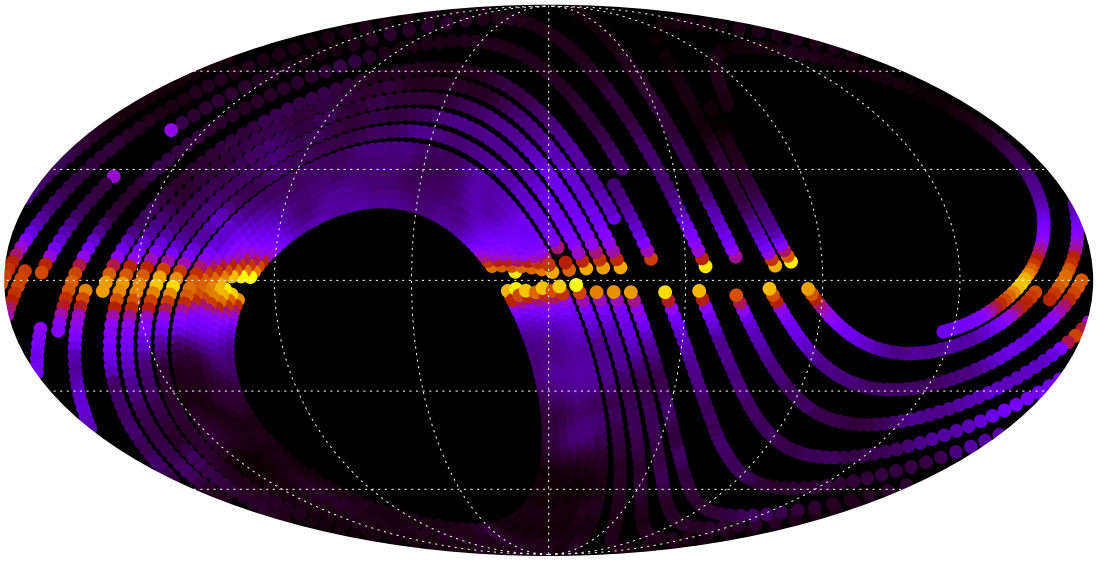


Figure 3.15: An example of (partial) drift scan lines for a simulated beam of 5° in *Mollweide* projection.

A drift scan map of the visible sky can be created by adjusting the declination of the instrument every (sidereal) day and letting the sky drift through the beam in right ascension. The recorded integrated signal can be re-binned to a reasonable grid size and plotted similarly to *fig.* 3.15.

3.9.2 Solar Radio Bursts

Solar radio bursts are transient phenomena, where the solar radio flux is temporarily significantly elevated above the baseline of the current phase of the solar cycle. These events are typically associated with the ejection of charged particles and magnetic reconnection, and can last multiple minutes, depending on their type and underlying process (e.g. [42, 43]). To detect bursts, it is typically sufficient to select a suitable frequency and record the continuum signal while tracking the Sun. Bursts will show as characteristic spikes in the data (*fig.* 3.16).

3.9.3 Radio Environment Map

The (thermal) emissions of the local environment (*fig.* 3.17) of a radio telescope can be captured by scanning the pointing range of the drive while recording the continuum flux. The resulting map can be used to construct a profile (2.9.2) of the local horizon for use with the software.

