MODS

Observing Scripts

Document Number:	OSU-MODS-2011-002
Version:	1.0.2
Date:	2011 August 10
Prepared by:	R.W. Pogge The Ohio State University

Distribution List		
Recipient	Institution/Company	Number of Copies
Richard Pogge Mark Wagner	OSU LBTO	[author] 1 (PDF)

Document Change Record			
Version	Date	Changes	Remarks
0.x	2011-07-20		First formal draft
0.5	2011-08-04	Numerous	First Beta Release
1.0	2011-08-07	Wordsmithing and mistakes	First Public Release

Contents

1 Introduction

1.1 Scope

This document describes how to create and use observing scripts for acquiring scientific data and associated calibration data with the Multi-Object Double Spectrographs at the LBT. This includes discussion of target acquisition and slit-mask alignment tools.

1.2 Reference Documents

1. *MODS1 Laboratory Acceptance Test Report,* R. Pogge, et al., OSU-MODS-2009-002.
2. *LBTO Coordinate System Description*, LBT 002s105b, D. Miller, 2010 Jan 22

1.3 List of Abbreviations and Acronyms

Abbreviation	Description
ACQ	Acquisition
ADC	Atmospheric Dispersion Corrector (as in MODS doesn't have one…)
AGw	Acquisition/Guide/Wavefront Sensor unit
ASCII	American Standard Code for Information Interchange
CAL	Calibration
CCD	Charge-Coupled Device
DETXY	Detector XY Coordinates (actually Rotator-angle invariant PCS_XY)
FITS	Flexible Image Transport System
GCS	Guider Control System
GUI	Graphical User Interface
IMCS	Image Motion Compensation System
LBT	Large Binocular Telescope
LBTO	LBT Observatory
LUCI	The instrument formerly known as LUCIFER
MMS	MODS Mask Simulator (multi-slit mask design program)
MODS	Multi-Object Double Spectrographs
MOS	Multi-Object Spectroscopy
OBS	Observation
OSU	The Ohio State University
PCS	Pointing Control System
ROI	Region of Interest
SDSS	Sloan Digital Sky Survey
TBD	To Be Determined
TCS	Telescope Control System
WFS	Wavefront Sensor

2 Overview

The LBT's Multi-Object Double Spectrographs (MODS1 and MODS2) are low- to medium-resolution optical two-channel CCD spectrographs covering the 350-1000nm wavelength range. They allow direct imaging in a 6x6-arcminute field of view and long-slit and multi-slit grating and prism spectroscopy.

MODS is a complex instrument with two spectral channels split by a dichroic, a 24-position slitmask cassette, 28 remotely-operated mechanisms, a 2-channel closed-loop IR-laser based flexure compensation system, and dozens of sensors and control systems. Multiply by two, and the need to keep the complexity from getting in the way of observing is paramount.

The most efficient way to use MODS is with its command scripting interface. This document describes the scripting interface, including many worked examples, and introduces the MODS script creation tools that should help observers craft efficient MODS observing programs.

2.1 The MODS Command Script Interface

MODS scripts are plain ASCII text files containing lists of instrument and telescope commands to be executed in order from the start to the end of the file. All functions of the MODS Control Panel graphical user interface are available via the scripting interface.

Scripts provide a way to effectively and flexibly automate routine observing tasks – telescope pointing, target acquisition, instrument configuration, and data acquisition – so as to maximize observing efficiency. They also take care of many of the fine details of operating MODS, including setting up and operating the flexure compensation system, making sure the instrument is in the correct state for the type of observing to be done. This should save observers time that might be lost to errors resulting from trying to remember all of the instrument and observing setup steps late at night or after a hiatus from the LBT.

There are four types of MODS scripts, distinguished by their filename extensions:

1. **Target Acquisition (.acq)** scripts that point the LBT to a new target, setup the AGw for guiding and active optics, and take through-slit and field acquisition images for alignment of the science target with the slit mask.
2. **Observing (.obs)** scripts that acquire science images of a target once the telescope is locked on the guide star and the target is aligned with the slit mask.
3. **Calibration (.cal)** scripts that acquire bias, flat field, and wavelength calibration data.
4. **Instrument Procedure (.pro)** scripts that perform instrument setup, shutdown, and housekeeping tasks.

Scripts are executed using special "script engines": programs run in Linux terminal shells that read and process the script files and then execute the script commands in a prescribed sequence. There are two script engines for MODS:

acqMODS – executes target acquisition (.acq) scripts

execMODS – executes observing, calibration, and instrument procedure scripts.

This division of functions recognizes the different requirements of target and data acquisition: a modern active-optics telescope like the LBT requires very close choreography between

telescope, instrument, and guiding/active optics systems during target acquisition, but once science integration begins most of the activity is centered on the instrument and consists primarily of passively monitoring integration process.

MODS scripts are re-entrant, which means that an observing script interrupted by errors or other problems may be restarted from any point within the script without the need to edit or alter the script at the telescope. The script engines also provide observers with robust error trapping and a point-of-fault recovery (abort/retry/ignore) facility.

To help observers create acquisition and observing scripts, we provide a suite of script preparation tools they can install on their own computers (Linux or Mac). These tools create scripts that can be used as-is or as templates for crafting more sophisticated observing sequences. These tools should ensure that observers start with syntactically correct script files as they prepare their MODS observing projects.

This document describes the structure of MODS acquisition, observing, and calibration scripts, the script generation tools, and the script execution tools. All of the example scripts have been successfully run at the LBT during MODS1 commissioning and early science observations, and can serve as template for making your own scripts, although we strongly advise you to use the script generation tools so that the archive and other partner-specific properties are correctly defined (and you avoid breaking a script by introducing errors when editing it by-hand).

2.2 A Typical Observing Sequence

For each MODS science target, you create two scripts: an acquisition (.acq) script to point the telescope and align the slit mask with the target, and an observing (.obs) script that acquires the science data. You preselect the guide star and target coordinates before coming to the telescope, the slit position angle (e.g., aligned relative to the parallactic angle to compensate of the lack of an ADC), and build the scripts, either using the script generation programs (preferred) or editing templates and previous examples (which requires greater care).

Consider a long-slit grating spectral observation of a z=6.5 QSO. You would create two scripts for this target:

1. `j1151.acq` – target acquisition script (thru-slit and field images)
2. `j1151.obs` – long-slit spectral observation script

At the telescope, you would execute the observation following these steps:

1. Point the telescope and take thru-slit and field images after locking on the guide star:
   ```
   acqMODS j1151.acq
   ```
2. When `acqMODS` pauses, copy the raw slit and field images into a local directory and use the `modsAlign` program to align the target with the slit:
   ```
   cp /newdata/mods1r.20111027.0018.fits j1151_slit.fits
   cp /newdata/mods1r.20111027.0019.fits j1151_field.fits
   modsAlign –l j1151_slit.fits j1151_field.fits
   ```
   … mark the slit and target, then accept and execute the computed offset …
3. Resume `acqMODS` from its paused state, and it will take a confirmatory thru-slit image to make sure the target is where you want it on the slit.

4. Reconfigure MODS for spectroscopy and start the science integrations:
   ```
   execMODS j1151.obs
   ```
   …

And so on.

Both long-slit and multi-slit spectroscopy should be a straightforward 3-step process like that above: acquire the target, align the targets with the slit, start taking science data. If problems arise, acquisition and observation scripts are fully re-entrant, allowing flexible resumption of observing once the root problem has been addressed.

3 MODS Scripts

3.1 Script Basics

MODS scripts are flat ASCII text files containing lists of commands to be executed, in order, from the top to the bottom of the file. Blocks of commands can be grouped together into labeled functional blocks. Blank lines are ignored, all text following a # character is treated as a comment, and indentation can be used for readability.

Consider this sample observing script

```
#
# A MODS Observing Script example
#
# R. Pogge, OSU Astronomy Dept.
# pogge@astronomy.ohio-state.edu
# 2011 Mar 17
#

OBSMODE

Archive:
  PARTNER OSURC
  PROPID  OSU_SDSSQSOs
  PI_NAME Pogge

Instrument:
  instconfig dual grating
  slitmask LS5x60x0.8

Exec:
  object J18485-0047
  exptime 600
  nimgs 3
  dgo

END
```

The main components of this script are broken down as follows:

3.1.1 Script Execution Mode

The first non-comment line of a script must be a **Script Execution Mode Command**. In our example above

```
OBSMODE
```

indicates that this is an observing script.

The script execution mode commands make sure that MODS is in the correct physical configuration for that mode and sets important execution parameters for the script engine.

There are four script execution modes, as follows:

OBSMODE – Observing Mode
: This is a science (“sky”) observing script. Instructs MODS to make sure the

instrument dark hatch is open to the sky, the calibration tower is stowed, and all calibration lamps are turned off before executing the rest of the script.

CALMODE – Calibration Mode

This is a calibration script. Instructs MODS to close the instrument dark hatch, retract the AGw stage, and deploy the calibration tower before executing the rest of the script.

ACQMODE – Target Acquisition Mode

This is a target acquisition script. ACQMODE must include an argument indicating what kind of acquisition is being performed:

`Imaging` – an imaging target acquisition
`Slit` – a long-slit spectroscopy target acquisition
`MOS` – a multi-object spectroscopy target acquisition

Like OBSMODE, it also makes sure the instrument is open to the sky and the calibration system stowed and turned off before executing the rest of the script.

SETUPMODE – Instrument Setup Mode

This is an instrument setup procedure script. Unlike OBSMODE, CALMODE, and ACQMODE, this will not check or change the current state of the instrument dark hatch, calibration system, or AGw stage before proceeding. Observers will rarely if ever create SETUPMODE scripts, although you may execute predefined setup scripts at the instruction of LBTO support scientists. SETUPMODE scripts will not be discussed further in this document.

3.1.2 Functional Blocks

Commands within a MODS script are grouped into labeled **functional blocks**. From our example script above, here are the first two functional blocks

```
Archive:
  PARTNER OSURC
  PROPID  OSU_SDSSQSOs
  PI_NAME Pogge

Instrument:
  instconfig dual grating
  slitmask LS5x60x0.8
```

A functional block begins with a single-word **block label** that is terminated by a colon (:). Block labels must appear on a line by themselves with no other commands or comments. While block labels are **case insensitive**, it is conventional to capitalize the first letter or used mixed case if the label is a compound of multiple words.

A functional block ends at the next block label or at the end of the script, whichever comes first. In the example above, the `Archive:` block ends with the last command before the `Instrument:` block label, while the `Exec:` block ends with the script.

Observers can define their own functional block names, for example to organize multipart observing scripts (see §6.5 for an example in a multi-lamp calibration script). There are, however five reserved block labels used by the scripting engines for specific purposes, as follows:

Archive:	Information required by the LBTO image data archive
Instrument:	Main instrument configuration block
Target:	Target information block of an acquisition script.
Exec:	Main execution block of an observing or calibration script
Acquire:	Main execution block of an acquisition script.

Each of these will be discussed in their proper contexts in the sections below.

For readability, a well-formatted script will indent the commands within a block by 2 or 3 spaces. Commands will be executed serially in the order in which they appear in the script.

Functional blocks are a key component of how MODS scripts implement re-entrancy.

3.1.3 Instrument and Telescope Commands

Commands for the MODS instrument or the telescope appear as text on lines by themselves, one command per line.

Commands may be a single word, or a command word followed by one or more arguments. For example:

```
object J18485-0047
exptime 600
nimgs 3
dgo
```

While the command words themselves are case insensitive, their arguments are passed to the instrument as is, preserving case.

A full list of commands and their argument are described in §7.

3.1.4 Script Flow Control and Printing Commands

MODS scripts are simple lists of commands to be run in sequence. A very limited number of internal script commands are provided to control the forward flow of a script and print messages to the console to give feedback as to script progress.

`PAUSE`
: Pause script execution and wait for the user to hit the Enter key to resume, or X to abort execution and stop the script.

`SLEEP` `sec`
: Suspend execution of the script and go to sleep for `sec` seconds. Gives the user a countdown to monitor the sleep interval.

`PRINT` `message text to print`
: Prints a message on the terminal screen.

Our intention is to keep MODS scripts simple, so we do not provide higher-level flow control like loops, branching, or logical if/else tests.

3.1.5 End Command

All scripts must end with the word "end" as the last line. Note that like all command words, end is case insensitive.

Any lines of the script following an END command will not be executed, but it is bad form to have commands after an END. Instead, comment out unused commands or functional blocks.

3.1.6 Script Comments

All text following a # character is treated as a comment and not executed. For example, the top section of the example script

```
#
# A MODS Observing Script example
#
# R. Pogge, OSU Astronomy Dept.
# pogge@astronomy.ohio-state.edu
# 2011 Mar 17
#
```

is a block of comments used to annotate the script and record its use and origins.

You may also use in-line and indented comments, for example

```
Exec:
  red filter i_sdss   # Select SDSS I filter
  nimgs 2
  #nimgs 3
  go
```

The in-line comment in the 2nd line explains the command ("this is what I'm doing here"), and the comment in the 4th line disables ("comments out") the "`nimgs 3`" command preventing it from being executed.

Scripts created by one of the script preparation tools described in §4 write a comment block at the top of scripts to record the date of creation, the program version number, the user who created it, and other information to help establish the provenance of the script.

Judicious uses of comments can be very helpful to the on-site observers, especially if you are doing something complex or unusual. Overuse of comments, however, can make scripts hard to read or repair if there are problems, so striking a balance is essential.

3.2 Target Acquisition (.acq) Scripts

Target Acquisition (.acq) scripts point the LBT to the coordinates of a new target, setup the guide star, and take acquisition images to align the target(s) with the slit mask. Acquisition scripts are executed using the `acqMODS` program (§5.1), and the standalone `modsAlign` program (§5.2) is used to align targets with slits.

