SLOPEGUARD · CW2

Simulation and Implementation of Mobile Robot Navigation

A risk-aware A navigation pipeline for a post-landslide UGV scenario*

Tomiwa Adeyemi | University of Hertfordshire

SCENARIO

From SlopeGuard concept to a working UGV layer

In CW1, SlopeGuard proposed a cooperative drone–UGV system for post-landslide response. CW2 takes the ground-navigation layer of that concept and demonstrates it end-to-end in simulation.

Inputs

Risk-weighted cost map
0.25 m grid, 25 × 25 m
Start, goal, hazard cues

UGV outcomes

Plan globally optimal risk-aware route
Replan on hazard updates

Constraints

Map updates mid-mission
Replan latency < 1.5 s
Differential-drive UGV

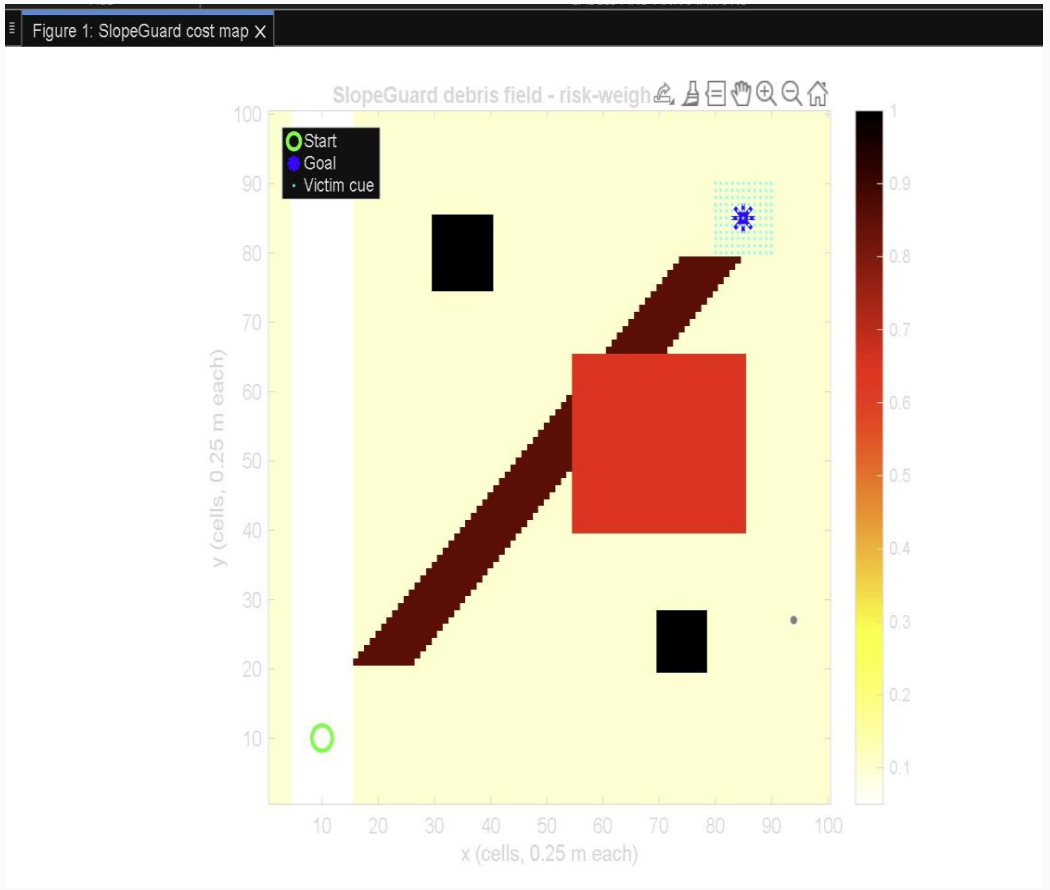


Figure 1. Debris-field cost map.

