DEVELOPMENT OF FERGUSSON COLLEGE OBSERVATORY:

TARA Project

Dr. Ms Raka Dabha & Manish Sharad Hiray

Dept. of Physics,

Fergusson College, Pune INDIA

Acknowledgement

We all have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

Firstly thanks are due to the IAU ODA and ISRO without whose financial assistance this project would not have been possible.

We are highly indebted to Prof. Subra Ananthkrishnan for his constant support and guidance at every stage of the project. His constant supervision and discussions helped us immensely and gave us new insights in carrying out this project.

We would like to express our gratitude towards Principal Dr. Ravindrasinh G Pardeshi & member of Physics Department as well as Astro Club Fergusson College for their kind co-operation and encouragement which help us in completion of this project.

We would like to express ourspecial gratitude and thanks to Dr. Ranjan Gupta and many IUCAA personnel for giving us such attention and time. We also appreciate the constant support from Prof. Niall Smith and the BCO team from CIT Ireland.

Our thanks and appreciations also go to our colleague in college in developing the project and people who have willingly helped us out with the best of their abilities, It would have been very difficult to complete the project smoothly without their kind support.

Dr. Raka Dabhade

Principal Investigator,

Head Dept. of Physics,

Ferugsson College, Pune,INDIA

Manish Shrad Hiray

Project Asst.

Table of Contents

Overview of the Project

Fergusson College Observatory is another feather in the cap for Fergusson College, Pune. With the support from ISRO and Physics Dept.- Astro Club we were able to complete the project in due time. The Astro Club at Fergusson College is host to a good batch of students every year who are very keen to learn the skills required to be a good astronomer. The ISRO project has taken this interest of students to the next level by giving them a chance to firsthand experience to operate telescope and record observations.

The main objective was to establish a completely functioned observatory at Fergusson College. Since we have already procured Telescope (Celestron 9.25") from other grants before, we concentrated on other aspects of the observatory. As the proposed site is in Fergusson (heart of city), we are surrounded by good amount of light pollution. Hence we have limited our objective to Solar and Stellar studies only. We found differential photometry is the best suitable thing for our surrounding conditions. We can do differential photometry with CCD and Photometer. Hence those were the prime detectors in our Observatory. To set up an observatory data processing lab and Dome was the prior requirement. Hence we listed our requirements priority wise as follows

1. Data Processing Lab
2. Dome /Shed
3. CCD
4. Photometer

As the initial part of the project we were to install Manual Operated Dome but as the part of remote control observatory we had to upgrade our requirement from manual dome to motorized dome. We found that it will take time to install a motorized dome hence as an alternative we proceed roll off shed from college grants. As the part of ISRO project we have achieved all the objectives both planned and updated.

The details of the product, procedure and objectives are discussed in the following chapters.

Introduction

1.1 Background

Fergusson College has many enthusiastic students who want to peruse their career in Astronomy and being a INAD node of IUCAA we feel the need of Observatory which will not only give the first hand experience to our students but also will help them to understand the Instrumentation concepts. Though we have some limitations as our observatory is being established in the city, the light pollution will play major role. Hence we have limited our aim to study two things, differential photometry of stars and solar study.

Differential Photometry:

Differential photometry is the simplest of the calibrations and most useful for time series observations. When using CCD photometry, both the target and comparison objects are observed at the same time, with the same filters, using the same instrument, and viewed through the same optical path. Most of the observational variables drop out and the differential magnitude is simply the difference between the instrument magnitude of the target object and the comparison object (ΔMag = C Mag – T Mag). This is very useful when plotting the change in magnitude over time of a target object, and is usually compiled into a light curve.

Automation:

As the part of automated observatory we also have to work on aspect like controlling of telescope remotely through internet. We have spent time to test different techniques to control the telescope which have been discussed separately in other chapters. Integration of Telescope and Mount was another challenge. Following are the tools which were tested and use for the automation

1. RTS2
2. ACP
3. Maxim DL

4. DS9
5. Stallarium

These tools are discussed separately in Automation chapter.

1.2 Aim and Objectives

Aim

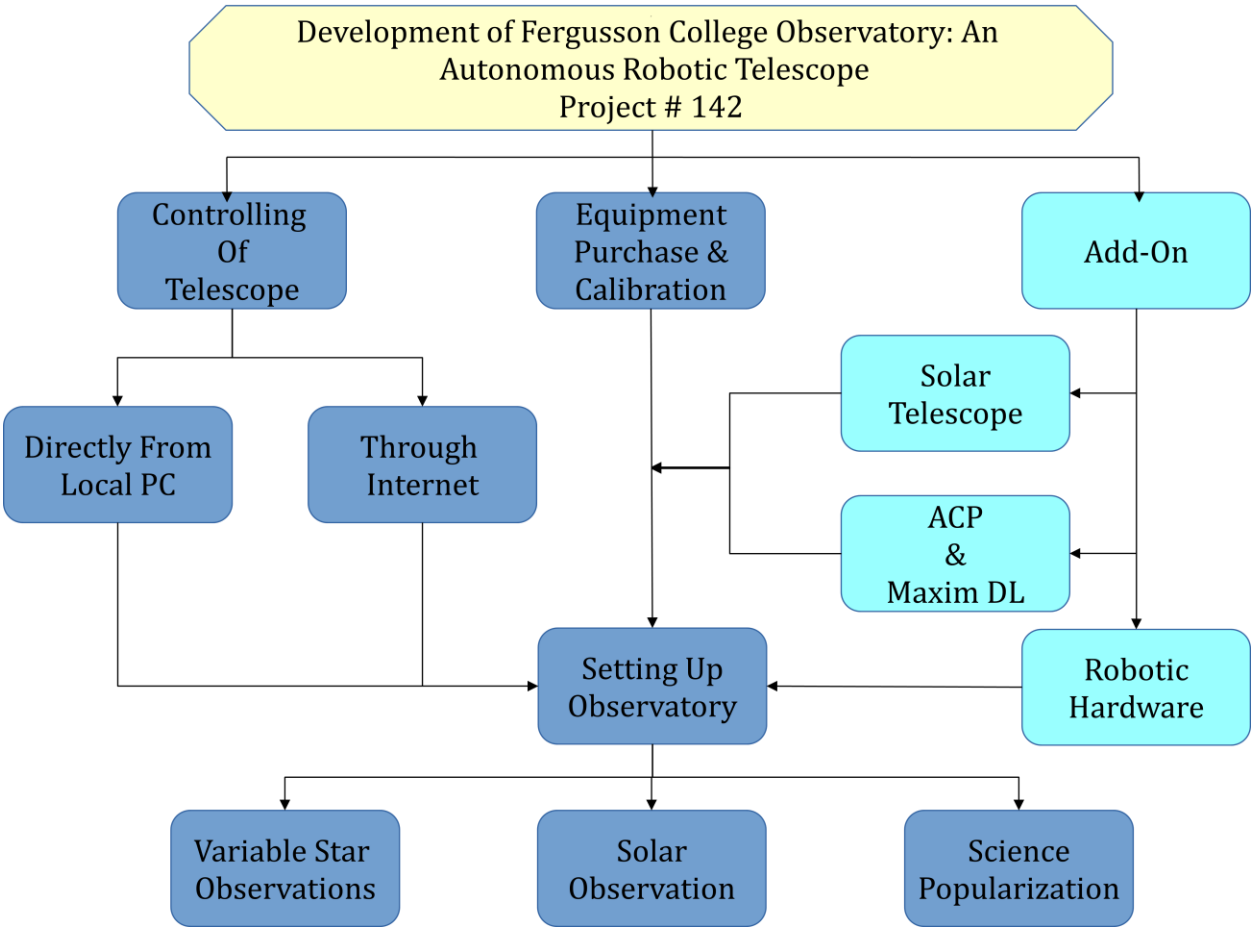
"To develop and establish a completely automated observatory which can be controlled through internet by authorized person." TARA Observatory.

Objectives

1. To identify site for the observatory
   i) To carry the site survey
   ii) Testing different aspect for the proposed site.
2. To procure the required instruments/ equipment for the observatory
   i) Understanding the research finding
   ii) Selection of instruments accordingly
3. To develop the data analysis lab
4. To erect the Motorized dome
5. To set up the Observatory
6. Automation
   i) Telescope control using different tools
   ii) Installation of necessary software
   iii) Installation of necessary hardware
7. To carry the Observing Sessions
   i) To test the current observatory set up
   ii) To improve the observation techniques
8. To Analyze the observed Data

Project Planning

2.1 Block Diagram



2.1.1Equipment Purchase and Calibration:

As mentioned earlier we have limitations due the surroundings hence we have choose the differential photometry technique. Differential photometry can be done in two ways a) CCD Photometry b) Traditional method using photometer. Selecting a photometer and CCD was a crucial task. With help from IUCAA we had selected a photometer and CCD to fulfill our requirements. The details of both detectors are discussed in separate chapter.

2.1.2Controlling of Telescope

Controlling of the telescope was the one of the most important aspect of entire project. Hence we spent considerable time to achieve this objective. We had gone to different software and hardware interfaces. After considerable literature survey on Telescope controlling we came to know about few programs like RTS2 which is a Remote Telescope System developed for linux system. We also come to know about some other software like ACP which is developed specially for the windows operating system. We have tested both of the software over period of time and had made conclusion which has to be installed for the observatory. The selection and reasoning is discussed in details in Automation chapter.

2.1.3Add- On

When we started this project our aim was only to observe night sky. But as we have blessed with ample sun light we decided to go for solar observations too. The first need was to purchase solar telescope. We managed to procure grants from DBT to purchase solar telescope.

While testing telescope controlling through internet we came to know that it was very difficult for student to operate the telescope using RTS2 since it runs on linux program and one has to have considerable knowledge of command line programming and operating. Hence we decided to go with some simple solution. As most of the students are well versed with Windows operating system we decided to go for ACP and Maxim DL for the automation purpose. Again we managed to procure grants from International Astronomical Union to purchase these software.

As our title of the project suggests its robotic telescope hence we wanted to make it completely robotic and minimize the human interference as less as possible. This was not possible as our current project proposal. Most important drawback was manually operated dome. Hence we decided to move on for the motorized dome and other hardware like robofocus, whether station, cloud sensor etc to make the observatory completely automated. The details are discussed in dome chapter

2.2 Time Schedule

2.2.1Pert Diagram for Project

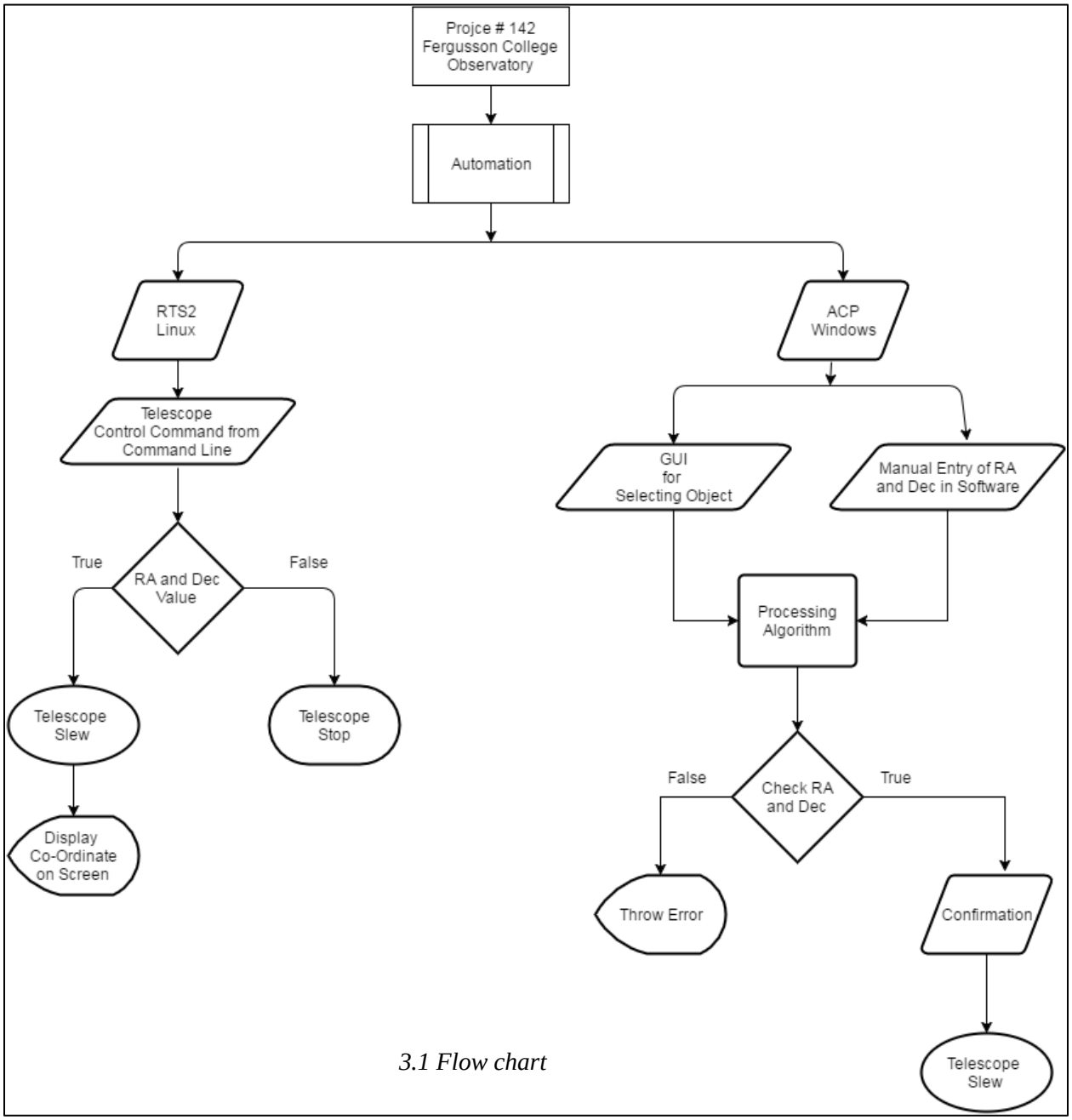
Month / Task	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	7	28	29	30
Literature Survey																														
Initial Investigation																														
Site Surve																														
List of Requirement																														
Selection of Equipment																														
Purchase of Equipment																														
Automation Part 1																														
Automation Part 2																														
Purchase Report																														
Hardware Installation																														
Purchase																														
Observation																														
Data Analysis																														
Closure Report																														

All the heading like site survey, Automation etc are discussed in details in following chapters.

Project Details

In this chapter we have discussed all the terms which were mentioned partly in previous chapters.

We have discussed the basic idea of the automation by using flowchart mentioned below.



3.1 Flow chart

3.1 Equipment / Instruments

We already had few instruments and we have procured several others from ISRO grant as well as other grants. The major purchases were CCD and Photometer. We had discussed with Dr. Ranjan Gupta about the purchase of CCD, Photometer and their necessary accessories. Then we finalized the purchase order and after the approval from Local Purchase Committee we procured the equipment and accessories.

3.1.1Instruments Owned by the college:

Sr. No	Instrument / Equipment		Make	Source / Grant
	Type	Model		
1.	Telescope	C-9.25”	Celestron	Old ISRO grant
2.	DSLR	Canon 50 D	Canon	Old ISRO
3.	Mount	CG-5	Celestron	Old ISRO

3.1.2 Instruments Procured from ISRO Grants

Sr. No	Instrument / Equipment		Make
	Type	Model	
1.	Refractor Telescope	C-70	Celestron
2.	CCD	ST402 ME	SBIG
3.	T TO C Adapter	SB-51 532	SBIG
4.	Flip Mirror	ORION 1.25"	Orion
5.	Celestron Powerseeker Telescope Accessory Kit	CE-94303	Celestron
6.	8-24 Zoom Eyepiece	CE-93230	Celestron
7.	12.5mm Plossl Illuminated Reticle Eyepiece	OFU-ILLRET	Opticsfuture
8.	Dehumidifier	EV-E-333	EVA

9.	EO Mount	OMNI CG-4	Celestron
10.	Polarscope	CG5A/X	Celestron
11.	Universal Adapter	CE-93626	Celestron
12.	Accessory Hard Case	OR-5999	ORION
13.	Photometer	SSP-3	Optec
14.	2-position filter slider	SSP-3 B 17050	Optec
15.	Johnson B (Blue)	SSP-3 B 17032	Optec
16.	Johnson V (Visual)	SSP-3 B 17033	Optec
17.	Johnson R (Red)	SSP-3 B 17034	Optec
18.	Johnson I (Infrared)	SSP-3 B 17035	Optec
19.	25-ft. serial cable for SSP photometer	SSP-3 B 17157	Optec
20.	USB-to-Serial converter	SSP-3 B 17690	Optec
21.	SSPDataq3 photometer control, data acquisition and data reduction package for UBVRI	SSP-3 B 17030	Optec
22.	T-Rings	C-93419	Celestron
23.	T-Adapter	C-93644	Celestron
24.	Universal Mounting Plate - CG5		Celestron
25.	Dual Axis Motor Drive (Advanced CG4)	C-93522	Celestron
26.	Polar Finder CG-4 (Omni Mount)		Celestron
27.	Binocular	Up-close 10x50	Celestron
28.	Universal Camera Adapter	1.25-Inch Variable	Orion
29.	Flexible Plastic Dew Shield		Astrozap
30.	Sricam SP Series Wireless HD IP Wifi CCTV	SP005Black	Sricam
31.	Guard CPS-816 (12V) 16 Channel Professional Power Supply (SMPS) for CCTV	CPS-816	iBall

3.1.3 Instruments/ Equipment Procured from various other Grants

Sr. No	Instrument / Equipment		Make	Source / Grant
	Type	Model		
1.	Solar Telescope	Lunt B1200 H Alfa	Lunt	DBT
2.	Mount	Paramount ME II	Bisque	MPLADF
3.	WIRELESS OBSERVATORY WEATHER STATION	TT-WEA2	Technical Innovation	MPLADF
4.	Aag2 Cloud Sensor	TT-AAG2	Technical Innovation	MPLADF
5.	ROBOFOCUS RF-3	TT-ROBOF	Technical Innovation	MPLADF
6.	Losmandy Dup Short	LO-DUPS	Losmandy	MPLADF
7.	D/V Dovetail Adapter	LO-DVA	Losmandy	MPLADF
8.	Delkin Sensor Scope System III	DDSS-SCOPE3	Delkin	MPLADF
9.	2.5 Lb Weight	LO-2.5CW	Losmandy	MPLADF
10.	6" Dovetail Bar For Lunt Clamshells	LS-150PS	Lunt	MPLADF
11.	Dovetail Shoe F2	SV-FBD	Stellarvue	MPLADF
12.	TPO Vixen Style Finder Shoe	OS-VFS	Vixen	MPLADF
13.	Motorized Dome	3 M	Scope Dome	MPLADF
14.	Observatory Software	ACP	DC3	**IAU-ODA**
15.	Imaging Software	Maxim DL V-6	Cyanogen Imaging	**IAU-ODA**
16.	Miscellaneous consumables			**IAU-ODA**

We have gathered all necessary equipment, hardware, software and accessories to run the observatory. The software installation part is done. With the set up we have carried out some observational sessions which are discussed in the Observation Chapter.

3.2 Celestron C9.25”

The Celestron C9 1/4 is Celestron’s first new Schmidt-Cassegrain optical system in over a decade. The new 9.25” Schmidt-Cassegrain features a precision optical system with 2350mm

focal length (f/10) and comes standard with Celestron's StarBright XLT coatings. The detailed parameters are given below

3.2.1 Specifications

Sr. No	Parameter	Value
1.	Optical Design	Schmidt-Cassegrain
2.	Aperture (mm)	235 mm (9.25 in)
3.	Telescope Type	Computerized (GOTO)
4.	Focal Length	2350 mm (93 in)
5.	Focal Ratio	10
6.	Focal Length of Eyepiece 1 (mm)	25 mm (0.98 in)
7.	Magnification of Eyepiece 1	94 x
8.	Finderscope	6x30
9.	Tripod	Adjustable, Stainless Steel
10.	Tripod Leg Diameter	2"
11.	Highest Useful Magnification	555 x
12.	Lowest Useful Magnification	34 x
13.	Limiting Stellar Magnitude	14.4
14.	Resolution (Rayleigh)	0.59 arc seconds
15.	Resolution (Dawes)	0.49 arc seconds
16.	Light Gathering Power (Compared to human eye)	1127 x
17.	Optical Tube Length	22" in (559 mm)
18.	Optical Tube Weight	20 lbs (9.07 kg)
19.	Mount Height (Max)	64" in (1626 mm)
20.	Mount Height (Min)	44" in (1118 mm)
21.	Tripod Weight	18 lbs (8.16 kg)
22.	Weight of Counterweights	2 x 12 lbs

3.3 CCD: SBIG ST402 ME

3.3.1 Design

The ST-402ME uses the same Microlensed, Blue Enhanced, Full Frame KAF-0402ME CCD as the ST-7XME camera. The array is 765 x 510 pixels at 9 microns square. The same technology that is used to achieve such high quantum efficiency in the KAF- 3200ME CCD is also used with the same effectiveness in the KAF-0402ME CCD. With a peak QE of nearly 85%, this camera bows its head to no other when it comes to recording faint detail in dim objects. A full frame download takes less than a second with the new USB 2.0 electronics. Focus mode updates the computer screen about 2X per second. The small size and light weight makes this camera very easy to handle and set up. A custom internal filter wheel and shutter lets you take dark frames and tri-color images automatically. Best of all, the low noise and extraordinary QE of the KAF-0402ME CCD makes this one of the most sensitive CCD cameras available to amateurs at any price. Simply put, there is nothing that can touch it in its class, except of course the dual sensor, self-guiding ST-7XME camera.

The specification of the CCD is given below.

3.3.2 Specifications

CCD

CCD:	Kodak KAF-0402ME
Pixel Array:	765 X 510 pixels, 6.9 X 4.6mm
Total Pixels:	390,000
Pixel Size:	9 X 9 microns
Full Well Capacity (NABG):	~100,000 e-
Dark Current:	1e/pixel/sec@0º C
Antiblooming:	Option (KAF-0401LE)

Readout Specifications

Shutter:	Electromechanical
Exposure:	0.04 to 3600 seconds, 10ms resolution
Correlated Double Sampling:	Yes
A/D Converter:	16 bits
A/D Gain:	2.3e-
Read Noise:	15e RMS
Binning Modes:	1 x 1, 2 x 2, 3 x 3
Full Frame Download Rate: On USB 2:	Up to 800,000 pixels per second.
On USB 1:	Up to 400,000 pixels per second
Full Frame Download time:	less than 1 second

Optical Specifications (8" f/10)

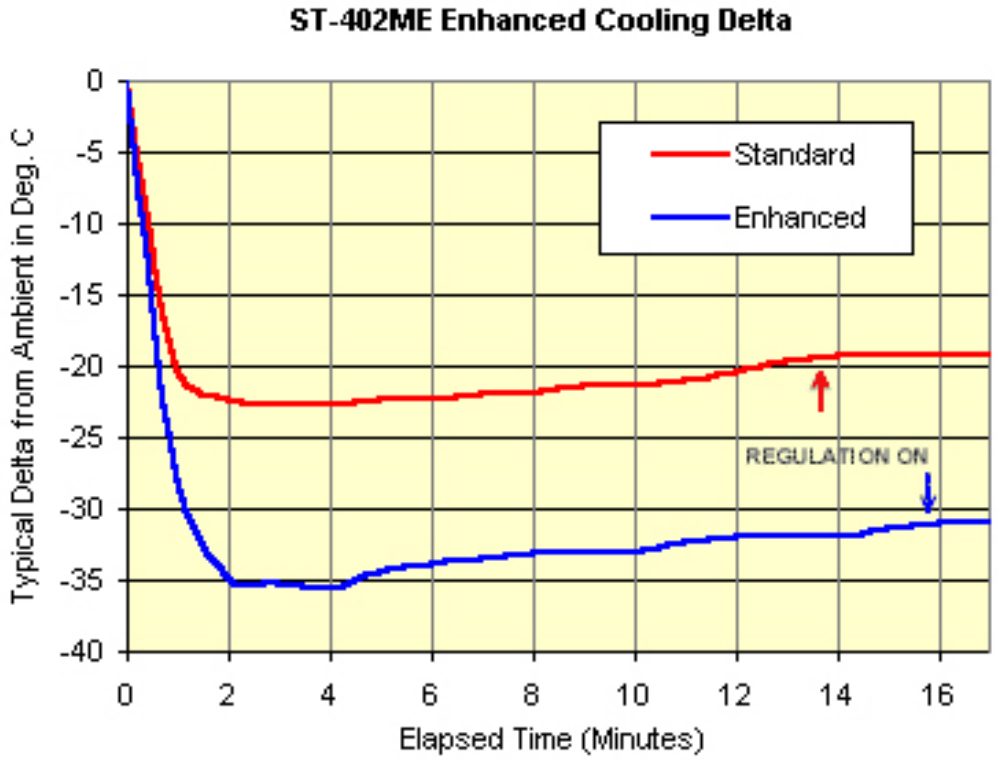
Field of View:	12 X 8 arcminutes
Pixel Size:	0.9 x 0.9 arcseconds
Limiting Magnitude:	Magnitude 14 in 1 second (for 3 arcsec FWHM stars, magnitude 18 in 1 minute)

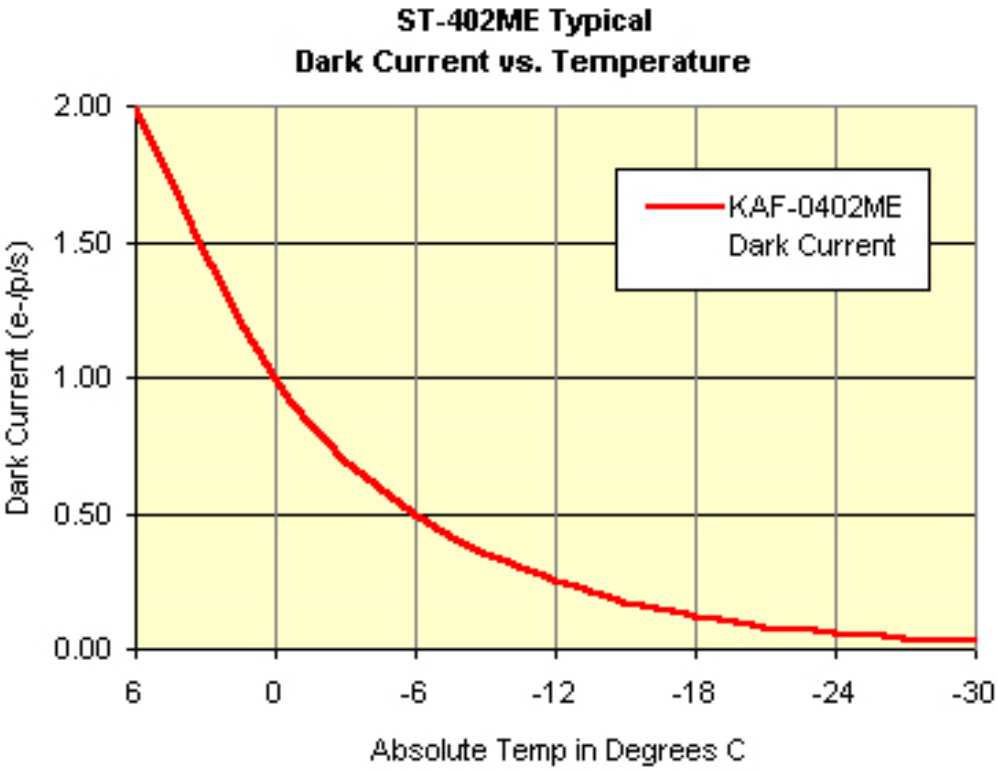
System Specifications

Cooling - Standard:	Single Stage Thermoelectric, Active Fan, -25C from Ambient Typical
Temperature Regulation:	+/-0.1ºC
Power:	12VDC, Power Supply included
Computer Interface:	USB 2.0 (USB 1.1 compatible)
Computer Compatibility:	Windows 98/200/Me/XP

