AstroPhotography Automation System (APAS)

User's Manual

v21

July 16, 2026

Web Publishing and Editorial Revision

Contents

Version History

This section records all changes to the manual. The most recent version is listed first. Major revisions (new sections, architectural changes) increment the first digit; minor revisions (additions, corrections, clarifications) increment by 0.1.

Version	Date	Author	Summary of Changes
3.1	2026-07-16	M. Milligan	Web publishing and editorial revision (v21): documented web publishing of finished images — the online image gallery (build_gallery.py / deploy_gallery.sh, sips-resized 2048px display + 600px thumbnails, gallery.json metadata), pipeline publishing via the dashboard board (W) Publish to Web and Finishing (Pw) Publish to Web Page with automatic final-image detection (Album→Gigapixel→Cosmos→Siril), Finishing (Ma) Move to Album, and dashboard (V) View in Preview; the phone imaging-calendar web app (build_calendar_app.py; astromenu (Vp) Publish App) with live up-now altitude, DSO search, and adjacent-month carryovers; and cPanel UAPI deployment with API-token rotation (Section 4.3). Documented the Viewing Calendar's plus-or-minus one-month carryovers, declination-based Northern/Southern bucketing in the Word export, and sensor-aware Vespera Pro / Vespera 2 routing (emission nebulae to the wide Vespera 2 at 18 arcmin and up, galaxies and clusters to the Vespera Pro until 45 arcmin). Applied a light professional-voice pass across all chapters, added a table of contents, and corrected the cover and running header to v21. Rebuilt Chapter 15 (How-To Quick Reference) into workflow-phase recipes that pair the astromenu path with command-line steps — covering acquisition through publishing, the interactive Cosmos and Gigapixel hand-offs (including resuming them from a

Version	Date	Author	Summary of Changes
			fresh terminal after an interrupted pipeline), album refresh, remote/parallel processing, and troubleshooting.
3.0	2026-07-03	M. Milligan	Parallel per-machine pipelines (v20): stack_sync replaces the single-job cap with per-machine worker slots — the Mini, Pro (C_MAIN), and M5 Air (C_NODE1) each run a full pipeline in parallel, one per machine. Remote workers are driven by astro_autoprocess --remote HOST, which delegates the whole chain to astro_remote_autoprocess. LAN-only worker selection (SSH, non-Tailscale); remote failures re-queue on the Mini (MAX_PIPELINE_ATTEMPTS=3). Supersedes the same-day round-robin stack-only draft. Section 8.4 rewritten.
2.8	2026-07-02	M. Milligan	Auto-pipeline documentation pass (v18): stack_sync auto-batch daemon; M4 Mini (C_STACK) primary stacking node + AstroRAID; RC Astro tools (BX/NX/SX); giga_finish one-command finishing; astro_log_snapshot; BatchResultsPDF; machine/drive table updates (Camel X10).
2.7	2026-06-24	M. Milligan	Editorial revision (v17): corrected Gigapixel description and removed unsupported training claims; clarified that the Mac mini provides optional archive/catalog services rather than the core APAS processing workflow; integrated Starless Processing (StarNet2) into workflow summaries, stage-code tables, and processing paths; standardized terminology for Gigapixel, VesperaPro, Vespera 2, GraXpert, StarNet2, and auto-process; clarified optional versus core stages and softened subjective processing guidance.
2.6	2026-06-	M. Milligan	Minor revision (v16): added AstroDen

Version	Date	Author	Summary of Changes
	23		(C_DEN) node and ASTROHUB stacking engine (Sections 5.1, 7.4); new astro_next_step post-collect dispatcher, astro_hub_push, and astro_hub_combine scripts; archive_stacked for stacked FITS backup (Section 7.6); SwiftBar menu-bar node indicator (Section 7.8); astro_netwatch extended to mirror StackingOutput (Section 8.1); astro_catalog SMB auto-mount fix (Section 8.2); remote daemon access from all nodes (Section 8.3); Recent Projects screen and Pipeline shortcut (PI) in astromenu (Section 7.1); auto-collect after download; GraXpert upgraded to 3.1 with CLI denoise strength and AI deconvolution (Section 7.5). Formality pass on Sections 1, 2, and 3 (third-person perspective; informal headings replaced).
2.5	2026-06-10	M. Milligan	Data-safety description corrected throughout: netwatch mirrors the raw FITS/TIFF frames to AstroArchive/RawTelescope/ (the backup), while astro_catalog maintains the JPG browse catalog (Sections 1, 8, 9). Documented the automatic combine workflow — the post-ingest combine offer and the Combine (Cb) submenu with pending opportunities and batch 'all' mode (Section 14.9). Documented netwatch's registry-pinned host mounting. Replaced stale ASTRO_ACTIVE_T9 references with D_MAIN / AstroMaster. Professional-tone pass on Section 1; worked-example version names aligned to the DSO-### convention.
2.4	2026-06-09	M. Milligan	Consistency audit: legacy T9 naming purged (vars renamed ASTRO_DATA_*, compat shims remain); astroenv.zsh promoted to full

Version	Date	Author	Summary of Changes
			path provider (ASTRO_WF_* stage vars, ASTRO_SHARED Dropbox tier root); retired astroprocess, astroallpull, provenance_report, helpastro; new astro_node_check health tool; CONVENTIONS.md; bin hygiene sweep. Shared-data design adopted: StackingOutput (handoff) and AstroAlbum (final) to live on Dropbox SharedData.
2.3	2026-06-04	M. Milligan	Drive upgrade: T9 retired, data on Nextorage AstroMaster, per-node Siril scratch (AstroScratch). New batch tools: stage_backlog, stack_backlog, newdrive. FITS-first / TIFF-fallback stacking; registry lookups read the canonical Dropbox registry.
2.2	2026-06-04	M. Milligan	Minor revision: new Starless (StarNet2) workflow — Tool Steps (Sl), siril_starless reference (Section 7.5), Section 3.5, and SL/SK/SF stage codes; GraXpert --denoise / --denoise-strength pass; sirilprocess now auto-skips subsky for _GX inputs (golden rule now automatic) and gains --no-denoise / --no-sharpen; Process (Pr) defer-sharpen prompt; CollectFits now runs regen_sources_json automatically; corrected the astromenu Starless StarNet2 hint. v10 adds built-in deconvolution to sirilprocess (--deconvolve: Siril makepsf + Richardson-Lucy on the linear image, with --deconv-iters / --deconv-psf and a Process (Pr) prompt), the setup_starnet replication helper, richer dso_checkout --list columns (TIFF count, Subs, Sess, Size, Obs dates), and CollectFits hardlinking of duplicate frames into new versions.
2.1	2026-05-31	M. Milligan	Minor revision: scope keys simplified (v2a/v2b→v2, vp1/vp2→vp for stacking); MoveToAlbum Apple Photos auto-import; dso_checkout/dso_checkin

Version	Date	Author	Summary of Changes
			remote-work scripts; archive_processed script; remote SSH shortcut; Cs/Gp/Co/Ci menu entries added; Utilities & File Management reorganisation; Tool Steps Gx/Si order corrected; astro_workflow_backup connection-drop fixes.
2.0	2026-05-27	M. Milligan	Major revision: formal APAS naming adopted throughout; new Chapter 2 (Getting Started / installation guide for new users); new Chapter 3 (Using astromenu — beginner workflow, GraXpert, Cosmos, and Gigapixel usage); new Chapter 4 (Complete Workflow — astromenu and CLI paths with examples). Existing chapters renumbered 5–15.
1.0	2026-05-26	M. Milligan	Initial release of the formal user manual. Covers complete pipeline from capture through archive, all scripts with CLI reference, daemon documentation, auto-process paths, tips and worked examples.

1 System Overview

The AstroPhotography Automation System (APAS) manages the full lifecycle of image data from Vaonis telescopes: from scope WiFi capture through network transfer, ingestion into the working pipeline, stacking, post-processing, and final album output. Once configured, the system is nearly fully automatic — no manual file-moving is required after a shooting session.

APAS separates the active processing workflow from optional archive and catalog services. Core collection, ingestion, stacking, and processing run from the active data drive and can be performed without the Mac mini. When available, the Mac mini provides background archive and catalog services: it discovers registered storage drives on the local network, mirrors new sessions into AstroArchive, and maintains a browsable session catalog. Interactive processing work (stacking, color calibration, optional Starless Processing, tone mapping, and upscaling) runs on the workstation that hosts the active D_MAIN SSD.

A single imaging session illustrates how the pipeline operates end to end. Consider a night spent imaging M42, the Orion Nebula, with the VP-1 VesperaPro. The scope stacks frames throughout the session, producing hundreds of calibrated FITS frames alongside a continuously updated preview image. The following morning, astromenu is launched and a scope pull is initiated. The system connects to the VP-1's WiFi access point, retrieves every FITS frame, TIFF file, and the most-recent JPG, and deposits them on the D_MAIN SSD in a timestamped observation folder under RawTelescope/. The JPG is the scope's on-board rendering of the image — useful for a quick visual check, but substantially less capable than the result obtained by stacking the raw frames through the full pipeline.

If the archive mini is running, it monitors the network in parallel with the active workflow. Within about a minute of D_MAIN appearing, the netwatch daemon detects the drive, mounts it, and begins mirroring the new observation folders — every raw FITS and TIFF frame, with the on-disk folder structure preserved — into AstroArchive/RawTelescope/ on the mini's storage drive. This raw mirror is an optional but important data safeguard: an additional copy of the frames that remains intact even if D_MAIN is later reformatted. A companion daemon, astro_catalog, copies the newest processed JPG from each session into a browsable Sessions/ and DSO/ catalog and updates the session index, providing a quick visual reference organized by date and target. In short, the core APAS workflow can run without the mini; the mini adds automated archive and catalog services.

For the best results, the raw FITS frames can be processed through the APAS pipeline. From astromenu, select Collect FITS/TIFFs (CI), select the M42 observation, and the system copies those FITS frames into a pipeline version folder — M42-001 — in a standardized layout that Siril can work with. This is the ingestion step, and its primary purpose is creating a clean, deduplicated, organized input for the stacker.

Next is the stacking process with Siril. Siril aligns every frame to a common reference (registration), identifies and rejects outlier pixels from cosmic rays, satellites, and other transients (rejection), and then averages the accepted frames together (integration). The rationale for stacking hundreds or thousands of frames is to maximize the signal-to-

noise ratio. Random noise averages toward zero as more frames are integrated, while the real signal from the nebula accumulates. M42 might produce 400–500 frames from a single night with the VP-1; stacking those frames produces a master image with far more recoverable detail in the faint outer regions than any individual frame could show.

The next step is to remove extraneous background light pollution and gradients. Even from a dark site, the sky is not perfectly uniform; unwanted light from the atmosphere and internal reflections in the optical train create a smooth, large-scale brightness variation across the image. GraXpert analyzes the stack, builds an AI model of those background variations, and subtracts them, yielding a stack with a flat sky background.

Next, sirilprocess performs two important operations in sequence. First, it runs Siril's SPCC — spectrophotometric color calibration — which plate-solves the image, then uses the known spectral types of stars in the field to correct the color balance. Second, it applies an auto-stretch, pulling the linear data into a displayable range. The output is a TIFF — the PO file — which is often sufficient as a review or presentation image and also serves as the input for optional Starless Processing, Cosmos tone mapping, or Gigapixel upscaling.

For targets with strong nebulosity or fine structure, Starless Processing can be run after sirilprocess. This optional stage uses StarNet2 to separate stars from the nebula or galaxy structure, allowing the nonstellar detail to be sharpened without enlarging star cores. For targets with extreme dynamic range such as M42, Cosmos Darkroom may then be used for tone mapping. Cosmos provides independent control over the bright core and faint outer structure — something a global stretch cannot achieve. For less extreme targets such as many clusters, galaxies, and diffuse nebulae, Cosmos is typically omitted.

Gigapixel is an optional output-preparation step for large-format display or printing. VesperaPro frames are roughly 9 megapixels; a 2× upscale produces approximately 60 megapixels, sufficient for a 20″×24″ print at full resolution. A final MoveToAlbum command copies the selected output into AstroAlbum/M42/ with a standardized filename, completing the session pipeline.

The end-to-end pipeline from scope pull to selected album output varies significantly, depending on the number of FITS files and network connection; there can be significant elapsed time waiting for scope downloads, preparing files for stacking, and for Siril or Gigapixel to complete their respective tasks.

1.1 Design Principles

- Zero-maintenance drive discovery via mDNS/Bonjour — adding a new drive requires only a registry entry, no code changes.
- Core processing can run on any configured Apple Silicon Mac; optional background archive and catalog services run as daemons on the Mac mini when that subsystem is enabled.
- Scripts and configuration live in Dropbox (synced to all machines). Image data never touches Dropbox.
- Every operation is idempotent — re-running any script is safe.

- Version identifiers (DSO-001, DSO-C01) track processing lineage without polluting the filesystem with redundant copies.

1.2 Two-Root Architecture

The system splits cleanly into two trees: ASTRO_CODE (scripts and config, synced via Dropbox to every Mac) and ASTRO_BASE (image data, lives on whichever data drive is currently active). Scripts are always up to date on every machine; data travels with the drive.

Variable	Points To / What Lives There
ASTRO_CODE	Dropbox/Astronomy — scripts, config, drive registry, ingest index
ASTRO_BASE	Active data drive (D_MAIN, AstroWork, etc.) — images, PhotosData, RawTelescope, Workflow

1.3 Pipeline at a Glance

Automatic path (M4 Mini): as of the C_STACK deployment, raw frames landing on AstroMaster are mirrored to AstroRAID on a dedicated M4 Mac mini, where the stack_sync daemon auto-ingests them, stacks and processes them (optionally with the RC Astro tools and GraXpert), and generates a batch review PDF -- with no interactive steps. See section 8.4.
A typical session moves through these stages automatically or with single-command prompts:

- Transfer data from scope: Vaonis scope captures frames over WiFi and stores them on-board.
- astro_allpull (or vespera_pull) transfers raw FITS, TIFFs, and processed JPGs to the D_MAIN SSD.
- Automatic archiving: astro_netwatch (mini daemon) detects D_MAIN on the network via SMB, registers new sessions, and mirrors the raw FITS/TIFF frames into AstroArchive/RawTelescope/.
- astro_catalog (mini daemon) copies each session's newest processed JPG into the AstroArchive Sessions/ and DSO/ browse catalog.
- CollectFits / CollectTiffs ingests raw frames into StackingInput/ for a pipeline version (e.g. M42-001).
- stackdso_fits_v1 / stackdso runs Siril to produce a master stack (SO##).
- graxpert_clean removes the sky background gradient (GX##).
- sirilprocess applies SPCC color calibration and an auto-stretch (PO##).
- PrepareCosmos / RunCosmos / cosmos_export provide optional HDR tone-mapping (CI##/CO##).
- PrepareGiga / RunGiga / MoveToAlbum upscale with Gigapixel and deposit the selected output in AstroAlbum.

2 Getting Started

This chapter covers installation of APAS and verification that the environment is ready for use. Background sections on the terminal and shell environment are included for those new to the Mac command line or to astrophotography image processing. Readers already comfortable with zsh and familiar with earlier APAS versions can proceed directly to Section 2.3.

2.1 Prerequisites

APAS runs on Apple Silicon Macs and is tested on macOS Ventura and later. The following are required before installation.

Hardware

- A Mac with Apple Silicon (MacBook Pro, MacBook Air, or Mac mini). APAS is not supported on Intel Macs.
- A Vaonis telescope — VesperaPro (VP-1 or VP-2) or Vespera 2. APAS is built around the Vaonis data format.
- A registered data drive — the portable SSD that holds the working image library (D_MAIN, currently the Nextorage AstroMaster). It connects to the workstation and, when the archive subsystem is enabled, can also be discovered and archived by the Mac mini over the network.
- A Mac mini configured as the optional archive and catalog hub. This is not required for core APAS processing, but it enables background archiving and browse-catalog maintenance. See Section 2.5 for mini setup.

Software — Required

- Dropbox, installed and signed in. APAS scripts live in the Dropbox Astronomy folder and sync automatically to every Mac.
- Siril (free). Download from siril.org. Siril performs the stacking step — aligning and averaging the raw frames into a master image.
- GraXpert (free). Download from graxpert.de. GraXpert removes sky background gradients using AI.

Software — Optional

- Cosmos Darkroom (paid). Tone-mapping application for high dynamic range targets. Useful for M42, M31, and similar bright nebulae. See Section 3.3.
- Topaz Gigapixel (paid). Neural-network-based upscaling for large-format display, detailed crops, and print preparation. See Section 3.4.

2.2 A Quick Introduction to the Terminal

The terminal (also called the command line or shell) is a text-based way to control the Mac. It can look intimidating, but day-to-day APAS work requires only three actions:

opening it, typing a command, and pressing Return. Most interaction with APAS happens through astromenu, a guided menu that handles the underlying details.

Opening Terminal

1. Press Command + Space to open Spotlight Search.
2. Type Terminal and press Return.
3. A window opens with a prompt that looks something like this:

```
michaels-mbp:~ michael$
```

The text before the $ is the computer name and username. Typed input appears after the $. Press Return to run a command.

Basic Navigation (Rarely Required)

APAS handles navigation automatically, but these three commands are worth knowing:

```
ls          # list files in the current folder
cd folder   # go into a folder (replace 'folder' with the name)
pwd         # print the path of the current folder
```

What is zsh?

zsh (Z shell) is the program running inside the terminal. It is the default shell on all modern Macs. All APAS scripts are written for zsh. No configuration is required — it is already there.

What is a PATH?

When a command such as astromenu is entered, the shell searches a list of folders called the PATH to find the program. APAS automatically adds its scripts to the PATH when astrorc is sourced (see Section 2.3). After that, any APAS command runs from any folder.

2.3 Installing APAS

Installation has three steps: verify Dropbox is in place, add one line to the shell startup file, and open a new terminal. That is all.

Step 1 — Verify the Dropbox Astronomy Folder

APAS scripts live at:

```
~/Library/CloudStorage/Dropbox/Astronomy/Tools/bin/
```

Open Finder, navigate to the Dropbox folder, and confirm that Astronomy/Tools/bin/ exists and contains files such as astromenu, astrorc, and astro_allpull. If the folder does not exist, Dropbox may still be syncing — wait for it to finish before continuing.

Step 2 — Add APAS to the Shell Startup File

Open Terminal and run the following command to open the shell configuration file in the nano text editor:

```
nano ~/.zshrc
```

Using the arrow keys, scroll to the very bottom of the file. Add this line:

```
source
~/Library/CloudStorage/Dropbox/Astronomy/Tools/bin/astrorc
```

Save and exit nano:

4. Press Control + O (the letter O, not zero). Nano prompts for confirmation of the filename — press Return.
5. Press Control + X to exit nano.

Back at the terminal prompt, run this to apply the change immediately:

```
source ~/.zshrc
```

Step 3 — Verify the Installation

Type:

```
astromenu
```

If the astromenu screen appears, APAS is installed. The header shows which data drive is active (or a message that no drive is mounted if D_MAIN is not connected), followed by the menu items. Press Q to quit.

