HOASYS

Holographie Adaptative et Systèmes

SLM-H06

User Manual — software version 0.3.5

July 2026

HOASYS SAS — www.hoasys.fr

Contents

1 Introduction

SLM-H06 displays computer-generated patterns on a spatial light modulator (SLM) or projector connected to address an OASLM, while showing a live, always-faithful preview on the main screen. It is designed for optics laboratory use:

- the pattern shown on the SLM is never disturbed — the mouse pointer is fenced off the SLM screen and no other window can appear above the output;
- everything is displayed in grayscale: on a phase SLM the gray level of a pixel encodes the phase delay, so color would be meaningless;
- patterns can be static images, looping videos, or live frames pushed from Python at video rate;
- geometric transforms (zoom, rotation, shift, mirror, 1:1 pixel mapping), gray-level operations (gain, inversion) and a circular iris are applied in real time from a built-in command console;
- elliptical or rectangular regions of interest, filled at full white (gray level 255), are drawn, moved, resized and rotated with the mouse directly on the preview;
- a per-device calibration curve (LUT) can be applied to the SLM output without affecting the preview; it is loaded automatically at startup.

SLM-H06 runs on Linux (X11), Windows 10 or later, and macOS. The user interface is identical on the three platforms.

2 Installation

2.1 Windows

Run the installer `SLM-H06-<version>-setup.exe` and follow the wizard. It installs the application (with `ffmpeg` for video playback and the UI fonts included), creates a Start Menu entry and an optional desktop icon, and registers an uninstaller.

Alternatively, the portable `SLM-H06-<version>-win64.zip` needs no installation: unzip it anywhere and run `SLM-H06.exe`.

2.2 macOS

Open `SLM-H06-<version>.dmg` and drag **SLM-H06** to the **Applications** shortcut. The application is not notarized: on first launch, right-click the app and choose *Open*.

Two optional steps:

- **Pointer fence** — to let SLM-H06 keep the mouse off the SLM screen, grant the Accessibility permission when prompted (System Settings → Privacy & Security → Accessibility → enable SLM-H06), then relaunch. Without it the application works normally; only the pointer fence is disabled.
- **Video playback** — install `ffmpeg`, for instance with Homebrew: `brew install ffmpeg`. Images and the Python interface work without it.

2.3 Linux

On Arch Linux, install the package built by `packaging/build-arch.sh` (`sudo pacman -U slm-h06-<version>-1-x86_64.pkg.tar.zst`). On any other distribution, use the AppImage: make it executable (`chmod +x SLM-H06-<version>-x86_64.AppImage`) and run it. Install `ffmpeg` from your distribution for video playback.

3 Quick start

1. Connect the SLM (or projector) as a **second monitor**, in *extended desktop* mode (not mirrored), at its native resolution.
2. Start SLM-H06. The SLM shows a built-in test pattern; the preview window opens on your main screen.
3. Load a pattern: **drag and drop** an image or video file onto the preview window (or type `load` in the console, or pass a file on the command line).

4. Adjust with console commands (`zoom2`, `rotate90`, `aspect`, …) — see the reference below.
5. The **ON/OFF button** in the preview footer toggles the projection: when OFF the SLM shows black while the preview keeps displaying the pattern.

To quit, press **Esc** or close either window. Pressing **Delete** clears the current media (the SLM shows black).

By default SLM-H06 projects on the first non-primary monitor. With more than two monitors, choose the output with `--screen` (see below).

4 Command-line options

SLM-H06 can be started from a terminal (on Windows, `SLM-H06.exe` prints to the terminal it was started from):

```
SLM-H06 [image-or-video] [options]

  --screen <name|index>   monitor to project on (default: first
                          non-primary). Use --list to see names.
  --port <n>              TCP port of the Python live interface
                          (default 5106, 0 disables it)
  --lut <file>            CSV calibration LUT applied to the SLM output
                          (without --lut, a LUTC.csv from the current
                          directory, the executable's directory or the
                          package data directory is loaded automatically)
  --list                  list detected monitors and exit
  --help                  show the built-in help
```

