Physics-Driven Machine Learning for Time-Optimal Path Planning in Stochastic Dynamic Flows

DDDAS 2020

Submission 11

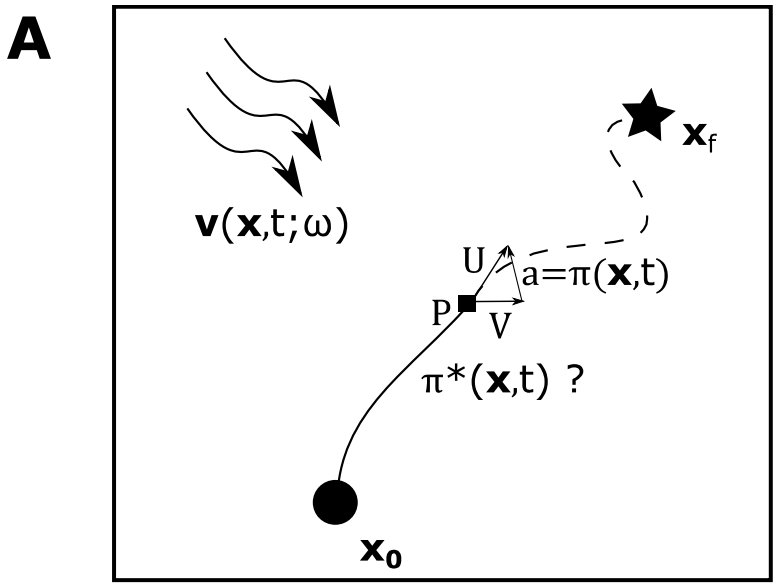
Rohit Chowdhury, Deepak Subramani

Dept. of Computational and Data Sciences

Indian Institute of Science Bangalore, India

Path Planning for Autonomous Agents

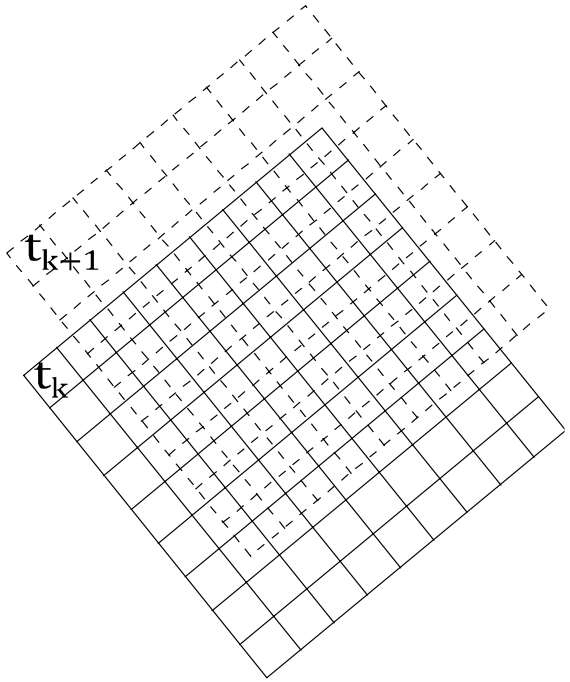
- Autonomous agent P in $\mathrm{v}(x, t; \omega)$
- Travel from x_0 to x_f
- Compute optimal policy that
  - minimizes expected travel time from x_0 to x_f
  - dynamically update it.
- Requires
  - in-mission measurement of $\mathrm{v}(x, t; \omega)$
  - data assimilation to get posterior field.
- Need to update policy based on assimilated field
- Hence, the necessity to use DDDAS[1,2] paradigm
- Earlier work on path planning using DDDAS[3,4]