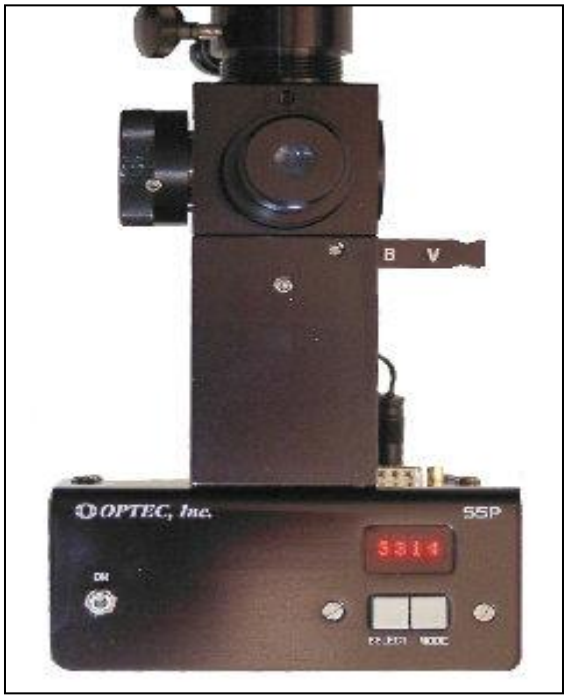
Physical Dimensions

Optical Head:	4 X 5 X 1.8 inches
CPU:	All electronics integrated into optical head; no CPU
Mounting:	T-Thread, 1.25" nosepieces included
Weight:	TBD
Backfocus:	0.69 inches


ST-402ME Enhanced Cooling Delta
Typical Delta from Ambient in Deg. C
0
-5
-10
-15
-20
-25
-30
-35
-40
0
2
4
6
8
10
12
14
16
Elapsed Time (Minutes)
Standard
Enhanced
REGULATION ON


ST-402ME Typical
Dark Current vs. Temperature
Dark Current (e-/p/s)
2.00
1.50
1.00
0.50
0.00
6
0
-6
-12
-18
-24
-30
Absolute Temp in Degrees C
KAF-0402ME
Dark Current

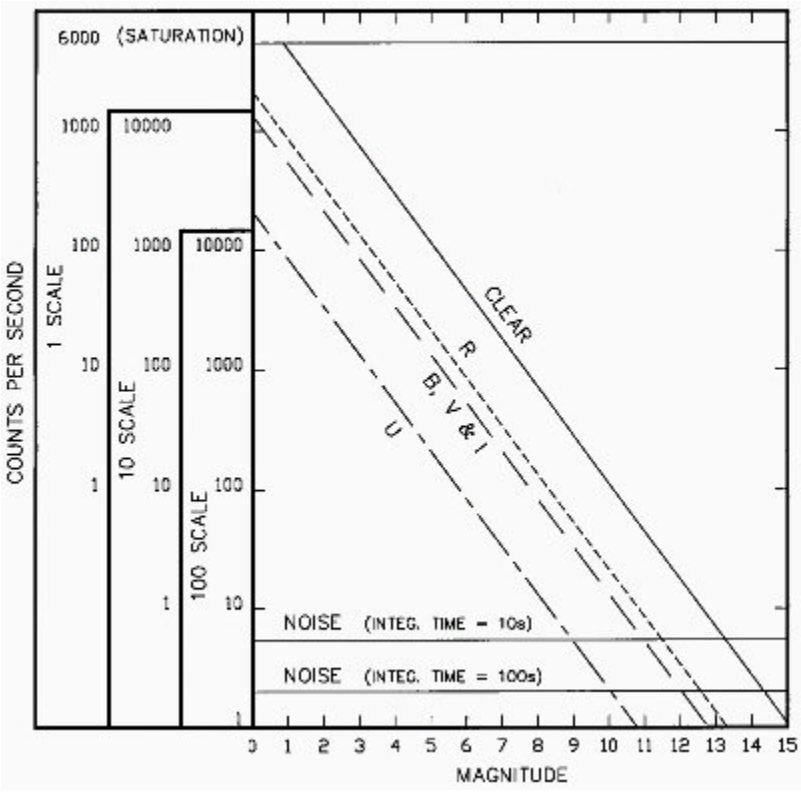
3.4Photometer: SSP3



The SSP-3 Photometer is the central part of a complete stellar magnitude measurement system. A complete set of precision filters, data acquisition interfaces, detector size and adapter options are available to fit any need. Light enters the photometer through the 1.25-inch telescope adapter and is directed either to the focusing eyepiece or the detector by means of a flip-mirror. The 1-inch focusing eyepiece comes with diopter adjustment and illuminated reticule that precisely defines the detector field of view. After a star is centered in the reticle ring, the flip-mirror is rotated to expose the detector. It is important to note that, unlike a photomultiplier system, the image of the star falls on the detector plane; therefore, a Fabry lens is not used and the photometer can be used with any focal length telescope.

A two position filter slider is mounted between the flip-mirror and the detector. Any pair of Johnson standard UBVRI or Clear filters can be factory mounted in a slider. Since the slider is easily pulled out of the photometer, sliders with other combinations of filters can be inserted. For most variable star observations, a single slider with B and V filters is recommended to begin with.

Powered by a 9-volt rechargeable battery, the SSP-3 offers unencumbered use at the telescope without annoying power cords. A single night's charge gives approximately 5 hours of operation, and if the observing session demands it, the battery charger can be used to give continuous power during use. The signal from the detector is converted to a frequency for counting and read on 4-character LED display. The integration times of 1, 5 or 10 seconds and voltage-to-frequency converter gains of 1, 10 and 100 are easily set by the two button menu on the front panel.



The SSP-3 has a serial output for connection to a PC serial (COM) port. The SSPDataq control and data acquisition program that comes with every unit operates all aspects of the photometer. It can record data for both fast and slow events and even perform a complete data reduction for B and V magnitudes based on the methods of Hendon and Kaitchuck for differential photometry. U, R and I data is not reduced by this program but are treated with the SSPData2.

To fill the needs of APT (Automatic Photoelectric Telescope) users, a motorized 6-position slider can be added to the SSP-3 making it an SSP-3A.

The performance of the SSP-3 can be seen on the graph above. The display output is expressed in counts per second vs. magnitude using the various filters. It should be noted that these are the approximate display counts, and that accurate magnitude should be determined using the accepted techniques of astronomical photometry. Each line on the chart represents the approximate relationship between stellar magnitude and counts from the SSP-3. The noise counts were determined by taking the standard deviation of 10 consecutive readings using either 1 or 10 second integration times. The intersection of the diagonal filter lines with the horizontal noise lines determine at what magnitude the signal-to-noise ratio is 1.

3.4.1 Specifications

Detector

Type	Silicon PN-photodiode
NEP	8x10-16 W/ *Hz* (typical)
Detector Size	1.3mm square (2.4mm square optional)
Spectral Range (5% points)	300 to 1100 nm
Shunt Resistance	50 G W (typical)
Surface Uniformity	<1%

Electrometer

Type	Current-to-Voltage
Bias Current .	15 pA Max.
Offset Voltage	<.25 mV
Open Loop Gain	100000 V/V Min.
Closed Loop Gain (Rf)	5x1010
Input Voltage Noise	4 μV(p-p) (.1 to 10Hz)
Input Current Noise .	003 pAAAA(.1 to 10Hz)
Maximum Output Voltage	6 V

A/D converter

Type	Voltage-to-Frequency
Full Scale Frequency	10 Khz
Full Scale Input Voltages	-66 mV (100 SCALE)
	-660 mV(10 SCALE)
	-6.6 V (1 SCALE)
Linearity	<0.3%
Offset	<.5mV (adjustable to 0)

Controller/display

Microcontroller	16F73 from Microchip
Oscillator	8.0000 MHz
Timer Accuracy	+/-25ppm at 25°C
Display	4-character 5x7 matrix
Character Height/Color	0.11 inch - Red

Power supply

Battery	9 volt NiCd (Type GE GC9 or Gould GS9T)
Operating Time	4 hours at 65°F
	3 hours at 20°F
Capacity	100 mA hours
Recharge Time	12 to 18 hours
	12 VDC Power Supply 12 volts DC regulated - 1000 mA

Eyepiece

Focal Length	25 mm
Type	Ramsden Reticle Illumination Green LED
Field of View	(80 inch FL) 0.4 degrees

Mechanical

Body Material	Aluminum 6061-T6 alloy
Finish Bright	Dip Black Anodized
Overall Length	9 inches (tip to tip)
Weight	2 lbs. 14 oz.
Telescope Coupler	1.25 inch (standard)

3.5 Instruments Assembled / Manufactured

During the project we had worked on several instruments, we had also participated in different workshops where we manufactured some equipment. Following is the list of it.

1. Night Sky Photometer at IUCAA
2. Coelostat
3. Safe Solar finder for Solar Telescope

3.5.1 Night Sky Photometer

It was manufactured in IUCAA in 2015 night sky photometer workshop. The aim of the workshop was to assemble a photometer which can measure the apparent brightness. The output

of the photometer was in volts. The circuit diagram is given below.

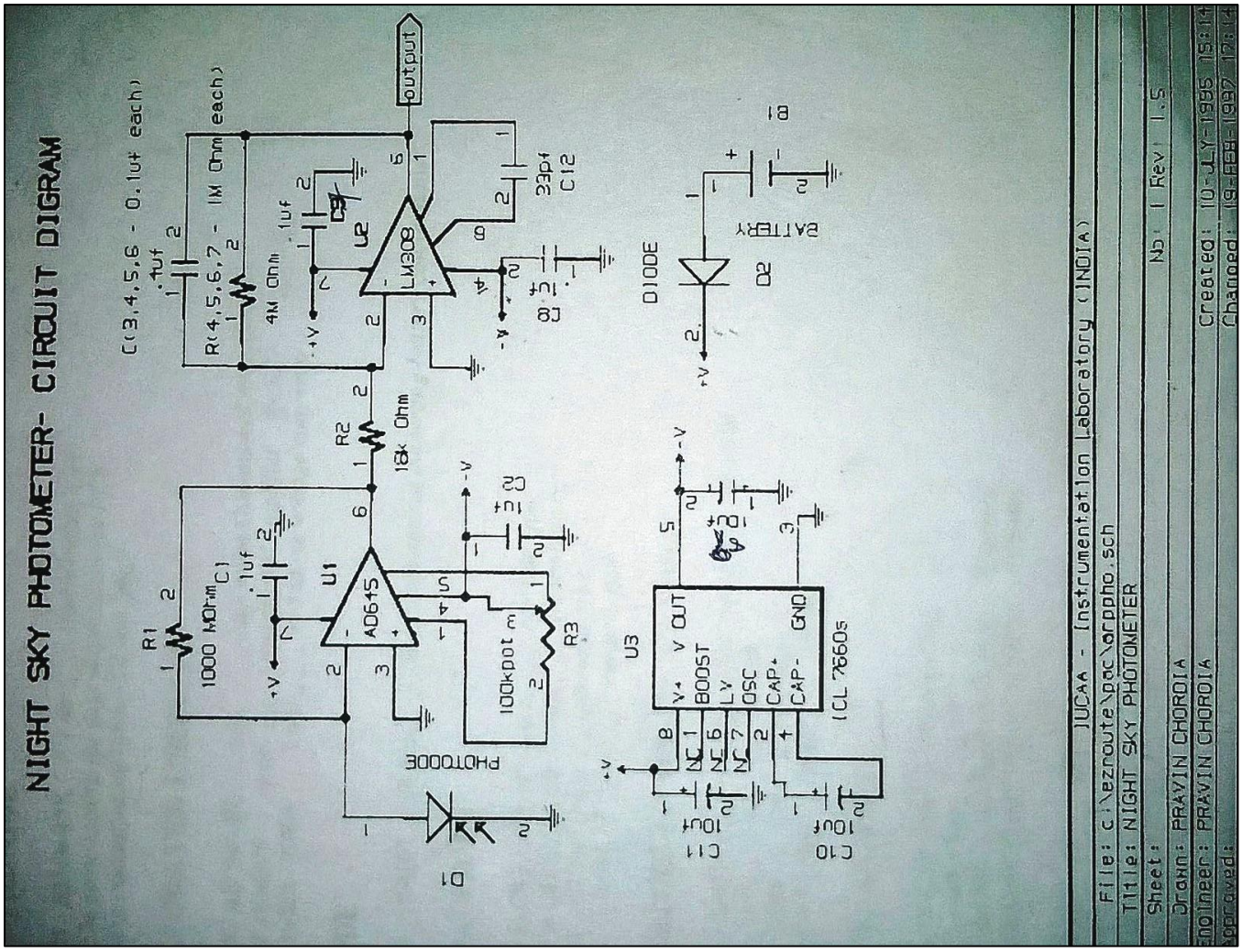


Fig. 3.5.1 Photometer Circuit Diagram

3.5.2 Coelostat

The Coelostat is being fabricated in Fergusson College which later will be used in Observatory. The hardware part is done and we are working on the tracking system. Two students from Astroclub named Chetan Thakur and Kiran Wani were assigned on this project and they were sent to IIA Banglore for two months to learn the design aspect of Coelostat.

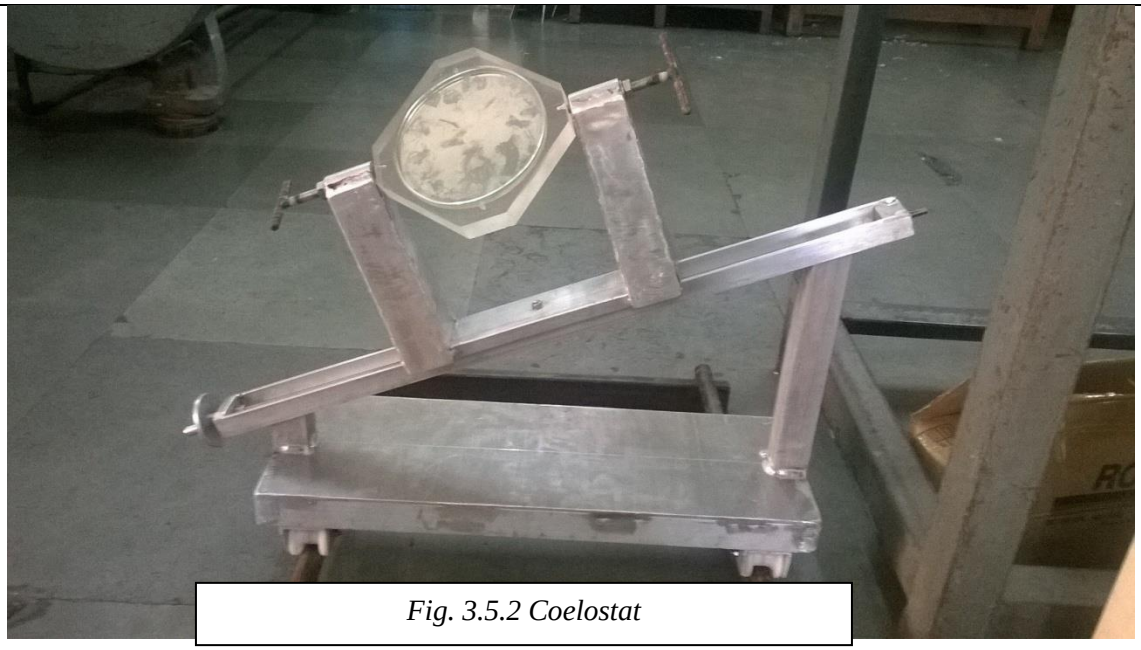


Fig. 3.5.2 Coelostat

3.5.3 Solar Finder

We had purchased a Lunt Solar Telescope from DBT grant for the observatory. We found that it was very difficult to point the solar telescope to the Sun; hence we developed simple devices which help us to locate the sun very precisely.

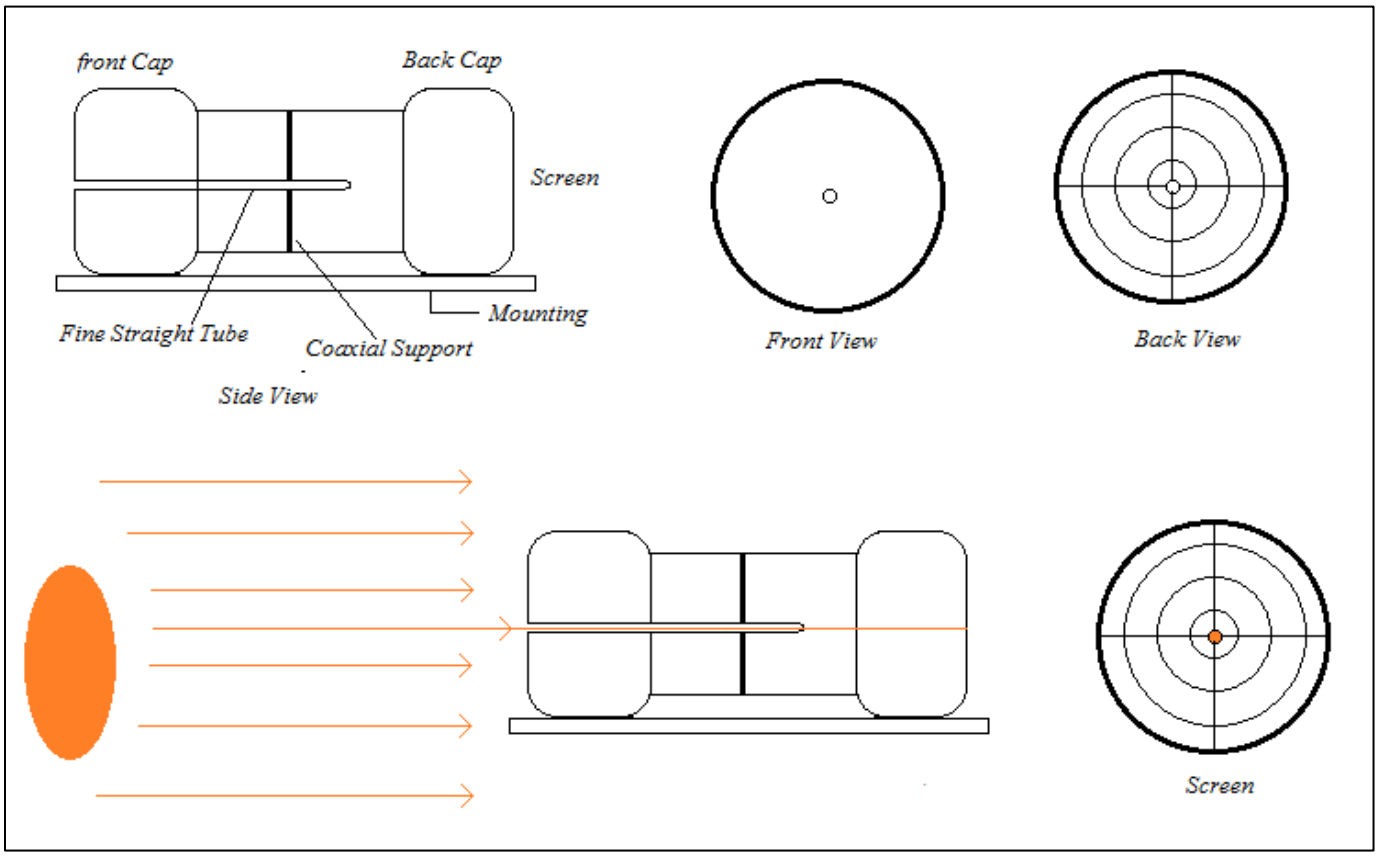


Fig3.5.1: Solar finder design

Automation

4.1 RTS2

Introduction:

Remote Telescope System (RTS-2) is an integrated package designed for remote observatory control. It is an linux based open source package. It is designed to run the observatory in fully autonomous mode by picking targets from database, taking images, storing and processing the images etc. It can control many mounts from various manufactures and also many types of cameras, focusers, and selectors. It is currently running on many observatories in the world.

The aim of the project was to make a client which will be an interface between RTS-2 running on an observatory and remote desktop.

In this document we start with basics of RTS-2. First will cover the basic installation and information related to commands to which familiarity is needed while working with RTS-2. At last we deal with automation scheme used for the counterpart observation. It describes in details, the methods used, why certain method is selected and how it has been implemented. We have controlled IUCAA's 'Celestron-Advanced GT' mount using RTS-2 installed on Ubuntu 14.04 LTS for testing purposes.

Getting Started :

In this section we will discuss about basic system and hardware requirements and controlling the mount by serial commands to test the connection. I'll be discussing about controlling the mount through linux. To connect NEXSTAR-compatible mount to a computer we need the following:

• RJ 8 (4P4C) to DB 9 (Female) cable
• USB to DB 9 (Male) Cable
• Serial command terminal (Available in Ubuntu repository with name 'gtkterm').
RJ 8 to DB 9

Before we start controlling the telescope via computer we need to configure the remote since the connection is through the remote. The RJ 8 end of the wire fits
at the bottom of the remote. To know how to configure the remote, one can refer to the user's manual for corresponding mount type 2 . Manual of 'Advanced GT' model can be found at http:

//www.celestron.com/c3/images/files/downloads/C80_C100ED_Master.pdf.

One can go through the section 'Hand Control' to know how to configure it. One has to set Latitude, Longitude, Date and Time and Alignment for mount to work properly. The NEXSTAR commands use serial communication protocol. Basics about serial communication can be found on following links and many other online sources,

http://www.z80.info/1656.htm
http://www.commfront.com/RS232_Protocol_Analyzer_Monitor/RS232_Analyzer_Monitor_Tester_Tutorial.htm

Now, to control telescope using serial commands one can follow following instructions. From Ubuntu 10.04 on-wards the USB to DB 9 converter has plug and play support and hence additional driver installation is not needed.

Steps to control mount using serial commands :

1. Connect the computer via two connectors to the bottom of the hand-controller.

2. Use commands ‘dmesg | grep ttyUSB’ or ‘ls /dev/tty’ to know which USB to serial port is communicating. It will usually be in the format ‘ttyUSB#’, where # is no. between 0 to 4.

3. Once you know which port is communicating we need to open serial port terminal to send serial commands. As mentioned ‘gtkterm’ is popular serial port terminal for Ubuntu and can be easily downloaded by ‘sudo apt-get install gtkterm’.

4. To connect to the port, root permission is needed hence to open the terminal one
1. should type ‘sudo gtkterm --port /dev/ttyUSB# ’, again # is the port number.

5. After opening the terminal one can pass Hexadecimal commands through the
2. gtkterm. The standard commands can be found here,

   http://www.celestron.com/c3/images/files/downloads/1154108406_nexstarcommprot.pdf

The baud rate and parity can be set from ‘configuration → port’. Apart from serial commands the mount can be controlled by popular Astronomy softwares such as ‘Stellarium’ 3 . By above method we can check whether the mount is properly responding or not. We will begin with installation of RTS-2 on Ubuntu.

Installation of RTS-2:

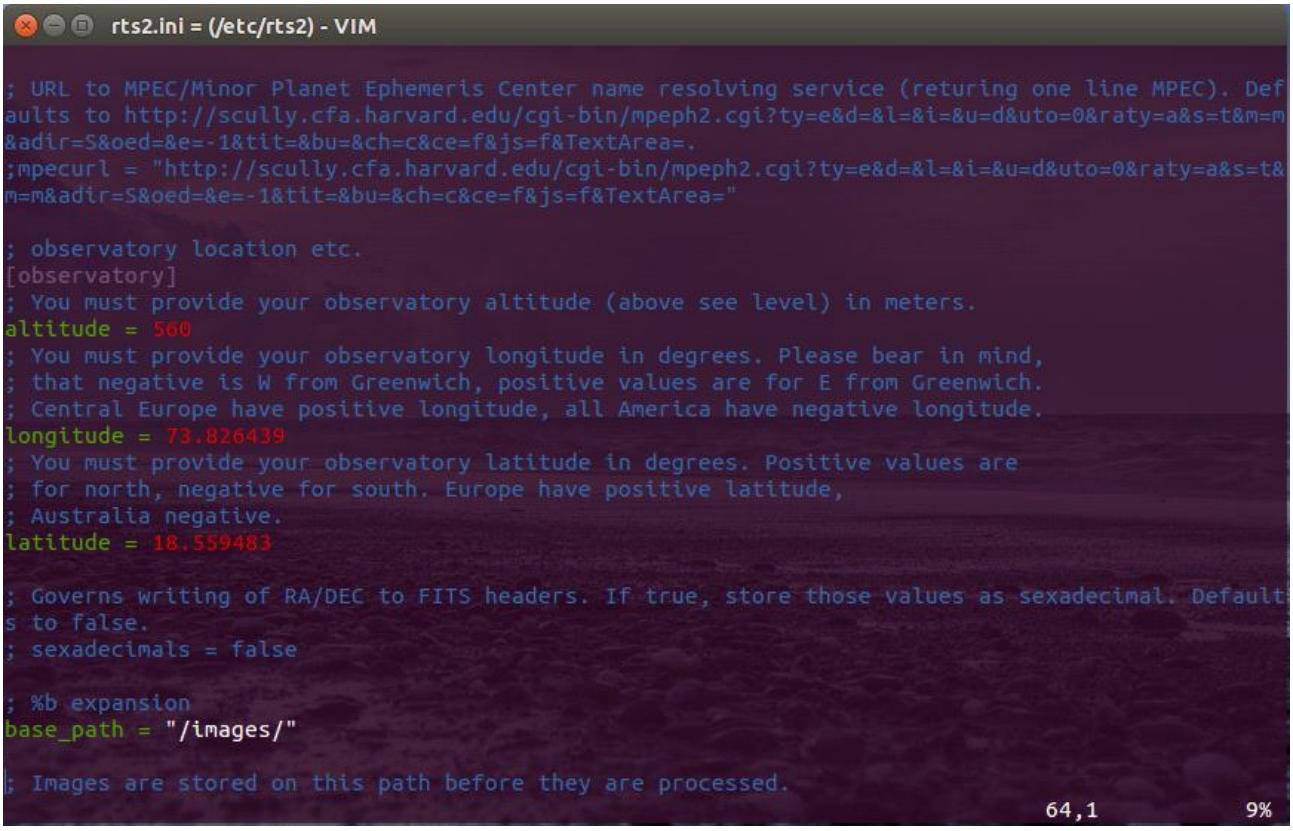
In this section we will discuss about installation of RTS-2. RTS-2 requires many other packages (e.g. postgressql, wcstools etc.). One needs to download them separately. Detailed information about installation can be found in Appendix B. It includes all the instructions as well as names of all the extra packages which are needed. One must take following precautions while installing RTS-2,

- Make sure you connect to a good internet connection because the size of downloads will be significant (~ 100 Mb).