If the shell reports command not found, the most likely cause is an incorrect path in Step 2. Verify the Dropbox path by running:

```
ls ~/Library/CloudStorage/Dropbox/Astronomy/Tools/bin/astrorc
```

If that file is not found, Dropbox may be installed in a different location on the machine. Contact the system administrator.

Step 4 — Connect the Data Drive

Plug in the D_MAIN SSD. Open a new terminal window (or run source ~/.zshrc again). The astromenu header now shows [AstroMaster], confirming that APAS has found the data drive. When astromenu is launched, all items that previously showed a warning are now available.

Note: *The first time a drive is connected after a fresh installation, APAS may take a few seconds to set the environment variables. If the drive is not recognized, try opening a brand-new terminal window.*

2.4 Per-Machine Configuration

Each Mac that runs APAS needs a small machine-specific configuration file that tells the system which scope WiFi networks are available and what this machine's role is. The file lives at:

```
~/.config/astro/config.zsh
```

This file is not synced via Dropbox — it is intentionally per-machine. A typical config looks like:

```
# Scope WiFi configuration
ASTRO_SCOPE_HOST="10.0.0.1"
ASTRO_SCOPE_SSID_VP1="VesperaPro-XXXXXX"
```

```
ASTRO_SCOPE_SSID_VP2="VesperaPro-YYYYYY"
ASTRO_SCOPE_SSID_V2A="Vespera2-XXXXXX"

# This machine's role
ASTRO_THIS_HOST="ASTRO_PRO_HOST"
```

Ask the system administrator for the SSID values for the specific scopes. They are printed on a label on each scope or are visible in the WiFi network list when the scope is powered on.

2.5 Setting Up the Mac mini (Optional Archive Hub)

The Mac mini runs two optional background daemons (astro_netwatch and astro_catalog) that archive and catalog data whenever a registered drive appears on the network. Core APAS collection, stacking, and processing do not require the mini. The archive hub only needs to be set up once.

After completing Steps 1–3 above on the mini, run:

```
astro_setup_mini
```

This single command creates the AstroWork and AstroArchive folder layouts, registers the mini's drives in the drive registry, initializes the ingest index, and installs both daemons as launchd LaunchAgents that start automatically at login. It is safe to run again if anything needs to be repaired.

Verify the daemons are running:

```
astro_netwatch --status
astro_catalog --status
```

Both should report running and show a recent scan timestamp. From this point on, the mini works without any user intervention — the daemons require attention only during troubleshooting (see Chapter 11).

2.6 Key Concepts for New Users

A few concepts provide useful orientation before the workflow.

FITS Frames vs. the Preview Image

The Vaonis scope produces two types of output during an imaging session. First, it continuously stacks frames internally and displays a preview — a processed JPEG that resembles a finished image. This is useful for a quick visual check, but it is not what APAS uses for final processing. Second, the scope saves individual raw FITS frames — the unprocessed sensor data for each exposure. APAS downloads and stacks these raw FITS frames using Siril, extracting far more detail than the scope's on-board processing can.

Stacking: Why More Frames = Better Images

A single 10-second exposure from the scope contains a mixture of real signal (photons from the nebula) and random noise (electronic noise, thermal noise, sky glow). Stacking hundreds of frames averages the noise toward zero while the real signal accumulates. A session with 400 frames produces a master image with far less noise and far more

recoverable detail than any single frame. This is the core of modern electronic astrophotography.

Version Identifiers

APAS tracks every processing run with a version identifier in the format DSO-###, for example M42-001. The DSO is the target name (M42 = Messier 42, the Orion Nebula). The number is the pipeline version. Combined multi-night versions use the format DSO-C## (e.g., M3-C01). These identifiers appear throughout astromenu and in all output filenames.

File Stage Codes

As a frame moves through the pipeline, each step produces an output file with a short code that indicates its origin:

Code	Stage
SO##	Stack Output — raw Siril stack, linear data
GX##	GraXpert Output — background gradient removed
PO##	Process Output — color calibrated, stretched TIFF
SL##	Starless Layer — stars removed, starless layer sharpened (optional Starless Processing step)
SK##	Star Layer — stars extracted by StarNet2, kept separate
SF##	Starless Final — sharpened starless layer recombined with stars
CI##	Cosmos Input — scaled file staged for Cosmos
CO##	Cosmos Output — tone-mapped TIFF
GI##	Giga Input — staged for Gigapixel
GO##	Giga Output — upscaled TIFF
BX##	BlurXTerminator (RC Astro) -- AI deconvolution on the linear stack. Written to RCXOutput/.
NX##	NoiseXTerminator (RC Astro) -- AI noise reduction on the linear stack. Written to RCXOutput/.
SX##	StarXTerminator (RC Astro) -- AI star removal, producing a starless linear image. Written to RCXOutput/.

3 Using astromenu

astromenu is the interactive hub for day-to-day APAS work. It collects choices, confirms them, and runs the right scripts without requiring command syntax. This chapter starts with the simplest practical workflow — enough to get from a completed imaging session to a processed image — and then introduces the optional tools and stages (GraXpert, Starless Processing, Cosmos, and Gigapixel) one at a time.

Note: Sections 3.2 through 3.5 cover optional tools and stages. Each stands alone. Any stage that is not yet needed can be skipped entirely — the beginner workflow in Section 3.1 produces a complete processed image without them.

3.1 The Beginner Workflow: Scope to Finished Image

This section covers the minimum steps to process a completed imaging session. Everything runs through astromenu using the auto-process option (Ap), which chains all the mandatory pipeline steps automatically. The menu requires about five minutes of interaction; the rest is waiting.

Prerequisites

- The scope has finished its imaging session and is powered on.
- The D_MAIN SSD is connected to the Mac.
- Siril and GraXpert are installed.

Step 1 — Open astromenu

Open Terminal and type:

```
astromenu
```

Typing a alone also works (it is an alias). The menu appears, and the header line shows the active data drive, such as [AstroMaster]. If no drive is shown, confirm that D_MAIN is plugged in and mounted.

Step 2 — Download Data from the Scope

From the main menu, type DI (Data Acquisition). A submenu appears. Type Sc (Scope download).

APAS lists all registered drives with their current mount status and free space, and prompts for a destination. Choose D_MAIN. The system then connects to the scope over WiFi, downloads all FITS frames and the processed preview image, and deposits them in RawTelescope/ on the D_MAIN drive.

Download time depends on the WiFi connection and the number of frames. A typical single-night session with a VesperaPro takes 5–20 minutes. APAS shows an estimated time based on the last few sessions.

Note: Confirm that the Mac is connected to the scope's WiFi access point before choosing Sc. The scope broadcasts its own WiFi network (look for VesperaPro-XXXXXX or Vespera2-XXXXXX in the WiFi list). After the download completes, switch back to the home WiFi.

After a successful download, astromenu automatically asks whether to proceed to stacking. For the beginner workflow, press N — Step 4 uses auto-process instead, which is more thorough.

Step 3 — Ingest the Data

From the main menu, type CI (Collect FITS/TIFFs). A list of un-ingested observations appears. Each row shows the target name, scope, date, and frame counts.

Select the downloaded observation by its number. APAS copies the raw frames into a standardized pipeline directory called M42-001 (named for the target). This is the Ingest step — it organizes the raw data so that Siril can find it.

Step 4 — Run auto-process

From the main menu, type Ap (auto-process). A list of versions ready for processing appears.

Select the version by number. APAS asks for a stretch level, which controls how aggressively faint detail is pulled out of the data. Use these guidelines:

Stretch Level	Best For
1 – 3	Bright targets: M42 (Orion Nebula), M31 (Andromeda), M45 (Pleiades), M8 (Lagoon)
4 – 6	Typical targets: most galaxies and nebulae — M3, M51, M101, NGC 2174
7 – 10	Faint or extended targets: low-surface-brightness galaxies, faint outer halos

When in doubt, use 5. Reprocessing later with a different level is always possible — it is completely safe and non-destructive.

auto-process now runs the core pipeline without further input. It will:

- Score every frame for quality (signal-to-noise, star roundness) and exclude poor ones.
- Stack all the good frames in Siril — aligning, rejecting outliers, and integrating.
- Run GraXpert to remove the sky background gradient.
- Run Siril's SPCC (spectrophotometric color calibration) and apply the auto-stretch.
- Cosmos
- Prepare the best available output file for Gigapixel when upscaling is requested, then prompt to open Gigapixel.

When it finishes, the calibrated, processed image is in ProcessedOutput/. If optional Starless Processing, Cosmos, or Gigapixel stages are completed, their outputs appear in StarlessOutput/, CosmosOutput/, or GigaOutput/. The album step moves the selected output to AstroAlbum/M42/ with a clean filename.

Step 5 — Results

The beginner workflow produces a color-calibrated, background-corrected, auto-stretched TIFF image ready for review, sharing, or additional optional processing. Total active interaction time: about five minutes. Total wall-clock time: 30–60 minutes, most of which is Siril stacking in the background.

Note: To refine the result — tuning gradient removal, applying Starless Processing, tone-mapping a bright target, or upscaling for large prints — see Sections 3.2 through 3.5 in any order.

3.2 GraXpert: Removing the Sky Background

GraXpert is a free, open-source application that uses a neural network to detect and remove sky background gradients from the stacked image. It runs automatically in the beginner workflow. This section explains what it does, when to adjust it, and how to run it manually.

What Is a Background Gradient?

Even from a dark site, the sky is never perfectly black and uniform. Light pollution from distant cities, natural airglow, and internal reflections inside the telescope create a smooth, large-scale brightness variation across the image. After stacking, this shows up as a brighter patch in one corner, a gradient from top to bottom, or an uneven 'dirty' background. GraXpert builds an AI model of this gradient and subtracts it, leaving a flat black sky.

This step matters most when:

- Imaging near a city or from a suburban backyard.
- The target has faint outer structure — outer nebula wisps or galaxy halos — that would be hidden by an uneven background.
- Cosmos tone mapping (Section 3.3) is planned — Cosmos works best on a flat background.

Running GraXpert Manually

In astromenu, go to Ts (Tool Steps) → Gx. Select the version. APAS runs GraXpert on the most recent stack and writes a GX## output file. The Gx step also offers an optional denoise pass (“Also denoise in this pass?”); answer yes and give a strength of 0.0–1.0 (default 0.5) to have GraXpert denoise the linear data right after background extraction. When denoise is applied, sirilprocess later detects this from the GraXpert provenance sidecar and skips its own denoise so the data isn’t denoised twice.

The default smoothing setting (0.5) handles most situations well. If problems appear:

Problem	Fix
Residual gradient — one corner is still brighter than the others	Lower the smoothing (e.g., 0.3). Run Ts → Gx again.

Problem	Fix
Over-correction — dark patches in the background, or the nebula itself is being subtracted	Raise the smoothing (e.g., 0.7). Run Ts → Gx again.
Both problems — gradient in one area, over-correction in another	This usually means the image has a very complex gradient from multiple light sources. Try division correction mode instead of subtraction.

Note: As of the June 2026 update, sirilprocess automatically detects a GraXpert (_GX) input file and skips its own background subtraction, so --no-subsky is no longer required for GraXpert output — the auto-process path, the Process (Pr) menu item, and a direct sirilprocess call all handle it. The --no-subsky flag still works to force the behavior. (Only an older sirilprocess, run by hand on a _GX file without the flag, would subtract the background twice and crush the black point.)

3.3 Cosmos Darkroom: Tone Mapping for High Dynamic Range

Cosmos Darkroom is a paid astrophotography processing application that provides interactive tone mapping. It provides independent control of the bright and faint regions of an image — something a global stretch cannot do. This is most valuable for targets where the core is so much brighter than the surrounding structure that any single stretch level either blows out the center or fails to reveal the outer detail.

When to Use Cosmos

Consider Cosmos when:

- M42 (Orion Nebula): The Trapezium region is extremely bright. Without tone mapping, a stretch level that shows the outer nebula will saturate the four central stars and the core into a white blob.
- M31 (Andromeda Galaxy): Similar bright nucleus versus faint halo problem.
- M8 (Lagoon Nebula), M20 (Trifid Nebula): Bright emission regions with faint surrounding structure.
- Spiral galaxies with color structure (M101, M51, M33): Cosmos color calibration pulls out the blue star-forming regions in spiral arms and neutralizes background tint. Useful for color even when the dynamic range is not extreme — VP data in particular benefits here, as the Siril stretch alone produces a near-monochrome result on these targets.

For targets without extreme dynamic range or strong color structure — globular clusters, elliptical galaxies, and many diffuse nebulae — the sirilprocess auto-stretch often produces a clean result on its own. Treat Cosmos as an optional enhancement: try it when the target justifies it, compare the result against the PO## or SF## output, and keep the version that looks most natural and informative.

The Most Important Setting: Already Stretched

Important: When Cosmos opens the image, it asks whether the data has already been stretched. Always choose ALREADY STRETCHED. The APAS pipeline stretches the data in sirilprocess before it reaches Cosmos. If Cosmos is told the data is linear (not yet stretched), it applies its own stretch on top and the result will be overexposed and unusable.

Running Cosmos via astromenu

From the main menu, go to Ts (Tool Steps) → Cs. Select the version.

6. APAS prepares a CI## input file scaled to fit Cosmos's 500 MB size limit.
7. Cosmos opens with the file loaded. Choose ALREADY STRETCHED.
8. Adjust tone mapping interactively: curves, shadows, highlights, saturation. Typical starting points for a galaxy: Curves gentle midtone boost → Commit. Shadows +20 to +40. Highlights -15 to -20. Saturation +20 to +35.
9. Export a 16-bit TIFF to ~/Downloads/ from within Cosmos.
10. Back in the terminal, press any key. APAS runs cosmos_export to move the TIFF into CosmosOutput/ as a CO## file.

The CO## file is now available for Gigapixel (Section 3.4). Running PrepareGiga next automatically selects the best available source according to the pipeline priority.

3.4 Gigapixel: Upscaling for Large Prints

One Command: giga_finish

giga_finish [DSO-###] runs the whole finishing flow in one step. It stages the best finished file, converts it to 16-bit (Gigapixel cannot open the pipeline's 32-bit-float TIFFs), opens Gigapixel AI, then watches the GigaOutput folder. Export a 16-bit TIFF there with any filename and giga_finish renames it <DSO>_GO##, writes a provenance sidecar linking the GigaInput file, and pushes it into Apple Photos via MoveToAlbum. Flags: --no-wait (just stage and open), --export (file an export already sitting in GigaOutput), --no-album (skip the Apple Photos step). It pins the data drive to AstroRAID; override with GIGA_BASE. The manual PrepareGiga -> RunGiga -> MoveToAlbum steps below still work for fine-grained control. Default version is NGC6992-001.
Topaz Gigapixel is a paid upscaling application that uses neural-network-based image enhancement to increase image dimensions and preserve perceived detail at larger display scales. Gigapixel was not specifically trained on astrophotography data, but it can perform well on processed astrophotography images when used conservatively. A 2x upscale takes a VesperaPro image from roughly 9 megapixels to approximately 60 megapixels, supporting prints up to 20" x 24" at 300 DPI.

When to Use Gigapixel

Use Gigapixel to print large, create a detailed crop, or share a high-resolution version of the image. For social media, email, or quick sharing, the unscaled PO## or CO## TIFF is usually sufficient.

Recommended Settings

Setting	Value	Notes
Model	Low Resolution	Often produces a natural result on VesperaPro images; compare with Standard on new targets.
Scale	2x	2x is the usual starting point; 4x rarely adds useful information and creates much larger files.
Sharpen	50 – 60	Use conservatively; excessive sharpening can create halos around stars.
Denoise	8	Light denoising; the stack already handled most noise
Gamma Correction	ON	Often useful for preserving luminance handling; verify visually on each target.
Fix Compression	0	Use 0 for TIFF input; only relevant for JPEG input

Running Gigapixel via astromenu

From the main menu, go to Ts (Tool Steps) → Gp. Select the version.

11. APAS selects the best available processed file (CO## if available, otherwise SF## if available, otherwise PO##) and stages it as a GI## file in GigaInput/.
12. Gigapixel opens with the file loaded.
13. Apply the settings above and export the result to ~/Downloads/.
14. Back in the terminal, press any key. APAS moves the upscaled TIFF to GigaOutput/ as a GO## file and then copies it to AstroAlbum/DSO/ as the selected image.

After Gigapixel export, processing is complete for that selected output. The album copy is the presentation-ready image.

3.5 Starless Processing with StarNet2 (optional)

The Starless Processing step uses StarNet2 to split a processed image into two layers — the nebulosity or galaxy structure with the stars removed, and the stars on their own — so the faint structure can be sharpened without enlarging star cores. It then recombines the two layers into an SF## output. This is an optional enhancement stage that runs after sirilprocess (the Pr step) and before Cosmos or Gigapixel.

Why Separate the Stars?

Sharpening a normal image sharpens everything — including the stars, whose bright cores swell into bloated discs and dark rings. By removing the stars first, the sharpen pass acts only on the nebula or galaxy, pulling out fine dust lanes and filaments, and the untouched stars are screened back on top afterward. The result is crisp structure with clean, tight stars.

One-Time Setup: the StarNet2 Binary

The Starless Processing step drives the StarNet2 command-line program (v2.5 or later) directly. No StarNet2 path is set inside Siril's Preferences — that older method no longer applies. Install the starnet2 binary; APAS auto-detects it at ~/StarNet/starnet2 (and a few other common locations), or it can be specified with --starnet-bin. Siril is still used internally to make the 16-bit TIFF StarNet2 requires and to perform the sharpen and recombine.

Running Starless via astromenu

From the main menu, go to Ts (Tool Steps) → Sl (Starless). Select a version that already has a Siril-processed PO## file.

- APAS asks for a sharpen radius and amount (default "1.0 0.7"; enter "0 0" to skip sharpening and just split/recombine).
- siril_starless runs StarNet2, sharpens the starless layer, and recombines. It writes three files to StarlessOutput/: the starless layer (SL##), the star layer (SK##), and the recombined final (SF##), each with a provenance sidecar.
- When it finishes, APAS offers to stage the SF result straight to Cosmos (it runs PrepareCosmos --from-starless). Answer yes to continue with Ts → Cs, or no to stop with the SF## file in StarlessOutput/.

Tip: For the best result, create the PO## input with sharpening deferred — at the Process (Pr) step, answer yes to "Defer sharpening to the Starless step?" (or run sirilprocess --no-sharpen). That way the image is sharpened only once, here on the starless layer, instead of twice.

```
siril_starless M42-001 --sharpen "1.0 0.7"    # command-line
equivalent
```

4 The Complete Workflow

This chapter presents every pipeline option in logical order, first as an astromenu walkthrough and then as a sequence of individual commands. If a step is optional, that is noted — along with guidance on when to use or skip it. Examples use M42-001 (Orion Nebula, version 1) as the working target.

4.1 Full Pipeline with astromenu

Work through the menu in this order. Not every step applies to every session — the optional steps are marked.

