AUTOMATION OF SPECTROSCOPIC OBSERVATIONS ON THE DARK SKY OBSERVATORY 32-INCH TELESCOPE

A Thesis
by
DANIEL EDWIN ROSENBERG

Submitted to the Graduate School
Appalachian State University
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

August 2016
Department of Physics and Astronomy

AUTOMATION OF SPECTROSCOPIC OBSERVATIONS
ON THE DARK SKY OBSERVATORY 32-INCH TELESCOPE

A Thesis
by
DANIEL EDWIN ROSENBERG
August 2016

APPROVED BY:

Richard O. Gray, Ph.D.
Chairperson, Thesis Committee

Michael M. Briley, Ph.D.
Member, Thesis Committee

Jennifer L. Burris, Ph.D.
Member, Thesis Committee

Max C. Poole, Ph.D.
Dean, Cratis D. Williams School of Graduate Studies

Abstract

AUTOMATION OF SPECTROSCOPIC OBSERVATIONS
ON THE DARK SKY OBSERVATORY 32-INCH TELESCOPE
August 2016

Daniel Edwin Rosenberg
B.S., University of North Carolina at Chapel Hill
M.S., Appalachian State University
Chairperson: Richard O. Gray

The 32-inch telescope at Dark Sky Observatory and the GM Spectrograph attached to it have been remotely operable via internet access since December of 2011, allowing users to carry out spectroscopic observations from the comfort of their homes. However remote access still requires an observer's frequent attention throughout the night. In addition, many observers are university faculty and often receive telescope time on weeknights. To make observing more convenient for faculty, and to improve observing efficiency and consistency, we sought to automate the existing hardware and software systems necessary to conduct astronomical spectroscopy, while making only necessary additions. This thesis details our automation efforts, namely incorporating the intelligence and caution of a human, and the efficiency and consistency of a computer.

We developed a program called `RoboticSpectroscopist`, with a graphical user interface, to follow the same observing procedure a human observer would execute. It communicates with three observatory computers, as well as with

a weather-monitoring program, `RoboticWeatherman`. Our custom software is written using the scripting language AutoIt, and incorporates a few standalone C programs. It controls devices and software using TCP/IP, X-10, ActiveX, and SMTP communication protocols. It has been designed to protect observatory equipment and can notify a user via text message should a problem arise. `RoboticSpectroscopist` has been noticeably more efficient than human observers, regularly acquiring upwards of one hundred spectra on clear nights.

Acknowledgements

First and foremost, I would like to thank Dr. Gray for the enormous amount of support, encouragement, guidance, and enthusiasm he has shown for this project. In addition, his familiarity with the observatory, telescope, and equipment have made `RoboticSpectroscopist` much more 'intelligent' than it would have been otherwise.

I extend sincere thanks to Lee Hawkins, whose patience and incredible knowledge of the observatory, computers, and control systems were vital to the completion of this project.

Dr. Burris has also earned my gratitude, for thoroughly proofreading various drafts of this thesis, while also holding the position of graduate program director. Her non-astronomical perspective has been an invaluable asset.

Additional thanks is due to Dr. Briley, department chair and another astronomical spectroscopist. He cooperatively tested many versions of our software, identifying problems and recommending additions.

We gratefully acknowledge funding for this project from National Science Foundation grant AST-1109158, from the Appalachian State University Research Council, and from North Carolina Space Grant. I also acknowledge receipt of the James C. Greene Fellowship for general academic expenses.

Finally, endless thanks to my friends and family who have always supported me, even when they couldn't understand my scientific babble.

Table of Contents

Chapter 1

Introduction

Appalachian State University (ASU) owns, operates, and maintains the Dark Sky Observatory (DSO). It is a telescope site roughly 45 minutes by car from the university campus, and is home to five optical telescopes. Dr. Dan Caton is the observatory director. ASU faculty and students use DSO for research, education, and community outreach.

In December of 2011, astronomical spectroscopists at ASU gained the ability to perform their observations remotely from any computer with internet access, rather than from the control room at DSO. This was a significant change in normal observing procedure. Observers save 90 minutes of travel, and can conduct their research comfortably from home. Unfortunately, they often receive a night on the telescope before or after a regular weekday of academic duties.

Except in inclement weather, remote observing still affords little time for sleep. A groggy observer may occasionally forget to acquire a calibration image, or accidentally doze for several minutes while the telescope idles. Even an attentive observer

may pause to choose the next target to observe, record details in an observing log, or check the weather forecast.

With such conflicts and time-consuming tasks in mind, ASU's astronomical spectroscopists began investigating the feasibility of automating the observatory. Not only can an automated system perform tasks more quickly, it does not get distracted or fatigued. In January 2015 we began converting the 32-inch telescope and attached GM Spectrograph to operate robotically.

Automation became functional around January 2016, and ASU's astronomical spectroscopists began to use it extensively at that time. From January to March 2016, automated observing produced an average of 13% more spectra per night than a human observer during the same period in 2015. In addition, automated observing was performed while the assigned human observer slept. This thesis describes the process of adapting the observatory for robotic operation and developing the control software.

Chapter 2 provides a brief overview of automated spectroscopy, including the requirements, advantages, and challenges, since it was first suggested around 1986. Chapter 3 describes spectroscopy at DSO, including the manual observing procedure used until the observatory was automated. The program we developed to control the observatory and perform observations, `RoboticSpectroscopist`, is the subject of Chapter 4. To ensure the safety of the observatory and equipment, we delegated weather monitoring to a separate program, `RoboticWeatherman`, which Chapter 5 details. Chapter 6 concludes our experiences throughout this project.

Chapter 2

History of Automated Spectroscopy

2.1 A Logical Development

As early as 1986, astronomers recognized the value of automating spectroscopic telescopes. Modern day technologies, such as charge coupled device (CCD) detectors and fiber-fed spectrographs, were cutting-edge at the time. Powerful computer processors and large volumes of digital data storage were rare and expensive. However, Bopp (1986) comments that such issues were "complications, not problems." This sentiment was repeated more recently by Kozłowski *et al.* (2014), with the caveat that "the profit of automating the entire observing procedure with a complex and expensive instrument could be questionable," but "[i]n the case of smaller instruments...optimizing efficiency will bring much profit and scientific payback." In short, human oversight is much more important for larger telescopes, where

errors and malfunctions can be extremely costly. Conversely, smaller telescopes receive the benefit of substantial increases in efficiency with less associated risk.

Ramsey (1992) provides two requirements for automated spectroscopic observatories, based on robotics being adept at simple repetitive tasks. First, the automated system must conduct observing procedures without relying on frequent human input or attention. Second, the instruments and programs used in the automation must be well-defined or strictly-constrained to prevent behaviors the automated system is not designed to handle. These prerequisites apply to automated photometry as well, but become more important for spectroscopy due to the increased precision required to focus an object on a spectrograph slit or optical fiber.

2.2 Benefits of Automation

One of the earliest implementations of an automated astronomical observatory was carried out by Tennessee State University (TSU). Adhering to the requirements listed by Ramsey (1992), they discovered that automated observations using photometric telescopes produced significant improvements in observing efficiency and data quality when compared to observations performed by human observers (Eaton, 1995). That experience became the foundation and motivation for TSU to explore spectroscopic automation.

2.2.1 Improvements in Observing Efficiency

Eaton (1995) mentions several benefits of the efficiency of automated observing, one of which is the reduction of monetary cost. Due to the remote locations (e.g., desert or mountaintop) of many telescopes, on-site observing may require travel abroad. Automation can eliminate these costs, since the observer's presence at the telescope site may not be required at all. Should the observer have to be present, the visit may be made either shorter or more productive by automation, resulting in a lower cost per observation.

Human observers may necessarily introduce pauses between observing tasks, perhaps to reconsider which target to observe next, to check the weather forecast, etc. Computers generally incur significantly less *overhead*, or time not spent observing, by virtue of performing target selection, weather monitoring, and other essential tasks in a matter of seconds. Thus a well-designed automated telescope has the potential to acquire data much more efficiently than a human could, thereby producing more data per observing run (Eaton, 1995).

Many astronomers are university faculty, and have academic duties in addition to research. By automating the repetitive and time-consuming observing process, they may devote more time and attention to planning observations, analyzing data, handling teaching responsibilities, etc. Eaton (1995) suggests $50 per hour as the value of an astronomer's time, which is "too valuable...to waste on such repetitive tasks as manual observing."

On their own, these improvements in observing efficiency already justify automation. The ability to obtain more data more quickly at less cost, without the constant attention of a human observer, could greatly accelerate astronomical study. But automation offers still more benefits over human observers, namely in the realm of data quality.

2.2.2 Improvements in Data Quality

Automation requires a well-specified observing procedure. Each separate task and any applicable constraints must be defined quantitatively. For example, the threshold between high- and low-quality observations might be defined as a specific signal-to-noise ratio over a given wavelength range in the spectrum. Such criteria can be calculated quickly by a computer, even in the midst of acquiring data, and can be used immediately to adjust the number of exposures required to obtain high-quality data for a given target (Eaton, 1995).

As mentioned in the previous section, an extensively-automated observatory may not require any human presence during observing. Eaton (1995) emphasizes the need for frequent instrument calibration and detailed quality-control reporting, especially without anyone on-site to diagnose and rectify problems as they occur. In addition, thorough information about observatory conditions throughout the night provides observers with context for collected data. Should an observer wish to omit an unusual observation, details provided in such a report can provide the necessary basis for doing so.

Beyond a single night of observing, automation allows for longer and more varied studies. Especially with periodic targets such as variable stars, pulsars, and quasars, observing the object nightly for a year might be scientifically beneficial, but logistically impossible for human observers. Observing time on major telescopes is awarded by a Time Allocation Committee (TAC), and observing runs may vary in duration from single nights to several weeks. Automation allows for a new possibility: queue-based observing (Eaton, 1995). Rather than award a full night to any given project, the TAC places all approved projects into a queue, giving each project or individual target a certain priority level. The observing control software selects targets based on priority, which may depend on target location, season, time of night, lunar phase, instrument availability, target periodicity, or other TAC-assigned factors.

Queue-based observing improves the chance for each target to be observed closer to its 'ideal' times, however that happens to be defined by the particular research project. It additionally ensures that telescope time is used as efficiently as possible by selecting the 'best' target from a larger pool.

These benefits of automation enumerated by Eaton (1995) respect the prerequisites from Ramsey (1992) and develop the advancements in automated spectroscopy predicted by Bopp (1986) into concrete guidelines, albeit without actually implementing them.

2.3 Successful Automated Spectroscopy

2.3.1 The Automatic Spectroscopic Telescope

In June 2003, Tennessee State University (TSU) achieved first light on their two-meter Automatic Spectroscopic Telescope (AST), housed at Fairborn Observatory in Arizona. The telescope and its systems, as well as target selection and data reduction software, are controlled by four Linux computers located at the observatory. TSU wrote their own target selection software based on the Automatic Telescope Instruction Set (ATIS); one version is described in Boyd *et al.* (1993). However TSU's software improves upon ATIS by accepting more user input regarding previously-observed targets. Due to the telescope's remote location from the University, the observing routine includes automatic transmission of data and log files via internet (Eaton & Williamson, 2004).

The AST was designed and constructed from the ground up to be an automatic spectroscopic telescope. This was a significant advantage; Eaton (1995) mentions that designing an entire observatory with complete automation in mind influences hardware and software choices. For example, a manual telescope requires well-designed user interfaces to make observation as efficient as possible, but such interfaces are rarely used on a robotic telescope. Rather, precedence is given to the ease of communication between the myriad programs required to control the telescope and its instruments. Thus TSU bypassed the issues associated with automating an existing manual observatory, such as Appalachian State University's Dark Sky Observatory (DSO), or the Swiss Euler telescope in La Silla, Chile.

2.3.2 Conversion of the 1.2-Meter Euler Telescope

While construction on the AST was ongoing in 2001, the team at the Geneva Observatory was celebrating three years of automated spectroscopy. They had adapted the manual 1.2-meter Euler photometric telescope to perform robotic observing in 1990, and installed the CORALIE spectrograph in 1998[1]. One disadvantage, Blecha *et al.* (2001) note, was that the telescope and off-the-shelf CCD camera were not designed for use in an automated system and had to be adapted to that purpose. TSU did not encounter the same challenge, thanks to their from-scratch design for the AST. Nevertheless, Euler was successfully converted to perform automated spectroscopy, and even included a few features not present on the AST (Blecha *et al.*, 2001).

Euler's novel features included custom graphical user interfaces (GUIs) for all services comprising the observing procedure. While the converted telescope was primarily intended to operate autonomously, two additional modes of operation were provided. One is 'automatic-attended,' where the user receives manual control after an error; the other is fully manual. Blecha *et al.* (2001) explain that the inclusion of these modes was based on the principal that "[t]he whole facility should be designed in a way to be operated by a single person (the observer) who is not necessarily an astronomer."

Euler's top-level control GUI provides sufficient interaction to allow a trained user to operate the telescope through the entire observing procedure in either

[1]European Southern Observatory 2016, 'Swiss 1.2-metre Leonhard Euler Telescope,' `https://www.eso.org/public/teles-instr/lasilla/swiss/`

automatic-attended or fully manual mode. Most other services, which range from cosmetic controls to modifications of telescope guiding parameters, were given their own GUIs. Observations can be conducted without adjusting any of these more advanced controls, but they allow easy on-the-fly modification of observing parameters for experienced astronomers (Blecha *et al.*, 2001).

Complete automation of the Euler telescope was not quite achieved, Blecha *et al.* (2001) admit. A few elements of the observatory system, namely data reduction and maintenance of cryogenic cooling systems, proved prohibitively difficult to automate. Developing algorithms to consistently identify targets and position them properly for observation was another challenge: target acquisition was estimated to be 95% automated, suggesting that it fails to acquire one out of every 20 targets. Nevertheless, Euler is an example of the well-executed automation of an existing manual observatory. Considering the cost of designing and constructing a new observatory, conversion is an attractive option.

2.3.3 Subsequent Automated Spectroscopic Telescopes

While the Euler telescope and Automatic Spectroscopic Telescope (AST) were the earliest spectroscopic observatories to operate robotically, a number of others have been developed since. They range from individual observatories such as STELLA, Solaris-4, TIGRE, and the Liverpool Telescope 2, to global networks of automated telescopes including Las Cumbres and Skynet. A discussion of these facilities and their contributions to spectroscopic automation follows.

The STELLA Robotic Observatory The Izaña (formerly Teide) Observatory in Tenerife, Spain is home to the STELLA robotic observatory, which houses two completely automatic telescopes, appropriately named STELLA-I and STELLA-II. The latter "is a highly specialized telescope with the single purpose to feed as much light as possible into an on-axis fiber," and is used for photometric observations (Granzer *et al.*, 2010). When it was installed, STELLA-I fed light into the STELLA Echelle Spectrograph (SES). In 2010, the wide field imaging instrument (WiFSIP) was installed on STELLA-I, and the SES was moved to STELLA-II (Weber *et al.*, 2012).

A remarkable feature of the STELLA observatory is that it can operate completely autonomously for days on end. An internet connection is required only to retrieve data, upload new target lists, and infrequently perform remote-access observing. This arrangement supports the extended-duration studies mentioned by Eaton (1995), especially useful for targets with properties that vary over a period of several days. The value of such a degree of autonomy is further augmented by STELLA's queue-based method of target selection (Granzer *et al.*, 2010).

STELLA's target selection algorithm is a type of *dispatch scheduling*, whereby approved observers submit targets to a master list, and every object on that list receives a merit value. An object's merit is based on factors such as desired observational frequency, permitted sky position, etc., and is updated in real-time. The observatory control system selects the object with the highest merit as its next target (Granzer *et al.*, 2010).

While target selection is comparatively easy, target acquisition is one of the most difficult processes to automate, Blecha *et al.* (2001) note. STELLA employs a method using plate solving, whereby the telescope slews to an object, acquires an image, and compares it to a catalog of astronomical objects. When the plate-solving algorithm matches the telescope image to known object coordinates, the telescope's pointing can be precisely adjusted to center the target star on the spectrograph aperture. Once a target has been acquired, the telescope begins guiding on the target and collecting data (Granzer *et al.*, 2010).