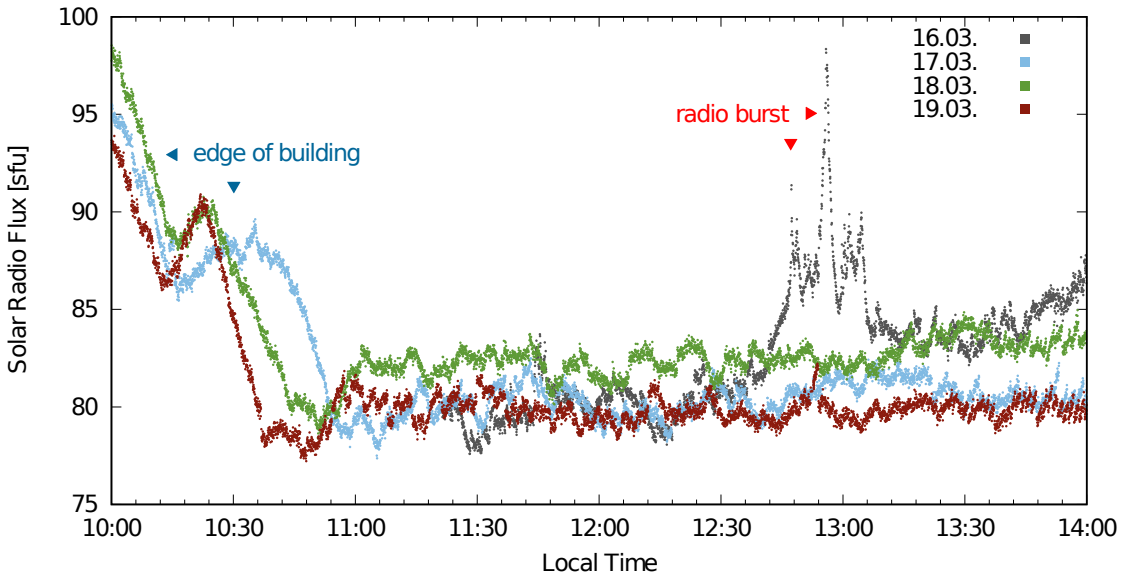


Figure 3.16: Solar activity was monitored for several days around local noon. A single radio burst event was recorded and is prominently visible. Note that the high levels of signal before about 10h30 come from the roof edge of the building next to the instrument. The distinct shape of the curves in this region are likely a case of *Fresnel diffraction* of the solar radio signal.

3.9.4 2D PSF reconstruction

A method of creating a map of a radio telescope's beam is to point it directly south and have the Sun drift through the meridian, while repeatedly scanning in elevation. The elevation reference zero-point is that of the Sun and thus changes during the observation. The beam is then reconstructed by computing the equivalent solar angular distance from the centre of the beam for a given measurement. An advantage of this method compared to just tracking the Sun while sampling the beam at different offsets, is control over the rotation of the field of view, which cannot be avoided in alt-azimuth mounts.

3.9.5 Instrument Spectral Response

The spectral response of the instrument represents the sensitivity over the frequency bandwidth and can be applied as a correction of a non-flat spectral baseline. To find this curve, the input to the receiver must be terminated appropriately, so only the internal characteristics of the electronics remain in the recorded signal. A typical response curve is shown in *fig.* 3.19.

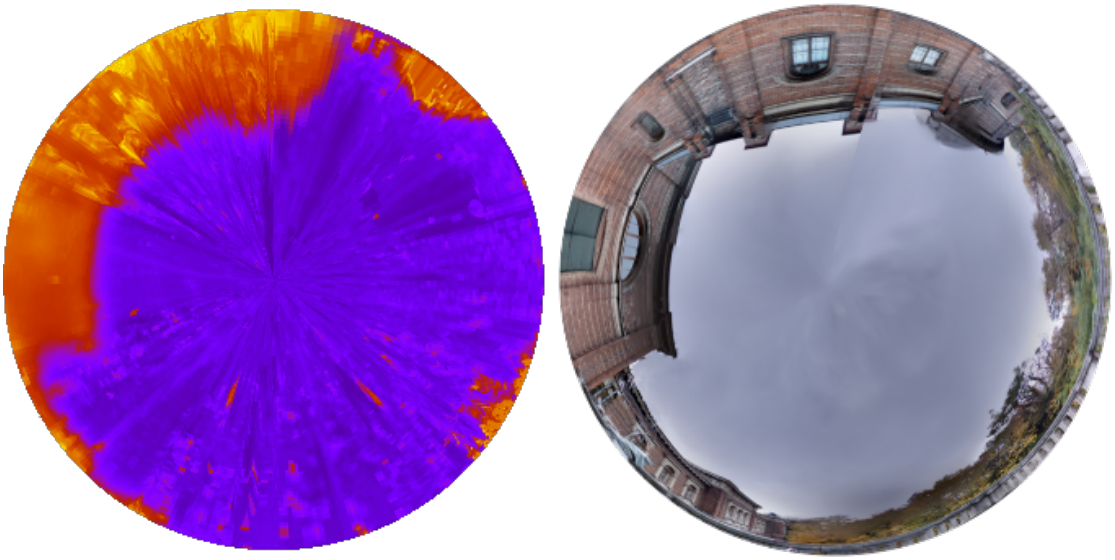


Figure 3.17: A map of the local environment of the SRT in radio is displayed on the left. For comparison, a fisheye-projected image of the optical view is shown on the right. The outlines of the building are clearly visible, even the shape of the telescope dome is recognisable, as are the emissions of a tall tree in the bottom right corner.

