OPENLANDER

AI-assisted autonomous landing assistant

SIMPLIFIED USER MANUAL

OpenLander combines visual perception, terrain scoring, reachability analysis, candidate tracking, and a mission-state timeline in one judge-friendly mission-control interface.

Quick start

Launch the packaged application or run the documented Python source.

System requirements

- **Computer:** Apple Silicon Mac (M1 or newer).
- **Operating system:** macOS 13 or later.
- **Display:** A modern browser window; 1440 × 900 or larger is recommended.
- **Storage:** Keep the application and the supplied processed_runs folder together.

Launch the standalone app

1. Open the CPWC2026-0021 submission folder in Finder.
2. Double-click CPWC2026-0021.app. The application starts a private local service and opens OpenLander Mission Control in the default browser.
3. Choose Analyze Raw Scenario for a new image sequence, or Open Processed Run for immediate cached replay.

> **FIRST LAUNCH**
>
> macOS may take longer while it verifies the self-contained app. If a security warning blocks the first launch, Control-click the app in Finder, choose Open, then confirm Open (no Terminal is required). Keep the app beside processed_runs so the examples are discovered automatically.

Run from source

The complete runtime source and dependency list are included in CPWC2026-0021_Source.md. From a folder containing app.py, engine.py, requirements.txt, model assets, and processed_runs:

```
python3.11 -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
python app.py
```

> **LOCAL OPERATION**
>
> The interface is served only on the local machine. Cached processes runs open without AI inference or an internet connection.

Two usage modes

Analysis and replay remain separate so judges can inspect results immediately.

Mode	What it does
Analyze Raw Scenario	Runs depth estimation, object detection, semantic perception, optical-flow drift, landing scoring, candidate selection, and the decision state machine. It saves a new self-contained processed run.
Open Processed Run	Loads cached JSON and rendered frames only. It is immediate, repeatable, and performs no AI inference.

Raw input folder format

Select the scenario folder itself. The minimum structure is:

```
scenario_name/
├── scenario.json
├── metadata.csv
└── frames/
    ├── 000000.png
    ├── 000001.png
    └── …
```

- **scenario.json:** scenario title, mission mode, vehicle size, and provenance.
- **metadata.csv:** one row per frame with frame ID, filename, timestamp, altitude, velocity, and camera telemetry where available.
- **frames/:** ordered RGB PNG or JPEG images at a consistent resolution.