One of STELLA's features that enables long unattended observing programs is a sophisticated weather monitoring and prediction system. An independent watchdog system monitors computer activity, and closes the observatory roof in the event of a computer crash. Two separate weather stations at the observatory provide a redundant level of protection for obtaining vital weather information. The decision to close the roof is made when a vital quantity reaches its assigned threshold, such as wind speed exceeding 20 meters per second (Granzer *et al.*, 2010).

Because of STELLA's location in Tenerife, humidity proved to be a more problematic quantity, and required the design of custom extrapolation software. The 'extrapolators' impose an additional criterion for roof closure: if measured humidity exceeds 70%, or if six of the seven extrapolators predict sustained humidity above 80%, the roof closes (Granzer *et al.*, 2010).

STELLA's humidity prediction algorithm illustrates one of the inherent challenges of astronomical automation; no single set of programs can be successful at every observatory. With existing telescopes using different and sometimes unique

hardware and software, most automation efforts will require some degree of customization. Building a new telescope for automated observing, however, allows its design to adhere to an existing control scheme, should it fit the science goals of the intended project.

Las Cumbres Global Network The Las Cumbres Observatory Global Telescope (LCOGT) Network takes advantage of designing hardware to to meet software specifications. It comprises photometric and spectroscopic telescopes around the world, many of which are designed to be identical. The network is not complete, but plans include two identical 2-meter telescopes, seventeen identical 1-meter telescopes, and twenty-three identical 40-centimeter telescopes (Brown *et al.*, 2013).

Every instrument will require its own calibrations, but the same control systems and data reduction pipeline will apply to each group of identical telescopes. Thus control software will have to be designed for the first 40-centimeter telescope, but should also work for the other 22 identical telescopes. Unfortunately, minor modifications, such as STELLA's humidity extrapolators, may still have to be made to account for physical differences in the individual observatories.

Skynet A similar telescope network is Skynet, organized by the University of North Carolina at Chapel Hill. Skynet currently incorporates as many as 21 optical photometric telescopes, located at various sites around the world. However most of these telescopes were not constructed from the same designs, nor for the same science goals, nor even by the same engineers. Consequently, unique control and data reduction software must be written for each type of telescope.

All twenty-three of LCOGT's 40cm telescopes can use the same control systems, calibration methods, and data reduction algorithms. This similarity could feasibly reduce the time required to program them by an order of magnitude, an advantage from which Skynet does not benefit. Nevertheless, it is a completely robotic network, and an excellent resource for teaching, simultaneous observations, and time-series photometry. It also benefits from inclusion of telescopes built by many different groups, thus bypassing the need to accrue support, funding, materials, and labor for their construction. In fact, two of the Dark Sky Observatory telescopes, the 14-inch and 18-inch, are connected to Skynet[2].

Solaris-4 & BACHES Échelle Spectrograph Kozłowski *et al.* (2014) reports satisfactory results of evaluation of a prototype BACHES échelle spectrograph on the Solaris-4 observatory's 0.5-meter telescope: "BACHES is a very compact and capable spectrograph well suited for remote and autonomous operation." Of particular note is the spectrograph's compact form; rather than many large spectrographs, which are bench-mounted away from the telescope and fed light via optical fibers, BACHES is mounted directly on the telescope. It receives light from a guiding and acquisition module (GAM), which contains a movable mirror to direct light through one of two output ports (Kozłowski *et al.*, 2014).

Unfortunately, mounting the spectrograph on the telescope introduces mechanical flexure, a problem absent from bench-mounted instruments. As the telescope assumes different positions throughout the night, it and attached instruments flex due to their own weight. To account for flexure, calibrations must be obtained

[2]Skynet 2016, Reichart, D., 'Telescopes,' `https://skynet.unc.edu/telescopes`

much more frequently throughout the night, usually before and after slewing the telescope.

In spite of flexure, mounting BACHES on the telescope allows Kozłowski *et al.* (2014) to achieve the goal of conducting both photometry and spectroscopy without interchanging instruments or physically modifying the imaging train. BACHES's success supports the idea that small spectroscopic telescopes can feasibly provide automated contributions to large-scale spectroscopic surveys (Kozłowski *et al.*, 2014).

TIGRE Robotic Spectroscopic Telescope The Telescopio Internacional de Guanajuato Robótico Espectroscópico (TIGRE) is a 1.2 meter telescope housed at the La Luz Observatory, and is operated and maintained by the Department of Astronomy at the University of Guanajuato (UG). Unlike BACHES, the échelle spectrograph on TIGRE is a fiber-fed bench-mounted instrument. One of UG's primary science goals for TIGRE is monitoring, or time-series astronomy. Monitoring involves regularly-repeated measurements of a particular target, with the goal of studying its behavior over time. Schmitt *et al.* (2014) cites the sunspot cycle as an area lacking sufficient time-domain data, particularly due to non-uniform sampling and varied quality of data.

This is one of the data quality problems that Eaton (1995) suggests could be mitigated by automation. Especially since astronomers can hardly study the universe except through observation, the time evolution of astronomical targets can provide information not available from single images (Schmitt *et al.*, 2014). Insufficiently frequent observation of variable objects causes aliasing, whereby their true vari-

able behavior cannot be determined. To accurately understand the sunspot cycle, for example, the sun's properties must be measured much more quickly than they vary. TIGRE and other monitoring or queue-based telescopes, such as STELLA, are well-equipped for such studies.

The TIGRE telescope system has separate control software for the telescope, the building, weather monitors, and the spectrograph. Software is written mostly using Java in a Linux environment, except for image analysis software for the CCD cameras, which is written using the C language. Data collected by TIGRE are processed automatically by a custom data reduction pipeline written in the Interactive Data Language (IDL). Schmitt *et al.* (2014) note that such automatic reduction becomes necessary to handle the substantial volume of data, which is available thanks to the efficiency of robotic observing.

Liverpool Telescope 2 Aimed at the realm of time-domain astronomy, the Liverpool Telescope 2 (LT2) is being designed as a four-meter-class rapid-response telescope, able to acquire and observe transient astrophysical phenomena within minutes of their detection. In particular, Copperwheat *et al.* (2015) list neutrino and gravitational wave emissions, which can precede photons from an event and be detected by other telescopes, as possible 'triggers' for LT2. In the time it would take a human observer to notice an event and telephone colleagues to begin emergency observations, a completely robotic global network could already be collecting data. Especially since phenomena may last only minutes, rapid acquisition of these targets-of-opportunity is vital (Copperwheat *et al.*, 2015).

LT2 will be capable of photometric and spectroscopic observing, although visible and infrared spectroscopy will be its main goals. Whether the spectrograph aperture will be a long slit or fiber bundle has not yet been decided[3]. Though LT2 will perform primarily spectroscopy, Copperwheat *et al.* (2015) point out that even images acquired for the purpose of aligning targets on the spectrograph aperture will themselves provide valid photometric data.

Automation of an observatory to respond to an event within 30 seconds of its detection adds strict precision constraints to various aspects of the robotic system. For example, a target may be detected on the opposite side of LT2's observable field, requiring a 180-degree slew. And since LT2 will be a spectroscopic telescope, placing the target on the spectrograph aperture means locating it to within one arcsecond, or 1/3600th of one degree. Maintaining sub-arcsecond pointing precision after such a large blind slew is infeasible. So LT2 has been given the goal of placing a target-of-opportunity within the field of view for photometric observation within 30 seconds of detection. Aligning the target on the spectrograph aperture will take longer (Copperwheat *et al.*, 2015).

An additional concern is acquisition of a guide star. Depending on a target's location, no stars in the visible field may be suitable for telescope guiding. This condition, which Copperwheat *et al.* (2015) call open-loop tracking, cannot necessarily be avoided. A transient phenomenon may evolve and disappear in the time it would take to acquire a guide star required for closed-loop tracking. For LT2, the "design requirement is that the tracking must allow for a monochromatic

[3]Liverpool John Moores University 2013, Liverpool Telescope 2, 'Telescope and Instrumentation,' `http://telescope.livjm.ac.uk/lt2/telescope.html`

exposure of at least ten minutes with an image elongation of no greater that 0.2 arcseconds" (Copperwheat *et al.*, 2015). Once a suitable guide star is acquired, the associated closed-loop guiding constraint is a one-hour monochromatic exposure with the same image elongation.

LT2 highlights the benefit of an individual robotic telescope being connected to the internet. Though not an official member of a global telescope network such as LCOGT, it can receive near-instantaneous notification of a target-of-opportunity, with enough time to acquire the target and begin observing before transient phenomena disappear. A human observer would be unlikely to replicate this feat.

2.4 The Future of Automated Spectroscopy

The purpose of automating astronomical spectroscopic observation is to make it somehow better than manual observation. Ramsey (1992) provided guidelines and Eaton (1995) enumerated motivations. In the two decades following, astronomers developed robotic telescopes to be more efficient, consistent, and tireless than human observers. It is still the job of humans to request observations, interpret data, and publish conclusions, but those tasks can now be accomplished while the telescopes operate themselves.

In spite of the known and unknown challenges of converting an existing observatory to near-complete autonomy, we began in January 2015 to adapt the 32-inch telescope at DSO for automated spectroscopy. We encountered many of the same difficulties and solutions described above, and even accepted or rejected some of the same software. The history of spectroscopy at DSO, the current manual observ-

ing process, and the software we have developed to carry out robotic spectroscopy are described in the following chapters.

Chapter 3

Spectroscopy on the 32-Inch Telescope and Gray-Miller Spectrograph

3.1 Overview and History

The Dark Sky Observatory (DSO) is a telescope site owned, operated, and maintained by Appalachian State University. It was established in 1981, and its telescopes are used by university faculty and students for observational research, as well as for public outreach endeavors. The current observatory director is Dr. Dan Caton, a faculty member in ASU's Department of Physics and Astronomy. Mr. Lee Hawkins is the observatory engineer, and performs much of the maintenance necessary to keep DSO operating safely and efficiently.

DSO houses five telescopes, in separate enclosures around the observatory site. The 14-inch telescope was assembled by former graduate student Adam Smith for his Masters thesis, and is connected to the Skynet global telescope network mentioned in §2.3.3. The 17-inch telescope was donated by Dean Glace. It is controlled remotely, either by ASU observers or by Dean Glace in South Carolina. The 6-inch robotic telescope was built to monitor stars, as part of the Young Solar Analogs project. The 6-inch, 14-inch, and 17-inch telescopes are used for photometric observing. An 18-inch telescope is fitted with a charge-coupled device (CCD) camera and filter wheel, and is used for photometry of asteroids and eclipsing binary stars. It is also connected to Skynet. The largest telescope is the 32-inch. Its main instruments are a CCD camera with a filter wheel, and the Gray-Miller (GM) Spectrograph. The CCD and filter wheel are used for photometry, while the spectrograph is used by several faculty members for medium-resolution spectroscopy of stars[4].

The GM Spectrograph was constructed in 1994. It is a Czerny-Turner spectrograph and uses one of two reflection gratings; one with 1200 grooves per millimeter, the other with 600 grooves per millimeter. It was designed and constructed by Dr. Richard Gray and Robert Miller, a former machinist who is now retired. The GM Spectrograph was installed and used on the 18-inch telescope until 1995, when the 32-inch telescope achieved first light. The spectrograph was then moved to the 32-inch telescope, where early observing was entirely manual. For the first few years of spectroscopy on the 32-inch telescope, the observer had to acquire targets, as

[4]Caton, D.B. 2016, 'Facilities,' `http://dso.appstate.edu/facilities`

well as keep the telescope pointed at the target, by looking through an eyepiece and pressing directional buttons on a hand paddle.

A TV system was later installed, which focused a video camera on the reflective spectrograph slit. This arrangement allowed the observer to guide the telescope from a comfortable desk in a warm room, rather than using the eyepiece. However telescope motion was still controlled with the hand paddle. This system was in place until 2007, when the observatory acquired a CCD guide camera and telescope guiding software. In general, guiding software analyzes an image from a CCD camera and determines the necessary telescope adjustment to keep the target in its optimal position. At DSO, it relayed those positional adjustments to the Telescope Control System (TCS), which in turn performed the telescope movement.

In December of 2011, DSO astronomers started performing spectroscopic observations remotely. That was the norm until this automation project began. Once an observer has installed the instruments required for his or her observations and rebalanced the telescope accordingly, observing can then be conducted on any computer with access to the internet and virtual network computing (VNC) software. Remote observing can even obviate the need for an observer to travel to DSO at all, if no physical changes to the telescope setup are needed.

Despite the conveniences of remote observing, an observer's constant attention is still necessary to set up the instruments, slew to targets, turn on and off lights, control the Guide and Acquisition Module (GAM) and CCD cameras, open and close the telescope dome, monitor the weather, etc. A description of that manual procedure follows.

3.2 Nightly Procedure for Spectroscopy

3.2.1 Instrument Installation and Evening

Installation The 32-inch telescope at DSO is used for different observing projects, each of which has its own required telescope and instrument configuration. Consecutive nights on the telescope are not necessarily assigned to observers with similar equipment arrangements. Most commonly, after the telescope is used for imaging or photometry, the next assigned spectroscopist must remove the photometry CCD and replace it with the spectrograph, rebalance the telescope, recalibrate instruments, and make necessary adjustments in software. For example, when installing the spectrograph, the observer must properly center the spectrograph slit on the CCD camera that handles telescope guiding.

Once the proper instruments are installed, observing can be performed entirely remotely via internet. Most remote interaction with the telescope occurs through a VNC connection. A few actions, such as providing power to the telescope dome motors and turning on and off various lights, are controlled by an X-10 controller addressable from the Activehome Command software. Others, such as powering the calibration arc lamp, are controlled by Digital Loggers web power switches accessible from a website.

Software Controls Necessary software adjustments can also be performed via internet. The telescope's tracking rate can be adjusted, for example to track a comet or asteroid, and must be properly set to sky-tracking rate for spectroscopy. Pixel binning for the CCD can be specified as well; 2x2 binning combines four pixels

into one, providing faster data transfer but at reduced resolution. The telescope must also be adjusted to place the focus of its optical path, the path light takes through the telescope, on the spectrograph slit.

Calibration - Darks Observing usually begins at evening nautical twilight, when the sun is 12 degrees below the horizon[5]. Before that time, the observer will have started the CCD cooler. Cooling the CCD limits thermal noise being detected as background noise. This 'dark current,' cannot practically be eliminated entirely. To account for it, an observer acquires calibration images called *darks*, with the camera shutter closed and no light sources illuminating the CCD. The resulting image contains the dark current noise generated by the camera electronics, allowing it to be subtracted from data images and comparison lamp images during data reduction. The observer usually acquires darks while the telescope is parked at *zenith*, meaning pointed directly overhead, and does so before evening twilight and after morning twilight.

Calibration - Biases Another source of noise appears when the CCD pixels are read out and converted to digital values. This readout noise appears nearly identically on all exposures, independent of exposure time. Thus a zero-second exposure allows no time for CCD pixels to collect real photons, while producing an image of the readout noise[6]. In addition, CCDs are designed to have a non-zero signal offset; that value is also contained in these *bias* calibration images. Biases are

[5]United States Naval Observatory 2011, 'Rise, Set and Twilight Definitions,' `http://aa.usno.navy.mil/faq/docs/RST_defs.php`

[6]Deep Sky Stacker 2016, 'Frequently Asked Questions,' `http://deepskystacker.free.fr/english/faq.htm`

subtracted from all data and calibration images, and are acquired at the beginning and end of the night.

Calibration - Flats Another important calibration is a *flat field* image, which describes each pixel's sensitivity, or response to identical illumination. The 32-inch telescope uses a flat lamp to achieve roughly uniform illumination across the spectrograph slit. Another option would be dome flats, created by shining a light at a diffuse flat surface inside the dome. Data images are divided by the dark-subtracted flat field response during data reduction. The observer acquires flats and a special set of *dark flats* at the end of the night.

