Project III Report for COM S 4/5720 Spring 2025: Risk–Aware Stochastic Planner for Three-Agent Pursuit–Evasion

Nicholas Morrow[1]

I. Problem Restatement

Project III preserves the grid-world and three-agent rules of Project II but introduces move probabilities. When an agent issues a move $a \in \mathcal{A}$ the environment executes $\text{left}(a)$ with probability p_1, a with probability p_2, and $\text{right}(a)$ with probability p_3. $\mathbf{p} = (p_1, p_2, p_3)$ is hidden and may differ for each run. The planner must therefore:

- maximise capture probability,
- minimise expected collision risk
- satisfy a strict real-time (single-step) budget.

II. Detailed Planner Logic

TABLE I

Symbol dictionary (code constants in parentheses).

w_t	target weight $= 1.0$ ('W_PREY')
w_p	pursuer weight $= 1.2$ ('W_PURS')
c_r	risk cost $= 40$ ('RISK_COST')
w_s	mobility weight $= 1$ (implicit)
d_t	Manhattan dist. to prey ('mhd(cur,prey)')
d_p	Manhattan dist. to pursuer ('mhd(cur,purs)')
E	# empty 4-neighbours around current cell
stayBias	-0.04 ('STAY_BIAS')
τ_0	base risk gate $= 0.12$ ('RISK_TH_BASE')
$\tau_{\max}$	hard cap $= 0.28$ ('RISK_TH_MAX')
k	adaptive slope $= 0.03$ ('ADAPT_RATE')
N	stall limit 10 ('STALL_LIMIT')

Project III introduces **actuation noise**: every move is executed as *left*, *straight*, or *right* with an *unknown* probability vector $\mathbf{p}$.We therefore extend the deterministic planner from Project II with *risk-aware* scoring, online probability learning, and a shallow expectimax look-ahead.

A. Risk Tensor and Expected-Value Scoring

Worst-case risk. Following the safety-first paradigm of risk-sensitive MDPs [1], we pre-compute a binary tensor

$$\mathcal{R}_{\text{wc}}[r, c, k] = \begin{cases} 1 & \text{if } \textit{any} \text{ rotation of } k \text{ crashes} \\ 0 & \text{otherwise.} \end{cases}$$

Moves with $\mathcal{R}_{\text{wc}} = 1$ are never executed, so the agent can never kill itself.

Online estimate of p. The realised rotation $o_t \in \{0, 1, 2\}$ is inferred from successive positions and used to update a Dirichlet posterior $(n_0, n_1, n_2) \leftarrow (n_0, n_1, n_2) + \mathbf{e}_{o_t}$, giving the current mean $\hat{\mathbf{p}}_t = \mathbf{n} / \sum_i n_i$ (Bayesian update [2]).

At run time each admissible action receives the expected-utility score [3]

$$\text{EV}(k) = w_t\, \mathbb{E}_{\hat{\mathbf{p}}_t}\Big[\tfrac{1}{d_t+1}\Big] - w_p\, \mathbb{E}_{\hat{\mathbf{p}}_t}\Big[\tfrac{1}{d_p+1}\Big] - c_r\, \mathcal{R}_{\text{wc}}[r, c, k] + w_s\, E + \textit{stayBias}$$

where the expectation is taken over the *learned*.

B. Step-wise Decision Pipeline

1) **Dynamic risk gate.** A tolerance $\tau = \min(\tau_0 + k \cdot \textit{idleSteps}, \tau_{\max})$ filters actions with $\mathcal{R}_{\text{wc}} \leq \tau$ (idea adapted from low-risk planning [4]).
2) **Capture lunge.** Issue the safest action $k^\star$ immediately if $\Pr[\text{capture} \,|\, k^\star] > \mathcal{R}_{\text{wc}}[r, c, k^\star]$.
3) **Emergency flee.** When the pursuer is within 3 Manhattan steps, choose the safe move that maximises $2d_p - d_t$ (heuristic used by [5]).
4) **Risk-bounded A* search.** Run A* [6] on the subgraph $\{(r, c) \mid \mathcal{R}_{\text{wc}} \leq \tau\}$.
5) **Stall breaker.** After N stagnant frames, pick the move that minimises the prey's escape exits [7].
6) **Lightweight expectimax (1.5-ply).** We build an expectimax tree of depth 1 and then re-evaluate only the three best root actions. Because the second level is explored only partially, the effective look-ahead is called "1.5-ply". Expectimax is a relatively popular algorithm which branches off of the Minimax algorithm, but assumes non-optimal play.

References

[1] R. A. Howard and J. E. Matheson, "Risk-sensitive markov decision processes," *Management Science*, vol. 18, no. 7, pp. 356–369, 1972.

[2] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[3] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson, 2016.

[4] L. F. Pérez and G. Rey, "Lra*: A low-risk anytime algorithm for motion planning," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 2543–2549.

[5] N. Iskander and S. M. LaValle, "Pursuit and evasion in a polygonal environment using a visibility graph," *IEEE International Conference on Robotics and Automation*, 2005.

[6] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[7] J. Jones, "Multi-agent choke point strategies," in *AAAI Workshop on Multi-Agent Path Finding and Other MAPF Variants*, 2019.

[1]Nicholas Morrow is with the Department of Computer Science, Iowa State University, Ames, IA 50011, USA nmorrow@iastate.edu