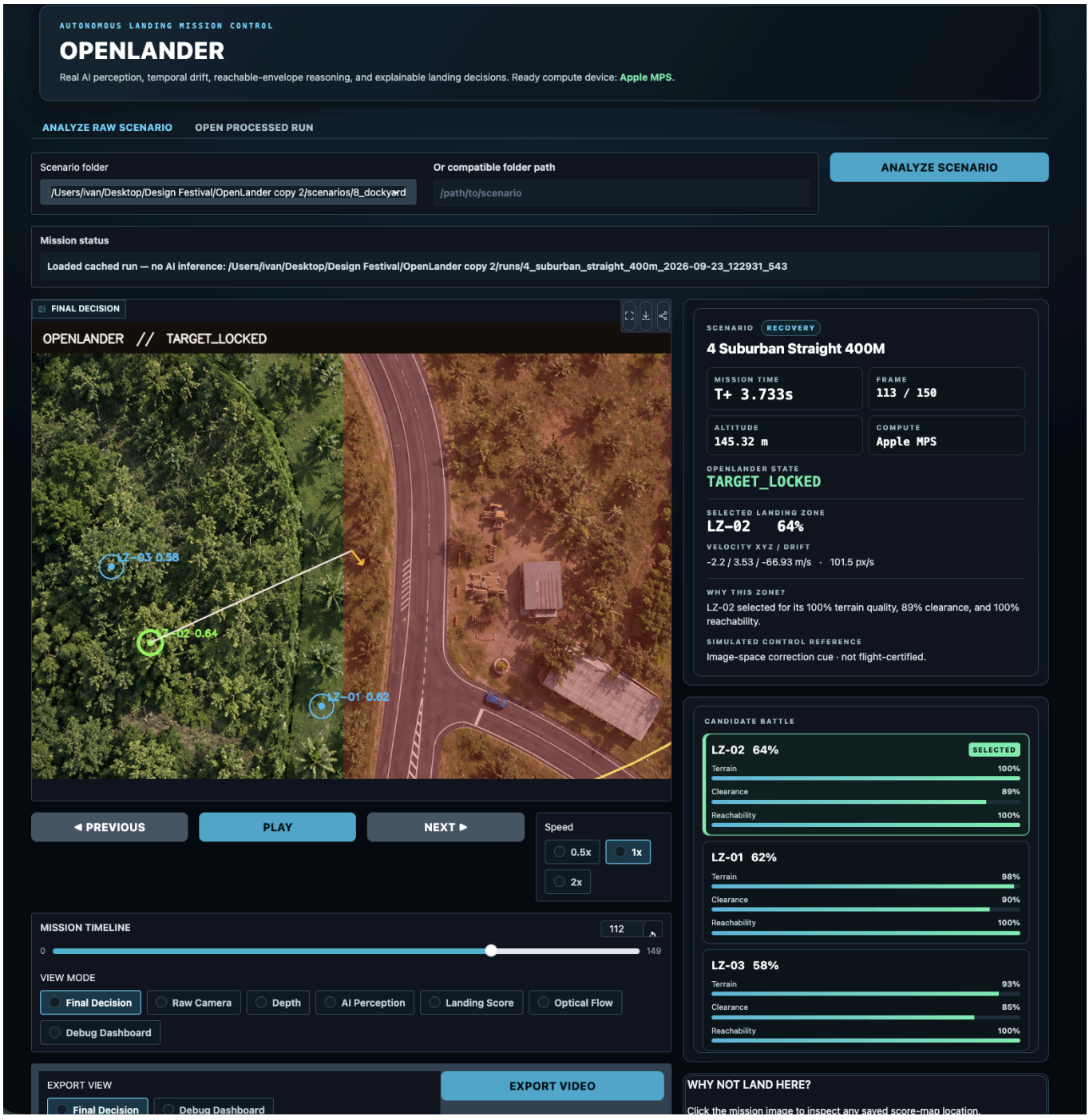
Processed run contents

Each run contains run.json, result.json, summary.txt, copied raw inputs, and cached visualization folders such as annotated, depth, perception, heatmap, flow, and dashboard. Exported videos are saved back into that run folder.

> **TRACEABILITY**
>
> Opening a processed run never changes its decision record. The replay is driven by saved result.json data and cached frames.

Mission-control interface

A compact, high-contrast layout keeps decisions, evidence, and controls visible together.