3.2.2 Normal Observing

Target Acquisition When calibrations are complete, the sky is sufficiently dark, and weather is clear, the observer opens the dome and enables dome and telescope tracking. 'Sufficiently dark' is defined by individual observers, according to the amount of ambient light allowable in their data images. Observing at DSO often begins at nautical twilight, when the sun is 12 degrees below the horizon. Tracking allows the dome and telescope to move to compensate for the rotation of the earth.

To begin observing, the observer locates a target in an astronomical object catalog, contained in a program called TheSky. TheSky sends the target's coordinates to the Telescope Control System (TCS), which slews the telescope to that location. The telescope uses a guide CCD camera, separate from the main CCD, to keep the target centered on the spectrograph slit during observations. As long as the slew places the target within the guide CCD's roughly 40-arcsecond field-of-view, the

observer can use a hand paddle to adjust the telescope *pointing*, or the direction the telescope points, until the target is centered on the spectrograph slit.

Calibration - Arc Lamp Despite being rigid, the telescope and installed instruments flex in unpredictable ways due to their own weight. This flexure depends on the angle of the telescope, temperature, etc., and can vary on timescales as short as 30 minutes. Flexure is particularly troublesome for spectroscopy. Some properties of astronomical objects, such as rotation and relative motion, can be inferred from the width and location of their spectral lines [7]. But if the expected location of stationary lines is unknown, no conclusion can be drawn about the target's motion based solely on the positions of its spectral lines.

To account for flexure and to enable proper wavelength calibrations of stellar spectra, a hollow-cathode iron argon arc lamp is used to obtain a wavelength calibration spectrum. During data reduction, a *wavelength solution* is developed from the calibration image, by plotting the wavelength of known lines against their pixel position and performing a regression analysis. The resulting model is then used to estimate the wavelengths of a target's spectral lines, based on their pixel position.

Before acquiring spectra of a target, the observer acquires an arc lamp spectrum. The lamp itself is mounted next to the spectrograph and points into a small hole in the telescope's guiding and acquisition module (GAM). The GAM contains a mirror on a translation stage to direct light through one of its four ports. Attached

[7] CSIRO 2016, Australia Telescope National Faciltiy, 'Information from Astronomical Spectra,' `http://www.atnf.csiro.au/outreach/education/senior/astrophysics/spectra_info.html`

to the translation stage is a second angled mirror which receives light from a fixed mirror in front of the arc lamp. The observer instructs TCS to move the GAM mirror to position 2, which orients the translation stage to direct light from the arc lamp to the spectrograph. The observer then turns on the arc lamp via web switch, acquires an arc lamp spectrum, returns the GAM to position 4, and turns off the arc lamp.

Spectroscopy Before acquiring spectra, the target may have to be centered on the spectrograph slit again using the hand paddle. The observer then enables telescope guiding. While guiding, the MaxIm DL camera control software actuates relays that cause minute adjustments to the telescope's pointing to keep the target centered on the spectrograph slit. MaxIm DL is described in detail in §4.3. The observer enters the desired number and duration of exposures into MaxIm DL and initiates observations. Targets normally receive between three and eight exposures of 300 seconds each. During that time, the observer can select the next target or perform other observing-related duties. MaxIm DL handles camera control, saving images, and guiding.

Clean-Up After MaxIm DL has successfully acquired the desired number of exposures on a target, the observer repeats the arc lamp calibration. The sky coordinates of the telescope's pointing may optionally be synched in TheSky, to improve future pointing accuracy. The observer also appends that night's log file with information about the observations and any problems. This completes the normal observing procedure for a single target and is simply repeated until dawn, weather permitting.

Contingency - Target Not Found When the telescope slews to a target, especially one far from its previous pointing, the new target may not fall within the guide CCD's field-of-view. The observer's first response is to attempt to locate the target manually by nudging the telescope using the hand paddle. If this fails, the observer may point the telescope at a nearby bright star, synch the telescope's new coordinates in MaxIm DL, and slew to the target again. While not guaranteed to work, this method has been successful in practice. Were it to fail, the observer could either synch the telescope on another bright star, or simply move on to a new target.

Contingency - Telescope Control System Reboot Occasionally, TCS may crash and have to be restarted. When TCS starts, it assumes the telescope is pointed at zenith, which may not be the case after a crash. To solve the problem, a five-degree field-of-view camera is installed on the telescope. It can acquire an image containing a number of stars, which the observer can then submit to plate-solving software such as Astrometry.net. Plate solving measures the positions and intensities of stars in an image, then compares them to a database of astronomical objects. A successful plate solve determines the sky coordinates of the center of the image, allowing the observer to synch the telescope coordinates in TheSky and resume normal observing. Astrometry.net has been consistently successful when given a five-degree image. However, should successive plate solves fail, the observer will have to drive to DSO and reorient the telescope in person.

Contingency - Inclement Weather Protecting the observatory equipment is a high priority and takes precedence over finishing observations on a target.

The weather display on the observatory computers provides information about temperature, humidity, wind, precipitation, cloud cover, etc. Its default behavior is to instruct TCS automatically to close the observatory dome, should conditions warrant it. This control may be disabled if the observer suspects that the sensor readings are spurious. For example, the cloud monitor occasionally detects brief spikes from clear to very cloudy weather. Each spike lasts only a few seconds, but the unusual behavior may repeat for a minute or more. Closing and opening the dome each take roughly 30 seconds, but TCS will attempt to close the dome for every spike. Thus quite a bit of time can be lost to a malfunctioning cloud sensor.

If weather legitimately dictates that the telescope close during the night, the observer may use the opportunity to obtain more dark and bias calibration images. However little else can be accomplished until the weather is sufficiently clear to allow observations to resume. In addition to high light levels, i.e., daytime, specific conditions requiring the dome to close include: temperature below 16°F, when the dome motors stop functioning properly; very cloudy sky; precipitation of any kind; sustained wind or gusts above their specified limits; relative humidity above its specified limit.

Observing Logs Periodically throughout the night, the observer makes notes regarding hardware or software problems, unusual observations, changes in weather conditions, etc. The nightly logs mainly help individual observers organize and interpret their large quantities of data. There are no strict format requirements for these logs. They may be consequently sparse in detail, since observers must write their comments between performing other vital tasks.

In addition to nightly individual logs, there is a long-term observing log in which each observer makes a brief entry after shutting down the telescope in the morning. This log describes any significant problems encountered during the night. Thus all observatory users can be informed of any restrictions on their normal observing routines, without having to discover the same issues themselves.

3.2.3 Dawn

At morning nautical twilight, the observer ceases regular observations to perform end-of-night procedures. The observer closes the dome, acquires darks and lamp flats, and parks the telescope at zenith. Rapid warming can damage the CCD, so the CCD cooler control has 'warm-up' functionality. This allows the CCD to warm slowly to ambient temperature, without incurring damage. Finally, the observer 'zips' and compresses the night's data using a software archiving program, then initiates a file transfer to a computer on campus, and writes a short entry in the long-term observatory log. This concludes the normal spectroscopic observing procedure.

Chapter 4

RoboticSpectroscopist

4.1 Motivation and Basic Functionality

The main goal of automating the Dark Sky Observatory's 32-inch telescope was to make nighttime observing schedules as compatible as possible with daytime academic schedules, but with corollary improvements in observing efficiency and data quality, as mentioned in §2.2. The telescope's primary users are university faculty, who cannot adjust their academic schedules every week to accommodate assigned telescope time. In addition, a robotic systems makes queue-based observing programs possible, and ensures that variable objects are observed sufficiently often.

To accomplish these goals, we designed and wrote a computer program, now called `RoboticSpectroscopist` (`RoboSpec`), to emulate a human observer. Fortunately, the normal observing procedure described in Chapter 3 has a well-defined, repetitive structure. In addition, since the entire observing process can be con-

trolled remotely, no elements of that process require manual actuation. Thus the hardware and software systems installed at DSO adhered to the criteria proposed by Ramsey (1992), suggesting they were suitable for robotic operation.

Unobtrusive Design Principal An important tenet in `RoboSpec`'s operation was that it be as unobtrusive as possible to existing observatory systems. Though the Windows 7 operating system installed at DSO embraces 'parallel' processing, and 'simultaneous' operation of numerous programs, they are not exactly parallel or simultaneous. Rather, the operating system switches rapidly between programs, allowing each to perform a number of tasks before allocating resources to the next queued program. Each additional piece of software requires time in the queue, effectively slowing the operation of the entire system. This description does not portray the complexity of task scheduling that the operating system performs, but aims to explain our attention to resource usage.

Also of concern were file access conflicts. Windows often denies access to a file that is already open in another program. This restriction prevents simultaneous changes to a file being saved, thereby rendering it unintelligible and useless. A single computer will rarely encounter this problem, but multiple computers may when sharing data over a network, such as those at DSO[8]. In particular, our weather-monitoring software reads a text file updated roughly every two seconds. We were concerned about crashing the weather hardware by reading and denying access to the text file, while the sensor simultaneously tries to write to it.

[8]Pyle, N. 2009, Microsoft Corporation, 'Understanding (the Lack of) Distributed File Locking in DFSR,' `https://blogs.technet.microsoft.com/askds/2009/02/20/understanding-the-lack-of-distributed-file-locking-in-dfsr/`

Fortunately, Windows 7 favors the New Technology File System (NTFS) developed by Microsoft[9]. NTFS allows multiple programs to access the same file in read-only mode, but allows only a single write-permission access[10]. No part of this automation project requires more than a single write-permission access to a file, so no conflicts have arisen.

Target Acquisition The first significant challenge while designing `RoboSpec` was target acquisition, namely locating and centering a target star on the spectrograph slit. A human may interpret a telescope image as a dark field with a several stars, each star's brightness decreasing smoothly and radially. But a computer interprets the same image only as a two-dimensional array of discrete numbers. From such an image, `RoboSpec` must be able to identify the location of a star's center, while ignoring bright cosmic rays and other noise.

Even during manual observing, the narrow field-of-view of the guide CCD camera occasionally made target acquisition difficult. To prevent `RoboSpec` from searching indefinitely for a target it failed to locate on its first try, the East Port CCD camera was installed. It has a large enough field-of-view to contain the target after even a large telescope slew. A program called `TCP_Camera3` processes the East Port image, and returns the pixel coordinates of the brightest object to `RoboSpec`. Following a short jog to bring the telescope closer to the desired coordinates, `RoboSpec` returns to the guide camera to more precisely center the target on the

[9]Microsoft 2016, 'Comparing NTFS and FAT32 File Systems,' `http://windows.microsoft.com/en-us/windows7/comparing-ntfs-and-fat32-file-systems`

[10]HairyFool 2013, Microsoft Community, 'Does Windows 7 file sharing allow simultaneous multiple user record updates,' `http://answers.microsoft.com/en-us/windows/forum/windows_7-files/does-windows-7-file-sharing-allow-simultaneous/e6e3b954-96b7-4612-b8bc-1d31d8d14296?auth=1`

spectrograph slit. The East Port camera is described in §4.2, and `TCP_Camera3` and the image processing procedure are described in §4.3 below. The final centering of a star on the spectrograph slit relies on a C program called `starfind3`, also described in §4.3.

This has proven to be a robust method of automatic target acquisition. However, it is limited to locating only the brightest star in the nearby field. Observers may have targets that are fainter than their neighbors. To observe such objects, `RoboSpec` will have to perform a plate solve on the East Port image. This functionality has not yet been incorporated into `RoboSpec`'s operation, but is coming in the near future.

Observatory Safety The second significant challenge was to program 'intelligence' into `RoboSpec`. The telescope and attached instruments are expensive, fragile, and difficult to replace. Any use of the observatory, whether by a human or robotic observer, must respect physical limits of hardware, handle errors generated by software, and monitor weather conditions. The `RoboSpec` script itself does not interact with any weather sensors. The decision to continue or cease observations due to weather is handled entirely by `RoboticWeatherman`, a separate script that is the subject of Chapter 5.

Respect for the physical limits of the observatory is handled by other scripts as well. An example is the positional limit for the telescope while tracking is enabled. The physical tracking limits fall around hour angle (HA) of positive and negative five. The local meridian, the line from the north to the south celestial pole and passing directly overhead, is defined as HA zero. An object with HA negative one

hour will pass overhead in one hour; an object with HA positive 30 minutes passed overhead 30 minutes prior. To avoid the telescope's physical HA limits, `RoboSpec`'s target selection logic ignores any object with HA more than two and a half hours from the meridian.

`RoboSpec` will never be capable of solving every problem that could occur during observing, so there is a built-in 'Safe Mode' that closes the observatory dome, disables tracking, and notifies the user via text message of an error. For some minor problems with simple and reliable solutions, Safe Mode is unnecessary. But more serious problems such as unsuccessful target acquisition, the GAM failing to reach its instructed position, a power outage, a local network disconnection, or TCS crashing require the attention, and often the physical presence, of a human.

Target Selection Efficiency is another aspect of robotic intelligence in which we sought to emulate a human observer. `RoboSpec` should be able to evaluate a list of desired target stars and choose the most appropriate one to observe next. It searches typically for stars that are near the meridian, but ultimately chooses a target based on a number of criteria, discussed in more depth in §4.4.4 below.

With target selection and acquisition programmed, and safety measures in place to protect the observatory, the process of developing `RoboSpec` was straightforward. It involved combining small single-function programs, some trial-and-error solutions, and substantial trouble-shooting. The remainder of this chapter is dedicated to a description of the hardware, software, and communication methods `RoboSpec` uses.

4.2 Hardware

A notable early concern for `RoboSpec` was establishing stable communication and coordination with the hardware and software systems already in place at DSO, while introducing as few changes as possible to those systems. To that end, we chose to write `RoboSpec` in the AutoIt programming language. From the program website: "AutoIt...is a freeware BASIC-like scripting language designed for automating the Windows GUI and general scripting. It uses a combination of simulated keystrokes, mouse movement and window/control manipulation in order to automate tasks in a way not possible or reliable with other languages"[11]. It was thus a reasonable platform for our program, to make an observer's interaction with the robotic system minimal and straightforward.

AutoIt also handles communication between the existing hardware and software systems with surprisingly few problems. `RoboSpec` must communicate via TCP/IP, X-10, web switches, and external software 'hooks' on various control programs, and AutoIt has proven capable of using all those protocols. The following paragraphs detail the hardware and software systems installed at DSO, the purpose, implementation, and challenges of each, and how `RoboSpec` communicates with them.

Computers All DSO computers are connected to a local network, simplifying communication between them. The control room for the 32-inch telescope contains three computers used in the observing process. The first is the DSO-Data computer, which is home to software including `RoboSpec`, MaxIm DL 6, and TheSky.

[11] AutoIt Consulting Ltd 2015, 'AutoIt,' `https://www.autoitscript.com/site/autoit/`

Since MaxIm DL controls the main CCD and guide CCD cameras, images from those cameras are saved to DSO-Data. During manual observing, most user interaction occurs with this machine.

The second computer is DSO-TCS, and handles commands to the Telescope Control System introduced in §3.1. It receives instructions from DSO-Data via ethernet Transmission Control Protocol/Internet Protocol (TCP/IP) socket connections.

The third computer is DSO-X10, which handles communication with the X-10 controller, and with MaxIm DL 5 operating the East Port camera. Like the TCS computer, it receives commands from DSO-Data via TCP/IP socket. It is connected directly to the X-10 controller, and contains Activehome software for scripted control of X-10 devices. Activehome Command (ahcmd.exe) is provided by X-10 and is installed on the DSO-X10 computer[12].

During manual observing, the user also controls web switches from a webpage on DSO-X10. Web power switches are provided by Digital Loggers and resemble a power strip, containing eight remote-addressable switching outlets and two non-switchable outlets. During automated observing, `RoboSpec` operates these switches from DSO-Data via a local IP address and port number. `RoboSpec` passes commands as arguments to uu.W32.exe, a program provided by Digital Loggers, the manufacturer of the web switch. Commands take the form "uu.W32.exe 123.456.0.789:10 username:password 5on" indicating in this case that switch 5 on the web switch with IP address 123.456.0.789 should be turned on.

