Aircraft path planning with power beaming

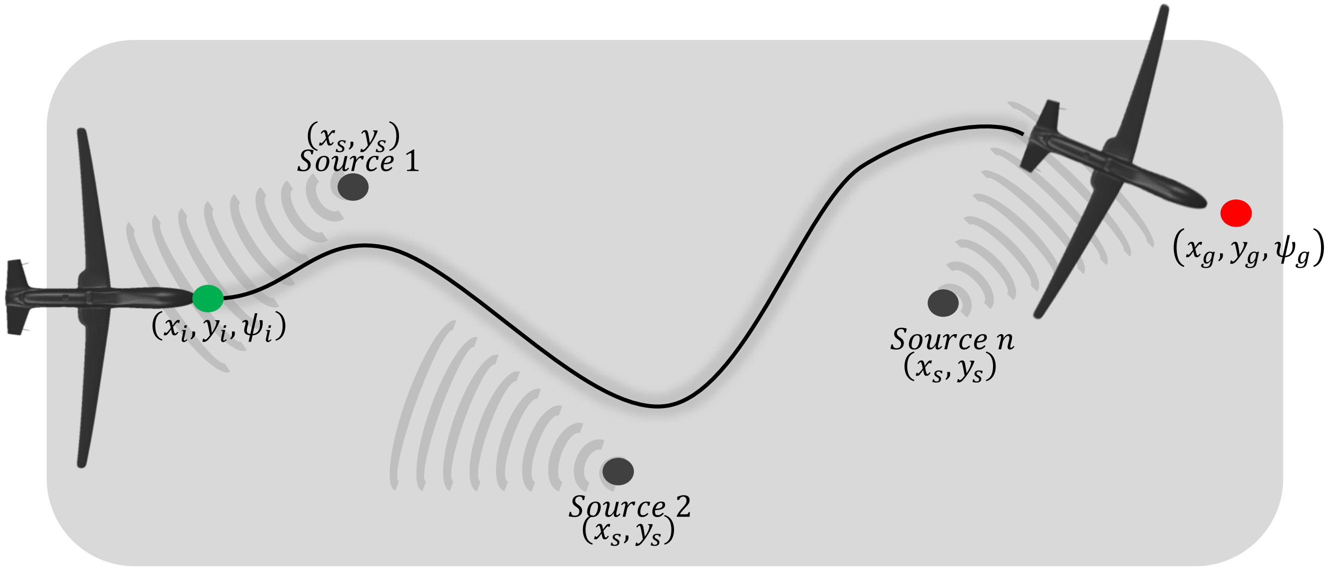
Nicholas Orndorff

MAE242 Project Presentation

12/08/2022

Problem description

- *Power beaming:*
  - Send energy to an aircraft via electromagnetic beams (e.g., optical or radio frequency)
  - Decouples an aircraft from its energy source
- *Goal:*
  - Find an optimal trajectory through a field of power sources

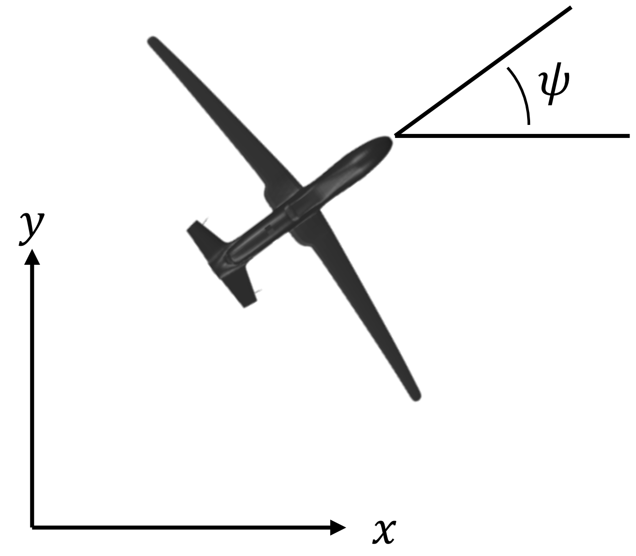


Problem description: continuous dynamics

- Continuous flight dynamics model: $\mathcal{S} = (x, y, \psi)$

$$\dot{x} = v \sin \psi$$
$$\dot{y} = v \cos \psi$$
$$\dot{\psi} = \theta$$

- The Earth-frame position is (x, y), the heading angle is $\psi \in [-\pi, \pi]$, and the action θ is the heading angle in the body-frame
- Assumptions:
  - Constant altitude
  - Constant velocity



Problem description: discrete dynamics

- Discrete dynamics (fixed time step *dt*)