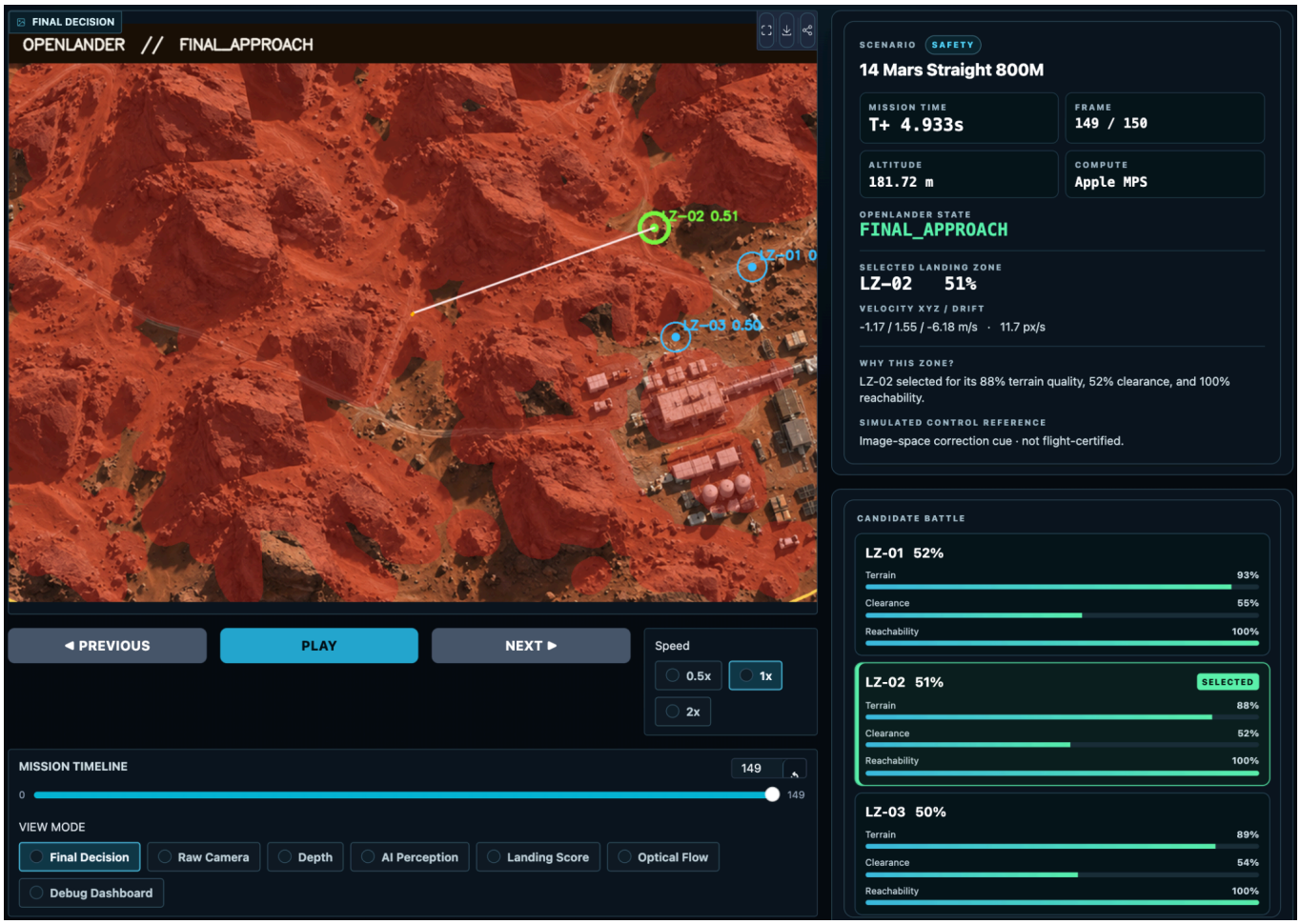


Full OpenLander interface with a bundled processed run loaded.

Area	Purpose
Primary view	Shows the selected visualization for the current frame.
Run input	Accepts a raw scenario path or processed-run path.
Status cards	Summarize mission state, altitude, drift, target, confidence, and frame time.
Candidate battle	Compares current landing-zone candidates and identifies the selected target.
Mission timeline	Shows major transitions rather than one event for every frame.

Read the landing decision

The competition-facing overlay keeps the final target and alternatives visually distinct.



Final Decision view: candidate landing zones, selected target, score, reachable envelope, footprint, and simulated control path.

Playback controls

- **Timeline slider:** scrub to any cached frame; the frame number and timestamp update together.
- **Play / Pause:** start or stop sequential replay.
- **0.5× / 1× / 2×:** change playback speed without altering the recorded decision sequence.
- **View buttons:** switch instantly between cached visualizations for the same frame.