[12]X10.com 2015, 'X-10 Basics,' `https://www.x10.com/x10-basics.html`

Three other computers that are relevant to `RoboSpec`'s operation execute various weather-monitoring procedures. However `RoboSpec` does not communicate directly with those machines; that task has been delegated to `RoboticWeatherman`, described in Chapter 5. One such computer is DSO-Cloud32, which additionally displays weather information on a computer monitor in the observatory control room.

Telescope Mount; Observatory Dome; TCS The 32-inch telescope is a Ritchey-Creitien reflecting telescope, and was built by DFM Engineering[13]. It sits on a yoke equatorial mount, with one axis parallel to the earth's rotation axis. This configuration allows the telescope to match the sky's rotation using a single motor, in contrast to altitude-azimuth mounts that must rotate about two axes.

Any movement of the telescope mount is controlled by the telescope control system (TCS). TCS was also built by DFM Engineering, and uses a proprietary communication protocol outlined in its user manual. It must be connected to a computer at the observatory, and commands are sent from that computer via serial port communication or TCP/IP communication. Each command known to TCS has a unique number, and additional arguments are also specified with numeric values. For example, command 21 is the GUIDER command, which controls the motion of the observatory dome. Sending "#21,0;" via TCP/IP connection instructs TCS to have the dome opening track the telescope's pointing.

Guide and Acquisition Module Attached behind the telescope's primary mirror is the guide and acquisition module (GAM). It houses an angled mirror on a

[13]Caton 2016, `http://dso.appstate.edu/facilities/32-inch-telescope`

translation stage, capable of directing light from the telescope's secondary mirror out one of several ports. Its purpose is to allow multiple instruments to be connected to the telescope simultaneously, and accessed quickly and easily. Moving the GAM mirror to a new position is controlled with a command to TCS, and takes about 30 seconds.

A separate TCS command queries the GAM for its status, corresponding to its position; that status is updated once the GAM successfully reaches its new position. However, the mirror occasionally fails to reach the position specified in the movement command. The error can be identified by querying the GAM's position and receiving its previous position. In most cases, repeating the TCS command is sufficient to complete the movement, but requires another 30 second delay before querying the GAM's position again.

The instruments that remain attached to the GAM are the East Port CCD camera described below and the arc lamp described in §3.2.2. The East Port camera is attached to the GAM east port. The arc lamp is affixed to the bottom of the GAM and passes light through a hole in the device housing, but does require that the translating mirror be in a certain position. Other instruments attached to the GAM depend on the type of observing to be performed. For photometry, an imaging CCD camera and filter wheel are attached. For spectroscopy, the GM Spectrograph and a different CCD camera are fitted in place of the filter wheel/CCD combination. Both attach to the direct cassegrain port of the GAM, which serves as a straight pass-through for light coming from the telescope's secondary mirror.

Building Lights; Dome, Telescope, & Tracking Power The observatory building has a number of electrical systems that can be controlled using X-10 communication protocol. The observatory's X-10 controller sends commands over the electrical wiring itself[12]. Each receiver has its own address, and ignores commands sent to any other address. To communicate a command, the controller first sends the receiver's address, then the command to be executed.

`RoboSpec` communicates with `TCP_Camera3` on the X-10 computer using a TCP/IP connection. For example, to turn off dome lights, `RoboSpec` sends command "2,3" over a TCP/IP socket. The command is received by `TCP_Camera3` and relayed to the physical X-10 controller installed at the observatory. X-10 communication is slow; commands are sent directly over existing electrical wiring, and are synchronized with the standard 60Hz AC power line signal. Each single command requires at least 25 power line cycles. Thus most commands are transmitted over roughly half a second (Rye, 2015). Fortunately X-10 devices are used infrequently during observation, so communication speed is of little concern.

X-10 devices installed at DSO are the lights inside the telescope dome, as well as relays connecting power to the motors controlling motion of the telescope mount and dome. During the normal observing procedure, they are appropriately deactivated or activated by `RoboSpec` at the beginning of the night and at dawn.

Comparison Lamp & Solenoid As mentioned above, the comparison arc lamp is powered with a web switch. The lamp's housing has an aperture that serves as a dust cover, and must be opened before a comparison spectrum can be acquired. The housing is attached to a solenoid, also powered by a web switch.

CCDs & Spectrograph The main spectroscopy CCD is an Andor Apogee device, and is mounted on the spectrograph. The East Port and guide CCD cameras were built by Santa Barbara Instrument Group (SBIG), a division of Diffraction Limited[14]. The guide CCD is mounted on the north port of the GAM, and has a 40x30 arc-second field of view. It images light reflected off the spectrograph slit mask.

The third CCD is the East Port camera, which was installed most recently. It is mounted on the east port of the GAM, and has a field of view of roughly 4x5 arc-minutes, substantially wider than the guide CCD. As mentioned in §4.1, it allows `RoboSpec` to acquire targets reliably. All three CCD cameras are controlled by MaxIm DL, a program developed and provided by Diffraction Limited.

4.3 Software

RoboSpec Executable File The latest version of `RoboSpec` is found on the Desktop of the DSO-Data computer, and is a compiled Windows executable file. AutoIt scripts can be run without being compiled; the standard AutoIt downloadable package includes an interpreter that will run scripts without compilation. However, compiling `RoboSpec` provides several advantages, the most apparent of which is a Windows executable file. Even without any astronomy background, locating and running `RoboSpec` will be familiar to anyone with experience using Microsoft Windows.

[14]Diffraction Limited 2016, `http://www.cyanogen.com/`

Efficiency is another benefit; the compiled program uses fewer computer resources while running than does an interpreted script. We have not encountered any related issues with the current version of `RoboSpec`. But DSO-Data hosts TheSky and MaxIm DL, both of which are vital to telescope operation. Running a compiled executable, which requires less processing power, is in line with our design principal of being as unobtrusive as possible to existing observatory systems.

Compilation also provides a level of security. AutoIt scripts can be opened and altered in any text editor. Doing so would almost certainly prevent the script from operating properly, if at all. But the compiled executable cannot be so easily edited. `RoboSpec` also includes IP addresses, usernames, and passwords that are required to perform various observing tasks. Compiling the AutoIt script prevents these private data from being accessed.

MaxIm DL 6 A single instance of MaxIm DL Version 6 controls the main CCD and guide CCD. The program displays its own GUI when executed, but can also be addressed by a script such as `RoboSpec`. Scripting MaxIm DL is done through ActiveX, and is compliant with the Astronomy Common Object Model (ASCOM) standard for instrument control[15]. The ASCOM platform allows "vendor-independent and language-independent plug-and-play compatibility between astronomy software and astronomical instruments on Windows computers"[16]

[15]MaxIm DL 2016, Scripting, MaxIm DL Version 6 User Manual, 'Introduction to MaxIm DL,' `http://www.cyanogen.com/help/maximdl/MaxIm-DL.htm`

[16]ASCOM Standards for Astronomy, 'About the ASCOM Initiative,' `http://ascom-standards.org/About/Index.htm`

A limitation of MaxIm DL is its inability to operate more than two cameras. However that was not a concern at DSO until the East Port camera was installed for this automation project. At that time, we explored several other software options for camera control, hoping simply to operate the East Port camera using a separate program on DSO-Data. However, we discovered that three cameras cannot be controlled from the same computer, even using different software.

TCP_Camera3 Rather than delve into the problem too deeply, we opted to control the East Port camera from an instance of MaxIm DL Version 5 on the DSO-X10 computer. When `RoboSpec` begins observing, it attempts to open a TCP connection to the script `TCP_Camera3`. Thus the user must already have manually started `TCP_Camera3` on DSO-X10. Its initial purpose was to receive instructions for the East Port camera from `RoboSpec` and relay them to MaxIm DL 5. But once the scripts necessary to communicate with the X-10 controller via software were functional, they were integrated into `TCP_Camera3` as well.

We also discovered that sending an image acquired by the East Port camera over TCP/IP connection, or even through a shared local network location, was slow compared to `RoboSpec`'s normal operation. In the interest of efficiency, the responsibility of analyzing East Port images was delegated to `TCP_Camera3`. The process of acquiring and analyzing an East Port image and returning the desired coordinates to `RoboSpec` requires a matter of seconds, rather than the minutes it could take to transfer the image.

When plate solving is integrated into `RoboSpec`, a future version of `TCP_Camera3` will perform that function as well. `RoboSpec` will send the RA/Dec sky coordinates

of the desired object to `TCP_Camera3`, which will then acquire an East Port image. It will pass the image to a local copy of Astrometry.net, receiving in return the RA/Dec coordinates of the center of the image. Knowing the pixel scale of the East Port image, `TCP_Camera3` will then calculate the pixel coordinates of the desired object, and return those values to `RoboSpec`. As mentioned in §4.1, plate solving will be necessary when attempting to observe stars that are not the brightest object in the nearby field.

It was necessary to incorporate 'handshaking' into the communication between `RoboSpec` and `TCP_Camera3`. For example, `RoboSpec` may wish the close the TCP/IP connection. Before doing so, it sends `TCP_Camera3` a 'Listen' command, indicating that it should also resume listening on that port for for new instructions from `RoboSpec`. Otherwise `TCP_Camera3` ceases listening on the port as soon as `RoboSpec` disconnects, and ignores any further instructions.

Guiding Another concern was MaxIm DL's telescope guiding algorithm. It was designed to keep a single bright object at the same place on a telescope's CCD. But the guide camera on the 32-inch telescope receives light reflected off the spectrograph slit mask, producing a bright object with a dark bar across it. It appears as two adjacent bright objects, a geometry MaxIm DL was not specifically programmed to interpret. Nonetheless, its standard guiding method has proven reliable, and so was not modified or replaced for `RoboSpec`.

TheSky TheSky contains a catalog of astronomical objects, and can send ASCOM commands if desired. Since TCS uses its own proprietary command library,

ThySky's ASCOM functionality is not used at DSO. However the normal observing procedure does use TheSky for the majority of telescope control.

`RoboSpec` creates two ActiveX connections to TheSky. One allows queries to and responses from TheSky's object catalog, primarily containing high-precision RA/Dec coordinates for targets. The other connection communicates instructions to TheSky to execute various telescope commands, such as Jog and Slew[17]. TheSky then relays those movement commands to TCS.

Observers may find that a few of the objects on their target list do not exist in TheSky's object catalog. In that case, when `RoboSpec` queries TheSky for the target's coordinates, it receives both RA and Dec coordinates of identically 0°. For `RoboSpec` to attempt to observe the object, the observer must have provided high-precision RA/Dec coordinates on the target list, rather than the normal low-precision coordinates used during target selection. If sufficiently-precise coordinates are available, `RoboSpec` will attempt to observe the object; otherwise it will be ignored and a new target chosen.

Cygwin AutoIt is a higher-level programming language than ANSI-C. Though it is easy to use and well-suited to GUI creation and communication between devices, it is comparatively sluggish when performing image analysis or mathematically intensive calculations. `RoboSpec` must occasionally locate a star in an image, or determine the local sidereal time. For such applications, the ability to execute a simple program written in C is desirable. Standard installations of the Microsoft

[17]Software Bisque, Inc 2016, 'Script TheSkyX,' `http://www.bisque.com/scripttheskyx/index.html`

Windows operating system contain no C compiler, so Cygwin is used. Cygwin is "a large collection of GNU and Open Source tools which provide functionality similar to a Linux distribution on Windows"[18].

A particularly useful feature of Cygwin is its ability to create Windows executable files when compiling a C program. Thus the compiled program becomes a standalone executable, and Cygwin need not be installed on the computer running said program. Having an executable also prevents inadvertent modification of the program code, as with `RoboSpec` itself; the executable cannot easily be edited after compilation. In addition, it runs significantly faster than an interpreted program, such as the same algorithm written in AutoIt.

NewTwilight & SiderealTime `RoboSpec` makes use of several C programs during normal observing. The first executed during the night is called `newtwilight`, and returns the times of evening and morning nautical twilight. Another script called `siderealtime` returns the local sidereal time (LST), which corresponds to the RA coordinate of astronomical objects on the local meridian. `RoboSpec` attempts to observe targets as close to the meridian as possible, and thus prioritizes targets with RA nearest the local sidereal time. The difference between an object's RA and the LST is identical to the object's hour angle, mentioned in §4.1.

StarFind3 A third program called `starfind3` analyzes the most recently-acquired image from the guide CCD camera, and returns the coordinates of the brightest star in that field. Retrieving the guide image requires a second ActiveX connection to MaxIm DL. The image is saved in an unusual format, perhaps due to an expec-

[18]Red Hat, Inc. 2015, 'Cygwin,' `https://www.cygwin.com/`

tation that observers won't care to use the guide image themselves. However once it is found, `starfind3` allows `RoboSpec` to calculate the adjustment to telescope pointing required to place the star on the spectrograph slit.

The algorithm `starfind3` implements is fairly simple. It first creates two arrays of values by 'collapsing' the image horizontally (by summing each row) and vertically (by summing each column). Each array is then smoothed with a 'boxcar,' which averages each element with several adjacent elements. The script then scans each array for the highest value, and returns those pixel numbers as the horizontal and vertical coordinates of the star. This works quite well for the guide camera image, even when the image contains multiple stars, as long as the target of interest is the brightest object.

Another vital task, namely updating the coordinates of the spectrograph slit, is also the responsibility of `starfind3`. Since the GM Spectrograph is mounted to the telescope rather than bench-mounted, it experiences mechanical flexure as the telescope pointing changes. The coordinates of the spectrograph slit can vary by as much as five pixels. Some targets may not be even five pixels in diameter, so knowing an accurate slit location is essential. Fortunately, determining those coordinates is a simple task.

Once `starfind3` detects the star, it begins searching the vertical array for a minimum between two peaks, indicating the spectrograph slit. It calculates the average position of the two peaks, weighted by their heights. The result is the estimated slit position, which it converts to an integer pixel number and returns

to `RoboSpec`. This allows `RoboSpec` to update its vertical coordinate of the spectrograph slit and maintain accurate pointing.

Image2xy Also useful for identifying stars using the East Port camera is a C program available from Astrometry.net called `image2xy`. `TCP_Camera3` makes use of a locally-installed version, although Astrometry.net functionality can be accessed by uploading an image to an internet server. By itself, `image2xy` attempts to identify star-like sources in an image, while ignoring cosmic rays. It returns a list of pixel coordinates of probable sources and the estimated brightness of each. A custom C program called `findbright` invokes `image2xy`, searches the output list for the brightest object, and returns its coordinates to `TCP_Camera3`.

SNSpectra A newer feature to `RoboSpec` is an estimate of the number of counts, corresponding to the amount of light, received in a certain wavelength region of acquired spectra. The calculation is handled by the C program `SNspectra`. It first sums a specific range of pixel values in the violet region of the spectrum. It then estimates the background counts by summing an equal-sized range away from the spectrum. Finally it subtracts the background counts from the violet-region counts, then divides by the number of pixel columns in the sum. The result is an estimate of counts per pixel along the dispersion direction of the spectrograph.

`RoboSpec` can use a target's aggregate counts to optimize its use of time. Each target starts at zero counts, with each new exposure adding to the total as calculated by `SNSpectra`. If the user's research requires a certain minimum number of counts per object, `RoboSpec` can attempt to observe each target until that threshold is reached, or it reaches some maximum number of exposures. This flexibility

eliminates extraneous observations during excellent weather conditions, and improves data consistency in light of less-favorable conditions.

Email Text Message Notifications Included in `RoboSpec` are several functions that send automated emails to the user assigned to the telescope. They use Simple Mail Transfer Protocol (SMTP), and require an existing email address[19]. Every cell phone has a unique email address, and receives emails sent to that address as text messages. Thus `RoboSpec` can notify a sleeping user of hardware problems.

Automatic File Transfer Just as a human observer does after a night of manual observing, `RoboSpec` compresses the night's data and sends it to the user's personal computer. UNIX commands TAR and GZIP are executed on Windows via Cygwin. The TAR command archives the desired images into a single file, then GZIP compresses them to allow for faster download. File transfer is performed using WinSCP, which is scripted for automatic operation through its COM library.