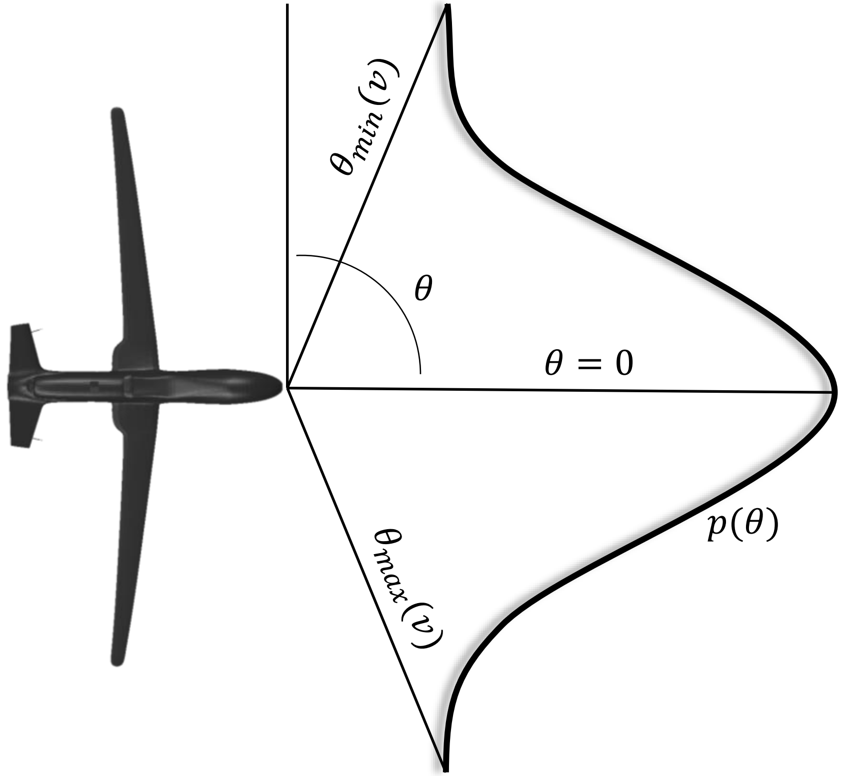
$$x_{t+1} = x_t + v \sin\psi \, dt$$
$$y_{t+1} = y_t + v \cos\psi \, dt$$
$$\psi_{t+1} = \psi_t + \theta$$

- Discrete action space

$$\mathcal{A} = \begin{bmatrix} \frac{-\pi}{6} & \frac{-\pi}{9} & \frac{-\pi}{18} & \frac{\pi}{18} & \frac{\pi}{9} & \frac{\pi}{6} \end{bmatrix}$$

$$P(\theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-0.5\left(\theta/\sigma\right)^2}, \forall\theta \in \mathcal{A}$$

- Captures the preference of the aircraft to continue along the current heading (conservation of momentum)



Problem description: reward

- Euclidean distance from the aircraft to a power source:

$$d = \sqrt{(x - x_s)^2 + (y - y_s)^2}$$

- Approximate a typical transmission-loss equation:

$$P_r = P_t G_t G_r \left(\frac{\lambda}{4\pi d}\right)^2 \approx \frac{k}{d^2}$$

- Use a minimization function to avoid undesirable dynamics:

$$\mathcal{R} = \min\left(h, \frac{k}{d^2}\right)$$

- Add the distance to the target state and create the final reward function:

$$\mathcal{R} = d_g^2 + \sum_{i=1}^{n} \min\left(h, {}^{k}/_{d_i^2}\right)$$

Markov decision process: summary

- MDP: $\langle \mathcal{S}, \mathcal{A}, P, \mathcal{R}, \gamma \rangle$
  - $\mathcal{S} = (x, y, \psi)$
  - $\mathcal{A} = \theta$
  - $P(a) = \frac{1}{\sigma\sqrt{2\pi}} e^{-0.5\left({}^{a}/_{\sigma}\right)^2}, \forall a \in \mathcal{A}$
  - $\mathcal{R} = d_g^2 + \sum_{i=1}^{n} min\left(h,\ {}^{k}/_{d_i^2}\right)$
    - And a substantial reward (+100) at the terminal state
  - $\gamma \in [0, 1]$

Policy iteration

- *Policy evaluation:*
  - Iteratively computes the value function $V(s)$ for all states and a given policy
- *Policy improvement:*
  - Finds a new action for every state that improves the action values $q(s, a)$
- *Implementation:*
  - Fixed state-space dimensions: (60x30x20)
  - Fixed starting state: (1,15), and fixed terminal state: (55,15)
  - Fixed power source locations
- *What to expect:*
  - convergence, slow, reliable and repeatable

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$; $V(\mathit{terminal}) \doteq 0$