Example — list monitors, then project a pattern on a chosen one:

```
$ SLM-H06 --list
[0] eDP1   1920x1080+0+0  (primary)
[1] HDMI1  1920x1080+1920+0
$ SLM-H06 hologram.png --screen HDMI1
```

5 The preview window

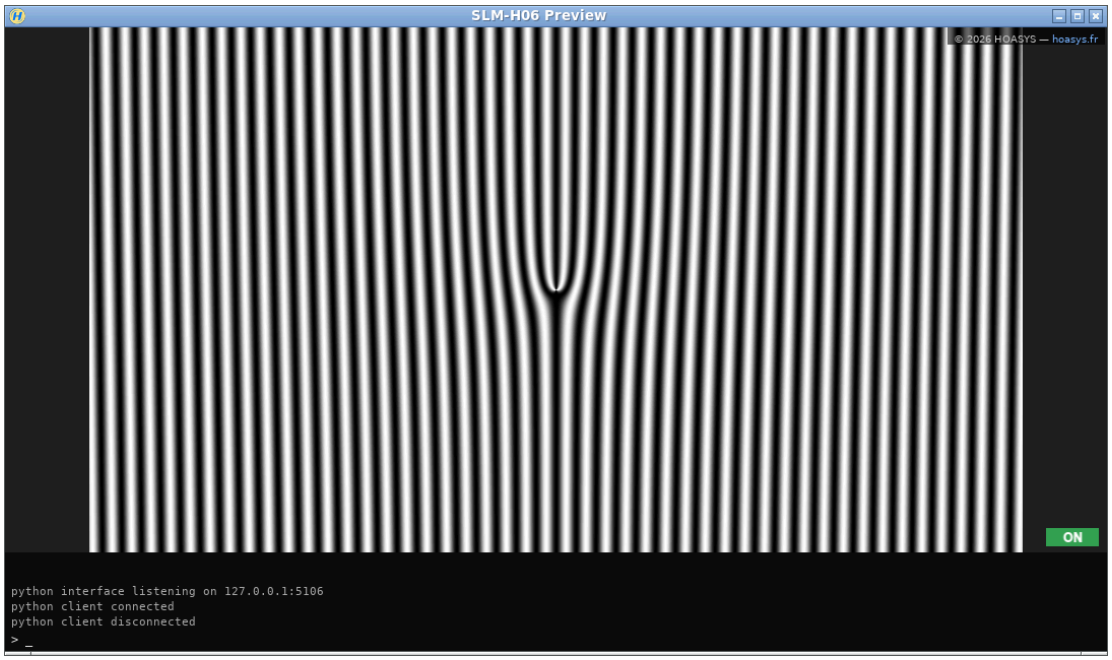

Figure 1: The preview window: live view of the SLM content (here a fork grating pushed through the Python interface), the ON/OFF projection toggle, and the command console.

The preview window shows the SLM content letterboxed at the SLM's aspect ratio, with three interactive elements:

- **ON/OFF button** (footer): toggles the projection. OFF projects black on the SLM; the preview keeps showing the pattern so you can prepare it.
- **Command console** (bottom): type commands, browse history with the ↑/↓ keys. The last eight result lines are shown above the input line.
- **hoasys.fr link** (top right): opens the HOASYS website.

Files can be dropped onto the window from any file manager. If a drop cannot be used (remote file, unsupported data), the reason is printed in the console.

6 Console command reference

Commands are case-insensitive. `<v>` is a number, or a fraction such as `1/3`.