Ideally, data will be present on campus by the time the user arrives in the morning. An automatic data reduction pipeline will eventually be implemented as well, and will provide the user with reduced data by late morning. That will allow the user to adjust the target list for the following night, should any targets warrant immediate repeated observation.

[19]Postel, J.B. 1982, The Internet Engineering Task Force, 'Simple Mail Transfer Protocol,' `https://www.ietf.org/rfc/rfc821.txt`

4.4 GUI Control and Normal Operation

`RoboSpec` started as a simple program. As it gained functionality, it necessarily began to require more information from the user. To prevent filling too much of the available screen space on the DSO-Data computer monitor, we have created a tabbed GUI for `RoboSpec`, shown in Figures 4.1, 4.2, and 4.3. Its three tabs are described individually below, in the context of the normal procedure required to prepare and carry out a night of automated spectroscopy.

4.4.1 User Tab

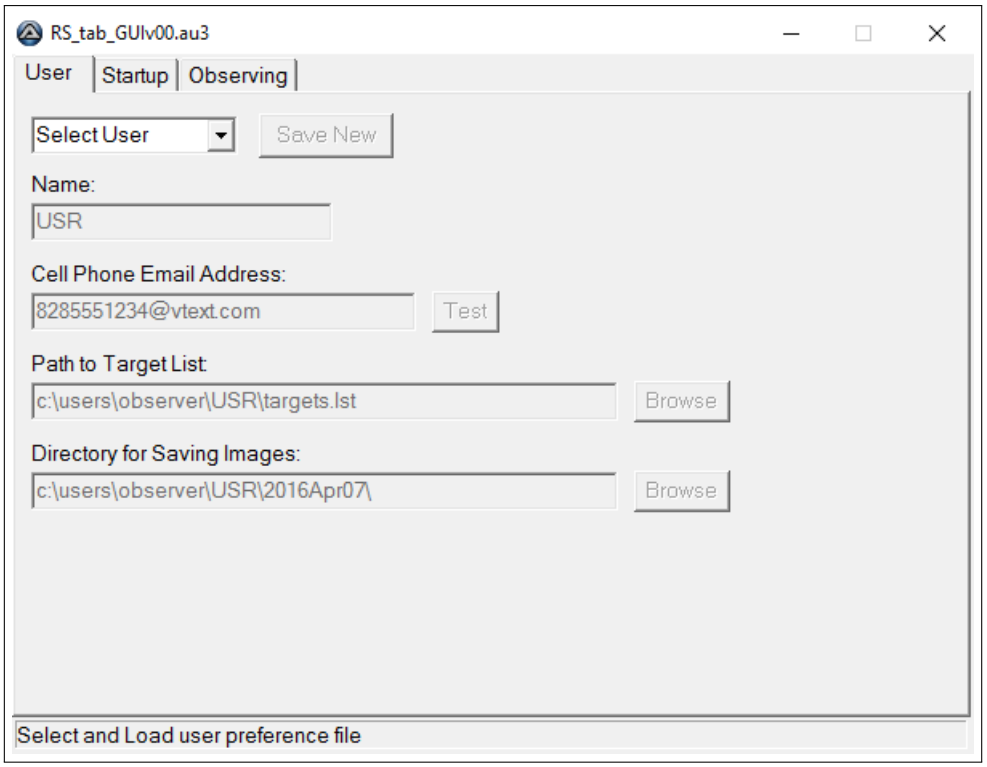


Figure 4.1: User tab of `RoboticSpectroscopist`, where the user may select an existing profile, or create a new one.

An observer may run the `RoboSpec` executable file at any time during the day preceding a night of observing. This allows e.g., faculty with late afternoon meetings to provide `RoboSpec` with all the required information, click 'Begin Observing,' and let the script operate until morning. This convenience is contingent on the spectroscopy equipment already being installed on the telescope, which is often the case due to consecutive nights being allocated to astronomical spectroscopists.

Existing User `RoboSpec` initially displays the 'User' tab shown in Figure 4.1, the data boxes filled with example values. A drop-down combobox lists the user preference files, or profiles, that `RoboSpec` was able to identify. If the user's profile exists, selecting it in the combobox will automatically load that profile and fill in the appropriate data boxes. The data boxes will also be enabled for editing, should an existing user want to save images to a different directory, for example. Any changes will be temporary; to make permanent changes, the user must create a new profile with a different name, or edit the profile text file manually.

Selecting an existing user also enables the 'Test' button and the two 'Browse' buttons. The Test button sends a sample text message to the email address in the 'Cell Phone Email Address' box, to verify delivery of notifications messages. `RoboSpec` uses this feature to notify the user of problems it cannot solve without human intervention, such as the GAM being mechanically stuck or TCS having crashed.

The Browse buttons behave as expected of the Windows operating system. The button next to the 'Path to Target List' data box presents the user with a familiar 'Open File' dialog box. The user may then select a single file and click 'OK,'

returning that file's path to `RoboSpec`. The button next to the 'Directory for Saving Images' data box presents a similar Open File dialog box, but requires the user to select a directory rather than a file.

New User If DSO is hosting a visiting observer, he or she likely will not have a `RoboSpec` user profile. For that purpose, the Select User combobox includes a 'new' option. Selecting it enables all the buttons on the User tab, including the 'Save New' button, and clears all the data boxes. The new user may then fill in the data boxes as appropriate, and click the Save New button to create a profile text file. The created profile will appear immediately in the combobox, under whatever text was entered into the 'Name' field.

Should there be a problem with the information in one of the data boxes when the user clicks Save New, `RoboSpec` will display a Message Box indicating which field or fields caused the error, and what the error was. Errors include: any fields left blank; name already taken; target list nonexistent; directory for saving images not found. When the user has resolved each displayed error, the profile will save successfully, and the Status bar at the bottom of the GUI will display a confirmation message.

Loading or creating a user profile is currently the entire purpose of the User tab. The tab may seem somewhat empty, but extra space is left for controls to be added in the future. For example, the FTP information required for automatic file archival and transfer is hard-coded into `RoboSpec`, and is not part of each user's profile. Should the script be made capable of transferring archived images to any

user, the necessary inputs will be added to the tab. But at present, once a profile is loaded or created, the user may move on to the Startup tab.

4.4.2 Startup Tab

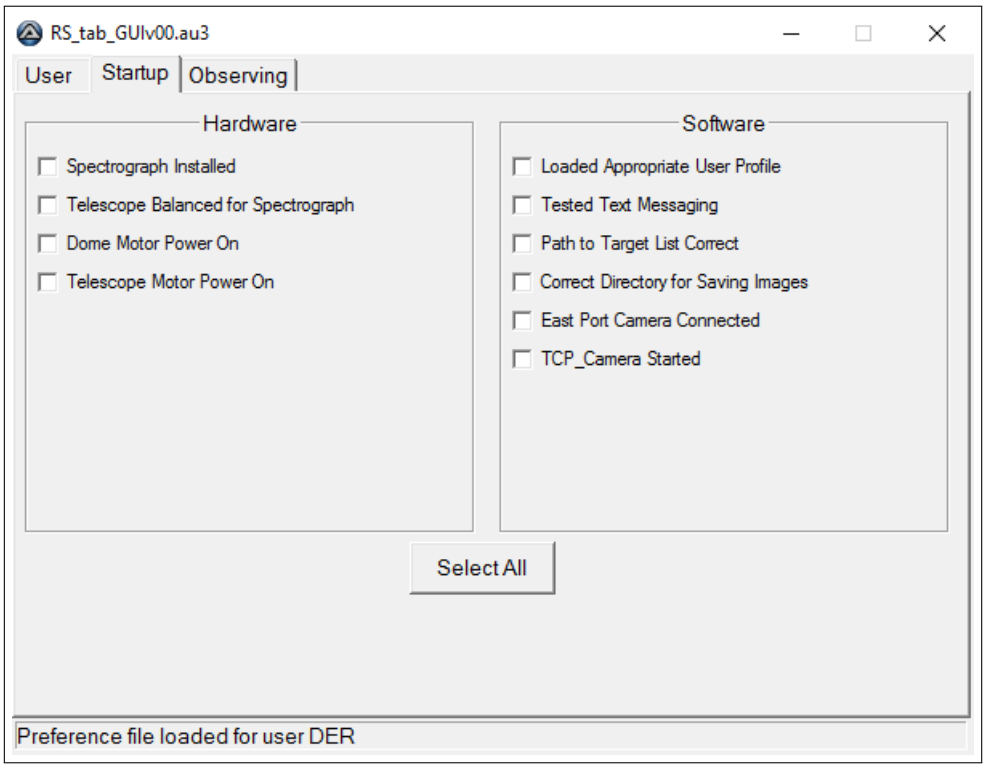


Figure 4.2: Startup tab of `RoboticSpectroscopist`, which includes beginning-of-night checklist items. `RoboSpec` will not begin its normal observing procedure until all items are checked.

The GUI 'Startup' tab includes a checklist of beginning-of-night startup tasks required for spectroscopy, shown in Figure 4.2. Experienced observers will not need the list, but it may be a good reference for unfamiliar users while installing hardware and starting software. As each item is completed, the user may check the

box next to it. `RoboSpec` takes no action regarding the checkboxes until attempting to begin observing.

As a precaution, `RoboSpec` prevents the user from initiating observations without checking the box next to every item on the list. Some items `RoboSpec` could not detect immediately, such as 'Dome Motor Power On.' Others, like 'Loaded Appropriate User Profile,' it would never detect. As a convenience to experienced users, a 'Select All' button is included, which checks all the boxes at once. However, it is good practice to verify each item manually.

Hardware `RoboSpec`'s startup checklist includes four hardware items, the first being 'Spectrograph Installed.' If the spectrograph is not already attached to the telescope, the user must carefully remove and store the photometry CCD and filter wheel, install the spectrograph, and rebalance the telescope.

The second hardware item is the rebalancing of the telescope. With the spectrograph attached, the telescope's center of mass is different than when the CCD and filter wheel are attached. Unless the telescope's center of mass coincides with its axis of rotation, the mount motors are incapable of raising it. Thus to counterbalance different instruments, a sliding weight is mounted on the side of the telescope. After installing the spectrograph, the user must adjust this weight to the position prescribed to rebalance the telescope.

The third and fourth hardware items are power connections for the dome motor and telescope motor. Both are controlled with X-10 switches from the DSO-X10 computer. `RoboSpec` could be made to click the buttons automatically; AutoIt

includes the ability to effect mouse clicks at any given location on the computer screen. However, a more reliable solution is to allow `RoboSpec` to interface with the Activehome Command X-10 control software installed on DSO-X10, and activate these switches automatically. Manual verification is nevertheless left to the user; observing would be impossible without the telescope and dome being powered.

Software There are currently six software items on `RoboSpec`'s startup checklist. The first asks the user to verify having loaded the correct user profile, or having successfully created a new profile. The three subsequent items require verification of each piece of information in the profile. An accurate cell phone email address prevents a non-assigned user from receiving text messages regarding `RoboSpec`'s operation in the middle of the night. Without a valid target list, `RoboSpec` cannot observe at all. And without a valid directory for saving images, the night's data could be hidden somewhere on the computer, or lost entirely.

The fifth software item is for connecting the East Port CCD camera, which `RoboSpec` uses to acquire a target. Without it, the telescope's pointing is insufficient to place most objects within the guide camera's field-of-view. The sixth item relates to the East Port camera, and asks the user to start `TCP_Camera3` manually on DSO-X10. This is the software that receives and executes X-10 and East Port commands from `RoboSpec`. Without `TCP_Camera3` running, `RoboSpec` cannot control X-10 devices or the East Port camera. Like with the dome and telescope motor power, executing `TCP_Camera3` automatically might be possible, but takes only a few seconds manually.

When the user has performed all the tasks listed on the Startup tab, and has checked the box next to each, the observatory will nearly be ready for automated observing. The few remaining tasks are found on the Observing tab.

4.4.3 Observing Tab

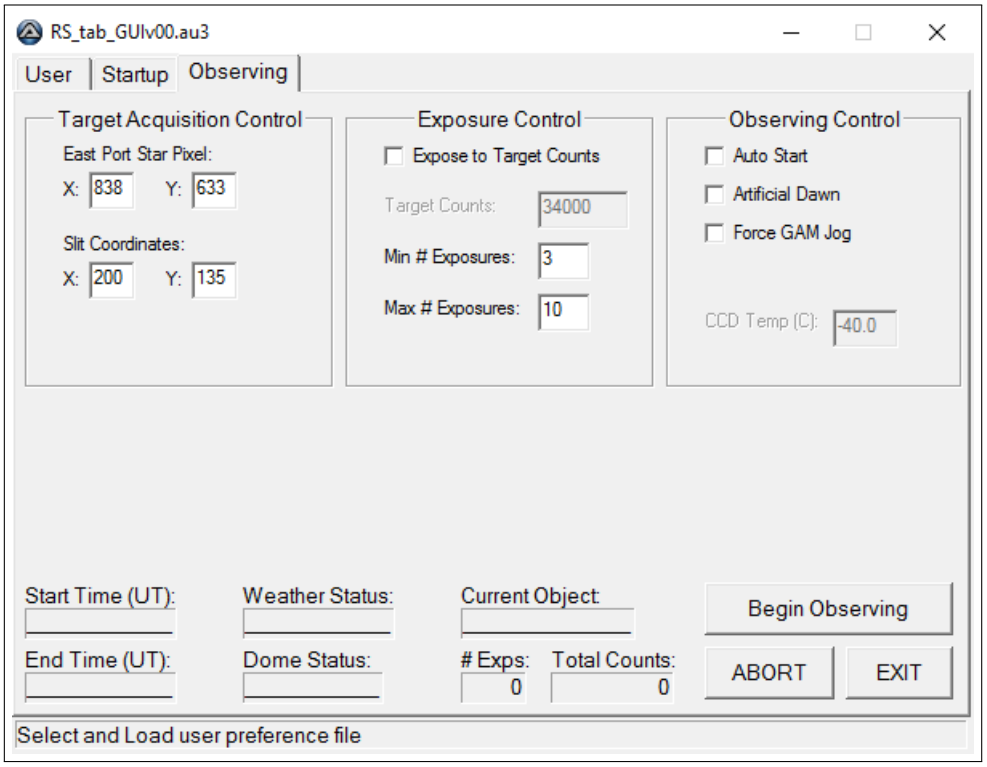


Figure 4.3: Observing tab of `RoboticSpectroscopist`, containing most of the controls and indicators necessary to adjust automated observations.

The final tab on the `RoboSpec` GUI is the 'Observing' tab. It contains indicators a user might want to see while `RoboSpec` operates, as well as controls used to begin, abort, and modify observations.

Target Acquisition Control Unlike the East Port camera, which is installed indefinitely on the GAM, the spectrograph is reinstalled after photometry observing. Doing so changes the path of light from the telescope through the instruments. Consequently, the user must make minute adjustments to the position of the guide CCD camera, to center the spectrograph slit in guide images. Since the user makes these adjustments by hand, the pixel coordinates of the slit center generally are not the same as after the previous installation.

`RoboSpec` needs those coordinates to properly place a target on the slit. Therefore after installing and adjusting the spectrograph, the user must acquire an image from the guide camera, find the pixel coordinates of the slit center, and enter those values into the 'Slit Coordinates' boxes in the Target Acquisition group.

The uppermost data boxes in the Target Acquisition Control group are for the 'East Port Star Pixel' coordinates. They correspond to the location of a star in an East Port image, when it is centered on the spectrograph slit as seen in the guide image. `RoboSpec` uses those coordinates during target acquisition to position a star near to the slit.

To determine the East Port Star Pixel coordinates, the user must first install and adjust the spectrograph as described above. Next, the telescope must be pointed at a bright star, the star placed on the slit, and telescope tracking and guiding enabled. The user should then acquire an East Port image and identify the pixel coordinates of the star's approximate center. Those are the values to be entered into the East Port Star Pixel data boxes.