ARCHITECTURE

Four-stage navigation pipeline

1 Build environment

Construct cost map
0–1 traversability
Save .mat

`build_environment.m`

2 Risk-aware A*

Hand-written A*
J(c) = w_d·d + w_r·r
8-connected grid

`risk_aware_astar.m`

3 Hazard + replan

Mutate cost map
Full A* re-search
Compare paths

`hazard_replan.m`

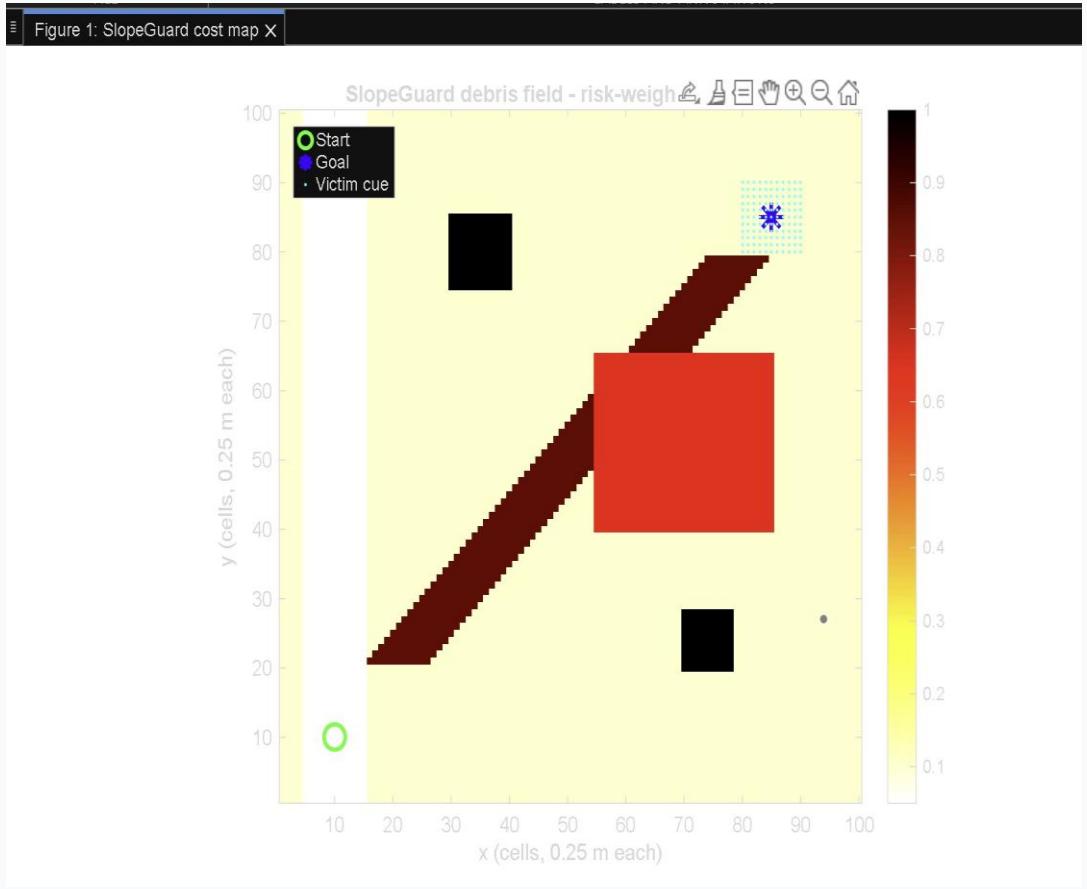
4 Execute on UGV

Pure Pursuit
Diff-drive kinematics
Log trajectory

`run_simulation.m`

Each stage is decoupled through .mat files — like a real producer-consumer navigation stack.

The cost map encodes traversability, not just occupancy



Background
`cost 0.10`
Easy ground

Diagonal scarp
`cost 0.85`
High slope severity

Loose-debris zone
`cost 0.65`
Roughness, sink risk

Boulders (×2)
`cost 1.00`
Impassable

Safe corridor
`cost 0.05`
Low-risk lane

Notionally produced by the SlopeGuard aerial layer (LiDAR + thermal).

Risk-weighted A*: a small change to the cost function

EDGE COST

$$J(c) = w_d \cdot d + w_r \cdot r$$

d = step distance r = per-cell risk w_d = 1.0 w_r = 5.0

Algorithm sketch

```
1. open ← {start};   gScore[start] = 0
2. while open not empty:
3.   pick node n with min f(n) = g + h
4.   if n == goal → reconstruct path
5.   for each 8-neighbour m of n:
6.     edge = w_d·d(n,m) + w_r·cost(m)
7.     if g(n) + edge < g(m): update m
```

Design choices

Hand-written, not toolbox

Full control over J(c). Earns the marks for correct algorithm implementation.

