GRIME3 0.1.0 RC9 Complete User Guide

Revision 1.0 - 15 October 2026

GRIME3 measures water level from images using an octagonal calibration target. The browser is the primary interface; the CLI exposes the same deterministic measurement engine. Normal measurement works locally without a cloud service. RC9 adds catalog/Data Roots, image compatibility, quality evidence and native browsing. Demonstration playback and hydrograph controls belong to RC10.

1. Choose an installation

Windows and Linux standalone packages include Python and their runtime. Extract the complete package and keep the executable beside `_internal`. Windows starts with `START_GRIME3.bat` or `GRIME3.exe`; Linux starts with `start-grime3.sh` or `./grime3`. Verify the archive checksum before extracting. Use the RC9 platform Quick Start for filenames and exact installation steps.

Conda users create a dedicated environment by package name from the published RC9 channel. The primary one-line command is in the Conda Quick Start. Activate the environment and run `grime3 web --open-browser`. Use another environment name if `grime3-rc9` already holds a development installation. Docker is intended for a server/headless deployment and retains GRIME3's own file browser. Pre-tag validation builds are evaluation artifacts, not published releases.

Runtime and distribution identity

Official RC9 distributions use headless OpenCV and exclude FFmpeg. GRIME3 measures still images; its measurement workflow does not require a separate video-codec installation. The native OpenCV version is 5.0.0, supplied by the prepared `opencv-python-headless` distribution version `5.0.0.93+grime3.1`.

Use the complete GRIME3 standalone package or the documented Conda channel. Replacing OpenCV or other runtime dependencies changes the validated installation. The RC9 Conda runtime currently requires Python 3.11 and a platform-specific `grime3-opencv` companion supplied by the RC9 channel. On Windows the companion selects the tested Python 3.11.9 runtime and native dependencies. Follow the Conda Quick Start; other Windows Python builds have not been accepted for RC9.

`BUILD_INFO.json`, the application's build identity and the packaged license inventories identify the source and components actually distributed. Keep this information with scientific results. Historical FFmpeg-containing validation artifacts are not the FFmpeg-free RC9 distribution.

2. Start and stop GRIME3

Use the production launcher and the exact browser URL it prints:

```
grime3 web --host 127.0.0.1 --open-browser
```

Standalone users can omit arguments to open the browser. Keep the launcher terminal open while using GRIME3. The About information and printed build ID identify the running version/commit. Closing the browser tab does not stop the server; press Ctrl+C in the launcher terminal to stop it.

An occupied port produces a clear error. Stop your own previous GRIME3 launcher or select another port, such as `--port 8010`. Do not terminate unrelated programs just because they use the desired port. Reload the exact new URL after switching builds; verify the build ID before measuring.

3. Load images and use native browsing

Choose the image folder using Browse or enter its path. Recursive discovery includes child folders when selected. Native folder selection loads the image folder immediately. Use Previous/Next, Fit, 1:1 and zoom to inspect original image geometry; display zoom does not change source-image pixels.

Local Windows uses shell dialogs; Linux uses its desktop portal when available. A native dialog may appear behind the browser; check the taskbar. Cancel leaves paths unchanged. Only one native picker can be active per server at a time. Docker, SSH, remote/headless and unavailable providers use the GRIME3 browser. Paths must be accessible to the backend computer.

Output-folder selection fills the field without starting measurement. Data Root and Site package selection likewise fill fields without registering them. Manual paths and Enter-to-load remain available.

4. Load or create a Site Configuration

A Site Configuration JSON is the canonical scientific authority: image identity/signature, ROIs, calibration geometry, facet length, units, datum, processing settings, persistent identities and provenance travel together. Keep it with the site dataset and results. Do not substitute separate ROI or calibration JSON files in the production measurement workflow.

To reuse one, Browse for its JSON file or type its path and press Enter/Load. Verify the site name, units, datum and calibration overlays on a representative image. Source geometry and pixel organization must be compatible with the configuration before calibration or measurement reuse.

To create one, load a suitable original image, define the octagon search ROI and valid waterline ROI, enter the Site name and Operator, facet length, units and bottom-left octagon corner waterlevel, and choose the processing settings. Create becomes available when those requirements are complete. Choose the JSON destination and inspect the detected target geometry before relying on it.

Inspect all eight octagon vertices at high zoom. Calibration follows the inner boundary between the target face and its black border, rather than the outside border. The configurable minimum checks both corner width and height in the original image. The default is 100.0 pixels; a positive finite value is required. The limit is a size check, not a guarantee of good geometry or uncertainty. Force-small detection and display zoom do not override it or other validity checks. Read the calibration reference for grid datum and corner acceptance.

5. Save, Save As and provenance

Save and Save As require a loaded/created configuration and a nonblank Operator. Enter the Operator in Calibration's creation controls if either button remains gray. Save writes to the current JSON path; Save As selects another JSON path. Cancel keeps the current path and existing file unchanged. An existing target requires overwrite confirmation; canceling it preserves the existing file.

Saving the minimum-octagon processing setting preserves existing calibration geometry and audit history. The creation-time setting remains in calibration metadata. Keep the full configuration JSON, not just a screenshot or its path, when sharing a scientific measurement setup.