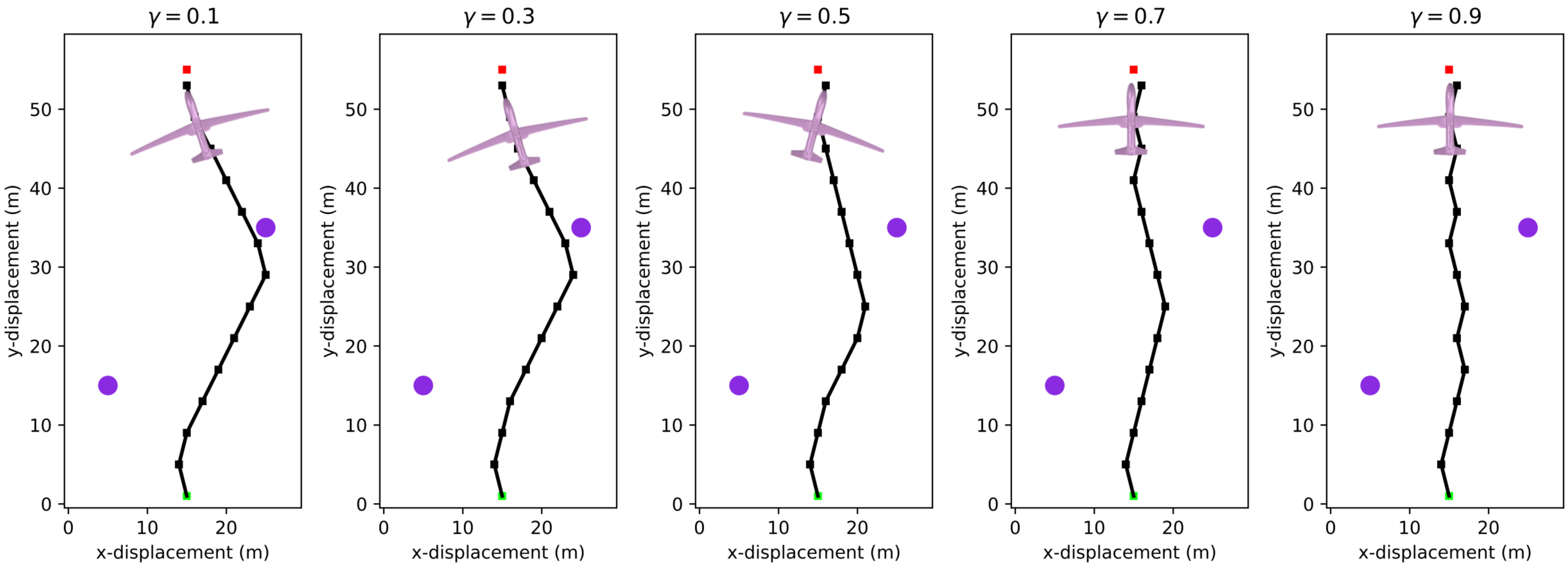
2. Policy Evaluation
   Loop:
   - $\Delta \leftarrow 0$
   - Loop for each $s \in \mathcal{S}$:
     - $v \leftarrow V(s)$
     - $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s))\big[r + \gamma V(s')\big]$
     - $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   - *old-action* $\leftarrow \pi(s)$
   - $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r \mid s, a)\big[r + \gamma V(s')\big]$
   - If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*

   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Policy iteration: results



Prioritizes the short-term reward → Prioritizes the large terminal reward

Policy iteration: convergence

- Major iterations
  - Iterations of evaluation and improvement
- Minor iterations
  - Evaluation iterations plus improvement iterations
- Requires more iterations with increasing discount factor
- Slow: 32,000 individual states