- While downloading the extra packages check for availability of newer version than what is listed in the 'Appendix A' of the document (e.g postgressql, wcstools might have newer releases).

- As mentioned in the documentation, after installation one need to configure the configuration file called '/etc/rts2/rts2.ini'. Make sure you put your correct Latitude, Longitude and Altitude pertaining to the place form where you will be operating (Figure 2.1). Also in this file you can change many things such as default path to save the images, default image-names etc. One can get more info about rts.ini by typing a command 'man rts2.ini'.

- While editing the '/etc/rts2/devices' makes sure you write correct port and driver name. Also you put appropriate device name. Device name will be needed in the commands. You may keep the default device name. Following Figure is a screenshot of '/etc/rts2/devices' file.



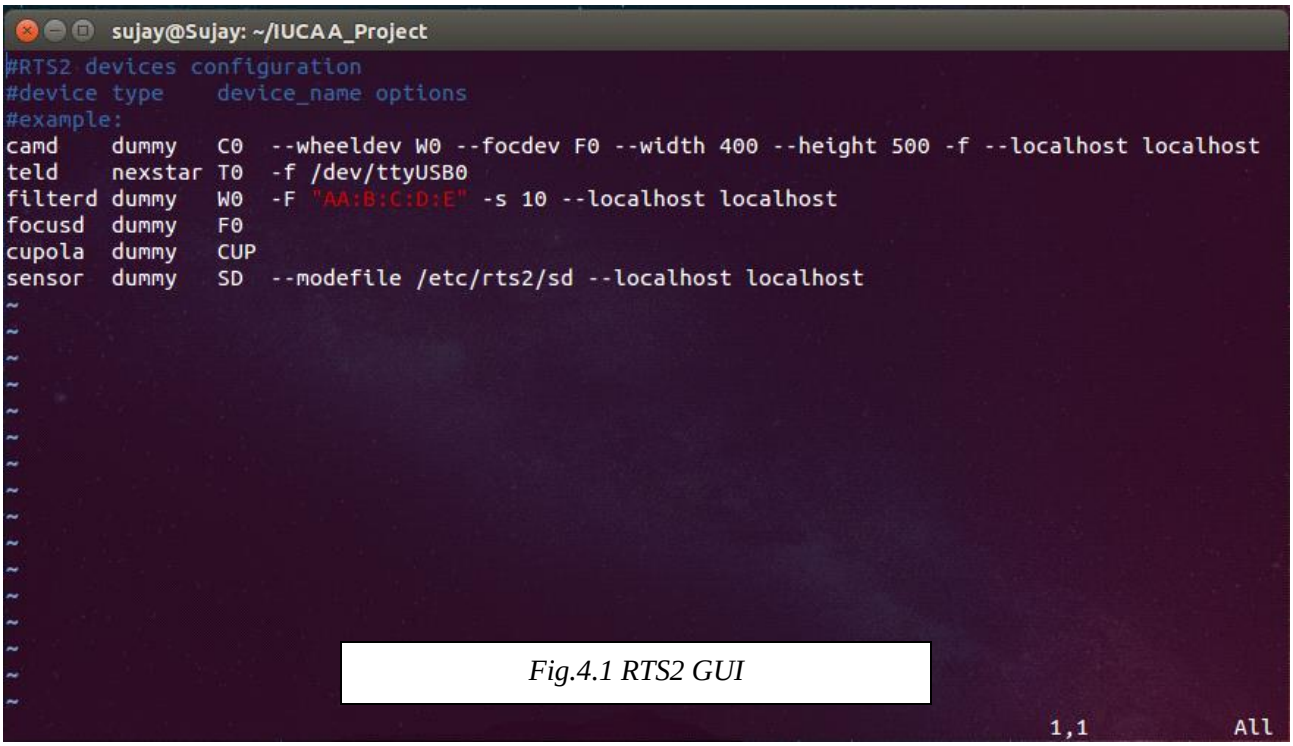


Fig.4.1 RTS2 GUI

Using RTS-2 :

In this section we will discuss about how to control the telescope using RTS-2. also, about writing short bash scripts to automate the observation. To start rts-2 to type the command,

```
sudo service rts2 start
```

Above command will start the rts2 service and will show device status as an output. The command also has ‘stop’ and ‘restart’ options. It is advisable to use ‘sudo service rts2 stop’ before disconnecting the telescope. One can view objects which are present at current time by typing following command,

```
rts2-targetlist
```

This will give detailed list of all objects (with RA-DEC and object ID) which are currently setting, rising or transiting.

A telescope can be controlled with RTS-2 via two ways. One is by typing commands in rts2-mon or via command rts2-scriptexec through which scripting is possible. Here we are going to discuss both ways briefly.

rts2-mon:

After starting the rts2 services if you type the command rts2-mon then a monitor is opened in the same terminal window (refer following figure). It shows all the connected devices and their detailed status.

The left pane shows the devices which are connected, the right pane shows detailed information about selected devices, the bottom pane display status and error messages. The space between

bottom pane and right pane is where a command is written. The ‘Tab’ key can be used to switch between panes and arrow keys can be used to pan through. To slew the telescope to given RA DEC following command can be used,

move 15:28:00.000 +73:27:33.00 where, the first argument is RA in HH:MM:SS.SSS and DEC in ±DD:MM:SS.SSS format.

If you want to stop the slewing in between, simply typing ‘stop’ will stop the motion. To come out of the monitor, press F10. More detailed info about rts2-mon can be found by typing man rts2-mon in terminal.

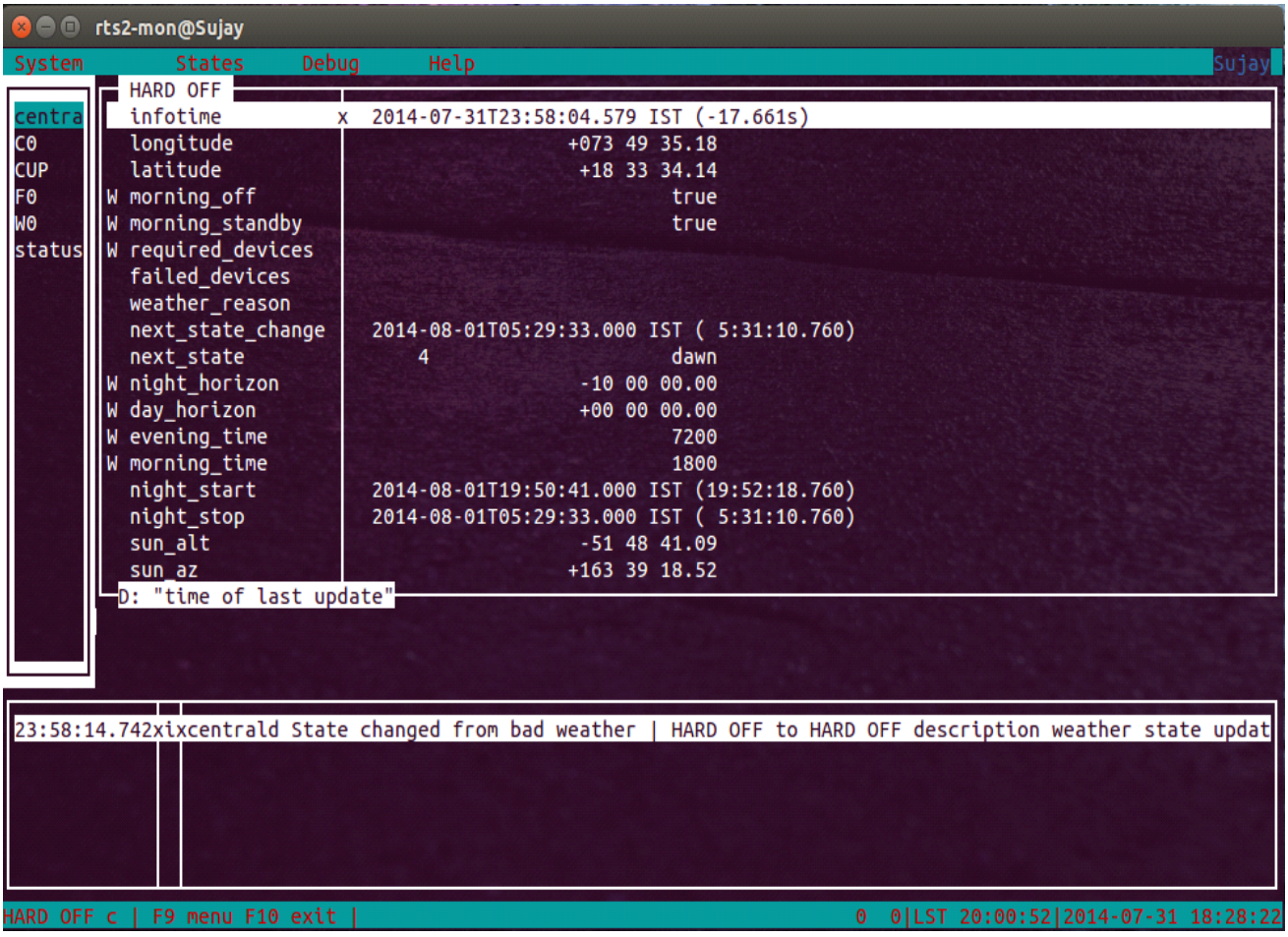


Fig.4.2 RTS2 terminal control

rts2-scriptexec:

This command allows us to use scripts to operate the telescope. We can write scripts simple scripts such as to take exposure of different time, take exposure and slew and again take exposure etc. and much more complex scripts. Scripting can be done in two ways, one by directly typing the commands in the command line or by making executable bash script file and passing it through rts2-scriptexec. Detailed info about the scripting commands can be found by typing man rts2.script. It is a very long manpage and hence it is advisable to open it in html format by typing following,

man –html=browser rts2.script where, ‘browser’ can be ‘firefox’ or ‘google-chrome’

Commands to pass in both the scripting ways are completely different. The third section of the manpage describes the commands which can be passed directly via command line. The nature of the terminal command is as follows,

rts2-scriptexec –d C0 –s ‘ ’ Note : Type the script between the quotes

Some sample scripts can be found at

http://rts2.org/wiki/doku.php?id=howto:taking_

images[]=scriptexec. This is a very tedious way of sending commands. More elegant method is the second one which is by writing a bash script and then executing the file using rts2-scriptexec. The fourth section of the manpage describes the commands which are accepted by the bash script.

All the commands in the script file are preceded by standard ‘echo’ or ‘read’ bash commands. For example, if you want to send a command to start the camera exposure, type,

```
echo exposure
read exp end
```

To set the exposure time type,

```
echo V C0 exposure = exposure time
```

To slew telescope to the given coordinates, type,

```
echo radec RA DEC
where RA DEC are in sexagesimal format as mentioned earlier
```

After writing the desired script in a file, the file should to be made executable and the script can be executed by typing following command,

```
rts2-scriptexec –d C0 –s ‘exe file path ’
```

Above command will run all the commands in the script line by line. It is advisable to open rts2-mon in another terminal window to check the status of the devices while the script is running. If there is any error in the script is will be shown in the rts2-mon. Following figure is the screenshot of typical script which can be passed,

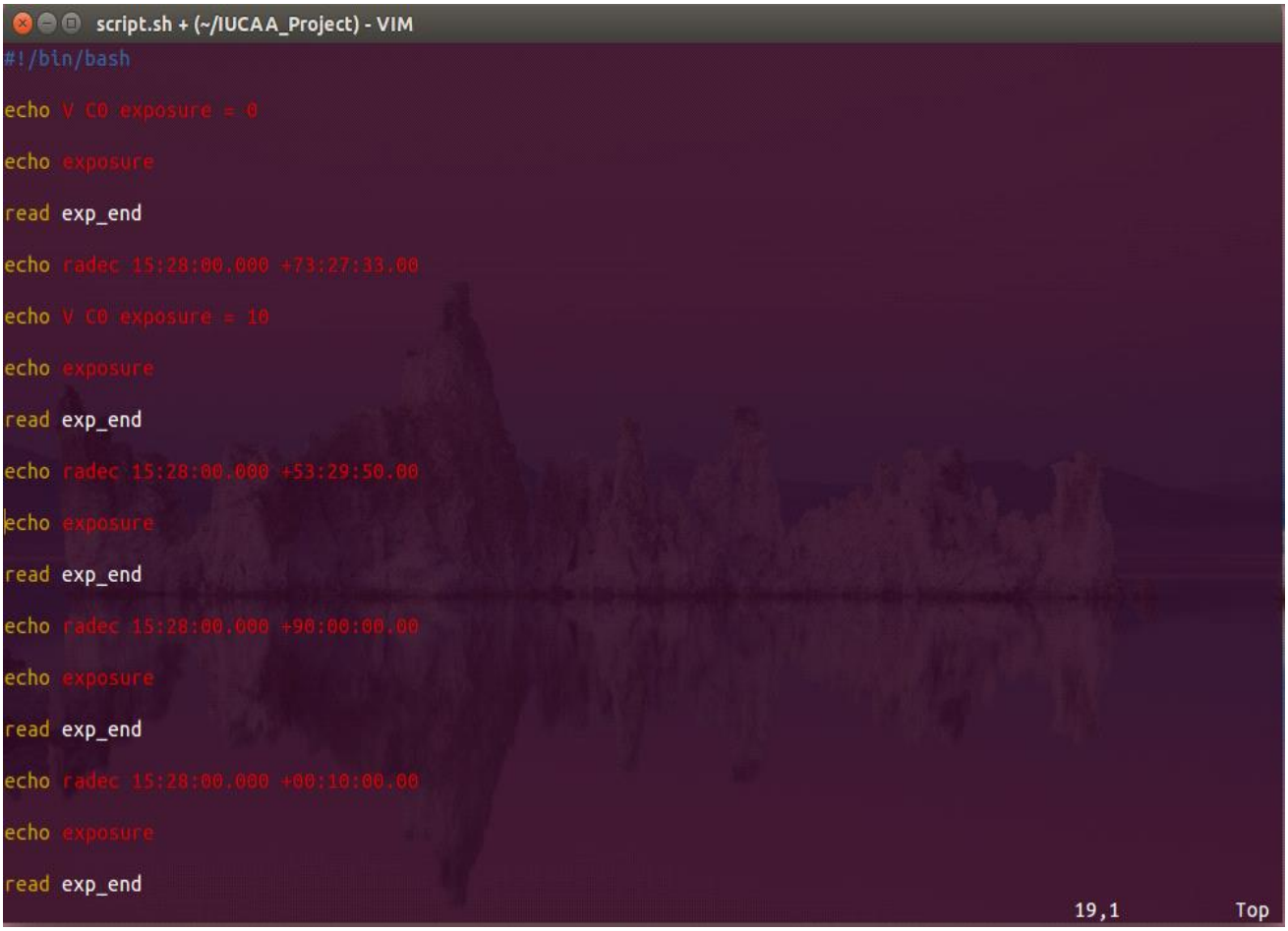


Fig.4.3 RTS2 Terminal

Above script tells telescope to point at four different coordinates and take 10 seconds exposure for each pointing. As mentioned the progress of the script can be seen in the rts2-mon 9window. Every time the script is executed, each exposure commands creates a FITS file with default name in the current directory. In our case its a blank file since we are using a dummy camera. This can be changed by changing the default directory configuration in rts2.ini or by putting commands to save file with particular name and address after the exposure command.

Automation of Telescope using RTS-2:

As the title of the document suggests, the primary goal of the project was to automate telescope using RTS-2. To be precise, following was the focus of the project,

- Communication with RTS-2: Explore various methods of sending targets to RTS-2.
- Create a "client" that can interface with RTS-2 via a protocol selected above. This client should take targets from say a file or a URL, and make sure RTS-2 observes them. To setup a automated communication a very robust protocol is needed. The protocol should send the targets properly to RTS-2, it should make sure that RTS-2 observes the target and should be able to make a log of observations/errors.

Following four methods were reviewed for communication,

a) Using rts2-scriptexec Client will make a bash script similar to one mentioned in section 3.2 and pass the script via rts2-scriptexec.

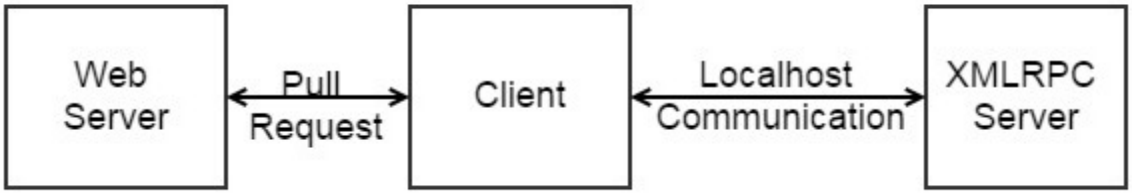
b) Target Creation: Client will create a new target with highest priority. Since the priority is high automatic selector of RTS-2 will select the newly created target.

c) GRBD daemon: RTS-2 has a 'grbd' daemon which looks for 'gcn' 1 notices and follow up the transient immediately. Create a similar daemon for GW trigger.

d) XMP-RPC Communication XML-RPC daemon of RTS-2 can receive remote HTTP calls and execute the tasks defined by the calls. It can listen to a specific server and port. One make XML-RPC to listen to certain server and send triggers over it.

The Client Program:

This section describes overview of the client server which will be receiving triggers and sending it to RTS-2. The overview of this scheme is shown in figure



The ‘web-server’ block is a central server which will be sending triggers to different observatories running the automation program. The program (client block) will send a pull request to get the co-ordinates from the server and send it to the RTS-2. Client will communicate with RTS-2 via JSON calls over localhost communication mentioned in previous section. Also, the client will send meta data of observations to the web-server so that it is possible to keep tracks of the observations. The pull requests are used specifically to avoid firewall permission issues since the web-server will not be running on the same network as the observatory.

The client program is a state machine with three states, Sleep/Idle, Observing, Image-Processing. Image-Processing can run parallel to the other two but the first two states are complimentary to each other. Reason for making it this way is that it is easier to code. For each state a separate code can be written and calls can be made in the program for state change. In following subsections I’ll talk briefly about the the tasks carried out by each state.

Sleep/Idle State:

As the name suggests, in this state, the program goes to sleep for fixed amount of time, ΔT. After every ΔT the program looks for target co-ordinates on the 'web-server' and as well send a health report of RTS-2 parameters so that the server knows that the connection is alive. If the server sends the co-ordinates, the program calls Observing State or else again goes to sleep for ΔT. A typical target file will contain following parameters,

1. Target Name
2. Target Right Ascension
3. Exposure Time for Image
4. Target Declination
5. No. of Images

Observing State:

When the Sleep/Idle state receives co-ordinates it calls this state. This state will crate a target in the RTS-2 database according to the co-ordinates specified. Then it will check for the constraints. If the constraints are met, it will overwrite the ongoing observation on RTS-2 with the newly created target and will observe the target. After the observation is done it will call Image-Processing state for basic image processing on observed image. Also, at the end of every observation it will also look for new target on server. If it receives the target, it will observe or else it will go to the sleep stat.

If constraints are not met, it will send an error message to the server and it will again look for new target. If is receives the target, it will again go through the above processing or else will go back to sleep state.

Image-Processing State:

It always better to know whether the observation has been completed or not. Since the observation is happening remotely, it might not be always possible to send an entire image to the server due to bandwidth limitations. Hence it is better to send metadata of image which will be in few kbs. To get this data, basic image processing has to be done. This includes astrometry to get co-ordinates of all the points in image and sextractor routine to get detection thresholds. Detection threshold information can be a clear indicator of quality of the image. All this information can be put into one file and it can be sent to the server. A typical metadata file will contain following parameters.

1. Observation Date and Time
2. RA and Dec of centre
3. RA and Dec of four corners
4. Exposure Time
5. Airmass
6. Filter
7. No. of stars in predefined magnitude range.
8. No. of Images

RTS-2 provides a great way to do this by providing python packages. It comes with rts2.astrometry and rts2.sextractor packages which can perform astrometry using astrome- try.net and can perform sextractor routines using sextractor installed on system. In the next section I will talk about the python code which I created for Observing State.

Section A

Some precautionary notes

1. It is better to connect the cord to the same USB port every time as connecting it to the different port might cause no detection problems.

2. If you are using any rechargeable power supply for the telescope, always keep it to the maximum charge by charging it frequently. Low power can cause the handcontroller to behave improperly and in turn the telescope.

3. While giving the coordinates try avoiding RA-DEC combination in which too much rotation is involved in both the axes. Such case might cause the plastic part of the mount to hit each other. If notic such thing might happen immediately pass ‘stop’ command through rts2-mon.

4. While writing the script for pointing the telescope to multiple targets, if you directly write first command as first target coordinate i.e ‘radec RA DEC’, then it ignores that target and goes to the second one. This can be resolved by giving zero second exposure before the first target coordinates.

Section B

This section contain all the instructions about installation of RTS-2. As mentioned earlier, RTS-2 is primarily designed for Linux system. It is best run on Ubuntu platform. To install it on any Ubuntu distribution follow following instructions. First you need to download the source code from following link,

http://sourceforge.net/projects/rts-2/files/

After you have the source code download the following dependencies by typing,

```
user@host:~$ sudo apt-get install subversion postgresql postgresql-server-dev
libecpg-dev automake libtool libcfitsio3-dev libnova-dev
libecpg-dev gcc g++ libncurses5-dev libgraphicsmagick++1-dev
libx11-dev docbook-xsl xsltproc libxml2-dev libarchive-dev
```

If you wish to configure RTS-2 with 'WCS' then you need to install wcstools manually. Current version of which can be found here,

http://tdc-www.harvard.edu/software/wcstools/

If you are on a 64-bit system, then while compiling it should be compiled with fPIC CFLAG. See INSTALL file in RTS-2 source code directory for more information about compilation of WCS. Once WCS is installed RTS-2 can be installed by going to the RTS-2 directory and typing following commands

```
user@host:~/rts2$ sudo ./configure
```

To configure without 'WCS', type,

```
user@host:~/rts2$ sudo ./configure --without-wcs
```

```
user@host:~/rts2$ sudo make
```

```
user@host:~/rts2$ sudo make install
```

```
user@host:~/rts2$ sudo ./rts2-init
```

The last command creates the /etc/rts2 folder. As described in Chapter 2 the configuration files in this folder can be edited. One may encounter following error after typing any of the RTS-2 commands,

```
rts2-mon: error while loading shared libraries: librts2db.so.0:
open shared object file: No such file or director
cannot
```

This is because the system is unable to locate shared libraries added by the RTS-2. This can be solved by manually adding the path of shared libraries for RTS-2 as follows,

```
user@host:~$ cd /usr/local/lib/
user@host:/usr/local/lib$ sudo ldconfig
user@host:/usr/local/lib$ sudo ldconfig -v | grep librts2db.so
librts2db.so.0 ->librts2db.so.0.0.0
```

More info about shared libraries can be found here http://www.cyberciti.biz/tips/ linux-shared-library-management.html. After this postgresql database set up is needed. Next set of commands describes the postgresql configuration, First create a RTS-2 user,

```
user@host:~$ rts2-user -a [username] --password [password]
```

After this log in as postgres user,

```
user@host:~$ sudo su postgres
```

Then create database users named ‘root’ and ‘username’ (This should be your system username which is also same as RTS-2 username),

```
postgres@host:~$ createuser root
postgres@host:~$ createuser [username]
```

Then create a new database named ‘stars’ and build it,

```
postgres@host:~$ createdb stars
postgres@host:~$ cd /home/username/rts2/src/sql
postgres@host:/home/username/rts2/src/sql$ ./rts2-builddb stars
```

Give permissions to access and edit database to root and RTS-2 user,

```
postgres@host:~$ psql stars
psql (9.3.5)
Type "help"for help.
stars=# create group observers
stars=# alter group observers add user root;
ALTER ROLE
stars=# alter group observers add user [username];
ALTER ROLE
stars=# \q
```

Keep in mind the ‘;’ at the end of the ‘alter’ command. Also, it is necessary to ‘root’ as user to the database along with RTS-2 user. Not adding it will give errors while accessing targetlist. After this, configure database with the devices,

```
postgres@host:~$ rts2-configdb stars -c [camera-name]
postgres@host:~$ rts2-configdb stars -t [telescope-name]
postgres@host:~$ rts2-configdb stars -f [focuser-name]
```

After this, rts2 can be started by typing following command,

```
user@host:~$ sudo service rts2 start
```

4.2 IRAF

IRAF (image reduction and analysis facility) is a collection of software written at the National Optical Astronomy Observatory (NOAO) geared towards the reduction of astronomical images in pixel array form. This is primarily data taken from imaging array detectors such as CCDs. It is available for all major operating systems for mainframes and desktop computers. Although written for UNIX-like operating systems, use on Microsoft Windows is made possible by Cygwin. It is primarily used on Linux distributions, with a growing share of Mac OS X users. IRAF is installed by default in Distro Astro, a Linux distribution for astronomers.

IRAF commands (known as tasks) are organized into package structures. Additional packages may be added to IRAF. Packages may contain other packages. There are many packages available by NOAO and external developers often focusing on a particular branch of research or facility. Of particular note are the STSDAS and TABLES packages by the STScI.

Just a few examples of functionality available in IRAF would be calibration of the fluxes and positions of astronomical objects within an image, compensation for sensitivity variations between detector pixels, combination of multiple images or measurement of the redshifts of absorption or emission lines in a spectrum

Installation

= IRAF Setup on Ubuntu/Debian Linux =

This page describes how to setup IRAF on a 32-bit Ubuntu/Debian machine.

Installation of X11IRAF, DS9 and key packages are also included.

It will be easiest if you just copy-paste the instructions one by one except

for the parts in [[ ]].

For 64-bit installation, please see –

http://www.astronomy.ohio-state.edu/~khan/iraf/iraf_step_by_step_installation_64bit

An excellent and easy alternative installation option is provided by the Ureka project –

http://ssb.stsci.edu/ureka/

```
sudo apt-get install tcsh libxss1 libncurses5 libXmu6:i386
```

```
wget ftp://iraf.noao.edu/iraf/v216/PCIX/iraf.lnux.x86.tar.gz
```

[[If the prompt does not return after 100% then hit ctrl+c and proceed]]

```
sudo mkdir /iraf
```

```
sudo mkdir /iraf/iraf
```

```
sudo mv iraf.lnux.x86.tar.gz /iraf/iraf/.
```

```
Cd /iraf/iraf
```

```
sudo tar –zxf iraf.lnux.x86.tar.gz
```

```
sudo rm iraf.lnux.x86.tar.gz
```

```
sudo ./install
```

[[Keep hitting Enter for all prompts]]

```
sudo mkdir /iraf/x11iraf
```

```
cd /iraf/x11iraf
```

```
sudo wget http://iraf.noao.edu/iraf/ftp/iraf/x11iraf/x11iraf-v2.0BETA-bin.linux.tar.gz
```

```
sudo tar –zxf x11iraf-v2.0BETA-bin.linux.tar.gz
```

```
sudo rm x11iraf-v2.0BETA-bin.linux.tar.gz
```

```
sudo ./install
```

[[Keep hitting Enter for all prompts]]

```
sudo wget http://ds9.si.edu/archive/linux/ds9.linux.7.2.tar.gz
```

```
sudo tar –zxf ds9.linux.7.2.tar.gz
```

```
sudo rm ds9.linux.7.2.tar.gz
```

```
sudo mv ds9 /usr/local/bin/.
```

```
Cd
```

```
wget http://www.astronomy.ohio-state.edu/~khan/iraf/iraf
```

```
sudo chmod u=rwx iraf
```

```
mkdir IRAF
```

```
cd IRAF
```

```
mkiraf
```

[[Select “xgterm” as your IRAF shell when prompted]]

```
exit
```

The command ./iraf from home directory will now launch a complete IRAF session containing DS9,

xgterm and ecl, based in the IRAF directory. Alternately, start “xgterm’ (typing xgterm),

cd into the directory where you did “mkiraf”, type “ecl” for enhanced IRAF or

“cl” for VO IRAF, and “ds9” for standalone DS9 launch.