6. Image compatibility and older configurations

New configurations record native width, height, channel/color organization, dtype and effective bit depth. Validation occurs before geometry or template reuse and measurement. An incompatible image is rejected with diagnostic provenance. The current pipeline uses 8-bit inputs; higher-bit-depth input is not silently converted into a supported image.

Older configurations without a full signature retain explicit dimension-only compatibility. Loading them does not silently rewrite them. The verified `migrate-image-signature` CLI command can add a signature using the original source image and checksum; consult its help and the compatibility reference. Recalibrate when camera/target geometry changes even if pixel dimensions match.

7. Catalog, Data Roots and assignments

The machine-local catalog remembers the locations of portable site metadata. Persistent UUIDs identify Data Roots, Sites and Site Configurations. Paths and folder names are locations, not scientific identity. Image folders keep their existing hierarchy; registration does not reorganize the images.

Use Catalog controls to register/inspect existing Data Roots and Site packages, refresh or sync metadata, and inspect their configurations and assignments. Browse fills the path field; use the explicit Register action when ready. Unregister removes the machine-local registration rather than deleting images. Portable root markers and site metadata allow relocation without changing scientific identity; update the registered location after moving data.

Configuration assignment has deterministic precedence: manual override, date/time range, relative path scope and site default. Ambiguity is an error, not a guessed choice. The assignments display/API is read-only. Verify the

resolved configuration explicitly before evaluating a dataset. CLI measurement continues to take an explicit `--site-config` for a reproducible run.

8. Measure, inspect and process folders

Set a writable output directory. Load the intended Site Configuration and a representative image, then measure one image first. Inspect found-octagon, calibration grid, virtual staff gauge, found-waterline and candidate-point overlays. Cursor coordinates and the calibrated ruler assist visual review. ROI changes and saved calibration settings must correspond to the source image.

Once representative results are acceptable, measure the folder. Browser folder processing shows progressive results and supports stop-after-current-image. The CLI supports one image, one folder or a folder of folders. The deterministic result remains official; preserve rejected/failed records and their reasons. A first Ctrl+C during CLI folder processing requests a cooperative stop; a second interrupt forces immediate interruption.

9. Quality, timestamps and output

Image-quality and linefind-quality are separate documented heuristic scores on 0.0-1.0 scales. Higher scores describe better measured evidence under each method, not calibrated probability or an uncertainty interval. Image statistics do not identify fog, ice or obstruction as physical causes. Availability, status, method/version, reason codes and diagnostic components are in exhaustive JSON. Quality scores assist evaluation without replacing acceptance checks.

Select either score for CSV when useful; the existing 20-column bound remains. JSON retains exhaustive scientific/diagnostic evidence. Timestamp-source choices and EXIF inspection help confirm time interpretation. Preserve the run manifest, measurement JSON/CSV, decision overlays, configuration and build identity with the source images or a stable reference to them.

Result schemas, units, datum, acceptance/rejection status and unavailable scores matter when comparing runs. Do not treat missing quality as zero or silently drop failed records. The field evaluation and CLI references below describe output choices and scientific provenance in more detail.

10. Troubleshooting and handoff

If a file or folder cannot be accessed, check the backend path, permissions and mounted drive; the browser computer may differ from the backend. If a native picker is unavailable, use GRIME3's own browser or manual entry. If saving is gray, check the loaded configuration and Operator. If an operation is busy, wait for it to finish before starting another.

For a compatibility rejection, inspect the native signature and intended camera configuration. For a target-size rejection, inspect original-image corner size and the configured minimum rather than changing display zoom. For suspicious stage results, inspect target/ROI geometry, units and datum before trusting quality scores or processing a large folder.

When reporting a problem, include the build ID/commit, platform, exact command or browser steps, configuration, representative image where permitted, and JSON/status output. Do not send passwords or unrelated personal files. Windows checks use native Windows, not WSL. Playback/hydrograph controls are deferred to RC10 and are not part of the RC9 measurement workflow.

Detailed references

The packaged CLI reference and the calibration, image compatibility, quality and native browsing chapters in this PDF provide the detailed contracts. The platform quick starts and release notes describe installation and release identity. Historical RC8 guides are retained for provenance; use RC9 guidance for the changed settings, compatibility, quality and browsing behavior.

Calibration reference

Calibration in GRIME 3

The **Minimum octagon size (pixels)** control sets the positive finite minimum for both the detected corner width and height in the original source image. The default is **100 x 100 source-image pixels**. Coordinates are measured before any display zoom or evidence-gathering upsampling. Equality is accepted. Force small-octagon detection mode does not change this setting.

The value is saved as `processing.minimum_octagon_span_pixels` in the canonical Site Configuration. Older configurations without the field use 100.0; loading them does not rewrite the file. Create, Load, Save, and Save As carry the setting. Changing and saving this processing setting preserves the existing calibration geometry and its audit history; the value used at calibration creation remains in calibration metadata. Creation and browser projection apply the saved limit. All other detection and calibration validity checks still apply.