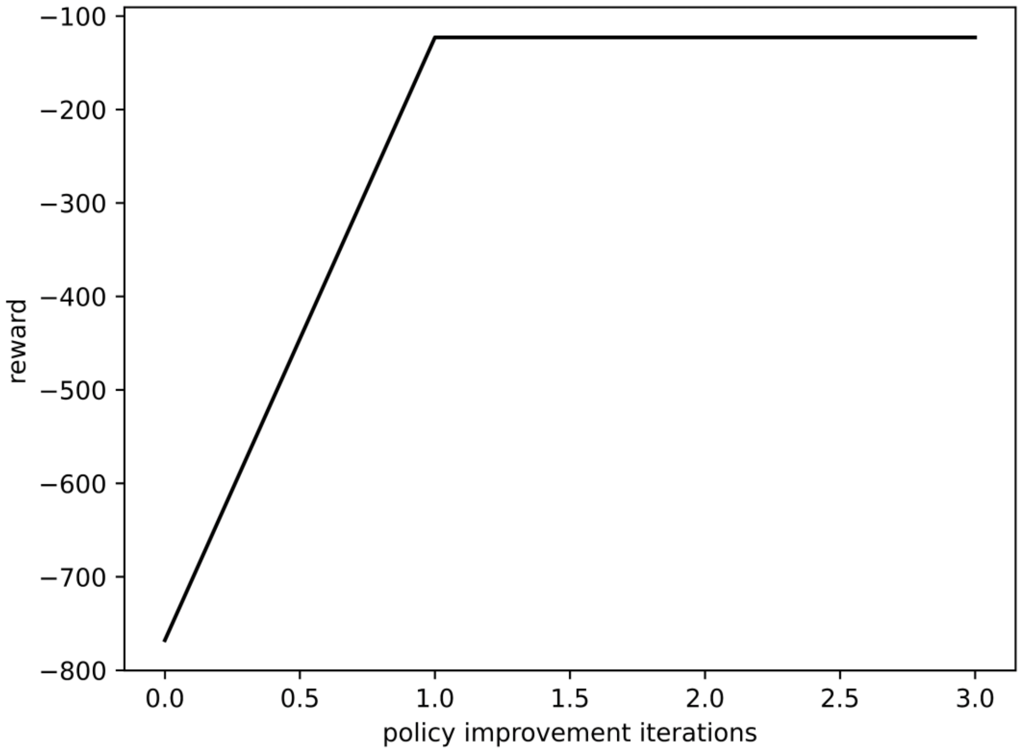


Table 1: Policy iteration summary.

γ	iterations	time (s)
0.1	10	228
0.3	12	230
0.5	14	234
0.7	21	358
0.9	33	389

SARSA

- *On-policy TD control:*
  - Estimates q_π instead of v_π
  - Policy improves based on a greedy q_π: $S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1}$
  - Convergence only guaranteed as state-action pair visits$\rightarrow \infty$
  - Trades exploration vs. exploitation (ϵ-greedy policies)

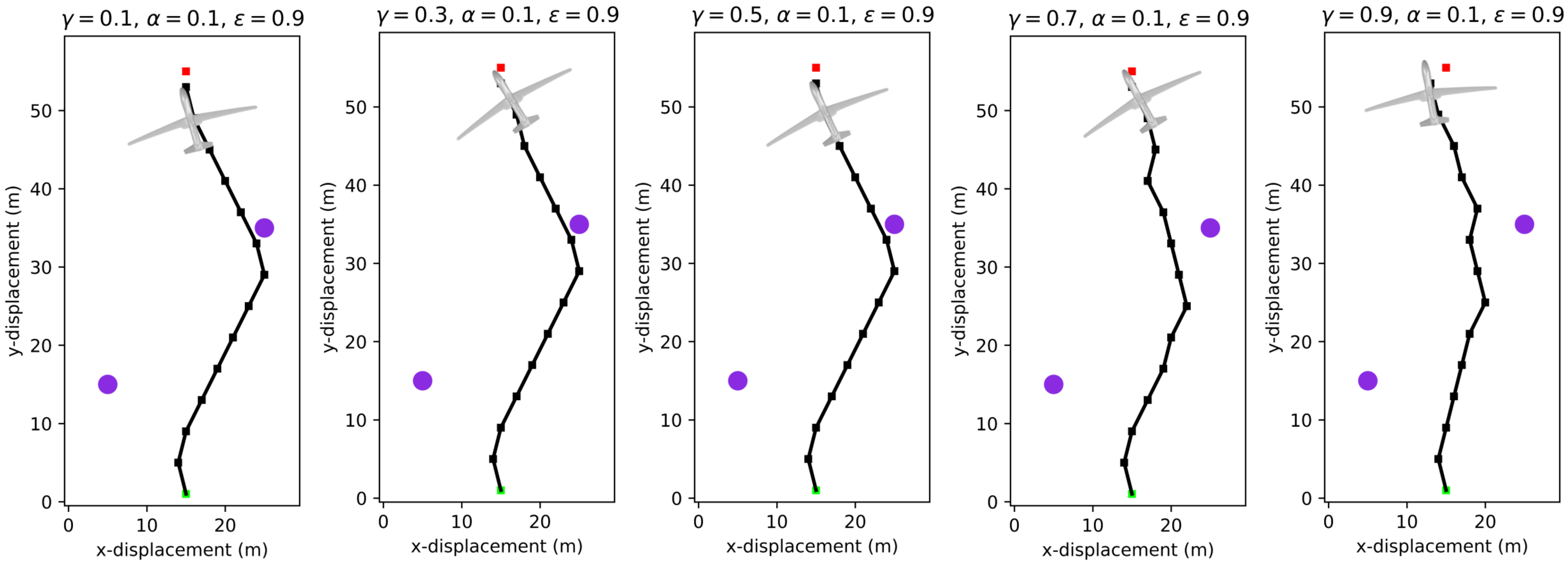
- *Implementation:*
  - Max-episodes = $\mathcal{O}(10{,}000)$
  - Max-time-steps = 50 (some episodes may not reach the terminal state in a reasonable amount of time)
  - Same state space, start/end states, and power source locations as with policy iteration

- *What to expect:*
  - Stochastic results (due to epsilon-greedy behavior)
  - convergence not guaranteed for fixed number of episodes and time steps

1. Initialize $\alpha \in (0,1]$, $Q(s,a) \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$

2. While **episode** $<$ **max-episodes:**
   Initialize S
   Choose A from S using policy derived from Q (e.g., ϵ-greedy)
   While **t** $<$ **max-time-steps:**
   Take action A, observe R,S'
   Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)
   $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma Q(S',A') - Q(S,A)]$
   $S \leftarrow S'$; $A \leftarrow A'$
   $t = t + 1$
   Until S is terminal