The following steps show how to add external packages. If you don’t need extra packages,

then you are done already.

```
cd /iraf/iraf/extern
```

```
sudo ./configure
```

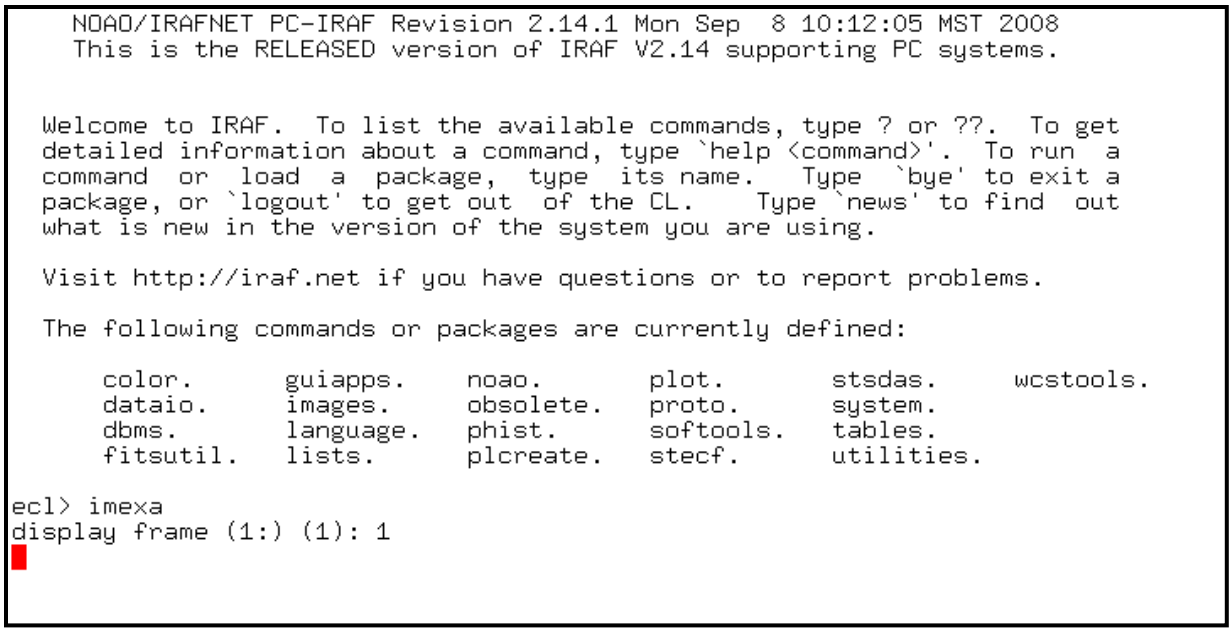
[[The next two lines will take some time to complete, even with fast internet. Be patient.]]

```
sudo make adccdrom ctio cfh12k esowfi mscdb mscred stsdas nfextern optic
```

```
sudo make deitab euv mem0 mtools rvsao song sqiid stecf ucsclris upsqiid xdimsum
```

```
exit
```

Fig.4.1 IRAF window


NOAO/IRAFNET PC-IRAF Revision 2.14.1 Mon Sep 8 10:12:05 MST 2008
This is the RELEASED version of IRAF V2.14 supporting PC systems.
Welcome to IRAF. To list the available commands, type ? or ??. To get detailed information about a command, type `help <command>'. To run a command or load a package, type its name. Type `bye' to exit a package, or `logout' to get out of the CL. Type `news' to find out what is new in the version of the system you are using.
Visit http://iraf.net if you have questions or to report problems.
The following commands or packages are currently defined:
color. guiapps. noao. plot. stsdas. wcstools.
dataio. images. obsolete. proto. system.
dbms. language. phist. softools. tables.
fitsutil. lists. plcreate. stecf. utilities.
ecl> imexa
display frame (1:) (1): 1

4.3 DS-9

What is DS-9

SAOImage DS9 is an astronomical imaging and data visualization application. DS9 supports FITS images and binary tables, multiple frame buffers, region manipulation, and many scale algorithms and colormaps. It provides for easy communication with external analysis tasks and is highly configurable and extensible via XPA and SAMP.

DS9 is a stand-alone application. It requires no installation or support files. All versions and platforms support a consistent set of GUI and functional capabilities.

DS9 supports advanced features such as 2-D, 3-D and RGB frame buffers, mosaic images, tiling, blinking, geometric markers, colormap manipulation, scaling, arbitrary zoom, cropping, rotation, pan, and a variety of coordinate systems.

The GUI for DS9 is user configurable. GUI elements such as the coordinate display, panner, magnifier, horizontal and vertical graphs, button bar, and color bar can be configured via menus or the command line.

How to Install DS-9 in Linux

The process of installing DS-9 is simple once the IRAF is installed in PC. Use the following commands.

```
cd ~/iraf
```

```
wget http://hea-www.harvard.edu/saord/download/ds9/linux/ds9.linux.5.6.tar.gz
```

```
tar -zxf ds9.linux.5.6.tar.gz
```

```
sudo mv ds9 /usr/local/bin
```

```
sudo chmod +x /usr/local/bin/ds9
```

IRAF Commands

1 Setting up your IRAF environment

Once the user has logged into IRAF he may want to customize his IRAF environment for that session. The *loginuser.cl* file can also be used for this purpose at IRAF startup time.

The IRAF CL maintains a table of "environment variables" which affects the operation of many IRAF tasks. A complete list of these variables can be viewed by typing the following (the output of SHOW is "piped" to the input of the PAGE command, a simple task for looking at one page of text at a time):

cl> **show | page**

The SET command only modifies the variable temporarily, i.e., while the current package is loaded. Using RESET will normally change the value of the variable for the remainder of the current IRAF session.*

IRAF environment variables can also be used to define IRAF logical names for directories. These logical names can be strings denoting either host level pathnames or other IRAF logical names. For example, the test image, *dev$pix,* that is used in many of the examples in this document, can be traced to the following host directory with the SHOW task. The *iraf* logical directory will be a different host level pathname for each installation, but the *dev* logical

directory will be the same on all hosts. IRAF logical directories are followed by a "$". In the last example, the task PATH is used to print the equivalent host pathname directly, including the node or machine name.

```
cl> show dev
        iraf$dev/
cl> show iraf
    /ursa/iraf/iraf/
cl> path dev$
    ursa!/ursa/iraf/iraf/dev/
```

This use of IRAF environment variables as logical directories is discussed further in §3.4.

Defining your terminal type

Your terminal type will be set at login time by the execution of the "stty" statement in the *login.cl* file. The terminal type was chosen by the user when MKIRAF was run initially. At startup time, IRAF will print a message on the screen echoing the terminal type that it is setting.

```
% cl
setting terminal type to gterm...
```

You can always check or reset your terminal type once IRAF is loaded.

```
cl> stty                    #check terminal type
cl> stty xtermjh nlines=24  #reset terminal type to xtermjh (24 lines)
```

The STTY command sets two environment variables, *terminal* (for text input and output) and *stdgraph* (for graphics output). If there is some question about what terminals and terminal emulators are supported by your IRAF distribution page through the *graphcap* file in the *dev* directory and look at the *STDGRAPH* entries.

```
cl> page dev$graphcap
```

Notice that in the default *login.cl* file that some checking at the host level is done first to attempt a first guess at the correct terminal type. The user can do something similar in the *loginuser.cl* if he wants to modify this scheme.

Selecting hard copy devices

Two environment variables are used to define the names of printer and plot devices that IRAF will use for hard copies. These devices will vary from site to site.

```
cl> show printer              # show default printer
cl> show stdplot              # show default plotter
cl> reset printer=<lwa>       # change default printer for this session
cl> reset stdplot=<lwb>       # change default plot device for this session
```

Typing the command **devices** (if the appropriate file has been edited for your site by your IRAF site manager) will list your options for these variables.

There are three built-in options for *stdplot* that do not send the output to a hard copy device, but rather create Encapsulated PostScript files on disk with names similar to "sgi12345.eps". These "device" names are *eps* (produces a plot in the middle of the page), *epsl* (produces a landscaped plot), and *epsh* (produces a plot at the top of the page).

Choosing an editor

IRAF does not have its own editor but rather uses host level editors. You can set the IRAF environment variable *editor* to the host level editor that you want to use (it of course must exist) and then execute the IRAF task EDIT to invoke this editor. The editor that you choose must have a *.ed* entry in the IRAF *dev* directory to work properly with the tasks EDIT, EPARAM and EHISTORY. Currently there is support for *vi*, *edt*, *emacs*, and *memacs*.

cl> **show editor** # show default editor

cl> **set editor=vi** # set the editor to *vi*

cl> **edit** <filename> # invoke the editor

Packages and tasks

The programs or commands that the user executes to perform specific functions are called tasks. Tasks that perform similar functions are grouped together into packages. The packages for the IRAF core system are listed below.

dataio	-	Data format conversion package (RFITS, etc.)
dbms	-	Database management package (not yet implemented)
images	-	General image processing package
language	-	The command language itself
lists	-	List processing package
local	-	The template local package
obsolete	-	Obsolete tasks
plot	-	Plot package
proto	-	Prototype or interim tasks
softools	-	Software tools package
tv	-	Image display load and control package
system	-	System utilities package
utilities	-	Miscellaneous utilities package

A package must be loaded in order to execute a task that is in it. A package is loaded by simply typing its name or enough characters to uniquely identify its name, i.e., **softools** or **so**. Note that IRAF is case sensitive! The prompt reflects the current package (the last package loaded). The

last package that was loaded can be unloaded by typing **bye**. What packages are currently loaded can be checked by typing**package**. Once a package is loaded its tasks are available for execution until the package is unloaded. Loading a new package does not unload the previous package. There is essentially no overhead in having a variety of packages loaded at the same time.

```
cl> package        # check to see what packages are loaded
cl> softools       # load the SOFTOOLS package
so> package        # check to see what packages are loaded
so> bye            # unload the SOFTOOLS package
cl>                #  prompt goes back to previous prompt
```

When you log into IRAF all the packages in the IRAF core system are automatically loaded for you (except SOFTOOLS), unless this has been changed by your site manager.

General help facilities

There are several online help facilities within the IRAF system.

Simple tools

There are several simple help tools that the user may find useful.

```
cl> ?              # list the tasks in the current package
cl> ?dataio        # list the tasks in a specific package - the package must be loaded
cl> ??             # list all tasks currently loaded
cl> package        # list all packages currently loaded
```

Online manual pages

All tasks in the IRAF system have detailed online help or manual pages. All packages have online menu pages that consist of a list of the tasks in each package plus a one line description of each task. These manual pages and menus are part of an IRAF online help database that is available to the user at login time, i.e., the tasks and packages do not need to be loaded to access these online help facilities.

The online help database is dynamic and can include IRAF software packages other than the IRAF core system when these packages are installed as part of the main IRAF system. Even local software can be installed so that their help pages are part of this database.

In the next few sections on "help", the syntax used for the task executions in the examples may not be obvious to the new user. Task parameters and the syntax for command execution are covered in more detail in §1.5. and §1.6.

The HELP task

The HELP task will page through the online manual pages for a specific task or package menu. The user can move forward through this help file spacing by line (**CR**), half page (**d**), or full page (space bar). A **q**quits the help. A short summary help line is printed at the bottom of each page during the execution of HELP. Try, for example,

cl> **help** # help for the current package
cl> **help help** # help for the HELP task itself
cl> **help rfits sec=examples** # help for the EXAMPLES section of the RFITS manual page

cl> **help rfits opt=source** # show the source code for the task
cl> **help dataio** # help for a specific package

If you are uncertain about what package a task is in, use the HELP task. The first line of the help page gives the package name for a task as well as the date of the last modification of the help page. For example, the RFITS help page shows

RFITS (Jan90) dataio RFITS (Jan90)

There are other documents in IRAF that have the same format as the online manual pages (file names ending in *.hlp*) but are not part of the IRAF online help database. These files can also be accessed with the HELP task.

cl> **cd mwcs** # change directories to the logical directory *mwcs*
cl> **help MWCS.hlp f+** # read this file with HELP
cl> **cd** # change directories back to your IRAF home directory

The PHELP task

The PHELP task is similar to the HELP task except that it allows the user to page backwards and forwards through the online manual pages. It also has a wider range of options for paging that can be queried with **?** at any time during its execution.

cl> **phelp phelp** # get help for PHELP task
? # print summary help line
q # quit PHELP
cl> **phelp combine all+** # get help on all tasks called "combine"
N # go to help page for next "combine" task
P # go to help page for previous "combine" task
q # quit help

Printing online manual pages

Hard copies of the online manual pages or any *.hlp* file can be generated using the HELP and LPRINT tasks along with the pipe symbol (|). The output printer can be the default printer specified by the environment variable *printer* (see §1.1.2.) or it can be set explicitly.

cl> **help help | lprint** # send help file to default printer device
cl> **help help | lprint dev=**<lw> # send help file to a specific printer
cl> **help mwcs$MWCS.hlp f+ | lprint dev=**<lw> # send a non-online help file to a specific printer
cl> **help dataio.* | lprint** # print all online help for the DATAIO package

Finding a task with a particular function (REFERENCES)

The REFERENCES task can be used to locate a task that will provide a certain functionality. The task searches the online help database for a string supplied by the user. Only the "one-liner" information from the package menus is used in this search. Since this can be rather slow an option is available to create a quick-reference file to make future searches faster; this quick-reference file is stored in your *uparm*directory.

cl> **references help** # list all tasks with "help" in their description
cl> **refer upd+** # generate a quick-reference file
cl> **refer smooth** # list all tasks with "smooth" in their description using the quick-reference file

Doing things with directories

IRAF uses a *virtual* file system. What this means to the user is that directory and file names from within IRAF will look the same on any computer. This is particularly noticeable in VMS/IRAF - try typing**dir** and then **!dir**.*

---

*The "!" is used as an escape to the host operating system and allows the user to execute a host level command from within IRAF.

There are many ways in IRAF to work with directories. Remember that your IRAF *home* directory is the host level directory from which you logged into IRAF. The IRAF *home* directory and the *uparm*directory are also both environment variables set to logical directories, at login time.

cl> **show home** # show pathname for IRAF home directory
cl> **show uparm** # show uparm pathname

Creating/removing directories

You can create a directory from within IRAF using the MKDIR task.*

---

*Be forewarned about long pathnames in IRAF. Many IRAF tasks have an upper limit of 63 characters to file names, including the full host level pathname.

---

cl> **diskspace** # check available diskspace
cl> **mkdir nite1** # create a subdirectory called *nite*1
cl> **mkdir home$nite1** # create a subdirectory in your IRAF *home* directory
cl> **mkdir** </tmp8/irafdir> # create a directory on a scratch disk (UNIX host)
cl> **mkdir** "<scr1:[irafdir]>"# create a directory on a scratch disk (VMS host)

Currently there is no way to remove a directory from within IRAF; the user must do this at the host level. Again note the use of the escape character (**!**) in some of the examples.

- The following is an example of how to remove a subdirectory called *nite1*, assuming a UNIX host. Be sure to delete all IRAF images in the directory first with IMDELETE (see §1.8.5.).

```
cl> cd nite1              # go to nite1 subdirectory
cl> imdelete *.imh        # delete all images
cl> cd ..                 # go up one level of directories
cl> !rm -r nite1          # delete directory nite1
```

- On a VMS host you must first remove all of the files in a directory, then change the protection on that directory, and then finally delete the directory itself.

```
cl> cd nite1                          # go to nite1 subdirectory
cl> imdelete *.imh                    # delete all IRAF images
cl> delete *                          # delete all remaining files
cl> cd ..                             # go up one level of directories
cl> !set protection=(O:RWED) nite1.dir;1  # unprotect directory
cl> !delete nite1.dir;1               # remove directory
```

Changing directories

The user can move around the IRAF directory structure using IRAF directory names, logical directories, or host level pathnames. Some examples may demonstrate these various methods.

```
cl> path                  # type current directory
cl> cd                    # go to home directory
cl> mkdir nite1           # create subdirectory nite1
cl> cd nite1              # change to the nite1 subdirectory
cl> cd ..                 # go up one directory level
cl> cd iraf$doc           # change to a subdirectory of the logical directory iraf
cl> back                  # go back to the previous directory
cl> set data=</tmp8/data/>    # define a logical directory (UNIX host)
```

```
cl> set data="<scr2:[data]>"  # define a logical directory (VMS host)
cl> cd data                   # change to the logical directory data
cl> cd </tmp8/data>/          # change to host directory (UNIX host)
cl> cd "<scr2:[data]>"        # change to host directory (VMS host)
```

Parameter files

IRAF tasks are the commands the user executes to perform certain operations. And the parameters for the tasks determine how those operations will be done. This section discusses the various ways to look at and to change task parameters.

The default parameter file for the RFITS task is listed below.

```
   fits_file = "mta"          FITS data source
   file_list =                File list
   iraf_file = ""             IRAF filename
  (make_image = yes)          Create an IRAF image?
 (long_header = no)           Print FITS header cards?
(short_header = yes)          Print short header?
   (datatype = "")            IRAF data type
      (blank = 0.)            Blank value
      (scale = yes)           Scale the data?
 (oldirafname = no)           Use old IRAF name in place of iraf_file?
     (offset = 0)             Tape file offset
      (mode = "ql")
```

There are two types of parameters, required (often called query or positional) parameters and hidden parameters. In our example above the first three parameters are required parameters; these parameters must be specified each time the task is executed. The parameters in parentheses

are called hidden parameters; if these parameters are not specified at execution time the current default parameter values will be used.

The user can customize the parameters for any task; these customized parameter files are then stored in the user's *uparm* directory. Whenever a task is executed IRAF first searches the user's *uparm* directory for a customized parameter file; if one does not exist the system default file is used.

A parameter file name in the *uparm* directory is a combination of the package and task name (the first two characters plus the last character of the package name followed by the first five characters plus the last character of the task name). Execute **dir uparm** to see what parameter files are in your *uparm* directory.

It is also possible for packages to have parameter files. In this case the values of the package parameters are used by all of the tasks in the package.

Listing parameters

Parameters are listed with the LPARAM task. The parent package for a task must be loaded before its parameter file can be listed. If a customized parameter file exists in the *uparm* directory then those parameter values will be listed and not the system defaults.

```
cl> lpar rfits          # list the parameters for RFITS
```

Editing parameters

Parameters can be edited with the EPARAM task. EPARAM calls up an interactive screen editor controlled by the environment variable *editor* (see §1.1.3.). In its simplest mode the user simply moves the cursor to a parameter (using either the "Return" key for downward motion, the up/down arrow keys, or possibly *Ctrl-J* or *Ctrl-K*) and types in the new value for the parameter

followed by a "Return". Upon exiting EPARAM with a **:q**, the edited parameter file will be saved in the *uparm* directory.*

We say that the parameters have been *learned*. If EPARM is exited with a **:q!** then the parameter file is left unchanged.

---

*The saved parameter file may be stored in memory for a bit before it actually gets written to the *uparm* directory, but for the most part this should be transparent to the user.

See IRAF Newsletter Number 7, June 1989, *Using and Customizing the EPARAM and EHISTORY Editors*, for some helpful hints.

---

cl> **epar rfits** # edit the parameters for RFITS

"Control" keys can also be used to exit EPARAM. An EOF that is defined at the host level, e.g., *Ctrl-Z* or *Ctrl-D*, can be used to exit EPARAM and update the parameters, while *Ctrl-C* can be used to exit EPARAM with no updating.

Parameters can also be changed by specifying the task, the parameter, and its new value explicitly. Different syntaxes are used for the different types of parameter values, in particular, boolean values are given as "yes" or "no" and string values must by quoted. The last example shows a way to query for a particular parameter value.

cl> **rfits.scale=no**
cl> **rfits.datatype="real"**
cl> **rfits.offset=100**
cl> **=rfits.datatype**

Resetting task parameters

Sometimes it is useful to UNLEARN or go back to the system default parameter values for a task. It is also possible to UNLEARN all of the tasks for a particular package including the parameter file for the package itself, if one exists. Note that what UNLEARN really does is delete the appropriate parameter files from the *uparm* directory!

cl> **unlearn rfits** # go back to system default parameters for RFITS
cl> **unlearn dataio** # go back to default parameters for all tasks in the DATAIO package, including any package parameters

Executing tasks

General syntax

Many task executions have been used in the previous sections as examples to the discussions at hand. It may not have always been clear to the user why a certain syntax was being used. We will try to outline some simple rules for task execution in this section.

It has probably already been noted by the reader that it is not always necessary to type the full name of a task or its parameters when executing that task. It is only necessary to type just enough characters so the task and parameters are uniquely defined.

The parameters are separated on the command line by spaces, so there can not be any spaces within the parameter specifications unless they are enclosed in double quotes, i.e., "".

Let us look at the parameters for the task IMSTATISTICS to use as a reference point for our current discussion.

images = Images

```
(fields = "image,npix,mean,stddev,min,max") Fields to be printed
 (lower = INDEF)        Lower cutoff for pixel values
 (upper = INDEF)        Upper cutoff for pixel values
(binwidth = 0.1)        Bin width of histogram in sigma
 (format = yes)         Format output and print column labels?
  (mode = "ql")
```

A simple way to execute IMSTATISTICS would be to use the "query" feature for the required parameters (in this case the single parameter at the top of the list that is not enclosed in parentheses). In this example the user simply types the task name, is "queried" for the required parameter, and answers accordingly. The current default values for the "hidden" parameters are used.

```
cl> imstat
Images: dev$pix
#  IMAGE      NPIX      MEAN    STDDEV       MIN       MAX
  dev$pix      262144     108.3       131.3        -1.         19936.
```

The user can use the "query" feature and also change "hidden" parameters on the command line at execution time. In this example the *format* parameter is switched to no - note the syntax! The "=yes" and "=no" part of boolean parameters can be specified with a "+" or a "-".

```
cl> imstat form-
Images: dev$pix
dev$pix  262144  108.3154  131.298  -1.  19936.
```

A task can also be executed putting all the parameters on the command line. Note that this syntax uses the "positional" feature of the required parameters in that these parameters must appear on the command line in the order that they appear in the parameter list. It is not necessary to specify the parameter name but only its value. Hidden parameters can also be specified on the command line but since they include the name of the parameter as well as its value they can appear in any order after the required parameters. Hidden parameters specified on the command line

are **not** learned; the command line values simply override the defaults for that execution of the task.

cl> **imstat dev$pix fields=mean,stddev**

In this next example we extend the long command line to a second line with the **\e** character. The break can come at the end of a parameter string, as in the example, or after a comma in the parameter string itself (in this case no preceding space is allowed).

cl> **imstat dev$pix fields=mean,stddev,min,max \e**
>>> **format-**

Of course, one can always run EPARAM on a task, modify the hidden parameters, and then execute the task while not specifying any hidden parameters on the command line.

cl> **epar imstat**
cl> **imstat dev$pix**

Another way to execute a task would be to run EPARAM on a task, modify all of the parameters including the required parameters, and then execute the task from within EPARAM with the **:go** command.

So there are many ways to execute tasks. Experience has shown that each user has his own favorite style. Many new users to IRAF find that the EPARAM technique mentioned just above is the most straightforward method for them.

Running tasks in the background

An IRAF task can be run in the background by ending the task execution command line with an **&**. The following IMSTATISTICS command will run as a background job. Since IMSTATISTICS prints information to the terminal during its execution, by default, the terminal output is redirected to a new file with the > sign

cl> **imstat dev$pix form- > statfile &**

If you do not specify all of the required parameters on the command line at execution time the CL will stop and wait for you to respond, as in the following sequence (the ">>" symbols are used to append to the *statfile* created earlier).

```
cl> imstat >> statfile &
[1] stopped waiting for parameter input
cl> service 1
Images: dev$pix
```

WARNING: Usually when this happens it is because the user **accidentally** omitted a required parameter so be sure you are being queried for the right parameter - remember position is important! It may be best to kill the job and start over.

The task JOBS will list the jobs that are in the background queue and give their current status. A background job can be killed with the command KILL plus the job number from the background queue.

```
cl> jobs
cl> kill 1
```

Background jobs submitted to UNIX hosts will continue to run even after the user has logged out of IRAF and off the machine. VMS users must submit jobs to a "batch" queue, as mentioned in the next paragraph, if they plan to log off the computer before the job has completed properly.

Background jobs can also be submitted to batch queues, if they are supported by the operating system. VMS is one such system. $ IRAF must be in the user's *login.com* file on VMS hosts in order to submit jobs to the batch queues. IRAF supports three logical batch queues - fast, batch and slow. See your local systems manager for more on your local batch queues. In the following example the task IMCOMBINE is executed with its terminal output being redirected to the file *logfile*, and the job is submitted to the VMS "batch" queue.

```
cl> imcombine <filenames> check expos+ > logfile & batch
```

Aborting tasks

Any interactive task can be aborted with *Ctrl-C* (the Control and the C key held down simultaneously). Although every effort is made to clean up properly after an abort sometimes things can be left in a weird state. It is generally good practice to type FLPRCACHE (fondly known as "flipper") after an abort. If problems occur after the initial FLPR, try one more FLPR. If problems still occur then it is probably necessary, or at least easiest, to log out of IRAF and then back in.

Generally speaking, each IRAF package is contained in a single executable file, not each separate IRAF task. When a task is executed its package executable is put into the user's "process cache". Each user can look at his own process cache with PRCACHE. When the user aborts a task it is possible for this executable to become "corrupted" and thus run incorrectly when another task from this same executable is executed. FLPR discards all executables from the process cache except the SYSTEM executable which is "locked" in at startup time. Fresh executables are then retrieved from the system for the next task executions.

It is particularly dangerous to abort tape tasks. Just be forewarned and do not be surprised if some rather strange error messages appear on your terminal screen. Try FLPR; if all else fails, logout and back into IRAF.

Backgrounds tasks should be aborted with the KILL command as mentioned in the previous section.

Data files

Since IRAF uses a *virtual* file system all file names in IRAF will appear the same regardless of the host computer, as mentioned in §1.4. IRAF does not support multiple versions of files like those on a VMS host (only the highest version numbers are used).