Overview

Calibration is the process of mapping image pixel coordinates to real-world coordinates.

In GRIME 3, calibration enables:

- conversion of pixel locations to physical measurements

- estimation of water level from image data

- consistent measurement across time and conditions

The calibration process is based on:

- known geometry of a calibration target

- identification of corresponding pixel points in the image

- computation of a projective transformation (homography)

RC6 inner-boundary calibration contract

For large-image calibration, the desired octagon is the **inner boundary between the calibration-target face and the black border**.

GRIME3 does not treat the following as calibration geometry:

- the outer edge of the black border;

- black-border width;

- scene content outside the black border.

The production large-ROI calibration path is grayscale-based and does not depend on color segmentation.

A normal successful large-image detection path is:

`adaptive_large_roi_grayscale_inner_facets`

Reduced-resolution grayscale localization supplies only an approximate target center, scale, and orientation.

Full-resolution grayscale processing then searches for evidence belonging to the eight expected inner facets.

Multiple polarity-correct facet hypotheses can be retained when appropriate. Those hypotheses are evaluated jointly using global projective-octagon consistency.

The projective model is used to select a geometrically consistent set of physical facet lines. Final precision corners remain intersections of the measured adjacent facet lines.

A completely closed contour is not required. Interrupted or missing sections of a facet can be tolerated when sufficient distributed edge evidence remains to fit the physical inner boundary reliably.

If a credible inner octagon cannot be established, calibration fails safely.

Calibration-resolution requirement

The final detected calibration octagon must meet the configured minimum in both native-image dimensions. The browser reports:

- red when either dimension is below the chosen minimum;

- yellow when both dimensions meet the minimum but either is below 1.25 times it;

- no routine resolution message when both meet the advisory value.

With the default setting, these boundaries are 100 and 125 pixels. The advisory boundary never changes acceptance. The minimum is a size check, not a guarantee of valid geometry or sufficient edge evidence.

The CLI can calibrate an image using an existing Site Configuration's ROIs and physical inputs. See CLI reference (cli.md) for `grime3 calibrate` and its `--minimum-octagon-span-pixels` override.

Image compatibility (RC9)

New configurations record native decoded image representation before color conversion. Calibration templates, projection, and measurement require matching dimensions, channels/color model, dtype, effective bit depth, and EXIF orientation. Older configurations retain explicit dimension-only compatibility and can be migrated using the checksum-verified original image. Higher-bit-depth inputs are rejected by the current 8-bit scientific pipeline. See Image compatibility (image_compatibility.md) for the complete policy and command.

Calibration grid datum

The yellow horizontal calibration-grid family is anchored to the physical bottom facet of the octagon.

One yellow horizontal grid line passes through the bottom-left and bottom-right octagon corners. All remaining horizontal grid lines are spaced relative to that bottom-facet datum.

The yellow grid is not arbitrarily anchored to calibration-world Y = 0.

Water-level zero datum

Calibration-world Y and hydrologic water level are related but are not the same datum.

For:

- `bottomLeftWorldY`: calibrated world-Y coordinate of the bottom-left octagon corner;

- `bottomLeftOctagonCornerWaterlevel`: entered water level at that corner;

the stage relationship is:

```
stage = bottomLeftOctagonCornerWaterlevel
        + (worldY - bottomLeftWorldY)
```

Therefore stage 0.0 occurs at:

```
worldY = bottomLeftWorldY
         - bottomLeftOctagonCornerWaterlevel
```

GRIME3 projects this true stage-zero line in blue and labels it:

`0.0 water level`

Image compatibility reference

Image compatibility in RC9

New Site Configurations store a version-1 `source_image.signature` alongside the existing source dimensions and checksum. Site Configuration schema version 2 remains current. The signature contains width and height, channel count, single-channel/interleaved organization, decoded color model (`GRAY`, `BGR`, or `BGRA`), NumPy dtype, effective bit depth, and EXIF orientation.

GRIME3 inspects an unchanged decode before its normal color conversion. Width and height describe the EXIF-oriented coordinates used by the production color decoder. Effective bit depth means bits per sample in the unchanged decoded array (for example `uint8` = 8 and `uint16` = 16). It is not inferred from the brightest pixel, sensor specifications, or compressed file size. A packed 1-bit image expanded to `uint8` by the decoder has an effective decoded depth of 8. The recorded color model likewise describes the unchanged decoded representation, not the camera's sensor mosaic or an embedded ICC profile.

Calibration creation, CLI calibration using a template, current-image projection, and single-image/folder/tree measurement share the same gate. A signature-bearing configuration requires an exact signature match before its ROIs, geometry, or processing settings are applied. A grayscale image does not match a BGR configuration, and alpha-bearing BGRA does not match BGR. File extension, compression format, and image checksum may differ across measurement frames when the signature matches. This gate does not establish that camera pose or optics have remained fixed.

The existing scientific pipeline supports 8-bit grayscale, BGR, and BGRA inputs. After acceptance it uses the established 8-bit BGR color decoder. Higher-bit-depth and floating-point inputs are rejected rather than silently truncated. Supporting those inputs scientifically requires a separate, explicit conversion policy and a new calibration.