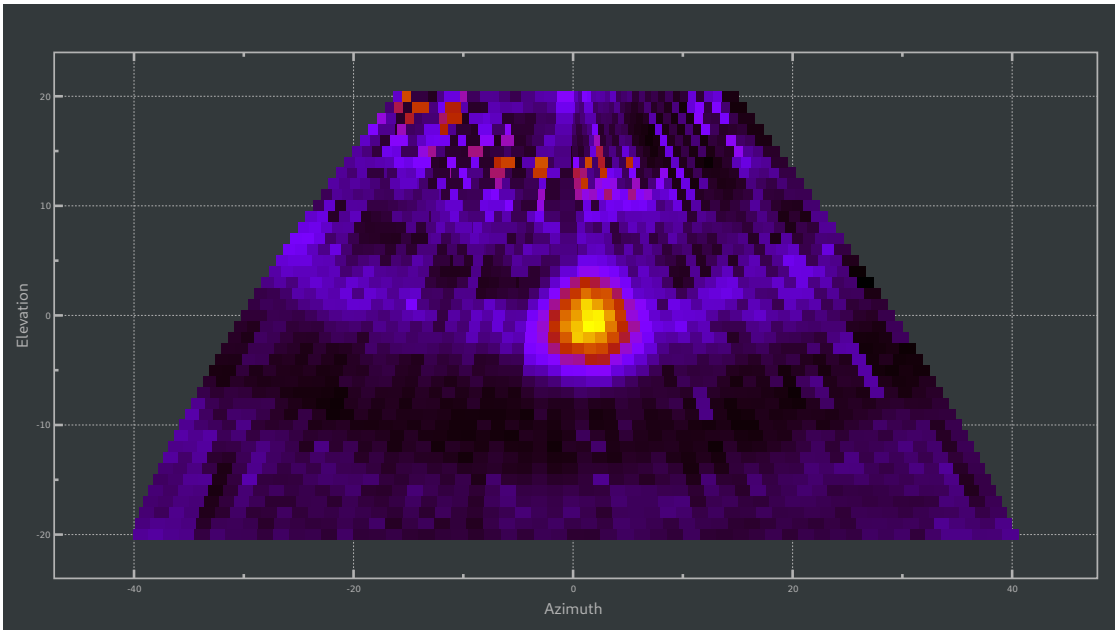


Figure 3.18: A map of the SRT beam captured by having the Sun drift through the meridian while repeatedly scanning in elevation. In the top region, radio reflections at a strip of sheet metal on the building wall are clearly noticeable, so is the variation background noise in elevation direction. Note that azimuth and elevation are reconstructed for the Sun's, not the telescope's zero reference. All scan lines are in fact strictly vertical and the upward curving features thus represent the inverse of the apparent motion of the Sun during the day.