Legal file names can contain the character set [A-Za-z0-9_.].*

*There is a bug in VMS/IRAF that can cause problems with file names containing capital letters. VMS users should avoid file names with capital letters until this bug is fixed.

When we talk about IRAF data files it is useful to think in terms of three different types of files: text (ASCII), simple binary, and image files. The discussion of IRAF image files is left to the next section since these are files that require special attention.

There are many tasks in the SYSTEMS package for working with text and simple binary files. When working with image files it is better to use the tasks in the IMAGES package.

The DIRECTORY task can be used to list files in the current or specified directory. Wildcards can be used to search for a particular pattern.

cl> **dir** # list current directory
cl> **dir uparm** # list files in "uparm" directory
cl> **dir *cl*** # list files with "cl" in the name
cl> **dir login.??** # list files with "login." and any two characters at the end
cl> **dir *.imh** # list all IRAF image files in this directory

Text files are used in many different ways in IRAF. They can be used as input to some tasks and can be produced as output from other tasks, often as database or log files. Terminal output from a task execution can be "redirected" to a text file. Text files can be edited using the EDIT command in IRAF. Simple operations of text files (and binary files, where appropriate) are done by many tasks in the SYSTEMS package.

concatenate - Concatenate a list of files
copy - Copy a file or files (use IMCOPY for imagefiles)
count - Count the number of lines, words, characters in a text file
delete - Delete a file or files (use IMDELETE to delete imagefiles)
head - Print the first few lines of a text file

lprint - Print a file on the line printer device
match - Print all lines in a file that match a pattern
page - Page through a file
rename - Rename a file
sort - Sort a text file
tail - Print the last few lines of a file
type - Type a text file on the standard output

Simple operations can be performed on lists of data by tasks in the LISTS package and the FIELDS and JOINLINES tasks in the PROTO package. Other packages like the NOAO packages have specific tasks for handling large text files that are produced during data processing as database files.

IRAF image files

IRAF supports several image formats. Only the original IRAF image format, called the OIF format, will be discussed in this section.

The OIF format is the default image format (unless it has been changed by your site manager at installation time). Your default image format can be checked and reset, as in the following examples, and depends on the value of the environment variable *imtype*.

```
cl> show imtype        # check image format
cl> set imtype=imh     # set image format to OIF
```

Generally speaking the *imtype* variable only affects new images, i.e., those read in with RFITS or some other reader. The output images from tasks will usually inherit the *imtype* of the input

images unless explicitly set, although this may vary depending on the image format. (Again see Appendix C for further discussion about other types of image formats.)

The OIF image format consist of two files: a header file (ending in *.imh*) and a pixel file (ending in *.pix*). On UNIX hosts there is also a ".." file associated with each *.imh* file that is used to protect the image from deletion accidentally with the DELETE task (you must execute **dir a+** to see these files).*

---

*See IRAF Newsletter Number 6, February 1989, *UNIX/IRAF Image File Protection (the `..' File*

The IRAF image header

The IRAF image header file is a binary file and has the extension *.imh*. A special task IMHEADER is used to look at the header information and associated keywords. The header looks very much like a FITS header but it is actually quite different.

cl> **imheader dev$pix** # list the short version of the image header
cl> **imheader dev$pix l+ | page** # list the long version of the image header

The image header has a finite length determined by the environment variable *min_lenuserarea*. There is currently room for about 250 80-character records or lines (20000 characters).

cl> **show min_lenuserarea** # show current value
cl> **set min_lenuserarea = 40000** # set to larger value

The task HEDIT can be used to edit the header information, adding new keywords and values or modifying existing ones. The task HSELECT can be used to list selected keyword values from the image header. Try the following (the *$I* in the HSELECT command assures that the image file name is listed as well).

cl> **imcopy dev$pix newpix** # make copy of \fIdev$pix

cl> **hedit newpix newkey "string" add+** # add new keyword \fInewkey

cl> **hselect newpix $I,newkey yes** # list *newkey* value

cl> **imdelete newpix** # delete copy of image

The IRAF pixel file

The IRAF pixel file is also a binary file and has the extension *.pix*. It does not necessarily have to be stored in the same directory as the header file. When IRAF images are created the environment variable*imdir* determines where the pixel (*.pix*) files will be stored. The pathname for the *.pix* file is then written into the header (*.imh*) file so that any task operating on images will be able to find the pixel files, no matter where they are stored. The *.imh* files go into the current directory by default. In the examples below, it is important to realize that *set imdir* does not create a directory; the user will need to create any directories with the MKDIR command (except for *HDR$pixels/* - this will get created automatically by IRAF).

In the examples below the trailing slash is important!

cl> **show imdir** # show the current pixel directory

cl> **dir imdir** # list the files in this directory

cl> **set imdir=HDR$** # set the pixel directory to the current directory

cl> **set imdir=HDR$pixels/** # set the pixel directory to a subdirectory of the current directory

cl> **set imdir=**</tmp8/irafdir/> # set the pixel directory to a scratch area (UNIX host)

cl> **set imdir="**<scr2:[irafdir]>" # set the pixel directory to a scratch area (VMS host)

The *HDR$* and *HDR$pixels/* logical directories are host independent directory pathnames and will appear as such in the image header (*.imh* file). The other examples are host dependent pathnames. Note the following headers produced by executing

cl> **imhead**<imagename> **l+ u-**

- In this example *imdir* has been set to *HDR$pixels/*. The "[NO PIXEL FILE]" comment after the pixel pathname is an IRAF "feature" - if you see the *.pix* file in the *pixels* directory do not be fooled by this statement.

newpix[512,512][short]: m51 B 600s
No bad pixels, no histogram, min=unknown, max=unknown
Line storage mode, physdim [512,512], length of user area 1459 s.u.
Created Tue 14:07:03 25-Sep-90, Last modified Tue 14:07:04 25-Sep-90
Pixel file 'HDR$pixels/newpix.pix' [NO PIXEL FILE]

- In this example *imdir* has been set to a host dependent directory. If the "[NO PIXEL FILE]" statement appears with the host dependent directory name this is an indication that IRAF cannot truly find the *.pix* file. Note in this example that the node name of the host machine is included with the pixel pathname, *tucana!*<pathname>.

newpix[512,512][short]: m51 B 600s
No bad pixels, no histogram, min=unknown, max=unknown
Line storage mode, physdim [512,512], length of user area 1459 s.u.
Created Tue 14:07:03 25-Sep-90, Last modified Tue 14:07:04 25-Sep-90
Pixel file 'tucana!/tmp3/jbarnes/newpix.pix' [ok]

Pixel data types

IRAF supports several pixel data types, among these are: short (16-bit signed integers), long (32-bit signed integers), real (32-bit floating point), and double (64-bit floating point). Limited support exists for complex numbers. The task CHPIXTYPE can be used to change the pixel type of an image. The pixel data type can be determined with the IMHEADER task.

```
cl> imhead dev$pix
dev$pix[512,512][short]: m51  B  600s
```

Generally speaking the IRAF core tasks produce the same pixel type on output as that of the input image. The internal calculations may be done in reals or double if complicated computations are performed on the input data, otherwise the calculation type may also be done in the same data type as the input image. Some tasks have parameters that allow the user to specify the calculation and output pixel types. Let this be a caution to the user who has integer data: it may be advisable to convert integer data to reals before doing any image processing if disk space is not a concern, otherwise be alert to the possibility of truncation or wraparound (large negative numbers suddenly appearing in the data).

Image dimensions

IRAF supports multi-dimensional images. The size and the dimension of an image can be determined with the IMHEADER task. Using *dev$pix* as input, IMHEADER reports that this image is 2-d with 512 pixels in each dimension.

```
cl> imhead dev$pix
        dev$pix[512,512][short]: m51  B  600s
```

Although a 1-d image can be represented by a 2-d image with one row/line or column, it can also be represented by a vector with just one dimension. This has been a popular format for 1-d data in the NOAO spectroscopic packages.

```
cl> imhead oned
        oned[820][real]: FEIGE 98
```

An "image section", can be used along with the task IMCOPY to reduce the dimensionality of an image. For example, the output of this task execution will be a 1-d vector image.

```
cl> imhead dev$pix
        dev$pix[512,512][short]: m51  B  600s
cl> imcopy dev$pix[*,20] oned
cl> imhead oned
        oned[512][short]: m51  B  600s
```

General image tools

A few general image tools will be highlighted in this section. It is important to use these tools rather than the similar tools for text and simple binary files provided in the SYSTEMS package since the image tools must deal with a more complicated file structure (two files* rather than one).

---

*UNIX hosts really have three files associated with each IRAF image since there is also the ".." protection file.

Three important tasks for image handling are IMCOPY, INRENAME, and IMDELETE. IMCOPY makes a new copy of the input image. IMRENAME will rename the current image. And IMDELETE will delete images. Using the distributed image of M51, try the following,

```
cl> imcopy dev$pix newpix        # make another copy of an image
```

cl> **imrename newpix newcop** # rename an image
cl> **imdelete newcop** # delete an image

The IMRENAME task has another useful feature besides simply renaming the image. It also moves pixel files to the current directory specified by the environment variable *imdir*. Thus whole directories of pixel files can be moved using this task, and the image header will be updated properly to point to this new directory. (Problems can arise with disconnected pixel files if these files are moved with host level facilities. It is not necessary to actually rename or move the headers to do this.

cl> **set imdir=**</tmp3/irafdir/> # set *imdir*
cl> **imrename *.imh .** # move pixel files to new *imdir*

The pixel values can be listed with the task LISTPIXELS. Note the use of "image sections" in these examples. Again, we use M51 as our input image. In the second example the output of LISTPIXELS is "piped" through to the input of the TABLE task (in the LISTS package) for reformatting before being displayed on the terminal.

cl> **listpix dev$pix[20:25,30:32]** # list the pixel values for an image section
cl> **listpix dev$pix[20:25,30:32] | table** # list and reformat

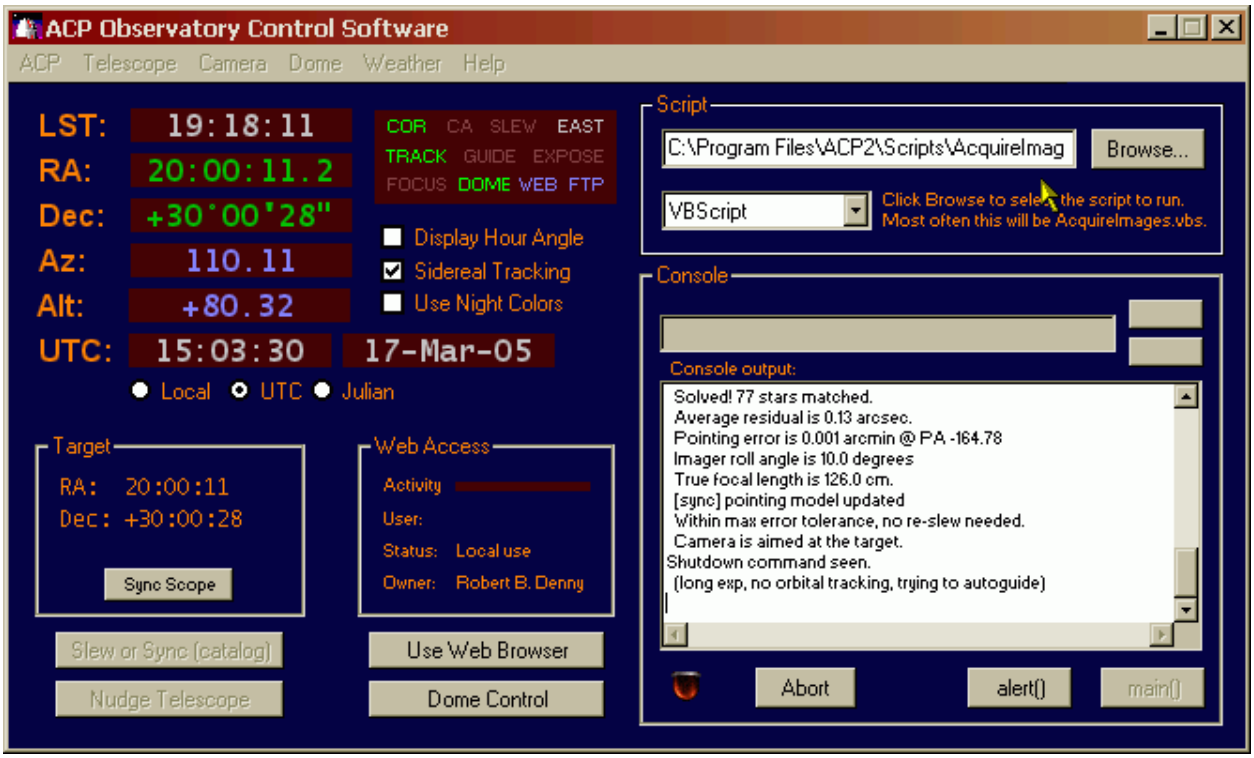
There are other useful tools for dealing with images in the IMAGES package as well as in the PROTO package. See Appendix A for a complete listing of the tasks in these packages for the V2.10 release.

4.3 ACP Automation

We had installed rts2 software which is linux based. It has many difficulties to use, to use rts2 one has to posses the basic knowledge of linux operating system. The installation system was not as simple as windows based software. The graphical user interface (GUI) of rts2 is not very sophisticated. In other words rts2 was too complicated to understand and to operate by the undergraduate students. Hence we had decided to move to more sophisticated software which has friendly GUI and which is easy to handle. ACP Observatory control Software was the solution of the problem

What is ACP

ACP Observatory Control Software, available from DC-3 Dreams S.P., is designed for automating and remotely operating observatories. It uses MaxIm DL for all camera control operations. ACP provides completely automated scripted control, allowing hundreds of images to be taken a night without human intervention. An optional built-in web server is available for internet remote control.



How does ACP Works?

ACP and other programs work together. Each program is useful in its own right, but when operated together by ACP, they form a system for acquiring images that blows away the competition. ACP provides the automation hub for the observatory. It sequences the observations and controls the telescope, CCD imager, filter wheel, auto-guider, focuser, and dome.

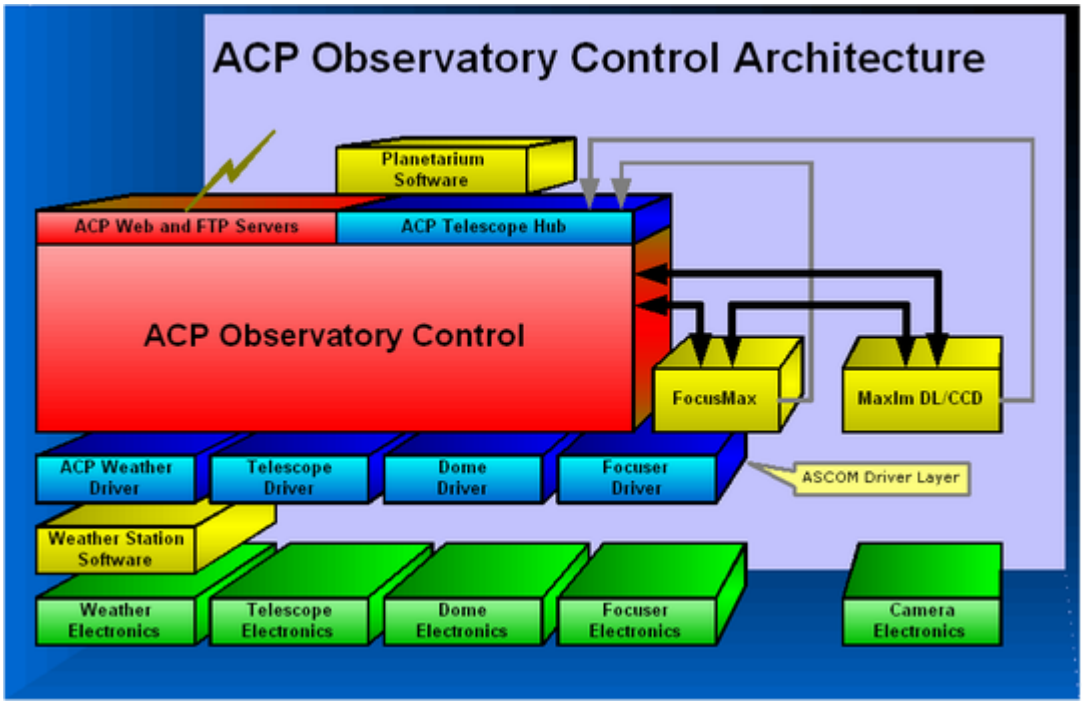


Fig4.3.1

Most GOTO mounts have imperfect pointing. ACP's unique Every Image Centered provides both self-learning model-based pointing correction and a "digital finder scope" function to solve this problem. While the pointing corrector learns about your mount's error, ACP ensures that every image places your target in the exact center.

From the first image you take, everything is centered! It learns by matching image stars against a catalog. This provides "closed-loop" control, ensuring that the telescope is always precisely on target no matter what. As it learns about your telescope's mechanical behavior, it forms a model which is used to correct pointing. Eventually, ACP learns enough to virtually eliminate last-minute centering adjustments. All of this is done automatically. No need to separately gather

mapping points, run an analysis program, etc. You just go to work and ACP gets better as time goes on.

MaxIm DL/CCD provides scriptable controls for the camera, filter wheel, and the auto guider. ACP's scripts use these controls.

FocusMax ties into the focuser, telescope, and CCD imager via ASCOM, sharing devices as needed. It offers an AutoFocus function that ACP uses to do automated autofocus. ACP monitors the Half Flux Diameter of every image it takes, and can optionally refocus if the image quality degrades. Or you can tell ACP to refocus periodically and/or after filter changes. You can also set up a table of filter focus offsets, eliminating between-filter auto-focus runs.

For advanced users, *nothing else* has the flexibility and power of this system. The huge advantage is that you can fully customize the entire system. You can edit the web pages. You can edit the control scripts. You can add image processing features, integrate in other hardware (e.g. dome, weather station), or whatever else you can think of.

With this combination, you get automatic image calibration, pointing corrections (or TPOINT usage), and auto-guiding without any intervention on your part. A detailed log is produced, allowing you to diagnose problems and tune your system for best operation. Images can be automatically flat and dark corrected, plus astrometry info in the form of WCS coordinates are added to the FITS headers. In short, this is a professional-class automation facility!

Installation of ACP

Software and Catalog Installation

For operating a robotic observatory, these are the applications you need to have installed before you run ACP for the first time. Do not uninstall and reinstall ACP if you encounter a problem! This is almost always a waste of time, and may obscure the real issue or create new problems.

ASCOM

The ASCOM Platform 5 (or later) must be installed on your computer. The latest version of the

ASCOM Platform is available for download at no charge from the ASCOM Initiative Web Site in the downloads section.

MaxIm DL

MaxIm DL/CCD V5.22 (or later) must be installed on computer. It is available from Diffraction Limited.

FocusMax

Automated observing needs automatic focusing. ACP has a built-in auto-focus feature that relies on FocusMax. If you have a focuser that can be used with FocusMax, **you** need FocusMax 3.7.0.36 or later for proper operation. If FocusMax isn't installed, get the installer from the FocusMax Autofocus Software web site and install it. If FocusMax is installed but the version is too old, get the latest FocusMax executable. Unzip and replace your existing FocusMax.exe with this one. Don't try to use ACP's auto-focus without the proper version.

PinPoint Reference Catalog - GSC or USNO A2.0

The PinPoint Astrometric Engine (included with ACP) requires either the Guide Star Catalog 1.1 or the USNO A2.0 catalog for star references. The GSC is 350MB in size; the A2.0 is over 6,000MB (6GB) in size. If your imager has a field of view smaller than 10 arc minutes, you'll want to use the USNO A2.0 for best reliability. Make sure one or both of these catalogs are installed. Start with the GSC.

Installing the GSC

The GSC Installer program that comes with ACP is available on the Start menu under ACP Observatory Control. This program will optionally copy the GSC from CD to your hard drive, and automatically configure ACP and MaxIm DL with the correct folder setting for the catalog. If you have the full PinPoint product installed, it can also configure Visual PinPoint with the correct catalog path. It also does a brief integrity check of the catalog (mostly to catch bad downloads).

a. Start the GSC Installer (Start menu, under ACP Observatory Control)

b. Turn on Configure ACP and MaxIm DL

c. Under Catalog Location, choose Install from CD-ROM (recommended) or Use from CD-ROM

d. Select the CD drive containing the ACP CD-ROM

e. Click "Setup GSC Catalog"

When the process completes, both ACP and MaxIm DL are properly configured and the catalog's integrity has been verified.

Guide Star Catalog:

As of October 2003, the STScI GSC in its original format (FITS tables) is no longer available from NASA Astronomical Data Center. In order to support you, DC3-Dreams has made the GSC Catalog available online.

Site Information

You need to know the geodetic position (lat/long) of your observatory. If you are lucky enough to own a GPS unit, this is a snap. Otherwise, use *Google Earth* or any of the geo-coding services on the net. Of course, you can use paper maps as well.

Your computer must be correctly set for *your* time zone, and whether or not you use daylight time. You should have set this up on your PC prior to installation of ACP. If not, you must do this now. Do not force UTC time on your computer!

- The accuracy of your PC clock is important, particularly if you want accurate "go to" and pointing. For the most in accuracy, you can use one of the 3rd party time sync apps to keep your PC clock locked to the USNO or NIST standard clocks. One such is Rob Chambers' Dimension **4**, which runs in the background and is very small. I have used it for years and never had a lick of trouble. Another time-setting program is AboutTime. If you are in the field, you can use a GPS for both position and time. See also PC Clock Slows Down.

There are too many parameters to be considered while installing ACP which can be seen in the following link

http://solo.dc3.com/ar/ACPRefGuide.html

4.4 Maxim DL

MaxIm DL is specifically designed for astronomical imaging and other low-light level applications. It operates scientific-grade CCD cameras, DSLR cameras, low-cost CCD imagers, and webcams and other video sources. It also controls filter wheels, focusers, autoguiders, telescopes, focal plane rotators, and observatory domes. A wide variety of hardware is supported through both ASCOM drivers http://ascom-standards.org and proprietary and third-party plug-in drivers. MaxIm DL also includes a wide variety of image processing and analysis features.

Installation of Maxim DL

System Requirements

Cyanogen Imaging MaxIm DL works on Windows operating system

MaxIm DL is compatible with both 32-bit and 64-bit versions of the above operating systems.

The following hardware is required:

- Recommended minimum memory size is 1 GB or larger. Processing larger images or opening multiple images simultaneously will require correspondingly larger memory. 2 GB memory is recommended for processing large arrays including DSLR images and CCD images larger than 6 megapixel.
- Disk Space – 100 MB for program installation
- Video Display – 1024x768, 16-bit color or higher.
- Mouse
- Internet Explorer

Internet Installation

Download the file onto your hard drive; usually you would put the file in a temporary folder or your desktop. You may wish to keep the file in order to reinstall the software in the future, if you have a problem with your computer. Double-click the installer and follow the instructions.

If you purchased the box software with manual and CD-ROM from our webstore, you will still receive the license email and can install the software immediately via download.

Updating to New Releases

Minor updates are released as required. To download updates, open MaxIm DL and select Help menu Check for Updates. Web browser will open to a special update page. This page will inform whether a new update is available.

The purpose of the upgrade subscription is to permit continuous product improvements, providing new capabilities to MaxIm DL users. Minor updates are not simply "bug fixes", as almost every minor update includes new features. Upgrade subscription will also apply to a major upgrade, if it is released prior to expiry date.

5 Observatory and Lab Set Up

5.1 Observatory:

An astronomical observatory is any structure containing telescopes and auxiliary instruments with which to observe celestial objects. Observatories can be classified on the basis of the part of the electromagnetic spectrum in which they are designed to observe. The largest numbers of observatories are optical; i.e., they are equipped to observe in and near the region of the spectrum visible to the human eye. Some other observatories are instrumented to detect cosmic emitters of radio waves, while still others called satellite observatories are Earth satellites that carry special telescopes and detectors to study celestial sources of such forms of high-energy radiation as gamma rays and X-rays from high above the atmosphere.

Optical observatories have a long history. The predecessors of astronomical observatories were monolithic structures that tracked the positions of the Sun, Moon, and other celestial bodies for timekeeping purposes. The most famous of these ancient structures is Stonehenge, constructed in England over the period from 3000 to 1520 bce. At about the same time, astrologer-priests in Babylonia observed the motions of the Sun, Moon, and planets from atop their terraced towers known as ziggurats. No astronomical instruments appear to have been used.

Indian astronomy has a long history stretching from pre-historic to modern times. Some of the earliest roots of Indian astronomy can be dated to the period of Indus Valley Civilization or earlier. Astronomy later developed as a discipline of Vedanga or one of the "auxiliary disciplines" associated with the study of the Vedas, dating 1500 BCE or older. The oldest known text is the Vedanga Jyotisha, dated to 1400–1200 BCE (with the extant form possibly from 700–600 BCE). Indian astronomy flowered in the 5th-6th century, with Aryabhata, whose Aryabhatiya represented the pinnacle of astronomical knowledge at the time.

There are different types of Observatories depending upon their purpose.

1. Ground Base Observatories
    i) Radio Observatory

ii) Solar Observatory

iii) Optical Observatory

(1) Portable Set Up (Roll Off)

(2) Fix Mounting

(3) Manual Rotating Dome

(4) Fully Motorized Dome

5.2 Portable Set Up (Roll Off)

An Observatory is any fixed site from which observations ae made. Hence we should first consider the cas where there are no permanent fixtures. Even if the mounting and telescope remains portable, there are certain measures which can be taken to make observing more convenient and productive, if it is regularly carried out from a fixed spot.

One of these is to have some kind of mark or marks on the ground indicating how the mount is to be set up for correct polar alignment, avoiding having to go through a time-consuming polar alignment procedure every night. With proper marking on ground we can predetermine the position of telescope mount. Most tripods and mounts, these need only be approximately accurate – find tuning of the polar azimuth alignment can later be made on the mounting itself.

We can keep the telescope and mount at some fixed place all the time and we can only move the covering of the telescope. When it is not an observing period we can shift the telescope back at safe house. This method is quite cheap and easy to implement as the roll off shed can be manufactured locally. Hence we have adopted this and we have set up a portable roll off observatory at the top of the Junior College Building. Design And Aspects are in discussed separately in Shed Design.

Shed constructed from College grants

Fix Mounting

A halfway-house to a roofed observatory is to have a permanently-fixed telescope pier and mounting outside, and a removable telescope, stored inside or in an outbuilding. There are distinct advantages to setting-up a permanent metal or concrete pier to carry the mounting, on proper foundations dug into the ground or on some rigid concrete surface. The labor of carrying around the heaviest parts of the telescope, the mounting and counterweights is removed and there is no necessity for repeated polar alignment or leveling of the mount. The challenge here is how to protect the mounting from corrosion and other damages when it is kept outside. To overcome this flexible scope cover are there in market or we can use the powder coated paint which withstand for the longer duration.

The next upgrade for the Fergusson College Observatory Roll Off Shed is to equip it with a good pier.

Manual Rotating Dome

In this type of Observatory there is an upper section that rotates, and walls that are fixed. The upper section classically is a dome, sliglty more than 50% of a sphere, but it can be a cylinder, an octagonal pointed shape, or some other complicated thing. The necessity of the Dome is described in the separate chapter.