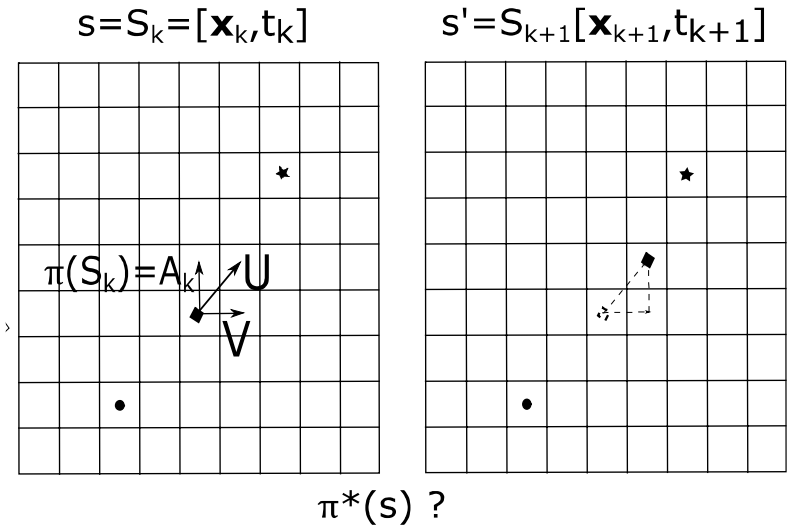


[1] Darema, F.: Dynamic data driven applications systems: A new paradigm for application simulations and measurements. In: International Conference on Computational Science. pp. 662–669. Springer (2004)
[2] Blasch, E., Bernstein, D., Rangaswamy, M.: Introduction to dynamic data driven applications systems. In: Handbook of Dynamic Data Driven Applications Systems, pp. 1–25. Springer (2018)
[3] Singh, V. and Willcox, K., Methodology for Path Planning with Dynamic Data-Driven Flight Capability Estimation, AIAA Journal, Vol. 55, No. 8, pp. 2727-2738, 2017.
[4] D. Sacharny and T. C. Henderson, "Optimal Policies in Complex Large-scale UAS Traffic Management," 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), Taipei, Taiwan, 2019, pp. 352-357

Problem Setup

- Problem posed as a Markov Decision Process (MDP)
- Discretized State Space: 3D spatio-temporal grid.
  - $S_k = [x_k, tk]$
- Discretized Action Space: Discretized heading
  - $A_k = Fcos\theta\hat{\imath} + Fsin\theta\hat{\jmath}$
  - Agent's speed, F is constant.
  - $\theta \in \{0, \delta, 2\delta, \ldots\ 2\Pi - \delta\}$
- One step reward,
  - $R_k = -\Delta t$
  - $t_{k+1} - t_k = \Delta t \quad \forall k$
- Policy $\pi(S_k) = Ak$
- What is the optimal policy, π^*?
  - How to compute it? – Dynamic Programming (DP)
  - Computationally feasible alternative ?



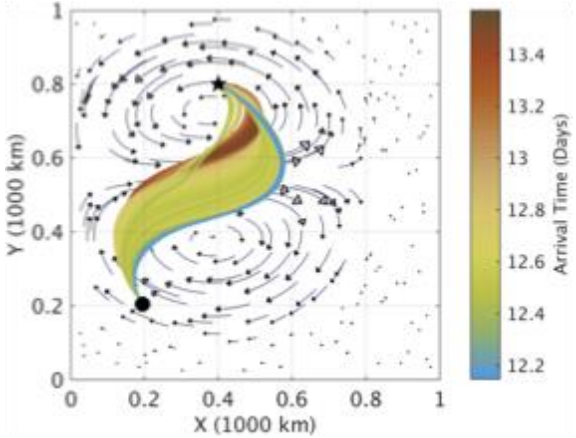


Modelling Framework

Data-Driven Probabilistic Environment Modelling

Environment Forecast: Dynamically Orthogonal (DO) barotropic Quasi-Geostrophic (QG) stochastic equations

Get N realizations of the stochastic velocity field



Stochastic PDEs for Exact Time-Optimal Path Planning

Compute reachability fronts using Hamilton-Jacobi (HJ) level-set S-PDE

For each ω calculate the optimal paths by using the particle back-tracking equation

Obtain distribution of exact time-optimal paths for every ω

HJ S-PDE

$$\frac{\partial \phi(x,t;\omega)}{\partial t} + F(t)\,|\nabla \phi(x,t;\omega)| + v(x,t;\omega).\nabla \phi(x,t;\omega) = 0\,,$$

$$IC:\ \phi(x,0;\omega) = |x - x_0|$$

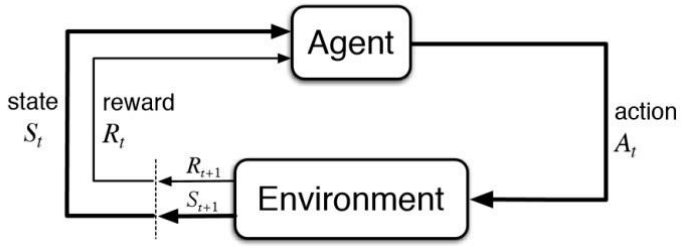
Reinforcement Learning (RL)

$q_\pi(s,a) = \mathbb{E}_\pi[\sum R_k\,|s,a]$

$q_*(s,a) = \mathbb{E}[R(s,a) + \gamma\, max_{a'}\{\, q_*(s',a')\}|s,a]$ Bellman Optimality

DP: Needs model, computationally expensive. Used as a benchmark for performance.

RL: Collect data $(S_k = s, A_k = a, R_{k+1} = r, S_{k+1} = s')$. Estimate q_* .Learning Q-values



Physics-driven Model-based Q-Learning

Phase 1: Transfer Learning (offline)

Compute N rzns of Velocity Field using DO stochastic eqns.