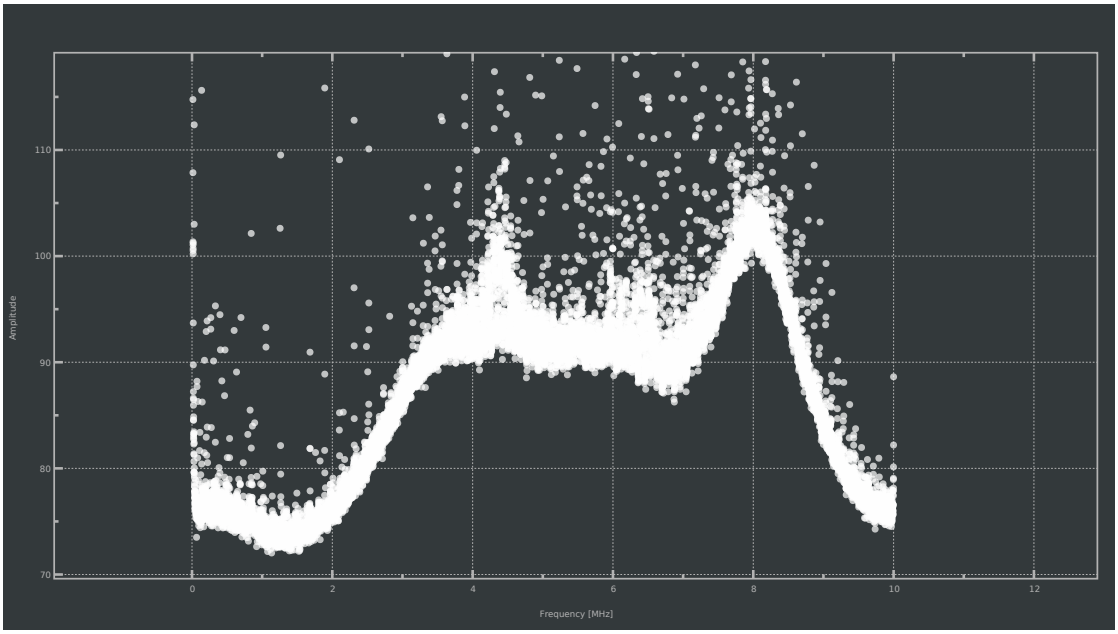


Figure 3.19: The characteristic spectral response of an instrument in the intermediate frequency band. Note that the amplitude is in arbitrary units. To obtain the curve, the pickup antenna input of the pre-amplifier was terminated with a 50 Ohm resistor.

4. Outlook



Figure 4.1: A custom grid observation programme of a section of the galactic plane. A pulse generator instructs the range steppers to generate the coordinate outputs for a grid scan between galactic latitudes $\pm 10°$ and longitudes 120° to 240°, which are converted to the telescope's reference system and then sent to the instrument. A source node for spectral data provides output, which is filtered and the result is displayed. Note that this figure shows only a subset of planned functional nodes.

Several generations of students have already successfully used this software after only a concise introduction to the user interface, suggesting that the layout and intended workflow are sufficiently well developed. The addition of the simulator plugin turned out to be particularly useful as a first point of contact, since errors in the handling of the command interface are unavoidable in the beginning, but now do not result in a negative outcome. Obviously, the actual hardware driving plugins do not accept user inputs which could damage the instrument. Still, a bad selection of target coordinates or parameters for the automated observation programmes can result in long, unpleasant waiting periods, as a real instrument is constrained by its drive speed, but the simulator is not.

While the software has reached a decent level of maturity, more work will be done with time, e.g. porting it to new major versions of the *GTK+* GUI library. More features will also be added, such as a method of freely programmable functionality by use of modular, interconnected software components (nodes) using the *gtknodes* library (created by the author). A partial implementation already exists (*fig.* 4.1), but a full set of nodes is still to be defined.

Several other hardware backend plugins have been implemented, but were not documented here. These include the SPID MD01 rotor controller and acquisition devices such as the Measurement Computing PCI-DAS4020/12 or RTL2832U based USB receiver dongles. Full functional verification of those plugins in a closed-loop hardware setup is pending. They will be made available as part of the source tree in the future.

Bibliography

[1] Thomas L. Wilson, Kristen Rohlfs and Susanne Hüttemeister. *Tools of Radio Astronomy*. 5th ed. Springer, 2009, pp. 3–4.

[2] Ronald N. Bracewell and George W. Preston. 'Radio Reflection and Refraction Phenomena in the High Solar Corona.' In: *Astrophysical Journal* 123 (Jan. 1956), p. 14. DOI: 10.1086/146125.

[3] Michael C. Kelley. *The Earth's Ionosphere - Plasma Physics and Electrodynamics*. 2nd ed. Academic Press, 2009.

[4] D. A. Naylor et al. 'Broad-band spectroscopy with the James Clerk Maxwell Telescope using a polarizing Fourier transform spectrometer'. In: *Monthly Notices of the Royal Astronomical Society* 260.4 (Feb. 1993), pp. 875–882. ISSN: 0035-8711.