Step 1 — Data Acquisition (DI)

From the main menu, type DI. The Data Acquisition submenu offers:

Code	Option	When to Use
Sc	Scope download	After an imaging session. Connects to each powered-on scope over WiFi and pulls all FITS frames and the processed preview image.
Ms	Mule sweep	When data was captured on a Mac with the Mule shuttle drive (4TB-Astro-Mule on the M5 Air). Copies everything to D_MAIN via SSH.
Cs	Camel sweep	Same as Mule sweep but for the Camel drive (T7-Astro-Camel on the M3 Air).
As	All sweeps	Runs Mule and Camel sweeps back-to-back. Use when multiple shuttles have data.

Why pull FITS and not just the preview JPG? The scope's on-board preview is produced by the scope's limited processing. A Siril stack, using all the raw FITS frames, has significantly better noise performance, finer detail, and accurate color calibration. The preview is only useful as a quick visual check.

Step 2 — Collect FITS/TIFFs (CI) — or use auto-process to do it automatically

From the main menu, type In. A list of observations not yet ingested into the pipeline appears, organized by target, with frame counts and dates.

Select the observation(s) to ingest — individual observations, a range (e.g., 1 3), or all (0). APAS runs astrostart to create the pipeline directory structure (M42-001), then CollectFits (or CollectTiffs for TIFF-mode observations) to copy the raw frames in, then regen_sources_json to record provenance.

Why have a separate Ingest step? Ingestion creates a clean, deduplicated, standardized input for Siril. If the same target is shot on three different nights, ingestion merges all three sessions' frames into a single M42-001 directory, with MD5 content

hashing to prevent duplicates. It also allows selection, before stacking, of which sessions to include.

Step 3 — Frame Scoring (optional, before stacking)

From the main menu, type SS (Score & Stack) or Sc (Score Frames). Scoring measures each frame's signal-to-noise ratio and star roundness, then writes a sidecar file that the stacker uses to exclude poor frames.

Use scoring when:

- The night had variable seeing or passing clouds — some frames will be noticeably softer than others.
- The imaging session was long (many hours) and atmospheric conditions changed.
- Data from multiple nights with different conditions is being combined.

Skip scoring when:

- The night was consistently good — scoring takes time and for excellent data it changes little.
- A quick result is needed. auto-process (Ap) always runs scoring automatically.

Step 4 — Stack (St)

From the main menu, type St. A list of DSOs with raw frames waiting in RawTelescope appears. Select a DSO, then select which observations to include (individual, range, or all). At the format prompt, choose:

Key	Format	When to Use
F	FITS only	VesperaPro and Vespera 2 data captured with FITS enabled — the standard path
T	TIFF only	Older Vespera 1 data, or sessions where FITS was not enabled
B	Both	When two independent stacks are wanted — one from FITS, one from TIFFs — for comparison

Stacking typically takes 5–30 minutes depending on frame count and the Mac's speed. The output is a FITS file (SO##) in StackingOutput/. It looks dim and strange when viewed directly — that is normal. It is linear data that has not yet been stretched.

Advanced options available at the Stack step:

- Combine (Cb): Pool frames from two or more pipeline versions before stacking. Use this for multi-night data. APAS merges the raw frames, deduplicates them, and creates a combined version M42-C01 that then proceeds through stacking normally.

- Dual Stack (Du): Runs two Siril stacking jobs simultaneously on two different Macs, using SSH to offload work to the M5 Air. Cuts stacking time roughly in half when working with very large frame counts.

Step 5 — Process (Pr)

From the main menu, type Pr. Select the version. APAS runs:

15. GraXpert — removes the sky background gradient (produces GX## file).
16. sirilprocess — runs Siril's SPCC spectrophotometric color calibration, then applies the auto-stretch (produces PO## TIFF).

At the stretch level prompt, use the guidance in Section 3.1. The Pr step then asks “Defer sharpening to the Starless step?” — answer yes only if the Starless step (Section 3.5) will be run next, so the image is sharpened once on the starless layer; otherwise answer no and sirilprocess sharpens normally. After processing completes, inspect the PO## file in Preview or another image viewer. If the stretch is too aggressive (grey background, blown-out core) or too conservative (faint nebulosity disappearing), run Pr again with a different level. Each run produces a new PO## file — the old one is never overwritten. The Pr step also asks “Apply deconvolution (Richardson-Lucy)?” — answer yes to sharpen fine detail on the linear data before the stretch (start with the default 10 iterations, and lower it if dark rings appear around bright stars).

Why run GraXpert before Siril processing, not after? GraXpert works on the linear SO## stack, before any stretch has been applied. The background gradient is easier to model and remove from linear data. Stretching first would amplify the gradient and make it harder for the AI to distinguish between background and faint nebula structure.

Step 6 — Starless Processing (optional)

Use Starless Processing to sharpen faint nebulosity, dust lanes, or galaxy structure without enlarging the stars. From the main menu, go to Ts → Sl after the Process step. APAS uses the PO## file as input and writes SL##, SK##, and SF## outputs. When Cosmos will follow Starless Processing, stage the SF## output rather than the original PO## file.

Step 7 — Cosmos Tone Mapping (optional)

For high dynamic range targets (M42, M31, M8), go to Ts → Cs after the Process step, or after Starless Processing when an SF## output exists. See Section 3.3 for full instructions.

Skip this step for: elliptical galaxies, globular clusters, and diffuse nebulae. For spiral galaxies with color structure (M101, M51, M33), consider running Cosmos for color calibration even when HDR tone mapping is not needed — the Siril stretch alone leaves these targets near-monochrome.

Step 8 — Gigapixel Upscaling (optional for large prints)

Go to Ts → Gp. See Section 3.4 for full instructions. PrepareGiga automatically picks the best available source: CO## if Cosmos was run, otherwise SF## if Starless Processing was run, otherwise PO##. This can be overridden with the --source flag on the command line (see Section 4.2).

Step 9 — Review and Status

At any point during or after the pipeline, use the Display Info submenu (Di):

Code	What It Shows
Di → As	Full pipeline status for all versions — which stages have output files
Di → DS	Status for a single target
Di → Pv	Provenance for a version — source observations, frame counts, scope used
Di → Lg	Pipeline operations log — history of every run
Di → Nr	Versions flagged as needing color re-calibration (Vizier was offline)

The Preview Album option (PA from the main menu) displays selected images directly in the terminal using a low-resolution ASCII or pixel preview.

4.2 Full Pipeline with Individual Commands

Every astromenu option has a direct command-line equivalent. This section shows the complete pipeline as shell commands, with the same M42-001 example. Use the command line to automate, script, or jump into the middle of the pipeline after an interruption.

Step 0 — Pull Data from Scope

```
# Pull all powered-on scopes over WiFi
astro_allpull --with-fits --dest D_MAIN

# Pull from a single scope only
vespera_pull vp1 --with-fits

# Sweep shuttle drives
mule_sweep          # copies Mule (M5 Air) → D_MAIN
camel_sweep         # copies Camel (M3 Air) → D_MAIN
sweep_all           # both at once
```

Scope pulls download the raw FITS frames, the TIFF, and the processed JPG by default — the raw FITS are the stacking source, so the default captures everything required. Pass --no-fits only to retrieve the preview images without the raw frames.

Step 1 — Create the Version and Ingest Frames

```
# Create the pipeline directory for M42
astrostart M42

# Copy FITS frames from RawTelescope into StackingInput/M42-001/
```

```
CollectFits M42-001 --fresh

# Rebuild the provenance sidecar (CollectFits now runs this
automatically; call by hand only to rebuild)
regen_sources_json M42-001

# For TIFF-mode observations (Vespera 1 or TIFF-only sessions):
CollectTiffs M42-001 --fresh
```

The --fresh flag clears the destination directory before collecting. Use it for a clean start. Without it, CollectFits adds only new frames (by MD5 hash), which is safe and useful when adding a new night's data to an existing version.

Step 2 — Score Frames (optional)

```
score_frames M42-001
```

score_frames writes a scored sidecar file that sirildispatch reads to exclude poor frames. Run it before stacking when data quality is variable.

Step 3 — Stack

```
# Interactive format chooser (prompts for T, F, B, or S)
sirildispatch M42-001

# OR specify format directly (for scripted use)
sirildispatch M42-001 --default F

# OR call the stacker directly
stackdso_fits_v1 M42-001       # FITS path (VesperaPro, Vespera 2)
stackdso M42-001                    # TIFF path (Vespera 1)
```

Output: StackingOutput/M42-001_SO01_fits.fit

Combining multiple sessions before stacking:

```
# Pool nights 1 and 2 into a combined version M42-C01
astrocombine M42-001 M42-002

# Then stack the combined version
sirildispatch M42-C01 --default F
```

Step 4 — GraXpert Background Removal

```
# Standard FITS stack
graxpert_clean M42-001 --stack-flavor fits

# If gradient is stubborn, try lower smoothing
graxpert_clean M42-001 --stack-flavor fits --smoothing 0.3

# If GraXpert over-corrects, back off
graxpert_clean M42-001 --stack-flavor fits --smoothing 0.7
```

Output: GraXpertOutput/M42-001_GX01_fits.fit

Important: sirilprocess now auto-detects a GraXpert (_GX) input file and skips its own subsky automatically, so the --no-subsky flag is no longer required — it is shown above for clarity and still works when an explicit flag is preferred. (Only an older sirilprocess without this auto-guard would run two background subtractions and destroy the black point.)

Step 5 — Color Calibration and Stretch (sirilprocess)

```
# Standard path after GraXpert
sirilprocess M42-001 vp1 \
  --input-file
$ASTRO_BASE/PhotosData/Workflow/GraXpertOutput/M42-
001_GX01_fits.fit \
  --no-subsky --stretch-level 3

# Without GraXpert (Siril handles background removal internally)
sirilprocess M42-001 vp1 --stretch-level 3

# Scope key options: vp (VP-1/VP-2), v2 (Vespera2-1/Vespera2-2), v1
```

Output: ProcessedOutput/M42-001_PO01_fits.tif

If the result needs a different stretch, run sirilprocess again. It creates PO02, PO03, etc. and never overwrites a previous result.

What sirilprocess does in detail: First it checks that the Vizier star catalog server (tapvizier.u-strasbg.fr) is reachable — SPCC needs it for color calibration. If Vizier is offline, it pauses and offers three options: R to retry, Y to proceed without color calibration, or any other key to abort. When processing continues without color calibration, the image is still fully processed but carries a flag noting it needs re-calibration when Vizier is back. It then plate-solves the image (determines exactly which patch of sky was photographed), identifies calibration stars by their known spectral types, and adjusts the RGB balance to match physical color. Finally it applies the auto-stretch and writes the TIFF.

Stretch Engines — MTF and GHS (Advanced)

sirilprocess can apply its stretch with either of two engines, chosen with the --engine flag. The default is MTF (Siril's autostretch, or Midtones Transfer Function): it sets a black point and target background automatically and gives a smooth, reliable result with no tuning. GHS (autoghs, or Generalized Hyperbolic Stretch) is an optional, manual alternative that can apply contrast selectively to protect bright cores while lifting faint signal, but it is sensitive to its settings and needs the --ghs-* knobs described below to behave well.

A 2026 study compared the two engines across object types: a globular cluster (M3), a bright galaxy (M104), and an emission nebula (the Rosette). MTF gave the best hands-off result every time. GHS was flat and lifeless on the cluster, finicky on the galaxy, and only reached a draw on the emission nebula — where it rendered faint nebulosity more smoothly but with less apparent detail, and even then required precise hand-tuning. The conclusion: MTF stays the universal default, and GHS is kept as a manual option for

deliberately hand-crafting a nebula. Nothing in the everyday workflow changes — sirilprocess already defaults to MTF.

To use GHS manually, add --engine ghs together with its tuning knobs: --ghs-sp (symmetry point — where the stretch is centered; 0 sits at the background, higher values darken the sky but can make a faint target vanish), --ghs-d (strength, 0 to 10), --ghs-b (local width — 0 is broad and gentle, around 8 is punchy; the built-in default of 13 is extreme), and --ghs-hp (highlight protection — lower values keep stars and bright cores tighter). The tuned recipe that worked best for an HOO narrowband nebula was: --engine ghs --ghs-sp 0 --ghs-d 6 --ghs-b 8 --ghs-hp 0.7, run with --bicolor.

Two research tools support this experimentation: siril_ghs_sweep, which sweeps the GHS knobs and opens every result in Preview, and siril_compare_engines, which renders a target both ways for a side-by-side comparison. These are research instruments, not part of the normal workflow. For a quick reference, run astrohelp stretch. The full study lives in Tools/docs/GHS_STRETCH_RESEARCH.md, and the underlying concepts (what symmetry point, strength, local width, and highlight protection actually do) are explained in Tools/stretching_tutorial.md.

Step 6 — Starless Processing (optional, command line)

Run Starless Processing after PO## exists
siril_starless M42-001 --sharpen "1.0 0.7"
Output: StarlessOutput/M42-001_SF01.tif (plus SL## and SK## layers)
Use Starless Processing when the target benefits from sharpening nebulosity, dust lanes, or galaxy structure separately from the stars. When Cosmos follows, stage the SF## result; otherwise keep the PO## output.

Step 7 — Cosmos Tone Mapping (optional)

Stage the best available file as a Cosmos-compatible input (SF## if using Starless Processing, otherwise PO##)
PrepareCosmos M42-001 --from-starless --stack-flavor fits # use --from-po if Starless was not run

```
# Open Cosmos with the CI## file
RunCosmos M42-001
# [In Cosmos: choose ALREADY STRETCHED, adjust, export 16-bit
TIFF to ~/Downloads/]

# Move the export into the pipeline
cosmos_export M42-001
```

Output: CosmosOutput/M42-001_CO01.tif

Why use PO## or SF##? PrepareCosmos can stage the GX## stack directly, but starting from a color-calibrated and stretched PO## or SF## gives a much better preview in Cosmos — the tone-mapping controls are far more intuitive on stretched data.

Step 8 — Gigapixel Upscaling (optional)

```
# Stage the best available file as GI## (auto-selects CO## > SF## > PO## > GX##)
PrepareGiga M42-001

# Force a specific source
PrepareGiga M42-001 --source po    # use PO## even if CO##
exists
PrepareGiga M42-001 --source co    # use CO##

# Open Gigapixel
RunGiga M42-001
# [In Gigapixel: apply settings from Section 3.4, export to
~/Downloads/]
```

Output: GigaOutput/M42-001_GO01.tif

Step 9 — Move to Album

Shortcut: giga_finish <DSO-###> replaces Steps 8-9 for the common case. It stages the file, converts to 16-bit, opens Gigapixel, then auto-files the export (rename to _GO##, write provenance) and runs MoveToAlbum into Apple Photos. See section 3.4.

```
# Copy the selected output image to AstroAlbum/M42/
MoveToAlbum M42-001
```

MoveToAlbum selects in priority order: GO## (Gigapixel output), then CO## (Cosmos output), then SF## (Starless final), then PO## (sirilprocess output). It copies the selected file into AstroAlbum/M42/ with a standardized filename that encodes the target, date, and scope.

One-command Shortcut: astro_autoprocess

RC Astro options (require the BlurXTerminator / NoiseXTerminator / StarXTerminator plugins): --blurx, --noisex, and --starx run on the LINEAR stack before Siril, producing BX/NX/SX files in RCXOutput/. --starx yields a starless PO. These are the automatic pipeline's default sharpening, denoise, and star-removal path.
For a hands-off run through the core processing path, use astro_autoprocess. It chains the mandatory steps (scoring, stacking, GraXpert, and sirilprocess) and then handles optional output preparation prompts as configured:

```
astro_autoprocess M42-001 vp1 --format fits --stretch-level 3 --
graxpert
```

The --graxpert flag includes the GraXpert step and automatically adds --no-subsky. Without the flag, GraXpert is skipped.

Additional examples:

```
# Galaxy with typical stretch, GraXpert
astro_autoprocess M101-001 vp1 --stretch-level 7 --graxpert

# Ingest and process in one command
astro_autoprocess NGC2174-001 v2b --collect --format fits --
stretch-level 5
```

```
# Combined version from two nights
astro_autoprocess M3-C01 vp1 --format fits --stretch-level 6 --
graxpert
```

Why Use Individual Commands Instead of astromenu?

astromenu is the right choice for most sessions. Individual commands are better when:

- Automating: Commands can be chained in a shell script for unattended overnight processing.
- Non-standard inputs: The --input-file flag on sirilprocess accepts any file, not just the latest output.
- Recovery: If a step failed, processing can resume at exactly the right point without re-running everything.
- Batch processing: A simple for loop processes multiple versions overnight.

```
# Batch example: reprocess all versions in StackingInput with
new stretch
for ver in M3-001 M42-001 NGC891-001; do
  sirilprocess $ver vp1 --stretch-level 5 --no-subsky
done
```

4.3 Publishing Finished Images to the Web

APAS publishes finished images to a public website and maintains a companion imaging-calendar web app, both hosted on the astro.milligangridsolutions.com subdomain. Publishing is integrated into the workflow: a finished target can be added to the online gallery directly from the dashboard board or the Finishing menu, and the planning calendar can be regenerated and uploaded from a single menu action. This section describes the gallery, the pipeline publishing actions, the calendar app, and the deployment mechanism.

4.3.1 The Image Gallery

The gallery presents finished images in a dark, responsive, lightbox layout at the subdomain root. It is generated by Tools/gallery/build_gallery.py, which discovers full-size exports placed in the gallery source/ directory — or accepts a single image passed in from the pipeline — and produces two web-sized derivatives with the macOS sips utility (with a Pillow fallback): a display copy of at most 2048 pixels on the long edge and a thumbnail of at most 600 pixels. The derivatives are written to images/ and images/thumbs/; the original masters are never uploaded.

Image metadata is recorded in gallery.json, one entry per image, carrying the object, scope, filter, integration time, capture date, and caption. A new image receives a blank metadata stub automatically, ready to be completed. The generator renders the entire index.html from this data, using the thumbnails for the grid and the display copies for the lightbox. A single image can be added from the command line with the add subcommand:

```
build_gallery.py add M3-C07_GO03.tif --object M3
```

4.3.2 Publishing from the Pipeline

Two workflow entry points publish a finished target without leaving astromenu. On the per-target dashboard board (the “what next?” screen), the (W) Publish to Web action resizes the target's final image and adds it to the gallery. In the Finishing menu, (Pw) Publish to Web Page performs the same operation and prompts for the target by name. Both call build_gallery.py add on the detected final image and then offer to deploy the updated gallery.

The final image is selected automatically by the shared final-image resolver, which searches the processing stages in finishing order — Album, then Gigapixel, then Cosmos, then Siril — takes the newest match, and reports which stage it came from. The related dashboard action (V) View in Preview opens a chosen version in macOS Preview for review before publishing; the version can be the Siril stretch (PO), the Cosmos output (CO), the Gigapixel output (GO), or Automatic, which selects the most-processed available file (GO, then CO, then PO).