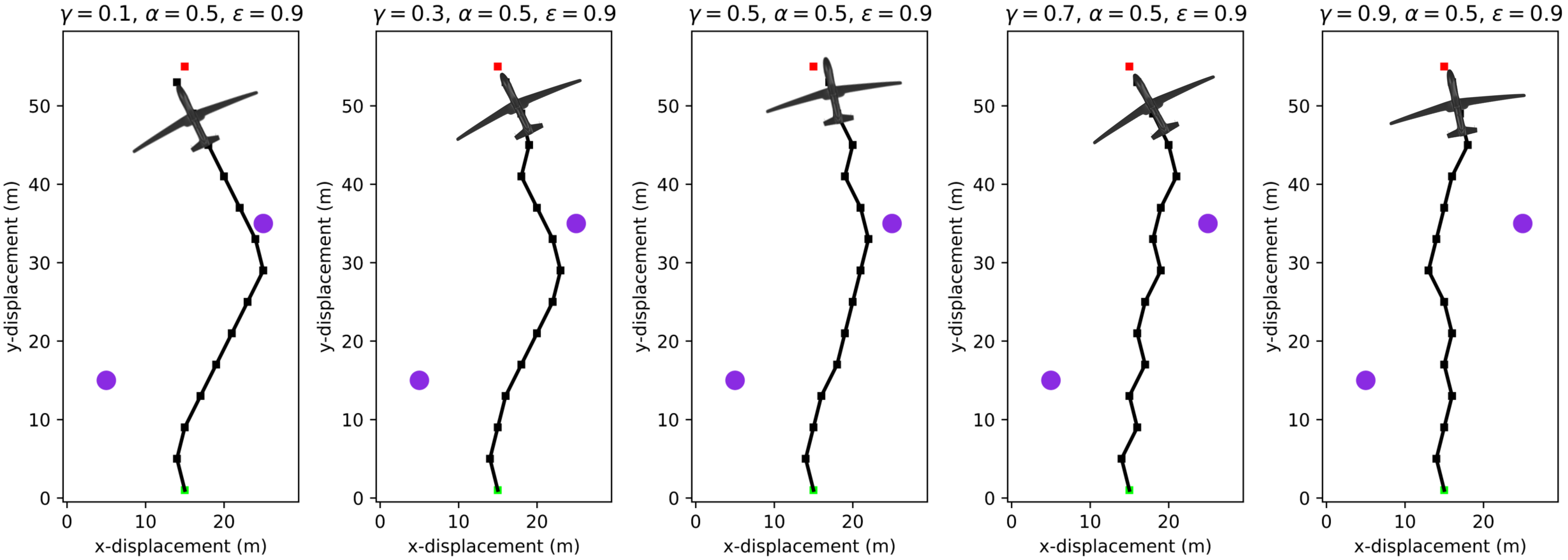
SARSA: results $\alpha = 0.1$



Prioritizes the short-term reward → Prioritizes the large terminal reward

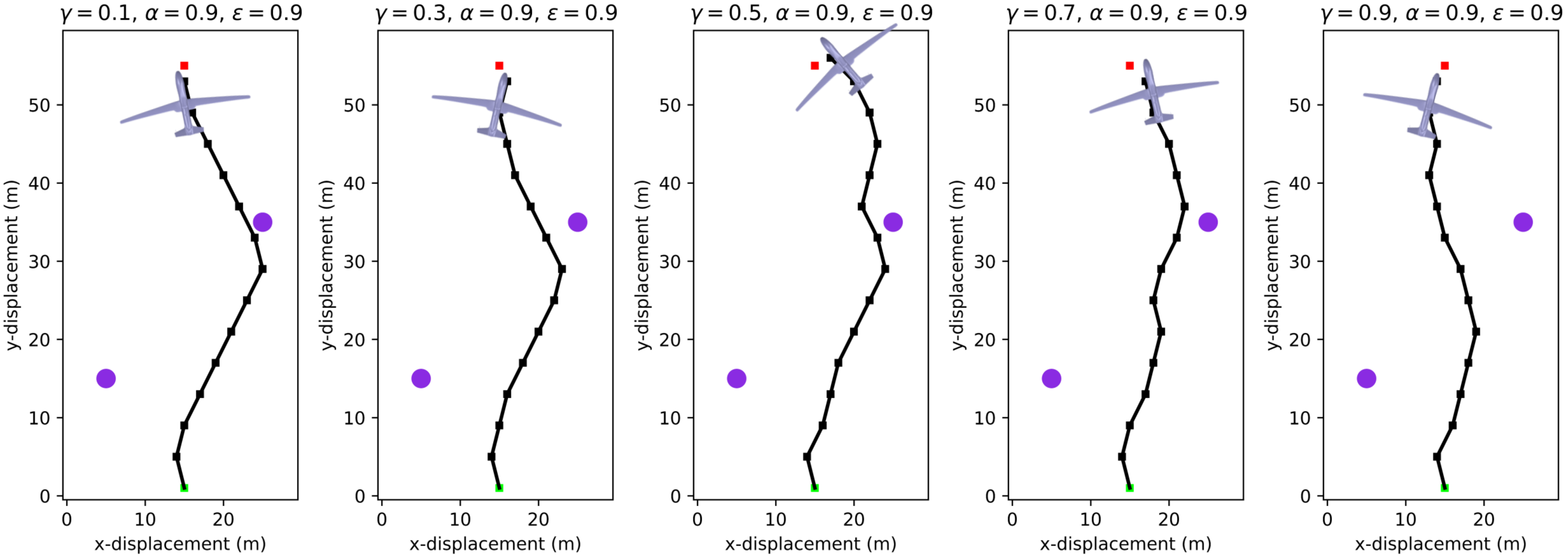
SARSA: results $\alpha = 0.5$

- As α increases, the solution converges faster but explores less



SARSA: results $\alpha = 0.9$