An example of a long-slit acquisition script is as follows:

```
#
# Example MODS Long-Slit Target Acquisition Script
#

ACQMode SLIT
```

```
Archive:
  PARTNER OSURC
  PROPID OSU_HighZ
  PI_NAME DeRosa

Target:
  OBJNAME J1420-1620
  OBJCOORDS 14:20:36.32 -16:02:30.18
  GUINAME GStar
  GUICOORDS 14:20:37.567 -16:06:30.08
  ROTATOR 0 POSITION
  PRESET ACTIVE

Instrument:
  SLITMASK LS5x60x0.8
  ACQCamera Red
  ACQFilter r_sdss
  ACQExpTime 60
  ACQROI 1Kx1K

Acquire:
  SlitGO
  AcqGO
  PAUSE
  SlitGO

END
```

A typical acquisition script will take the following actions, in order:

1. Put the instrument and data-taking system into acquisition mode for long-slit (`ACQMode SLIT`), making sure MODS is open to the sky and the calibration system is stowed.
2. Load the Partner ID, Proposal ID, and PI Name into system for inclusion in subsequent image FITS headers so that the images taken will be searchable in the LBTO data archive (`Archive:` block)
3. Upload the target and guide star coordinates and the desired the rotator position angle to the telescope and execute an active preset (`Target:` block) to point to the target, lock on the guide star, and start the active optics correction loop (`preset active`).
4. Configure MODS to use the named slit mask and setup for acquisition images using the named camera, filter, exposure time, and CCD region-of-interest subframe readout.
5. Acquire a pair of acquisition images through the slit (`SlitGO`) and with the slit retracted (`AcqGO`), then pause and wait for the observer to offset the target onto the slit using the `modsAlign` program, and then on resuming execution after the `PAUSE`, take a final confirmatory through-slit image (second `SlitGO`)

Acquisition scripts use special commands that take care of many of the details of pointing and instrument configuration, and coordinating all of the activities of the telescope, guiding, and instrument control systems.

3.2.1 Acquisition Mode (ACQMode)

The first non-comment command in an acquisition script must be **`ACQMode`**, specifying the acquisition mode to use:

`ACQMode Imaging` – acquire a direct imaging target

`ACQMode Slit` – acquire a target for long-slit spectroscopy

`ACQMode MOS` – acquire a target for multi-object spectroscopy

Imaging target acquisitions are the simplest: they point the telescope at the field and, if requested, snap a confirmatory acquisition image.

Long-slit and MOS target acquisition modes enable commands to take rapid sequences of thru-slit and field (no mask) images including the mask insert/retract motion.

The external program `modsAlign` (§5.2) is provided to compute and execute telescope offsets to align targets with the slit. Blind offsets are also possible as described in §3.2.6.

3.2.2 Archive: Block

The `Archive:` block consists of three commands that define the FITS header information needed to make your data searchable and accessible in the LBTO data archive.

```
Archive:
  PARTNER OSURC
  PROPID OSU_HighZ
  PI_NAME DeRosa
```

All three are required for every script. See §7.2 for descriptions of these commands.

3.2.3 Target: Block

The `Target:` block specifies the target and guide star coordinates, the rotator position angle (and mode) to use, and the type of telescope preset to execute. From our example above:

```
Target:
  OBJNAME J1420-1620
  OBJCOORDS 14:20:36.32 -16:02:30.18
  GUINAME GStar
  GUICOORDS 14:20:37.567 -16:06:30.08
  ROTATOR 0 POSITION
  PRESET ACTIVE
```

The required `Target:` block commands are as follows:

`OBJNAME` – name of the science target, max 40 characters.

`OBJCOORDS` – science target coordinates, RA in sexagesimal hours, Dec in sexagesimal degrees, equinox J2000. **The colons (:) and sign on the Dec are required**.

`GUINAME` – name of the guide star. GStar is a good placeholder.

`GUICOORDS` – same format and equinox as `OBJCOORDS`.

`ROTATOR` – instrument slit position angle in **decimal degrees** and the rotator mode:
`POSITION` – Celestial Position Angle, measured North through East

`PARALLACTIC` – Rotator Parallactic Angle

You should always use `POSITION` for routine observing.

`PRESET` – telescope preset mode, choices are:

`ACTIVE` – guide and apply active optics corrections

`GUIDE` – guide on the target but make no active optics corrections

`TRACK` – track open loop on the target (no guiding or active optics)

Nearly all observations should use **`ACTIVE`** presets.

Restrictions:

`PRESET` must always be the **last command** in the `Target:` block.

No other commands should appear in the `Target:` block.

3.2.4 Instrument: Block

The `Instrument:` block configures the instrument for target acquisition imaging:

```
Instrument:
  SLITMASK LS5x60x0.8
  ACQCamera Red
  ACQFilter r_sdss
  ACQExpTime 60
  ACQROI 1Kx1K
```

The commands for this block are as follows:

`SLITMASK` `maskID`

Name of the slit mask to use. **`maskID`** must be one of the mask names listed in the current mask table on the LBTO wiki.

Mask cassette device position numbers should never be used.

If `ACQMode` is `Imaging`, use `Imaging` for the `maskID`.

`ACQCamera` `[red|blue]`

Camera for taking thru-slit and field target acquisition images. Only one camera is used for acquisition images.

`ACQCamera` must come <u>before</u> all subsequent `ACQxyz` commands.

`ACQFilter` `filterID`

Camera filter to use. Recommended filters are:

Red Camera: `r_sdss`

Blue Camera: `g_sdss`

but other choices are valid (e.g., z_sdss for acquiring z>6 quasars).

`ACQExpTime` `seconds`

Acquisition image exposure time in seconds, 1 second minimum.

`ACQROI` `ccdConfig`

Optional: change the CCD sub-frame readout Region-of-Interest configuration for the acquisition images. If omitted, it uses the default imaging ROI (3K×3K). Valid ROIs for acquisition images are:

`1Kx1K` – 1024×1024 pixels centered in the imaging field

`2Kx2K` – 2048×2048 pixels centered in the imaging field

`3Kx3K` – 3072×3072 pixels centered in the imaging field [**default**]
The typical unbinned readout times are X, Y, and Z seconds, respectively.

Restrictions:

`ACQCamera` must come **before** all other `ACQxyz` commands in the block.

For imaging target acquisitions (ACQMode Imaging), specify `SLITMASK Imaging`.

3.2.5 Acquire: Block

The Acquire: block executes the target acquisition images after the preset has been completed and the instrument configured for imaging. For example:

```
Acquire:
  SlitGO
  AcqGO
  PAUSE
  SlitGO
```

Two commands are provided for thru-slit and field (no mask) acquisition images:

`SlitGO` – insert the slit mask and take a thru-slit image.

`AcqGO` – retract the slit mask and take a field (no mask) image.

Which you use and in what order depends on what you are doing.

Direct Imaging Mode: A single `AcqGO` to confirm the field in the science camera usually suffices. `SlitGO` makes no sense (no slit).

Long-Slit Mode: The usual practice is to take a thru slit and field images in that order. The first thru-slit image (`SlitGO`) will be taken while active optics is still converging, and by the time the following field image is taken (`AcqGO`) active optics will usually have converged. After the `AcqGO`, you `PAUSE` and use the `modsAlign` program with the `-l` flag ("el" for long-slit) run in another terminal window to compute and execute the telescope offset to put the target in the slit. A final `SlitGO` takes a confirmatory thru-slit image.

MOS Mode: The practice is the same as with Long-Slit spectra, except now the full 2D alignment mode of `modsAlign` is used to align the slit mask in offset and rotation after the `PAUSE`. Confirmatory thru-slit images are usually required for MOS spectra to be sure the mask is properly aligned.

The script example in this section uses the `PAUSE` command to suspend the acquisition script while you are running the `modsAlign` program in another terminal window.

3.2.6 Blind Offsets to Faint Targets

Blind offsets may be performed given an ($\Delta\alpha$,$\Delta\delta$) offset in celestial coordinates (units of arcseconds) between a reference star and the faint target. The changes to the acquisition sequence would be as follows:

1. `OBJCOORDS` in `Target:` block would be the RA and Dec of the **reference star**
2. The reference star is acquired on the slit using `SlitGO` and `AcqGO` as above.

3. In the Acquire: block the second confirmatory `SlitGO` is followed with a relative `OFFSET` command to move the faint target into the slit. For example, this `Acquire:` block is for a faint target offset by -32.3 arcsec West and +12.5 arcsec North of a bright, readily acquired nearby reference star:

```
Acquire:
  SlitGO
  AcqGO
  print Put the reference star on the slit using modsAlign -l
  PAUSE
  SlitGO
  offset -32.3 12.5 rel
  updatepointing

END
```

See §7.8.1 for a description of the `OFFSET` command.

3.3 Science Observing (.obs) Scripts

Observing (.obs) scripts configure the instrument and acquire science data after a successful target acquisition. Observing scripts are executed using the `execMODS` program (§5.3).

A typical observation script is as follows:

```
#
# MODS Observing Script
#

OBSMode

Archive:
  PARTNER OSURC
  PROPID OSU_HighZ
  PI_NAME DeRosa

Instrument:
  instconfig dual grating
  slitmask LS5x60x1.2

Exec:
  object J1420-1620
  exptime 1200
  nimgs 6
  go

END
```

Observation scripts take the following actions, in order:

1. Make sure MODS is open to the sky, retract the calibration tower if in, and turn off all of the calibration lamps (`OBSMode`).
2. Load the archive info (`PARTNER` et al.) for inclusion in subsequent image FITS headers so that the images taken will be searchable in the LBTO data archive. The Archive: block should be the same as in the corresponding acquisition script.

3. Configure MODS for the desired instrument mode (dual- or single-channel operation, imaging, grating spectroscopy, or grism spectroscopy), and inserts a slit mask.
4. Setup and execute the science exposures on the target.

The following sections describe each of the sections of the observing script.

3.3.1 Observation Mode (OBSMode)

The first executable command in an observation script must be **`OBSMode`**. This confirms that the instrument is open to the sky with the calibration system stowed and all calibration lamps turned off before executing any observing commands.

3.3.2 Instrument: Block

The `Instrument:` block configures the instrument for science observing:

```
Instrument:
  instconfig dual grating
  SLITMASK LS5x60x1.2
```

Commands for this block are as follows:

`INSTCONFIG` `config disperser`

Configure the instrument. **`config`** defines the instrument optical configuration:
- **`dual`** – Dual channel (Red+Blue) operation
- **`blue`** – Blue-only operation (red channel is blocked and idle)
- **`red`** – Red-only operation (blue channel is blocked and idle)

`disperser` selects the disperser option:
- **`imaging`** – imaging flat mirror(s)
- **`grating`** – reflection grating(s)
- **`prism`** – double-pass prism(s)

`INSTCONFIG` also sets up the default CCD readout properties (ROI and binning appropriate to that instrument mode), selects the default spectroscopic filters and the dichroic configuration, clears exposure flags, and locks on the image motion compensation system (IMCS). Preset instrument configurations make for highly efficient instrument setup in which all mechanisms are moved simultaneously, so that the time to configure the instrument is set by the slowest mechanism (the grating turrets).

`SLITMASK` `maskID`

Insert the named slit mask into the science field. **`maskID`** must be a valid loaded mask name as listed in the mask table at the observatory.
Mask cassette device position numbers should never be used.
In imaging mode, use the maskID=Imaging mask (a field stop that is part of the internal stray light baffles).f

3.3.3 Execution (Exec:) Block

The `Exec:` block contains all of the commands that setup and take the exposures for the script. For example, to take dual-grating spectroscopy with 2 hours of total exposure time divided into six 1200-second single exposures would look like this:

```
Exec:
  object J1420-1620
  exptime 1200
  nimgs 6
  go
```

Because MODS is a 2-channel instrument, adding the RED or BLUE qualifier before a command will direct the command to only that channel. For example, the Exec: block for an imaging script that cycled through the ugriz imaging filters in the red and blue channels might look this:

```
Exec:
  object NGC 1234 ugriz
  # Set 1 of 3 - 2*(g+r)
  blue filter g_sdss
  red filter r_sdss
  exptime 15
  nimgs 2
  go
  # Set 2 of 3 - u+2*i
  blue filter u_sdss
  red filter i_sdss
  blue exptime 30
  red exptime 15
  blue nimgs 1
  red nimgs 2
  go
  # Set 3 of 3 - u+z
  red filter z_sdss
  red exptime 30
  red nimgs 1
  go
end
```

This sequence takes two 15-second g, r, and i images, two 30-second u images, and one 30-second z image. The synchronization of exposure and readout times in the middle u+2×i images in this example is sloppy and inefficient, but it serves to illustrate the principle.

The original 4-command `Exec:` block example in this section exploited the fact that the `OBJECT`, `EXPTIME`, and `NIMGS` commands, when used without a Red or Blue channel qualifier, will set those parameters the same on both channels. These are examples of so-called "implicit dual" commands. For example, the sample `Exec:` block the first dual-channel observation is equivalent to

```
Exec:
  red object J1420-1620
  blue object J1420-1620
  red exptime 1200
  blue exptime 1200
  red nimgs 6
  blue nimgs 6
  go
```

However, if a single channel configuration is selected, the implicit dual commands will only set parameters on the active channel (thus if we were to have selected "`instconfig red`

`grating`", the unqualified `object`, `exptime`, and `nimgs` commands would only set the red channel).

Be careful, however, with GO. If, for example, in the above "fully qualified" version of the Exec: block above, we had instead of a single GO the red and blue GO commands

```
blue go
red go
```

it would first take the blue images **then** take the red images, whereas

```
go
```

starts the red and blue images simultaneously – you could also use `DGO` to make the "Dual GO" operation explicit (see §7.5.13).