The Finishing menu also surfaces (Ma) Move to Album, which copies the final GO, CO, or PO file into AstroAlbum/<DSO>/ and imports it into Apple Photos. This action shares the same MoveToAlbum tool that appears under File Management; it is presented in the Finishing menu because that is where it belongs in the workflow.

4.3.3 The Imaging Calendar Web App

A companion mobile web app publishes the imaging calendar to the /calendar/ path of the subdomain. It is generated by Tools/app/build_calendar_app.py, which produces a self-contained index.html with its data baked inline, along with calendar_app_data.json, from the same imaging_calendar.json used by the desktop calendar. The app reuses the calendar's visibility and plus-or-minus one-month carryover engine, so targets curated for an adjacent month that are still imageable appear alongside the current month.

The interface is a mobile card layout defaulting to the current month, with month-to-month navigation and a live “up now” altitude that is recomputed each minute from each target's coordinates. It includes a night-vision red mode, a Denver / South Park site toggle, and filters for sky region, scope, priority, dual-band targets, and current visibility. A search box queries a baked index of roughly 1,600 named and bright deep-sky objects that clear 30 degrees from Colorado, each annotated with a January-to-December visibility strip that highlights its best months. The app is rebuilt and uploaded from astromenu under Observation Tools & Planning → (Vp) Publish App.

4.3.4 Deployment and Credentials

Both the gallery and the calendar app deploy over HTTPS to cPanel. The gallery uses Tools/gallery/deploy_gallery.sh and the calendar app uses Tools/app/deploy_calendar_app.sh; both upload through cPanel's UAPI Fileman::upload_files endpoint on the cPanel port (2083), authenticated with a scoped cPanel API token. Uploads must pass overwrite=1, because cPanel refuses to replace an existing file otherwise. This HTTPS path is used because the hosting plan does not offer SFTP and the network blocks the passive-data ports that plain FTP and FTPS require.

Credentials are stored in Tools/app/deploy.conf with 0600 permissions and are shared by both deploy scripts. The stored secret is a scoped cPanel API token rather than the account password, and it is rotated after setup. The subdomain's document root is the astro.milligangridsolutions.com folder at the account's home level, not under public_html.

5 Hardware & Drives

The pipeline can run across a small cluster of Apple Silicon Macs, but the core APAS workflow does not require every node to be online. Each machine has a defined role: the MacBook Pro is the primary imaging workstation and AstroMaster host; the Mac mini provides optional archive and catalog services; the two MacBook Airs are secondary processing nodes and shuttle drive hosts. Roles are enforced by the environment configuration — most scripts will warn or refuse if run on the wrong machine.

5.1 Machines

Machine	Role	Network Name	IP
MacBook Pro (Pro)	Primary workstation; AstroMaster plugged in during sessions; runs astro_allpull and sweeps	ASTRO_PRO_HOST	10.0.0.218
M5 Air (Air)	Secondary node; holds Mule shuttle drive	Astro-M5Air.local	10.0.0.172
M3 Air	Secondary node; holds Camel SSD	Astro-M3Air.local	—
Mac mini M2 (mini)	Optional archive and catalog hub; AstroArchive attached; runs astro_netwatch and astro_catalog daemons (C_MINI)	`astro-mini.local`	10.0.0.224
AstroDen (Den)	No active role at present. Former primary stacker (ASTROHUB); superseded by the M4 Mini (C_STACK). Its 8 TB RAID was relocated to the M4 Mini and is now AstroRAID.	AstroDen.local	—
Mac mini (M4, Stack)	Primary stacking node -- runs the stack_sync auto-pipeline and hosts the AstroRAID RAID. Replaced AstroDen as the primary stacker.	Astro-M4Mini.local	--

5.2 Drives

Label	Volume Name	Type	Role
AstroMaster	AstroMaster	4 TB SSD (Nextorage)	Primary working drive (D_MAIN) — replaced the retired T9 on 2026-06-04.
Mule	4TB-Astro-Mule	4 TB SSD (travel)	Secondary working drive — shuttle, normally on M5 Air
Camel	X10-Astro-Camel	4 TB SSD (travel) — new drive (X10), replaced T7	Working SSD — normally on M3 Air
AstroRAID	AstroRAID	8 TB RAID (M4 mini)	Auto-pipeline volume on the M4 Mini (C_STACK) — raw

Label	Volume Name	Type	Role
			mirror + Workflow outputs; giga_finish / MoveToAlbum auto-resolve target. This is the 8 TB RAID relocated from AstroDen; formerly named AstroWork.
AstroArchive	AstroArchive	16 TB HDD (mini)	Long-term archive — Sessions/ + DSO/ layout; populated by astro_catalog
AstroScratch	AstroScratch2TB	2 TB SSD (Nextorage)	Per-node Siril scratch (Pro). Auto-selected by astrorc when D_MAIN is connected. Never holds scope data.

5.3 AstroRAID (8 TB RAID, formerly AstroWork)

Note: this 8 TB RAID was relocated from AstroDen (C_DEN) to the M4 Mini (C_STACK), where it is now presented as AstroRAID and used as the auto-pipeline volume (see section 5.1). AstroDen has been superseded as the primary stacker. The recreation steps below apply to this RAID on its current host (the M4 Mini) — verify the member-disk identifiers there before running them.
AstroWork-8TB is an APFS concatenated RAID of two 4 TB SSDs on the mini. It was created with:

```
diskutil appleRAID create concat AstroWork-8TB APFS disk4 disk5
```

It follows the same Astronomy/PhotosData/RawTelescope layout as AstroMaster, so astromenu treats it identically to a travel drive. If it needs to be recreated, erase both member disks to ExFAT first:

```
diskutil eraseDisk ExFAT TEMP disk4
diskutil eraseDisk ExFAT TEMP2 disk5
diskutil appleRAID create concat AstroWork-8TB APFS disk4 disk5
```

6 Shell Environment

The entire pipeline is coordinated through a shared shell environment sourced on every machine at login. Two files provide this environment: astrorc (sourced from ~/.zshrc) and astroenv.zsh (sourced by every script at the top of its shebang line). Understanding these two files is important for diagnosing problems when a script misbehaves or a drive is missing.

6.1 astrorc

astrorc is the login-time environment configurator. It lives at $ASTRO_CODE/Tools/bin/astrorc and is sourced by ~/.zshrc on every machine. At shell startup it:

- Sets ASTRO_CODE to the Dropbox Astronomy folder.
- Reads drive_registry.json and detects the first mounted registered drive — sets ASTRO_BASE, ASTRO_PHOTOS, ASTRO_WORKFLOW.
- Adds ASTRO_BIN to PATH and symlinks all scripts to ~/bin for convenient access.
- Sources ~/.config/astro/config.zsh for per-machine scope WiFi and host config.
- Prompts for terminal background color (R/B/G or Enter for none).
- cd's into ASTRO_BASE if a data drive is mounted. Also defines the remote <name> SSH shortcut (alongside mini/pro/air5/air3): remote M3, remote M5, remote Pro, remote Mini — case-insensitive, refuses if already on target. SSH sessions switch Terminal to black bg / white text for the session.

6.2 astroenv.zsh

astroenv.zsh is the script-time environment bootstrapper. It is sourced as the very first line of every pipeline script: source "${0:A:h}/astroenv.zsh". It resolves ASTRO_BASE from the drive registry (or falls back to the D_MAIN volume path for standalone invocations), sets all the derived path variables, and verifies that a data drive is mounted before any destructive operation runs.

6.3 Key Environment Variables

Variable	Example Value	Description
ASTRO_CODE	~/Library/CloudStorage/Dropbox/Astronomy	Scripts and config (Dropbox)
ASTRO_BIN	`$ASTRO_CODE/Tools/bin`	All executable scripts
ASTRO_BASE	/Volumes/AstroMaster/AstroData	Root of active data drive
ASTRO_PHOTOS	`$ASTRO_BASE/PhotosData`	PhotosData folder
ASTRO_WORKFLO	`$ASTRO_BASE/PhotosData/`	Stacking and

Variable	Example Value	Description
W	`Workflow`	processing folders
ASTRO_ARCHIVE_VOL	/Volumes/AstroArchive	Archive drive mount point
ASTRO_STACK_DIR	/Volumes/AstroScratch2TB/AstroStack	Per-node Siril scratch, auto-resolved by astrorc from the registry (never the data drive)
ASTRO_SCOPE_HOST	10.0.0.1	Vaonis scope AP address
ASTRO_MINI_HOST	`astro-mini.local`	Mac mini hostname
ASTRO_AIR_HOST	Astro-M5Air.local	M5 Air hostname
ASTRO_M3_HOST	Astro-M3Air.local	M3 Air hostname

6.4 Navigation Shortcuts

These shell functions (defined in astrorc) cd into the relevant pipeline directory. They require a data drive to be mounted.

Command	Destination
goastro / astronomy	`$ASTRO_BASE`
workflow	`$ASTRO_WORKFLOW`
`stackin`	`$ASTRO_WORKFLOW/StackingInput`
`stackout`	`$ASTRO_WORKFLOW/StackingOutput`
gigain	`$ASTRO_WORKFLOW/GigaInput`
gigaout	`$ASTRO_WORKFLOW/GigaOutput`
processed	`$ASTRO_WORKFLOW/ProcessedOutput`
album	`$ASTRO_PHOTOS/AstroAlbum`
bin / astrobin	`$ASTRO_BIN`
gocode	`$ASTRO_CODE`

7 Scripts Reference

This section documents every pipeline script. Scripts are grouped by function: interactive menus, data acquisition, ingestion, stacking, processing, archiving, daemons, and utilities. All scripts live in $ASTRO_BIN (Tools/bin/) and are symlinked to ~/bin at login. Scripts that accept a <DSO-###> argument always expect the version identifier (e.g. M42-001), not just the DSO name.

7.1 astromenu

astromenu is the interactive hub for all day-to-day astrophotography work. It presents a two-character menu of every pipeline operation, handles argument collection, confirms choices, and shells out to the appropriate scripts. Launch it from any terminal with:

```
astromenu    # alias: a
```

The header line always shows which drive is active (e.g. [AstroMaster]). Items that require the D_MAIN data drive are guarded — if D_MAIN is absent, astromenu displays a warning and returns to the main menu rather than silently failing. Menu codes are typed without pressing Enter; if two items share a first letter, astromenu dims the others and waits for the second letter.

7.1.1 Main Menu Reference

Code	Item	D_MAIN Req?	Description
Dl	Data Acquisition	No	Pull scope data over WiFi; route to disk automation
In	Ingest	No	Collect raw frames into StackingInput for a version
St	Stack	Yes	Collect TIFFs/FITS → run Siril stacking
Cb	Combine	Yes	Pool 2+ versions into a combined version, then stack
R	Review Frames	Yes	Generate per-frame PDF review sheets
Sc	Score Frames	Yes	Quality-score individual frames; flag keepers/rejects
SS	Score & Stack	Yes	Score → pick best frames → stack in one flow
Du	Dual Stack	Yes	Stack same version on Pro + Air in parallel
Pr	Process	Yes	GraXpert gradient removal

Code	Item	D_MAIN Req?	Description
			→ Siril SPCC + stretch → PO output
Cs	Cosmos	Yes	Tone-map HDR targets in Cosmos Darkroom: PO/SF → CI → CO
Gp	Gigapixel	Yes	GI → GO
Ap	auto-process	Yes	Core pipeline: score → stack → process; optional output preparation
Ts	Tool Steps	No	Run a single pipeline step
Di	Display Info	Yes	Submenu: status, flow, DSO info, logs, lineage, provenance
U	Un-ingested	Yes	Show observations not yet in StackingInput
Cu	Current Projects	No	Active imaging targets with visibility windows
Vc	Viewing Calendar	No	Monthly best targets for Denver & South Park, CO. Integration shown as h:mm; Az R→S column shows rise/set azimuth.
Cf	Compare File Tree	No	Verify a backup copy matches its source
Co	Check Out DSO	No	Check out a DSO to Mule/Camel for remote work on M3/M5 Air
Ci	Check In DSO	No	Sync changes back to D_MAIN and clear the checkout lock
Fm	File Management	No	Disk pull, Cosmos/Giga exports, archiving, album moves (submenu within Ut)
Ut	Utilities & File Mgmt	No	Maintenance, audits, drive registry, migrations; File Management submenu (Fm

Code	Item	D_MAIN Req?	Description
			→)
PA	Preview Album	No	Quick-preview AstroAlbum images at the command line
H	Help	No	Show the astrohelp reference sheet
Q	Quit	—	Exit astromenu

7.1.2 Data Acquisition Submenu (DI)

Scope download (Sc) is the live option: lists all registered drives with mount status and free space, asks a destination, then calls astro_allpull to pull the latest observation from every powered-on Vespera scope over WiFi. An estimated download time is shown based on a rolling average of the last five sessions for each scope. Disk download (Dk) is dimmed — handled automatically by astro_netwatch and astro_catalog without user action. After any successful pull to D_MAIN, astromenu automatically runs Collect FITS/TIFFs (equivalent to the CI step) and, when newly collected frames join existing versions, triggers an auto-combine pass — so stacking can begin immediately from the Stack (St) or Process (Pr) menu without a separate collection step.

7.1.3 Stack Submenu (St)

Shows every DSO with raw TIFFs or FITS waiting in RawTelescope: observation count, total frame counts, and most-recent session date. Select a DSO by number to see the observation sub-list; each row shows scope, date, frame counts, and a format badge: [T] TIFFs, [F] FITS, [B] both, [---] empty. At the selection prompt type a number or range (e.g. "1 3"), 0 for all, F to collect FITS only, or t for TIFFs only.

7.1.4 Process Submenu (Pr)

Lists all versions in StackingOutput, optionally filtered by scope type (V1/V2/VP/ALL). For the selected version, runs: Siril SPCC color calibration → GraXpert background extraction → Cosmos darkroom export in sequence. The Pr filter tokens Un (unprocessed only) and Pr (processed only) narrow the list quickly. A persistent DSO-name filter is also available: type a target designation (e.g. M101, NGC2237) to restrict the list to that target's versions, or ALL to clear. The GraXpert (Gx) tool-step menu supports the same filter and also shows a frame-count column drawn from each stack's provenance sidecar.

7.1.5 auto-process Submenu (Ap)

Chains the full pipeline end-to-end for a version already in StackingInput. Steps: frame scoring, stacking, GraXpert, sirilprocess, optional Cosmos, PrepareGiga, RunGiga, MoveToAlbum. The Un-ingested (U) auto-process path automatically includes --graxpert. Enter a comma-separated list of row numbers (e.g. "1,3") to combine multiple

versions into a combined version (DSO-C##) which then proceeds automatically. Single-frame versions (test shots) are flagged visually so they can be skipped. The opening astromenu screen now includes a Recent Projects section showing up to six DSOs with activity in the last 14 days. Typing (Rp) opens the full Recent Projects screen, where a number or DSO name selection jumps directly to DSO Status with pipeline shortcuts (St/Gx/Pr/Cs/Gp). The Pipeline shortcut (Pl) opens a consolidated one-screen dispatcher that presents the next logical step for a selected version.

7.1.6 Tool Steps Submenu (Ts)

Run exactly one pipeline stage on a specific version:

Code	Step	I/O	Description
Gx	GraXpert	SO → GX	Gradient and background extraction
Si	Siril	GX/SO → PO	Stretch and SPCC spectrophotometric color calibration
Sl	Starless	PO → SL/SK/SF	StarNet2 split, sharpen starless layer, recombine
Cs	Cosmos	PO/SF → CI → CO	Cosmos stage and export ingest
Gp	Gigapixel	PO/SF/CO → GI → GO	Upscaling and album integration
Sd	Siril DB	—	Check and download offline catalog files for Siril
Dr	Drive Reg.	—	Browse and edit the drive registry

7.2 Data Acquisition Scripts

These scripts transfer raw imaging data from the Vaonis scopes (over WiFi) or from shuttle drives attached to other Macs in the cluster. All pull scripts are idempotent — re-running skips already-downloaded files via lftp resume or rsync checksums.

astro_allpull

Pulls the latest observation from every powered-on Vespera scope over WiFi, one scope at a time. Switches to each scope's WiFi SSID, runs lftp mirror --continue to download the FITS, TIFF, and processed JPG (FITS included by default; pass --no-fits to skip them), then switches to the next. Displays an estimated download time based on a rolling 5-session average.

```
astro_allpull [--no-fits] [--dest <drive>] [--scope <key>]
```

Argument / Flag	Description
--no-fits	Skip the raw FITS mirror — download only the newest TIFF and JPEG. FITS are included by default.
--dest <drive>	Destination drive short name from the registry (default: D_MAIN).
--scope <key>	Pull from a single scope only (vp1, vp2, v2a, v2b).

```
astro_allpull
astro_allpull --dest Mule
astro_allpull --scope vp1
```

vespera_pull

Pulls data from a single Vespera scope. Scope key is one of: vp1, vp2, v2a, v2b, v1. Used when only one scope was powered, or when astro_allpull is not needed.

```
vespera_pull <scope_key> [--no-fits] [--dest <drive>]
vespera_pull vp1
vespera_pull v2a --dest Mule
```

mule_sweep

Pulls all data from the Mule shuttle drive (4TB-Astro-Mule, normally on the M5 Air) to D_MAIN via rsync over SSH. Run from the Pro. Requires passwordless SSH to Astro-M5Air.local and 'Allow full disk access for remote users' enabled on the M5 Air.

```
mule_sweep [--dry-run]
mule_sweep --dry-run
mule_sweep
```

Note: *The Mule uses a flat RawTelescope layout (no Astronomy/PhotosData prefix).*

camel_sweep

Same as mule_sweep but targets the Camel drive (T7-Astro-Camel) on the M3 Air.

```
camel_sweep [--dry-run]
```

sweep_all

Sweeps both Mule and Camel in one pass. Available via astromenu Dl → As.

```
sweep_all [--dry-run]
```

disk_pull

Ingest data from a locally connected drive or folder that does not use the Mule/Camel rsync path. Lists what would be copied in dry-run mode; use --execute to actually copy. Available via astromenu Fm → Dp.

```
disk_pull <source_path> --scope <key> [--execute] [--dry-run]
disk_pull /Volumes/OldDrive/RawTelescope --scope vp1 --dry-run
```

7.3 Ingestion Scripts

Ingestion is the step that moves raw frames from RawTelescope into the pipeline directory structure. These scripts create a version directory (DSO-001) under StackingInput/, copy or hardlink raw frames into it with a standardized naming scheme, and record provenance. After ingestion, a version is ready for stacking.

astrostart

Creates or reuses the standard pipeline directory structure for a DSO. Always assigns version DSO-001. Creates StackingInput/<DSO-001>/, AstroAlbum/<DSO>/, and all Workflow stage folders if needed. Prints the version string on stdout so callers can capture it: VERSION=$(astrostart M42).

```
astrostart <DSO>
astrostart M42
astrostart NGC2174
VERSION=$(astrostart Orion)
```

Note: *Combined versions (DSO-C##) are unaffected by the always-001 rule and continue to use their own C## serial.*

CollectFits

Copies raw FITS frames from RawTelescope into StackingInput/<DSO-###>/. Uses MD5 content hashing to prevent duplicate frames. As of June 2026, CollectFits runs regen_sources_json automatically whenever the version gained frames, so the .sources.json provenance sidecar is never left stale (the call is guarded so a regen failure cannot abort CollectFits, and falls back to printing the manual command). Run regen_sources_json by hand only to force a rebuild.

```
CollectFits <DSO-###> [--fresh] [--scope <key>]
CollectFits M42-001 --fresh
CollectFits NGC2174-001
```

CollectTiffs

Copies pre-stacked TIFF frames from RawTelescope into StackingInput/<DSO-###>/. Used when the imaging target was captured in TIFF mode or when the scope did not produce FITS output.

```
CollectTiffs <DSO-###> [--fresh]
CollectTiffs M45-001 --fresh
```

regen_sources_json

Builds or rebuilds the .sources.json sidecar for a StackingInput version. Records the provenance of every frame (source observation folder, scope, session date). CollectFits now runs this automatically; call it by hand only to force a rebuild.

```
regen_sources_json <DSO-###>
regen_sources_json M42-001
```

astrocombine

Pools raw frames from two or more existing StackingInput versions into a new combined version named DSO-C## (e.g. M3-C01). Validates that all inputs are the

same DSO, deduplicates frames by filename and MD5 hash, and creates a fresh .sources.json.

```
astrocombine <version1> <version2> [<version3> ...]
astrocombine M3-001 M3-002 M3-003
astrocombine NGC891-001 NGC891-002
```

Note: Via astromenu: St (Stack) or Ap (auto-process) both accept a comma-separated row list (e.g. '1,3') to trigger astrocombine automatically.

stage_backlog

Batch ingest. For every DSO that still has frames not in StackingInput, mints a fresh DSO-### version and collects FITS + TIFF into it (runs FindUnprocessed, CollectFits, CollectTiffs, regen_sources_json). Dry-run by default; removes any version where no new frames landed.

```
stage_backlog [<DSO>] [--execute]
stage_backlog            # dry run: show the plan
stage_backlog --execute  # stage the whole backlog
```

Note: Discovery is TIFF-based (FindUnprocessed); a FITS-only DSO with no outstanding TIFFs needs a manual CollectFits. See astrohelp stage-backlog.