- As α increases, the solution converges faster but explores less (for fixed # of episodes)
- The results for $\alpha = 0.9$ are different from policy iteration

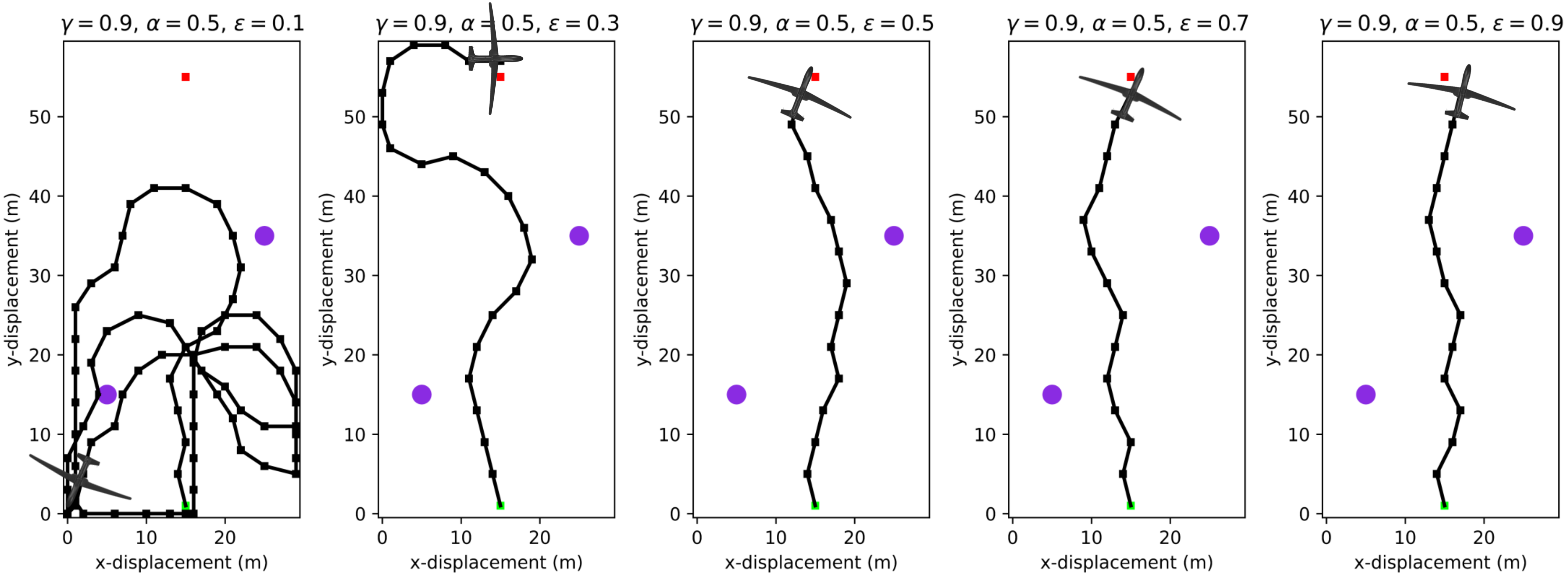


SARSA: results (and the effect of ϵ)

- Exploration vs. exploitation
- Fixed number of episodes: 10,000

$|Q_{\epsilon=0.9} - Q_{\epsilon=0.7}| = 3744$ $|Q_{\epsilon=0.7} - Q_{\epsilon=0.5}| = 5017$