This type of observatory in-house all things telescope, mounting, detectors and computer. Generally the Data processing Lab is just under the dome or in the dome itself which allow observer to collect the date locally.

The upper part of the dome is rotated with hand i.e. manual operation. In this type generally wide slit is desirable, as this means less frequent dome rotation. As slit is wider the dome behaves more like a run-off roof observatory. Protection from vibration due to wind is the job of dome, wider slit violet the condition. Scattered light is also the problem with this arrangement.

Fully Motorized Dome

In this type of observatory every rotating part is controlled from computer the slit and the dome itself. We can synchronize the motion of the dome and slit with telescope mount hence all the time we get good and sound data. Due to this the size of the slit can be reduced since it is always aligned with the telescope.

Due to fully automated/ motorized dome we can make the observatory control from remote places since the manpower needed to control all the things simultaneously is reduced drastically and in case of emergency we can terminate the observation session remotely and close the dome and slit.

It mandatory that if the observatory is fully motorized then it should have some other sensors like weather monitor, cloud sensor, humidity sensor, temperature sensor in order to make the remote observation.

Ground-based observatories

Ground-based observatories, located on the surface of Earth, are used to make observations in the radio and visible light portions of the electromagnetic spectrum. Most optical telescopes are housed within a dome or similar structure, to protect the delicate instruments from the elements. Telescope domes have a slit or other opening in the roof that can be opened during observing, and closed when the telescope is not in use. In most cases, the entire upper portion of the telescope dome can be rotated to allow the instrument to observe different sections of the night sky. Radio telescopes usually do not have domes.

For optical telescopes, most ground-based observatories are located far from major centers of population, to avoid the effects of light pollution. The ideal locations for modern observatories are sites that have dark skies, a large percentage of clear nights per year, dry air, and are at high elevations. At high elevations, the Earth's atmosphere is thinner thereby minimizing the effects of atmospheric turbulence and resulting in better astronomical "seeing". Sites that meet the above criteria are near to perfect to make optical observation from ground which leads us to next challenge i.e. selection of site for Fergusson College Observatory (FCO)

Observatory Site

The selection of the observatory site was the most important parameter of the setting an Observatory. The prospective observatory builder may have very little choice over his or her observatory site, or a lot, depending on how much space he has available. A rooftop may seem to be an ideal location, with good, unobstructed views, but in fact, in most locations, this situation will also expose the observatory to the maximum of wind and the maximum of local artificial lighting. In general the region of the sky below 20° altitude is not much use for observing anything it suffers too much from atmospheric absorption and refraction and turbulence. Obstruction of the 20° nearest the horizon is not much of a loss, and will be a good thing in many locations, where there are streetlights and house lights to contend with. True, interesting phenomena often do occur low down, but they tend to be the things at which there is little point aiming a large telescope.

Light pollution is clearly the most important problem facing the 21st century astronomer, and here it might be worthwhile saying something more about dealing with light pollution at source. You do not necessarily have to put up with the fact that your observing site is flooded with artificial light, and you do not necessarily have to move. Sources of artificial light can be categorized into private, i.e. un-curtained windows, security floodlights, etc., and public, i.e. streetlights and illumination of sports pitches and the like. As Fergusson College is in the center of the city it is very difficult to find the light pollution free site within the campus. Initially we have thought to overcome this problem by using the halogen filter but practically it is not possible.

Though there is a good aspect, as the area surrounded by Fergusson College is majorly commercial hence after 10pm almost each surrounded building switch off their light. At the north there is Khikee Armament Area (Range Hill) which is light free zone, East site there is hill which blocks sufficient light, the south side the campus buildings help to prevent some scattered light. Problem is with east site as it is FC road and on the east site main city is situated hence there is sufficient amount of light pollution from that side.

Light pollution is not the only problem; the site location should not be too exposing, since there are vibration due to wind. Atmospheric scintillations and airmass are one of the issues but as far as Fergusson College Observatory is considered we are going to use telescope within the range of 6" to 14" hence scintillations doesn't play much role.

Fergusson College is now recognized as Heritage from Govt. of India hence we had some limitation while selecting site as no permanent new infrastructure is allowed over some buildings of the college. Also we cannot modify the existing structure.

The most important parameter in selecting site is Security. There had been several unpleasant incidents in College where burglary and riot were occurred few times. Hence we had to be sure that our observatory site should be protected well enough.

The Parameters Considered While Selecting Site:

1) Access
2) Light Pollution
3) Sky Exposure
4) Security
5) Exposure

Considering all these parameter the scientific site surve was conducted to select site for the observatory and report had been submitted to committee. We had identified few sites in the campus and from each site we had grab the images for longer exposure with Canon D5 Camera. We had taken the images at three different times (8pm, 11pm and 2am) at the new moon night. Three different sets were taken (pre new moon night, new moon night and post new moon night). Our aim was to get the amount of light pollution and to get the fraction of the city light presented in the images. The camera was pointed directly to the zenith and the 30 sec. exposure was given. By analyzing the data we had found some difference from the different site but the value was not physically significant to be taken into consideration hence we prepared the report with considering rest of the parameters.

Identified Proposed Sites for an Observatory

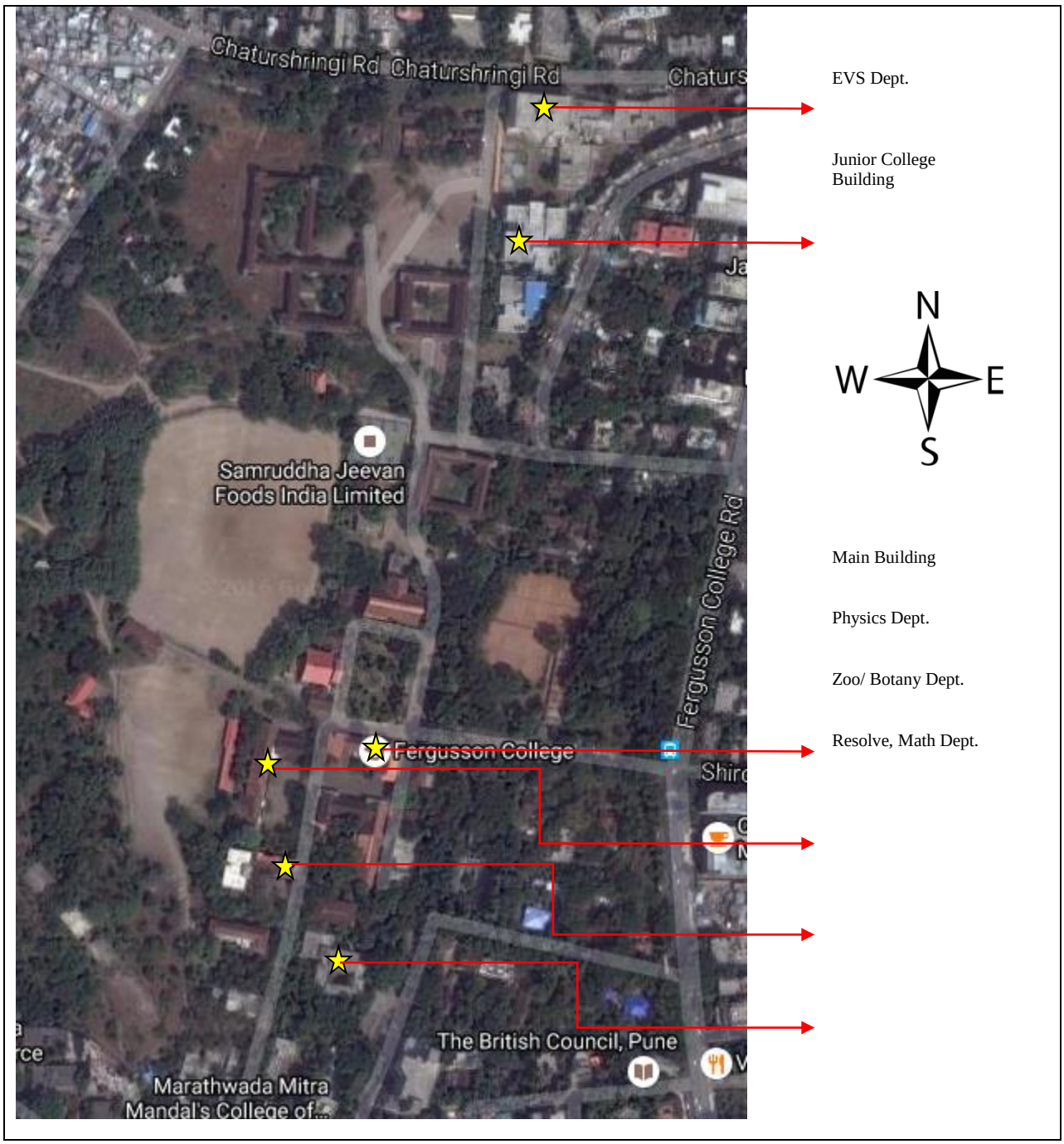


Fig. Map showing the possible identified sites for the Observatory location (Courtesy Google maps)

Observatory Site Report

<table>
<tr><td colspan="3"><u>Site A : Above the Resolve building</u></td></tr>
<tr><td>Pros</td><td>Cons</td><td>How to resolve the problem</td></tr>
<tr><td rowspan="3">1. Already have one dedicated room for telescope
2. Not too high from ground</td><td>1. The terrace area is covered by several trees hence it is very difficult to locate the pole star</td><td>We shall need to chop the trees every year</td></tr>
<tr><td>1. The roof of Recreation hall is an obstacle</td><td>We can make 10ft platform</td></tr>
<tr><td>2. In future computer department may expand their infrastructure at that time we shall need to move the observatory.</td><td>No solution</td></tr>
<tr><td colspan="3"><u>Site B : Above Botany and Zoology department</u></td></tr>
<tr><td>Pros</td><td>Cons</td><td>How to resolve the problem</td></tr>
<tr><td rowspan="3">1. An antique telescope with dome is already established
2. Near to Physics department</td><td>1. Due to present telescope room the southern sky cannot be observed</td><td>No solution : We cannot move the old telescope</td></tr>
<tr><td>2. Need to get permission from two (Botany and Zoology) departments to get access to the roof</td><td>We can make the ladder separately which would not disturb both departments but we will be compromising with security</td></tr>
<tr><td>3. Security is the main issue (in the past some instrument were stolen from same room)</td><td>Need security for all time
365 × 7 × 24</td></tr>
<tr><td colspan="3"><u>Site C : Above Physics Department</u></td></tr>
<tr><td>Pros</td><td>Cons</td><td>How to resolve the problem</td></tr>
</table>

1. Telescope will be in Physics department 2. Easy access to roof 3. Will be always surrounded by teachers	1. The structure of Physics dept and Amphitheatre make the pole star invisible	We can build the 20 ft Platform
	2. Security is an issue	Need security for all time 365 × 7 × 24
	3. Very compact space to carry instruments	We can make another staircase from backside of the department
Site D : Environmental Science Department		
Pros	Cons	How to resolve the problem
1. This Dept. is already running weather station which will help us to collect the atmospheric Data	1. Near to Dnyaneshwar Paduka Chowk (Too Much Light Polution)	No Solution
Site E: Junior College Building		
Pros	Cons	How to resolve the problem
1. Good Area to set up an observatory and other instruments 2. Highly secure premises 3. North star is visible 4. Ready Platform for Dome Erection 5. Ideal for Night Observation	None	None
Site F : Above Main Building		
Pros	Cons	How to resolve the problem

1. Pole star is easily visible and maximum sky can be observed compare to all other sites 2. Easy to access 3. As it is in main building, no issue of security 4. Audio Visual hall is great advantage for public outrage 5. Will add beauty to heritage structure 6. Ideal for night observations	1. Staircase is very small	We can replace the smaller staircase with larger one.

Rating wise Result

Site	Accessibility	Security	Observational Conditions	Upgradable	Result
A	☆☆☆	☆☆☆☆	☆☆☆	☆☆	☆ X 12
B	☆☆	☆☆☆	☆☆	☆☆	☆ X 9
C	☆☆☆☆	☆☆	☆☆☆	☆☆	☆ X 11
D	☆☆	☆☆☆☆	☆☆	☆☆☆☆	☆ X 12
E	☆☆☆☆	☆☆☆☆☆	☆☆☆☆	☆☆☆☆	☆ X 17
F	☆☆☆☆	☆☆☆☆☆	☆☆☆☆	☆☆	☆ X 15

Rating Particulars

Accessibility	It includes possession of keys, permission from Dept. and Infrastructure to access the site
Security	It includes dedicated staff for security, access to site, CCTV and exposure to public

Observational Conditions	It includes the astronomical aspect as well as logistics, Comfort of observer during night stay, Infrastructure for night stay.
Upgradable	It includes the possibility to expand the observatory in future, Erection of permanent structures and available area.

Site E means Above the Junior College building is the most obvious choice since it is well maintain and secure than any other building. There are two permanent security guards at the entry and each floor is covered with the security CCTV camera. There is enough places available on the terrace for the future development. And as it is new building, apart from MNC permission there are no other restrictions for permanent infrastructure development.

The area over the staircases is ideal for the motorized dome to be mounted since the pole star is visible from here and it is well covered from city light. Just below it we can construct he the data processing lab. Hence we have selected this particular site for Fergusson College Observatory.

Proposed Site

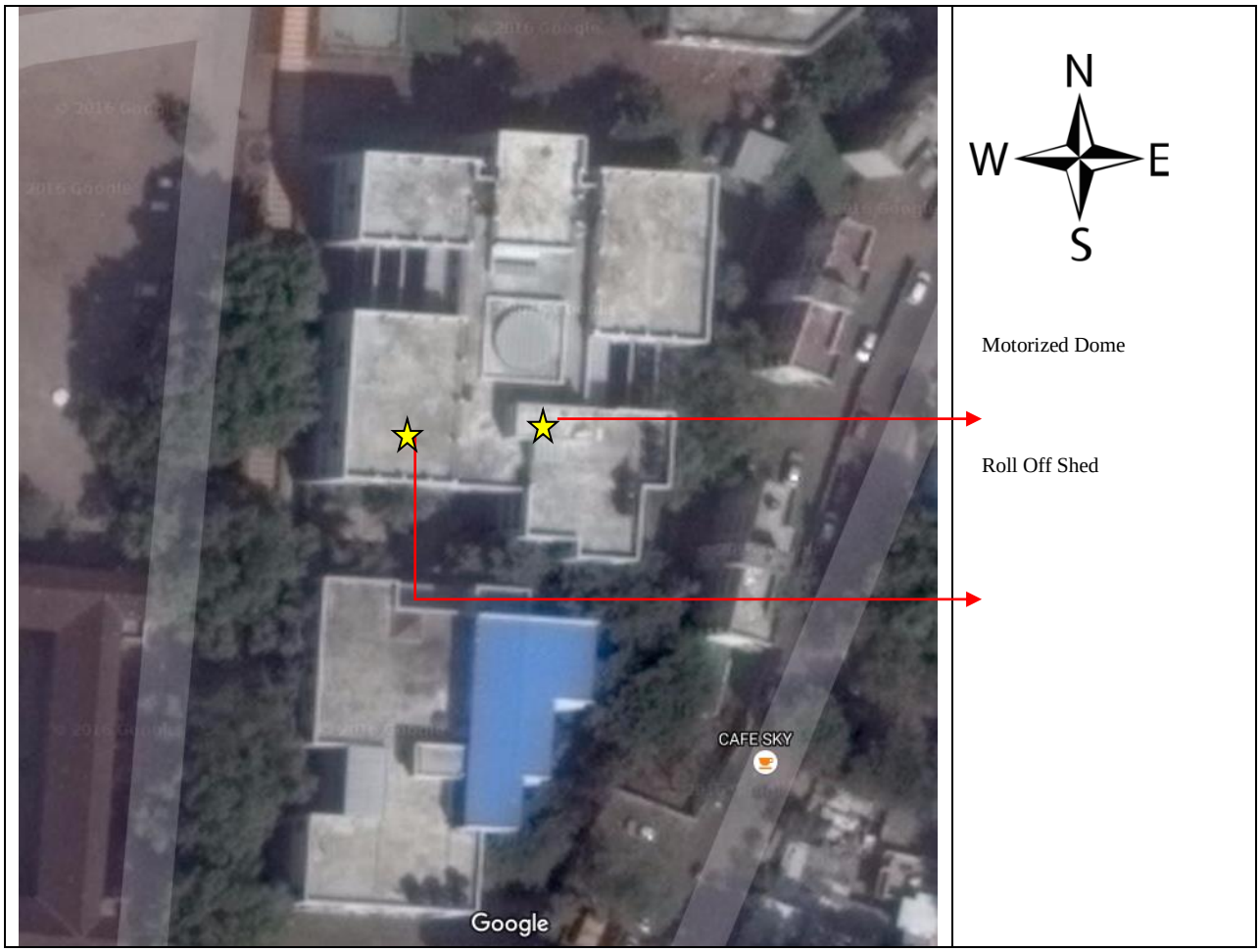


Fig. Proposed site for the Fergusson College Observatory (Courtesy Google Map)

5.3 Roll Off Shed and Dome

When the project was started Fergusson College was supposed to built or purchase the dome. For the same purpose we had visited different vendors and collected designs and quotations from them (Vendor Listed in Appendix). We are supposed to have two different telescope in near future. We already have Celestron 9.25" and we are getting some grants to purchase another telescope. Most likely we will be going with Celestron 14". Hence we need to built two housing for the telescopes. As the part of ISRO project we decide to built a roll off shed for Celestron 9.25" telescope. And we had procured grants from Member of Parliament to procure the Dome.

Roll Off Shed

We had chosen this design because of its simplicity, cost effective, locally producible, easy to install and low maintenance cost. The design was made and quotations were called up from different vendors. As per the Local Purchase Committee decision we gave purchase order to Mr. Hardikar. The shed was manufactured in Pune and installed in 6 working days.

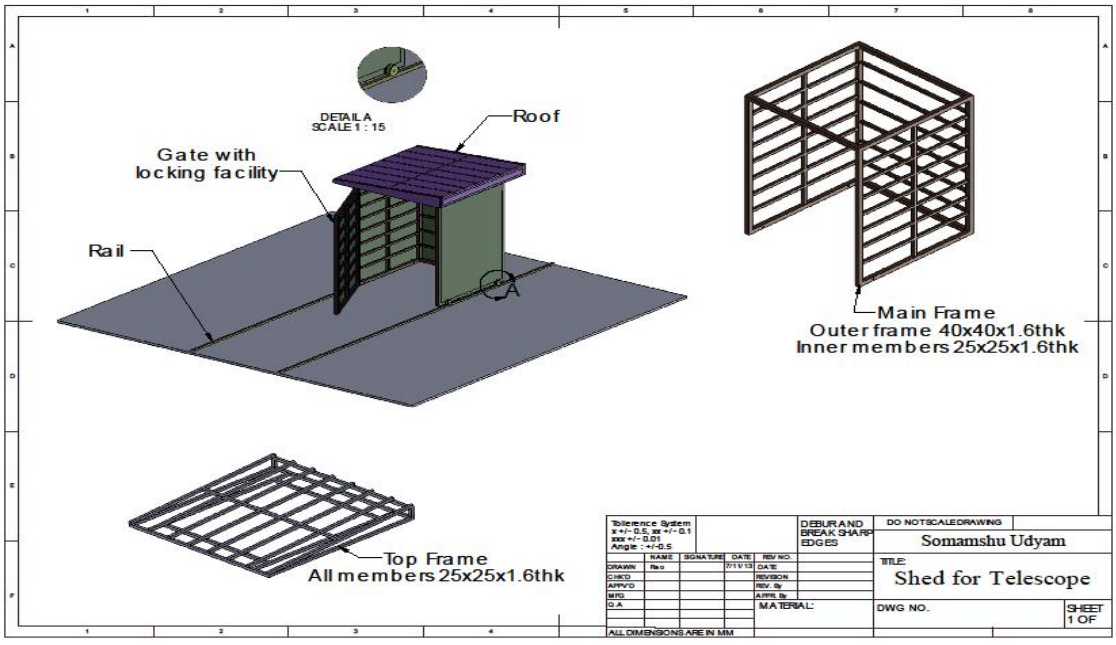


Fig.5.1

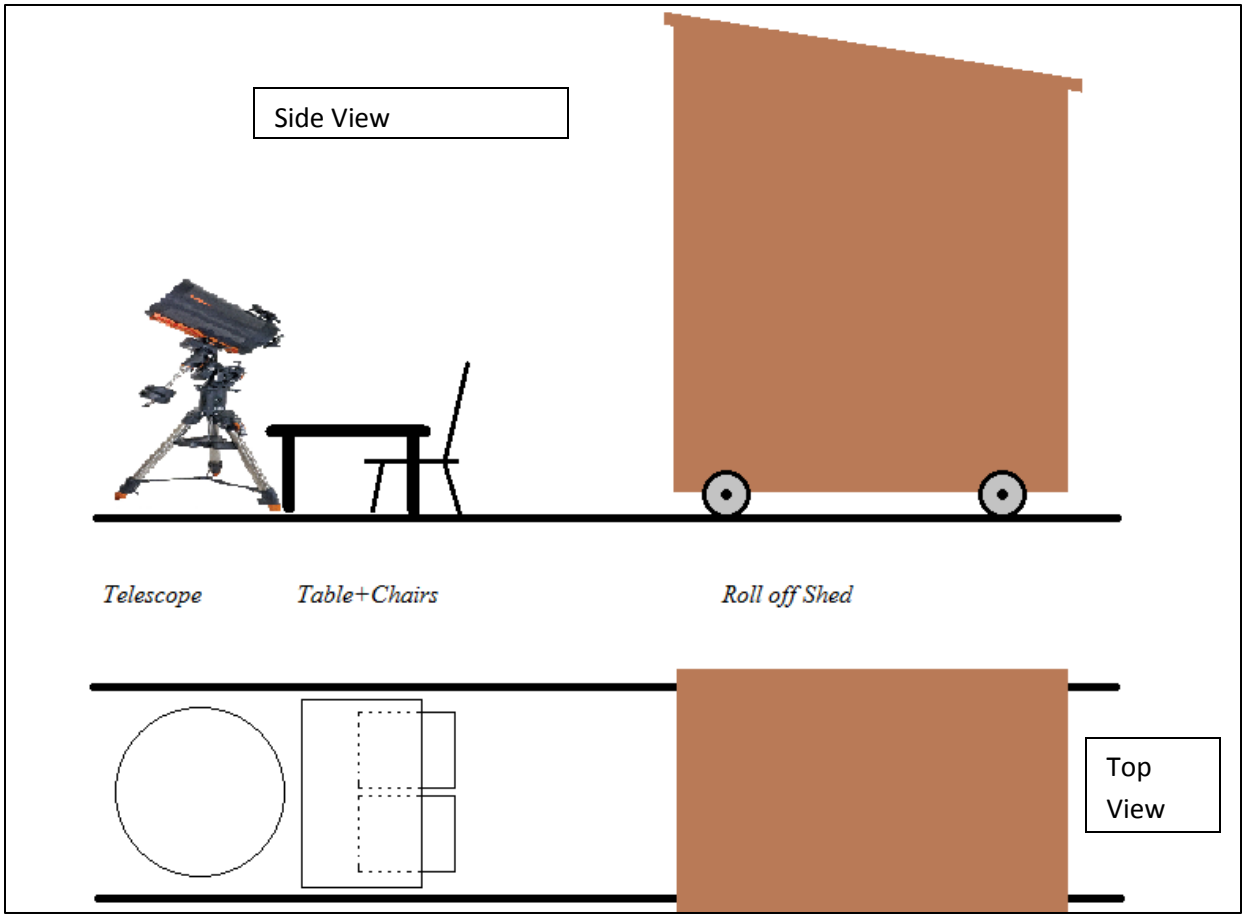
Design

The assembly includes three fixed wall and two doors attached at one of the sides. The entire assembly is rested on guided tracks with the help of grooved wheels. It is to make sure whenever we are sliding the shed it should follow the particular path in such way that it should disturb the telescope and observational set up. The another aspect was as we are using grooved wheels and tracks the friction is considerably less and hence a single student can handle this assembly alone.

The face of door is towards north since we are using the equatorial mount and its home position is facing south hence the design allows more room in this way. The material used for this assembly is powder coated to assure its long endurance. The internal design (matrix like structure) is made in such way that it can be covered with Styrofoam substance which acts as a heat insulator. There is an arrangement to install the ip camera and few sensors in future if needed. There is also an arrangement for electric points for charging of the power bank and other purposes.

The roll off shed has the track locking system. At the either ends of the track the locking groove are provided to make sure it doesn't move while the observation or when it is locked. Separate lock is provided at the door. As the shed is not closed from the bottom we had limitations on size of it. It is good enough to house a telescope, a working table and two chairs.

The shed has been tested in last monsoon and it was functioning as expected from design. The shed is mounted on the little slope which allows water to flow easily. Tracks are installed in the direction of the slope hence it don't block the flow of the water. In rainy season we kept telescope and other instrument in the Lab for the precaution but the shed provides great protection from sudden raining. At one of the incidents we had all our instruments and telescope inside shed when it started raining. The rain was continued another 3 days and there no sign of water leakage inside of the shed.



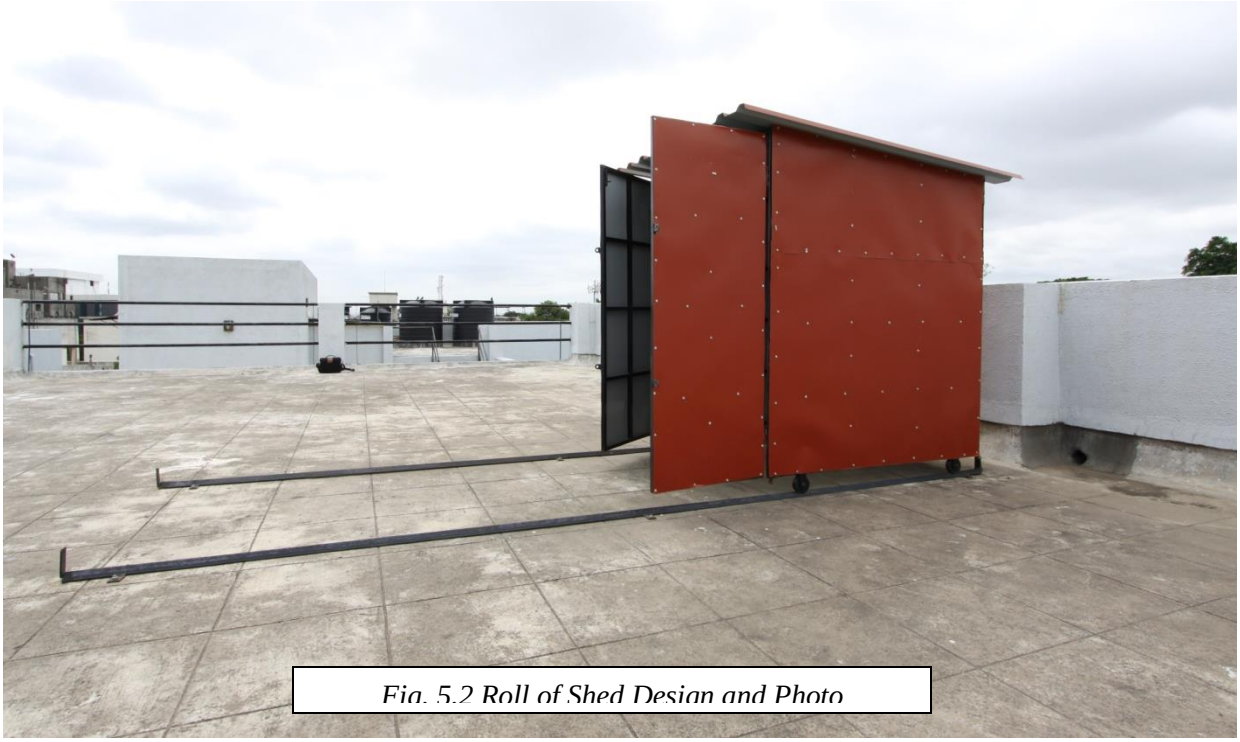


Fig. 5.2 Roll of Shed Design and Photo

5.3 Motorized Dome

When the idea of Robotic observatory came, it was very oblivious it should be controlled remotely. Manual operated dome was not the option. We had to make the decision whether to manufacture a motorized dome in India or to purchase the dome readymade from abroad.