7.4 Stacking Scripts

Stacking mathematically combines many individual frames into a single calibrated master image. Siril handles registration (frame alignment), rejection (outlier pixel removal), and integration (sigma-clipping or winsorized stacking). Input: StackingInput/<version>/. Output: StackingOutput/<version>_SO##[_fits].fit.

sirildispatch

Interactive stacking dispatcher. Detects which frame formats are present (TIFFs, FITS, both), presents a format chooser (T/F/B/S), and calls the appropriate stacker. B (both) writes two independent stacks with _fits flavor tagging. S (skip) exits without stacking.

```
sirildispatch <DSO-###> [--default T|F|B|S]
sirildispatch M42-001
sirildispatch NGC2174-001 --default F
```

stackdso_fits_v1

Stacks FITS frames using Siril with debayer + two-pass registration. This is the current standard path for VP-1, VP-2, and Vespera 2 FITS data. Output: StackingOutput/<version>_SO##_fits.fit.

```
stackdso_fits_v1 <DSO-###> [--scope <key>]
stackdso_fits_v1 M42-001
```

Note: Typical registration rate: 85–95% on VP-1/VP-2, 80–92% on Vespera 2.

stackdso

Stacks pre-processed TIFF frames. Used for Vespera 1 legacy data or when TIFF mode was used. No debayer — scope's on-board processing is assumed. Output: StackingOutput/<version>_SO##.fit.

```
stackdso <DSO-###>
stackdso M45-001
```

score_frames

Quality-scores individual frames on SNR and star metrics, writing a scored sidecar that sirildispatch uses to exclude poor frames. Run before stacking when data quality is variable.

```
score_frames <DSO-###> [--threshold <N>]
score_frames M42-001
```

astro_remote_stack

Offloads stacking to another Mac in the cluster (e.g. the M5 Air) by copying the StackingInput version via rsync and running sirildispatch on the remote host. The output stack is pulled back to D_MAIN when complete.

```
astro_remote_stack <DSO-###> <host> [--format T|F|B]
astro_remote_stack M3-C01 Astro-M5Air.local --format F
```

astro_dual_stack

Runs two Siril stacking jobs simultaneously — one on the Pro and one on the M5 Air — for the same version or two different versions, using all available CPU across both machines. Available via astromenu Du.

```
astro_dual_stack
```

stack_backlog

Batch per-DSO stacking. For each DSO, stacks the latest version FITS-first (a real FITS dataset of at least --fits-min frames, default 8) or TIFF otherwise, keeping FITS and TIFF stacks separate. Splits work across the Pro (local) and M5 Air (remote); skips versions already in StackingOutput unless --force. Dry-run by default.

```
stack_backlog [<DSO>] [--execute] [--local] [--force] [--
combine-tiff] [--fits-min N]
stack_backlog            # dry run: per-DSO plan + Pro/Air split
stack_backlog --execute  # run the backlog
```

Note: Largest stacks go to the Pro. Scratch (ASTRO_STACK_DIR) is auto-resolved to the node scratch drive, never the data drive. See astrohelp stack-backlog.

Runs as a launchd daemon on AstroDen (com.astro.hub-combine, 30-minute interval). Polls astro_combine_pending for DSOs with uncombined versions whose frames are already on ASTROHUB, and fires astrocombine for each candidate. Can also be invoked manually. Logs to ~/Library/Logs/astro/hub_combine.log. (Currently inactive: AstroDen has no active role at present.)

7.5 Processing Scripts

Processing transforms a raw Siril stack into a calibrated, stretched color image that can be reviewed directly, sent to optional Starless Processing or Cosmos, prepared for Gigapixel upscaling, or moved to the album. The standard core path is: graxpert_clean (background removal) → sirilprocess (SPCC color calibration + auto-stretch). Starless Processing, Cosmos, and Gigapixel are optional target-dependent stages.

graxpert_clean

Runs GraXpert AI background extraction on the most-recent SO## stack file for the given version. Produces GraXpertOutput/<version>_GX##[_fits].fit. sirilprocess auto-detects this _GX file and skips its own subsky, so --no-subsky is no longer required (the flag still works for explicit control). With --denoise, GraXpert runs a denoise pass on the linear data after background extraction; sirilprocess reads the provenance sidecar and skips its own denoise to avoid processing twice. As of GraXpert 3.1 the --denoise-strength flag (0.0–1.0) is available via the CLI (it required the GUI in 3.0.x); GraXpert 3.1 also adds AI deconvolution, accessible from the GUI or via Siril's GraXpert-AI.py script — not through graxpert_clean directly. For CLI-scriptable deconvolution, use sirilprocess --deconvolve (Siril Richardson-Lucy, no GraXpert required).

```
graxpert_clean <DSO-###> [--stack-flavor fits] [--smoothing <N>]
[--correction <mode>] [--denoise] [--denoise-strength <N>]
```

Argument / Flag	Description
--stack-flavor fits	Required when the stack came from FITS frames (_fits suffix).
--smoothing <N>	GraXpert smoothing level (default 0.5; try 0.3 for persistent gradients).
--correction <mode>	Correction mode: subtraction (default) or division.
--denoise	Run a GraXpert denoise pass on the linear data after background extraction. sirilprocess then skips its own denoise (read from the _GX provenance sidecar) so the data isn't denoised twice.
--denoise-strength <N>	Denoise strength 0.0–1.0 (default 0.5). Only meaningful with --denoise.

```
graxpert_clean M42-001 --stack-flavor fits
graxpert_clean M3-C01 --stack-flavor fits --smoothing 0.3
```

Note: *If GraXpert over-corrects (dark patches, nebula structure subtracted), increase smoothing to 0.7.*

sirilprocess

Runs Siril spectrophotometric color calibration (SPCC) and auto-stretch on a stack or GraXpert output. Before processing begins, checks Vizier connectivity; if unreachable, pauses with R (retry) / Y (proceed without color cal) / any key (abort). Output: ProcessedOutput/<version>_PO##[_fits].tif. As of June 2026 it auto-guards two steps:

when the input filename carries the _GX tag it skips its own subsky (GraXpert already removed the gradient), and when that _GX file's provenance shows GraXpert already denoised, it skips its own denoise. The --no-subsky and --no-denoise flags force those behaviors manually; --no-sharpen omits the unsharp pass (used by the Starless workflow so sharpening happens once, on the starless layer). --deconvolve runs Siril's own built-in deconvolution (makepsf + Richardson-Lucy, rl -tv) on the linear image just before the stretch — no AI and no GraXpert required; tune it with --deconv-iters <N> (default 10) and --deconv-psf stars|blind.

```
sirilprocess <DSO-###> <scope_key> [--stretch-level <N>] [--no-subsky] [--no-denoise] [--no-sharpen] [--deconvolve] [--deconv-iters <N>] [--deconv-psf stars|blind] [--input-file <path>]
```

Argument / Flag	Description
<scope_key>	Scope key: vp (both VP units), v2 (both Vespera 2 units), v1. Pull scripts still use per-unit keys vp1/vp2/v2a/v2b; stacking scripts use consolidated keys.
--stretch-level <N>	Stretch aggressiveness 1–10 (see stretch table). Default: 5.
--no-subsky	Skip Siril background subtraction. Auto-enabled for _GX (GraXpert) inputs; pass it explicitly to force the behavior on other inputs.
--deconvolve	Apply Siril's built-in Richardson-Lucy deconvolution to the linear image, just before the stretch (no AI, no GraXpert). Off by default.
--deconv-iters <N>	Richardson-Lucy iterations (default 10). More = sharper, but risks dark rings around bright stars and noise amplification — lower it if rings appear.
--deconv-psf stars|blind	PSF estimation method. "stars" (default) models the PSF from detected stars — best for star-bearing DSO frames; "blind" estimates it from the image (sparse-star fields, lunar).
--input-file <path>	Override default input file (use when starting from GX## output).
--no-denoise	Omit Siril's denoise step. Auto-enabled when the GraXpert input was already denoised; pass it to skip denoising for any other reason.
--no-sharpen	Omit Siril's unsharp pass. Use when the Starless step will run next, so sharpening happens once on the starless layer. The Process (Pr) menu prompts for this.

```
sirilprocess M42-001 vp1 --stretch-level 3
sirilprocess M3-C01 vp1 --input-file \
  $ASTRO_BASE/PhotosData/Workflow/GraXpertOutput/M3-C01_GX01_fits.fit \
```

```
  --no-subsky --stretch-level 5
```

siril_starless

Optional. Splits a Siril-processed (PO##) image into a starless layer and a star layer using the StarNet2 CLI, sharpens the starless layer only, then screen-recombines the stars on top. Writes three files to StarlessOutput/: the sharpened starless layer (SL##), the star layer (SK##), and the recombined final (SF##), each with a provenance sidecar. Drives the StarNet2 binary directly (auto-detected at ~/StarNet/starnet2, or set with --starnet-bin) — no StarNet2 path is configured in Siril. Best run on a PO## made with sirilprocess --no-sharpen, so sharpening happens once, here.

```
siril_starless <DSO-###> [--sharpen "R A"] [--input-file <path>]
[--no-recombine] [--stride N] [--upsample] [--starnet-bin
<path>]
siril_starless M42-001 --sharpen "1.0 0.7"
```

Pass "0 0" to --sharpen to split and recombine without sharpening; --no-recombine keeps the SL and SK layers separate (recombine manually in GIMP). Via astromenu: Ts (Tool Steps) → Sl. See Section 3.5. To install StarNet2 on a machine — or replicate it across the Macs — run setup_starnet: it detects the chip (arm64/x64), installs to ~/StarNet, unlocks and verifies it. setup_starnet --stash publishes the working build into Tools/vendor/starnet/ via Dropbox, so each other same-chip Mac installs it with a plain setup_starnet.

PrepareCosmos

Stages the best available processed file as CosmosInput/<version>_CI##[_fits].tif, scaled to fit within Cosmos's 500 MB input size limit. The --from-po flag selects the PO## file; --from-starless selects the recombined Starless (SF##) file from StarlessOutput/.

```
PrepareCosmos <DSO-###> [--from-po] [--stack-flavor fits]
PrepareCosmos M42-001 --from-po --stack-flavor fits
```

RunCosmos

Opens the Cosmos Darkroom application with the CI## file for the given version. After tone-mapping and exporting a 16-bit TIFF to ~/Downloads/, run cosmos_export to move the result into the pipeline.

```
RunCosmos <DSO-###>
```

Important: In Cosmos, always choose 'ALREADY STRETCHED' — sirilprocess has already applied autostretch.

cosmos_export

Moves the Cosmos output TIFF from ~/Downloads/ into CosmosOutput/<version>_CO##.tif. Detects the newest TIFF in Downloads/ that matches the version name pattern.

```
cosmos_export <DSO-###>
cosmos_export M42-001
```

Note: *Known bug: cosmos_export drops the observation suffix from the version string in some cases. If PrepareGiga cannot find CO##, check CosmosOutput/ for a truncated name and rename manually.*

PrepareGiga

Stages the best available processed file as GigaInput/<version>_GI##.tif, selecting in priority order: CO## (Cosmos output), PO## (Siril processed), GX## (GraXpert). The --source flag overrides the auto-selection.

```
PrepareGiga <DSO-###> [--source po|co|gx]
PrepareGiga M42-001
PrepareGiga M3-C01 --source po
```

RunGiga

Opens Topaz Gigapixel with the GI## file for the given version. After upscaling and exporting to ~/Downloads/, run giga_export or MoveToAlbum to complete.

```
RunGiga <DSO-###>
```

Note: *Recommended Gigapixel settings: Model=Low Resolution, Scale=2x, Sharpen=50-60, Denoise=8, Gamma=ON.*

giga_export

Moves the Gigapixel output from ~/Downloads/ into GigaOutput/<version>_GO##.tif.

```
giga_export <DSO-###>
```

MoveToAlbum

Copies the best available selected image (GO##, then CO##, then PO##) into AstroAlbum/<DSO>/ with a standardized filename, then automatically imports it into the “AstroAlbum” album in Apple Photos (skip check duplicates — safe to re-run). Requires one-time Automation permission: System Settings → Privacy & Security → Automation → Terminal → Photos. This is the final step of every pipeline run.

```
MoveToAlbum <DSO-###>
MoveToAlbum M42-001
PrepareGiga M42-001 && RunGiga M42-001 && MoveToAlbum M42-001
```

astro_autoprocess

Chains the full pipeline end-to-end: score_frames → sirildispatch → graxpert_clean → sirilprocess → PrepareCosmos prompt → PrepareGiga → RunGiga → MoveToAlbum. Use --collect to run CollectFits/CollectTiffs as the first step.

```
astro_autoprocess <DSO-###> <scope_key> [--format fits|tiff] [--
stretch-level <N>] [--graxpert] [--collect]
```

Argument / Flag	Description
--format fits|tiff	Frame format to stack (default: auto-detect).
--stretch-level <N>	Stretch level 1–10 (default: 5).
--graxpert	Include GraXpert background extraction step (adds --no-

Argument / Flag	Description
	subsky automatically).
--collect	Run CollectFits (FITS) or CollectTiffs (TIFF) as the first step.
--blurx	Run BlurXTerminator (RC Astro) AI deconvolution on the linear stack (-> RCXOutput/ _BX##). Requires the BXT plugin.
--noisex	Run NoiseXTerminator (RC Astro) AI denoise on the linear stack (-> _NX##). Requires the NXT plugin.
--starx	Run StarXTerminator (RC Astro) AI star removal, producing a starless linear image (-> _SX##). Requires the SXT plugin.

```
astro_autoprocess M3-C01 vp1 --format fits --stretch-level 6
astro_autoprocess M42-001 vp1 --stretch-level 3
astro_autoprocess M101-001 vp1 --stretch-level 7 --graxpert
astro_autoprocess NGC2174-001 v2b --collect --format fits
```

7.6 Archive & Backup Scripts

These scripts manage the optional long-term archive and backup of both raw and processed data. In the current installation, the archive is the AstroArchive drive on the Mac mini.

```
archive_stacked --list [<DSO>]
```

astro_backup

One-time or periodic rsync backup of the full PhotosData tree from D_MAIN to AstroArchive on the Mini. Uses --checksum for reliable incremental transfers. Safe to re-run.

```
astro_backup [--dry-run] [--status]
astro_backup --dry-run
astro_backup
```

astro_workflow_backup

Rsync backup of the Workflow/ directory (excluding large reconstructable staging folders StackingInput, CosmosInput, GigaInput) from D_MAIN to the Mini's AstroArchive. Uses rsync --checksum — idempotent, safe to re-run. Bandwidth-limited at 4000 kB/s to protect the spinning HDD; Mini caffeinate uses -s -i -m; SSH timeout 300 s.

```
astro_workflow_backup [--dry-run] [--status]
```

archive_fits

Moves raw FITS frames for a DSO from D_MAIN StackingInput to AstroArchive, freeing space on D_MAIN while preserving source data. Safe to run after stacking is confirmed complete.

```
archive_fits <DSO> [--dry-run]
archive_fits M42 --dry-run
archive_fits M42
```

archive_processed

Copies (does not delete) processed outputs from Workflow to a preserved archive on D_MAIN: ProcessedOutput → PhotosData/ProcessedArchive/ProcessedOutput/, GigaOutput → ProcessedArchive/GigaOutput/, CosmosOutput → ProcessedArchive/CosmosOutput/. Uses rsync --checksum — fully idempotent. Note: Mini daemons only mirror raw FITS to AstroArchive; processed outputs are not covered. Access via astromenu Ut → Fm → Ap (Archive Processed).

```
archive_processed [--dry-run] [--list] [--no-confirm] [<DSO>]
archive_processed --dry-run
archive_processed M42                # archive outputs for one
DSO only
```

archive_giga

Moves Gigapixel output files from GigaOutput/ to AstroArchive/GigaArchive/, keeping D_MAIN free for active work. Available via astromenu Fm → Ag.

```
archive_giga [--dry-run]
```

backup_scripts

Creates a timestamped copy of all active scripts in $ASTRO_BIN to a backup folder. Useful before making significant changes to the pipeline.

fix_mini_sshd

Patches the Mini's sshd_config for long-running rsync sessions: sets ClientAliveInterval 60 and ClientAliveCountMax 240 (4-hour tolerance). Backs up the existing config with a timestamp, patches via sed, and restarts sshd via launchctl. Run once after initial Mini setup, or any time rsync sessions drop with "Connection reset by peer."

```
fix_mini_sshd [--dry-run] [--host <hostname>]
fix_mini_sshd --dry-run
fix_mini_sshd
backup_scripts
```