Command	Effect
`zoom<v>`	zoom by `<v>` on both axes (`zoom10`, `zoom1/3`)
`zoomx<v>`, `zoomy<v>`	zoom along one axis only
`rotate<v>`	rotate clockwise by `<v>` degrees (`rotate-90` = counterclockwise)
`move(x,y)`	shift the pattern center by (x, y) SLM pixels
`move`	move the pattern with the mouse over the preview; click to fix it
`flipx`, `flipy`	mirror horizontally / vertically (toggles)
`aspect`	show the image at its native size — one image pixel per SLM pixel, centered — instead of stretching it to the SLM (toggle)
`<v>*ans`	multiply the gray levels by `<v>` (`0.9*ans`); factors accumulate, values above 1 saturate at white
`inv`	invert the gray levels (toggle)
`iris(<v>)`	show the pattern only inside a centered circle of radius `<v>` SLM pixels; `iris(0)` removes it
`roi(c)`, `roi(r)`	define an elliptical / rectangular region of interest with the mouse, filled with white (see *Regions of interest* below)
`roi(rot,<v>)`	rotate the last ROI to exactly `<v>` degrees clockwise
`roi(off)`	remove all ROIs
`lut`	load a CSV calibration LUT, applied to the **SLM output only** (see next section); `lut(off)` removes it
`ans`	print the current state (gain, zoom, rotation, offset, flips, aspect, inversion, iris, ROIs)
`load`	open a file chooser to load an image or video
`save(name)`	save the current view as `name.png`, next to the opened file
`init`	reset every transform (zoom ×1, rotation 0°, gain 1, …)
`clear`	delete everything: the media (the SLM shows black), all transforms and ROIs; the calibration LUT is kept
`help`	one-screen summary of the commands
`quit`	close the application (`exit` also works)

All transforms combine: for instance `aspect` followed by `zoom2` shows exactly 2 SLM pixels per image pixel; `move` then shifts that mapping.

7 Regions of interest (ROI)

A ROI is an elliptical or rectangular area filled at full white (gray level 255), drawn on top of the pattern — for instance to activate a chosen area of an OASLM. ROIs are defined and edited entirely with the mouse on the preview window, and appear identically on the SLM in real time.

- **Create** — type `roi(c)` (ellipse) or `roi(r)` (rectangle), then press the left button on the preview and drag: the shape stretches between the press point and the mouse; release to finalize. The console prints the center and size in SLM pixels.
- **Move** — press inside an existing ROI and drag it.
- **Resize** — press near one of its four corners and drag; the opposite corner stays fixed.
- **Rotate** — hold **Shift** and drag anywhere on the ROI: it rotates around its center following the mouse. For an exact angle, use `roi(rot,<v>)` (degrees clockwise, applied to the last ROI).
- **Delete** — right-click on a ROI removes it; `roi(off)` removes them all; `clear` and `init` also remove them.

The mouse cursor changes over a ROI to show what a drag would do (move, resize or rotate). Several ROIs can coexist; overlapping grabs pick the most recently created one.

ROIs live in SLM coordinates: they are not affected by `zoom`, `move`, `rotate` or the other image transforms, and the `iris` clips them like the rest of the pattern. Their white is the device's full level 255, independent of `inv`, `*ans` gain and the calibration LUT.

8 Images, videos and grayscale

Images (PNG, JPEG, BMP, TGA, and other common formats) are converted to grayscale on loading using the Rec. 601 luma formula (Y = 0.299 R + 0.587 G + 0.114 B).

Videos — any format readable by `ffmpeg` — are decoded by an external `ffmpeg` process, converted to grayscale with the same convention, paced at the frame rate declared by the file, and looped forever. `ffmpeg` and `ffprobe` must be available: they are bundled with the Windows package; on Linux and macOS install them with your package manager.

By default the pattern is stretched to fill the SLM exactly. Use `aspect` for an unstretched, centered, pixel-exact (1:1) display — recommended when the pattern was computed at the SLM's native resolution.

9 Gray-level calibration (LUT)

Real SLMs rarely have a linear phase response. SLM-H06 can apply a per-device lookup table that remaps each of the 256 gray levels **on the SLM output only** — the preview and `save()` deliberately keep showing the uncalibrated pattern.

Type `lut` and choose a CSV file. The format is flexible:

- one value per line, or `input,output` pairs (the last number of each line is used); a header line is ignored;
- values either in 0–255 or normalized 0–1;
- any number of entries (at least 2): the curve is linearly resampled to 256 levels.

The order of gray-level operations on the SLM is: inversion (`inv`), then gain (`*ans`), then the LUT. `lut(off)` removes the calibration. An example file inverting the levels, `lut_invert.csv`, ships with the application.

