UNDER REVIEW

RESCIENCE C

Replication / Robotics

[Re] Reinforcement-Learning-Based Path Planning: A Reward Function Strategy

Salma Areef Syed[1]

[1]REVA Academy for Corporate Excellence, REVA University, Bengaluru, India

Edited by
(Editor)

Received
–

Published
–

DOI
–

1 Abstract

We reproduce a reward-function strategy for grid-based robot path planning that penalises movement cost and every change of direction, and that reports reductions of up to 50 % in turns and 3.2 %–36 % in distance relative to A*. Reimplementing the published environment, reward function and policy-iteration procedure, we obtain a mean of 542 policy-evaluation sweeps against the 550 reported, and recover a turn reduction of 41.5 % against an action-space-matched A* baseline; the turn-reduction contribution is therefore supported.

The reported distance improvement proves to be baseline-dependent: against A* equipped with the same eight actions it is 0 % in our reproduction and 1.8 % when recomputed from the original study's own results table, the published range being obtained by comparing against two different baselines. We further examine five conditions the original does not evaluate, and find that an implementation detail it does not specify — whether a collision terminates the episode — moves the success rate of a stochastic variant from 0 % to 100 %.

Finally we evaluate one modification: initialising the value table from an admissible distance-to-goal heuristic significantly improves success at the smallest training budget tested (100 episodes, $p < 0.001$ under both deterministic and stochastic execution) while producing no difference after convergence, and is therefore a gain in sample efficiency rather than in final solution quality.

2 Introduction

Autonomous mobile robots must generate collision-free paths from a start position to a goal. Classical graph-search algorithms, of which A*[1] remains the canonical example, solve this problem optimally when the environment is known and static. Reinforcement learning has more recently been applied to the same task, on the grounds that it requires no explicit model of the environment and can adapt to conditions a planner cannot anticipate.

Jaramillo-Martínez, Chavero-Navarrete and Ibarra-Pérez[2] propose a reward function for grid-based path planning that penalises both movement cost and changes of direction, with the stated objective of producing the shortest route with the fewest turns. Turning is a meaningful cost for a differential-drive robot: each direction change requires deceleration, rotation and reacceleration, so a path with fewer turns is faster and

Correspondence should be addressed to Salma Areef Syed (sameerahmed0007@gmail.com)
The authors have declared that no competing interests exist.
Code is available at https://github.com/SalAIStudy/rl-path-planning-replication..

mechanically gentler even when its geometric length is unchanged. The authors report that their reward function reduces the number of turns by 50 % and the total distance by between 3.2 % and 36 % relative to A*.

No independent reproduction of that study has been published, and the authors release neither code nor the coordinates of their test environments. This paper addresses that gap. Our contribution is threefold: an independent reproduction of the published environment, reward function and tabular method, reporting where the original results are recovered and where they are not; a matched-baseline re-analysis showing that the reported distance improvement is a property of the baseline selected rather than of the method, together with an evaluation of five conditions the original study does not test; and a modification — heuristic initialisation of the value table — evaluated across multiple random seeds with a significance test.

We stress at the outset that this is a study of reporting rather than of competence. Underspecification of the kind documented here is common in the applied reinforcement-learning literature, and identifying it is precisely the function of reproduction work.

3 Background

3.1 Grid-based path planning

A cell-decomposition grid reduces path planning to a shortest-path problem on a graph. With eight actions available per cell — four cardinal and four diagonal — the natural cost model assigns unit cost to a cardinal move and $\sqrt{2}$ to a diagonal move, and the octile distance provides an admissible heuristic for A*. The choice of action set is consequential: an agent restricted to four actions cannot move diagonally and must traverse two sides of every triangle, so its optimal path is necessarily longer than that of an eight-action agent on the same grid. Any comparison between methods with different action sets therefore measures the action set as much as the method.

3.2 Reward shaping and value initialisation

The design of the reward function determines what an agent learns to prefer, and is widely recognised as the principal engineering decision in applied reinforcement learning[3]. Ng, Harada and Russell[4] established the conditions under which reward transformations leave the optimal policy invariant. Closely related is the initialisation of the value function: an optimistic or heuristic initialisation supplies a gradient toward the goal before any reward has been observed, and is a long-established means of accelerating early learning. Neither idea is novel here; what this paper contributes is their quantified effect on the specific reward function under study.