7.7 Drive & Index Management

dso_checkout

Copies all data tiers for a named DSO from D_MAIN to a shuttle drive (Camel preferred, Mule fallback — auto-detected locally or via SSH on M3/M5 Air when not physically connected). Tiers: RawTelescope obs folders, FitsArchive obs folders, AstroStack work dirs, AstroAlbum TIFFs. Writes .checkout_manifest.json on the drive and registers a lock in Tools/data/dso_checkout_state.json. While a DSO is checked out, astromenu

shows a soft conflict warning before any write to D_MAIN for that target. Use --m3 or --m5 to constrain the remote SSH probe to a specific Air.

```
dso_checkout <DSO> [--dry-run] [--force] [--drive <path>] [--m3]
[--m5]
dso_checkout --list                # show active checkouts (DSO,
drive, host, FITS count, TIFF)
dso_checkout M101                  # check out to Camel or Mule
(auto-detect)
dso_checkout M101 --m3             # constrain remote probe to M3
Air only
```

Note: Checkout layout on drive: <drive>/Checkout/<DSO>/ mirrors D_MAIN layout. Set ASTRO_BASE=<drive>/Checkout/<DSO> on the Air to use existing pipeline tools. Via astromenu Co (Check Out DSO).

dso_checkin

Syncs changes from the checkout drive back to D_MAIN and clears the lock. Auto-detects drive from the state file. Uses rsync --update (newer file wins) for each tier. If both D_MAIN and Air produced conflicting stacks for the same version, rename one (e.g. M101-C06) before running dso_checkin to keep both as candidates.

```
dso_checkin <DSO> [--dry-run] [--no-clear] [--list]
dso_checkin M101 --dry-run
dso_checkin M101
```

Note: Via astromenu Ci (Check In DSO).

astro_drive

Manages drive_registry.json — the central database of all known drives. The registry lets pipeline scripts refer to drives by short name instead of /Volumes/ paths, and tells astro_netwatch which SMB shares to watch for.

```
astro_drive <subcommand> [args]
```

Subcommand	Description
list	Show all registered drives with mount status and free space.
register <name> <path>	Add a new drive. Creates standard RawTelescope folders if mounted.
unregister <name>	Remove a drive from the registry.
init <name>	Create standard folder structure on a registered but unmounted drive.
check <name>	Exit 0 if the named drive is currently mounted.

```
astro_drive list
astro_drive register Ox /Volumes/Ox-Astro
```

```
astro_drive init Ox
astro_drive unregister OldDrive
```

astro_index

Manages the shared ingest index (Dropbox/Astronomy/data/ingest_index.json). All machines share this file; concurrent writes are protected by fcntl file locking. Tracks every known scope session through its status lifecycle: pending → archived → stack_queued → stacked.

```
astro_index <subcommand> [args]
```

Subcommand	Description
stats	Show session counts by status.
list [--status <s>] [--dso <name>]	List sessions, optionally filtered.
pending	Shorthand for --status pending.
discover <raw_tel_path> --source <drive>	Scan a RawTelescope path and register new sessions.
status <key>	Show full entry for one session.
update <key> --field <f> --value <v>	Update a single field on a session.

```
astro_index stats
astro_index pending
astro_index list --dso M42
astro_index discover
/Volumes/AstroMaster/AstroData/PhotosData/RawTelescope --source
D_MAIN
```

newdrive

Migrates the data tree (PhotosData) onto a freshly-formatted drive using ditto, which preserves hardlinks. This matters because StackingInput is ~969 GB of hardlinks into FitsArchive; a hardlink-unaware copy explodes every pooled version into full copies. Verifies that link counts match; dry-run by default.

```
newdrive /Volumes/<NewName> [--go] [--all]
newdrive /Volumes/AstroMaster          # dry run
newdrive /Volumes/AstroMaster --go     # run the hardlink-
preserving copy
```

Note: Format the new drive APFS. After copying, run astro_drive register D_MAIN /Volumes/<NewName>; astrorc then auto-detects it and sets ASTRO_BASE / ASTRO_STACK_DIR. See astrohelp newdrive.

7.8 Status & Utility Scripts

astrostatus / astat

Prints a full pipeline status report: for each version in StackingInput, shows which stages have completed (SO##, GX##, PO##, CI##, CO##, GI##, GO##) and whether the selected image is in AstroAlbum.

Note: SwiftBar must be installed on each Mac; the plug-in itself syncs automatically via Dropbox. Run install_swiftbar.sh on each node after first installing Dropbox.

```
astrostatus [<DSO>]
astrostatus
astrostatus M42
astat M3-C01
```

sourceinfo

Shows full provenance for a pipeline version: source observations, scope key, frame counts, color calibration status, and all output files at each stage.

```
sourceinfo <DSO-###>
sourceinfo M42-001
sourceinfo M3-C01
```

find_target

Scans every pipeline stage directory and reports where the given DSO appears — useful when it is unclear which stages have output.

```
find_target <DSO>
find_target M42
find_target NGC891
```

dso_log / astro_log.zsh

Views the pipeline operations log. Each pipeline run logs a structured entry (DSO, operation type, scope, timestamp, result). Use --op to filter by operation type.

```
dso_log [--op <type>] [--dso <name>]
dso_log
dso_log --op needs-recolor
dso_log --dso M42
```

dsoinfo

Shows catalog information for a target: common names, type, distance, size, and current visibility from Denver / South Park. Alias resolution handles common names (Orion → M42).

```
dsoinfo <DSO>
dsoinfo M42
dsoinfo Orion
dsoinfo NGC1499
```

astro_setup_mini

One-time setup script for the Mac mini. Creates folder layouts on AstroWork and AstroArchive, registers the mini drives, initializes the ingest index, and installs the LaunchAgent plists for both daemons. Idempotent — safe to re-run.

```
astro_setup_mini [--dry-run]
```

8 Daemons

Several background daemons run on the Mac minis as launchd LaunchAgents. They start automatically at login, restart if they crash, and run indefinitely without user intervention. Together they form the archive and catalog subsystem: connecting the drive after a session prompts the mini to mount it within about 90 seconds and begin archiving the new data. The two daemons divide the work — netwatch mirrors the raw FITS/TIFF frames into the archive (the data-safety copy), while astro_catalog maintains a browsable JPG catalog of each session. Core APAS collection and processing can still run without this subsystem.

8.1 astro_netwatch

astro_netwatch is the network discovery and archive daemon. It continuously monitors the local network for registered SMB shares and, when a registered data drive appears, mounts it, registers new scope sessions in the ingest index, and runs two background mirror operations: raw FITS/TIFF frames are mirrored into AstroArchive/RawTelescope/ (the primary raw-data backup), and stacked FITS outputs (_SO##.fit and provenance sidecars) are mirrored into AstroArchive/StackingOutput/ (protecting the expensive stacking work). It runs on the Mac mini.

When a new drive appears (e.g. D_MAIN is plugged into the Pro and shared via SMB), netwatch detects it within one scan cycle (default 60 seconds) and mounts it on the host pinned for that drive in the registry (resolved from the drive's node entry). It then runs astro_index discover on the RawTelescope path to find new sessions, marks them pending, launches a background rsync that mirrors the raw frames into AstroArchive/RawTelescope/, and writes an ingest_trigger file to wake astro_catalog immediately.

How netwatch discovers drives:

- Browses mDNS (_smb._tcp) for 8 seconds to find all SMB hosts on the network.
- For each drive with smb_share_name in the registry, mounts it on its pinned host — resolved from the drive's node entry in the registry (e.g. D_MAIN to MacBook-Pro.local) — via osascript. The pinned host is trusted directly rather than enumerating its shares first, since share enumeration is unreliable on authenticated (password-protected) hosts.
- Pre-checks port 445 (2 s timeout) before attempting mount — avoids 30 s hangs.
- Resolves Bonjour hostnames to IPv4 addresses via socket.getaddrinfo() to bypass .local DNS quirks.
- Remembers the last successful host per share in netwatch_state.json for fast subsequent scans.

Command	Description
`astro_netwatch`	Run one scan cycle (interactive, useful for testing)
`astro_netwatch --watch`	Continuous mode (same as daemon)

Command	Description
`astro_netwatch --dry-run`	Scan and report without mounting
`astro_netwatch --install`	Install as launchd LaunchAgent on the mini
`astro_netwatch --uninstall`	Remove LaunchAgent
`astro_netwatch --status`	Show daemon state (running/stopped, last scan time)
`astro_netwatch --interval 120`	Override scan interval (seconds)

File	Purpose
~/Library/Logs/astro_netwatch.log	All scan activity, mount attempts, errors
~/Library/Application Support/Astro/netwatch_state.json	Last-known host per share (speeds up subsequent scans)
~/Library/LaunchAgents/com.astro.netwatch.plist	LaunchAgent plist (created by --install)

8.2 astro_catalog

astro_catalog is the catalog daemon. It reads pending sessions from the ingest index and copies each session's newest processed JPG into a clean, browsable layout under AstroArchive/Sessions/ and AstroArchive/DSO/. It does not handle the raw frames — that is netwatch's responsibility (see 8.1). It runs on the Mac mini. When the source drive (e.g. D_MAIN) is not already mounted on the Mini, astro_catalog now attempts an on-demand SMB mount via osascript (using macOS Keychain credentials) before skipping a session; sessions are only skipped if the source host is genuinely unreachable. This eliminates the previous failure mode where sessions were silently skipped because D_MAIN happened not to be mounted when the daemon ran.

For each pending session, ingest searches the observation folder for the newest processed JPG in the standard firmware-dependent paths (02-images-initial/, 01-images-initial/, images-initial/, or root). It copies the JPG to AstroArchive/Sessions/YYYY-MM-DD_<DSO>_<ScopeShort>/ alongside a provenance.json, then creates a hardlink in AstroArchive/DSO/<TARGET>/ for zero-byte duplicate cost. Sessions with only raw .fits frames (no stacked JPG) are marked fits_only and skipped by the catalog — their raw frames are still mirrored to the archive by netwatch, so no data is lost.

Command	Description
`astro_catalog`	Process all pending sessions once
`astro_catalog --`	Continuous mode (30 s poll + trigger file)

Command	Description
`watch`	
`astro_catalog --dry-run`	Preview without writing
`astro_catalog --verbose`	Debug output — shows every decision
`astro_catalog --session <key>`	Process one specific session
`astro_catalog --install`	Install as launchd LaunchAgent on the mini
`astro_catalog --uninstall`	Remove LaunchAgent
`astro_catalog --status`	Show daemon state

8.3 Installing & Managing Daemons

As of June 2026, the Daemon Status (Da), Watch Daemon Log (Wd), and Restart Daemons (Rd) functions in astromenu Utilities are available from any node, not just the Mini. When run from a remote node, each function SSHs to Astro-Mini.local using BatchMode key authentication. The Watch Daemon Log option probes the Mini for available log files and streams the selected log via ssh -t tail -f. SSH key setup is required: run ssh-copy-id from each node to Astro-Mini.local using the id_ed25519_astro key.

Task	Command
Install netwatch	`astro_netwatch --install`
Install ingest	`astro_catalog --install`
Check status	`astro_netwatch --status && astro_catalog --status`
View netwatch log	tail -50 ~/Library/Logs/astro_netwatch.log
View ingest log	tail -50 ~/Library/Logs/astro_catalog.log
Restart netwatch	launchctl unload ~/Library/LaunchAgents/com.astro.netwatch.plist && launchctl load ~/Library/LaunchAgents/com.astro.netwatch.plist
Restart ingest	launchctl unload ~/Library/LaunchAgents/com.astro.catalog.plist && launchctl load ~/Library/LaunchAgents/com.astro.catalog.plist

Task	Command
Daemon	**Default Interval**
`astro_netwatch`	60 seconds (mDNS scan)
`astro_catalog`	30 seconds (pending poll)

8.4 stack_sync (Automatic Stacking Pipeline)

stack_sync is the automatic batch pipeline, running as a launchd LaunchAgent (com.astro.stack_sync) on the M4 Mini (C_STACK). Each cycle (default 5 minutes) it: (1) checks that AstroMaster is reachable on the local LAN -- it skips when the host resolves over Tailscale, so cabin bandwidth is never used; (2) rsyncs RawTelescope from AstroMaster to AstroRAID; (3) waits for each observation folder's FITS count to hold steady before ingesting; (4) runs CollectFits to stage a new StackingInput version; (5) launches astro_autoprocess <version> <scope> --graxpert in the background; and (6) when the pipeline queue drains, builds a slide-sorter review PDF with BatchResultsPDF (into Workflow/ReviewPDF/).
Parallel per-machine pipelines. Each ingested version runs the FULL pipeline (stack, GraXpert, BXT, NXT, STX, Siril stretch) on ONE machine, and the Mini, the Pro, and the M5 Air each process a different version at the same time — up to three in parallel, with no two Siril instances ever sharing a box. Every job launches through astro_autoprocess; a remote worker is selected by adding --remote HOST, which delegates the whole run to astro_remote_autoprocess on that host (push StackingInput to the host, run the chain there, pull ProcessedOutput PO## back to AstroRAID). Workers each cycle are “local” (the Mini, always) plus every REMOTE_WORKER_NODES peer — C_MAIN (MacBook-Pro.local) and C_NODE1 (Astro-M5Air.local) — that is reachable on the LAN over SSH (port 22, non-Tailscale, the same policy as the sync check). Dispatch fills every free machine, one pipeline per box; when a job finishes, the next queued version starts on whatever machine just freed up. A remote pipeline that fails (peer drops, no PO## returns) is re-queued pinned to the Mini, up to MAX_PIPELINE_ATTEMPTS (3), so work never stalls on a flaky peer; --skip-stack reprocess jobs run local-only. Each peer must have Siril, GraXpert, and a licensed rc-astro installed. stack_sync --status shows each worker as reachable, busy, or idle.
Commands: stack_sync --status (queue + running pipelines), --install / --uninstall (LaunchAgent), --watch (foreground), --reprocess (re-queue completed versions with --skip-stack), --dry-run, --no-ingest, --no-pipeline, --verbose. Logs: ~/Library/Logs/stack_sync.log and per-version stack_sync_pipeline_<VERSION>.log. State: ~/Library/Application Support/Astro/stack_sync_state.json. A stable, un-ingested folder is retried each cycle up to MAX_INGEST_ATTEMPTS (5), then it gives up loudly. The daemon holds its code in memory, so after editing the stack_sync script reload it once: launchctl kickstart -k gui/$(id -u)/com.astro.stack_sync. Normal reboots and crashes are handled automatically (RunAtLoad + KeepAlive).

8.5 astro_log_snapshot (Daily Log Snapshot)

astro_log_snapshot is a LaunchAgent (com.astro.log_snapshot) that writes a daily one-way snapshot of the stack_sync logs (tails) plus a copy of stack_sync_state.json into Dropbox/Astronomy/Tools/logs/mini_snapshots/, host-tagged and pruned after 14 days. It lets the Mini's logs be read from any machine without putting the live, constantly-appended log -- or the daemon state file -- into Dropbox, which would cause sync churn and conflicted-copy corruption. Install: astro_log_snapshot --install [--time HH:MM] (default 08:00). Run a one-shot snapshot with astro_log_snapshot.

8.6 BatchResultsPDF

BatchResultsPDF builds a slide-sorter PDF -- one final PO## image per DSO version, one per page -- into Workflow/ReviewPDF/. stack_sync runs it automatically when the pipeline queue drains; it can also be run by hand: BatchResultsPDF [versions...] [--open] [--label "Night 2"]. It needs a Python with Pillow/numpy/tifffile on PATH.

9 Data Flow & Automation

This section traces data from the moment an observation ends on the scope through to the selected image in AstroAlbum. There are two broad paths: the automatic archive path (astro_netwatch + astro_catalog, no user action required) and the interactive pipeline path (astromenu or command-line scripts, requires user initiation).

9.1 Optional Archive Path (Scope → AstroArchive)

Step-by-step, from scope to archive:

- Shoot with Vaonis scope — images stored in app on scope WiFi.
- Connect scope to Mac (or use astro_allpull) — Vaonis app or lftp syncs to D_MAIN RawTelescope/<scope>/<obs_folder>/.
- D_MAIN appears on the network (shared via SMB from the Pro).
- astro_netwatch detects D_MAIN (the AstroMaster share) via mDNS within 60 seconds.
- netwatch mounts D_MAIN on its registry-pinned host via SMB and runs astro_index discover — new sessions marked pending.
- netwatch launches a background rsync that mirrors the raw FITS/TIFF frames into AstroArchive/RawTelescope/ (the data-safety copy), then writes an ingest_trigger file.
- astro_catalog detects trigger (or wakes on 30 s poll) and processes pending sessions.
- For each session, astro_catalog copies the newest JPG to AstroArchive/Sessions/ and hardlinks it into AstroArchive/DSO/ (the browse catalog).
- Session marked archived in ingest_index.json.

9.2 Interactive Pipeline Path (D_MAIN → AstroAlbum)

After data is on D_MAIN, the interactive pipeline produces the final calibrated, upscaled image. The five auto-process paths are:

Path	Entry Point	Steps	Notes
A — Full Auto (menu)	U → auto-process	Ingest → Score → Stack → GraXpert → Siril → optional	Recommended for most sessions. Comma-separated

Path	Entry Point	Steps	Notes
		Starless/Cosmos/Giga → Album	row list triggers astrocombine.
B — Full Auto (CLI)	`astro_autoprocess`	Same core chain, driven from command line	Use --graxpert flag; adds --no-subsky automatically.
C — Stack only	St or sirildispatch	Collect → Stack (SO##)	When only a fresh stack is wanted; process later.
D — Process only (SO exists)	Pr or graxpert_clean + sirilprocess	GraXpert → Siril (→ Starless → Cosmos → Giga as needed)	When a stack already exists and re-processing is wanted.
E — Jump-in at any step	Manual CLI commands	Start at the required stage	See Section 13 Pipeline Reference for per-step commands.

9.3 Shuttle Drives (Mule & Camel)

Mule (4TB-Astro-Mule, M5 Air) and Camel (T7-Astro-Camel, M3 Air) are shuttle drives for sessions captured away from the primary workstation. When back at the primary workstation, run mule_sweep or camel_sweep on the Pro to pull data via rsync over SSH. Both shuttle drives use a flat RawTelescope layout (no Astronomy/PhotosData prefix), unlike D_MAIN. This is reflected in their drive_registry.json entries.