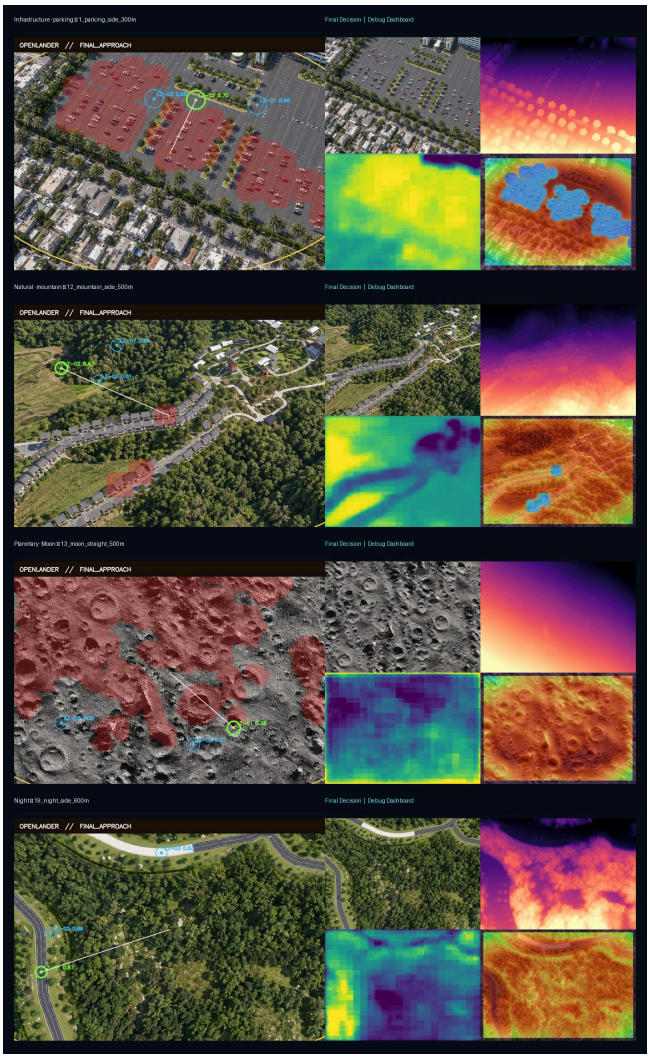
View modes

Final Decision	Clean presentation overlay with the selected LZ and flight context.
Depth	Relative monocular depth estimate used as one terrain cue.
Perception	Semantic and object-perception visualization.

Landing Score	Spatial suitability heatmap combining normalized safety factors.
Drift	Optical-flow field and measured visual motion.
Debug Dashboard	Composite diagnostic view for reviewing perception and decision behavior.

Inspect, question, and export

Cached evidence supports technical review without repeating inference.



Debug Dashboard selected during processed-run replay.

“Why Not Land Here?”

Click a location in the main image. OpenLander reports the local terrain-safety, clearance, reachability, and facility-preference components, then identifies the primary reason the point is weaker or stronger than the selected target. This is an explanation of the system score, not human-validated ground truth.



Landing zone selection and rejection analysis

Export video

1. Open a processed run and choose Final Decision or Debug Dashboard.
2. Select EXPORT VIDEO.
3. OpenLander encodes the cached frames only. No AI models are rerun.
4. The MP4 is saved inside the processed run as landing_replay.mp4 or dashboard_replay.mp4. GIF is used only if MP4 encoding is unavailable.

Landing terminology

Plain-language definitions used throughout the interface.

Landing Zone / LZ	A candidate image location that is large enough and scored as suitable for touchdown.
Terrain Safety	Estimated local surface regularity and suitability derived from visual and depth cues.
Obstacle Clearance	Estimated separation between the candidate footprint and detected or inferred hazards.
Drift	Apparent lateral scene motion measured from optical flow; shown in pixels per second.
Reachability	Whether a candidate remains inside the vehicle's simplified reachable envelope.
Landing Footprint	The projected space needed by the vehicle, including the configured safety margin.
Target Lock	The state in which a candidate has remained sufficiently strong and stable to become the committed target.
Retargeting	A controlled change from one selected candidate to another when evidence or reachability changes.
Final Approach	The late descent state after target commitment, with emphasis on stability and clearance.
Safety mode	Prioritizes terrain quality, clearance, stability, and reachability in natural or uncertain environments.
Recovery mode	Also values proximity to human infrastructure while retaining safety and reachability constraints.

How OpenLander reasons

Multiple perception signals are combined into an explainable decision sequence.

Perception stack

- **Depth estimation:** Depth Anything V2 produces relative monocular depth structure from each RGB frame.
- **Object detection:** YOLO identifies visible objects that may influence obstacle clearance.
- **Semantic perception:** CLIPSeg-style text-guided perception identifies broad scene concepts and useful regions.
- **Temporal motion:** OpenCV optical flow estimates visual drift between frames.
- **Decision logic:** A shared scoring configuration combines terrain safety, clearance, reachability, motion, and mission-mode preference. Candidate tracking and the state machine suppress unstable frame-to-frame changes.

Decision states

SEARCHING → CANDIDATE_FOUND → TRACKING → RETARGETING (when needed) → TARGET_LOCKED → FINAL_APPROACH

Important interpretation

> **DECISION SUPPORT, NOT CERTIFICATION**
>
> OpenLander demonstrates AI perception, candidate selection, replay, and explainability. The PID/control path is a simulated control reference for visualization. It is not a flight-certified controller and must not be used as one.

Troubleshooting

- **App does not open:** wait for macOS verification. If a security warning appears, Control-click the app in Finder, choose Open, and confirm Open. Confirm the Mac uses Apple Silicon and a supported macOS version.
- **Run is not listed:** confirm it is a direct child of processed_runs and contains run.json, result.json, and complete cached frame folders.
- **Raw analysis fails:** confirm scenario.json, metadata.csv, and frames/ agree on filenames and frame count.
- **Video export fails:** confirm the run contains complete annotated or dashboard PNG sequences and that the folder is writable.

End of manual • OpenLander • Competition Entry CPWC2026-0021