8-connected grid

Allows diagonal motion. Heuristic = Euclidean distance, admissible when w_d ≥ 1.

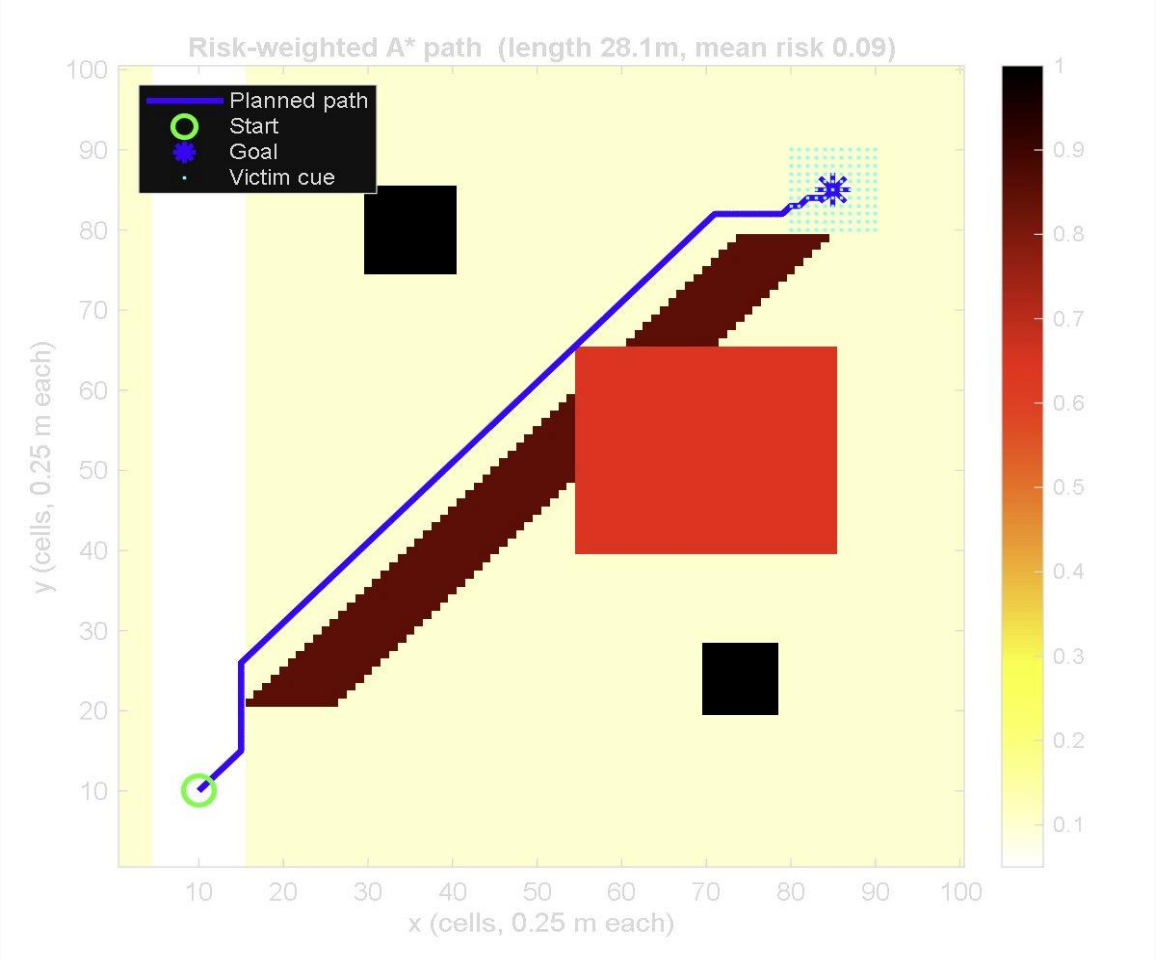
maxCost = 0.95 cut-off

Hard impassability for boulders. Hazards (0.92) are heavily penalised but still searchable.

Why not RRT* / Hybrid A*?

RRT* is non-deterministic; Hybrid A* adds little when motion is slow diff-drive.