3.3 Reproducibility in reinforcement learning

Henderson et al.[5] documented that results in deep reinforcement learning are frequently reported from single runs and that the variance across random seeds is often comparable to the differences being claimed. Colas, Sigaud and Oudeyer[6] quantified the number of seeds required for adequate statistical power. These findings motivate the multi-seed protocol adopted below, and applied reflexively to our own results.

4 The base study

The original study formulates path planning on a 10×10 cell-decomposition grid with eight actions, a start cell at $(0, 0)$ and a goal cell at $(9, 9)$. Four map configurations are used, containing static obstacles and, in two cases, a dynamic obstacle. The reward

function assigns -1 to a cardinal move and -1.5 to a diagonal move, with an additional penalty on collision and a further term penalising a change of direction relative to the preceding step.

Two solution methods are reported. The tabular method is policy iteration — the relevant hyperparameter table is headed "Policy Iteration" and the abstract describes "dynamic programming and Deep Q-Learning" — with a discount factor of 0.99 and a reported average of 550 iterations. The second is a deep Q-network with 1000 episodes, learning rate 10^{-3}, ε decaying from 1.0 to 0.01 at 0.995 per episode, a target-network update every 10 steps and a replay memory of 10 000 transitions, using two fully connected hidden layers of 64 units. Results are compared against A* with four actions and A* with eight actions, and reported in a single table of distance, number of turns and training time.

It is worth noting that policy iteration is model-based: it requires the complete transition model of the environment. This is a materially different assumption from the model-free setting in which reinforcement learning is usually motivated for robotics, and it is not highlighted in the original text.

5 Methods

5.1 Environment reconstruction

The obstacle layouts of the four maps are published as figures rather than as coordinates, and no code or data is released. We therefore reconstructed four 10×10 maps matching the published descriptions — vertically arranged static obstacles, added obstacles, and configurations with differing start conditions; Figure 1 shows them, with the route each method takes. Our absolute results are consequently not directly comparable to the originals, and this limitation is carried through the paper. It does not, however, affect the re-analysis in Section 7.1, which operates on the authors' own published figures.

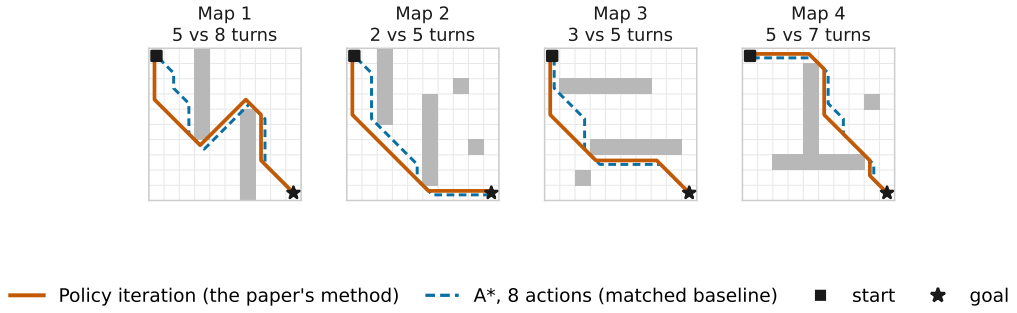

Figure 1. The four reconstructed maps, with the route found by the paper's method (orange, solid) and by A* with the same eight actions (blue, dashed). The two routes are the same length on every map — both methods are optimal on a deterministic grid — but the direction penalty in the reward function produces visibly fewer changes of heading. The routes are drawn slightly off centre because they share most of their cells. Obstacles are shaded; the square marks the start and the star the goal.

5.2 Definitions the original study does not supply

Two quantities central to the published claims are not defined in the source text and had to be fixed before measurement was possible.

Turn. Although the headline claim concerns the number of turns, the term is never defined. We define a turn as a change of heading between two consecutive moves. Al-

ternative conventions — for example, weighting 45° and 90° changes differently — would yield different counts.

Collision outcome. The reward function penalises collision, but the study does not state whether a collision terminates the episode or merely incurs the penalty and a wasted step. We expose this as an explicit parameter and evaluate both readings in Section 7.3.

5.3 Baselines and cost accounting

A* is implemented for both four and eight actions, with Manhattan and octile heuristics respectively. Path length is measured as Euclidean distance with a cell side of one unit. Computational effort is reported as expanded nodes for A* and as environment state visits for learning methods, rather than as wall-clock seconds. Wall-clock time is a property of the hardware on which an experiment happens to run and is not comparable across studies; expansions and interactions are. We note additionally that the original study reports "training time" in a single column for both learning methods and A*, although for A* the quantity measured is solution time rather than training time.