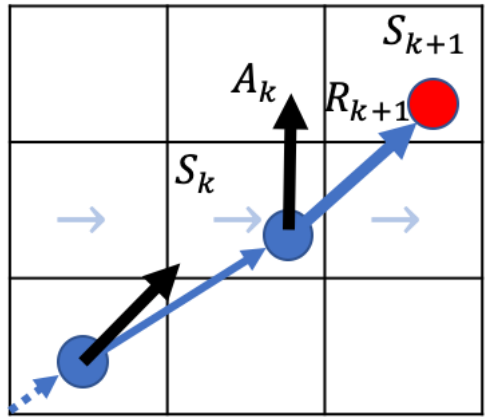
Compute exact time-optimal paths from HJ-PDE & backtracking eqns.

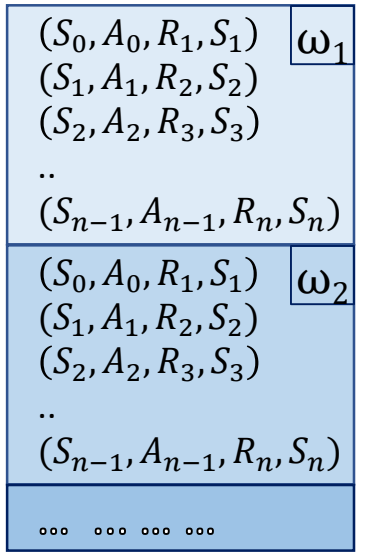
Agent performs action sequence $\{A_k\}$ to follow exact-time optimal path

Build experience buffer by storing tuples $(S_k, A_k, R_{k+1}, S_{k+1})$

Update Q-values using experience buffer

Compute initial policy π^1 from Q-values





Experience Buffer

Phase 2: Dynamic Data Driven Policy Update (in mission)

Perform action according to π^1

Collect mission env. data and compute posterior forecast

Perform future "rollout-simulation" using ε-greedy Q-Learning

Perform next action based on updated Q-values

Q-Learning Update Eqn:

$$Q(S_k, A_k) \leftarrow Q(S_k, A_k) + \alpha\left[R_{k+1} + \max_a Q(S_{k+1}, a) - Q(S_k, A_k)\right]$$

Compute Policy:

$$\pi^1(s) = \max_a Q(s, a)$$

Physics-driven Model-based Q-Learning

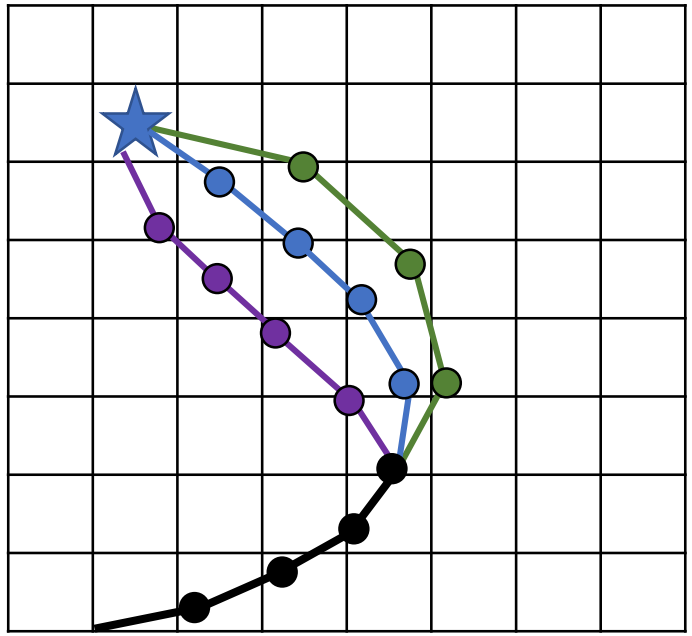
Phase 1: Transfer Learning (offline)

Compute N rzns of Velocity Field using DO stochastic eqns. → Compute exact time-optimal paths from HJ-PDE & backtracking eqns. → Agent performs action sequence $\{A_k\}$ to follow exact-time optimal path

Build experience buffer by storing tuples $(S_k, A_k, R_{k+1}, S_{k+1})$ → Update Q-values using experience buffer → Compute initial policy π^1 from Q-values

Phase 2: Dynamic Data Driven Policy Update (in mission)

Perform action according to π^1 → Collect mission env. data and compute posterior forecast → Perform future "rollout-simulation" using ε-greedy Q-Learning → Perform next action based on updated Q-values