We worked on both things for couple of months. We had identified many vendors and manufactures from India and visited many automation companies for the solution. Mr. Saraf was one of the persons who was interested in making the dome motorized. The initial idea was to manufacture dome from either Mr. Soman or Mr. Parag Mahajani and then get motorized from Mr. Saraf.

Mr. Shailen Agrahari is the professional dome manufacturer in India and we had discussed with him on this project. We had asked quotations from him as well. There was some shortfall in his design like the weight of the dome, incompatibility with new software and the high maintenance cost. As we wanted to set up an observatory over the roof, the weight of the dome was the issue. It was not an option to install a metal dome on the roof of the junior college building.

Sr. No	Vendor	Dome Specifications	Merritt / Demerit
1.	Amalendu N. Soman Pune	3 M Semi Automatic FRP Material	• No interface with new system • Not Upgradable • High cost
2.	Shailen Aggrahari	3 M Manual 3 M Motorized Metal	• Fully metal body • Too heavy • Maintenance cost is higher • Difficult to Install • Too Expencive
3.	Mr.Parag Mahajani	3 M Manual FRP Material	• Not Motorized • Leakage problem

As we wanted a light weight fully motorized and upgradable dome, we try and look for vendors from abroad, surprisingly we found more cost effective products.

We had called quotations from vendors located abroad. The list of all vendors is in Appendix. After the careful study of all parameter and ability to ship it to India our Local Purchase Committee had finalized the Scope Dome from Dublin and had placed the purchase order. We have received the dome and we are now in the process to mount it on the proposed site. The technical details of the dome are as follows.

Specifications

Diameter of the dome:	3000 mm
Shutter width:	1000 mm
Dome height:	2400 mm
Diameter of the base:	2760 mm
Weight (with pallet):	~350 kg

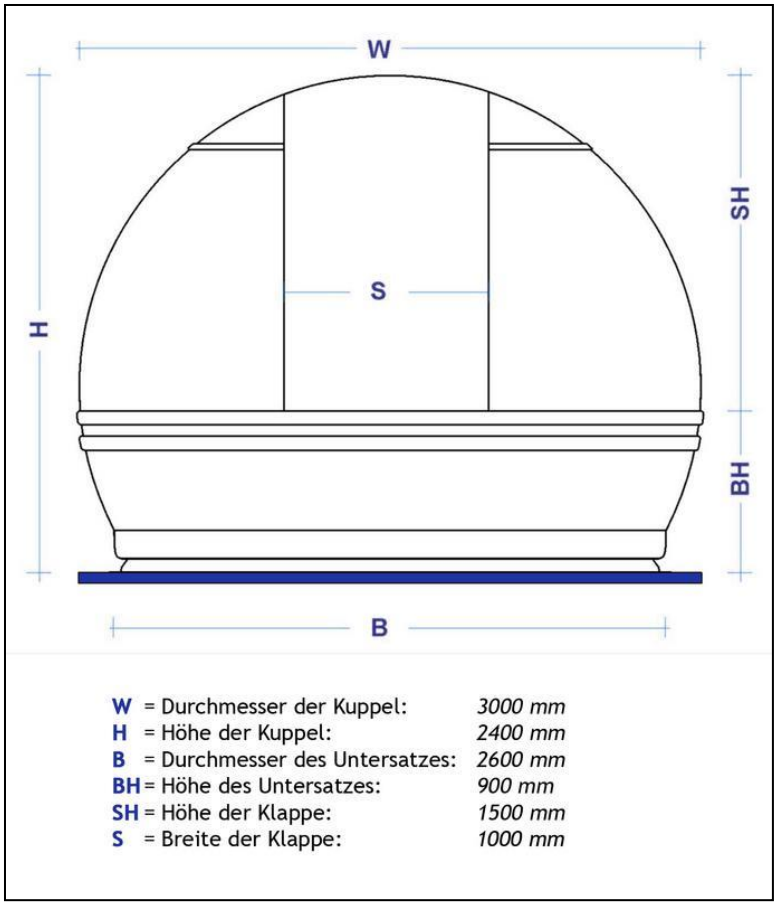


Fig. 5.3 Dome Specs

Fig. 5.4 M Fully Motorized Scope Dome with Door Assembly Purchased from Member of Parliament grant.

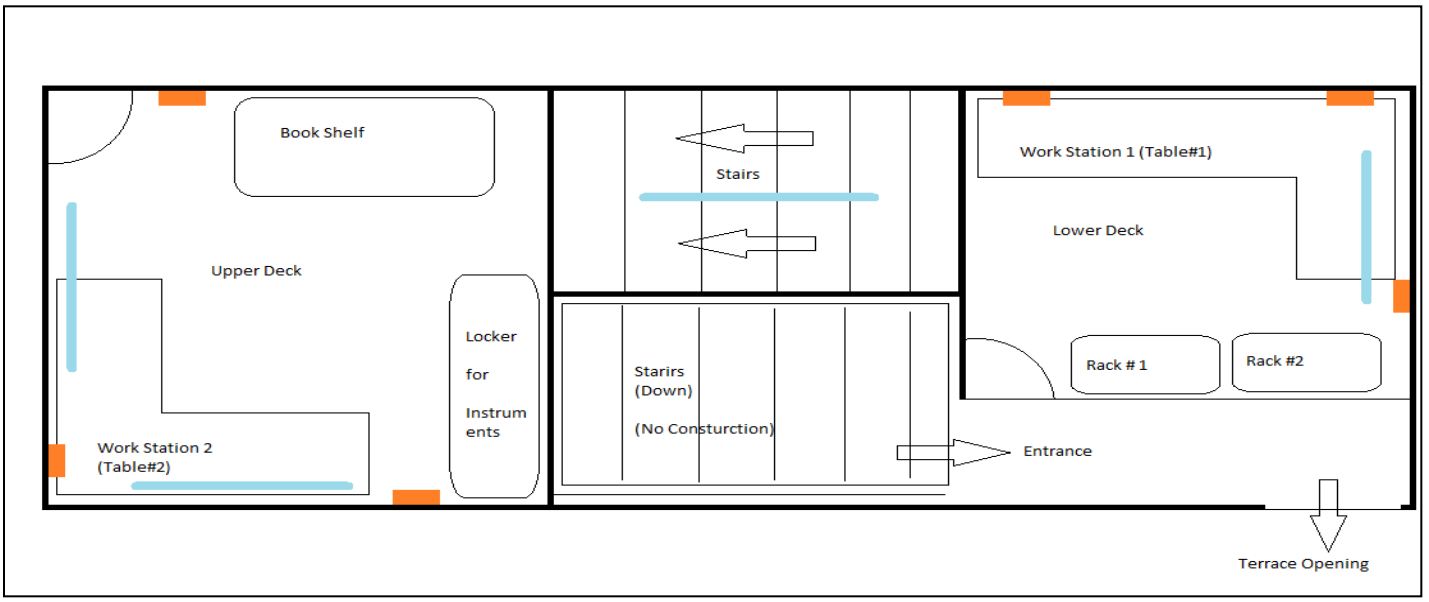
The details of the installation and other parameters are given in Appendix

5.4 Lab Design

Designing of the Lab was another challenge as we had limited space available. We wanted to install the Dome over the staircase we had little space available. So we designed the lab in such a way that it will provide place for data analysis, work station and operating deck for the observatory.

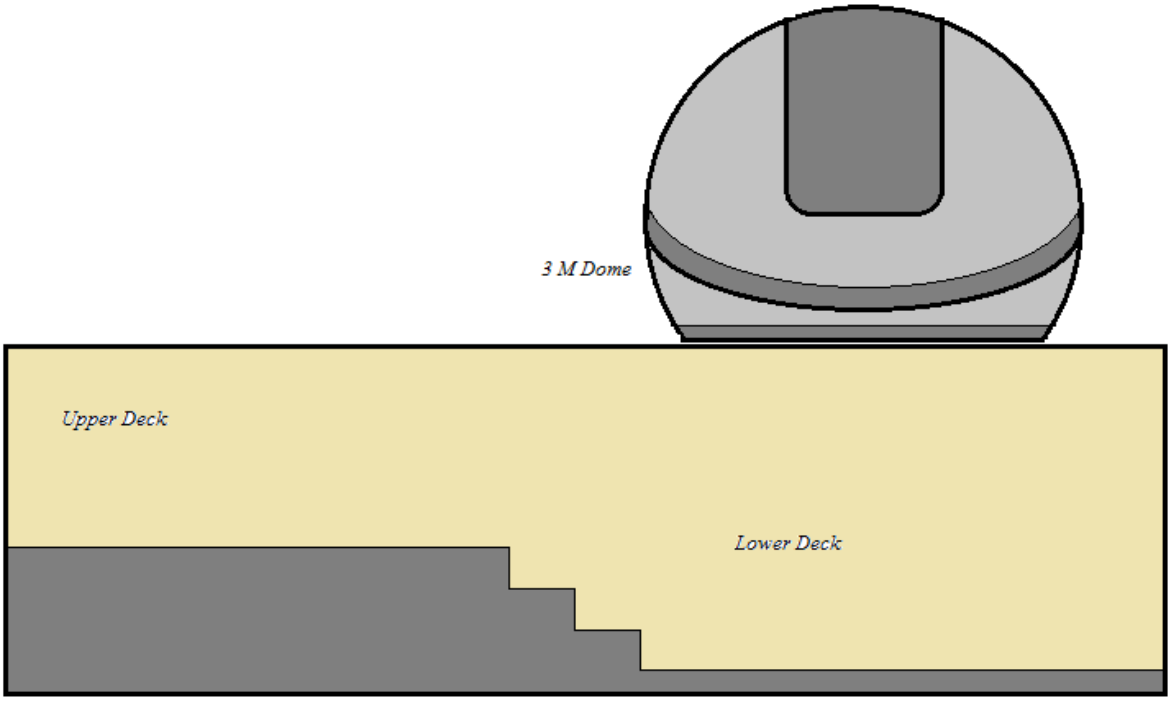
The lab also contains storage facility for equipments, book shelf, locker and metal rack for extra objects.

The storage facility is equipped with Air Conditioning and Humidity control equipment. It is to maintain the optics and ccd in good condition. The lab has the ups back up for all electronic and electrical appliances. It is provided with separate internet connection so that it will always have high speed internet. Following is the design of the lab.



- Upper Deck in above figure is Observing Deck
- Drawing is not as per the scale

Fig.5.5 Lab Desing


3 M Dome
Upper Deck
Lower Deck

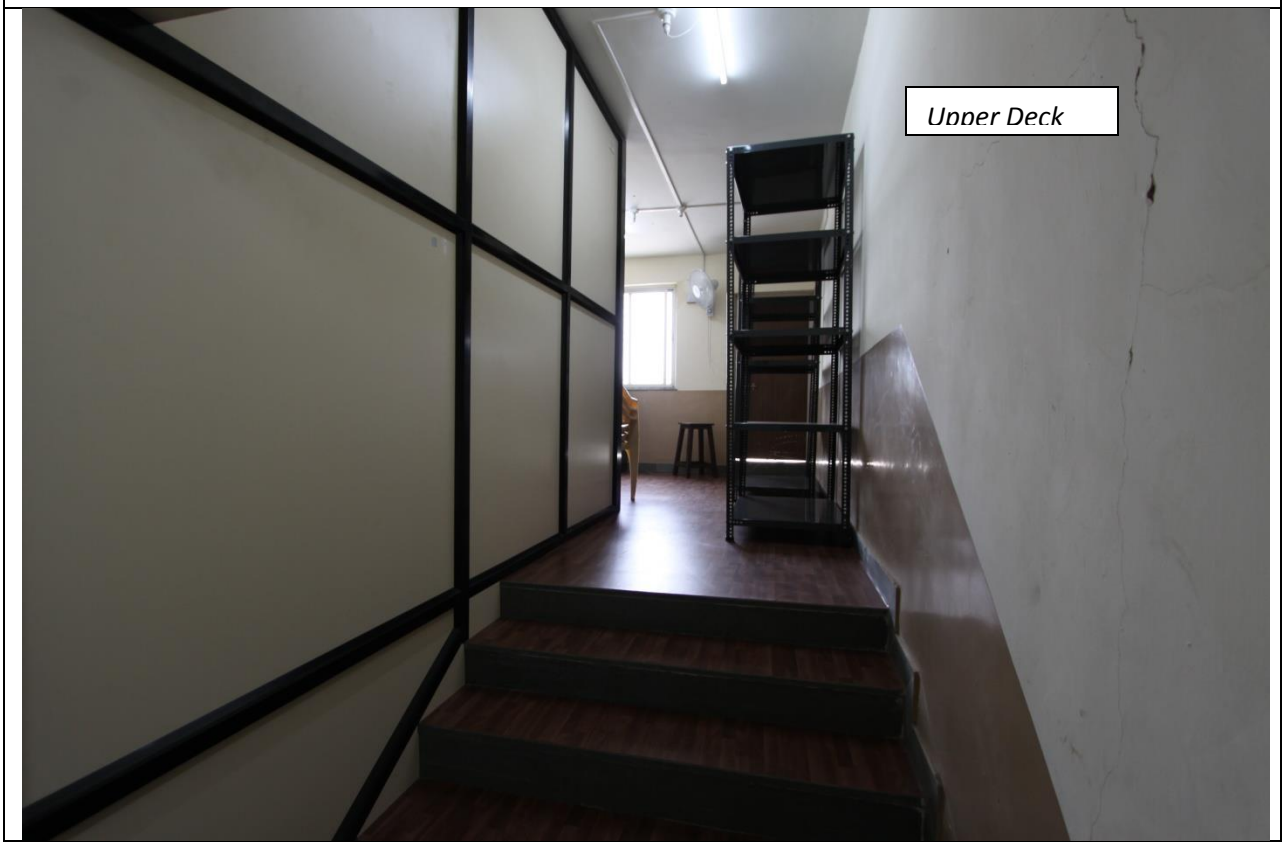

Upper Deck

Fig. 5.6 Photos of Lab

6 Observations

We had conducted several observing sessions using the current observatory set up. Initially our aim was to understand the system and to eliminate the errors and to make modifications to set up for the best observing experience. After each observation we used to make notes in which we noted the shortfall during the observations, the issues to be solved in next observing session and the chronology of the observing session.

This helped us a lot, we encountered and resolved many problems like hot to set telescope pointing to north in minimum time. We had encountered many situations where we learned new things. In one of such situations when we were taking observations we found that our telescope stopped tracking objects then we tried everything that was possible, initially we thought battery must be out but it was charged enough to carry the observations for another six hours. After one hour we found the solution: it was due to the meridian flip. Then we understood it was the problem with equatorial mount, whenever the meridian flips we have to reorient the telescope. Like this we had came across many situations and for many of them we used to find solutions for rest we had contacted the expertise from IUCAA, CIT etc and we had solved them.

We had carried out observing sessions for several nights but all of them were not research purpose. Major part of our observing sessions was to understand the working of telescope, mount, CCD and Photometer still we managed few observing sessions for the data which we can use for scientific purpose.

We had used Differential Photometric Technique to analyze the data. The technique, observing session's details and result are discussed in the next session.

6.1 Differential Photometry

Definitions

Upper case (capital) letter is what a "standard observer" measures. (V)

Lower case (small) letter is what you measure with your equipment. (v)

Subscript * is the variable star. Subscript c is the comparison star.

k is the first order extinction coefficient.

X is the air mass.

T is the transformation coefficient.

Transformation Equation

Transformation from instrumental magnitude (v) to standard magnitude (V) is a function of air mass (X) and color (B-V). The transformation equation is V = v - kX + T(B-V) + constant. Now, the transformation equations for the variable star (*) and the comparison star (c) look like this:

$V_* = v_* - kX_* + T(B-V)_* + \text{constant}$

$V_c = v_c - kX_c + T(B-V)_c + \text{constant}$

Rearrange the equations a little bit to put your observed magnitudes on the left side:

$v_* = V_* + kX_* - T(B-V)_* - \text{constant}$

$v_c = V_c + kX_c - T(B-V)_c - \text{constant}$

Now, calculate the difference between *your* observations of the variable star (*) and the comparison star (c):

$v_* - v_c = (V_* - V_c) + k(X_* - X_c) - T[(B-V)_* - (B-V)_c] - [\text{constant} - \text{constant}]$

Clearly, the constant term cancels out. (Simplification #1)

Extinction Term Cancels Out (Simplification #2)

How Differential Photometry Works?

In differential photometry, the variable star (*) and comparison star (c) are very close together in the sky. This means their air masses are practically the same. The result is that $X_* - X_c = 0$ (almost exactly). So, the extinction term, $k(X_* - X_c)$, cancels out!

The difference of instrumental magnitudes reduces to this:

$v_* - v_c = (V_* - V_c) - T[(B-V)_* - (B-V)_c]$

Color Term Cancels Out (Simplification #3)

If the variable star (*) and comparison star (c) are chosen to have the same color, then $(B-V)_* = (B-V)_c$, and the color term also cancels out!

Why Everyone Gets Nearly the Same Result

After canceling the extinction and color terms, the result is $v_* - v_c = V_* - V_c$. The magnitude difference you measure is identical to what a standard observer measures!

If this is true for one observer, then it may be true for most observers. This explains why different observers can get nearly the same results, even without transformation.

Complication - What if Colors Aren't the Same?

If the variable star (*) and comparison star (c) don't have the same color, then things aren't quite so simple. The practical worst case is that the color difference $(B-V)_* - (B-V)_c = 2.0$ magnitudes. Fortunately, the transformation coefficient is typically rather small, say T= 0.1. This is because the spectral response of your equipment probably isn't horribly different from the standard system. You do use an astronomical V filter and an astronomical CCD camera, don't you? Therefore, the color term might equal (0.1)(2.0) = 0.2 magnitude in the "practical worst" case. Unless you transform your observations, this is the bias that your data would have compared to what a standard observer measures.

If another observer uses the same equipment as you (Example - you both have a V filter from the same supplier, and you both use the same type CCD camera), then you might still get the same

results, even if they are biased. Again, different observers may agree even if they don't transform the data.

Is it worth it to transform your observations? Without transformation, you can get very consistent results, though they may have bias. For example, you may get results that are systematically 0.1 magnitude fainter than the standard system. Another way to see this bias is if your measurements are systematically above or below the AAVSO light curve.

If you just want to achieve comparability with visual observers, it may not be necessary to transform. If you want to determine the minimum of an eclipsing binary, transformation probably isn't necessary. But if an asteroid and comparison star are separated by several degrees or more, and you are observing over a wide range of air masses, then you will want to transform.

(Courtesy: AVVSO Blog from Andy)

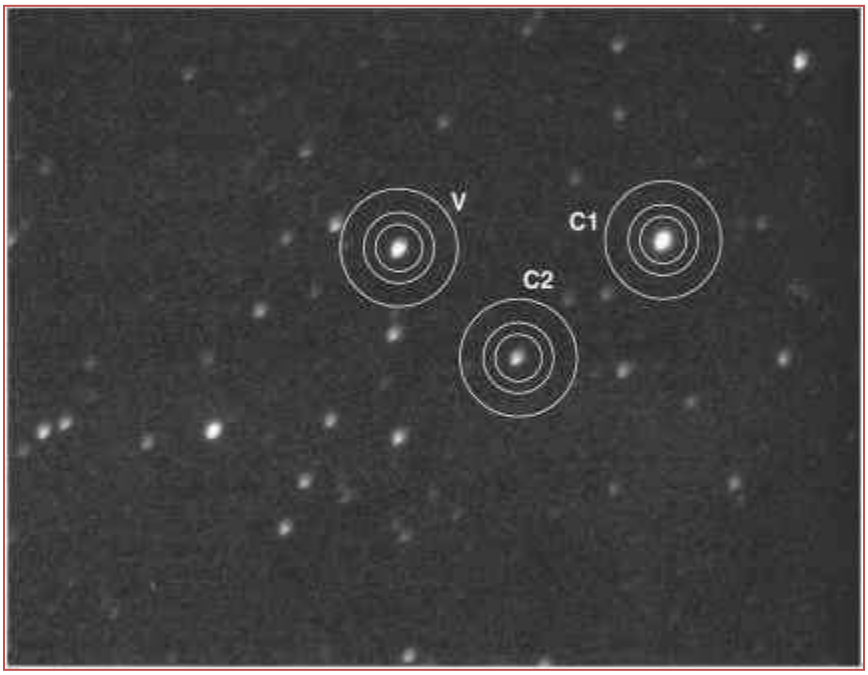


Fig6.1: Formation of circles for Differential Photometry, source: Internet

The video by Dr. Varun Bhalerao (IUCAA) helped us a lot to understand the photometry, following is the link for it.

https://www.youtube.com/watch?v=kjz8O2XTL58

Aperture Photometry Tool

We have done the differential photometry using APT i.e. aperture photometry tool.

Aperture Photometry Tool (APT) is software for astronomical research, as well as for learning, visualizing and refining aperture-photometry analyses. Image overlays, graphical representations, statistics, models, options and controls for aperture-photometry calculations are brought together into a single package. Professional astronomers appreciate APT's rich set of features and functions, and you will, too. APT is free of charge under a license that limits its use to astronomical research and education. It is perfect for the science center, observatory, and classroom. Also, it is quite simply the best FITS-image viewer you will find.

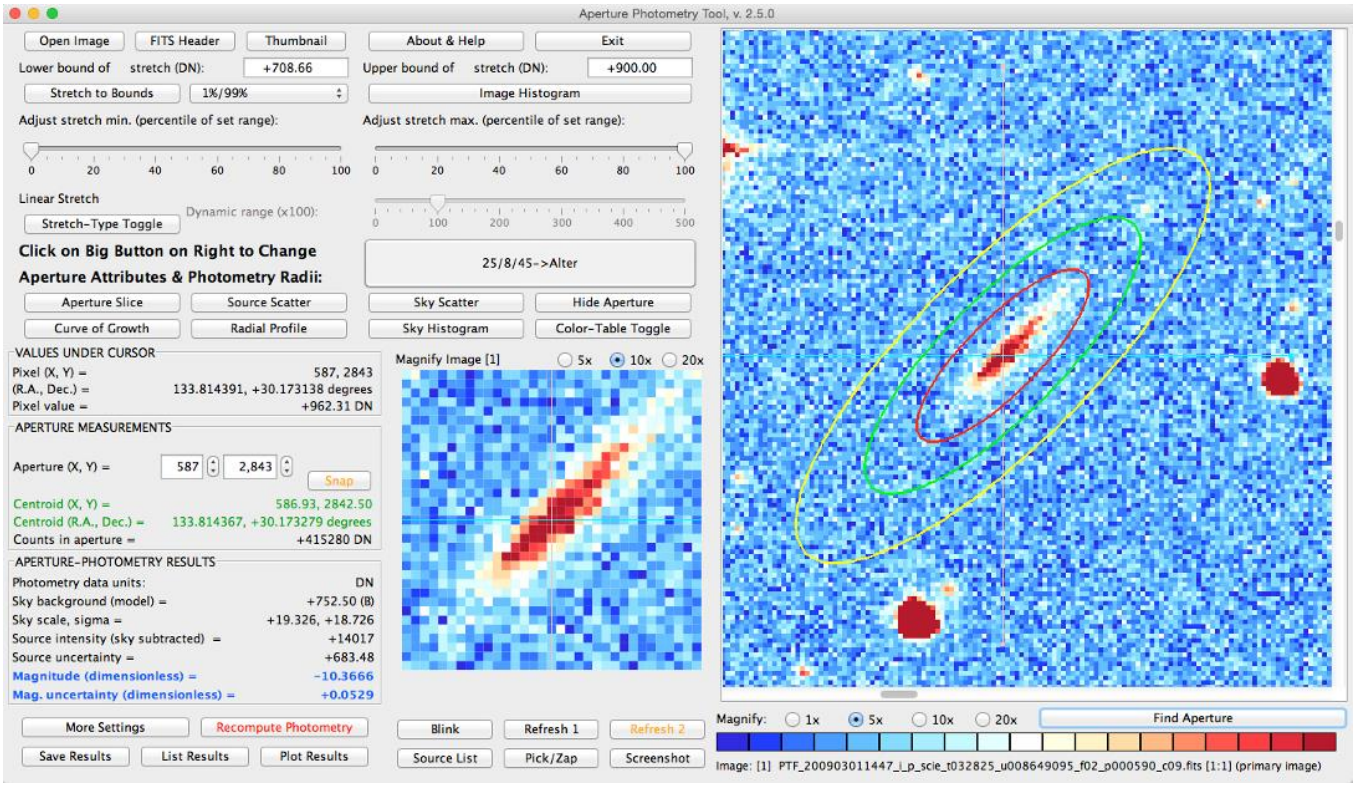


Fig6.2: APT's Main Graphical User Interface (GUI).