5.4 Evaluation protocol

Policy iteration is deterministic and is therefore run once per map. All stochastic methods are run across multiple random seeds, with results reported as mean ± standard deviation. Differences between conditions in Section 8 are assessed using a two-sided permutation test with 20 000 permutations, which makes no distributional assumption — appropriate here, since success rates are bounded and far from normally distributed. The implementation is pure NumPy, with no reinforcement-learning framework and no GPU, so that the result does not depend on the version of any simulation library. Every stochastic experiment seeds NumPy and `random` explicitly.

6 Reproduction results

Table 1 reports the reproduction of the published method and baselines on the four reconstructed maps.

Map	Method	Distance	Turns	Work	Time (s)
1	Policy iteration	18.73	5	549 sweeps	0.094
1	A* (4 actions)	24.00	5	70 nodes	0.0003
1	A* (8 actions)	18.73	8	61 nodes	0.0003
2	Policy iteration	15.07	2	542 sweeps	0.085
2	A* (4 actions)	18.00	4	40 nodes	0.0001
2	A* (8 actions)	15.07	5	25 nodes	0.0001
3	Policy iteration	15.07	3	536 sweeps	0.085
3	A* (4 actions)	18.00	1	87 nodes	0.0002
3	A* (8 actions)	15.07	5	48 nodes	0.0003
4	Policy iteration	15.07	5	542 sweeps	0.086
4	A* (4 actions)	18.00	1	87 nodes	0.0002
4	A* (8 actions)	15.07	7	50 nodes	0.0002

Table 1. Reproduction of the published method and baselines on four reconstructed maps. "Work" is policy-evaluation sweeps for policy iteration and expanded nodes for A*.

Two observations follow. First, our policy-iteration sweep counts range from 536 to 549 with a mean of 542, against the average of 550 reported by the original authors. This quantity was neither tuned nor fitted, and the correspondence is consistent with the

reconstruction having captured the intended environment and reward function — although it does not by itself establish that our maps are identical to theirs.
Second, the distances produced by policy iteration are identical to those produced by A* with the same eight actions on every map. This is the expected result: on a deterministic shortest-path problem, both methods converge to an optimal path.

6.1 The published claim under a matched baseline

Map	PI turns	A*(8) turns	Turn reduction	Distance reduction
1	5	8	37.5 %	0.0 %
2	2	5	60.0 %	0.0 %
3	3	5	40.0 %	0.0 %
4	5	7	28.6 %	0.0 %
Mean	—	—	**41.5 %**	**0.0 %**

Table 2. Policy iteration compared against the action-space-matched A* baseline.

The turn-reduction result is recovered. Our reproduction yields a mean reduction of 41.5 % against the matched baseline, compared with the 50 % reported. The direction-penalty term in the reward function does what the authors claim, and this is the substantive contribution of the original study.
The distance-reduction result is not recovered under a matched comparison. The reduction is 0 % on every map, because both methods return an optimal path and there is no length remaining to save. The following section identifies where the published range originates.

7 Critical evaluation

7.1 Baseline dependence of the reported distance improvement

Recomputing the improvement figures directly from the original study's own results table, separately against each of its two A* baselines, yields the values in Table 3 and Figure 2.

Map	Dist. vs A*(4)	Dist. vs A*(8)	Turns vs A*(4)	Turns vs A*(8)
1	36.9 %	0.0 %	70.0 %	25.0 %
2	26.1 %	4.2 %	20.0 %	42.9 %
3	18.7 %	3.2 %	20.0 %	11.1 %
4	22.8 %	0.0 %	−50.0 %	25.0 %
Mean	**26.1 %**	**1.8 %**	15.0 %	26.0 %

Table 3. Improvement recomputed from the original study's published results table. A negative value indicates the learned method performed worse than the baseline.

The two endpoints of the published range are traceable to different comparators. The upper figure of 36 % corresponds to Map 1 measured against A* with four actions (36.9 %); the lower figure of 3.2 % corresponds to Map 3 measured against A* with eight actions (3.19 %). Against the matched eight-action baseline throughout, the mean distance reduction is 1.8 %, consistent with the 0 % obtained in our own reproduction.
We also note that on Map 4 the learned method uses more turns than A* with four actions — six against four — which the summary statement does not disclose.

