GS Platform (Localhost UI) — Step-by-step User Guide

This guide explains how to run and use your local Gaussian Splatting platform (FastAPI + H
WSL Ubuntu with your `gs` **conda env**.

---

1) What this platform does

You use a web UI (served locally via Uvicorn/FastAPI) to run your existing `gs_cli.py` pipeline:

- Choose a **Workspace** folder (project output folder)
- Choose input as either:
- **Video file** (the platform extracts frames), OR
- **Images directory** (already extracted frames)
- Choose COLMAP settings (matcher, overlap, camera model, GPU usage, etc.)
- Choose training settings (iterations, save/checkpoint iterations, resume)
- Click **Run** → platform creates a **Job**
- Open the Job page to see:
- Live log output
- The exact command used
- Status (running/done/error) and PID

---

2) Folder layout (expected)

Typical project layout under a workspace you pick (example):

- `WORKSPACE/`
- `images/` *(created if using video)*
- `colmap/` *(created during COLMAP step)*
- `undistorted/` *(created during undistort)*
- `gs_out/` *(training outputs)*
    - `point_cloud/iteration_XXXX/point_cloud.ply`
    - checkpoints (if enabled)

The platform **does not** require you to create this manually; `gs_cli.py run` creates what it needs.

---

3) Requirements (one-time setup)

A) You must have these working already

- WSL Ubuntu
- Conda environment: `gs`
- Your gaussian-splatting repo path, e.g.: `/home/aditya/gaussian-splatting`

- `gs_cli.py` inside that repo
- COLMAP installed (CUDA build recommended)

B) Platform files

Your platform directory looks like:

- `~/gs_platform/app/`
- `server.py`
- `templates/`
  - `index.html`
  - `_browse_modal.html`
  - `_jobs.html`
  - `job.html`
- `jobs_db/` *(created automatically, stores job metadata + logs)*

---

4) Start the platform (every time)

1) Open a terminal and enter WSL.

2) Activate your environment:

```
conda activate gs
```

3) Go to the platform app folder:

```
cd ~/gs_platform/app
```

4) Start the server:

```
uvicorn server:app --host 0.0.0.0 --port 7860
```

5) Open in your browser (Windows browser is fine):

- `http://127.0.0.1:7860`

  If the page doesn't load, confirm the terminal shows: "Uvicorn running on ...: 7860 ".

---

5) Understanding the UI

Main page sections

- **Workspace**: where project outputs go
- **Input**:

- **Video path** OR **Images dir** (choose only one)
- **Extraction** settings (only used if you provide video)
- **COLMAP** settings (matcher/camera model/GPU)
- **Training** settings (iterations/save/checkpoints/resume)
- **Jobs list**: all jobs created from this platform

The Browse button

The Browse button opens a modal file browser that lists folders/files from WSL paths.

- Workspace browsing usually starts at something like `/mnt/c/gs_data/projects`
- Video browsing starts at `/mnt/c/Users`

You can: - Click folders to navigate - Search (especially useful for video selection) - Select a
apply it back to the form - Create a new folder (where enabled)

---

6) Step-by-step: Run a new project from a Video

Step 1 — Choose workspace

1) Click **Browse** next to Workspace. 2) Navigate to the parent folder where you keep projects
`/mnt/c/gs_data/projects` 3) If needed, Click **New folder** and create a project name. 4)
Select the project folder and confirm.

✅ Example: - Workspace: `/mnt/c/gs_data/projects/horse auto`

Step 2 — Choose video

1) In **Video path**, click **Browse**. 2) Navigate to your video locat `/mnt/c/Users/Guest_/
Downloads` 3) Use the search to filter by filename if needed. 4) Click the video file (m
etc.) to select it.

✅ Example: - Video: `/mnt/c/Users/Guest_/Downloads/Auditorium.mp4`

> Important: If you select Video, leave Images dir empty.

Step 3 — Set extraction parameters

- **FPS**: how many frames per second to extract
- Typical: `1–3`
- Higher FPS = more frames = slower COLMAP + training
- **JPG quality**:
- `1` is best quality but larger size
- Common: `1–3`

Step 4 — COLMAP matcher choice

- **Sequential matcher** (recommended for video):
- Best for frames in time order

- Uses `overlap` to match nearby frames
- **Exhaustive matcher**:
- Tries matching every image to every other
- Great for small datasets, expensive for large videos
- **Vocab/loop detection**:
- Helps when camera revisits areas (loops)
- Useful in rooms/360 walkthroughs

Settings: **Overlap**: - For sequential: `10-30` - More overlap = more matching = more compute **Loop detection**: - Enable if you walk around and return to similar views

Step 5 — Camera model

You must pick a camera model because COLMAP needs an intrinsics model.