[5] P. R. Griffiths and J. A. de Haseth. *Fourier Transform Infrared Spectrometry*. 2nd ed. John Wiley and Sons, Inc., Hoboken, New Jersey., 2007, p. 28.

[6] William C. Jones et al. 'A Polarization Sensitive Bolometric Receiver for Observations of the Cosmic Microwave Background'. In: ed. by Thomas G. Phillips and Jonas Zmuidzinas. Vol. 4855. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. Feb. 2003, pp. 227–238.

[7] J. A. Garcia-Barreto et al. 'Magnetic Field Structure of the Star-forming Region W3(OH): VLBI Spectral Line Results'. In: *The Astrophysical Journal* 326 (Mar. 1988), p. 954.

[8] Thomas L. Wilson, Kristen Rohlfs and Susanne Hüttemeister. *Tools of Radio Astronomy*. 5th ed. Springer, 2009, pp. 85–86.

[9] J. Ruze. 'The effect of aperture errors on the antenna radiation pattern'. In: *Il Nuovo Cimento* 9.S3 (Mar. 1952), pp. 364–380.

[10] A. D. Olver. *Corrugated Horns for Microwave Antennas*. Electromagnetic Waves. Institution of Engineering and Technology, 1984.

[11] R. C. Hansen. 'A One-Parameter Circular Aperture Distribution with Narrow Beam-Width and Low Sidelobes'. In: *IEEE Transactions on Antennas and Propagation* (July 1976), pp. 477–480.

[12] Allan W. Love. 'Central Blocking in Symmetrical Cassegrain Reflectors'. In: *IEEE Antennas and Propagation Magazine (Antenna Designer's Notebook)* 34.1 (Feb. 1992), pp. 47–49.

[13] J. D. Kraus. *Antennas*. 2nd ed. New York: McGraw-Hill, 1988, p. 574.

[14] J. D. Kraus. *Antennas*. 2nd ed. New York: McGraw-Hill, 1988, p. 831.

[15] *ECSS-E-70-41A Space engineering - Ground systems and operations - Telemetry and telecommand packet utilization*. ESA Publications Division, 2003, pp. 207–212.

[16] L. Kimball and P. Pratap. 'A Small Radio Telescope for Undergraduate Instruction'. In: *American Astronomical Society Meeting Abstracts*. Vol. 193. American Astronomical Society Meeting Abstracts. Dec. 1998, 34.07, p. 34.07.

[17] Harald Chmela. *Monitoring and Analysis of Electromagnetic Interference in Radio Astronomy*. 2018. DOI: 10.25365/THESIS.53683.

[18] Rodolfo Montez. *Electronic Noise Calibrator for the Small Radio Telescope*.

[19] A. E. E. Rogers. *Firmware and software changes to support fractional counts from the motor drives, SRT Memo #022*.

[20] A. E. E. Rogers. *Pointing corrections due to azimuth axis tilt and lack of orthogonality of elevation axis, SRT Memo #002*.

[21] A. E. E. Rogers. *SRT receiver communication protocol, SRT Memo #012*.