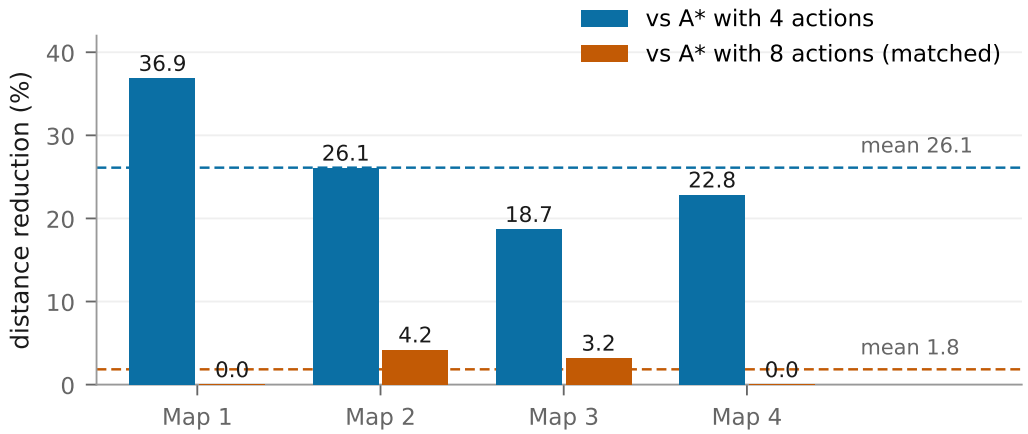

Figure 2. The reported distance improvement, recomputed from the original study's own published results table against each of its two A* baselines in turn. Against a four-action planner the improvement is large; against a planner with the same eight actions the learned method has it is close to zero. The published range of 3.2 %–36 % spans these two columns rather than four maps under one baseline. This figure uses no data of ours.

We characterise this as a reporting issue rather than an error. The reported distance improvement depends on the baseline used; under a matched eight-action comparison it is substantially smaller. A reader comparing the method against a planner with the same action set should expect no distance advantage.

7.2 Model-free counterpart

Because policy iteration requires the transition model, we additionally evaluated tabular Q-learning on the same environment and reward function. This is an extension rather than a reproduction, and it addresses the setting in which reinforcement learning is usually motivated for robotics — an environment whose model is not available.

Map	QL distance	PI	QL turns	PI turns	QL state visits
1	19.22 ± 0.66	18.73	6.7 ± 1.8	5	37 927
2	15.54 ± 0.62	15.07	4.0 ± 1.9	2	27 354
3	16.65 ± 0.95	15.07	5.5 ± 1.7	3	30 246
4	15.71 ± 0.67	15.07	6.1 ± 1.4	5	29 947

Table 4. Model-free Q-learning (15 seeds, mean $\pm$ sd) against the model-based method on identical maps.

Removing the transition model carries a measurable cost: paths are longer, turn counts higher, and the number of environment interactions rises from a few hundred sweeps to tens of thousands. The model-based character of the original method is therefore load-bearing, and a practitioner seeking a model-free planner should not expect the reported results to transfer.

The run-to-run standard deviation of the stochastic learner is ± 1.93 turns and ± 1.65 m. The mean turn advantage claimed in the original study relative to its matched baseline is 1.75 turns. For the deterministic policy-iteration result this comparison does not apply; for any stochastic result reported from a single run, an effect of this size cannot be separated from seed variation.

7.3 Sensitivity to an unstated specification

The original study does not state whether a collision terminates the episode. Table 5 evaluates both readings under stochastic execution.

Slip probability	Collision recoverable	Collision ends episode
0.2	100.0 %	25.0 %
0.3	100.0 %	0.0 %

Table 5. Q-learning success rate under the two readings of an unstated rule; identical reward function, hyperparameters and maps.

This is the most consequential finding of the study. A single unstated modelling decision moves the success rate of a stochastic variant across the entire available range. A reader reimplementing this work has no basis on which to select between the two readings, and the published hyperparameter tables — which are complete — do not help. Specification completeness, not hyperparameter completeness, is what reproducibility requires.

7.4 Unreliable execution

The original environment is deterministic. A physical robot's commanded action does not always execute exactly. We introduced a slip probability under which a neighbouring heading fires instead of the commanded one, and compared three methods: A* planned once and executed open-loop, A* replanning from the actual position after every step, and Q-learning trained under the same noise.