Older configurations

Schema-version-1 and version-2 configurations without a signature still load. Their saved width and height must match, and the image must have a supported 8-bit representation. If the current image's checksum matches the original calibration source, the decision is `legacy_source_verified`; otherwise it is `legacy_dimensions_only`, with an explicit warning that original photometry cannot be verified. A size mismatch is `legacy_dimension_mismatch` and is rejected. Loading, processing, and ordinary Save/Save As never invent a signature or silently rewrite the original file.

Recalibration creates a fully signed configuration. Alternatively, explicitly migrate using the **exact original source image**, verified by its saved SHA-256:

```
grime3 migrate-image-signature --site-config "/path/to/legacy.json" --source-image
  "/path/to/original.jpg" --output "/path/to/signed.json" --operator "Operator Name"
```

Use quoted Windows paths on Windows. Migration preserves existing identities, calibration/ROI geometry, settings, and component creation/save audit history; it updates file-save provenance. Version-1 configurations receive their normal generated identities on loading, which become persistent when saved as version

2. Existing destinations require `--overwrite`. If the original image is unavailable or its bytes have changed, migration refuses; recalibrate instead.

Result provenance

Projection responses and exhaustive per-image/measurement JSON contain an `image_compatibility` object with `accepted`, `status`, `message`, `expected_signature`, and `observed_signature`. Matching signed configurations report `signature_match`; mismatches report `signature_mismatch`. Unsupported inputs report `unsupported_image_format`. Decode failures have no observed signature and report `image_decode_failed`. A later detection or measurement failure retains the successful compatibility decision.

Rejected images produce no scientific stage value. Persistent per-image JSON also records rejection when enabled; in-memory workflows retain the same information without forcing a per-image file. Mixed accepted/rejected folder runs can continue and serialize their diagnostic summaries. The curated public measurement CSV columns and their existing meanings remain unchanged.

Measurement-quality reference

Measurement quality evidence (RC9)

GRIME3 records two separate descriptive scores, `image_quality` and `linefind_quality`, on a 0.0 to 1.0 scale. They assist evaluation and filtering. They are **not probabilities, accuracy estimates, or acceptance criteria**. They do not change the selected deterministic pass, calibrated water level, octagon acceptance, line acceptance, or failure status.

JSON contract

Every per-image measurement payload and measurement-summary record contains the scalar `image_quality` and `linefind_quality` fields and a `quality` block. An error-record excerpt is:

```
{
  "image_quality": null,
  "linefind_quality": null,
  "quality": {
    "schema_version": 1,
    "image_quality": {
      "score": null,
      "status": "unavailable",
      "method": "native_roi_evidence",
      "method_version": 1,
      "reason_codes": ["image_decode_failed"],
      "components": {},
      "diagnostics": {}
    },
    "linefind_quality": {
      "score": null,
      "status": "unavailable",
      "method": "waterline_search_evidence",
      "method_version": 1,
      "reason_codes": ["image_decode_failed"],
      "components": {},
      "diagnostics": {}
    },
    "selected_pass": "none"
  }
}
```

Actual `pass_quality` always contains separate `raw` and `preprocessed` records using the same evidence format. The top-level linefind score uses the pass already selected by the deterministic measurement algorithm. Its supporting selected points and raw/preprocessed diagnostics remain in the exhaustive JSON regardless of CSV selection.

`available` means the method's evidence was evaluated. `partial` means an image ROI or cross-pass comparison was unavailable; inspect the reasons and components before comparing that score with complete records. `unavailable` uses JSON `null`, never a fabricated zero. `no_line_found` uses a score of 0 when swath search evidence exists but no line was selected. A low score does not imply that measurement failed; a high score does not prove correctness.

Quality is computed per image, including errors. Decoding or compatibility failure produces unavailable scores. If ROI resolution fails after decoding, image evidence from the configured ROIs remains available. Successful ROI resolution replaces waterline image evidence with the resolved ROI. In-memory browser results use the same contract without forcing a persistent file.

Image method: `native_roi_evidence`, version 1

The method inspects the original grayscale image at native pixel resolution before either deterministic pass applies preprocessing. It evaluates the configured octagon search rectangle and the resolved waterline quadrilateral, falling back to configured waterline coordinates when resolution fails. Invalid or out-of-bounds polygons are explicitly unavailable, not silently clipped. A region needs at least 25 included pixels and 9 interior pixels.

Each region retains its coordinates, pixel count, grayscale 5th/50th/95th percentiles, fractions at or below 2 and at or above 253, and native-pixel Laplacian variance. The Laplacian uses `CV_64F`, `ksize=1`, and interior ROI pixels. Sampling uses the established 8-bit grayscale processing representation.

The three region components are:

- **Contrast:** `min(1, ((P95 - P05) / 255) / 0.10)`.

- **Detail:** `Laplacian_variance / (Laplacian_variance + 1.0)`.

- **Exposure:** `max(0, 1 - dark_fraction - bright_fraction)`.