[22] P. M. W. Kalberla, J. Kerp and U. Haud. 'H I filaments are cold and associated with dark molecular gas. HI4PI-based estimates of the local diffuse CO-dark H_2 distribution'. In: *A&A* 639, A26 (July 2020), A26. DOI: 10.1051/0004-6361/202037602.
[23] U. Haud. 'Gaussian decomposition of the Leiden/Dwingeloo survey. I. Decomposition algorithm'. In: *A&A* 364 (Dec. 2000), pp. 83–101.
[24] HI4PI Collaboration et al. 'HI4PI: A full-sky H I survey based on EBHIS and GASS'. In: *A&A* 594, A116 (Oct. 2016), A116. DOI: 10.1051/0004-6361/201629178.
[25] B. Winkel et al. 'The Effelsberg-Bonn H I Survey: Milky Way gas. First data release'. In: *A&A* 585, A41 (Jan. 2016), A41. DOI: 10.1051/0004-6361/201527007.
[26] P. M. W. Kalberla and U. Haud. 'GASS: The Parkes Galactic All-Sky Survey'. In: *A&A* 578 (June 2015), A78. ISSN: 1432-0746. DOI: 10.1051/0004-6361/201525859.
[27] Kalberla, P. M. W. et al. 'GASS: the Parkes Galactic all-sky survey - II. Stray-radiation correction and second data release'. In: *A&A* 521 (2010), A17. DOI: 10.1051/0004-6361/200913979.
[28] N. M. McClure-Griffiths et al. 'Gass: The Parkes Galactic All-Sky Survey. I. Survey Description, Goals, and Initial Data Release'. In: *ApJS* 181.2 (Apr. 2009), pp. 398–412. DOI: 10.1088/0067-0049/181/2/398.
[29] James W. Cooley and John W. Tukey. 'An algorithm for the machine calculation of complex Fourier series'. In: *Mathematics of Computation* 19.90 (May 1965), pp. 297–297. DOI: 10.1090/s0025-5718-1965-0178586-1.
[30] Ronald N. Bracewell. *The Fourier Transform and its Applications*. Third. McGraw-Hill, 2000, pp. 269, 275–278, 283–285.
[31] W. C. Daywitt. *Error analysis for the use of presently available lunar radio flux data in broad-beam antenna-system measurements*. 1984.
[32] S. F. Smerd. 'Radio-Frequency Radiation from the Quiet Sun'. In: *Australian Journal of Scientific Research (A)* 3 (Feb. 1950), p. 34.
[33] W. N. Christiansen and J. A. Warburton. 'The Distribution of Radio Brightness over the Solar Disk at a Wavelength of 21 Centimetres. III. The Quiet Sun-Two-Dimensional Observations'. In: *Australian Journal of Physics* 8 (Dec. 1955), p. 474. DOI: 10.1071/PH550474.
[34] D. J. Fixsen. 'The Temperature of the Cosmic Microwave Background'. In: *The Astrophysical Journal* 707.2 (Dec. 2009), pp. 916–920. DOI: 10.1088/0004-637X/707/2/916.
[35] G. E. P. Box and Mervin E. Muller. 'A Note on the Generation of Random Normal Deviates'. In: *The Annals of Mathematical Statistics* 29.2 (1958), pp. 610–611. DOI: 10.1214/aoms/1177706645.
[36] J. H. Piddington and H. C. Minnett. 'Radio-Frequency Radiation from the Constellation of Cygnus'. In: *Australian Journal of Scientific Research A Physical Sciences* 5 (Mar. 1952), p. 17. DOI: 10.1071/CH9520017.
[37] J. W. M. Baars et al. 'Reprint of 1977A&A....61...99B. The absolute spectrum of Cas A; an accurate flux density scale and a set of secondary calibrators.' In: *A&A* 500 (Oct. 1977), pp. 135–142.
[38] J. D. Kraus. *Radio Astronomy*. 2nd ed. New York: McGraw-Hill, 1986, pp. 8–44.
[39] J. D. Kraus. *Radio Astronomy*. 2nd ed. New York: McGraw-Hill, 1986, pp. 8–93.
[40] Jo Bovy. 'Galactic rotation in Gaia DR1'. In: *Monthly Notices of the Royal Astronomical Society: Letters* 468.1 (Feb. 2017), pp. L63–L67. DOI: 10.1093/mnrasl/slx027.
[41] J. P. Vallée. 'New Velocimetry and Revised Cartography of the Spiral Arms in the Milky WAY - A Consistent Symbiosis'. In: *The Astrophysical Journal* 135 (4 Apr. 2008), pp. 1301–1310.
[42] Peter V. Foukal. *Solar astrophysics*. 3., rev. ed.. Wiley-VCH, 2013, pp. 341–342.
[43] Markus J. Aschwanden. *Physics of the solar corona*. Springer, 2004, pp. 637–669.