Baseline: the planner correctly prefers safety over distance



28.13 m

Path length

Geometrically efficient — the corridor is used heavily

0.090

Mean cell risk

Almost entirely on near-background terrain

4925

Nodes expanded

Search space explored by the planner

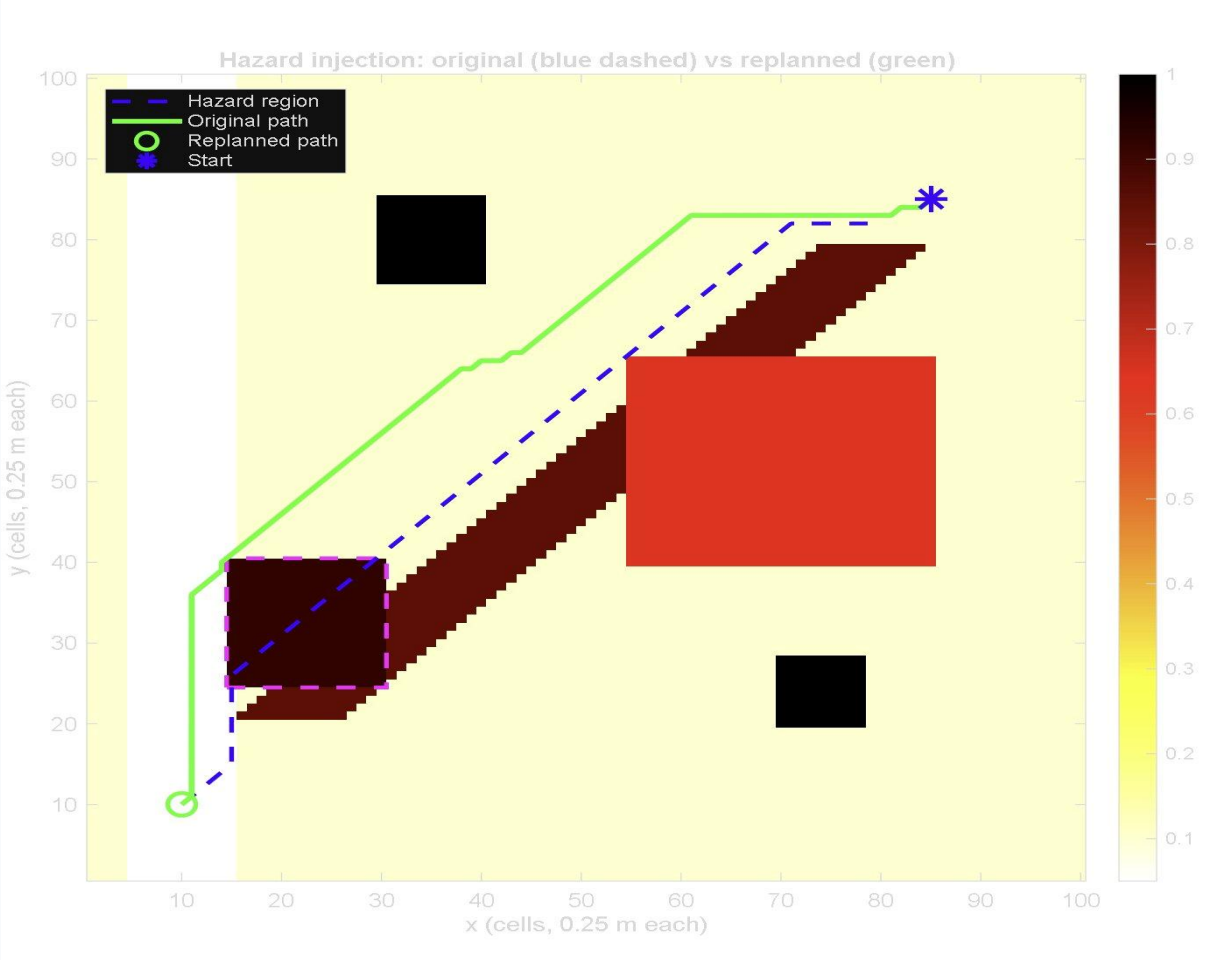
48.6 ms

Planning time

Well inside the 1.5 s SlopeGuard budget

STAGE 3 · REPLANNING

Mid-mission hazard: the original plan would walk straight into the slip