Common choices: `SIMPLE_RADIAL` (recommended default) - Works well for many phone/action cams - 1 radial distortion param `PINHOLE` - No distortion (only if lens is very clean/no wic angle) - `OPENCV` - More distortion parameters (helpful for very wide lenses)

Single camera - Enable if the video is from a single physical camera and intrinsics should For a single phone video: ✅ enable.

Step 6 — GPU options

- Enable **use_gpu** for feature extraction / matching if your COLMAP build supports it.

Step 7 — Training settings

- **Iterations**: total training steps
- Typical: `30k-60k`
- **Save iters**: which iterations to save point clouds
- Example: `7000 10000 15000 20000 30000`
- **Checkpoint iters**: which iterations to save checkpoints
- Example: `10000 30000 60000`

Step 8 — Run

Click **Run**.

You will be redirected to `/job/<job_id>`.

---

7) Step-by-step: Run a project from an Images directory

Use this when you **already extracted frames** yourself.

1) Workspace: choose a project folder. 2) Leave **Video path empty** 3) Set **Images dir** to your frames folder: - Exam `/mnt/c/gs_data/projects/bultt/images` 4) COLMAP + training settings same as usual. 5) Click **Run**.

---

8) Monitoring jobs

Jobs list

The homepage shows a list of j `queued / running / done / error / stopped` - pid (when running) - workspace path

Click **Open** to see the job details.

Job page

The job page shows: - state + pid + return code - workspace/video/images info - exact co **Live log** updating every few seconds

---

9) Resume training (continue from 30k to 60k)

This works if: - your training saved checkpoints (Checkpoint iters) - and you re-run with `iterations` and **resume enabled**

In the UI: 1) Select the **workspace** 2) Set **iterations** to `60000` 3) Enable **resume** 4) Set checkpoint iters to include new ones, e.g. `40000 50000 60000` 5) Click **Run**

The job should resume from the latest saved checkpoint.

---

10) Viewer: How to open the result in SIBR

You mentioned your SIBR viewer directory on Windows: - `C:\Users\Guest_\OneDrive\Desktop\viewers\bin`

Typical viewer command (Windows PowerShell example):

```
cd "C:\Users\Guest_\OneDrive\Desktop\viewers\bin"
.\SIBR_gaussianViewer_app.exe --model-path "C:\gs_data\projects\horse auto\gs_out" --iteration 60000
```

Notes: - The model output lives in WSL `/mnt/c/...` which corresponds to Windows `C:\...`. - You choose an iteration that exists in `gs_out/point_cloud/iteration_XXXX`.

---

11) Common mistakes and what happens

A) You set both Video and Images dir

The platform rejects it (400) because it’s ambiguous.

Fix: clear one input.

B) Wrong workspace path

- Job runs but fails when `gs_cli.py` tries to create folders or write outputs.

Fix: choose a writable folder under `/mnt/c/...` (Windows drive) or your Linux home.

C) Wrong gs repo path

- The platform errors immediately because it cannot find `gs_cli.py`.

Fix: set `gs_repo` to the actual repo folder, e.g. `/home/aditya/gaussian-splatting`.

D) Video path not found

- `ffmpeg` step fails.

Fix: ensure the file exists in WSL path (`/mnt/c/...`) and you can `ls` it.

E) COLMAP fails

- Often due to matcher choice, insufficient overlap, or tough video.

Fixes: - Use sequential matcher for video - Increase overlap (e.g. `20` → `30`) - Lower
reduce number of frames

---

12) How to recover from a broken job

If a job got corrupted or you changed code mid-run:

1) Stop it from the job page. 2) Delete its job folder:

```
rm -rf ~/gs_platform/app/jobs_db/<job_id>
```

3) Re-run from UI.

---

13) Recommended presets

A) Typical phone video (walk around object)

- FPS: `2`
- JPG quality: `1`
- Matcher: `sequential`
- Overlap: `20–30`
- Loop detection: Off (turn on if you circle back)
- Camera model: `SIMPLE_RADIAL`

- Single camera: On
- Iterations: `30000` then resume to `60000`
- Save iters: `7000 10000 15000 20000 30000`
- Checkpoint iters: `10000 30000 60000`

B) Very long video / too many frames

- FPS: `1`
- Overlap: `15–20`
- Iterations: `30000`

---

14) Quick “happy path” checklist

1) Start server 2) Workspace selected ✅ 3) Video OR Images dir selected ✅ 4) Sequ
matcher + overlap set ✅ 5) SIMPLE_RADIAL + single camera ✅ 6) Iterations + save/check
set ✅ 7) Run → Job page opens ✅ 8) Watch live log ✅ 9) Open output in SIBR viewer ✅

If you want, I can add a **“Validate Inputs”** button in the UI (checks that paths exist, and warns if
video+images set) before you press Run.