Appendix

Acronyms

ADC	Analog to Digital Converter
API	Application Programming Interface
CPU	Central Processing Unit
DAC	Digital to Analog Converter
DSP	Digital Signal Processor
FIFO	First In — First Out
FWHM	Full Width at Half Maximum
GCC	GNU Compiler Collection
GUI	Graphical User Interface
HPBW	Half-Power Beam Width
LNA	Low-Noise Amplifier
MASER	Microwave Amplification by Stimulated Emission of Radiation
PLL	Phase-Locked Loop
POSIX	Portable Operating System Interface
PSF	Point Spread Function
RAM	Random-Access Memory
SNR	Signal-to-Noise Ratio
SoC	System On Chip
VLSR	Velocity of the Local Standard of Rest

Glossary

Analog to Digital Converter (ADC)

An Analog to Digital Converter is a system that converts an analog signal into a quantized digital signal. Its counterpart is the Digital to Analog Converter (DAC).

Application Programming Interface (API)

The Application Programming Interface defines how a developer can write a program that requests services from an operating system or application. APIs are implemented by function calls composed of verbs and nouns, i.e. a function to execute on an object.

Callback

Callback functions are a concept in programming, which allow executable code segments to pass the location of a function to a different code segments at runtime. That other segment can then use the callback to provide data to the original caller.

Central Processing Unit (CPU)

The Central Processing Unit is the electronic circuitry that interprets instructions of a computer program and performs control logic, arithmetic, and input/output operations

specified by the instructions. It maintains high-level control of peripheral components, such as memory and other devices.

Digital Signal Processor (DSP)

A Digital Signal Processor is a specialised processor with its architecture targeting the operational needs of digital signal processing.

Digital to Analog Converter (DAC)

A Digital to Analog Converter is a system that converts a quantized digital signal into an analog signal. Its counterpart is the ADC.

First In — First Out (FIFO)

In FIFO processing, the "head" element of a queue is processed first. Once complete, the element is removed and the next element in line becomes the new queue head.

Full Width at Half Maximum (FWHM)

The Full Width at Half Maximum is the width of a curve measured between two points on the y-axis which are at half the maximum amplitude. It is used as a measure of an instrument's resolution in the context of the (idealised) PSF.

GNU Compiler Collection (GCC)

The GNU Compiler Collection is a compiler system produced by the GNU project. It is part of the GNU toolchain collection of programming tools.

Graphical User Interface (GUI)

A Graphical User Interface is an interactive visual interface of computer software. It displays rendered objects that convey information to the user. These objects may serve purely as output, or they may represent actions which can be taken by the user, such as symbolic buttons or range dials. The user typically interacts with these by use of an electronic pointer device or a keyboard.

Half-Power Beam Width (HPBW)

The Half-Power Beam Width is the width of the main lobe of the beam where the power received from a point source is half of its peak value. It is usually identical or at least analogous to the FWHM.

Low-Noise Amplifier (LNA)

A Low-Noise Amplifier is an electronic device designed specifically for amplification of very low-power signals without significant degradation of their signal-to-noise ratio.

OpenMP

OpenMP is a multi-platform API which consists of compiler directives and libraries that allows programming of shared-memory, multi-threaded, multi-processing software by (mostly) automated parallelisation.

Phase-Locked Loop (PLL)

A phase-locked loop is a control system whose output signal phase is related to the phase of its input signal. It is a mechanism to stabilise and maintain the phase relations between the signals.

Point Spread Function (PSF)

A Point Spread Function describes the two-dimensional distribution of light in the focal plane of a telescope. The PSF for a perfect system is derived from Fraunhofer diffraction theory and is also called the *Airy Pattern*.

Portable Operating System Interface (POSIX)

The Portable Operating System Interface is a family of standards specified by the IEEE Computer Society for maintaining compatibility between operating systems.

Random-Access Memory (RAM)

Random-Access Memory is a type of memory where each memory cell may be accessed directly via their memory addresses.

Signal-to-Noise Ratio (SNR)

The signal-to-noise ratio is a comparison between the level of a desired signal to the level of background noise.

System On Chip (SoC)

A System On Chip is an integrated circuit that combines all components of a computer or other electronic system into a single chip.

Velocity of the Local Standard of Rest (VLSR)

The Local Standard of Rest (LSR) follows the mean motion in the neighborhood of the Sun within the Milky Way. If a velocity of an object is specified in this reference, it corresponds to the radial velocity relative to the movements of the Sun-Earth system.

Widget

In the context of a GUI, a widget is a self-contained functional (control) element used in the construction of an interface. Common examples are buttons, scroll bars or entry fields.