The region score is the geometric mean of these three components. The image score is the minimum available region score, with all region scores retained. Missing regions make the aggregate `partial`; no available region makes it `unavailable`. The constants are explicit heuristic scales, not thresholds learned from a population of field images.

Reason codes include `low_contrast` below the contrast full-score scale, `low_detail_or_blur` below Laplacian variance 1.0, and `dark_clipping` or `bright_clipping` when the corresponding fraction exceeds 0.05. These describe measured evidence. They do not identify fog, haze, ice, obstruction, or another physical cause. Such effects may lower contrast/detail, but a validated classifier or independent evidence is needed to label their cause.

Detail depends on scene texture, scale, ROI placement, noise, and camera processing. Noise can increase Laplacian variance. This method does not claim a universal blur detector or a noise-corrected sharpness estimate. Compare scores with consistent site, representation, resolution, and ROI policy; inspect components rather than treating the aggregate as ground truth.

Line method: `waterline_search_evidence`, version 1

The method uses the actual selected line and point set from each existing deterministic pass. Support is counted from unique selected swath identities; fit residuals are recomputed against the final line from the selected points. The original reported residual evidence remains in the existing result JSON.

The composite is the product of these bounded components:

- **Support:** selected swaths / requested swaths.

- **Coverage:** selected ordinal swath span / full ordinal swath span.

- **Continuity:** `1 - longest_missing_swath_run / requested_swaths`; missing runs include leading and trailing gaps.

- **Residual:** `1 / (1 + RMS_perpendicular_residual_px / 2.0)`.

- **Edge response:** `mean_response / (mean_response + 5.0)`, where the response is in 8-bit grayscale units per profile sample. The coherent line provides its selected mean response; a rank-1 MAD fit uses the absolute strongest first-derivative responses of its accepted swaths.

- **Cross-pass stability**, when both line models exist: `1 / (1 + RMS_line_separation_px / 2.0)`. Separation is evaluated at the selected points' x coordinates in perpendicular selected-line pixel units.

Unavailable cross-pass geometry omits the stability component and marks the record `partial`; the omission is explicit in diagnostics. Missing mandatory point, swath, final-model, or response evidence makes the score unavailable. Residual or cross-pass RMS above 2 pixels, response below 5, incomplete support, coverage below 0.8, and missing swath gaps produce corresponding reason codes. Rejected lines that retain complete geometry can still have a descriptive score, with `line_rejected_by_measurement` explicitly recorded.

Stability here compares raw and preprocessed results for the same image; it is not temporal stability or an independent truth measurement. Blur can improve fit residuals while weakening the edge, so residual alone is not used as the quality score. Mild blur can improve geometric agreement enough to raise linefind quality while image quality declines; the two metrics measure different evidence. All constants and units are retained in diagnostics. Changes to these calculations or scales require a new method version.

Browser and CSV selection

The browser result panel shows both scores with availability status. Open **CSV Columns** and select **Image quality**, **Linefind quality**, or both. Selections use the existing browser preference persistence. Quality fields are optional and initially unselected; all required CSV fields remain present. The field catalog now offers 22 choices, while each emitted CSV remains limited to 20 selected columns. Deselect other optional fields if that limit is reached.

For `grime3 measure image`, `folder`, or `folders`, use either or both flags:

```
--csv-image-quality --csv-linefind-quality
```

These flags augment default columns unless an explicit `--csv-column` selection is supplied; with an explicit selection they augment that selection. The existing `--csv-column image_quality --csv-column linefind_quality` form also works. Missing scalar values are blank in CSV and `null` in JSON. The CSV checkpoint interval remains 10 images.

Reserved PNG metadata names

For future PNG result output, reserve `grime3.image_quality` and `grime3.linefind_quality` for the corresponding JSON number or literal `null`, and `grime3.quality` for the complete serialized quality block. This release does not add PNG output or modify image files.

Native browsing reference

Native file and folder browsing

On a local desktop, start `grime3 web --host 127.0.0.1 --open-browser` and use the exact browser URL printed by the launcher. The same public command works in Conda; the desktop packages invoke the same launcher.

Browse now selects image folders, measurement output folders, existing Data Root folders, Site package folders, and Site Configuration JSON files. Create and Save As select a JSON destination. Image folders and configurations load after selection; output, Data Root, and Site package selections fill their path fields without starting a run or registering a root. Native folder selection requires an existing directory. Manual output paths can still name a new directory for the normal output service to create. Manual path entry and Enter-to-load remain supported.

Windows uses the shell `IFileOpenDialog` for files/folders and `IFileSaveDialog` for JSON destinations. Linux uses `org.freedesktop.portal.FileChooser`, version 3 or later, through the installed GIO/GLib libraries. It needs a local graphical session, its session D-Bus, the desktop portal, and a FileChooser backend. The Legion inspection found version 3 with the GTK backend on XFCE/X11. No Qt, Tk, `gi`, `dbus`, or `dbus_next` dependency is added to GRIME3.

Cancel leaves the path unchanged and does not load, create, or save anything. Only one native picker may be active per server process, including across browser tabs. The browser disables measurement controls while selection is pending. Native dialogs are independent windows because GRIME3 does not own the web browser's window handle; check the taskbar if one appears behind it.