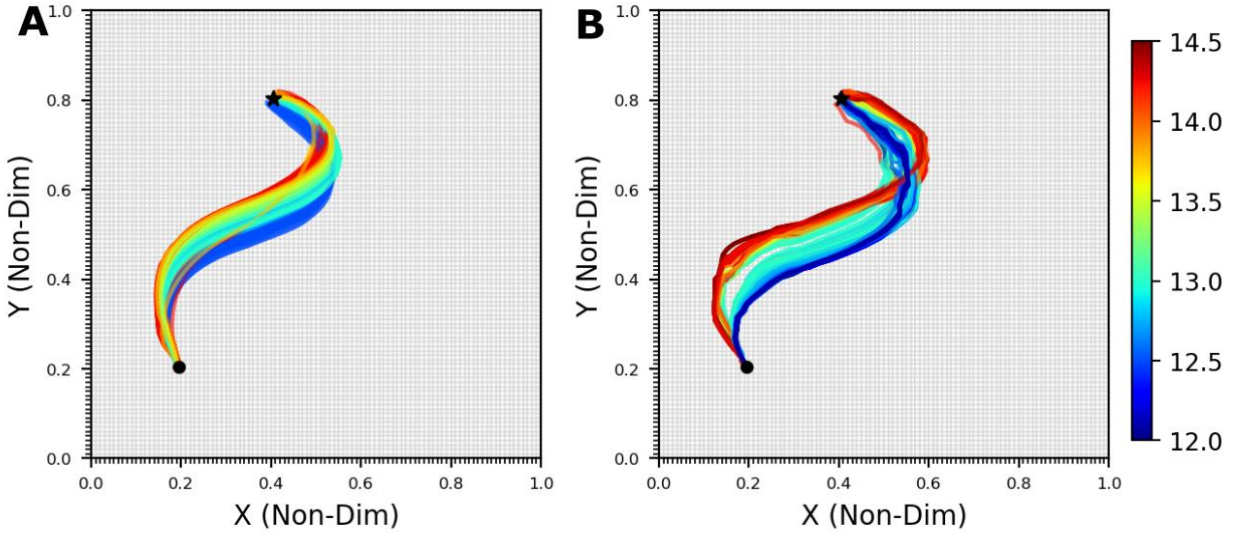


Application

Canonical stochastic Quasi-Geostrophic Double Gyre flow scenario[5]

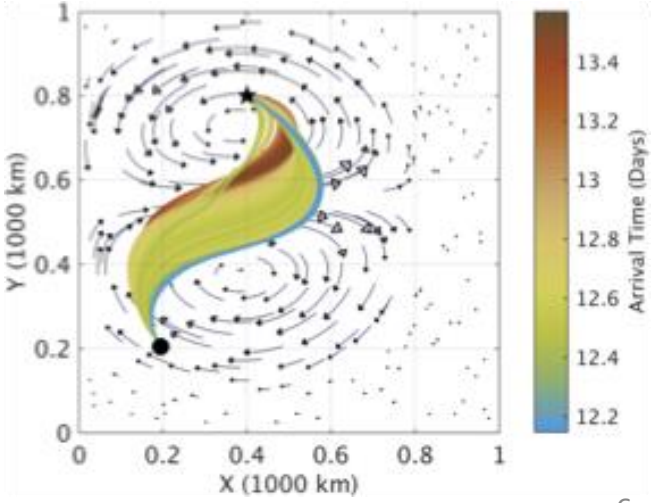
Problem parameters:

- Square Domain: $1000km \times 1000km$
- Grid Size: $100 \times 100 \times 60$
- $\Delta x = \Delta y = 10km$
- $\Delta t = 0.24\ days$
- Vehicle Speed = $40\ cm/s$
- Action Space: $\{0, 22.5, 45, \ldots 337.5\}$
- No. of realizations, $N = 5000$
  - 4000 for learning initial policy in Phase 1
  - 1000 for testing the algorithm



C

	QL	DP
E[T]	13.1 days	12.6 days
p(fail)	0.46	0.43
Comp. Time	11.4 mins	90 mins



[5] Subramani, D.N., Wei, Q.J., Lermusiaux, P.F.J.: Stochastic time-optimal path-planning in uncertain, strong, and dynamic flows. CMAME 333, 218–237 (2018)

Conclusion and Future Work

- Novel, computationally efficient algorithm to learn a near time-optimal policy.
- Phase 1: Transfer Learning
  - Learn initial policy with Q-Learning update equation from a distribution of exact paths from HJ-PDE in an environment modelled with DO eqns.
- Phase 2: Dynamic Data Driven Policy Update
  - Update policy in-mission using DDDAS principle.
  - Perform data assimilation and future rollout Q-learning simulations.

Next Steps:

- Improvement in data assimilation scheme.
- Evaluation of proposed algorithm in other realistic environments.