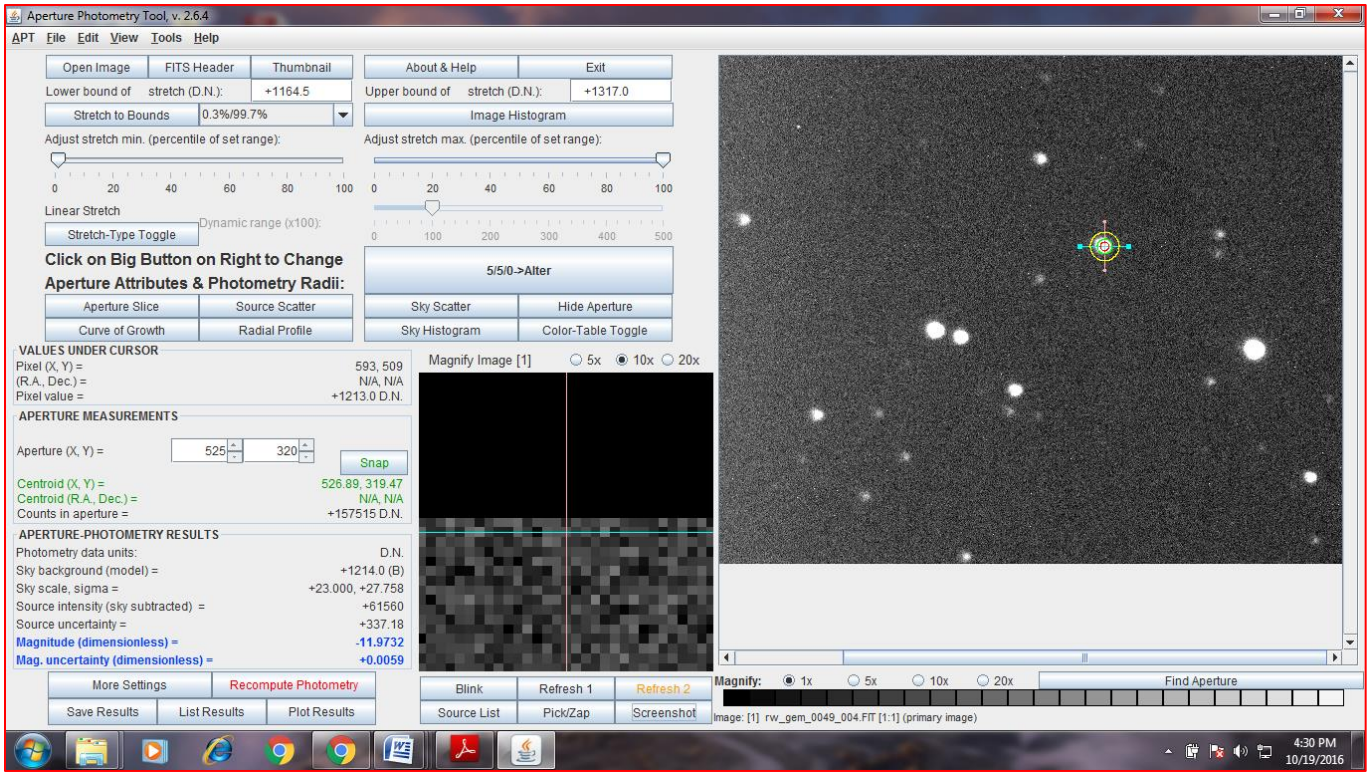
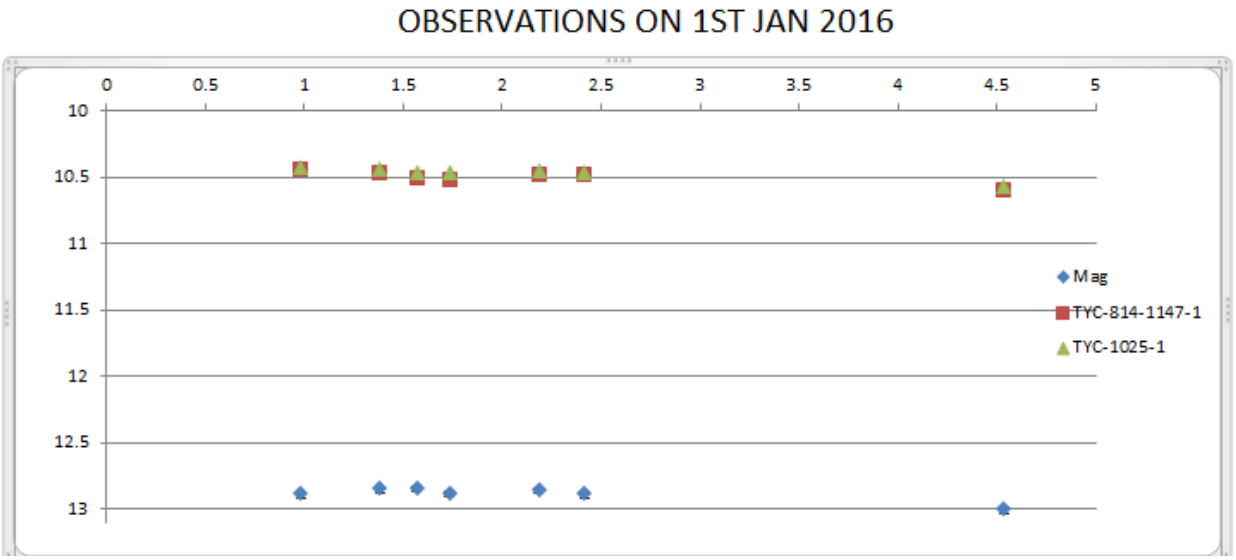


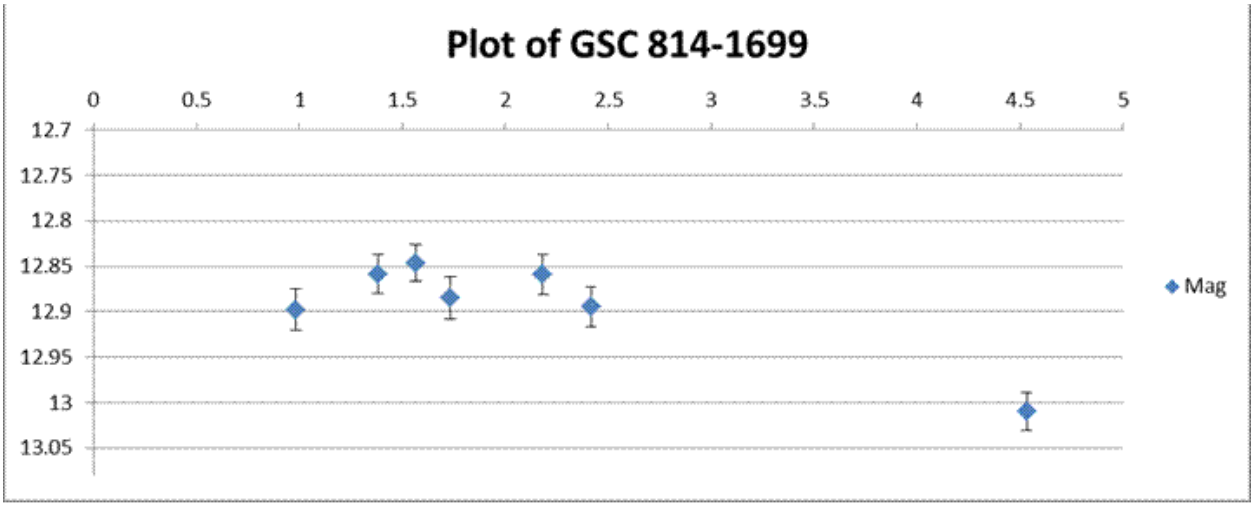
Fig6.3: Screen Shot taken while data analysis

Observation Details

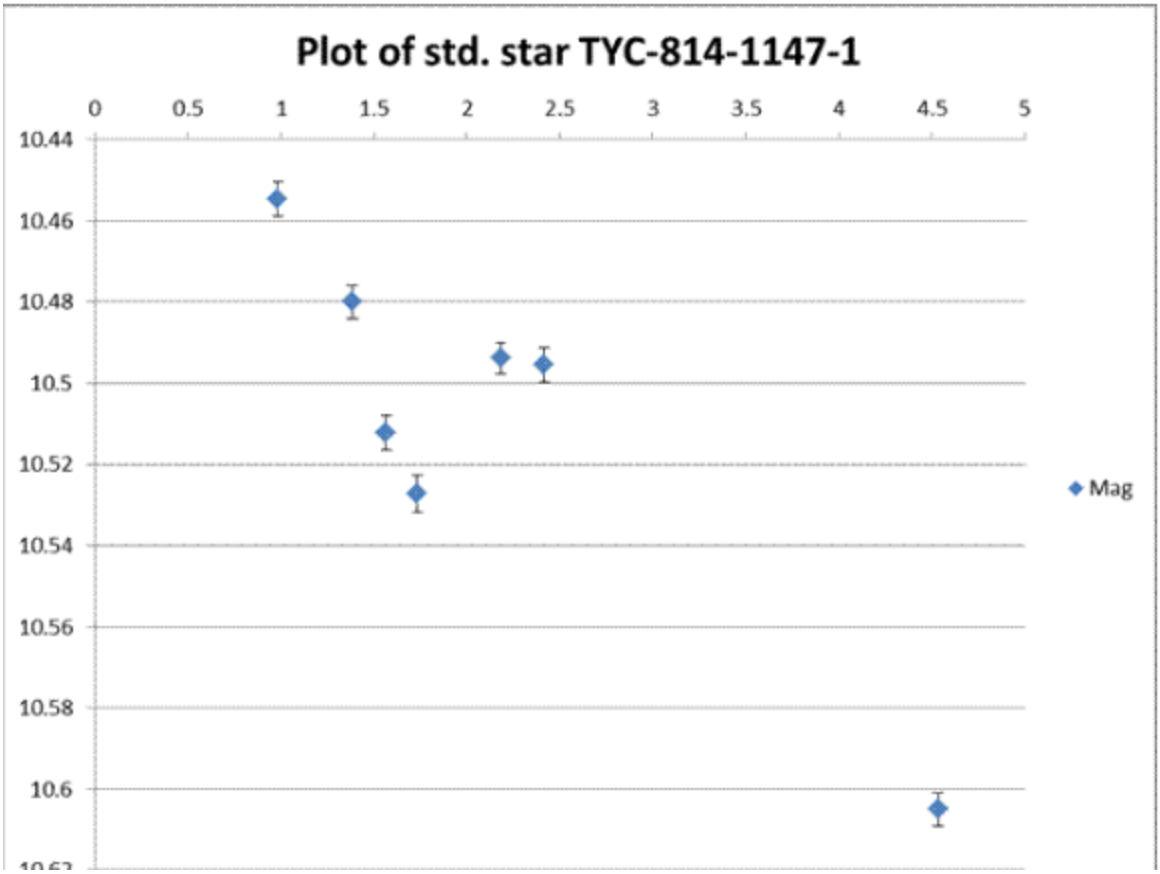
Sr. No	Date	Name of Objects
1.	4th December 2015	Algol, Pi Per, Betelguese
2.	5th December 2015	Algol, Pi Per
3.	6th December 2015	Algol
4.	29th December 2015	Algol, M5, M67
5.	1st January 2016	M67
6.	19th January 2016	Im_Aur
7.	20th January 2016	Im_Aur
8.	2nd February 2016	RW_Gem
9.	3rd February 2016	RW_Gem

Observation Result



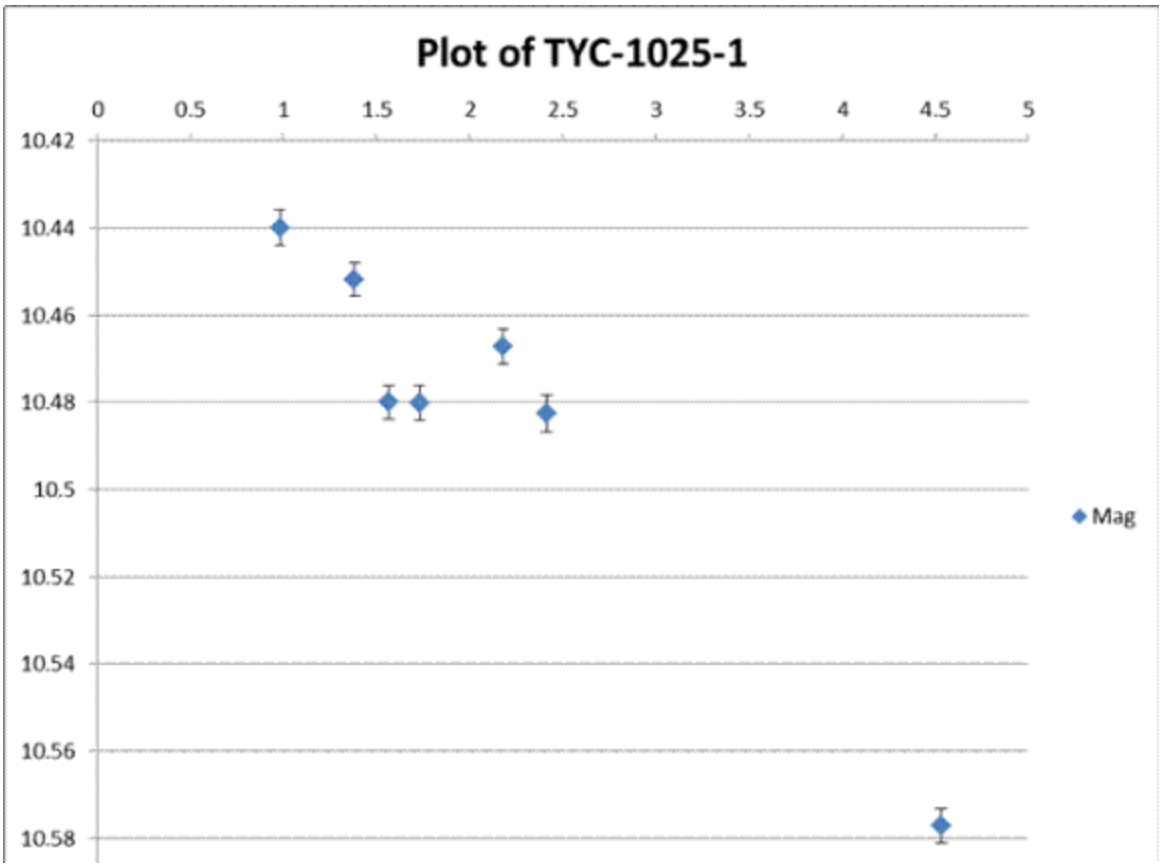


Time	Mag	Mag Unc.
0.983333	12.8977	0.0223
1.3833	12.8587	0.0216
1.5666	12.8463	0.0205
1.733	12.8849	0.0229
2.18333	12.859	0.0216
2.4166	12.8943	0.0223
4.5333	13.0101	0.0208



For TYC-814-1147-1

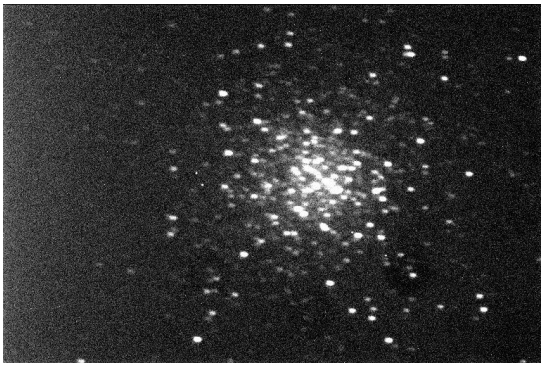
Time	Mag	Mag unc.
0.983333	10.4546	0.0042
1.3833	10.48	0.0041
1.5666	10.5122	0.0043
1.733	10.5272	0.0046
2.18333	10.4939	0.0039
2.4166	10.4955	0.0042
4.5333	10.605	0.0042



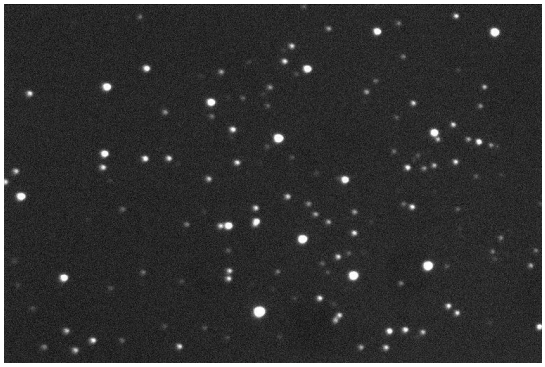
For TYC-1025-1

Time	Mag	Mag unc.
0.983333	10.44	0.0041
1.3833	10.4518	0.0038
1.5666	10.48	0.0039
1.733	10.4802	0.004
2.18333	10.4672	0.0039
2.4166	10.4825	0.0042
4.5333	10.5771	0.0039

Few Images from CCD

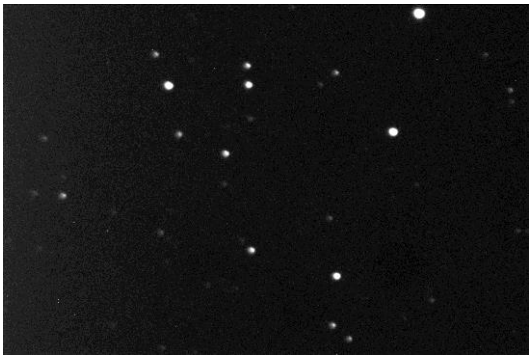


M13



M65

RW_Gem



Random

Fig. 6.4 Images from CCD

7 Future Aspect

In recent future we are collaborating with Cork Institute of Technology, Ireland under the project named TARA. The Irish contingency have already visited the college and we are now signing a MOU for the TARA observatory node in the college.

TARA Observatory will be a unique project which proposes to combine a highly configurable robotic telescope array with an innovative sensor array network. TARA, the Indian word for Star and Tara, an ancient ceremonial site in Ireland with astronomical roots represents a technological partnership between Fergusson College, Pune and Cork Institute of Technology, namely Black Rock Observatory, BCO, Ireland. The first two TARA nodes have been installed in Cork (Ireland) and California (USA).The third node will be set up in Fergusson College in collaboration with the Cork Institute of Technology CIT. At a later stage number of other nodes from various parts of the world can be added to TARA. For this we have already purchased a Robotic Dome of 3m diameter and a Paramount form Rajyashabha member grant.

TARA Observatory will:

- Provide a platform for the development of new technologies and techniques.
- Provide a unique environment for the further education of students.
- Provide a technological show case for Fergusson College, CIT and California.
- Create an outreach program to allow schools to conduct observations.
- Strengthen the already growing partnership between the Fergusson and CIT.

Future Proposal for a Pune node:

Installation of a telescope on a professional-grade robotic mount at the site already chosen in Fergusson College – the “Pune Node” This node will be connected to the nodes in Ireland and USA and will be available as part of an agreed scheduling process; as will the other nodes from Pune The Pune node will include a network of sensors which will aid in the operation of the observatory Highly flexible and scalable configuration facilitates allows for more telescopes to be added to a given node or more nodes to be added to the TARA array in due course of time.

Engineering Case, the potential for the acquisition of large volumes of data at remote sites creates a significant computing opportunity with respect to effective data analysis and storage. The innovative use of a network of sensors to monitor the telescope system as a whole is planned. Sensors including Temperature Sensors, Accelerometers, Strain gauges and Weather Station Arrays will be used to create a smart observatory. Real time measurements, such as weather conditions and optical telescope assemble temperature measurements, will aid in the acquisition of data while other sensors such as strain gauges and accelerometers will be used to predict hardware failures and aid in preventative servicing routines.

Public Outreach:

The Astro Club of Fergusson College has been actively engaged in public outreach since 1998, hence an integral to the project will be the implementation of a public outreach scheme to allow schools and colleges to utilise the telescope system as a learning and inspirational tool. Students will learn remote observation techniques and carry out remote observations.

The TARA node will provide a unique opportunity for many college students to perform observations - manually as well as remotely - of their own and to stimulate interest in Physics, Electronics and Astronomy/Astrophysics. Till date Fergusson College has already produced more than 45 professional Astronomers working at various institutions in India and abroad.

- Critical remit of the TARA Observatory project is the interaction with the public and schools
- Guided remote access to the observatory will be provided to schools in India, Ireland and USA.
- Live observing will be possible, where students actually control a TARA node in real-time
- Facilitating communications and interactions between groups for common projects will help share ideas between students. Through collaborative observations students not only learn about astronomy, science and engineering but will also learn about other cultures.

  a) As Fergusson College is going in for Autonomy, new courses at UG and PG courses can be introduced as presently India is in need for more than 800 professional astronomers by 2020.

b) The Observatory will attract more projects and grants for its development and collaborations with more institutions.

The TARA FC Node:

TARA FC will be a platform that would allow students to carry out research in various disciplines (Astronomy, Electronics and Instrumentation, Geology, Computer Sci. etc.) using the setup for remote observations. The node would also let students and teachers use this telescope for live observing sessions as well as scheduled observations. The difference in time zones is also ideal for day time astronomy – Solar studies.

Other Work in the following areas can also be carried out:

- Variable star observations - some student have already carried out these observations with IUCAA facility.
- DSLR photometry of Variables – DSLR photometry of Algol has been carried out.
- Meteor observations - Our students are regular in Meteor shower observations and contribute regularly to the International Meteor Organization. We were the first Indians to attend the IMO conf. at Armagh, Northern Ireland and present our work carried out.
- Asteroid and Comets observations – Student have taken part in International Asteroid hunt programs.
- Eclipses, Transits and Occultation's – public programs of Total/Annular Solar Eclipses, Venus Transits have been carried out.
- Solar studies:India is blessed with many hours of sunshine throughout the year many students can be involved in the activity during college hours itself (Sunshine hours).

8 Books

Following books had been purchased from ISRO grant.

Sr No	Name of The Book	Name of Author	Publication
1.	Electronic Imaging in Astronomy: Detectors and Instrumentation	Ian S. McLean	Springer
2.	Practical Astronomy With Your Calculator	Peter Duffett-Smith	Cambridge University Press
3.	Observational Astrophysics	R. C. Smith	Cambridge University Press
4.	Astrophysical Techniques	Kitchin	CRC Press
5.	The Physical Universe	Frank Shu	University Science Books,U.S.
6.	The Sky is Your Laboratory	Robert K. Buchheim	Springer
7.	A Student's Guide to the Mathematics of Astronomy	Daniel A. Fleisch and Julia Kregenow	Cambridge University Press
8.	Optical Astronomy Spectroscopy	C. Kitchin	Optical Astronomical Spectroscopy. Series

Bibliography:

Vendor List and Website Address

1. SBIG CCD Camera model ST-7XMEI-C1 OR ST-8 etc with all standard accessories

2. CFW-9 filter wheel adaptor for above with all cables/accessories etc

3. Pelican Carrying case for above

4. 1.25" Orion make flip mirror (available from Tejraj and Co Mumbai)

Vendors:

1. Anacortes Telescopes
9973 Padilla Heights Road
Anacortes, WA 98221
Fax: 001 (360) 588-9100
E-mail: herbyork@mac.com

2. Astronomics and Christophers, Ltd.
680 S.W. 24th Ave.
Norman, Oklahoma 73069
Fax: 001 (405) 447-3337
E-mail: questions@astronomics.com

3. Santa Barbara Instrument Group
147-A Castilian Drive Santa Barbara, CA 93117
Fax: 001 (805) 571-1147
E-mail: sbig@sbig.com

4. Albert Lim, MSc (Astronomy)
General Manager, Astro Scientific Centre Pte Ltd Omni Theatre, Singapore Science Centre, 21 Jurong Town Hall Road, Singapore 609433.
Fax: 65-65674826 Email: sales@astro.com.sg

5. M/S Delphotel
A-81 Vishrantika CGHS Ltd.

Plot No. 5-A, Sector-3 Dwarka
New Delhi 110075
INDIA

Telephone: +91 9312401067
Email:delphotel@yahoo.co.uk

6. Audo Viso Private Limited,
E-48, Connaught Place,
New Delhi 110 001
INDIA
FAX: +91-11-4356 1829
2341 5527
email: retail@audoviso.com or audoviso@rediffmail.com

Model SSP-3 Solid State Stellar Photometer, Generation 2. Includes: telescope coupler, 1 mm diameter photodiode with 1 mm field aperture mounted, 9V rechargeable battery, 12 VDC power supply/charger for 100-240 VAC, serial data/control I/O (USB port), and Install CD with SSPDataq for B V photometry.
OPTEC Part no: 17003
with 2nos 2-filter slider having B and V filters and R & I filter

Carrying & Storage case for SSP3 OPTEC part no: 17200

USB to Serial Converter 1-Port OPTEC part no. 17690

Vendors:

1. M/S Delphotel
A-81 Vishrantika CGHS Ltd.
Plot No. 5-A, Sector-3 Dwarka
New Delhi 110075
INDIA

Telephone: +91 9312401067
Email:delphotel@yahoo.co.uk

2. Albert Lim, MSc (Astronomy)
General Manager, Astro Scientific Centre Pte Ltd
21, Jurong Town Hall Road, Omni-Theatre, Singapore Science Centre,
Singapore 609433 Tel : +65 65674163 Fax : +65 65674826
E-mail : sales@astro.com.sg

3. Adorama
42 West 18th Street
New York, NY 10011
U.S.A.

Email: michaelp@adorama.com

4. Optec Inc.
199 Smith Street Lowell, Michigan 49331
U.S.A.

email: sales@optecinc.com
FAX: 616-897-8229

5.Audo Viso Private Limited,
E-48, Connaught Place,
New Delhi 110 001
INDIA
FAX: +91-11-4356 1829
2341 5527
email: retail@audoviso.com or audoviso@rediffmail.com

---

c) Celestron Finderscope Kit Quick Release (bracket and 50 mm finderscope) (#93784)

d) Celestron Polar Finder CG-5 (Advanced Mount) (#94224)

e) Celestron Diagonal, Star - 1.25" (#94115-A)

f) Celestron Micro Guide Eyepiece 12.5 mm 1.25" (#94171)

g) Celestron Zoom Eyepiece - 8-24 mm 1.25" (#93230)

h) Celestron 9.25-inch Dovetail Bar (CGE) (#94217) -- OR for larger Celestron telescope as the case maybe

---

Possible vendors:

1. Audo Viso Private Limited,
E-48, Connaught Place,
New Delhi 110 001
INDIA
FAX: +91-11-4356 1829
2341 5527
email: retail@audoviso.com or audoviso@rediffmail.com

2. Albert Lim, MSc (Astronomy)
General Manager, Astro Scientific Centre Pte Ltd
21, Jurong Town Hall Road, Omni-Theatre, Singapore Science Centre,
Singapore 609433 Tel : +65 65674163 Fax : +65 65674826
E-mail : sales@astro.com.sg

3. Hands on Optics
26437 Ridge Road
(Rt. 27) Next to McDonalds
Damascus, MD 20872, USA
001-301-482-1779 Fax
email: info@handsonoptics.com

4. Adorama
42 West 18th Street (Between 5th & 6th Ave.'s) New York, NY 10011
fax:001-212-741-9087
e-mail: sales@adorama.com

Identified Vendors

Oceanside Photo and Telescope, Inc
918 Mission Ave
Oceanside, CA
M-F: 10-7
Phone: 1(800) 483-6287 ext 101
Local Phone: 1(760) 722-3348
E-mail: chris@optcorp.com
www.optcorp.com
http://www.optcorp.com/pro-services/

Jacek Pala
ScopeDome Team
http://www.ScopeDome.com
tel. +48 602 315 947

Few Other Important Links

http://www.dome-observatories.com/dome_observatory_automation_system.htm

http://users.eoni.com/~garlitzj/index.html

http://www.twilightlandscapes.com/IRAFtutorial/

http://cdsweb.u-strasbg.fr/astroWeb/astroweb/full_text.html

http://www.physics.uofl.edu/williger/Astronomybootcamp/Iraf_tutorials/fernandez_ucf201510.pdf

http://astronomy.ege.edu.tr/~keskinv/avaii/docs/AVAII-IRAF/IRAF4.pdf

http://physics.bgsu.edu/~layden/BGSU_Observatory/Photom/photom_cookbook_v2.htm

http://**iraf**.noao.edu/ftp/docs/daophot2.ps.Z

http://solo.dc3.com/ar/acprefguide.html

https://arxiv.org/pdf/1011.1300.pdf