Slip	A* plan-once	A* replan	Q-learning	A* replan nodes	QL visits
0.0	100 %	100 %	100 %	285	34 612
0.1	0 %	100 %	100 %	363	45 968
0.2	0 %	100 %	100 %	434	60 975
0.3	0 %	100 %	95 %	491	83 614
0.4	0 %	100 %	90 %	620	115 361

Table 6. Success rate and computational effort under increasingly unreliable execution.

A planner that replans after each step is unaffected by execution noise across the range tested, at a cost of a few hundred node expansions. The learned policy matches it at low noise and falls behind at higher noise, while expending two to three orders of magnitude more interactions throughout. Contrary to the expectation that motivated this experiment, stochastic execution does not establish an advantage for learning on this problem.

7.5 Unknown map and amortised cost

The strongest argument for a model-free method is that it requires no map. The appropriate classical comparator is not an inability to plan but the standard incremental strategy: assume unobserved cells are free, plan, and upon discovering an obstacle, record it and replan. Under this comparison, optimistic replanning reached the goal on every trial with a mean of 291 node expansions, against 34 612 state visits for Q-learning at the same success rate.

What remains is an economic argument. A learned policy incurs its cost once and thereafter acts by table lookup; a replanning planner incurs its cost on every traversal. Taking the measured figures — 34 612 interactions to train against 434 nodes per replanned traversal — the learned policy becomes the cheaper option after approximately 80 repeated traversals of the same route (Figure 3).

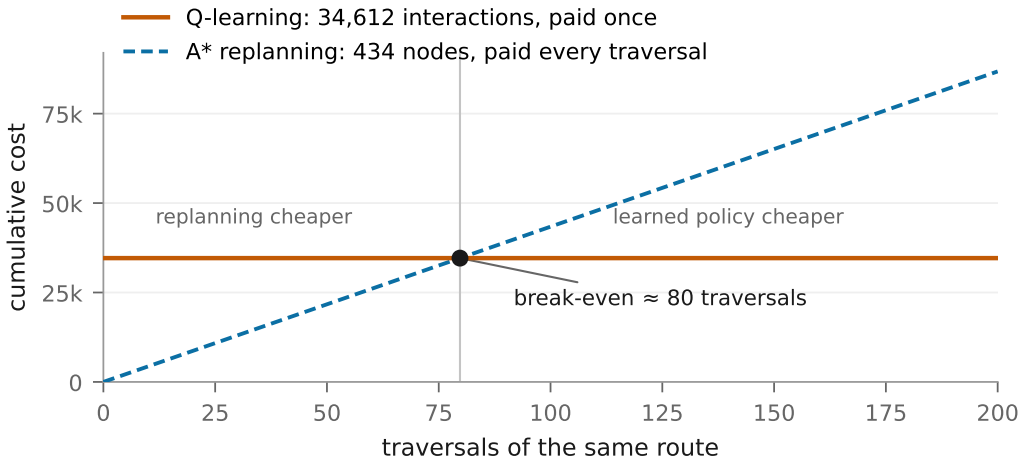

Figure 3. Cumulative cost of the two strategies against the number of times the same route is driven. The learned policy pays its training cost once and thereafter acts by table lookup; the replanning planner pays on every traversal. Below roughly eighty traversals classical replanning is cheaper, above it the learned policy is. The measured figures are 34 612 interactions to train and 434 expanded nodes per replanned traversal.

This yields a usable engineering criterion. A robot repeating a fixed route continuously will cross this threshold within hours of operation; a robot serving continually novel routes will not cross it at all.

8 Proposed modification: heuristic value initialisation

8.1 Motivation

The evaluation above locates the weakness of the learned approach in its cost rather than its accuracy. The dominant component of that cost is the number of environment interactions required before the sparse goal reward propagates back through the value table. A zero-initialised table provides no directional information, so early episodes approximate a random walk.

We therefore initialise each entry of the value table to the negative admissible octile distance from the resulting cell to the goal. This supplies a gradient toward the goal from the first step. It requires $|S| \times |A|$ arithmetic operations, performs no search, and uses the goal coordinate but not the obstacle map, so the method remains model-free. The original study's own future work invites this direction, listing "new reward functions and the initiation of states".