The dialog runs in a child process with its own Windows STA or Linux GLib context. Interaction is bounded to four minutes, with a short allowance for worker startup/cleanup. Linux explicitly closes a portal request on timeout; worker timeout terminates the child. Cancellation is distinct from provider failure. An unavailable provider, malformed/nonlocal URI, invalid selection, or worker failure opens the existing GRIME3 browser instead. JSON filters aid selection, and the backend still requires a `.json` path. Only filesystem paths accessible to the backend are returned.

Choosing a save destination does not write the configuration. Windows disables its test-file creation behavior; existing GRIME3 Create/Save validation and overwrite confirmation remain authoritative. The system save picker may also ask before replacing an existing file. Cancelling either confirmation leaves the existing file untouched.

Local access and fallback

Native browsing is enabled only by the production launcher when binding `127.0.0.1`, `localhost`, or `::1`. The browser must use that launcher's exact origin and port. Native endpoints require a loopback peer, the expected Host, matching Origin for selection, and the token issued by that server process. Forwarded/proxy and cross-site requests are refused. Uvicorn does not rewrite peer addresses from proxy headers. Tokens are memory-only, are not persisted in browser storage, and capability/selection responses use `Cache-Control: no-store`. These guards apply to native selection; they do not add general remote authentication to the existing GRIME3 APIs.

The default `create_app()` has no native desktop authority. Direct ASGI hosting, non-loopback bindings, Docker/containers, SSH, headless sessions, unsupported platforms, unavailable GIO/portal services, and Windows noninteractive/remote desktops retain the existing GRIME3 browser. No services are started by Linux capability probing. Manual entry and browser browsing remain available.

`GET /api/native-picker/capability` reports availability, provider, supported kinds, reason, and a session token only for an eligible local browser. `POST /api/native-picker/select` accepts `kind` (`folder`, `open_json`, or `save_json`), `initial_path`, `title`, and `token`. The response status is `selected`, `cancelled`, `busy`, or `fallback`; only `selected` contains a path. The private `_native-picker-worker` command is a stdin/stdout JSON protocol used by the server, not a user workflow. Source/Conda workers use Python's module entry point; frozen workers re-enter the same executable with that private command.

Validation gates

Automated tests cover local-origin/token restrictions, remote/proxy requests, strict request fields, single-dialog exclusion, worker errors/timeouts, paths with spaces and Spanish characters, save without file creation, Windows COM provider call/cleanup contracts, real GLib variant marshalling, and browser selection/cancellation/fallback/overwrite flows under Node or JavaScriptCore. Linux transport tests start a private D-Bus and simulated portal, never a GUI; they cover capability version, early Response signals, native URI decoding, cancellation, invalid results, and timeout/Close. They skip when system GIO or D-Bus is unavailable or the execution environment forbids private sockets.

Before releasing RC9, validate these on real local desktops and built packages:

1. Linux XFCE/X11 and, when available, Wayland: Browse each folder control, load JSON, create a new configuration, Save As, and cancel each kind. Check that selecting a Data Root does not register it automatically.

2. Native Windows, without WSL: repeat folder/open/save/cancel, use Spanish and spaced paths, cancel overwrite, and try two tabs. Confirm only one picker opens.

3. Frozen Windows and Linux builds: confirm the child command opens a native dialog without launching another server and that cancellation closes it.

4. Conda desktop: repeat both platform checks through `grime3 web`.

5. Docker, SSH/headless, a non-loopback server, and an unavailable Linux portal: confirm the existing browser opens and no native server-side window appears.

6. Close a picker, start another, and exercise normal measurement to confirm server lifecycle, scientific results, quality evidence, and CSV behavior.

Linux XFCE/X11 browser smoke checks passed for image/output/Data Root/Site folder selection, JSON loading, Save As cancellation, and repeated dialog use. Folder selection does not register a Data Root or Site automatically. Full Create/save/overwrite interaction checks, native Windows desktop checks, and frozen-package validation remain release gates; provider contracts do not substitute for those checks.

Platform API references:

- XDG FileChooser (https://flatpak.github.io/xdg-desktop-portal/docs/doc-org.freedesktop.portal.FileChooser.html)

- XDG Request (https://flatpak.github.io/xdg-desktop-portal/docs/doc-org.freedesktop.portal.Request.html)

- GIO signal subscription (https://docs.gtk.org/gio/method.DBusConnection.signal_subscribe.html)

- Windows IFileDialog (https://learn.microsoft.com/en-us/windows/win32/api/shobjidl_core/nn-shobjidl_core-ifiledialog)

- Windows dialog options (https://learn.microsoft.com/en-us/windows/win32/api/shobjidl_core/ne-shobjidl_core-_fileopendialogoptions)

Command-line reference

GRIME3 Command-Line Interface

GRIME3 provides one supported command-line entry point:

`grime3`

The command exposes the browser server and deterministic water-level measurement workflows. The CLI and browser use the same canonical Site Configuration and the same scientific measurement implementation.