$|Q_{\epsilon=0.5} - Q_{\epsilon=0.3}| = 5980$

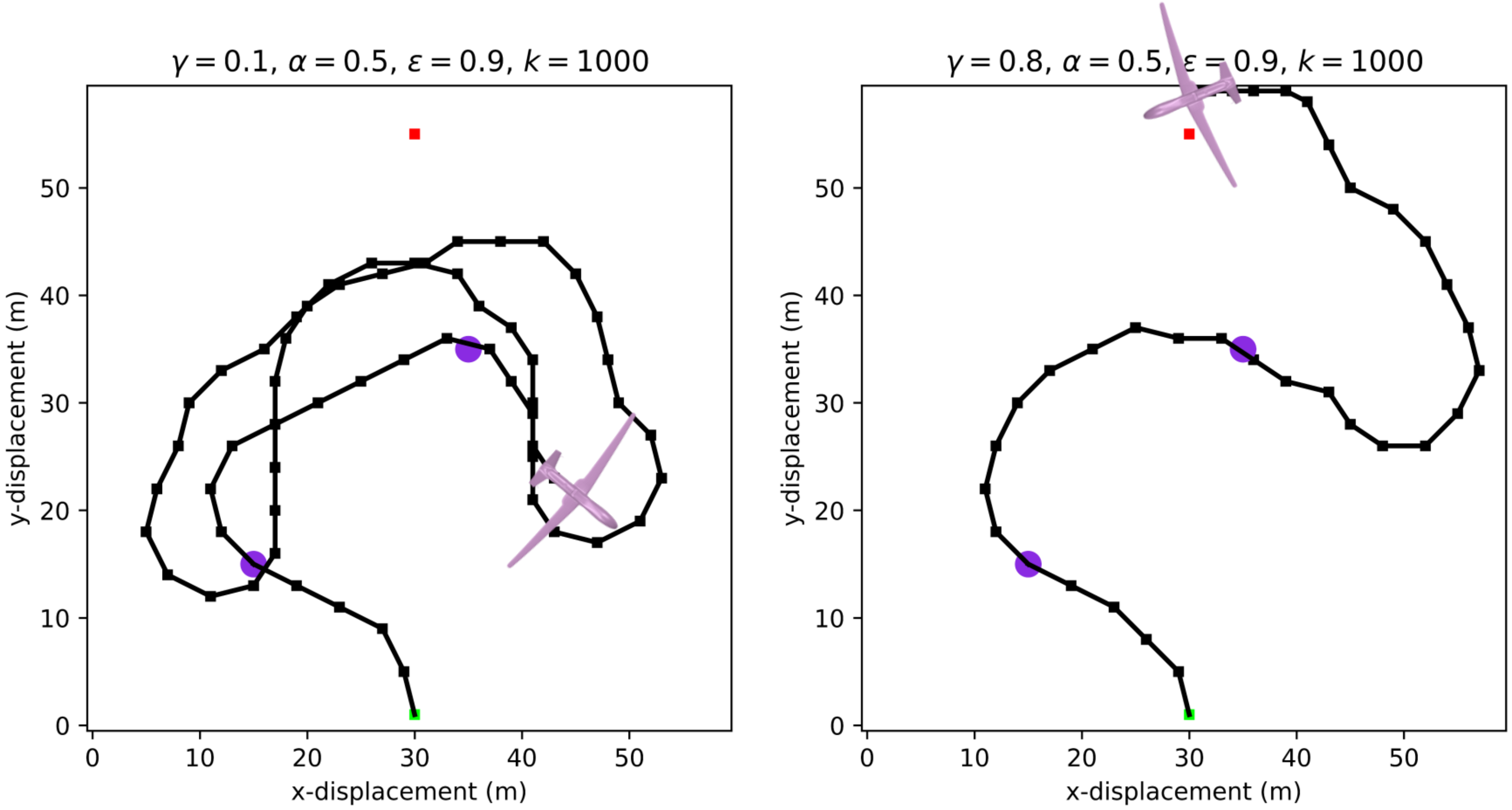


More exploration → More exploitation

Doesn't converge | Better convergence

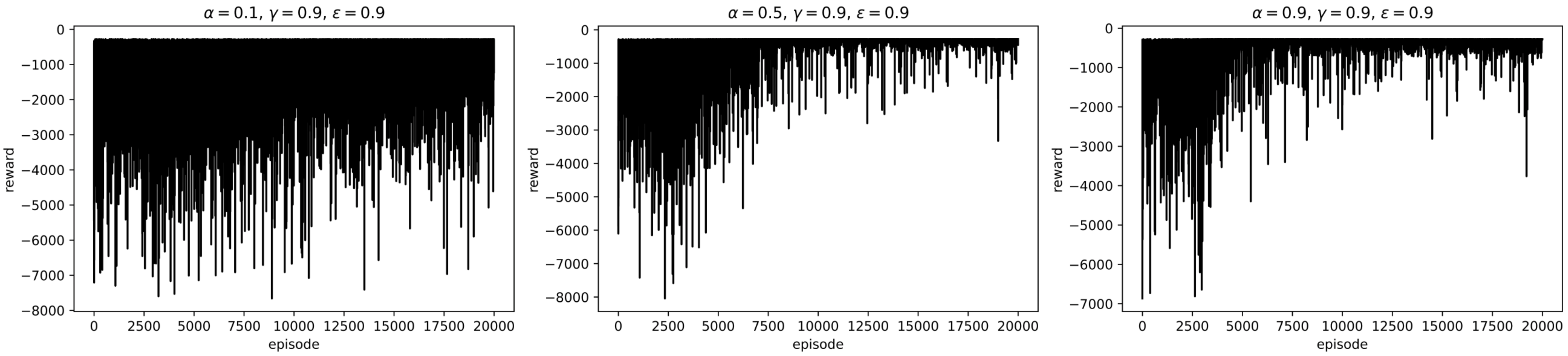
SARSA: poorly weighted rewards

- $\mathcal{R} = d_g^2 + \sum_{i=1}^{n} {}^{k=1000}/_{d_i^2}$



SARSA: convergence

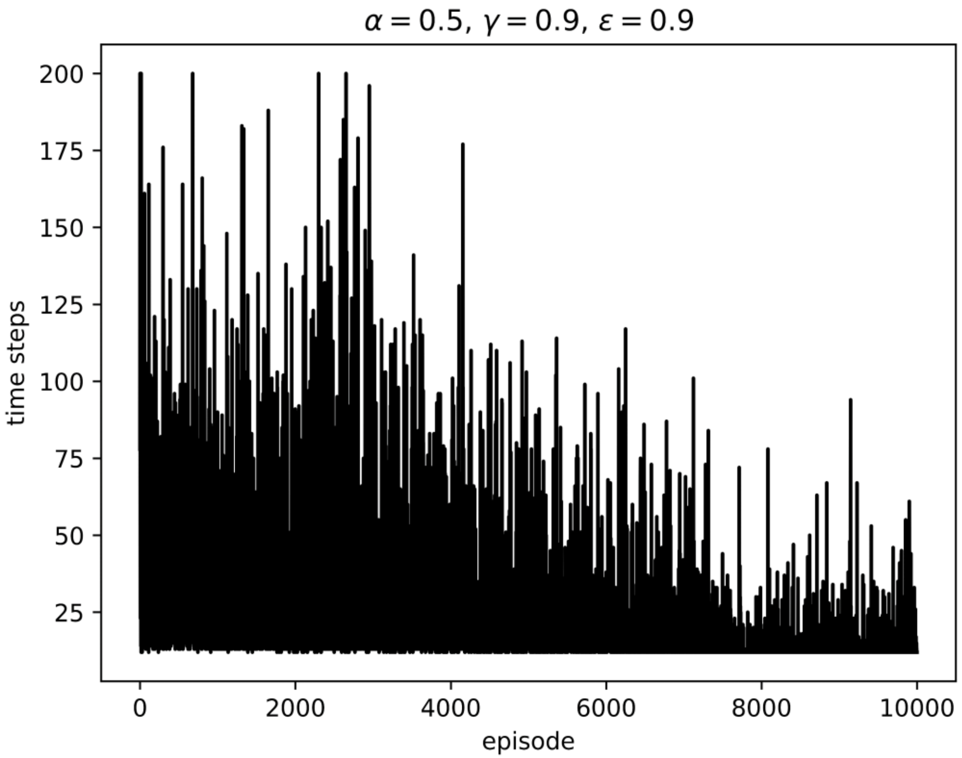
- Finishes 10,000 episodes in 65s with max time steps = 50
- ϵ-greedy policy results in “noisy” plots



Poor convergence → Better convergence

SARSA: convergence

- Plots time steps vs. episode for SARSA
- Quantifies the exploratory nature of the algorithm: as the action values converge, the number of time steps decreases



Conclusion

	Policy iteration	SARSA
Time (s)	228-389	65
Convergence	Yes	Not always (dependent on ϵ, α, γ, max episodes, and max time steps)
Qualitative analysis	Good for complex problems with local minimums	Good for large state-spaces

- *Algorithms:*
  - Policy iteration converges very reliably for large state-spaces, but is very slow
  - SARSA is generally fast, but can struggle to yield reasonable trajectories when the max episodes are capped
- *Lessons learned:*
  - Continuous dynamics & large state-spaces → slow
  - All methods are highly dependent on the reward function
    - And it's difficult to create a reward function that results in reasonable trajectories

References

Hiroshi Kawano. "Study of path planning method for under-actuated blimp-type UAV in stochastic wind disturbance via augmented-MDP". In: *2011 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. 2011, pp. 180–185. DOI: 10.1109/AIM.2011.6027018.

Shankarachary Ragi and Edwin K. P. Chong. "UAV Path Planning in a Dynamic Environment via Partially Observable Markov Decision Process". In: *IEEE Transactions on Aerospace and Electronic Systems* 49.4 (2013), pp. 2397–2412. DOI: 10.1109/TAES.2013.6621824.

Wesam H. Al-Sabban, Luis F. Gonzalez, and Ryan N. Smith. "Wind-energy based path planning for Unmanned Aerial Vehicles using Markov Decision Processes". In: *2013 IEEE International Conference on Robotics and Automation*. 2013, pp. 784–789. DOI: 10.1109/ICRA.2013.6630662.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.