Exposure Control The Exposure Control group contains criteria dictating the number of exposures `RoboSpec` should take on a single target. The 'Min # Exposures' and 'Max # Exposures' indicate the minimum and maximum number of exposures, respectively, that the user will allow for any given target.

The 'Expose to Target Counts' checkbox instructs `RoboSpec` to continue acquiring spectra of a target, until the 'Total Counts' estimated by `SNSpectra` reaches the value in the 'Target Counts' data box. However the limit imposed by Max # Exposures takes precedence over Exposure to Target Counts, and will force `RoboSpec` to move on to its next target.

Observing Control Observing Control contains modifications to `RoboSpec`'s normal observing procedure. When implemented, the 'Auto Start' checkbox will be particularly useful for busy observers, who may not have time to locate the pixel values for Target Acquisition Control. Auto Start will instruct `RoboSpec` to attempt automated identification of the East Port Star Pixel and Slit Coordinates. Preliminary tests have shown Auto Start to be successful, when the spectrograph is properly installed and adjusted.

Auto Start will have other necessary start-up tasks to perform as well, such as starting and monitoring the CCD cooler. Selecting the Auto Start checkbox will therefore enable the 'CCD Temp (C)' data box. There, the user may set the desired temperature for the CCD cooler to maintain. If the cooler fails to reach the set temperature, `RoboSpec` will notify the user that human intervention is required.

The second checkbox, 'Artificial Dawn,' forces `RoboSpec` to terminate its regular observing routine, perform end-of-night calibrations, and shut down. This feature is useful when inclement weather arrives after a clear period, and the forecast shows no improvement. Rather than let `RoboSpec` sit until dawn, needlessly checking `RoboticWeatherman`'s output file for instruction, the user may select Artificial Dawn and go to bed.

The 'Force GAM Jog' checkbox preempts the GAM failing to reach its position after the first command, and sends a second command immediately after the first completes. The process requires roughly 30 seconds extra, but may prevent the GAM from becoming stuck and requiring human intervention.

Indicators Several indicators are included on the Observing tab to provide the user with real-time information relevant to observation. 'Start Time' and 'End Time' are calculated by `newtwilight`, and display the UTC times of evening and morning nautical twilight, respectively. These are the times that `RoboSpec` will start and stop spectroscopic observations, although it can perform calibrations outside this window.

The 'Weather Status' indicator shows the most recent value `RoboSpec` receives from `RWeather`; clear, partly cloudy, very cloudy, or precipitation. The 'Dome Status' indicator shows the expected status of the observatory dome; open or closed. No sensors are attached to the dome itself, so Dome Status updates when a dome control command is sent to TCS. Verifying the accuracy of the displayed status is left to the user; `RoboSpec` operates as if all commands are successful. We

are not immediately concerned about this assumption; in 20 years of observing, a dome command has been ignored once[20].

The name of the target being observed is displayed by the 'Current Object' indicator. This is the name sent to TheSky when querying high-precision coordinates. Below it are '# Exps,' indicating the number of spectrum exposures acquired for the current target, and 'Total Counts,' indicating the estimated aggregate counts in the spectra for that target. None of the Observing tab indicators are essential to `RoboSpec`'s operation. However they provide the user some insight into `RoboSpec`'s actions, such as pausing observation or moving unexpectedly to its next target.

Buttons There are three buttons on the Observing tab. The largest is the 'Begin Observing' button. Clicking it will not always initiate observing immediately, as `RoboSpec` requires that several conditions be met first. In particular, all checkbox items on the Startup tab must be checked. If they are, `RoboSpec` will enter its main loop, but will delay initial calibrations until some time before Start Time. If the weather is clear when calibrations complete, `RoboSpec` will open the observatory dome and begin its normal observing routine.

The 'Abort' button is an old feature of `RoboSpec`, but a new addition to the GUI. Some of `RoboSpec`'s common actions, such as acquiring a five-minute exposure of a star, prevent the GUI from responding to inputs for their duration. In fact, inputs seem to be lost entirely if not received in a timely manner, with the longest feasible delay determined experimentally to be about one second. This behavior is a consequence of `RoboSpec`'s script-based rather than event-driven design. Thus

[20] Gray, R.O. 2016, Private communication

an abort button on the GUI seemed infeasible, and was introduced as a standalone script with its own GUI.

`RoboSpec` executed the old Abort script automatically when it initialized, then read a single value from a text file at various points in its operation. Clicking the button on the Abort GUI changed that value, signaling to `RoboSpec` that it should abort its current action. It was a functional, albeit inelegant solution.

Fortunately, AutoIt provides a 'push-like' property for checkboxes. A push-like checkbox appears as a button, but 'latches' when clicked. This behavior allowed us to eliminate the Abort GUI, and instead have `RoboSpec` query the state of the Abort button: unchecked or checked. In addition, the button remains operable even during a long exposure. If the user hastily clicks the Abort button, then realizes there is no reason to abort, clicking the button again before `RoboSpec` queries its state will prevent an unnecessary interruption to observing.

The last control on the Observing tab is the 'EXIT' button. Clicking it closes `RoboSpec`, but not before ensuring all connected systems are in an appropriate state, and then closing those connections properly. When the user checks on `RoboSpec` the morning after a night of observing, the observatory systems will likely already be disconnected, and the program will be idling. In that case, the EXIT button simply closes the script as expected.

4.4.4 Necessary Files

Throughout its normal observing procedure, `RoboSpec` reads information from several different text files, which have already been mentioned in the description of its operation. More detail is provided in this section.

User Profile User preference files or profiles are intended to simplify the process of preparing `RoboSpec` for observing. They contain basic user information that will rarely change. The Directory for Saving Images is an exception; a separate directory is generally created for each night of observing. Thus the user could save the parent directory to the profile, then simply append the night's directory name as part of evening preparations.

Target List The user must provide `RoboSpec` with a list of targets to be observed. `RoboSpec` invokes the `DetermineObservable` function to scan the list and select the best candidates for observing. In addition to an object's name, data required on the target list are low precision RA/Dec coordinates, desired number of exposures, and a user-determined priority level.

The low precision RA/Dec coordinates are used to calculate the object's approximate hour angle. As mentioned in §4.1, `RoboSpec` will not attempt to observe any target more than two and a half hours from the local meridian. Rather than have `DetermineObservable` query TheSky for high-precision RA/Dec coordinates of every object on the target list, it uses low-precision values to produce a narrower range of candidates.

If Expose to Target Counts is not selected on the Observing tab, `RoboSpec` will acquire as many exposures as are specified on the target list. It will ignore the minimum and maximum number of exposures specified on the Observing tab, and those input boxes will be disabled.

Finally, the user must provide a priority level for each object on the target list. Possible values are 0 indicating 'Don't Observe,' 1 indicating 'Highest Priority,' and 2 indicating 'Lowest Priority.' The ability to indicate Don't Observe allows the user to compile a long list including all objects desired for a particular research project, then select a subset with non-zero priority flags. In addition, objects that have been observed sufficiently may be removed from consideration by setting their priority to zero.

The non-zero priority flags simply require `RoboSpec` to consider certain objects before others. There has not yet been a need for more than two priority levels, but such a system could be required for queue-based observing. With targets from multiple users, a third value indicating 'Highest Priority' might be awarded to whichever observer had been assigned telescope time on a given night. For example, one user's target list might contain objects that are less likely to be observed, e.g., simply based on their sky position. A Highest Priority would ensure that all users' targets receive roughly equal consideration by `RoboSpec`. However `RoboSpec` is not yet capable of queue-based observing, and so does not currently need a flag to indicate user priority.

Weather Condition File As mentioned in §4.1, we have delegated the responsibility of monitoring the weather to `RoboticWeatherman`. Communication be-

tween that separate script and `RoboSpec` occurs through third text file, called 'weather.txt.' It contains a five-digit number, each digit providing information about the weather conditions. The third digit contains `RWeather`'s instruction to `RoboSpec`. The other digits provide corollary information, which is instructive to a user being informed of a problem. For a detailed description of each digit's purpose, see §5.5.

Once the user has properly installed hardware, provided the `RoboSpec` GUI with appropriate information, and verified the existence of the necessary files, observations may begin. With a few exceptions, `RoboSpec` then follows the normal observing procedure used by human observers and described in §3.2. A discussion of the necessary changes follows.

4.5 Changes to Observing Procedure

Through the beginning-of-night calibrations, `RoboSpec` behaves much the same as a human observer. The first and most frequent departure from manual procedure occurs during target acquisition. Rather than acquire a guide camera image after slewing to a target, `RoboSpec` instead acquires an East Port image.

Target Acquisition Thanks to the East Port camera's significantly larger field-of-view, there is little concern that its image will not contain the target. And because this method has provided reliable target acquisition, there was no reason for it to be a contingent behavior rather than the default one. Time not spent acquiring and analyzing the guide camera image is an additional benefit.

The GAM position required for acquisition of East Port images is the same as that required for comparison spectra using the arc lamp, a calibration performed before and after observing each target. A human observer slews, takes guide camera images to center the star on the slit manually, then moves the GAM mirror to acquire a comparison spectrum, finally moving the GAM back to the position required by the guide camera. Since `RoboSpec` instead uses the East Port camera to place the star near the slit, it acquires a comparison spectrum before moving the GAM mirror back to the guide camera position, thereby reducing the number of required GAM movements.

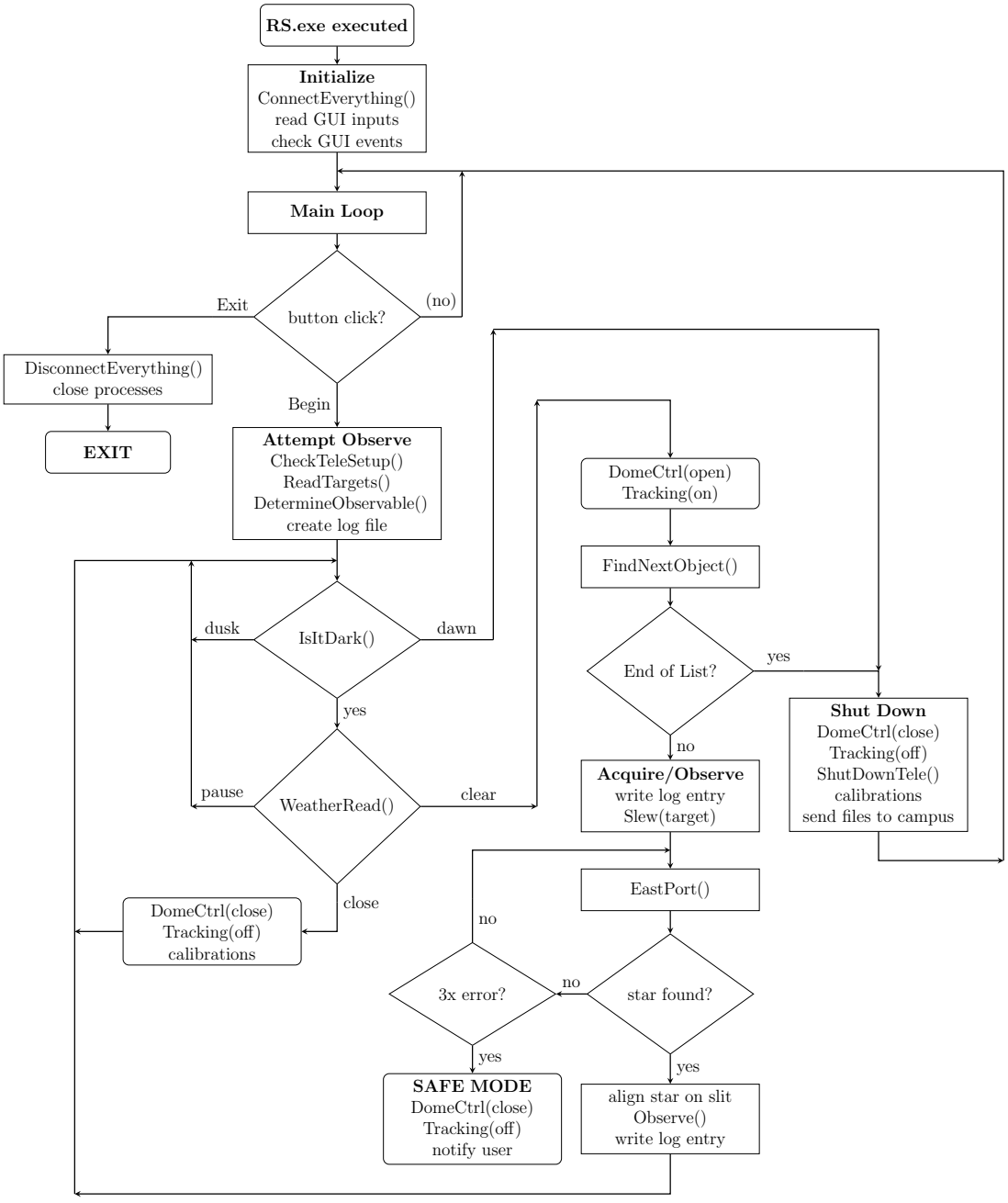
Inclement Weather Delay Should `RoboSpec` close the observatory dome due to unfavorable weather conditions, it will remain closed until conditions become clear and remain that way for a quarter of an hour; that logic is programmed into `RWeather`. In the same situation, a human observer could monitor satellite images and nearby Doppler radar, and perhaps determine after only five minutes that conditions had become suitable to resume observing. Thus the human observer could have a ten-minute advantage over `RoboSpec`. Nonetheless, observatory safety takes precedence over a few spectra, and `RoboSpec`'s efficiency exceeds that of human observers by more than a few cautionary periods.

Nightly Observing Logs Like human observers, `RoboSpec` writes its own nightly log files, containing information about the targets observed, the time, number, and duration of exposures, weather conditions, and hardware or software problems. Writing a few lines to the log takes `RoboSpec` no longer than a second. Thus the automated observing logs can contain significantly more detail than do

manual logs, while incurring effectively no extra overhead. However, a disadvantage is that `RoboSpec` is much less adept at identifying anomalous observations than are human observers. Should the user discover an image that is clearly invalid, `RoboSpec`'s nightly log might still require some scrutiny to identify the likely cause of the problem.

Aside from these differences, `RoboSpec` carries out spectroscopic observations as a human would do with the same equipment. Rather than attempt to describe the intricacies of its behavior in words, we opt to use a flow chart. The following section contains a visual descriptions of `RoboSpec`'s basic operation. The entirety of `RoboSpec`'s AutoIt code is available from a link at the URL `http://www.appstate.edu/~grayro/Robotic/`. Also available from that webpage is the AutoIt code for `TCP_Camera3`.

4.6 Logic Flow Chart - Top-Level Operation



Chapter 5

RoboticWeatherman

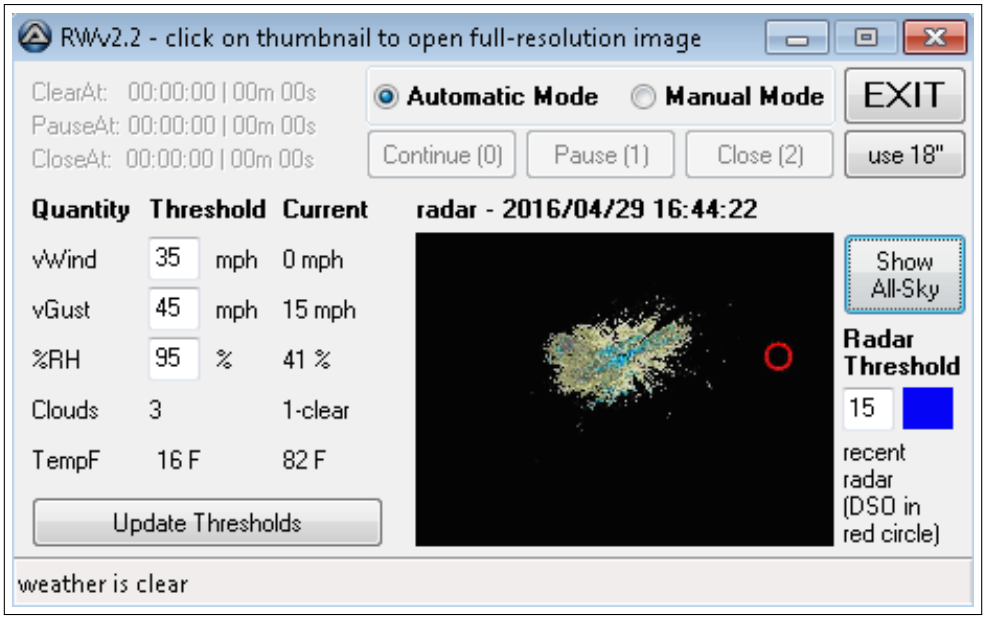


Figure 5.1: Graphical user interface for `RoboticWeatherman`, showing the Doppler radar image downloaded from National Weather Service.