> **Never use a channel-qualified GO command in an observing script unless you really intend to leave the other channel idle.**

3.3.4 Exec: Block Commands

While in principle any valid MODS instrument or telescope interface command can be used in an `Exec:` block, the following are the most common and useful commands that will let you execute most observing programs.

`OBJECT` `target name`
: Name of the target and the image type. The target name is stored in the OBJECT header keyword, and the image type in the IMAGETYP keyword. In addition to `OBJECT`, valid image types are `BIAS, FLAT, COMP, STD, SKY`, and `DARK`. See §7.5 for details.

`EXPTIME` `seconds`
: Set the single-frame exposure time in seconds. The minimum exposure time is 1 second, the longest practical time is about 1800s (limited by cosmic ray accumulation on the red channel's deep-depletion CCD detectors).
NOTE: `EXPTIME` must FOLLOW `object, flat, comp`, etc.

`NIMGS` `n`
: Set the number of single-frame exposures to acquire per `GO`

`GO`
: Start exposures on the active cameras(s). The total exposure time will be
`TotalExp = (NIMGS×EXPTIME)`.

`[RED|BLUE] FILTER` `filtID`
: Put the named filter `filtID` into the beam in the named camera (red or blue). Available filters are listed on the instrument website.
For spectroscopic modes, suitable default filters are selected by `instconfig`.

Other commands are available, see §7 for a full summary and the example scripts in §6.

3.4 Calibration (.cal) Scripts

Calibration (.cal) scripts take closed-instrument internal calibration images: bias, flat-field, wavelength comparison lamp ("comps") and dark frames.

```
#
# Dual Grating Mode Hg+Ne Comparison Lamps
#
# Ne + Hg Lamps - wait 30s for Hg to warm up
# ND1.5 filter in both channels.  3 images per side
#

CALMode

Archive:
  PARTNER CALIBRATION
  PROPID  CALIBRATION
  PI_NAME ALL

Instrument:
  instconfig dual grating
  slitmask LS5x60x0.6

Exec:
  lamp hg ne on
  print Waiting 30s for the Hg lamp to warm up
  sleep 30
  comp Ne + Hg[Ar] Lamps
  blue exptime 2
  blue filter ND1.5
  red exptime 1
  red filter ND1.5
  nimgs 1
  dgo
  lamp off

end
```

This script incorporates a number of new script elements as we describe below.

3.4.1 Calibration Mode (CALMode)

The first executable command in a closed-instrument calibration script must be **`CALMode`**. This makes sure that the instrument dark hatch is closed, the AGw stage retracted and stowed, and the calibration tower inserted into the beam. The guide probe XY position is stored so it can be restored later using the "`OBSMode Restore`" command if desired (see §7.1.2).

The calibration procedures for MODS were worked out during commissioning, and are described in a separate *MODS Calibration Procedures* document. Please read that document for the prescribed procedures for calibrating your data. A number of stock calibration scripts have been created that should cover most of your needs for routine calibration, and provide good templates for custom calibrations. This section is intended to help you understand the contents of those scripts.

As a final note, standard star observations (spectrophotometric flux standards or photometric standard fields) are treated the same as science observing scripts since they collect light from astronomical sources, and so are not covered in this section.

3.4.2 Calibration Archive: Block

The `Archive:` block for a calibration images requires special values for the `PARTNER` and `PROPID` keywords:

```
Archive:
  PARTNER CALIBRATION
  PROPID  CALIBRATION
  PI_NAME ALL
```

This makes it easier for all observers to access common calibrations for a queue/service run, and lets other observers get access to all calibrations in case they need backups. If you require special calibrations beyond the standard set, you may set the PI_NAME to your name to enable you to search for them in the LBTO data archive.

3.4.3 Internal Calibration Lamps

The Exec: block for this example script shows how to use the internal calibration lamps.

```
lamp hg ne on
print Waiting 30s for the Hg lamp to warm up
sleep 30
comp Ne + Hg[Ar] Lamps
...
lamp off
```

In this excerpt, we introduce the LAMP command:

`LAMP` `id1 id2 ... ON`

Turn on the named lamps id1, id2, etc. Valid lamps are:

`Ne` – Neon comparison lamp
`Hg` – Mercury (Hg+Ar) comparison lamp
`Kr` – Krypton comparison lamp
`Ar` – Argon comparsion lamp
`Xe` – Xenon comparison lamp
`QTH1` – Quartz-Halogen continuum lamp #1
`QTH2` – Quartz-Halogen continuum lamp #2
`VFLAT lev` – Variable intensity continuum lamp, `lev=1..10`

`LAMP OFF`

Turn off power to all of the calibration lamps. If an optional lamp ID is given, it will turn off only that lamp.

The uses of these calibration lamps are explained in the *MODS Calibration Procedures* document. During instrument commissioning we worked out pairs of lamps, filters and exposure times that will give good, balanced calibration spectra in both channels to aid in calibration efficiency. You should use the facility calibration scripts wherever possible, or at least use them as templates for custom calibration scripts.

In this example we are using the Hg(Ar) lamp which needs to warm up for at least 30 seconds before the lines are bright enough to be useful. To do this, we introduced a `SLEEP` command after the `LAMP` command, along with a `PRINT` statement to let the observer why it is sleeping:

```
lamp hg ne on
print Waiting 30s for the Hg lamp to warm up
```

```
  sleep 30
```

Only the Hg(Ar) lamp requires a warm-up interval (why is it not automatic? In future versions of the lamp control system it might be…).

At the end of the calibration integrations, we include an explicit `LAMP OFF` command to turn off all lamps. All calibration scripts that use the internal lamps should do this.

3.4.4 Bias and Dark Calibrations

Bias and Dark calibration frames do not collect light, but they do need the instrument to be closed and dark so `CALMode` should be used. They can, however, omit the `Instrument:` block. For example:

```
#
# Acquires 10 each Red and Blue CCD bias images
# Full-Frame 8Kx3K Spectroscopic ROI unbinned

CALMode

Archive:
  PARTNER CALIBRATION
  PROPID  CALIBRATION
  PI_NAME ALL

Exec:
  bias 8Kx3K Bias Frames
  nimgs 10
  blue roi 8Kx3K
  red roi 8Kx3K
  dgo
  nimgs 1

end
```

This sample bias calibration script only has an `Archive:` block and an `Exec:` block. The `BIAS` command automatically sets the exposure time to 0-seconds.

To take dark frames, you would use these commands

```
Exec:
  dark 600s Dark Frames
  exptime 600
  nimgs 3
  blue roi 8Kx3K
  red roi 8Kx3K
  go
  nimgs 1
```

This example takes three 600-second dark frames for both cameras. Note that `exptime` must follow `dark` in the script.

Also note that we have put in an explicit statement of the CCD readout region of interest (ROI), here 8Kx3K. The reason is that “dark” calibrations like bias and darks do not make explicit instrument configuration changes (except those encapsulated in CALMode), and so we must set the CCD readout ROI explicitly to ensure the correct data are acquired.

4 Script Preparation Tools (modsTools)

To help you create good, well-structured MODS scripts, we provide the modsTools script preparation tool package:

`modsProject` – Create a working directory for a new MODS observing project

`mkMODSAcq` – Create a template target acquisition (.acq) script from user inputs.

`mkMODSObs` – Create a template observing (.obs) script from user inputs

`mms2obs` – Create template acquisition and observing scripts from a MODS multi-object slit mask design (.mms) file.

More tools will be added as we gain experience observing with MODS during regular partner science blocks. Ideas for new script preparation tools are always welcome.

4.1 Downloading and installing the modsTools package

The modsTools package works on Linux (CentOS and Fedora) and Mac OS/X 10.6 (Snow Leopard) system. They are written in Perl 5, and do not require any special Perl modules beyond those in standard Perl 5.6 (the version installed on recent versions of Linux and Mac OS/X). The programs are designed to run in a terminal shell.

Download the latest version of the modsTools scripts from

www.astronomy.ohio-state.edu/MODS/ObsTools/modsTools/

This webpage gives links to the latest tar archive of the public release version, installation, use, and release notes. We will also collect a library of example scripts, and have a bug reporting form.

The modsTools package will install itself in its own directory, and you should put that directory into your default login shell's executable path so they can be run from any subdirectory without the need to type the full path. You do not need root or admin privileges to install modsTools.

Updates will be available before each semester early in the MODS science operations phase, so we advise you to check the website above before your observing run or at least at the start of each semester.

4.2 Start a new MODS observing project – modsProject

`modsProject` creates a working project directory and sets up the archive information that will be used in common for all MODS scripts created for the project:

```
modsProject –p partnerID propID
```

`modsProject` will create a `propID` subdirectory, and adopts the name of the directory as the `PROPID` archive keyword. It then ask the user a series of questions to setup the `.modsProject` file that will reside in this directory. This file is used by the other modsTools programs to automatically create scripts with the correct information in the Archive: blocks.

For example, to setup a MODS observing project named OSU_HighZ for an OSURC partner observer, the `modsProject` session would go something like this
(`%` is the shell prompt, user responses to prompts are in **`bold`**):

```
% modsProject –p OSURC OSU_HighZ

** Welcome to the MODS Observing Project Setup Script **

modsProject will create and configure a project directory for
the OSU_HighZ observing project.

This is the directory where you will create all of the MODS
target acquisition and observing scripts for this project.

Is this what you want to do? <Y|N>: Y

Creating OSU_HighZ subdirectory...

   Using LBT Partner: OSURC
   PI Name(s)? Assef, Pogge, Kochanek

The /home/darkstar/pogge/MODS/OSU_HighZ directory has been configured.

Good Luck!

modsProject done.
```

You will now find a new directory name OSU_HighZ in your current working directory. This directory will have a single file in it named `.modsProject` that looks like this:

```
#
# MODS Project File
#
# Created 2011 Jun 12 – 17:40:52 UTC by pogge@darkstar
# Program: modsProject v0.1.2
#

# Parameters for the Archive: Block

PARTNER OSURC
PROPID OSU_HighZ
PI_NAME Assef, Pogge, Kochanek
```

You may, of course, edit this file to make changes (e.g., add a partner or change the PI list), but this is generally discouraged. For example, the `–p` option only lets you give one partner. If you omit this option `modsProject` will prompt you for the PARTNER info so you can enter multiple partner IDs separated by commas.

4.3 Make a target acquisition (.acq) script – mkMODSAcq

This program is run in a MODS project directory created with `modsProject`. It uses the `.modsProject` file and your command-line inputs to create an acquisition script.

Before running `mkMODSAcq`, you need to have the following information in hand:

1. The RA and Dec of your target in J2000 coordinates
2. The RA and Dec of the guide star in J2000 coordinates

3. The rotator position angle (measured in degrees from North thru East).
4. The observing mode (imaging, longslit, or MOS), the slit mask, the camera and filter you wish to use for the acquisition images, and the exposure time.

The `mkMODSAcq` program will take these inputs and create a template .acq script in the current working directory:

```
mkMODSAcq [options] rootName
```

the options are:

```
-o objName     Object name (within 's if objName has spaces in it)
-c 'RA Dec'    Object RA/Dec coordinates, hh:mm:ss +dd:mm:ss, in 's
-p PA          Position Angle in degrees North thru East (default 0)
-g 'RA Dec'    Guide Star RA/Dec coords, hh:mm:ss +dd:mm:ss, in 's
-m obsMode     Observing mode, one of imaging, longslit, or mos
-a [Red|Blue]  Camera to use for acquisition images (default: Red)
-f filterID    Camera filter to use (default depends on -a)
-r ccdROI      Acquisition region of interest (defaults below)
-e [sec]       Acquisition exposure time in seconds (default: 60s)
-s slitID      Slit mask to use (required for -m longslit or mos)
-l width       Long-Slit width in arcsec for -m longslit.
                 current choices are 0.3, 0.6, 0.8, 1.0, and 1.2
```

The default parameters are:

```
Red Camera, 60s and the r_sdss filter [-a red –e 60 –f r_sdss].
If -a Blue, the defaults are 60s and the g_sdss filter.
If -m Imaging or MOS, use a 3Kx3K ROI (full imaging field)
If -m longlist, use a 1Kx1K ROI (central 2-arcmin)
```

Examples:

```
mkMODSAcq -o 'NGC 1234' -c '12:13:14.5 +22:33:44' –g '12:13:16
+22:30:40' -m imaging n1234_img
```

creates an acquisition script named `n1234_img.acq` that uses the red camera to take a single 60-second field acquisition image with the red camera and r_sdss filter, and use the facility imaging stop slit mask at PA=0°.

```
mkMODSAcq -o 'NGC 1234' -c '12:13:14.5 +22:33:44' -p 45 -g '12:13:15.6
+22:30:40' -m longslit -l 0.6 n1234_ls
```

creates `n1234_ls.acq` that uses longslit mode with the 0.6-arcsec wide longslit mask oriented along PA=45° and takes a sequence of 60-second thru-slit and field images of the target with the red camera and r_sdss filter. If instead you typed

```
mkMODSAcq -o 'NGC 1234' -c '12:13:14.5 +22:33:44' -p 45 -g '12:13:15.6
+22:30:40' -m longslit –a blue -l 0.6 n1234_ls
```

it would take the acquisition images with the blue camera and the g_sdss filter.

```
mkMODSAcq -o 'NGC 1234 Std' -c '12:13:14.5 +22:33:44' -p 45 -g
'12:13:15.6 +22:30:40' -m longslit –s LS60x5 –e 10 n1234_wide
```

will acquire the target through the LS60x5 spectrophotometric 5-arcsec wide-slit with the default red camera and r_sdss filter, but with a 10-second acquisition snapshot exposure time, creating the `n1234_wide.acq` script.