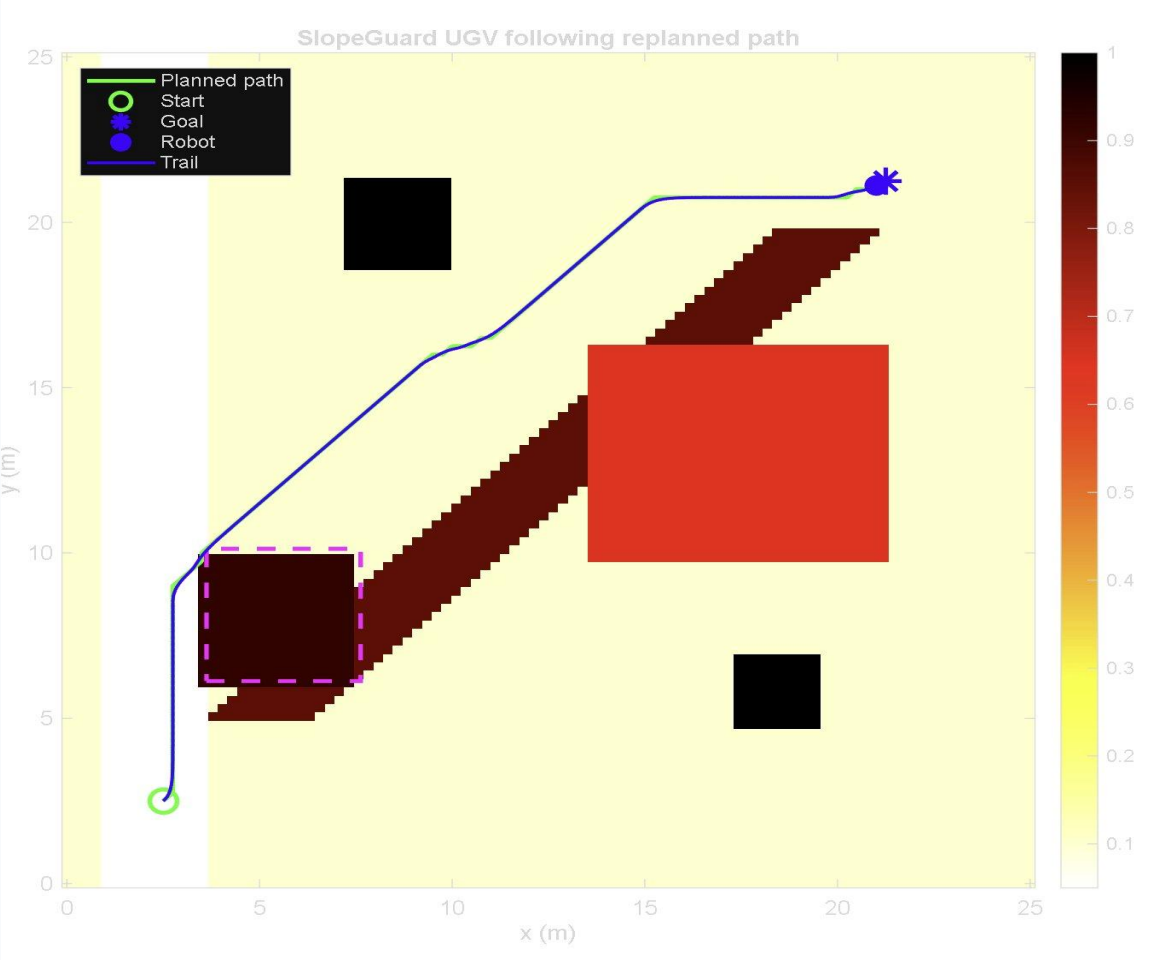


	Original	→	Replanned	
Path length	28.13 m	→	30.32 m	*+7.8%*
Mean cell risk	0.090	→	0.084	*safer*
Nodes expanded	4925	→	5997	*+22%*
Planning time	35.6 ms	→	51.7 ms	*+45%*

Detour is 7.8% longer, but actually slightly safer and it doesn't cross the new slip.

STAGE 4 · EXECUTION

The UGV actually drives it: Pure Pursuit + diff-drive kinematics



Run summary

59.1 s

Time to goal

29.55 m

Distance travelled (vs. 30.32 m planned)