9.4 fits_only Sessions

Sessions with only raw FITS frames and no processed JPG are marked fits_only by astro_catalog and skipped by the JPG browse catalog. Their raw frames are still mirrored to AstroArchive/RawTelescope/ by netwatch, so no data is lost. They remain tracked in the index and can be promoted to the stacking pipeline via CollectFits at any time. View them with:

```
astro_index list --status fits_only
```

10 Directory Structure

10.1 AstroMaster / AstroWork (Data Drive)

Additional Workflow folders (auto-pipeline): RCXOutput/ holds the RC Astro outputs (BX##/NX##/SX##); ReviewPDF/ holds the batch review PDFs from BatchResultsPDF. On the M4 Mini the same Workflow tree is mirrored under /Volumes/AstroRAID/AstroData/PhotosData/.

```
/Volumes/AstroMaster/AstroData/  (or /Volumes/AstroWork-8TB/)
  Astronomy/
    PhotosData/
      RawTelescope/
        VP-1/
          YYYY-MM-DD_HH-MM-SS_observation_<DSO>/
            02-images-initial/    ← processed frames
        VP-2/
        Vespera1/
    Vespera2-1/
    Vespera2-2/
      Workflow/
        StackingInput/<DSO-###>/  ← ingested raw frames
        StackingOutput/           ← SO## stack files
        GraXpertOutput/           ← GX## background-removed
        ProcessedOutput/          ← PO## colour calibrated
        CosmosInput/              ← CI## staged for Cosmos
        CosmosOutput/             ← CO## tone-mapped
        GigaInput/                ← GI## staged for Gigapixel
        GigaOutput/               ← GO## upscaled
      AstroAlbum/
    <DSO>/          ← selected images
        versionLog.txt
```

10.2 AstroArchive (Mini)

```
/Volumes/AstroArchive/
  Sessions/
    2025-01-04_M42_VP1/
      img-final.jpg
      provenance.json
  DSO/
    M42/
      2025-01-04_VP1/    ← hardlinks into Sessions/ (zero bytes)
      2025-02-02_VP1/
  Calibration/           ← reserved for dark/flat libraries
  One-Time-Workflow-Backup/
```

10.3 Dropbox / ASTRO_CODE

```
Dropbox/Astronomy/
  Tools/
    bin/                    ← all pipeline scripts
  data/
    drive_registry.json     ← registered drives
    ingest_index.json       ← shared session state
    dso_catalog.json        ← DSO alias and metadata
    projects.json           ← current imaging projects
  config/
    astroenv.zsh            ← script-time environment
```

11 Troubleshooting

Most problems in the pipeline fall into a small number of categories: drives not found, sessions not appearing, skipped ingestion, and environment issues. Work through the applicable checklist before assuming a bug.

11.1 Drive Not Discovered by netwatch

- Confirm the drive is physically connected and its SMB share is active (System Settings → Sharing → File Sharing → share listed).
- Check that smb_share_name in drive_registry.json exactly matches the macOS share name.
- Run astro_netwatch --dry-run interactively on the mini to see which hosts are found.
- Verify port 445 is reachable from the mini:

```
nc -zv <host_ip> 445
```

11.2 Sessions Not Showing as Pending

- Run astro_index stats to see current counts.
- Run astro_index discover <raw_tel_path> --source D_MAIN manually to register sessions.
- Check that the observation folder name matches: YYYY-MM-DD_HH-MM-SS_observation_<DSO>.

11.3 astro_catalog Skipping Sessions

Check the log:

```
tail -80 ~/Library/Logs/astro_catalog.log
```

Common skip reasons:

- No science subfolder — observation folder has no image subdirectory and no JPGs at root. Session had no output.
- fits_only — only raw .fits frames present; no stacked JPG was produced. Normal for interrupted sessions.
- Source drive not mounted — D_MAIN, Mule, or Camel went offline. Re-plug and re-run; sessions retry automatically.
- Operation timed out (errno 60) — drive disconnected mid-run. Drive re-added to offline_drives set; remaining sessions retry on next cycle.

11.4 Siril Plate-Solve / Color Calibration Failure

If sirilprocess reports a plate-solve failure and writes no-color-cal to the .status file, the image is still fully usable — it just lacks spectrophotometric color calibration. Before processing begins, sirilprocess checks Vizier connectivity (tapvizier.u-strasbg.fr) and, if

unreachable, pauses with three choices: R = retry · Y = proceed without color cal · any other key = abort. When proceeding with Y, flag the version for later re-coloring:

```
log_op "DSO-###" "needs-recolor" "sirilprocess" ...
dso_log --op needs-recolor    # find all flagged versions
```

11.5 Terminal Text Hard to Read

astrorc sets both background and foreground (white text) together. Black text on a colored background indicates the machine is running an older astrorc. Re-source it:

```
source ~/.zshrc
```

11.6 No Data Drive on Mini

If AstroWork is not recognized (ASTRO_DATA_AVAILABLE=0):

- Check /Volumes/AstroWork-8TB is mounted: diskutil list
- Verify the drive is registered: astro_drive list

11.7 Shuttle Drive Layout Differences

Both Mule and Camel use a flat RawTelescope layout (no Astronomy/PhotosData prefix). The registry entry for each reflects this. D_MAIN uses the full Astronomy/PhotosData/RawTelescope path. astro_catalog handles both automatically via the registry.

11.8 APFS RAID Issues

If AstroWork needs to be recreated:

```
diskutil eraseDisk ExFAT TEMP1 disk4
diskutil eraseDisk ExFAT TEMP2 disk5
diskutil appleRAID create concat AstroWork-8TB APFS TEMP1 TEMP2
```

12 Quick Reference

12.1 Daily Operations Cheat Sheet

Task	Command
Check session pipeline status	`astro_index stats`
See pending sessions	`astro_index pending`
Force immediate ingest	`astro_catalog --verbose`
Preview ingest without writing	`astro_catalog --dry-run`
Process one specific session	`astro_catalog --session VP-1/2025-01-04_...`
Check daemon health	`astro_netwatch --status && astro_catalog --status`
View netwatch log	tail -50 ~/Library/Logs/astro_netwatch.log
View ingest log	tail -50 ~/Library/Logs/astro_catalog.log
List registered drives	`astro_drive list`
Show all sessions for a target	`astro_index list --dso M42`
Full status report	`astrostatus`
Version details for a DSO	sourceinfo M42-001

12.2 Filename Staging Codes

Code	Stage and Folder
SO##[_fits]	Stack Output — StackingOutput/
GX##[_fits]	GraXpert Output — GraXpertOutput/
PO##[_fits]	Process Output — ProcessedOutput/
SL##	Starless Layer — StarlessOutput/
SK##	Star Layer — StarlessOutput/
SF##	Starless Final — StarlessOutput/
CI##[_fits]	Cosmos Input — CosmosInput/
CO##	Cosmos Output — CosmosOutput/

Code	Stage and Folder
GI##	Giga Input — GigaInput/
GO##	Giga Output — GigaOutput/
BX##[_fits]	BlurXTerminator (RC Astro) -- RCXOutput/
NX##[_fits]	NoiseXTerminator (RC Astro) -- RCXOutput/
SX##[_fits]	StarXTerminator (RC Astro), starless -- RCXOutput/

12.3 Scope Keys

Key	Scope
vp	VP-1 and VP-2 VesperaPro — 250mm FL IMX676. Stacking/processing key (consolidated May 2026).
vp1 / vp2	Per-unit keys for vespera_pull / astro_allpull (WiFi download only). Not used for stacking — use vp.
v2	Vespera 2 (both units) — 200mm FL IMX585. Stacking/processing key (consolidated May 2026).
v2a / v2b	Per-unit keys for vespera_pull / astro_allpull (WiFi download only). Not used for stacking — use v2.
v1	Vespera 1 (retired from download; processing still supported)
mixed	Multiple scopes combined in one version

12.4 Stretch Level Guidance (--stretch-level N)

Level	Target / Examples
1 – 3	Bright — M42 (Orion), M31, M8, M45, M20
4 – 6	Typical — M3, M51, M101, most galaxies and nebulae
7 – 10	Faint / extended — M81, M82, faint outer halos, low-surface-brightness

13 Processing Pipeline Reference

13.1 Ingestion vs. Stacking

These two operations are always separate and must happen in order. INGESTION moves raw frames from the telescope into the pipeline directory structure: CollectFits and CollectTiffs scan RawTelescope/<scope>/<observation>/ and copy frames into StackingInput/<version>/ using a standardized naming convention. MD5 content hashing ensures no duplicate frame lands in two version folders. After CollectFits, run regen_sources_json to build the .sources.json sidecar.

STACKING is the mathematical process by which Siril combines many individual frames into a single calibrated master image: registration (aligning frames), rejection (discarding outlier pixels), and integration (stacking survivors). Input: StackingInput/<version>/. Output: StackingOutput/<version>_SO##[_fits].fit. A version with frames in StackingInput/ but no SO## file has been ingested but not yet stacked.

13.2 Standard Pipeline

RC Astro (optional): BlurXTerminator, NoiseXTerminator, and StarXTerminator run on the LINEAR stack between stacking and Siril, via astro_autoprocess --blurx / --noisex / --starx (outputs BX/NX/SX in RCXOutput/). They are the automatic pipeline's default sharpen/denoise/star-removal stage.

Ingest → Stack (SO##) → GraXpert (GX##) → sirilprocess (PO##) → [Starless SL##/SK##/SF##] → [Cosmos CI##/CO##] → [Gigapixel GI##/GO##] → AstroAlbum

GraXpert background extraction runs automatically in the Process (Pr) and auto-process (Ap) flows. When GraXpert has run, sirilprocess automatically skips its own background subtraction for _GX inputs; running both background-removal steps would destroy the black point. Starless Processing is optional after PO##; use it when independent sharpening of nebulosity or galaxy structure is desired. Cosmos is optional; use it for high dynamic range targets (M42, M31, M8) where the auto-stretch leaves the core overexposed.

13.3 Jump-In Reference — Starting at Any Step

Use when a run was interrupted or a start from a specific stage is needed:

Step 0 · Ingest Raw Frames

```
CollectFits DSO-###          # FITS frames (VP-1, VP-2, V2)
CollectTiffs DSO-###         # TIFF frames
regen_sources_json DSO-###   # automatic after CollectFits; run
by hand only to rebuild
```

Step 1 · Stack

```
sirildispatch DSO-###                # interactive format chooser
stackdso_fits_v1 DSO-###             # FITS stack
stackdso DSO-###                     # TIFF stack
# Output: StackingOutput/<version>_SO##[_fits].fit
```

Step 2 · GraXpert Background Removal

```
graxpert_clean DSO-### --stack-flavor fits
# Output: GraXpertOutput/DSO-###_GX01_fits.fit

# ALWAYS use --no-subsky when feeding GraXpert output:
sirilprocess DSO-### vp1 \
  --input-file
$ASTRO_BASE/PhotosData/Workflow/GraXpertOutput/DSO-
###_GX01_fits.fit \
  --no-subsky --stretch-level 5
```

Step 3 · Siril Only (no GraXpert)

```
sirilprocess DSO-### vp1 --stretch-level 5
# Output: ProcessedOutput/DSO-###_PO##[_fits].tif
```

Step 4 · Starless Processing (optional)

siril_starless DSO-### --sharpen "1.0 0.7"
Output: StarlessOutput/DSO-###_SF##.tif plus SL## and SK## layers

Step 5 · Cosmos Tone-Mapping (optional)

PrepareCosmos DSO-### --from-starless --stack-flavor fits # use --from-po if Starless was not run

```
RunCosmos DSO-###
# IMPORTANT: select 'ALREADY STRETCHED' in Cosmos
# [tone-map in Cosmos, export 16-bit TIFF to ~/Downloads/]
cosmos_export DSO-###
# Output: CosmosOutput/DSO-###_CO##.tif
```

Step 6 · Gigapixel Upscaling

PrepareGiga DSO-###   # stages best available file as GI## (CO## > SF## > PO##)

```
RunGiga DSO-###         # opens Gigapixel
# [upscale and export from Gigapixel to ~/Downloads/]
```

Step 7 · Move to Album

```
MoveToAlbum DSO-###
```

14 Tips & Tricks for Better Processing

14.1 Choosing the Right Stretch Level

The stretch level controls how aggressively sirilprocess pulls out faint detail. Too low: faint nebulosity disappears into the background. Too high: the background goes grey and the core blows out. Use the table in Section 12.4 as a starting point. Combined versions with high frame counts (1000+) often support one step higher than single-night data because the improved SNR gives more headroom. Reprocessing is always safe — each run creates a new PO## file and old ones are never deleted. Experiment freely.

14.2 Background Subtraction and GraXpert

GraXpert removes the background gradient by subtracting a smooth AI-derived model. Siril's subsky step performs the same operation using a polynomial fit. Running both on the same image subtracts an already-flat background and creates an artificially dark sky that crushes faint nebulosity — historically the most common processing mistake in the pipeline. As of June 2026 this is handled automatically: sirilprocess detects a GraXpert (_GX) input by its filename and skips its own subsky step, whether invoked directly or through astromenu. The --no-subsky flag remains available and is harmless to include for explicitness. Double-subtraction is only possible when running a sirilprocess version predating the auto-guard (pre-June 2026 backups).

14.3 Tuning GraXpert Smoothing

Default GraXpert smoothing is 0.5, which handles most gradients well. If a residual gradient remains in the processed output (a bright corner or light-pollution band), try a more aggressive setting:

```
graxpert_clean M3-C01 --stack-flavor fits --smoothing 0.3
```

If GraXpert over-corrects (dark patches where the sky should be flat, or nebula structure being mistakenly subtracted), back off to 0.7. For broadband targets like M3, 0.3–0.5 is the typical working range.

14.4 When to Use Cosmos — and When to Skip It

Cosmos serves two purposes: (1) HDR tone mapping for targets with extreme dynamic range — M42 (Orion Nebula), M31 (Andromeda), M8 (Lagoon); and (2) color and tone refinement for spiral galaxies with strong color structure — M101, M51, M33. For globular clusters, elliptical galaxies, and many diffuse nebulae, sirilprocess or Starless output is often sufficient. Treat Cosmos as optional: keep the result only if it clearly improves the presentation. File size limit: Cosmos has a 500 MB input cap. Always feed it a CI## file staged by PrepareCosmos, never a raw stack.

14.5 The Cosmos 'Already Stretched' Gotcha

When a CI## file is opened in Cosmos, it asks whether the image has already been stretched. Always choose ALREADY STRETCHED. sirilprocess applies autostretch before writing the PO## file. If Cosmos is told the data is not yet stretched, it applies its

own stretch on top of the existing one and the result will be blown out and unusable. A good Cosmos workflow for a typical galaxy: Curves — gentle midtone boost → Commit. Shadows +20 to +40. Highlights -15 to -20. Saturation +20 to +35. Color temperature +5 to +10 if the image looks too blue.

14.6 Gigapixel Settings for Astrophotography

From M101 VP-1 testing: Model = Low Resolution. Scale = 2x (4x usually creates much larger files without useful additional information). Sharpen = 50–60 as a starting point; avoid excessive sharpening because it can create halos around stars. Denoise = 8. Gamma correction = ON. Fix compression = 0 for TIFF input. At 2x with these settings a VP-1 image produces roughly 60 megapixels, supporting approximately 20"×24" prints at 300 DPI.

14.7 Registration Rate: What to Expect

Not all frames successfully register to the reference. Typical rates: VP-1/VP-2 (IMX676, 250mm) 85–95% on rich fields. Vespera 2 (IMX585, 200mm) 80–92%. Vespera 1 (IMX462, 50mm) 40–65% — sparse star fields hurt more on the smaller aperture. A rate below 30% usually signals a real problem: wrong focal length in Siril, a mosaic observation fed to the single-field stacker, or an extremely sparse star field.

14.8 Multi-Night Combines: When and How

Combining multiple nights of data dramatically improves SNR and reveals faint outer structure. Pool raw frames before stacking — do not stack first and then try to combine stacks. Via astromenu, St (Stack) and Ap (auto-process) both accept a comma-separated list of row numbers (e.g. "1,3"). astrocombine validates that all selected versions are the same DSO, deduplicates frames by filename and MD5, and creates a combined version named DSO-C## (e.g. M3-C01).

14.9 Automatic Combine: Adding New Data to Existing Targets

astromenu can drive the combine workflow automatically. When a new observation of a target that already has earlier versions on the drive is ingested, astromenu detects the existing siblings and offers to combine them on the spot: Combine [version] with it now? [Y = combine + autoprocess, l = later, n = no]. Choosing Y runs astrocombine on the new version together with its existing siblings, creates the next combined version (DSO-C##), and then runs astro_autoprocess on that combined version with GraXpert. In other words, adding a night's data onto an existing target — recombining, stacking, and processing — happens in a single step. Choosing l (later) records the opportunity for batch processing; n declines.

Deferred and newly discovered combine opportunities are tracked by astro_combine_pending and surfaced in the astromenu Combine submenu (Cb). The submenu header shows a count of DSOs that have new data not yet combined with their existing observations; press p to review the list. A combine opportunity is any DSO with at least one un-combined original version and two or more combinable units in total

— found either because Later was chosen at the post-ingest prompt, or by a live scan of versionLog.txt and the .sources.json provenance sidecars.

From the pending list, act on a single DSO or type all to process every pending target in batch. The all run works in two phases: it first combines and stacks each pending DSO (skipping any that are TIFF-only), then auto-processes each newly combined version with default settings. This is the fastest way to fold a backlog of new sub-frames into existing targets and bring every combined result up to a selected image in one pass. Use d N to dismiss a DSO (stop prompting until more data arrives) or x N to forget it.

14.10 Reprocessing Is Always Safe

Every pipeline step writes a new incrementing output (PO01, PO02, PO03...) and never overwrites an older one. If a stretch level is unsatisfactory, run sirilprocess again with a different level — the result is PO02. PrepareGiga always picks the latest PO## unless otherwise specified. The same applies to stacking: sirildispatch again produces SO02, SO03. Old stacks are never deleted automatically. Note: the always-001 versioning model means the version folder itself is always DSO-001, but output files within StackingOutput/ProcessedOutput/etc. still increment normally.

15 How-To Quick Reference

This chapter is a practical companion to the rest of the manual. Every routine task is shown two ways: the astromenu path and the command-line equivalent. They do the same work — the menu fills in the arguments automatically. Common tasks are written as short numbered recipes; a quick-lookup table at the end of each section covers the rest. Throughout the chapter, the notation DI → Cs means: from the main menu choose DI, then choose Cs in the submenu. Examples use M42-001 (Orion Nebula, version 1) as the working target; substitute the real version and scope key (vp1, vp2, v2a, v2b, v1).

15.1 Acquire Data

Pull last night's data from the scopes

1. astromenu: DI → Sc, then choose D_MAIN as the destination.
2. Confirm the Mac is joined to each powered-on scope's WiFi network. APAS downloads all raw FITS frames plus the preview image, then automatically collects them into pipeline versions.

Command line:

```
astro_allpull --with-fits --dest D_MAIN
vespera_pull vp1 --with-fits   # one scope only
```

Bring shuttle-drive data home

Sessions captured away from the workstation land on the Mule (M5 Air) or Camel (M3 Air) shuttle drives. Back at the Pro, sweep them into D_MAIN over SSH. Add --dry-run first to preview.

```
mule_sweep             # Mule (M5 Air) -> D_MAIN
camel_sweep            # Camel (M3 Air) -> D_MAIN
sweep_all              # both, in one pass
```

Import from a local disk or SSD

```
disk_pull <source> --scope <key> --execute
```

Task	astromenu	Command line
Pull last night from all scopes	DI → Sc	`astro_allpull --with-fits --dest D_MAIN`
Pull from one scope only	DI → Sc (pick one)	`vespera_pull vp1 --with-fits`
Sweep Mule (M5 Air) → D_MAIN	DI → Ms	`mule_sweep (--dry-run to preview)`
Sweep Camel (M3 Air) → D_MAIN	DI → Cs	`camel_sweep (--dry-run to preview)`
Sweep both shuttles → D_MAIN	DI → As	`sweep_all (--dry-run to preview)`

Task	astromenu	Command line
Import from a local disk/SSD	Fm → Dp	`disk_pull <source> --scope <key> --execute`

15.2 Ingest & Combine

Ingest an observation into a version

Ingestion copies the raw frames into a standardized, deduplicated pipeline directory (for example M42-001). After a scope pull, astromenu runs this automatically; run it by hand when needed.