If you give the command without the `-g` flag to define the guide star coordinates, the .acq file will contain a blank `GUICOORDS` line that you must remember to fill in later when you select a guide star. However, we do not recommend doing this as a matter of habit, for if you forget to fill in that line, the acquisition will fail at the telescope.

The defaults built into `mkMODSAcq` were developed during instrument commissioning, and most observers will be able to these .acq scripts as-is with little or no further modification. At the very least they will provide you with a good starting point for most imaging and long-slit target acquisition scripts.

While you can prepare multi-object target acquisitions with this program, most observers will find the `mms2obs` program (see §4.5) much more convenient to use since most of the information needed for the `mkMODSAcq` command line is already embedded within an MMS mask design file.

4.4 Make an observing (.obs) script – mkMODSObs

Unlike acquisition scripts, which follow a prescribed sequence in most cases, observing scripts cover a vast range of possibilities. For most routine long-slit and MOS spectroscopic observations, however, observing scripts should be simple.

The `mkMODSObs` program will generate observing scripts that can be executed as-is, or that may be used as templates for crafting more complex observing scripts – the files created by `mkMODSObs` will have all of the necessary boilerplate in the proper format, and your custom observing procedure should only require you to edit the Exec: block.

The command syntax for `mkMODSObs` is:

```
mkMODSObs [options] rootName
```

the options are:

```
-o objName      Object name (within 's if objName has spaces in it)
-m mode config  instrument mode {dual|blue|red} and channel config
                   {imaging,grating,prism} to use
-s maskID       Slit mask to use (no defaults)
-l width        Long-Slit width in arcsec (alternative to –s).
                  current choices are 0.3, 0.6, 0.8, 1.0, and 1.2
-e [sec]        Base exposure time in seconds
-n [#imgs]      Number of images to acquire
-bfilter [ID]   Blue-channel filter to use
-rfilter [ID]   Red-channel filter to use
-filter [ID]    If mode=red|blue, shorthand for rfilter/bfilter.
```

Note that the filter options would only be used if you were making a template imaging script or choosing a non-standard spectral filter (the default filters optimized for each channel for the spectroscopic modes are part of the mode and configuration selections made via the `-m` option so you don't have to specify the standard spectral-mode filters explicitly).

For example, a dual-channel grating spectrum of a target of 3600s total integration divided into three 1200s exposures through a 0.8-arcsec slit would be created with

```
mkMODSObs mrk573 –o 'Mrk 573' –m dual grating –l 0.8 –e 1200 –n 3
```

This would create the file `mrk573.obs` in the current working directory. Paired with a suitable .acq file created with `mkMODSAcq`, you have everything you need to execute the observation.

A number of example observing scripts are given in §6.

4.5 Make MOS observing scripts from MMS mask files – mms2obs

Because most of the information needed to construct acquisition and observing scripts already resides in the MODS mask specification (.mms) file for the target field, the `mms2obs` program is provided to create the .acq and .obs script templates for an MOS observation with only a little additional input:

```
mms2obs [options] rootName
```

the options are:

```
-o objName     Long form of the object name in single quotes.
-c [Red|Blue]  Camera for acquisition (default: Red)
-f [filterID]  Camera filter to use
-t [sec]       Acquisition exposure time (default: 60seconds)
-m mode disp   Instrument mode (dual|blue|red) and dispserer
                 disp=(grating|prism) for the science observations
-e expTime     Base science exposure time in seconds (no default)
-n numExp      Number of science exposures to acquire (default 1)
```

To use `mms2obs`, you need to have a copy of the mask .mms file for your target in the working project directory created using `modsProject`. In general, the .mms and observing scripts should be bundled together, as the .mms files are also needed to run the `modsAlign` program at the telescope.

For example, an MOS mask file `a1689.mms` is copied into the working project directory, and you wish to take 3600s of total integration in dual-grating mode, divided into four 900 second integrations. The `mms2obs` command to build the necessary acquisition (.acq) and observing (.obs) scripts would be:

```
mms2obs a1689 -o 'Abell 1689 Field 1' -c red -f r_sdss -t 60 -m dual
grating -e 1200 -n 3
```

This reads `a1689.mms` and creates two scripts: `a1689.acq` and `a1689.obs` that would be ready to execute as-is. Note that all of the target, guide star, rotator, and mask information needed for the acquisition and observing scripts is taken directly from the .mms file. Scripts created using `mms2obs` have been successfully used as-is at the telescope.

5 Observing with MODS Scripts

At the telescope, the MODS acquisition and observing scripts are executed using the following programs:

`acqMODS` – execute an acquisition (.acq) script

`execMODS` – execute an observing (.obs) or calibration (.cal) script

Additional MODS script, observing, and command tools are

`modsAlign` – align a target with the slit mask using acquisition images taken with an .acq script (computes and executes the telescope offset and rotation).

`listMODS` – list the contents of an MODS script, providing line numbers and block labels for script restart.

A common feature of all of these programs is that if invoked without command arguments they will print a full usage message reminding you of the command syntax and options.

IMPORTANT:

At the telescope, `acqMODS` and `execMODS` require that the MODS Control Panel GUI is running and the instrument initialized and ready. Once the instrument and telescope are live, **one and only one observing console and account must be used to run these programs.** Attempting to execute scripts from two different consoles will result in command conflicts and confusion, the risk being loss of time rather than damage to the instrument, but no safety measures are 100% guaranteed.

For safety these programs should never be executed by a remote login except under locally supervised circumstances.

5.1 acqMODS – Execute a target acquisition (.acq) script

`acqMODS` executes a target acquisition script, pointing the telescope and taking acquisition images.

```
acqMODS [options] file.acq
```

the options are:

```
-n    dry-run the acquisition w/o execution (alias --dryrun)
-a    only run the acquisition image sequence (alias --acquire)
-p    only load coordinates and preset the telescope (alias --preset)
-i    only configure the instrument (alias --inst)
-l    only load target coordinates but don't execute the preset
        (alias --load)
```

The options provide ways to execute only portions of the script, but generally `acqMODS` is run without options. For example:

```
acqMODS ngc1234_ls.acq
```

will execute the named acquisition script from start to finish. Acquisition progress will be printed to the terminal screen.

If there are faults, `acqMODS` will print the error message and give you the option to abort, retry, or ignore the error:

`abort` – stops the acquisition and returns to the shell prompt.

`retry` – attempts to re-execute the command that gave the error.

`ignore` – will ignore the error and execute the next command.

In a typical long-slit or multi-slit target acquisition, after the thru-slit and field images are taken, the script will pause and ask you to run the `modsAlign` program to align the targets with the slit mask (see §5.2). On resuming, the usual practice is to take a final, confirmatory thru-slit image to make sure you got the target(s) in the slit(s).

If something goes wrong at the alignment step, the usual response is to re-execute just the acquisition block of the script, rather than going through the considerable overhead of a full preset and instrument configuration. For example:

```
acqMODS –a ngc1234_ls.acq
```

will just execute just the Acquire: block of the script and re-take the thru-slit and field images and start over again with `modsAlign` again.

This is, however, inefficient, as the initial thru-slit image should not change much. A better way is to use MODS Control Panel's instrument dashboard GUI to extract the slit mask and take a new field image, then run `modsAlign` again, send a new offset, and put the mask back in and snap a confirmatory thru-slit image.

Yet another way is to use the low-level `modsCmd` command, which lets you send single MODS commands from the terminal shell. For example, to take just a field acquisition image, you would execute these two commands from the shell:

```
% modsCmd slitmask out
% modsCmd red go
```

(or, if you were using the blue camera for acquisition, "`blue go`"). You could then use the original thru-slit image and the new field image to run `modsAlign` again. To take a final, confirmatory thru-slit image, you would type these commands:

```
% modsCmd slitmask in
% modsCmd red go
```

and the next image will be through the mask.

Where possible you should use the GUI when you have to drop down to "manual" operation.

5.2 modsAlign – Align a slit mask with target objects

`modsAlign` is a standalone Python program that will interactively align your targets with your slit mask. It launches a private, named ds9 image display and uses the IRAF pyraf interface to compute the offset and/or instrument rotational offset needed to align the mask with the targets, and executes the offset.

There are two modes: long-slit and multi-slit.

Once an acquisition script pauses after taking the thru-slit and field images, you need to copy these images into the same working directory as the .mms files (if doing a multi-slit mask acquisition). The raw data files will be in the /newdata/ archive staging directory, and their filenames should be printed in the `modsDisp` monitor window (assuming you are running

`modsDisp`, which you should be). The usual practice is to copy the raw images from /newdata/ and give them simple names to readily identify them, for example:

```
cp /newdata/mods1r.20110527.0011.fits n1234_slit.fits
cp /newdata/mods1r.20110527.0012.fits n1234_field.fits
```

What you do next depends on whether this is a long-slit acquisition or a multi-object acquisition.

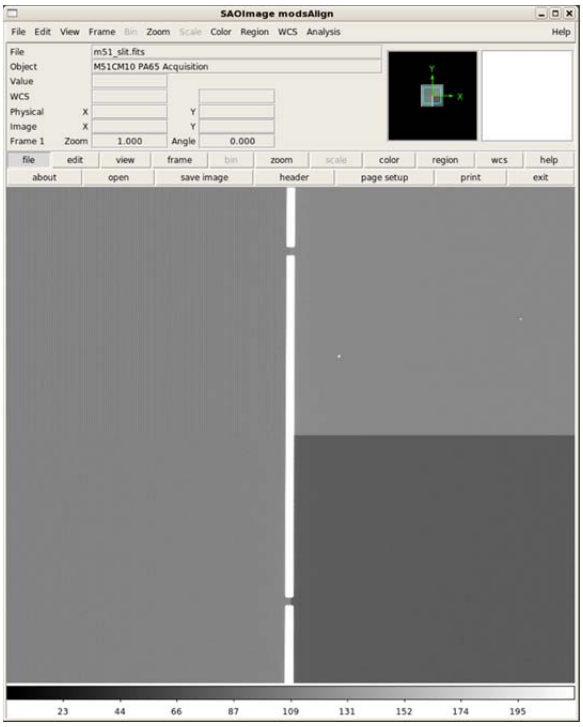
5.2.1 Long-Slit Target Alignment

To align a target with a long-slit, run `modsAlign` with the `-l` (letter "el" for "longslit") option:

```
modsAlign -l n1234_slit.fits n1234_field.fits
```

The order of images matters: thru-slit then field image.

`modsAlign` will launch a dedicated ds9 window if none is already open, and display the thru-slit image:



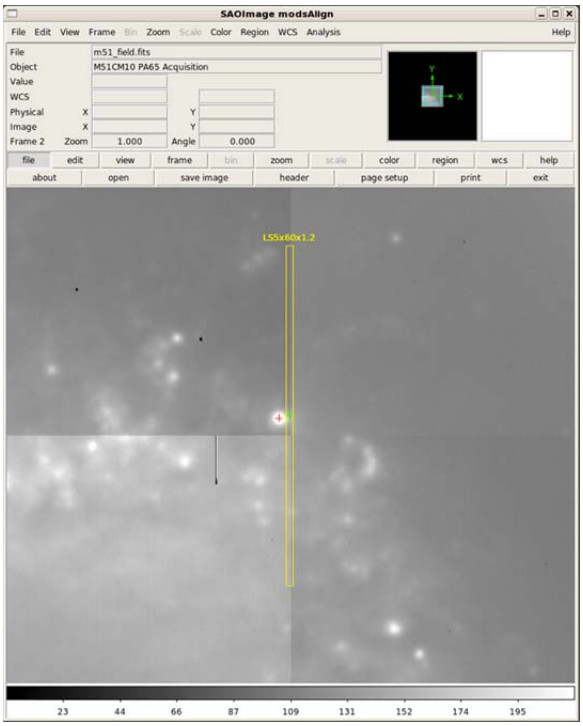
`modsAlign` will put the cursor on the ds9 display and ask you to mark the desired slit position for the target.

Mark this by putting the cursor on the desired position (to the nearest pixel) and type the "`x`" key to mark it

Type "`q`" when finished to proceed to the next step.

After getting the cursor XY position, `modsAlign` will compute the X-axis slit center from the image data and take the Y-axis (along slit) position of the cursor as given.

It will then display the field image and overlay a box depicting a slit segment using the position just computed (it gets the slit mask ID from the image header), marking the desired slit position with a green +, as shown below:



`modsAlign` will then put the `imexamine` cursor on this display and ask you to mark the target object:

Type the "`a`" key to measure the centroid if the target is bright enough for `imexamine` to get a good centroid.

If the target is too faint or too saturated for centroiding, mark it to the nearest pixel with the "`x`" key.

Type "`q`" when done.

The target position (centroid or nearest pixel) will be marked with the red + on completing this step.

`modsAlign` then computes the DETXY coordinate telescope offset that will move the target into the desired slit position, and asks if you want to send the offset to the telescope.

Below is the screen output of the `modsAlign` run for the above images:

```
% modsAlign -l m51_slit.fits m51_field.fits


** Mark where you would like the object on the slit with 'x'.
   and then press 'q' when finished.
Log file _modsalign.log open
z1=213.1422 z2=445.09
 512.00  541.00   1823. 1842.0
z1=187.0711 z2=2784.

** Mark the target with an 'a' (or an 'x' if faint or saturated)
   and then press 'q' when done

Log file _modsalign.log open
#   COL    LINE   COORDINATES
#     R    MAG  FLUX   SKY  PEAK  E   PA BETA ENCLOSED MOFFAT DIRECT
 495.94  537.31 495.94 537.31
  40.83  10.05 958573. 1704. 2349. 0.09 81 3.31  20.95  14.12  22.79

Computed Slit Alignment Offset:
   dX:   1.852 arcsec
   dY:   0.454 arcsec

MODS Offset Command:
   offsetxy  1.852  0.454 rel

Send the offset to the Telescope (Y|N)?:
```

Answering `Y` will send the offset to the telescope, which can take a few seconds to execute and relock guiding. On return, you can resume the acquisition script and take a confirmatory thru-slit image to make sure your target is where you want it.