Top-level commands

```
grime3 --version
grime3 --help
grime3 web --help
grime3 calibrate --help
grime3 measure --help
```

`grime3 measure` contains three measurement operations:

```
grime3 measure image
grime3 measure folder
grime3 measure folders
```

Pre-public batch command names are not part of the RC9 public interface.

Site Configuration

Measurement commands require one canonical GRIME3 Site Configuration JSON file:

`--site-config PATH`

The same Site Configuration can be reused for any number of independent single-image, folder, or folder-of-folders runs. A `measure folders` run applies the selected Site Configuration to each child folder in that run.

ROI geometry, calibration, datum, world units, grid settings, and other site-specific scientific state come from that file. Measurement commands do not expose separate legacy ROI, calibration, unit, datum, or small-octagon options.

Calibrate using an existing Site Configuration (RC9)

`grime3 calibrate` uses the existing configuration as an input template: octagon ROI, waterline ROI, physical facet length, units, datum, and detector mode. Select an image to which those ROI coordinates apply. The command runs the same calibration service as the browser and writes a new canonical configuration. It preserves the Site ID and creates a new Configuration ID.

```
grime3 calibrate --site-config "/path/to/existing.json" --image "/path/to/source.png"
  --output "/path/to/new.json" --operator "Operator Name" --minimum-octagon-span-pixels 60
```

On Windows, use the same options with quoted Windows paths.

Omit `--minimum-octagon-span-pixels` to use the template's saved value; old configurations without the setting use 100.0. An override must be a positive finite number and is applied to both corner spans in original image pixels. The effective value is recorded in the output configuration and calibration metadata. The template remains unchanged when a different output path is used. An existing output requires `--overwrite`.

This command supplies CLI calibration for an existing configuration. Initial ROI authoring and first configuration creation remain available in the browser. Measurement commands validate image compatibility before applying this file.

Migrate an older image signature (RC9)

New configurations record image dimensions, native decoded channels/color model, dtype, effective bit depth, and EXIF orientation. Those fields must match before projection or measurement. Older configurations retain dimension-only checks with an explicit provenance warning. The current scientific pipeline supports 8-bit grayscale, BGR, and BGRA; it rejects higher-bit-depth inputs instead of silently truncating them.

To add a signature without recalibrating, supply the exact original image:

```
grime3 migrate-image-signature --site-config "/path/to/legacy.json" --source-image
  "/path/to/original.jpg" --output "/path/to/signed.json" --operator "Operator Name"
```

Migration verifies the saved source checksum and preserves existing identities, ROI/calibration geometry, settings, and component audit history. File-save provenance records the operator. Existing outputs require `--overwrite`. Changed or unavailable original images require recalibration. See Image compatibility (image_compatibility.md) for signature definitions, legacy decisions, and result provenance.

Measure one image

Unix-like shell:

```
grime3 measure image \
  --image "/path/to/images/image 001.jpg" \
  --site-config "/path/to/site config.json" \
  --output-dir "/path/to/results"
```

Windows Command Prompt:

```
grime3 measure image ^
  --image "C:\Data\Images\image 001.jpg" ^
  --site-config "C:\Data\site config.json" ^
  --output-dir "C:\Data\results"
```

PowerShell:

```
grime3 measure image `
  --image "C:\Data\Images\image 001.jpg" `
  --site-config "C:\Data\site config.json" `
  --output-dir "C:\Data\results"
```

Use:

```
grime3 measure image --help
```

for the authoritative option list.

Measure one folder

Unix-like shell:

```
grime3 measure folder \
  --images "/path/to/images" \
  --site-config "/path/to/site_config.json" \
  --output-dir "/path/to/results"
```

Add `--recursive` when supported images should also be found below subdirectories.

Timestamp selection is controlled with:

```
--timestamp-source auto
--timestamp-source exif
--timestamp-source filename
--timestamp-source none
```

When filename timestamps are used, a `strptime`-compatible pattern can be supplied with:

```
--timestamp-filename-format FORMAT
```

Optional scientific CSV fields can be selected by repeating:

```
--csv-column FIELD
```

Required measurement columns are always retained by the CSV contract.

For image, folder, or folder-of-folders measurement, include either or both quality scores with `--csv-image-quality` and `--csv-linefind-quality`. These augment the default CSV columns, or an explicit `--csv-column` selection when supplied. The equivalent explicit field names are `image_quality` and `linefind_quality`. Both scores and their diagnostic evidence are always in result JSON, including unavailable/error records. CSV remains limited to 20 selected columns; unavailable scores are blank. Quality never changes measurement acceptance or the official water level. See Measurement quality evidence (measurement_quality.md) for formulas and limits.

Decision overlays can be written during the same scientific run:

```
--decision-overlays
--overlay-scale 50
```

The overlay scale accepts integer percentages from 10 through 100.

Measure a folder of folders

For a data root whose immediate child directories are independent image folders:

```
grime3 measure folders \
  --images-root "/path/to/data_root" \
  --site-config "/path/to/site_config.json" \
  --output-dir "/path/to/results"
```

Each non-empty immediate child folder is processed independently. Parent rollups combine the completed child results.