5.1 Motivation and Basic Functionality

A substantial and vital undertaking when automating the observatory is protecting it from inclement weather. To accomplish this, we have written an AutoIt program called `RoboticWeatherman` (`RWeather`), which runs independently of `RoboticSpectroscopist`. `RWeather` is intended to handle all interpretation of weather information, and provide `RoboSpec` with a single integer indicating whether it may continue observing, should pause and wait for conditions to improve, or must close immediately.

`RWeather` makes use of several pieces of weather-monitoring hardware already installed at the Dark Sky Observatory, as well as a Doppler Radar image downloadable from the National Weather Service website. `RoboSpec` maintains actual control of the telescope mount, dome, and mirror and dome shutters, as well as the observing procedure. However, it parses and responds immediately to a simple text file created by `RWeather`, weather.txt. `RWeather` also includes a manual mode of operation, which an observer may use to override the program's output. `RWeather` switches to manual mode automatically in the case that all sources of critical weather information are unavailable.

5.2 Hardware

Two ClarityII (CII) weather-monitoring sensor arrays are already installed at the Dark Sky Observatory; one is located at the 32-inch telescope, the other at the 18-inch telescope. By default, `RWeather` uses the 32-inch sensor, as its proximity

to the telescope makes it the preferred device. If the 32-inch sensor is unavailable, `RWeather` will attempt to switch to the 18-inch sensor. If neither is available, `RWeather` sets an output flag to close the dome and notify the user via text message.

The CII sensors create and append text files, which contain information describing temperature, wind, rain, irradiance, etc., and are parsed by `RWeather`. The quantity of interest is an integer indicating whether the sky is clear, partly cloudy, overcast, or rainy. Both CII instruments update their output file with new readings every 2.1 seconds, writing several additional header lines every ten lines of data, and sensor status information every 50 lines of data.

Also installed at DSO is a separate weather station we call WXData, after its output file wxdata.txt. From this file `RWeather` obtains measurements of ambient temperature, relative humidity, wind speed, and gust speed. Despite the availability of these quantities from the CII output files, experience suggests that the values reported by WXData are less prone to erroneous measurements. In the case that wxdata.txt is unavailable, `RWeather` sets an output flag notifying `RoboSpec` to continue normal operation. Information provided by WXData is considered non-critical to `RoboSpec`'s operation, but the user will be informed of its absence. The WXData output file is overwritten with new data roughly every two minutes.

5.3 Operation

`RWeather` is executed automatically when an observer first runs the `RoboSpec` executable file. We opted to create a separate script to monitor weather, rather than add the pertinent functionality to `RoboSpec`. The decision was motivated by

the need to have `RWeather` monitor weather conditions continually. Tasks such as acquiring a spectrum causes `RoboSpec` to 'hang up' for as long as five minutes. The AutoIt language is not designed to support multi-thread programs, whereby two independent processes may be operate in parallel from a single script. However, Microsoft Windows handles threaded procedures with ease. Creating two separate GUIs allows `RWeather` to monitor weather every 2.1 seconds, while `RoboSpec` concurrently acquires a 300-second spectrum; the parallel design of the Windows operating system takes care of running the two programs simultaneously.

When `RWeather` terminates properly, it deletes its weather.txt output file, indicating to `RoboSpec` that it is no longer running. However should `RWeather` crash, it may not execute the code required to delete weather.txt. To prevent such an incident resulting in damage to the observatory, we added a heartbeat digit to `RWeather`'s output. By incrementing every time `RWeather` analyzes weather data (roughly every 3.14 seconds), the heartbeat digit indicates to `RoboSpec` that `RWeather` has not crashed, and that the information contained in weather.txt is accurate as of several seconds prior.

Should `RWeather` crash, `RoboSpec` will read the same heartbeat digit from weather.txt for more than two consecutive 3.14 second periods. Thus it will know that weather information is unavailable, and must close the dome and notify the user. `RWeather` may simply be restarted manually, allowing `RoboSpec` to carry on observing regularly.

5.4 Control

`RWeather` defaults to its automatic mode of operation any time it is executed. It immediately attempts to locate the data file from the 32-inch ClarityII sensor. If unsuccessful, it attempts to switch to the 18-inch data file. Failing this, `RWeather` sets an output flag indicating to `RoboSpec` that the dome should be closed and the assigned user notified of the issue.

The user will find `RWeather` in manual mode, and if the weather is clear, may then use the recently-enabled buttons to write control values to weather.txt manually. These values are 0 (weather is clear; continue observing), 1 (weather is partly cloudy; pause observing), and 3 (weather is overcast or worse; close dome immediately). The observer may also attempt to reinitialize automatic mode at any time, although `RWeather` could once again fail to locate its required data files, and subsequently revert to manual operation.

In addition to `RWeather`'s built-in ability to switch ClarityII sources, the GUI includes a button that an observer may use to force such a switch. If one sensor is online but obviously malfunctioning, for example, switching to the other would be the quickest and easiest solution to attempt.

While operating in automatic mode, `RWeather` uses certain thresholds to identify weather conditions that are safe for observing. These maximum acceptable values for relative humidity, wind speed, gust speed, and radar base reflectivity can be modified by the observer, although all quantities default to reasonable values. Each value is adjusted individually, then all changes applied simultaneously by

the 'Update Thresholds' button. Each new value is verified before being applied; for example, the relative humidity threshold must be a number between zero and 100. Inputs that violate their respective criteria are coerced to the nearest legal value, e.g., a relative humidity limit of 103 would be set to 100.

A significant portion of `RWeather`'s GUI is occupied by one of two images. The default image is a screen capture from a wide-field all-sky infrared CCD camera at the 32-inch telescope. `RWeather` downloads this image automatically as part of its initialization procedure, and then replaces it every six minutes, as frequently as the online image is updated. All-sky images are not processed at all by `RWeather` to help determine weather conditions, but should be sufficient for a user to differentiate between clear and overcast skies at a glance.

The second available image is the National Weather Service (NWS) Doppler radar image, centered on the Knoxville/Tri-cities TN radar site. It provides information about the amount of precipitation, measured as base reflectivity, per pixel[21]. Each pixel corresponds roughly to one square kilometer[22]. The image displayed on the GUI is of too low resolution to convey detailed information to the observer, but can provide an estimate of the amount of nearby precipitation. Figure 5.1 displays a Doppler radar image rather than an image from the all-sky camera.

[21]National Oceanic and Atmospheric Administration, National Weather Service, 'RIDGE Radar Frequently Asked Questions,' `http://www.srh.noaa.gov/jetstream/doppler/radarfaq.html`

[22]National Oceanic and Atmospheric Administration, National Weather Service, 'Zooming and Panning,' `http://www.srh.noaa.gov/jetstream/doppler/ridge.html#range`

The full-size downloaded radar image, unlike the all-sky image, is used by `RWeather` to determine weather conditions, and is replaced every ten minutes. The displayed image also has a superimposed red circle, indicating the area that `RWeather` searches for pixels containing significant precipitation. A button on the GUI switches freely between the webcam and radar images. In addition, the full-size versions of both images can be opened simply by clicking on the thumbnail image displayed on `RWeather`.

After downloading a new Doppler radar image, `RWeather` analyzes each pixel in the red circle. The circle itself corresponds to a 15-kilometer radius around DSO. Another image available from the NWS is the color key for radar images. That key was analyzed, and hexadecimal values for each color extracted. Those values were then hard-coded into `RWeather`; the program will encounter some difficulty should NWS ever change the key colors. For reference, Figure 5.1 has a threshold value of 15, corresponding to the blue color displayed in the swatch next to the input box. That particular blue has a hexadecimal RGB value of 0300F4.

Before scanning the radar pixels within 15km of DSO, `RWeather` reads the Radar Threshold input box on the GUI. It then scans each pixel in range, comparing its color to those at and above the value specified as the Radar Threshold. If any pixel in range is found to have a color matching a value of base reflectivity higher than the Radar Threshold, `RWeather` immediately instructs `RoboSpec` to close the observatory dome. This behavior protects the telescope and instruments from fast-moving weather fronts, which may not appear on the DSO weather sensors in time to close the dome.

Other thresholds used in RWeather's logic are user-definable; sustained wind speed, gust speed, and relative humidity can all be adjusted using the vWind, vGust, and %RH input boxes, respectively. The default values are those displayed in Figure 5.1. None of the threshold values, including the Radar Threshold, are updated within RWeather's logic until the user clicks the Update Thresholds button.

Each threshold indicates what value of the corresponding measured quantity causes RWeather to issue an immediate 'close' command to RoboSpec. In all cases except temperature Fahrenheit, threshold values indicate an upper limit; TempF is a lower limit, since the dome motors stop functioning properly at low temperatures. The thresholds for Clouds and TempF are hard-coded into RWeather, and cannot be modified by the user.

5.5 Communication

Communication between RWeather and RoboSpec occurs through the weather.txt text file. It currently contains five digits, each conveying a different piece of information to RoboSpec. They are described in this section.

The first digit indicates whether the ambient temperature is above (0) or below (1) 16 degrees Fahrenheit. The motors controlling the dome, and more importantly the shutters that close the dome, begin malfunctioning around 14°F. To protect the telescope and instruments, RWeather instructs RoboSpec to close the dome immediately should the temperature reach 16°F.

The data from WXData are not vital to `RoboSpec`'s operation, although the additional information provides greater protection for the observatory equipment. If `RWeather` cannot locate wxdata.txt, for example due to a network outage, it sets the second digit in weather.txt to 1; otherwise that digit is 0. Should the user notice that wxdata.txt is absent during questionable weather conditions, he or she may opt to shut down `RoboSpec` to prevent any damage to the observatory.

The third digit indicates the action `RoboSpec` should take, in light of all available weather information. A value of 0 indicates that `RoboSpec` should continue observations, 1 indicates that it should pause and wait for conditions to improve, and 2 indicates that it should close the dome immediately. Additional error values are 3, indicating that vital weather information is absent, and 4, indicating that the user has switched `RWeather` to manual mode but has not yet chosen an output value. `RoboSpec` enters safe mode if it receives one of the error values. For more detail about the logic determining this digit, see the flow charts in §5.5. In addition, any action requiring the dome be closed immediately, such as temperature below 16°F, will force this digit to 2.

The fourth digit relates to images displayed on the `RWeather` GUI. The program attempts to download and display the latest all-sky infrared camera image from DSO, as well as a nearby Doppler radar image. Neither is vital to `RoboSpec`'s operation, but both are useful to a user scrutinizing `RWeather`'s behavior. A value of 0 for this digit indicates that both images were properly located and downloaded. A value of 1 indicates a missing all-sky image, a value of 2 indicates a missing Doppler radar image, and a value of 3 indicates that both images are absent. Since

RWeather makes user of the radar data to detect incoming weather, its absence may be of concern. However, RWeather will not alter its instruction to RoboSpec due solely to lack of radar data.

The final digit in weather.txt is RWeather's 'heartbeat.' When operating properly, weather.txt is updated roughly every three seconds, and this digit changes with each update. It counts up from 0 to 9, then repeats. When the RWeather script terminates, it normally destroys weather.txt, which signals to RoboSpec that weather information is not available. A concern, however, is that RWeather could crash and somehow terminate without destroying weather.txt. RoboSpec would continue reading the same value from the file, never recognizing a problem. To prevent such a situation, RoboSpec has been adapted to compare its most recent reading from weather.txt to the previous reading. If the heartbeat digit does not change over the course of roughly seven seconds, RWeather may have crashed, and RoboSpec should enter safe mode.

5.6 Notes on Logic

As with RoboSpec, we have opted to describe specifics of RWeather's operation using flow charts, due to the complexity of some of the logic. A few points warrant explicit explanation, however.

Delayed Action RWeather does not respond immediately to *improving* weather conditions, regarding its observing instruction to RoboSpec. Similar behavior is used at the STELLA Observatory; Granzer *et al.* (2010) refer to it as a 'retard time' after a weather event.

Weather conditions at DSO tend to fluctuate unpredictably. Patchy clouds are common, and rain showers may be punctuated by brief periods of clear sky. We want `RoboSpec` to avoid any attempt to resume observing if favorable weather will only last a few minutes, so we require that conditions remain clear for a fifteen minutes before `RWeather` change its instruction. Rain on the radar image immediately causes `RWeather` to change its output flag to 2 (close). If the blip happens to be a flock of birds migrating, for example, it may disperse quickly, and the weather will again be clear. `RWeather` will start its ClearAt timer, and change its output back to 0 (clear) only when the timer reaches fifteen minutes.

Conversely, if a single cloud happens to pass over the ClarityII sensor, we want to prevent `RoboSpec` ceasing operation entirely. Thus partly-cloudy weather causes `RWeather` to change its output to 1 (pause) and start its PauseAt timer. If `RWeather` remains paused for fifteen minutes due to persistent patchy clouds, it then changes its output to 2 (close). However, if conditions clear during a pause, and remain that way for fifteen minutes as indicated by the ClearAt timer, `RWeather` will instead change its output to 0 (clear). Thus the overhead associated with closing the dome and disabling tracking is avoided, by preventing a dome close being triggered by a rogue cloud. This and `RWeather`'s other governing logic are illustrated in the following section.

Oscillating Conditions An additional concern is that patchy clouds may cause the CII sensor array output to oscillate between clear and partly cloudy. Without detecting that unusual case, `RWeather` would sit in the 'pause' case indefinitely. It would alternate between the PauseAt and ClearAt timers, with neither ever

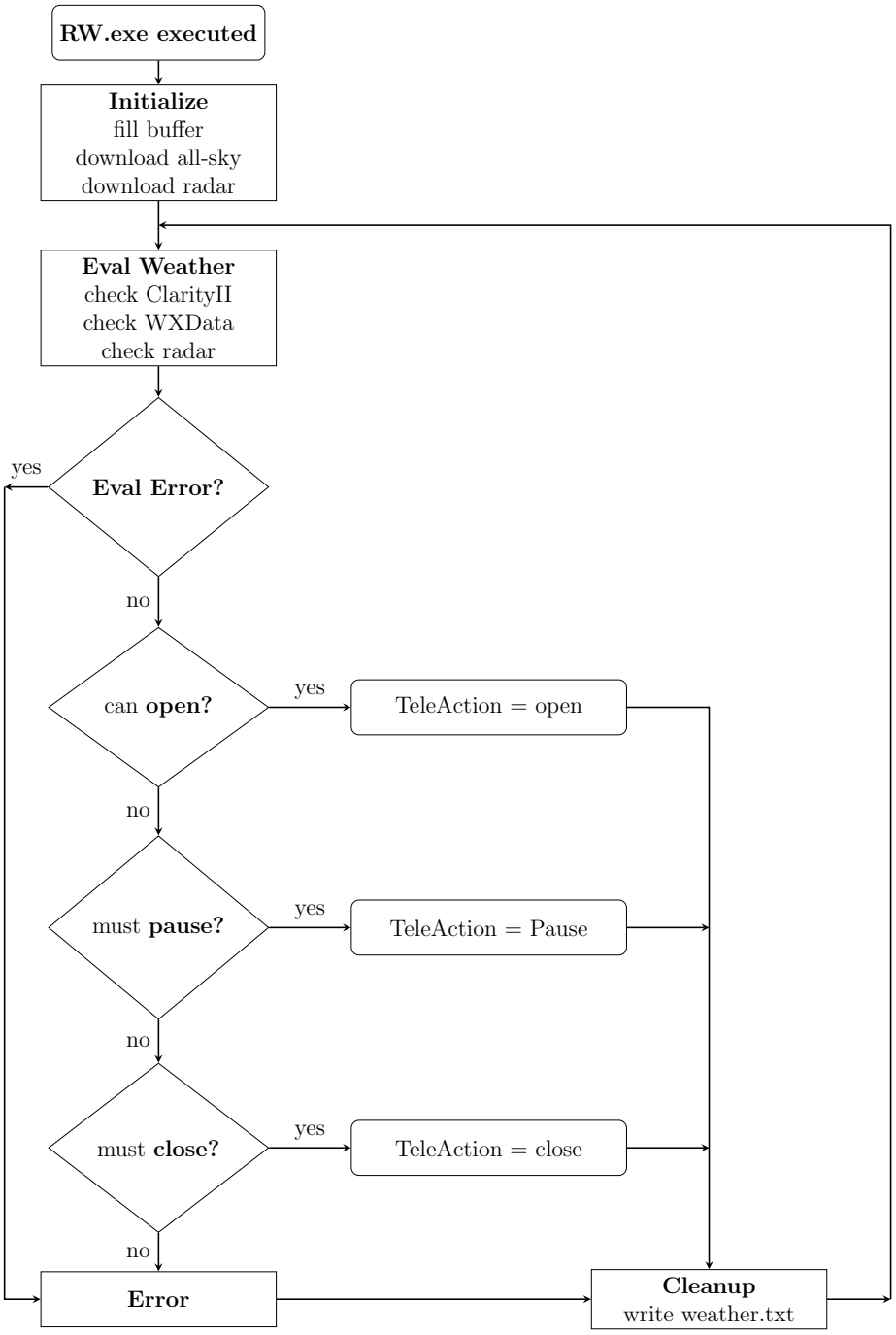
reaching the fifteen minutes required to close or resume. Pausing observations does not disable telescope tracking, and could damage the observatory by driving the telescope to its physical limits.