If you answer `N` at the prompt no offset will be sent. This is what you do if the program computes a bad offset. Common causes of bad offsets are failure to centroid the target (too faint or too saturated), or failure to find the slit center (slit image too short and sky too faint).

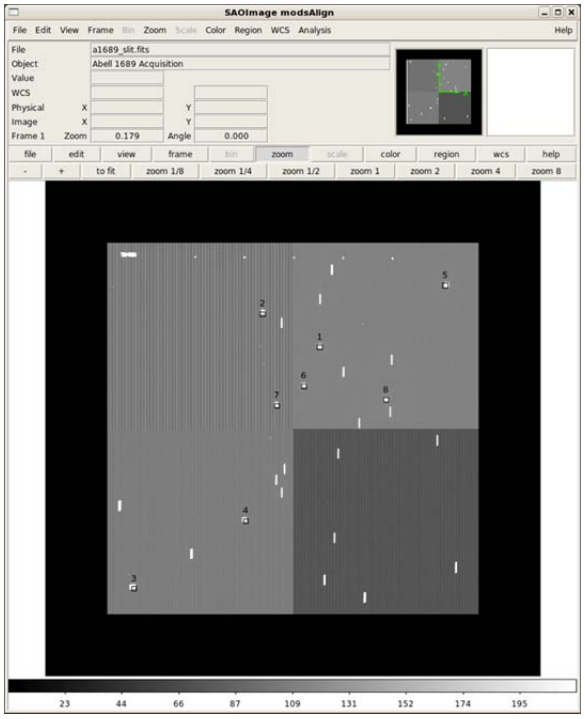
5.2.2 Multi-Slit Target Alignment

To align targets with a multi-slit mask, you use reference stars to determine the offset and rotation required. In this case you run `modsAlign` in its full mode, giving it the name of the .mms mask file and names of the thru-slit and field images:

```
modsAlign n1234.mms n1234_slit.fits n1234_field.fits
```

The order of files matters: mms then thru-slit image then field image.

`modsAlign` will first display the thru-slit image, overlaying the positions of the alignment star boxes, numbering each box:



The reference star box positions shown are where it computes they should be from the coordinates given in the .mms file.'

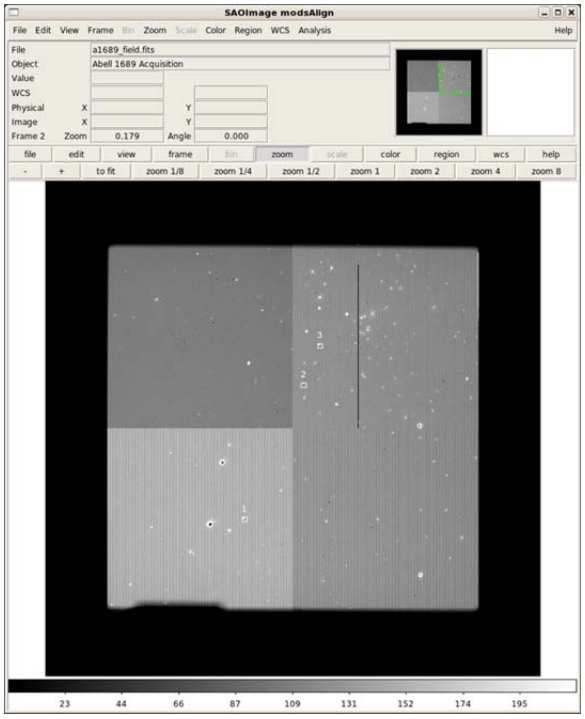
The calculated positions will usually be displaced by 1-2 arcsec or so from the thru mask image.

You are then instructed to mark at least two of the alignment boxes, in any order, by typing the "`x`" key.

Type "q" when done to move to the next step.

`modsAlign` will measure the precise centers of the reference star boxes in the thru-slit image and compute the transformation between the .mms file mask coordinates and CCD pixel coordinates.

It then displays the field image with the your selected alignment star boxes put in the correct locations. This time it shows only the boxes you selected, and renumbers them starting with 1 in the order you picked them (it discards the original reference star box numbers at this step):



The selected reference star boxes are drawn in white.

modsAlign puts the IRAF imexamine cursor on the images and instructs you to measure each of the N reference stars on the field image by typing the "a" key.

These must be marked in number order 1..N as shown in the ds9 window.

Once you have finished measuring the stars, type the "q" key to proceed to the next step.

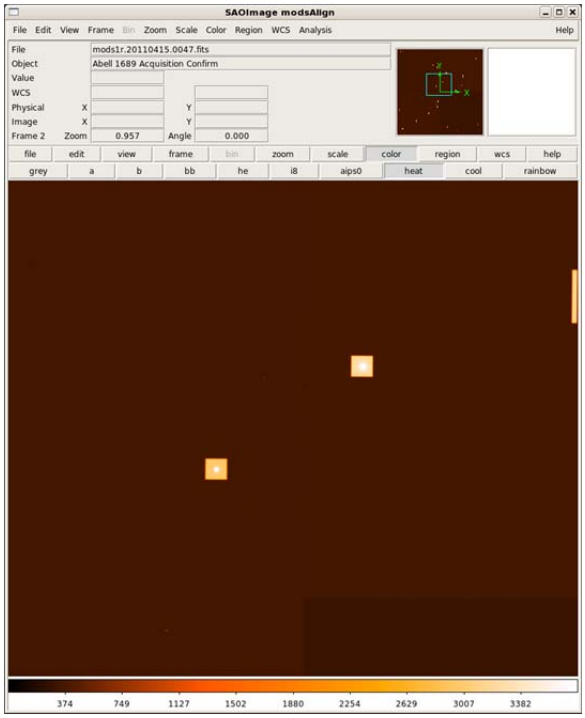
`modsAlign` then computes the XY offset and rotation offset needed to align the mask with target field, displays the results on the ds9 window by putting crosses on where the reference stars will land after the offset, and asks if you want to send offset to the telescope.

Here is an example run that was used for the images above up to the "send it?" prompt:

```
% modsAlign a1689.mms a1689_slit.fits a1689_field.fits

z1=217.2555 z2=452.9805

** Mark Alignment Star Boxes to use with a 'x'.
   and then press 'q' when finished.
   *** You must mark at least two (2) boxes ***

Log file _modsalign.log open
1135.26  784.17   3639.
1625.67 1898.73   3315.
1759.41 2221.95   3399.
z1=1469.197 z2=4442.

** Mark the center of each alignment star with an 'a'
   and then press 'q' when finished.
   Stars must be selected in the *SAME ORDER* as the boxes.

Log file _modsalign.log open
1129.46  798.01 1129.46 798.01
  21.92  11.42 269842. 3557. 4067. 0.10 28 5.77   6.11   7.76   7.26
1621.39 1909.75 1621.39 1909.75
  21.96  12.34 115742. 3200. 1127. 0.17 -4 2.84   7.37   7.66   7.32
1754.89 2233.01 1754.89 2233  1
  16.01  10.37 709896. 3190. 15990. 0.03  -47 2.55  5.50  5.42   5.34

Computed Mask Alignment Offset:
```

```
    dX:   0.713 arcsec
    dY:  -1.545 arcsec
   dPA: -0.0717 degrees

MODS Offset Command:
   offsetpointing -0.0717  0.713 -1.545 detxy rel
Send the offset to the Telescope (Y|N)?:
```

Resuming the paused acquisition script, a confirmatory thru-slit image will be acquired. You can then display this image in a ds9 window. If the offset was successful, you you should see the reference stars centered in the alignment boxes, as shown below:



Confirmatory thru-slit image after the offset in XY and rotation was applied. A color scale was added in a vain attempt at clarity.

At this point you are ready to execute the multi-slit observations with confidence that you have all your targets aligned with the slits.

5.3 execMODS – Execute an observing (.obs) or calibration (.cal) script

`execMODS` executes an observing (.obs) or calibration (.cal) script, configuring the instrument and acquiring science or calibration data as indicated. It can also execute instrument setup procedure (.pro) scripts.

The command syntax is:

```
execMODS [options] scriptFile
```

Here `scriptFile` may be a .obs file, a .cal file, or a .pro file. **`execMODS` will not run target acquisition (.acq) files.**

The command options are:

```
-n       dry-run the acquisition w/o execution (alias --dryrun)
-r N     run the script starting from command number N
          (alias –runform; use listMODS to get command numbers)
-e       run from the Exec: block to the end (alias --exec)
-f name  run from the name: block to the end (alias --fromblock)
-b name  run only the name: block commands   (alias --runblock)
-u       run unattended and abort on errors  (alias --unattended)
```

The options provide ways to execute only portions of the script, or restart from any where in the script after clearing a fault, or doing a quick dry-run to see what the script will do. More often `execMODS` will be run without options. For example:

```
execMODS ngc1234_ls.obs
```

will execute the named observing script from start to finish. The instrument setup and data-acquisition progress will be printed to the terminal screen.

If there are faults, `execMODS` will print the error message and give you the option to abort, retry, or ignore the error, same as `acqMODS`. A difference is that it also prints the command number where the fault occurred, so you could restart the script from that command after clearing the fault (provided the fault wasn't with the script proper).

If the `–u` (`--unattended`) option is given, `execMODS` will abort on all faults rather than prompting for abort/retry/ignore. This option must be used if running a calibration (.cal) script unattended by an observer (e.g., starting biases or calibration lamp spectra then going to bed). This ensures that if there are problems, the instrument should be able to safe itself given nobody is around to do that. Unattended operation should be done with forethought as to what will happen if things go wrong – in particular, never run a new, untested, or unproven script in unattended mode.

5.4 Restarting an Observing or Calibration Script

MODS observing scripts are highly re-entrant: it is possible to restart an observing script from any line or functional block, and you can run just the commands in a single functional block if needed. This property enables flexible and rapid resumption of observing after clearing fault conditions without requiring observers to modify an observing script at the telescope.

The `listMODS` program shows you the contents of an observing or calibration script as `execMODS` sees it, giving the breakdown into functional blocks and assigning command numbers (the order in which commands will be executed from start to finish).

For example:

```
% listMODS J1420.obs

*** Listing of J1420.obs ***

   1: OBSMODE
   2: *nimgs 1*
Archive:
   3: PARTNER OSURC
   4: PROPID OSU_HighZ
   5: PI_NAME DeRosa
```

```
Instrument:
   6: instconfig dual grating
   7: *imcslock*
   8: slitmask LS5x60x1.2
Exec:
   9: object J1420-1620
  10: exptime 1200
  11: nimgs 6
  12: dgo
  13: end
```

A few things to note:

1. Blank line and comments are not listed (they are ignored in the same way `execMODS` ignores them)
2. Functional block labels are noted so you can see the breakdown of the script into functional blocks, but the labels themselves are not numbered.
3. A few commands in the listing begin and end with asterisks (*), as in lines 2 and 7 in the example above. These are examples of hidden commands that take care of some of the subtle details of how MODS operates (here, it makes sure the number-of-images is initialized as 1, forgetting what may have been done by a previous script, and it puts in an explicit `imcslock` command after setting the instrument configuration). There are very few instances of hidden commands in scripts and these are the two most common.

`listMODS` is designed to only work with observing, calibration, and procedure scripts – acquisition scripts are different and `listMODS` will not work with .acq scripts.

You have three options for restarting a script:

1. Restart from a command number (`-r` or `--runfrom`)
2. Restart from a functional block label (`-f` or `--fromblock`)
3. Run only the contents of a single block (`-b` or `--runblock`)

5.4.1 Restart from a command number

When a fault occurs, `execMODS` reports the fault condition, the command being run when the fault occurred and its command number. If you elect to abort, you can restart from that command number after clearing the fault by typing

```
execMODS –r 11 J1420.obs
```

Command numbers are not the same as line numbers: use `listMODS` to list the command numbers if you forgot to note the command number from the error message. In the example at the start of this section, the command above would restart execution with command 11 (`nimgs 6`).

5.4.2 Restart from a functional block label

To restart a script beginning with the first command in a functional block and continuing to the end of the script use the `–f` (`--fromblock`) option:

```
execMODS –f Instrument J1420.obs
```

Here the script will restart with the first command of the Instrument: block in J1420.obs and continue on until the end. Note that functional block labels are **case-insensitive** (see §3.1.2), **and that the colon (:) is omitted**. The colon is not part of the label name.

Since the most common recovery mode is to re-execute the Exec: block, we provide a convenience option, `-e`:

```
execMODS –e J1420.obs
```

that is equivalent to

```
execMODS –f Exec J1420.obs
```

5.4.3 Run only the commands within a functional block

The `–b` (or `--runblock`) option will run only the commands *within* a single functional block and then stop when the script reaches the next functional block or the end of the script, whichever comes first. For example

```
execMODS –b Instrument J1420.obs
```

will only run the Instrument: block, and stop before the start of the Exec: block in this example.

5.5 Scripts of Scripts

Is it possible to run a number of scripts one after the other by embedding them within a shell script (bash, csh, etc.). For example, if you want to run three calibration scripts: bias3k.cal, bias8k.cal, and prbias.cal, you would create a shell script (this example is a csh script) that looks like this (called `doBias`)

```
#!/bin/csh
execMODS bias3k.cal
execMODS bias8k.cal
execMODS prbias.cal
exit 0
```

You would then run it from the shell by typing

```
% chmod +x doBias
% ./doBias
```

and away it would go. Technically, the shell script above is sloppy: there is no error trapping so that if any of the .cal scripts experienced a fault, it would freeze at an abort/retry/ignore prompt. A better implementation would be

```
#!/bin/csh
execMODS –u bias3k.cal
execMODS –u bias8k.cal
execMODS –u prbias.cal
exit 0
```

which would run the three calibration scripts in unattended (-u) mode. It is still kind of sloppy, as if any script throws a fault, it will move on and try the next script. This could lead to a fault cascade, with all the attendant difficulty that might involve.