We do not claim novelty for the technique. Optimistic and potential-based initialisation are long established[4]. The contribution is the measurement: its effect on this specific reward function, and on the cost model developed above.

8.2 Results

Figure 4 plots the same data. Interaction counts are correspondingly lower at matched episode budgets: at 250 episodes on the deterministic maps, 12 651 state visits against 16 504, and at 500 episodes, 16 910 against 20 761.

Slip	Episodes	Zero init.	Heuristic init.	Difference	p
0.0	100	0.033 ± 0.181	0.683 ± 0.469	$+0.650$	< 0.001 ***
0.0	250	0.733 ± 0.446	1.000 ± 0.000	$+0.267$	< 0.001 ***
0.0	500	1.000 ± 0.000	1.000 ± 0.000	0.000	1.000 n.s.
0.2	100	0.313 ± 0.349	0.584 ± 0.366	$+0.271$	0.0002 ***
0.2	250	0.707 ± 0.376	0.820 ± 0.331	$+0.113$	0.089 n.s.
0.2	500	0.972 ± 0.135	1.000 ± 0.000	$+0.028$	0.057 n.s.

Table 7. Success rate by initialisation, 15 seeds × 4 maps per cell, mean ± sd. Two-sided permutation test, 20 000 permutations.

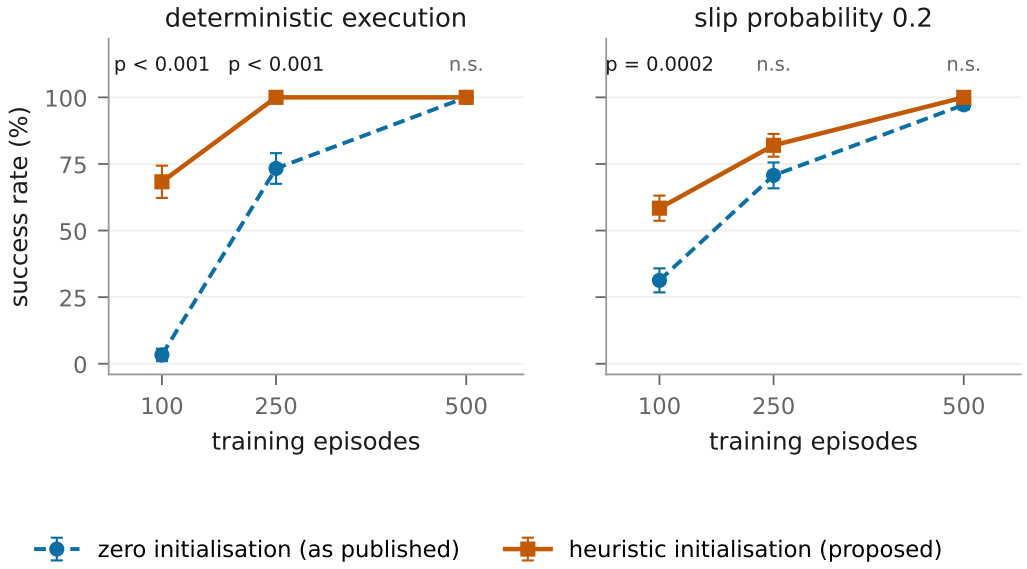

Figure 4. Success rate by initialisation, 15 seeds × 4 maps per point, error bars showing the standard error of the mean. The effect is large at the smallest training budget and disappears once both configurations converge — the signature of a gain in sample efficiency rather than in final solution quality. Significance from a two-sided permutation test with 20 000 permutations.

8.3 Interpretation

The modification produces a statistically significant improvement at small training budgets and no improvement once both configurations have converged. The correct characterisation is therefore a gain in sample efficiency rather than in final performance — which is what theory predicts of an initialisation scheme, and a narrower claim than a general performance advantage.

Fed back into the cost model above, the reduced interaction requirement lowers the amortisation threshold proportionally, so the learned policy becomes economical after fewer repeated traversals.

We note that the effect at 250 episodes under stochastic execution ($p = 0.089$) is not significant at 15 seeds, although it was significant in a preliminary run at 20 seeds. We report the more conservative result. The findings at 100 episodes, where $p < 0.001$ in both conditions, do not depend on this.