The command supports the corresponding timestamp, CSV-column, recursive, decision-overlay, and overlay-scale options shown by:

```
grime3 measure folders --help
```

Reusing one Site Configuration

A Site Configuration is not consumed or modified by a measurement run. Therefore one configuration can be reused for multiple datasets.

For example, a Unix-like shell can define the path once:

```
SITE_CONFIG="/data/site/site_config.json"

grime3 measure folder \
  --images "/data/run_01" \
  --site-config "$SITE_CONFIG" \
  --output-dir "/results/run_01"

grime3 measure folder \
  --images "/data/run_02" \
  --site-config "$SITE_CONFIG" \
  --output-dir "/results/run_02"
```

The CLI intentionally does not maintain hidden session state for a "currently loaded" configuration. Each invocation records the explicit configuration path in its provenance. Future Site Catalog and Data Root assignment functionality can provide higher-level persistent configuration selection without making individual measurement commands ambiguous.

Measurement outputs

Single-image and folder measurement use the same underlying output contract.

The exhaustive scientific information is retained in JSON. The primary scientific CSV is intentionally curated and contains no more than 20 user-selectable fields, while always retaining its required fields.

Typical folder outputs include:

```
waterline_measurement_summary.json
waterline_measurement_summary.csv
dual_pass_batch_summary.json
dual_pass_batch_summary.csv
run_manifest.json
```

Per-image exhaustive JSON is written using the image stem and the `_dual_pass_waterline.json` suffix.

When decision overlays are enabled, they are written beneath:

```
decision_overlays/
```

Folder-of-folders runs additionally produce child status information and combined parent rollups such as:

```
folder_child_run_summary.json
folder_child_run_summary.csv
all_waterline_measurement_summary.json
all_waterline_measurement_summary.csv
all_dual_pass_batch_summary.json
all_dual_pass_batch_summary.csv
```

CSV durability

During folder measurement, GRIME3 durably checkpoints the scientific CSV after every 10 completed images. RC9 keeps this interval fixed.

At normal completion, any remaining partial checkpoint group is also flushed and the final CSV is written.

Ctrl+C behavior

For folder and folder-of-folders measurement, the first Ctrl+C requests a cooperative stop.

GRIME3 finishes the image currently being processed, finalizes completed outputs, and exits with a nonzero interrupted status.

For single-image measurement, a first Ctrl+C likewise requests completion of the current image followed by normal output finalization and an interrupted exit status.

A second Ctrl+C forces immediate interruption. The incremental CSV writer is closed through the shared folder execution `finally` path, preserving rows that were already completed, although a forced second interruption can prevent later summary or manifest finalization.

Exit behavior

A normal successful measurement exits with status 0.

Invalid input, processing failure, or a completed run containing scientific processing errors exits nonzero.

A cooperative or forced Ctrl+C interruption exits with status 130.

Programs and shell scripts should use the process exit status rather than parsing human-readable status messages.

Paths with spaces

All supported CLI path arguments use normal filesystem path handling and are tested with paths containing spaces. Quote such paths according to the active shell.

Browser server

The same installed package also starts the browser interface:

```
grime3 web --help
```

The browser and CLI are interfaces over the same GRIME3 scientific implementation; they are not separate measurement engines.

Document font license

Bitstream Vera Fonts Copyright

The fonts have a generous copyright, allowing derivative works (as long as "Bitstream" or "Vera" are not in the names), and full redistribution (so long as they are not *sold* by themselves). They can be be bundled, redistributed and sold with any software.

The fonts are distributed under the following copyright:

Copyright =========

Copyright (c) 2003 by Bitstream, Inc. All Rights Reserved. Bitstream Vera is a trademark of Bitstream, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of the fonts accompanying this license ("Fonts") and associated documentation files (the "Font Software"), to reproduce and distribute the Font Software, including without limitation the rights to use, copy, merge, publish, distribute, and/or sell copies of the Font Software, and to permit persons to whom the Font Software is furnished to do so, subject to the following conditions:

The above copyright and trademark notices and this permission notice shall be included in all copies of one or more of the Font Software typefaces.

The Font Software may be modified, altered, or added to, and in particular the designs of glyphs or characters in the Fonts may be modified and additional glyphs or characters may be added to the Fonts, only if the fonts are renamed to names not containing either the words "Bitstream" or the word "Vera".

This License becomes null and void to the extent applicable to Fonts or Font Software that has been modified and is distributed under the "Bitstream Vera" names.

The Font Software may be sold as part of a larger software package but no copy of one or more of the Font Software typefaces may be sold by itself.

THE FONT SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF COPYRIGHT, PATENT, TRADEMARK, OR OTHER RIGHT. IN NO EVENT SHALL BITSTREAM OR THE GNOME FOUNDATION BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, INCLUDING ANY GENERAL, SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF THE USE OR INABILITY TO USE THE FONT SOFTWARE OR FROM OTHER DEALINGS IN THE FONT SOFTWARE.

Except as contained in this notice, the names of Gnome, the Gnome Foundation, and Bitstream Inc., shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Font Software without prior written authorization from the Gnome Foundation or Bitstream Inc., respectively. For further information, contact: fonts at gnome dot org.