In general, scripts of scripts should be used with care, and for standard calibrations we have a number of robust, tested examples available, and we will develop a library of such scripts as we gain experience with the instrument. Ask the support scientists for details.

6 Example Scripts

The following are examples of MODS scripts taken from among those successfully run during MODS1 commissioning and early science verification observing. They should give you an idea of what can be done with scripts and serve as good templates of working scripts for preparing your own MODS observations.

Copies of all of these example scripts are available at

www.astronomy.ohio-state.edu/MODS/ObsTools/modsTools/Examples/

Caveat: if you jumped to this section without reading the rest of the document first, you really should read the manual before jumping in here.

6.1 Long-Slit Target Acquisition and Observing Scripts

This pair of scripts acquire a QSO pair and takes three 1200s dual-mode long-slit grating spectra through the 0.8-arcsec segmented long slit. The trick to this is having excellent astrometry so there is no doubt about the slit position angle (Note: to protect the original, as-yet uncompleted MODS science verification project, I've changed the target name, coordinates, and rotator PA).

qsopair.acq:

```
#
# MODS Target Acquisition File
#

ACQMode SLIT

Archive:
  PARTNER LBTB
  PROPID  MPIA_QSOPair
  PI_NAME Hennawi

Target:
  OBJNAME Close QSO Pair
  OBJCOORDS 21:50:36.187 +37:14:00.99
  GUINAME GStar
  GUICOORDS 21:50:46.717 +37:12:43.20
  ROTATOR 52.6 POSITION
  PRESET ACTIVE

Instrument:
  SLITMASK LS5x60x0.8
  ACQCamera Red
  ACQFilter r_sdss
  ACQexptime 60
  ACQROI 1Kx1K

Acquire:
  SlitGO
  AcqGO
  print Align the slit using modsAlign -l
  PAUSE
  SlitGO
```

```
end
```

qsopair.obs:

```
#
# MODS Observing Script
#
# Created: 2011 May 29 at 07:23:53 UTC by LBTO@obs2
# Program: mkMODSObs v0.4.5
#

OBSMODE

Archive:
  PARTNER LBTB
  PROPID MPIA_QSOPair
  PI_NAME Hennawi

Instrument:
  instconfig dual grating
  slitmask LS5x60x0.8

Exec:
  object Close QSO Pair
  exptime 1200
  nimgs 3
  dgo

end
```

6.2 Multi-Object Spectrum Acquisition and Observation

This is for observations in the Bootes field. The mask was created by `MMS` as `Bootes.mms`, and then the .acq and .obs files were generated using `mms2obs` (§4.5). The command was

```
mms2obs Bootes –o 'Bootes MOS Field' –e 1200 –n 3 –t 60
```

We found a problem with the original guide star and selected another. We've left the original commented out in the `Target:` block for reference. Also, we edited the .obs file after generation to change the OBJECT parameter to something more sensible than the cryptic information created by MMS. The `Bootes.mms` file was used to create the mask.

Bootes.acq:

```
#
# MODS Target Acquisition File
#
# Created: 2011 May 29 - 05:24:44 UTC by LBTO@obs2.lbt.as.arizona.edu
# Program: mms2obs v0.4.2
# Mask File: Bootes.mms
# Mask ID: M012bianmods+524713
#

ACQMode MOS

Archive:
  PARTNER LBTSDT
```

```
  PROPID SV_Bian
  PI_NAME Bian

Target:
  OBJNAME Bootes MOS Field
  OBJCOORDS 14:32:45.560 +34:07:24.700
  GUINAME GStar
  GUICOORDS 14:32:26.861 +34:07:02.20
  # GUICOORDS 14:32:31.262 +34:04:32.030
  ROTATOR 75.000 POSITION
  PRESET ACTIVE

Instrument:
  SLITMASK ID524713
  ACQCamera Red
  ACQFilter r_sdss
  ACQexptime 60

Acquire:
  SlitGO
  AcqGO
  print Align the mask using modsAlign...
  PAUSE
  SlitGO

end
```

Bootes.obs:

```
#
# MODS Observing Script
#
# Created: 2011 May 29 - 05:24:44 UTC by LBTO@obs2.lbt.as.arizona.edu
# Program: mms2obs v0.4.2
# Mask File: Bootes.mms
# Acquisition Script: Bootes.acq
#

OBSMODE

Archive:
  PARTNER LBTSDT
  PROPID SV_Bian
  PI_NAME Bian

Instrument:
  instconfig dual grating
  slitmask ID524713

Exec:
  object Bootes MOS Field
  exptime 1200
  nimgs 3
  dgo

end
```

6.3 Flux Standard Star Acquisition and Observation

These are for observations of the spectrophotometric flux standard star GD140. We are using the wide LS60x5 5-arcsec wide "fat slit" for the observations to avoid any slit losses or problems due to atmospheric dispersion. LBTO will have a library of standard star acquisition and observing scripts for general use, with exposure times worked out during commissioning. These scripts were written by hand.

gd140.acq:

```
#
# GD140 at PA=180 and the Spectrophotometric 5-arcsec slit
#
# There are no guide stars in the patrol field for PA=0
#
# Takes a single thru-slit image, if we are far off, we can
# follow with an acq snap
#
# Modifications: 2011 Mar 20 [rwp/osu]
#

ACQMODE MOS

Archive:
  partner CALIBRATION
  propid  CALIBRATION
  pi_name Pogge

Telescope:
  OBJNAME GD140 PA180
  OBJCOORDS 11:37:05.613 +29:47:59.33
  GUINAME GStar
  GUICOORDS 11:37:07.524 +29:50:45.49
  ROTATOR 180 POSITION
  PRESET ACTIVE

Instrument:
  SLITMASK LS60x5
  ACQCamera Red
  ACQFilter r_sdss
  ACQexptime 1
  ACQROI 1Kx1K

Acquire:
  ACQexptime 30
  SlitGO
  print If star is in slit, abort, otherwise resume and snap field
  PAUSE
  ACQexptime 1
  AcqGO

end
```

Here we get a little tricky. This is designed to take calibration spectra in dual grating and red-only grating modes, so we have omitted the instrument: block and inserted two custom

functional blocks under the Exec: block that will change configuration and take two sets of spectra.

gd140.obs:

```
#
# GD140 Dual+Red for 2011 May SV
#
# Shoots dual and red-only grating configs for
# the standard star.
#

obsmode

Archive:
  partner CALIBRATION
  propid  CALIBRATION
  pi_name Pogge

Exec:
  slitmask LS60x5

DualGrating:
  instconfig dual grating
  object gd140 dual grating
  exptime 180
  go

RedGrating:
  instconfig red grating
  red object gd140 red grating
  red exptime 180
  red go

end
```

6.4 Bias Calibration Frames

This takes 8Kx3K, 3Kx3K, and 4Kx3K bias images in both channels, 10 frames per configuration. These three readout modes are the defaults for grating spectra, imaging, and prism spectra, respectively.

allbias.cal:

```
#
# MODS All-Mode Bias Calibration Script
#
# Acquires 10 each Red and Blue CCD bias images for
# 1) 8Kx3K Grating Spectra ROI unbinned
# 2) 3Kx3K Imaging ROI unbinned
# 3) 4Kx3K Prism Spectra ROI unbinned
#
# Execution Time: 35 minutes for the entire set
#
# R. Pogge, OSU Astronomy Dept
# pogge@astronomy.ohio-state.edu
```

```
# 2010 Dec 16
#

CALMode

Archive:
  PARTNER CALIBRATION
  PROPID  CALIBRATION
  PI_NAME ALL

Exec:
  Bias8K:
  print 8Kx3K Grating Spectral Bias Frames
  bias Bias 8Kx3K Grating Mode
  nimgs 10
  blue roi 8Kx3K
  red roi 8Kx3K
  dgo
  nimgs 1

  Bias3K:
  print 3Kx3K Imaging Bias Frames
  bias Bias 3Kx3K Imaging Mode
  nimgs 10
  blue roi 3Kx3K
  red roi 3Kx3K
  dgo
  nimgs 1

  PrBias:
  print 4Kx3K Prism Spectral Bias Frames
  bias Bias 4Kx3K Prism Mode
  nimgs 10
  blue roi 4Kx3K
  red roi 4Kx3K
  dgo
  nimgs 1

end
```

Here, because we may not know which mode the instrument was left in (dual or single-channel), we use the explicit "`DGO`" command which takes simultaneous images in both channels.

It is not required to put each bias group into its own functional block, but doing so helps the overall organization, and will let us go back in and re-run a functional block if, for example, there was a problem found later and we need to re-run PrBias (e.g., `execMODS –b PrBias allbias.cal`).

6.5 Grating Wavelength Calibration Lamps

This script takes a full set of wavelength calibration lamps for the dual-grating mode. It uses pairs of lamps, filters, and exposure times that let us keep both channels active and run efficiently. We only need one spectrum of each lamp as the entire lamp ensemble will give you more than 100 lines in each channel, and exposures are so short cosmic ray hits are not an

issue. Spectral ND1.5 filters are used to keep from saturating all but the very brightest lines. This script, or a variant on it, is the recommended wavelength calibration set for the dual grating mode.

grlamps.cal:

```
#
# Dual Grating Mode Lamps
#
# Sequence of Comparison Lamp spectra taken
# with the LS5x60x0.6 long slit in dual mode
# and the grating dispersers
#
# R. Pogge, OSU Astronomy Dept.
# pogge@astronomy.ohio-state.edu
# 2010 Dec 17
#
#------------------------------------------------------------

calmode

Archive:
  PARTNER CALIBRATION
  PROPID  CALIBRATION
  PI_NAME ALL

Instrument:
  instconfig dual grating
  slitmask LS5x60x0.6

Exec:

NeHgLamps:
  # Ne + Hg Lamps - wait 30s for Hg to warm up
  # ND1.5 filter in both channels.
  print Ne+Hg Lamp Spectra
  lamp hg ne on
  print Waiting 30s for the Hg lamp to warm up
  sleep 30
  comp Ne + Hg[Ar] Lamps
  blue exptime 2
  blue filter ND1.5
  red exptime 1
  red filter ND1.5
  nimgs 1
  dgo
  lamp off

KrXeLamps:
  # Kr+Xe Lamps - Red: ND1.5 1s, Blue: Clear 15s
  print Kr+Xe Lamp Spectra
  lamp xe kr on
  comp Xe+Kr Lamps
  blue exptime 15
  blue filter clear
  red exptime 1
  red filter nd1.5
```

```
  nimgs 1
  dgo
  lamp off

ArLamps:
  # Argon Lamp - Red: ND1.5 1s, Blue: Clear 30s
  print Ar Lamp Spectra
  lamp ar on
  comp Ar Lamp
  blue exptime 30
  blue filter clear
  red exptime 1
  red filter nd1.5
  nimgs 1
  dgo
  lamp off

end
```

This has a number of features to note:

1. It uses a print statement to inform the observer as to which set of calibrations are in progress.
2. It explicitly sets the number of images (nimgs) for each set
3. It explicitly turns off all lamps for each set.

All of these, while redundant, make for safe, informative scripts.

6.6 ugriz Photometric Standard Star Field

This takes ugriz images in dual imaging mode of a standard star field. There is a brief target acquisition script, then an observing script that goes through the different filters with different exposure times per filter. Note that because this script works with the instrument open to the sky, it is OBSMode not CALMode, and we give it the .obs file extension to remind us it should be paired with a suitable .acq script.

photcal.obs:

```
#
# photcal – ugriz images of a photometric
# standard field
#
# Fill in the field name under OBJECT below
# in the Exec: block
#

obsmode

Archive:
  PARTNER CALIBRATION
  PROPID  CALIBRATION
  PI_NAME Pogge

Instrument:
  instconfig dual imaging
  slitmask imaging
```

```
Exec:
  OBJECT M71B SDSS Field
  # 2x(g+r)
  nimgs 2
  blue filter g_sdss
  blue exp 30
  red filter r_sdss
  red exp 30
  dgo

  # u + 2xi
  blue filter u_sdss
  blue exp 60
  blue nimgs 1
  red filter i_sdss
  red exp 30
  red nimgs 2
  dgo

  # u + 2xz
  red filter z_sdss
  red exp 30
  red nimgs 2
  dgo

nimgs 1

end
```

The exposure times are longer for the u filter, and we mix them up to keep the amount of open shutter time as high as possible. It is not really possible to balance exposure and readout times given the short exposures needed on standard fields with an 8.4m telescope, and it is pointless to try for a gain of only a few 10s of seconds for what are minutes of execution time.

This is an example of a script "stub", a basic script that does the same thing, but can be edited to change the object name and then pair it with .acq scripts created using mkMODSAcq. For a lot of repetitive observations, this will likely be a common practice.

6.7 Long Slit Spectra of a QSO dithering along the slit

In this script we acquire a QSO, then take 4 dual-channel grating spectra of 600sec each through the 0.8-arcsec segmented long slit, dithering by 10-arcsec either side of the original position and back. We will use absolute offsets in DETXY coordinates since our slit is oriented approximately along the parallactic angle for the time of this observation

Q1537.acq:

```
#
# MODS Target Acquisition File
#

ACQMode SLIT

Archive:
  PARTNER OSURC
```

```
  PROPID OSU_C4QSOs
  PI_NAME Kochanek, Pogge, Peterson, Assef

Target:
  OBJNAME Q1537-0146
  OBJCOORDS 15:37:25.36 -01:46:50.3
  GUINAME GStar
  GUICOORDS 15:37:17.427 -01:49:13.94
  ROTATOR 20 POSITION
  PRESET ACTIVE

Instrument:
  SLITMASK LS5x60x0.8
  ACQCamera Red
  ACQFilter r_sdss
  ACQexptime 60
  ACQROI 1Kx1K

Acquire:
  SlitGO
  AcqGO
  print Align the slit using modsAlign -l
  PAUSE
  SlitGO
  UPDATEPOINTING

end
```

The last command, `UPDATEPOINTING`, makes sure that the pointing reference definition of absolute DETXY coordinates (0,0) is set to the new position after we have aligned the target with the slit. The way MODS implements UPDATEPOINTING is to make the "DETXY" coordinates explicit.