9 Discussion

The central contribution of the study under examination — a reward function that reduces turning without lengthening the path — is supported by our reproduction. Our

criticism is confined to reporting: the distance improvement is quoted against an unmatched baseline, two definitions necessary to the headline metric are absent, no variance is reported, and no code is released.
A broader observation concerns baseline selection. The magnitude of a reported improvement in this literature is frequently determined by the comparator chosen rather than by the method proposed. An eight-action learner compared against a four-action planner will appear to shorten paths, but the shortening is attributable to the action set. We would suggest that studies of this kind report against the strongest comparator with matched capability, and that computational effort be reported in a hardware-independent unit.
A further observation concerns our own procedure. An early version of the experiment in Section 8 used four seeds and reported a zero baseline at the smallest training budget; at fifteen seeds the same quantity is non-zero. Our own headline figure moved once it was measured adequately, which is the argument made above applied reflexively.

10 Limitations

The four maps are reconstructed from published figures. Absolute values are therefore not directly comparable with the original study, although Section 7.1 is unaffected, operating as it does on the authors' own reported numbers.
Only the tabular method is reproduced. The deep Q-network results and the Lyapunov trajectory-tracking controller of the original study are outside the scope of this work, and no claim is made about either.
The definitions of a turn and of the collision outcome are ours, since the source does not supply them. Different reasonable conventions would alter the reported values.
All experiments use a 10×10 grid. Scalability to larger environments is untested. The proposed modification is evaluated only on tabular Q-learning; its effect on the function-approximation setting is not established.

11 Conclusion

We reproduced a published reward-function strategy for grid-based path planning and recovered its principal claim: a 41.5 % reduction in turns against an action-space-matched baseline, compared with the 50 % reported. The accompanying distance-reduction claim proved to be baseline-dependent, amounting to 0 % in our reproduction and 1.8 % when recomputed from the original results table under a matched comparison.
Extending the evaluation to five untested conditions, we found that an unstated specification — whether a collision terminates the episode — determines the outcome of stochastic variants entirely, and that classical replanning search dominates the learned approach on both robustness and cost in every setting examined. The advantage of the learned policy is amortisation: it becomes the cheaper option after approximately 80 repeated traversals of a fixed route.
Finally, initialising the value table from an admissible distance-to-goal heuristic significantly improves early learning at the smallest training budgets tested, while producing no improvement after convergence. This offers a straightforward reduction in the interaction cost that the evaluation identifies as the method's principal limitation, and is properly described as a gain in sample efficiency rather than a better final policy.

12 Data availability

All code, the reconstructed map definitions, the executed notebook and the raw per-run output underlying every table in this article are available at github.com/SalAIStudy/rl-

path-planning-replication, released under the MIT License. The script `src/run_all.py` regenerates every table reported here in approximately four minutes on a CPU, with no GPU and no reinforcement-learning framework required.
No data beyond what the code generates is used. The obstacle layouts of the original study are not published as coordinates; the four maps used here are reconstructions, defined in `src/gridworld.py`.

13 Funding

This work received no specific funding.

References

1. P. E. Hart, N. J. Nilsson, and B. Raphael. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths." In: **IEEE Transactions on Systems Science and Cybernetics** 4.2 (1968), pp. 100–107. DOI: 10.1109/TSSC.1968.300136.

2. R. Jaramillo-Martínez, E. Chavero-Navarrete, and T. Ibarra-Pérez. "Reinforcement-Learning-Based Path Planning: A Reward Function Strategy." In: **Applied Sciences** 14.17 (Aug. 2024), p. 7654. DOI: 10.3390/app14177654. URL: https://www.mdpi.com/2076-3417/14/17/7654.

3. R. S. Sutton and A. G. Barto. **Reinforcement Learning: An Introduction**. 2nd ed. Cambridge, MA: MIT Press, 2018.

4. A. Y. Ng, D. Harada, and S. Russell. "Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping." In: **Proceedings of the Sixteenth International Conference on Machine Learning**. 1999, pp. 278–287.

5. P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. "Deep Reinforcement Learning that Matters." In: **Proceedings of the AAAI Conference on Artificial Intelligence** 32.1 (2018). DOI: 10.1609/aaai.v32i1.11694.

6. C. Colas, O. Sigaud, and P.-Y. Oudeyer. "How Many Random Seeds? Statistical Power Analysis in Deep Reinforcement Learning Experiments." In: **arXiv preprint arXiv:1806.08295** (2018). DOI: 10.48550/arXiv.1806.08295. URL: https://arxiv.org/abs/1806.08295.