1. astromenu: In, then pick the DSO and the observation(s) to ingest — a number, a range (1 3), or 0 for all.

Command line:

```
astrostart M42
CollectFits M42-001 --fresh        # CollectTiffs for TIFF-mode obs
regen_sources_json M42-001
```

Combine multiple nights into one version

Combining pools two or more versions of the same target into a single DSO-C## version, then stacks all of their frames together.

1. astromenu: Cb (or Ap, then enter a comma-separated list such as 1,2,3 at the version prompt).

Command line:

```
astrocombine M3-001 M3-002 M3-003   # creates M3-C01
sirildispatch M3-C01 --default F
```

Adding a night to an existing target: Re-pull and re-collect — CollectFits adds only new frames by MD5 hash, so it is safe to run against an existing version. astromenu then offers to combine the new data with the earlier siblings on the spot.

Task	astromenu	Command line
Ingest raw obs into a version	In	`astrostart <DSO>; CollectFits <ver> --fresh; regen_sources_json <ver>`
Ingest TIFF-mode observations	In	`CollectTiffs <ver> --fresh`
Combine versions, then stack	Cb	`astrocombine <v1> <v2> … then sirildispatch <DSO-C##>`
Combine via auto-process	Ap (enter	`astrocombine … then`

Task	astromenu	Command line
	1,2,3)	`astro_autoprocess <DSO-C##> <scope>`
Force a provenance rebuild	—	`regen_sources_json <ver>`

15.3 Stack

Stack a version

1. astromenu: St, then pick the version. Choose the frame format at the prompt (F FITS, T TIFFs, B both).

Command line (sirildispatch chooses the right engine — stackdso_fits_v1 for FITS, stackdso for TIFF):

```
sirildispatch M42-001 --default F
```

Score frames, then stack

Scoring writes a per-frame quality sidecar that stacking reads to exclude the poorest frames.

```
score_frames M42-001
sirildispatch M42-001 --default F
```

Re-stack a target (fresh cull or fix)

A re-stack rebuilds the master from the raw frames, keeping the best percentage by weighted FWHM. Use it after adding frames or to apply culling.

```
astro_restack M101-001              # per-target (dashboard R)
run-all --restack                  # whole backlog (M4 Mini)
```

Offload a stack to another Mac

```
astro_autoprocess M101-001 vp1 --remote <HOST> --format fits
```

Task	astromenu	Command line
Stack a version	St	`sirildispatch <DSO-###> --default F|T|B`
Score frames, then stack	SS	`score_frames <ver>; sirildispatch <ver>`
Dual stack (Pro + Air in parallel)	Du	–
Re-stack one target	dashboard R	`astro_restack <ver>`
Re-stack the whole backlog	dashboard B → 2	`run-all --restack [--cull-pct N] [--only VER]`

Task	astromenu	Command line
Offload a pipeline to a peer Mac	St → remote prompt	`astro_autoprocess <ver> <scope> --remote <HOST>`

15.4 Process

Run the process step (GraXpert + Siril)

1. astromenu: Pr, then pick the version.
2. Enter a stretch level (see the table below).
3. Answer the prompts: “Defer sharpening to the Starless step?” (yes only if Starless will follow) and “Apply deconvolution?” (optional edge sharpening).
4. Inspect the resulting PO## file. If the stretch is wrong, run Pr again with a different level — each run writes a new PO##; the old one is never overwritten.

Command line:

```
graxpert_clean M42-001 --stack-flavor fits
sirilprocess M42-001 vp1 --no-subsky --stretch-level 3
```

Hands-off auto-process

auto-process chains scoring, stacking, GraXpert, and the Siril stretch to a finished PO## with no prompts.

```
astro_autoprocess M42-001 vp1 --format fits --stretch-level 3
--graxpert
```

Choose a stretch level

Target brightness	Stretch level	Examples
Bright	1 – 3	`M42 (Orion), M31 (Andromeda), M45 (Pleiades), M8 (Lagoon)`
Typical	4 – 6	`Most galaxies and nebulae — M3, M51, M101, NGC 2174`
Faint / extended	7 – 10	`Low-surface-brightness galaxies, faint outer halos`

Deconvolution & denoise: Add --deconvolve to sharpen the linear image before the stretch (tune with --deconv-iters N, default 10 — lower it if dark rings appear around bright stars). sirilprocess auto-skips its own background subtraction and denoise for GraXpert (_GX) inputs; force them with --no-subsky / --no-denoise, and omit the unsharp pass with --no-sharpen when a Starless step will follow.

Run a single tool step

The RC Astro plugins (BlurXTerminator, NoiseXTerminator, StarXTerminator) run on the linear stack between stacking and Siril, writing BX/NX/SX outputs to RCXOutput/.

Task	astromenu	Command line
Full process (GraXpert + Siril)	Pr	`graxpert_clean <ver> --stack-flavor fits; sirilprocess <ver> <scope> --no-subsky --stretch-level N`
Siril only (no GraXpert)	Ts → Si	`sirilprocess <ver> <scope> --stretch-level 5`
Hands-off auto-process	Ap	`astro_autoprocess <ver> <scope> --stretch-level N [--graxpert]`
GraXpert step only	Ts → Gx	`graxpert_clean <ver> --stack-flavor fits [--smoothing N] [--denoise]`
RC Astro (BlurX / NoiseX / StarX)	Ts → Rx	`astro_autoprocess <ver> <scope> --blurx --noisex [--starx]`
Deconvolve without AI	—	`sirilprocess <ver> <scope> --deconvolve [--deconv-iters N]`

15.5 Finish: Starless, Cosmos, Gigapixel & Album

The interactive hand-off — read this first. Cosmos Darkroom and Topaz Gigapixel are interactive apps. The pipeline stages an input file, opens the app, and then waits for you. Both apps export their result to your ~/Downloads/ folder, and the cosmos_export and giga_export scripts simply pick up the newest matching TIFF from there. That means the remaining steps do NOT depend on the pipeline terminal that is waiting: if it is interrupted, or you come back a day or two later, just finish in the app and then run the export and downstream steps by hand from any fresh terminal. Nothing is lost.

Starless sharpening (StarNet2)

Starless processing sharpens faint nebulosity and galaxy structure without enlarging the stars. Best run on a PO## created with --no-sharpen so sharpening happens once, on the starless layer. Outputs SL/SK/SF; stage the SF## output if Cosmos will follow.

```
siril_starless M42-001 --sharpen "1.0 0.7"
```

Tone-map in Cosmos Darkroom

1. astromenu: Ts → Cs, then pick the version (or run PrepareCosmos + RunCosmos on the command line). This stages a CI## file and opens Cosmos.

2. In Cosmos, always choose ALREADY STRETCHED (sirilprocess already applied the stretch). Tone-map, then export a 16-bit TIFF to ~/Downloads/.

Command line to stage and open:

```
PrepareCosmos M42-001 --from-po --stack-flavor fits   # or --
from-starless
RunCosmos M42-001
```

The next step after you save in Cosmos

Do this whenever you finish in Cosmos — immediately, or days later in a brand-new terminal. It does not matter whether the original pipeline is still running.

1. In Cosmos, export a 16-bit TIFF to ~/Downloads/ (if you reopened the file, choose ALREADY STRETCHED again).
2. Move the export into the pipeline. This creates CosmosOutput/<ver>_CO##.tif:
   ```
   cosmos_export M42-001
   ```
3. Continue to Gigapixel (next recipe), or finish now by sending the Cosmos result straight to the album:
   ```
   MoveToAlbum M42-001        # picks CO## automatically
   ```
4. Publish to the web if desired (Section 15.6).

If a later step cannot find CO##: cosmos_export occasionally drops the observation suffix from the filename. Check CosmosOutput/ for a truncated name (e.g. M42_CO01 instead of M42-001_CO01) and rename it to match the version.

Upscale in Topaz Gigapixel

1. astromenu: Ts → Gp, then pick the version (or run PrepareGiga + RunGiga). This stages a GI## file — preferring CO## if Cosmos was run, else PO## — and opens Gigapixel.
2. Apply the recommended settings (Model = Low Resolution, Scale = 2x, Sharpen = 50–60, Denoise = 8, Gamma = ON), then export to ~/Downloads/.

Command line to stage and open:

```
PrepareGiga M42-001            # --source po|co|gx to override
RunGiga M42-001
```

The next step after you save in Gigapixel

1. In Gigapixel, export the upscaled image to ~/Downloads/.
2. Move the export into the pipeline, creating GigaOutput/<ver>_GO##.tif:
   ```
   giga_export M42-001
   ```
3. Copy the finished image into the album (next recipe), then publish (Section 15.6).

Move a finished image to the album (and refresh it)

MoveToAlbum copies the best available final — GO## (Gigapixel), else CO## (Cosmos), else PO## (Siril) — into AstroAlbum/<DSO>/ with a clean filename, and imports it into the “AstroAlbum” album in Apple Photos. It is duplicate-safe, so re-

running it after producing a better final simply refreshes the album copy — this is how you make a new version of an album image.

1. astromenu: Finishing → (Ma) Move to Album, or run it directly:

```
MoveToAlbum M42-001
```

One-time setup: MoveToAlbum needs Automation permission the first time — System Settings → Privacy & Security → Automation → Terminal → Photos.

Task	astromenu	Command line
Starless sharpening (StarNet2)	Ts → Sl	`siril_starless <ver> --sharpen "1.0 0.7"`
Stage + open in Cosmos	Ts → Cs	`PrepareCosmos <ver> --from-po|--from-starless; RunCosmos <ver>`
After Cosmos-save: import result	Fm → Ce	`cosmos_export <ver>  (→ CO##)`
Stage + open in Gigapixel	Ts → Gp	`PrepareGiga <ver> [--source po|co]; RunGiga <ver>`
After Gigapixel-save: import result	Fm → Ge	`giga_export <ver>  (→ GO##)`
Move / refresh in the album	Fm → Ma	`MoveToAlbum <ver>`

15.6 Publish to the Web

Publishing adds a finished image to the online gallery at the astro.milligangridsolutions.com subdomain. See Section 4.3 for the full description of the gallery, the calendar app, and deployment.

Publish a finished image to the gallery

1. From the per-target board press (W) Publish to Web, or from the Finishing menu press (Pw) Publish to Web Page. APAS auto-detects the most-finished version (Album → Gigapixel → Cosmos → Siril), resizes it, adds it to the gallery, and offers to deploy.
2. To do it from the command line, add the image and then deploy:

```
build_gallery.py add M42-001_GO03.tif --object M42
deploy_gallery.sh
```

3. Fill in the image's metadata (object, scope, filter, integration, date, caption) in gallery.json; a blank stub is added automatically for each new image.

Rebuild and publish the imaging-calendar app

1. astromenu: Observation Tools & Planning → (Vp) Publish App. This regenerates the mobile calendar app and uploads it to the /calendar/ path.

```
build_calendar_app.py && deploy_calendar_app.sh   # equivalent
on the CLI
```

Deployment & credentials: Both deploy scripts upload over HTTPS through cPanel's UAPI on port 2083, authenticated with a scoped API token in Tools/app/deploy.conf (0600 permissions; rotated after setup). Uploads pass overwrite=1. See Section 4.3.

Task	astromenu	Command line
Publish a final to the gallery	board (W) / Fm → Pw	`build_gallery.py add <final.tif> --object <DSO>; deploy_gallery.sh`
Add an image without deploying	—	`build_gallery.py add <final.tif> --object <DSO>`
Deploy the gallery only	—	`deploy_gallery.sh`
Rebuild + publish calendar app	Ob → Vp	`build_calendar_app.py; deploy_calendar_app.sh`

15.7 Remote & Parallel Processing

Offload one pipeline to another Mac

```
astro_autoprocess M101-001 vp1 --remote <HOST> --format fits
```

Run the automatic batch (M4 Mini)

run-all watches for new data, ingests it, and dispatches a full pipeline per worker machine. Its per-version pipeline is: stack (with cull) → GraXpert → BlurX → NoiseX → Siril stretch → PO. It normally runs as a launchd daemon; check on it with --status.

```
run-all               # one pass: sync + ingest + dispatch,
then exit
run-all --watch       # continuous daemon mode
run-all --status      # daemon state, queue, running pipelines
```

Reprocess or re-stack the whole backlog

From the cabin, trigger the Mini to reprocess every finished target; the work stays local to the Mini.

1. SSH in over Tailscale, then restart the daemon so it loads the current flags:

```
remote m4mini
launchctl kickstart -k gui/$(id -u)/com.astro.run-all
```

2. Queue the job (or press B on the Mini's dashboard):

```
run-all --reprocess    # GraXpert+RC+Siril, reuse stacks
run-all --restack      # full re-stack + cull
```

Verify a batch finished

```
astro_batch_verify --since 2
astro_batch_verify M101-001 M3-C01
```

Task	astromenu	Command line
Offload one pipeline to a peer	St → remote	`astro_autoprocess <ver> <scope> --remote <HOST>`
Run the batch once	—	`run-all`
Batch daemon (watch)	—	`run-all --watch`
Batch status	—	`run-all --status`
Reprocess all finished targets	dashboard B → 1	`run-all --reprocess`
Re-stack all finished targets	dashboard B → 2	`run-all --restack`
Verify batch completion	dashboard B → Vf	`astro_batch_verify [versions] [--since N]`

15.8 Status, Provenance & Troubleshooting

Check status and trace a version

```
astrostatus              # whole pipeline (alias: astat)
astrostatus M42          # one target
sourceinfo M42-001       # provenance summary
find_target M42          # locate a target across all stages
```

Read the operations log

```
dso_log                       # recent operations
dso_log --op needs-recolor    # versions flagged for re-color-calibration
```

Common problems and fixes

Symptom	Likely cause	Fix
command not found after install	astrorc not sourced in this shell	`Open a new terminal, or run: source ~/.zshrc`
astromenu shows no data drive	D_MAIN not mounted	`Plug in / remount the SSD; re-open the menu`
Stretch too aggressive or too flat	Wrong stretch level	`Re-run Pr with a different level (each run writes a new`

Symptom	Likely cause	Fix
		`PO##)`
Dark rings around bright stars	Too many deconvolution iterations	`Lower --deconv-iters (or omit --deconvolve)`
no-color-cal / plate-solve failure	Vizier (SPCC) unreachable	`At the prompt press R to retry or Y to proceed; re-run when Vizier is back. Flag it: dso_log --op needs-recolor`
PrepareGiga cannot find CO##	cosmos_export dropped the suffix	`Rename the truncated file in CosmosOutput/ to match the version`
Terminal closed while Cosmos/Giga open	Interactive step was interrupted	`Finish in the app, then run cosmos_export / giga_export by hand (Section 15.5)`
Batch target looks done but is stale	Old PO## from a prior run	`Run astro_batch_verify to confirm a fresh PO##`

Help on any command or topic: Press H in astromenu, or run astrohelp <topic> at the command line.

15.9 Planning & Drives

Plan tonight's targets

The Viewing Calendar lists the month's best targets for the Denver and South Park sites, with visibility windows and a recommended scope. It also drives the Word export and the phone app.

```
# astromenu: Vc (Viewing Calendar)  ·  Ob → Vp (phone app)
```

Which scope for a target? The calendar's recommendation is sensor-aware: diffuse emission nebulae route to the wide, faster Vespera 2 (at ≥18′), while galaxies and clusters stay on the higher-resolution Vespera Pro (until ≥45′). Framing — fewer mosaic tiles — wins first. See Section 4.3.

Register or initialize a drive

```
astro_drive list
astro_drive register <name> <path>
astro_drive init <name>
```

Back up and archive

```
backup_scripts                  # back up the bin scripts
astro_backup --dry-run          # preview a PhotosData backup
archive_fits M42                # move raw FITS off D_MAIN
archive_giga                    # move Gigapixel output off
D_MAIN
```

Task	astromenu	Command line
List drives + mount status	Ts → Dr	`astro_drive list`
Register a new drive	Ts → Dr → Re	`astro_drive register <name> <path>`
Create folder structure on a drive	Ts → Dr → In	`astro_drive init <name>`
Keep external drives awake	Ut → Ka	`keepalive_drives`
Back up bin scripts	Ut → Bk	`backup_scripts`
Back up PhotosData → archive	—	`astro_backup (--dry-run, --status)`
Archive raw FITS off D_MAIN	Fm → Af	`archive_fits <DSO>`
Archive Gigapixel output	Fm → Ag	`archive_giga`
Viewing calendar	Vc	—
Publish calendar to phone app	Ob → Vp	`build_calendar_app.py; deploy_calendar_app.sh`

15.10 Worked Examples

Capture through publish, in one sitting

```
astro_allpull --with-fits --dest D_MAIN
astrostart M42
CollectFits M42-001 --fresh && regen_sources_json M42-001
astro_autoprocess M42-001 vp1 --format fits --stretch-level 3
--graxpert
# tone-map in Cosmos, export 16-bit TIFF to ~/Downloads/
cosmos_export M42-001
PrepareGiga M42-001 && RunGiga M42-001
# upscale in Gigapixel, export to ~/Downloads/
giga_export M42-001
MoveToAlbum M42-001
build_gallery.py add M42-001_GO01.tif --object M42 &&
deploy_gallery.sh
```

Resume a finish two days later (interrupted pipeline)

The auto-process run opened Cosmos and the terminal was later closed. Pick up from a fresh terminal:

```
# finish in Cosmos, export 16-bit TIFF to ~/Downloads/
cosmos_export M101-004
PrepareGiga M101-004 && RunGiga M101-004
# upscale in Gigapixel, export to ~/Downloads/
giga_export M101-004
MoveToAlbum M101-004
# then publish: Finishing (Pw), or build_gallery.py add … &&
deploy_gallery.sh
```

Multi-night combine (M3, three sessions)

```
astrostart M3
CollectFits M3-001 --fresh && regen_sources_json M3-001
astrocombine M3-001 M3-002 M3-003   # creates M3-C01
astro_autoprocess M3-C01 vp1 --format fits --stretch-level 6
--graxpert
```

Jump in at GraXpert (stack already exists)

```
graxpert_clean M3-C01 --stack-flavor fits
sirilprocess M3-C01 vp1 \
  --input-file
$ASTRO_BASE/PhotosData/Workflow/GraXpertOutput/M3-
C01_GX01_fits.fit \
  --no-subsky --stretch-level 5
PrepareGiga M3-C01 && RunGiga M3-C01 && MoveToAlbum M3-C01
```