Q1537.obs:

```
#
# MODS Observing Script
#
# Created: 2011 May 29 at 06:13:32 UTC by LBTO@obs2.lbt.as.arizona.edu
# Program: mkMODSObs v0.4.5
#
# Modified to add dithering [rwp/osu 2011 May 30]
#

OBSMODE

Archive:
  PARTNER LBTSDT
  PROPID SV_OSUCIV
  PI_NAME Kochanek, Pogge, Peterson, Assef

Instrument:
  instconfig dual grating
  slitmask LS5x60x0.8

Exec:
  object Q1537-0146
```

```
  exptime 600
  nimgs 1
  go
  offsetxy 0 +10 abs
  go
  offsetxy 0 -10 abs
  go
  offsetxy 0 0 abs
  go

end
```

We are using `OFFSETXY` to make offsets in the DETXY coordinate system (which is really rotator-angle invariant PCS_XY coordinates that is aligned to the MODS long-slit "Y" axis). This is roughly along the X,Y axes of the red and blue CCD detectors, modulo small rotations of order 0.5º for each channel. Moving in +Y in DETXY coordinates moves along the slit north (when PA=0º), and ±X is perpendicular to the slit.

Note that each `OFFSETXY` command must be followed by an explicit `GO` command in order to take an exposure. This is different from LUCI scripts, where an `OFFSET` command really means "OFFSET-and-EXPOSE".

7 Script Command Summary

This section lists the MODS script commands, including arguments and brief functional descriptions. Arguments denoted by <> are required arguments, while arguments denoted by []'s are optional. Note that the <>s and []s are never typed.

7.1 Script Execution Mode Commands

One of these must appear as the first executable non-comment line of a script to establish what kind of script it is

7.1.1 **ACQMode** – Target Acquisition Mode

Usage: **ACQMode <mode>**

Where: <mode> = Imaging, Slit, or MOS

Only used in acquisition (.acq) scripts. It makes sure that the instrument is ready to point ("preset") the telescope to a science target and acquire a guide star with the AGw system. It opens the instrument dark hatch if closed, retracts the calibration tower if in the beam and turns off all of the calibration lamps.

The different modes (Imaging, Slit, and MOS) determine the behavior of the target acquisition commands (§7.6).

ACQMode scripts may only be executed using acqMODS.

7.1.2 **OBSMode** – Observing Mode

Usage: **OBSMode [restore]**

Where:

restore Moves the AGw stage back to the last position before the last CALMode

Only used in observing (.obs) scripts. It makes sure that MODS is open to the sky, the calibration system is retracted, turns off all of the calibration lamps, and if the restore argument is given it moves the AGw guide probe back to the XY position stored on the previous CALMode command.

OBSMode scripts may only be executing using execMODS.

7.1.3 **CALMode** – Calibration Mode

Usage: **CALMode**

Only used for calibration (.cal) scripts where the instrument is closed to sky and configured with the calibration tower in the beam. Used for flat field, comparison lamp, bias, and dark calibrations. It retracts the AGw stage to the home position, storing the XY probe position in the science field for future reference (see OBSMode restore), and then inserts the calibration tower into the beam so that the science cameras, if exposed, are viewing the calibration integrating sphere.

CALMode scripts may only be executed using execMODS.

7.1.4 **SETUPMODE** – Instrument Setup Procedure Mode

Usage: **SETUPMODE**

Used to denote instrument setup procedure (.pro) script used for setup, shutdown, and routine housekeeping activities that do not, generally, care about the opened/closed state of the instrument to the sky (in contrast to the other 3 script modes).

7.2 LBTO Data Archive Commands

These are commands that load information into the CCD detector control computers for storage in the image FITS headers. These header keywords are used by the LBTO Data Archive to catalog and provide search and download access to raw MODS data.

7.2.1 **PARTNER** – set the LBTO Archive Partner ID header keyword

Usage: **PARTNER <partnerID[,partner2,partner3]>**

Where: parnterID is one of the valid LBTO Partner codes, multiple partners separated by commas.

Sets the PARTNER keyword in image FITS headers used to identify the LBTO partner the data belongs to. Multiple partners may be listed together separated by commas without spaces.

Valid partnerID codes are:

Partner Institutions: AZ, INAF, LBTB, OSURC, LBTO

Cross-partner Data: CALIBRATION, COMMISSIONING, LBTSDT

A PARTNER keyword must appear in the FITS headers or you will not be able to search for and retrieve raw MODS data from the LBTO data archive.

7.2.2 **PROPID** – set the LBTO Archive Proposal ID header keyword

Usage: **PROPID <propString>**

where: propString is a string with the Proposal ID

PROPID is used to identify the data within a PARTNER as being taken as part of a partner TAC-approved LBT observing program. Each LBTO partner institution has their own convention. It is used by the LBTO data archive to further narrow down the ownership of raw data.

7.2.3 **PI_NAME** – set LBTO Archive project PI header keyword

Usage: **PI_NAME <names of the PIs>**

Gives the name(s) of the principal investigators of PROPID for PARTNER.

This FITS header keyword serves as another way to search for data in the LBTO Data Archive.

7.3 Target Preset Commands

These commands only appear in Target Acquisition (.acq) scripts as described in §3.2.

7.3.1 **OBJNAME** – set the Object Name of the science target

Usage: **OBJNAME <target name string>**

Sets the name of the science target for the observation. Appears in the acquisition image FITS headers as the OBJECT keyword. May be up to 40 characters long, but brevity is good.

May only appear in an acquisition (.acq) script.

7.3.2 **OBJCOORDS** – set the Science Target RA and Dec coordinates

Usage: **OBJCOORDS <hh:mm:ss.s ±dd:mm:ss.s>**

Where:

hh:mm:ss.ss is the Right Ascension in sexagesimal hours
±dd:mm:ss.s is the Declination in sexagesimal degrees, the sign is **required**.

All coordinates must be specified in Equinox J2000.0.

Note that the colon (:) must appear as the separator between the sexagesimal components of the coordinates – **spaces are not allowed**.

May only appear in an acquisition (.acq) script.

7.3.3 **GUINAME** – set the name of the guide star

Usage: **GUINAME <guide star ID>**

If there is no name for the guide star, GStar is an acceptable placeholder.

Stored as the GUINAME keyword in the image FITS headers, it has no other function at present except bookkeeping. Future versions of the LBTO TCS interface may do more with this.

May only appear in an acquisition (.acq) script.

7.3.4 **GUICOORDS** – set the Guide Star RA and Dec coordinates

Usage: **GUICOORDS <hh:mm:ss.s ±dd:mm:ss.s>**

Where:

hh:mm:ss.ss is the Right Ascension in sexagesimal hours
±dd:mm:ss.s is the Declination in sexagesimal degrees, the sign is **required**.

All coordinates must be specified in Equinox J2000.0.

Note that the colon (:) must appear as the separator between the sexagesimal components of the coordinates – **spaces are not allowed**.

May only appear in an acquisition (.acq) script.

7.3.5 **ROTATOR** – set the rotator position angle and mode

Usage: **ROTATOR <angle> <mode>**

Where:

angle = rotator angle in decimal degrees
mode = POSITION or PARALLACTIC

Sets the rotator position angle to use for the observation. For nearly all observing scripts with MODS the mode is POSITION, and the angle is the celestial position angle measured in decimal degrees from North thru East following standard astronomical practice. The rotator will track to maintain that celestial position during the observation.

May only appear in an acquisition (.acq) script.

7.3.6 **PRESET** – set the Telescope Preset mode

Usage: **PRESET <presetMode>**

Where: presetMode is one of

active – guide and use active optics to actively collimate the telescope
guide – guide only without using the active optics wavefront sensor
track – only track open-loop at the sidereal rate but do not guide or use active optics

Gives the mode for the telescope preset to be executed by acqMODS. Most observations will make active presets (the other modes are for engineering activities).

May only appear in an acquisition (.acq) script.

7.4 Instrument Configuration Commands

These commands are used to configure the instrument optical chain for the observation and turn on and off comparison lamps.

7.4.1 **INSTCONFIG** – Select the instrument optical configuration

Usage: **INSTCONFIG <mode> <disperser>**

Where:

mode – is the instrument mode, one of **dual**, **red**, or **blue**
disperser – is the disperser to use, one of **imaging**, **grating**, or **prism**

Configures the instrument for the observation:

Dual Mode operates the blue and red channels with the dichroic
Red Mode operates the red channel bypassing the blue (red fold flat)
Blue Mode operates the blue channel bypassing the red (no dichroic)

The choice of disperser can be one of

imaging – configure for direct imaging using the imaging flat
grating – configure for R=2000 grating spectroscopy using the gratings
prism – configure for low-res prism spectroscopy with the double-pass prisms

Both channels are configured the same (hybrid modes are impractical) in dual mode.

Instrument configuration takes up to 60 seconds maximum depending on the longest grating turret travel required, but is usually less. This includes about 20 seconds of IMCS lock-on overhead.

This command should normally only appear in observation, calibration, and procedures scripts; acquisition scripts autoconfigure.

7.4.2 **SLITMASK** – Select the slit mask used for an observation or acquisition

Usage: **SLITMASK <maskID>** or **SLITMASK < in|out>**

Where:

maskID – ID of the slit mask to deploy
in – insert the current slit mask into the science beam
out – extract the current slit mask from the science beam and stow in the cassette

Typical slit mask exchange times require between 10 and 20 seconds maximum, including insert/retract overhead.

When used in Acquisition scripts (§3.2.4), the slit mask is automatically inserted or retracted depending on the acquisition command (see §7.6).

When used in Observing or Calibration scripts, it automatically inserts the mask into the science beam after the instrument configuration step.

In Imaging Mode, the default Imaging field stop mask is used, but it should be specified explicitly. Note that the imaging field stop mask is part of the focal plane baffling system, and so using no mask in the beam for imaging will increase stray light into the instrument.

Facility long-slit and test masks have fixed names, e.g., LS5x60x0.5 (0.5-arcsec wide segmented long-slit) as described on the MODS website.

Custom multi-object masks must be named ID######, where ###### is the 6-digit unique ID code assigned by the MMS mask design program.

The explicit IN or OUT directive only applies to the currently selected maskID. If given on the command line with the maskID, the IN/OUT directive is ignored.

7.4.3 **FILTER** – Select a camera filter

Usage: **[Blue|Red] FILTER <filterID>**

Where:

Blue – select the Blue Camera filter
Red – select the Red Camera filter
filterID – name of the filter to select in the blue/red camera

Selects the filter to be used for an observing script (.obs, .cal, or .pro). Not used in acquisition scripts (use `AcqFilter` instead).

Filters are selected by name, see the MODS website for the current filter complement. Note that there are a mix of imaging (square) and spectral (rectangular) filters available. The `INSTCONFIG` command will select the default filter for spectral modes.

Note that because the filter thickness is part of the optical prescription, use of no filter is bad – the cameras will be out of focus. For an unfiltered image, use the Clear filter in each camera.

7.4.4 **LAMP** – Select and Control Calibration Lamps

Usage: **LAMP <lamp1 [lamp2 lamp3 …] > ON** or **LAMP [lamp1 lamp2] OFF**

Where:

lamp1 – first lamp to select
lamp2 – second lamp, …
ON – turn the selected lamp(s) ON
OFF – turn the selected lamp(s) OFF – if no lamps selected, TURN ALL OFF

Turns calibration lamps on or off. The current selection of lamps is listed on the MODS webpage.

Lamps may be turned on one or many at a time.

The LAMP OFF command turns off all of the calibration lamps.

A special variable-intensity flat-field lamp allows for an additional selection of the lamp brightness:

LAMP VFLAT level ON

will turn on the variable-intensity lamp. The level is an integer between 1 and 11.

LAMP must only be used in calibration (.cal) scripts with the instrument in CALMode.

7.5 Exposure Control Commands

These commands setup and take exposures on the blue and red channels.

7.5.1 **OBJECT** – Setup to take an object exposure

Usage: **OBJECT <target name>**

Sets the OBJECT header keyword to “target name” and IMAGETYP to OBJECT. This is an open-shutter science observation of the given exposure time.

If used without the Blue or Red qualifier, it sets the object name and image type the same on both channels, otherwise use “blue object” and “red object” to set each channel individually.

7.5.2 **BIAS** – Setup to take an bias (zero) frame

Usage: **BIAS <bias name>**

Sets the OBJECT header keyword to “bias name” and IMAGETYP to BIAS. This is a zero-duration closed-shutter bias frame: the CCD is erased then immediately readout. Sets EXPTIME to 0.

If used without the Blue or Red qualifier, it sets the name and image type the same on both channels, otherwise use “blue bias” and “red bias” to set each channel individually.

Ideally the instrument is closed to the sky in CALMode during bias frames.

Note that the detailed 2D structure in a BIAS image depends on the CCD ROI mode (see §7.5.11) so you need separate biases for each ROI you use for science data.

7.5.3 **FLAT** – Setup to take an flat-field exposure

Usage: **FLAT <flat name>**

Sets the OBJECT header keyword to “flat name” and IMAGETYP to FLAT. This is an open-shutter calibration observation of the internal flat-field lamps of the given exposure time.