To prevent such a situation, we have introduced a PauseOutputStartedAt timer. Any time `RWeather` instructs `RoboSpec` to pause observations, the PauseOutputStartedAt timer starts running. It is stopped and reset if `RWeather`'s instruction to `RoboSpec` changes to something other than pause. However if the timer runs for half an hour, it sets a flag that forces `RWeather` into the 'close' state. The longer time limit gives patchy clouds time to clear, should the weather event be short-lived. This logic is handled in `RWeather`'s Cleanup function, shown in §5.6.1, but is not shown in detail.

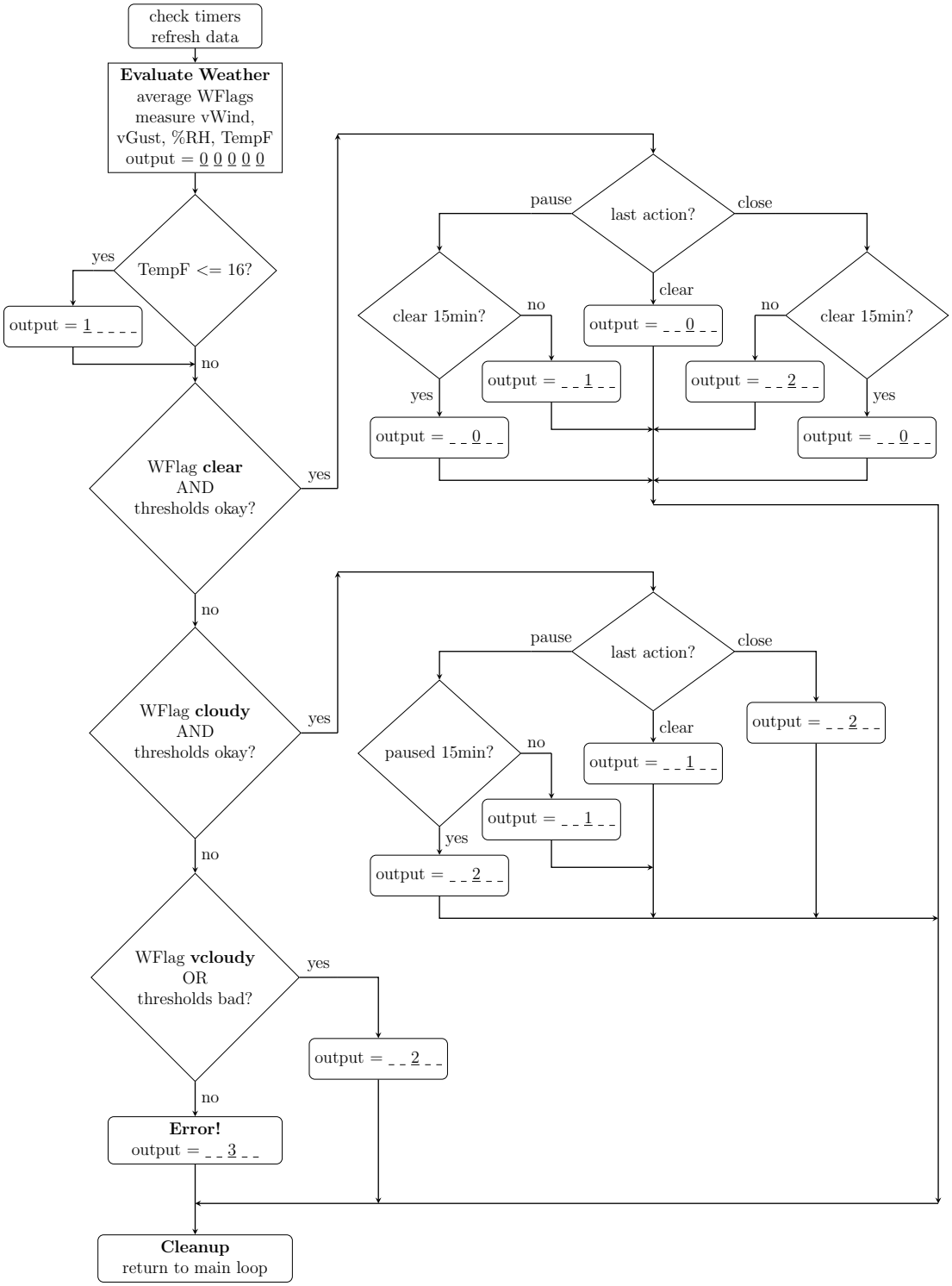
The following section contains flow charts describing `RWeather`'s general behavior and weather analysis logic. The entirety of `RWeather`'s AutoIt code is available from a link at the URL `http://www.appstate.edu/~grayro/Robotic/`.

5.7 Logic Flow Charts

5.7.1 Top-Level Operation



5.7.2 Evaluate Weather Conditions



Chapter 6

Conclusion

6.1 Efficiency

ASU's astronomical spectroscopists began using `RoboSpec` extensively in January 2016. Between then and the end of March 2016, `RoboSpec`'s record for spectra acquired during a single night of observing was 131. For comparison, a human observer's record between January and March of the previous year, when `RoboSpec` was not used, was 124 spectra. However, those spectra were obtained from a single object. That type of observing incurs almost no overhead whatsoever, since it is never necessary to slew to and acquire a new target. When following the normal observing procedure of acquiring several spectra from one target then slewing to another, the record for a human observer was 119 spectra in a night[23].

Thus `RoboSpec` showed an improvement of roughly 10% in absolute maximum number of spectra acquired in a single night, compared to human observers, when

[23]Gray, R.O. 2016, Private communication

using a similar observing procedure. During those three-month periods, the average number of spectra acquired per night was 69 for human observers and 78 for `RoboSpec`, not counting nights with zero spectra. This was an improvement of roughly 13% for `RoboSpec`[23].

Unfortunately, weather conditions play a significant role in observing efficiency, and can vary substantially from year to year. Notes from observing logs indicate that 2015 was relatively clear between January and March, whereas the same period in 2016 was comparatively cloudy. Thus the above statistics could be skewed in favor of human observers. Before a definite conclusion can be made about `RoboSpec`'s efficiency, more data are required. A full year of automated observing might be sufficient, but two or three years would hopefully mitigate the effects of long-term weather variation[23].

Even without extensive statistics describing observing efficiency, `RoboSpec` appears to have a non-anomalous advantage over human observers. And in addition to a slight improvement in raw volume of data obtained, automated observations are performed while the assigned observer sleeps. Dr. Richard Gray describes the longest nights of manual observing as "being Herculean tasks"[23]. Delegating that responsibility to `RoboSpec` is a welcome change, even were automated observing only equally as productive as manual observing.

6.2 Future Additions

Auto Start As mentioned in §4.4, `RoboSpec`'s Auto Start feature has not yet been implemented. It will allow users to set `RoboSpec` running in the afternoon,

and pay it little heed until the next morning. This will be particularly useful for busy faculty with afternoon and evening obligations. Thus automation will prevent research from being neglected, despite increasingly demanding academic schedules.

Queue-Based Observing Appalachian State University's astronomical spectroscopists benefit from nearly identical instrument configurations. Thus their research projects are well-suited for queue-based observing, requiring few, if any, physical modifications whatsoever to the observatory setup. This would increase `RoboSpec`'s efficiency further, allowing targets from separate projects to be observed as close as possible to their preferred time.

Plate Solving Plate solving will also be added, allowing users to observe stars that are not the brightest in the nearby field. The target acquisition procedure discussed in §4.1 will require fundamental modification, since a simple brightest-pixel approach will no longer suffice. However, plate solving will allow studies of densely-populated regions of targets, such as star clusters, where a comprehensive survey including every star is desirable.

Guest FTP The new tabbed GUI we designed for `RoboSpec` accounts for the possibility of guest observers. As mentioned in §4.3, regular users receive archived and compressed data files automatically via FTP after `RoboSpec` completes observing. However, that feature is not currently available to guest observers. We hope to add it eventually, allowing visiting astronomical spectroscopists to conduct research using the 32-inch telescope without ever having to visit the observatory.

Automatic Data Reduction The only drawback to the automated file transfer routine is that the data are not reduced. The user is still responsible for applying calibrations, extracting spectra, etc. manually. An automatic data reduction pipeline is currently under development, and will provide users with analysis-ready data before noon the day after observations. Should any particularly interesting phenomena appear, the user may update the next night's target list to revisit those objects. In addition, data reduction can be tedious; an astronomer's time can be much better spent.

6.3 Concluding Remarks

Just as remote access drastically changed observing at the DSO, "automated observing has completely revolutionized spectroscopy at the Dark Sky Observatory"[24]. Not only are faculty astronomers able to obtain over one hundred spectra on clear nights, all with proper calibrations, they may keep regular sleep schedules. And upon arriving at their office the next day, they will find their data automatically downloaded and available by mid-morning. `RoboticSpectroscopist`, with help from `RoboticWeatherman` and various other scripts, conducts automated spectroscopic observations adeptly and reliably.

Through this automation project, we achieved our goal of making observational astronomical spectroscopy more compatible with the academic obligations and schedules of university faculty. We also realized the improvements to observing efficiency and data quality proposed by Eaton (1995) near the outset of robotic

[24]Gray 2016, Private Communication

spectroscopy. `RoboSpec` will continue to improve, no doubt, but is already a remarkable and useful addition to the Dark Sky Observatory.

Bibliography

Blecha, A., Weber, L., Queloz, D., Mayor, M, & Udry, S. 2001, *Astronomische Nachrichten*, 322, 5/6, 317

Bopp, B.W. 1986, *International Amateur-Professional Photoelectric Photometry*, 25, 127B

Boyd, L.J., Epand, D., Bresina, J., Drummond, M., Swanson, K., Crawford, D.L., Genet, D.R., Genet, R.M., Henry, G.W., McCook, G.P., Neely, W., Schmidtke, P., Smith, D., Trueblood, M. 1993, *International Amateur-Professional Photoelectric Photometry*, 52, 23B

Brown, T.M., Baliber, N., Bianco, F.B., Bowman, M., Burleson, B., Conway, P., Crellin, M., Depagne, É., De Vera, J., Dilday, B., Dragomir, D., Dubberley, M., Eastman, J.D., Elphick, M., Falarski, M., Foale, S., Ford, M., Fulton, B.J., Garza, J., Gomez, E.L., Graham, M., Greene, R., Haldeman, B., Hawkins, E., Haworth, B., Haynes, R., Hidas, M., Hjelstrom, A.E., Howell, D.A., Hygelund, J., Lister, T.A., Lobdill, R., Martinez, J., Mullins, D.S., Norbury, M., Parrent, J., Paulson, R., Petry, D.L., Pickles, A., Posner, V., Rosing, W.E., Ross, R., Sand, D.J., Saunders, E.S., Shobbrook, J., Shporer, A., Street, R.A., Thomas, D., Tsapras, Y., Tufts, J.R., Valenti, S., Vander Horst, K., Walker, Z., White, G., & Willis, M. 2013, *arXiv:1305.2437v2*

Copperwheat, C.M., Steele, I.A., Barnsley, R.M., Bates, S.D., Bersier, D., Bode, M.F., Carter, D., Clay, N.R., Collins, C.A., Darnley, M.J., Davis, C.J., Gutierrez, C.M., Harman, D.J., James, P.A., Knapen, J.H., Kobayashi, S., Marchant, J.M., Mazzali, P.A., Mottram, C.J., Mundell, C.G., Newsam, A., Oscoz, A., Palle, E., Piascik, A., Rebolo, R., & Smith, R.J. 2015, *Experimental Astronomy*, 39, 119

DFM Engineering 2004, WinTCS Operations Manual for DFM 32 Inch Telescope and Accessories

Eaton, J.A. 1995, *Astronomical Society of the Pacific Conference Series*, 79, 226E

Eaton, J.A., & Williamson M.H. 2004, *Astronomische Nachrichten*, 325, 6-8, 522

Granzer, T., Weber, M., & Strassmeier, K.G. 2010, *Astronomy & Astrophysics*, 2010, 980182

Kozłowski, S.K., Konacki, M., Ratajczak, M., Sybilski, P., Pawłaszek, R.K., & Hełminiak, K.G. 2014, *Monthly Notices of the Royal Astronomical Society*, 443, 158

Ramsey, L.W. 1992, Robotic Telescopes with Fiber-Coupled Spectrographs, *Astronomical Society of the Pacific Conference Series*, 34, 227R

Rye, D. 2015, X-10 (USA), Inc., The X-10 POWERHOUSE Power Line Interface Model # PL513 and Two-Way Power Line Interface Model # TW523, Technical Note, Revision 2.4

Schmitt, J.H.M.M., Schröder, K.-P., Rauw, G., Hempelmann, A., Mittag, M., González-Pérez, J.N., Czesla, S., Wolter, U., Jack, D., Eenens, P., & Trinidad, M.A. 2014, *Astronomische Nachrichten*, 335, 8, 787

Weber, M., Strassmeier, K.G., & Granzer, T. 2012, *Astronomical Society of India Conference Series*, 7, 165

Vita

Daniel Edwin Rosenberg was born in Bradford, P.A. in 1990. He moved with his parents to Boone, N.C. in 1991, and graduated from Watauga High School in June of 2009. He attended the University of North Carolina at Chapel Hill for undergraduate studies, and conducted research as part of the RESOLVE team led by Dr. Sheila Kannappan. In December 2013, he received his Bachelor of Science in Physics with a concentration in Astronomy and Astrophysics.

In August of 2014, Mr. Rosenberg began pursuing a Master of Science in Engineering Physics at Appalachian State University, with a concentration in Laboratory Automation. He began working with Dr. Richard Gray in January of 2015 to develop a method of robotic astronomical spectroscopy. He benefitted from a National Science Foundation grant awarded to Dr. Gray, in addition to research funding from Appalachian State University's Research Council and from North Carolina Space Grant. He received additional funding for general academic expenses from the James C. Greene Fellowship.

Mr. Rosenberg began work at CITI-LLC, a private systems integration company headquartered in Charlotte, N.C, and received his Master of Science in August 2016.