−2.6 %

Tracking undershoot — Pure Pursuit smooths sharp corners

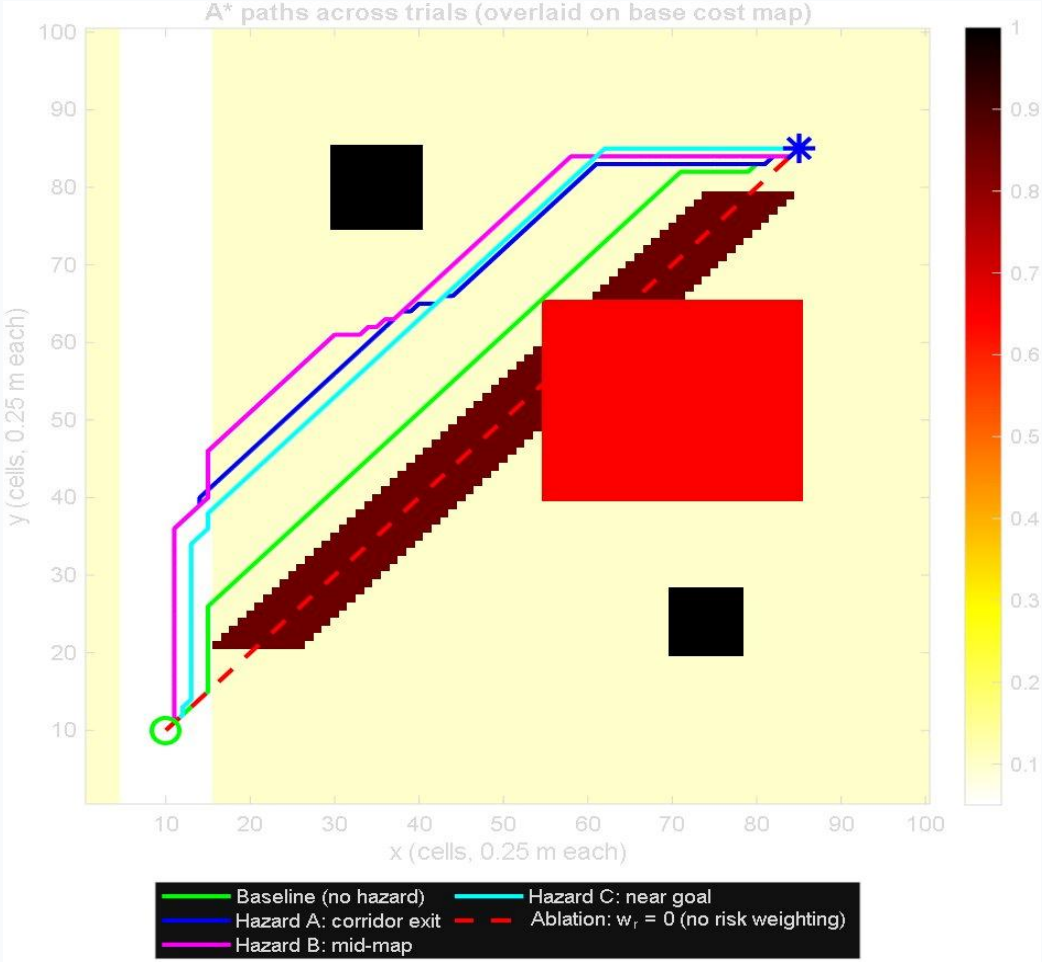
0.27 m

Final distance to goal — inside the 0.30 m radius

Parameters

`v = 0.5 m/s` `ω_max = 1.5 rad/s` `lookahead = 0.6 m`

Five trials, one cost map: behaviour is consistent and predictable



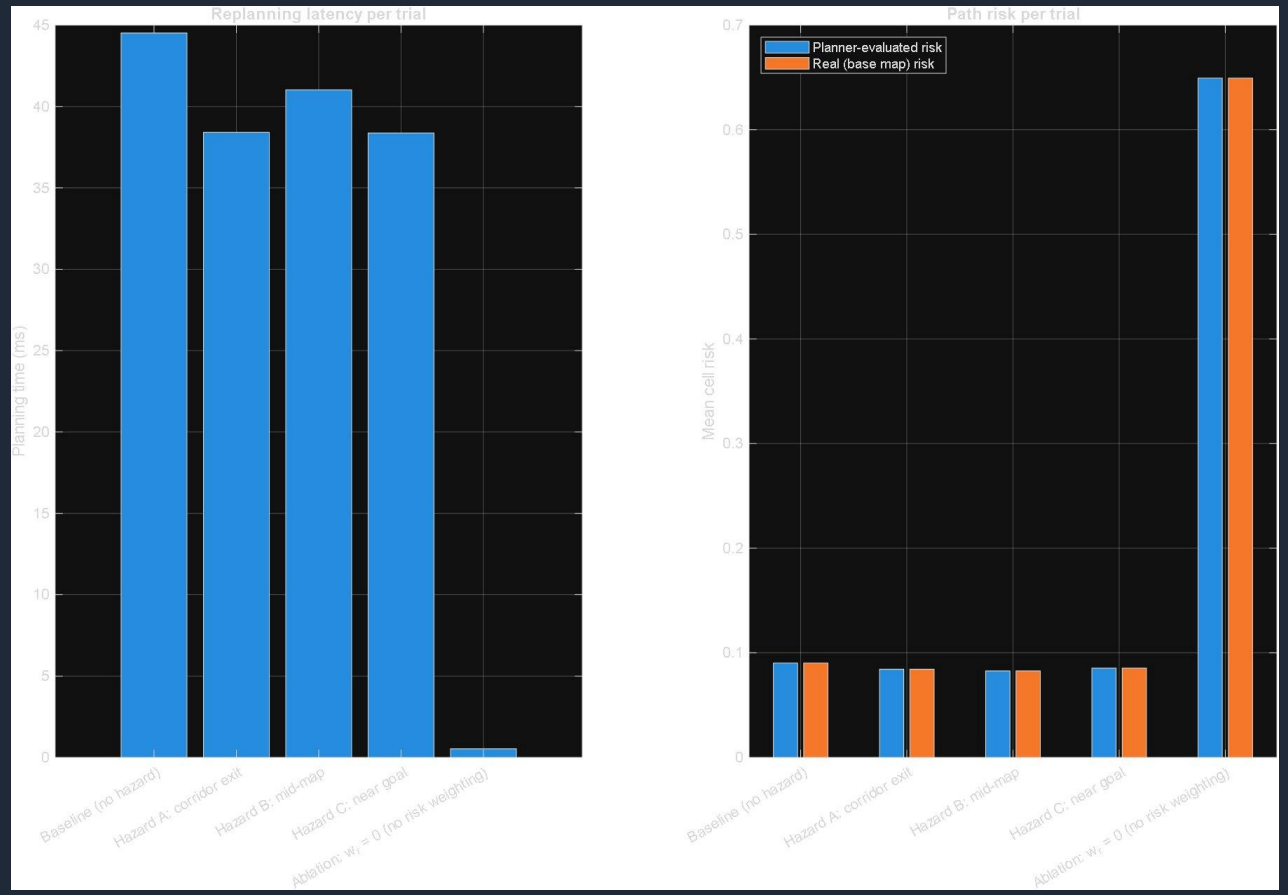
Trial	Length (m)	Real risk	Time (ms)
Baseline	28.13	0.090	48.6
Hazard A · corridor	30.32	0.084	44.8
Hazard B · mid-map	31.06	0.083	40.2
Hazard C · near goal	29.88	0.085	35.1
Ablation · w_r = 0	**26.52**	**0.649**	**2.4**

Key observation

Across all four risk-weighted runs, mean risk varies in a tight 0.083–0.090 band. The planner consistently routes through the same low-risk corridor regardless of where the hazard appears predictable behaviour matters when a human supervisor approves UGV moves.

HEADLINE FINDING

Without risk weighting, the planner walks the UGV through the scarp



0.090

Real risk

risk-weighted

0.649

Real risk

ablation

4925

Nodes expanded

risk-weighted

75

Nodes expanded

ablation

7× more real-map risk when the planner ignores terrain.
The ablation looks faster but the path it produces is unsafe.

REFLECTION

What worked, what didn't, and what's next

What worked

Risk-weighting changes path quality dramatically

Quantified by the ablation: 7× lower real-map risk.

Replanning is fast enough for the scenario

All replan latencies < 55 ms well below 1.5s budget.

Pipeline stages are cleanly decoupled

Each script runs independently, mirroring real stacks.

Limits & next steps

Full A* re-search instead of D* Lite

Incremental repair would be faster on bigger maps.

Static cost map between updates

A streaming aerial update at 1 Hz would test cooperation.

No observation noise in cost map

Stochastic perturbations would test robustness.

THANK YOU

Navigation as a safety-critical decision layer.

The drone discovers, the map reasons, the UGV acts only when the evidence says the route is acceptable.

CODE REPOSITORY

https://github.com/Tommieboi16/slopeguard-cw2

MATLAB scripts + saved environment + experimental data