If used without the Blue or Red qualifier, it sets the object name and image type the same on both channels, otherwise use “blue flat” and “red flat” to set each channel individually.

Because it implies it is looking at the internal calibration sources, it should only be used in CALMode.

See LAMP for how to turn on the flat field lamps.

7.5.4 **SKY** – Setup to take a twilight sky flat-field exposure

Usage: **SKY <sky flat name>**

Sets the OBJECT header keyword to “sky flat name” and IMAGETYP to SKY. This is an open-shutter twilight sky flat observation of the given exposure time.

If used without the Blue or Red qualifier, it sets the object name and image type the same on both channels, otherwise use “blue sky” and “red sky” to set each channel individually.

Note that because it is observing the twilight sky, this can only be used in OBSMode.

7.5.5 **COMP** – Setup to take a wavelength comparison lamp exposure

Usage: **COMP <comp name>**

Sets the OBJECT header keyword to “comp name” and IMAGETYP to COMP. This is an open-shutter exposure of comparison lamps in the internal calibration system.

If used without the Blue or Red qualifier, it sets the object name and image type the same on both channels, otherwise use “blue comp” and “red comp” to set each channel individually.

Note that because it is observing the calibration system, it should only be used in CALMode.

See LAMP for how to turn on the comparison lamps.

7.5.6 **STD** – Setup to take a flux calibration object exposure

Usage: **STD <flux standard name>**

Sets the OBJECT header keyword to “flux standard name” and IMAGETYP to STD. This is an open-shutter exposure of on a flux calibration target (either a photometric calibration field image or a spectrophotometric flux standard star spectrum).

If used without the Blue or Red qualifier, it sets the object name and image type the same on both channels, otherwise use “blue std” and “red std” to set each channel individually.

Note that because it is observing an astronomical target, it must only be used in OBSMode.

7.5.7 **DARK** – Setup to take a dark exposure

Usage: **DARK <dark name>**

Sets the OBJECT header keyword to "dark name" and IMAGETYP to DARK. This is an closed-shutter integration of the given exposure time.

If used without the Blue or Red qualifier, it sets the object name and image type the same on both channels, otherwise use "blue dark" and "red dark" to set each channel individually.

Note that because it is ostensibly measuring dark signal accumulation on the unexposed CCD detectors, is should only be used in CALMode.

7.5.8 **EXPTIME** – Set the single-image exposure (integration) time

Usage: **EXPTIME <seconds>**

Sets the single-image exposure time in seconds. If used without the RED or BLUE qualifiers, it sets the exposure time the same on both cameras, otherwise use "red exptime" and "blue exptime" to set the exposure times separately on the red and blue cameras.

The minimum exposure time is 1 second. The longest practical single-image exposure time is ~1800 seconds (limited by cosmic ray accumulation, especially on the thick red CCDs).

IMPORTANT: EXPTIME must **FOLLOW** OBJECT, FLAT, COMP, etc. in a script, and **CANNOT** be used with BIAS.

7.5.9 **NIMGS** – Set the number of exposures to acquire

Usage: **NIMGS <num>**

Where: num is the number of images to take (1 to 99).

Sets the number of exposures to be taken with the next GO command. If given without the RED or BLUE qualifier, it sets the same number of exposures to take on both cameras, , otherwise use "red nimgs" and "blue nimgs" to set the number of exposures separately on the red and blue cameras.

The total integration time will be (nimgs)×(exptime) seconds.

7.5.10 **CCDBIN** – set the CCD on-chip binning factor

Usage: **CCDBIN <n>** or **CCDBIN <nX nY>**

Sets the CCD on-chip binning factor.

If only 1 number is given (e.g., ccdbin 2), both axes are binned the same (n×n), otherwise if two numbers are given, the CCD is binned nX×nY. In any case, the binning factor must be 1 or a power of 2 up to 8, thus only n=1,2,4,8 are allowed.

If no RED or BLUE qualifier is used, the binning factor is set the same on both channels, otherwise use "red ccdbin" or "blue ccdbin" to set the binning separately on the two channels.

Unbinned operation is set by default by INSTCONFIG, so any setting of CCDBIN must come after INSTCONFIG in a script.

NOTES:

Binning is not "sticky". If you set binning to 2×2 pixels in one script, it will get reset to the default 1×1 by the next script.

In general, the readout noise of $2.5e^-$ is so low that there is little or no noise penalty for binning in software after the fact. The only gain is in readout time, but because of irreducible setup and readout overhead, the readout time does not scale inversely with the product of the binning factors. See the MODS website for the current readout times in different binning modes.

7.5.11 **ROI** – Set the CCD subframe readout (Region-Of-Interest) mode

Usage: **<RED|BLUE> ROI <roiMode>**

Sets the CCD subframe readout (region-of-interest) mode. Only preset ROIs are allowed, and these are all centered on the CCD (the intersection of the 4 readout amplifier quadrants). The preset ROI modes available are:

1Kx1K – Central 1024×1024 pixels
2Kx2K – Central 2048×2048 pixels
3Kx3K – Central 3072×3072 pixels (full Imaging Field)
4Kx3K – Central 4096×3072 pixels (Prism-mode full MOS footprint)
8Kx3K – Full frame readout (Grating-mode full MODS footprint)

Note that ROI must include the RED or BLUE qualifier.

ROI is not "sticky" – INSTCONFIG will automatically set the unbinned default ROI mode for the three major instrument modes:

Imaging – 3Kx3K
Grating – 8Kx3K
Prism – 4Kx3K

ROI should only be used in OBSMode or CALMode, as the ROI mode for Acquisition Scripts is set by the AcqROI command (§7.6.4).

7.5.12 **GO** – Start an integration sequence

Usage: **GO**

Starts an integration sequence. On each active channel, GO takes NIMGS IMAGETYP integrations of EXPTIME each.

The behavior of GO without a RED or BLUE qualifier depends on the instrument mode set by INSTCONFIG:

Dual Mode – start integration sequences simultaneously on the blue and red channels
Blue Mode – start the integration sequence on the blue channel, but leave red idle
Red Mode – start the integration sequence on the red channel, but leave blue idle

This can be overridden by adding the RED or BLUE qualifier. For example, in Dual Mode, "BLUE GO" will start an integration sequence on the blue channel but leave red idle, and vis-versa for "RED GO".

NOTE: GO should not be used in ACQMode, as the special AcqGO and SlitGO commands are provided to make sure of the slit position in the science field (see §7.6.5 and 7.6.6).

7.5.13 **DGO** – Start an integration sequence on both channels

Usage: **DGO**

Starts integration sequences on both blue and red channels, overriding the instrument mode (dual, blue, or red) set by INSTCONFIG. It is most often used in "dark" calibration acquisitions like bias and dark frames, since those image types are independent of the optical configuration of the channels.

DGO should not be used in ACQMode scripts.

7.6 Target Acquisition Commands

These are special commands used in Target Acquisition (.acq) scripts to select the configuration for the target acquisition images and to start thru-slit and field images.

7.6.1 **ACQCamera** – Select the camera to use for acquisition images

Usage: **ACQCamera <red|blue>**

Selects the Red or Blue camera for acquiring targets. Only one channel may be used for target acquisition.

IMPORTANT: ACQCamera must come **BEFORE** all other ACQxyz command.

7.6.2 **ACQFilter** – Select the camera filter for acquisition images

Usage: **ACQFilter <filterID>**

Select filterID as the camera filter for acquisition images. The filterID must be a valid filter loaded in ACQCamera.

This command must come **after** ACQCamera in acquisition scripts.

7.6.3 **ACQExpTime** – Set the acquisition image exposure time

Usage: **ACQExpTime <seconds>**

Sets the acquisition image integration time in seconds. Only a single acquisition image is taken at a time with AcqGO or SlitGO.

This command must come **after** ACQCamera in acquisition scripts.

7.6.4 **ACQROI** – Set the CCD readout ROI for acquisition images

Usage: **ACQROI <roiMode>**

Where: roiMODE is one of the valid CCD ROI modes defined in §7.5.11 above. The default is 3Kx3K (full imaging field).

For ACQMode MOS target acquisition, the default 3Kx3K ROI should be used.

For ACQMode Slit target acquisition, either 3Kx3K or 1Kx1K are often used.

Also, as with ROI, this setting is not "sticky", it will be reset by the next .obs script.

7.6.5 **AcqGO** – Take a field (no slit) acquisition image

Usage: **AcqGO**

Retract the slitmask and take a single image of ACQExpTime seconds of the target field.

7.6.6 **SlitGO** – Take a thru-slit acquisition image

Usage: **SlitGO**

Insert the slitmask and take a single image of ACQExpTime seconds through the slit.

Note that you can change the exposure time by setting ACQExpTime immediately before SlitGO, though this only only usually done for very bright targets that would saturate the field images and where a longer thru-slit image is needed to see the slit against the sky.

7.7 Script Flow Control and Printing Commands

The commands are internal to `acqMODS` and `execMODS`, and are used to control the flow of execution (pause and sleep) or print messages (print) on the screen for the observers.

7.7.1 PAUSE – Pause execution of a script

Usage: **PAUSE**

Pauses execution of the script, printing a prompt asking the user to hit the Enter key to resume execution or X to abort the script.

It is good script form to precede a PAUSE command with a PRINT statement telling the observer why the script is pausing.

7.7.2 **PRINT** – Print a message to the terminal screen

Usage: **PRINT <message string to print>**

Prints a message string on the terminal where the script is running.

Print statements are used to give a progress report in the case of long scripts (e.g., long calibration sequences), or to print messages before PAUSE or SLEEP statements to let the observers know why the scripted is PAUSEd or SLEEPing.

7.7.3 **SLEEP** – Stop script execution and sleep for a time interval

Usage: **SLEEP <seconds>**

Stop script execution and sleep for a number of seconds. A countdown is printed on the screen, and any broadcast messages from the MODS data-taking system are printed on the terminal as usual while a sleep is in progress (“sleep with one eye open”).

Common uses of sleep are to pause for a fixed time to allow certain operations to complete, like waiting for the Hg(Ar) comparison lamp to warm up for 30 seconds. It is most often used, however, in engineering instrument procedure (.pro) scripts for pauses in testing scripts.

7.8 Telescope Commands

These commands are used to offset the telescope and pause/resume guiding.

7.8.1 **OFFSET** – Offset the telescope in celestial (RA,Dec) coordinates

Usage: **OFFSET** <**dRA dDec> <ABS|REL>**

Where:

dRA – RA offset in arcseconds, + = E, – = W (includes the cosδ correction)
dDec – Dec offset in arcseconds, +=N, – = S
ABS – make an absolute offset relative to the pointing reference
REL – make an offset relative to the current position

OFFSET is used when moving the telescope in celestial coordinates ("RADEC" coordinates in LBT TCS terminology). Absolute offsets are made relative to where the telescope landed and locked on the guide star after the preset (the "pointing reference"), whereas relative offsets are made from the current location.

The guide probe follows the offset (unless you drive the probe outside limits, which will cause a fault and guiding/active optics will fail and the telescope will track open loop).

OFFSET is normally only used for executing blind offsets or some imaging dither patterns. For doing offsets along the slit or to place targets on the slit, use OFFSETXY.

7.8.2 **OFFSETXY** – Offset the telescope in the Slit XY coordinates

Usage: **OFFSETXY <dX dY> <ABS|REL>**

Where:

dX – X-axis offset in arcseconds, +/– X_{CCD} motion (perpendicular to the slit)
dY – Y-axis offset in arcseconds, +/– Y_{CCD} motion (parallel to the slit)
ABS – make an absolute offset relative to the pointing reference
REL – make an offset relative to the current position

OFFSETXY moves the telescope in slit plane XY coordinates ("DETXY" coordinates in LBT TCS terminology, but the coordinate system is really rotator-angle invariant PCS_XY coordinates that are roughly aligned to within 0.5 degrees of the MODS CCD detectors).

Absolute offsets are made relative to where the telescope landed and locked on the guide star after the preset (the "pointing reference"). This can be reset using the UPDATEPOINTING command.

Relative offsets are made from the current location:

A +X offset moves a star in the +X direction on the CCD, perpendicular to the slit.

A +Y offset moves a star in the +Y direction on the CCD, parallel to the slit.

OFFSETXY is what you use to move an object onto the slit (it or a variant are what is sent by modsAlign), and OFFSETXY in the +/–Y direction will dither an object along one of the segmented long-slit masks.

7.8.3 **UPDATEPOINTING** – Set ("Zero") the Telescope Pointing Reference

Usage: **UPDATEPOINTING**

Instructs the telescope control system to make the current position where the guider is locked-on and guiding the (0,0) position for absolute offsets – the so-called "pointing reference".

This command should follow any OFFSET or OFFSETXY command that centers a target in the slit (either a blind offset or an alignment offset). Note that the `modsAlign` program sends this command after executing the offset of the targets onto the slits, so it is not a routine command for most acquisition scripts.

7.8.4 **GPAUSE** – Pause guiding and active optics but keep tracking open loop

Usage: **GPAUSE**

Instructs the GCS to stop closed-loop guiding and (if an active preset) suspend active optics corrections. The telescope and rotator will continue to track open loop.

This would be used to temporarily pause guiding for some instrument operation that would interfere with normal guiding and active optics (e.g., closing the hatch to take a quick dark).

Guiding can be resumed with GRESUME.

Note that this mode has been tested, but not extensively by MODS during commissioning. Use at your own risk (note: calibration-in-place is rarely indicated for MODS, as described in the *MODS Calibration Procedures* document and is risky).

7.8.5 **GRESUME** – Resume guiding and active optics after a GPAUSE

Usage: **GRESUME**

Attempts to resume closed-loop guiding and (if an active preset) active optics corrections following a GPAUSE.

Has not been extensively tested with MODS at this writing.