Automatic loading. At startup, SLM-H06 looks for a file named `LUTC.csv` — in the directory it was started from, next to the executable, and in the application's data directory — and loads the first one found as the calibration (the console reports it). Every installation package ships the factory calibration `LUTC.csv` of the SLM-H06 head, so a fresh install is calibrated out of the box. To recalibrate, replace that file or place an updated `LUTC.csv` in the launch directory (which takes precedence); `--lut <file>` on the command line overrides the search entirely.

10 Python real-time interface

SLM-H06 listens on `127.0.0.1:5106` (loopback only; change the port with `--port`, disable with `--port 0`). The bundled Python module `slm_h06.py` pushes NumPy arrays or matplotlib figures; each frame replaces the displayed pattern immediately, at video rate:

```
import numpy as np
from slm_h06 import SLM

with SLM() as slm:                          # connects to 127.0.0.1:5106
    image = np.zeros((1080, 1920))          # work at the SLM resolution
    image[400:600, 800:1100] = 1.0
    slm.show(image)
```

`SLM.show(array)` accepts float arrays (clipped to [0, 1]) or integer arrays (clipped to [0, 255]); RGB(A) arrays are converted to grayscale (Rec. 601). `SLM.show_figure(fig)` renders a matplotlib figure — call it again after each modification to animate. Console transforms (`zoom`, `iris`, ROIs, `*ans`, the LUT, …) apply on top of live frames as well.

Two commented examples ship with the application: `example_basics.py` (step-by-step tour) and `example_live.py` (a drifting grating animated at 30 frames per second).

The module requires only `numpy` (and `matplotlib` for `show_figure`). The wire protocol is documented in the appendix for use from other languages.

11 Protecting the SLM display

Any pixel change on the SLM screen disturbs the optical experiment, so SLM-H06 defends it:

- the **mouse pointer is fenced out** of the SLM monitor while the application runs (if the pointer starts there, it is moved back to the main screen);
- the output window asks the system to stay **above all other windows**;
- the pointer fence is released automatically when the application exits.

Platform notes: on macOS the fence requires the Accessibility permission (section *Installation*); on Windows and Linux it works out of the box. If the fence cannot be installed, a warning is printed and the application runs normally otherwise.

12 Troubleshooting

The SLM shows the test pattern but my file does not load. Check the console message. For videos, `ffmpeg` must be installed and reachable (on macOS, `brew install ffmpeg`).

"warning: no non-primary monitor found — using the primary". The SLM is not seen as a second monitor: check the cable and, in the system display settings, select *extend* (not *mirror*) at the SLM's native resolution.

The projection is on the wrong monitor. Run `SLM-H06 --list` and pick the output with `--screen <name>`.

"could not listen on 127.0.0.1:5106". Another program (or another SLM-H06 instance) uses the port. Start with `--port <other>` and pass `port=` to the Python `SLM()` constructor.

The mouse can still enter the SLM screen (macOS). Grant the Accessibility permission to SLM-H06 (System Settings ⟶ Privacy & Security ⟶ Accessibility), then relaunch the application.

Drag-and-drop does nothing (Linux). Make sure the file comes from a local file manager supporting XDND. The `load` command is an alternative (requires `zenity` or `kdialog`).

The preview text is missing. No usable font was found; a warning on the terminal lists the details. The packaged builds bundle their own fonts, so this only affects source builds on unusual systems.

13 Appendix

13.1 Live-frame protocol

TCP, loopback (`127.0.0.1`), default port 5106. Each frame is:

Field	Size	Content
magic	4 bytes	ASCII `"SLM1"`
width	4 bytes	unsigned 32-bit, little-endian
height	4 bytes	unsigned 32-bit, little-endian
pixels	width × height bytes	8-bit gray levels, row-major, top-left origin

Frames are displayed as they arrive; if the sender is faster than the display, only the most recent frame is kept. Maximum dimension: 16384.

13.2 Bundled third-party software

The packaged builds include the DejaVu fonts (Bitstream Vera license) and, on Windows, FFmpeg (GPL — see `LICENSE-ffmpeg.txt` in the installation directory). See the license files distributed with each package.

13.3 Support

For questions and support:

HOASYS SAS
120 route des macarons
06560 Valbonne, France

E-mail: **contact@hoasys.fr**
Web: **www.hoasys.fr**