Extended Abstract

Problem Terminal radar approach control (TRACON) is a demanding sequential decision problem in air traffic control. Within roughly 30–50 nautical miles of an airport, controllers issue heading, altitude, and speed commands to guide arriving aircraft from published arrival routes onto a stabilized final approach. Every command shapes not only the immediate trajectory but also eventually localizer capture and separation from other traffic, making this a long-horizon problem. Prior learning-based work tends to isolate one slice of the job such as conflict resolution, traffic separation, arrival sequencing, or trajectory prediction without modeling the full workflow from arrival route to landing. We bridge this gap by training a policy to model the end-to-end actions a controller takes during such procedures.

Approach We build a TRACON-style simulator with six STAR procedures. It takes controller-style commands and simulates aircraft dynamics, supports Instrument Landing System (ILS), and warns about the loss of separation. The policy observes a seven-dimensional state describing the aircraft's positions and states and emits controller-style heading, altitude, and speed commands. We develop the policy in four phases of increasing scope, beginning with single-aircraft landing, evaluating on both success rate and trajectory cleanliness, and building toward dense multi-aircraft coordination where landing success and separation must be handled jointly.

Single-aircraft control We first train a Gaussian Mixture Model (GMM) policy by behavioral cloning from 157 initial human-controller demonstrations. Because human demonstrations cover only a narrow band of the state space, a small drift can place the aircraft in rarely-seen states. We therefore augment the dataset with DAgger corrections, distillation from a Flow Matching teacher trained only on runway-relative positions, and Gaussian state perturbation. Even though these changes trade away raw success rate, they produce more reasonable trajectories simulating human controller behaviors and improve recovery from off-demonstration states, which is ideal for a base policy improved further using reinforcement learning.

We then fine-tune the policy with Proximal Policy Optimization (PPO) under controller-informed dense intermediate rewards including but not limited to a localizer-intercept heading term, a base-to-final turn term, a loop penalty, and a hard out-of-zone termination that keeps the policy inside an expert-derived corridor. The final policy reaches 97.1% landing success rate while keeping trajectories realistic (clearly showing downwind, base, and final legs) and clean (minimal wiggling, loops, and no corner cutting).

Multi-aircraft coordination We extend the controller to dense multi-aircraft airspace in two ways. First, we freeze the single-aircraft policy and train an auxiliary traffic-aware, zero-initialized correction head that observes a radar-style angular density of nearby traffic to augment the GMM command. The policy is trained using PPO with an additional reward penalizing the loss of separation detected by our environment. While the reactive correction shows some last-moment dodging behaviors and reduces the separation warning time by 14.7% relative to the single-aircraft policy baseline, it does not effectively prevent crashes (138 per 512 trajectories). This suggests that human intuitions about crash causation from collision-warning behavior do not transfer cleanly to the learned policy, where the deeper issue can be the lack of coordinated traffic-level behavior.

Building on this idea, we leverage cheap simulator rollouts and treat the single-aircraft GMM policy as a multimodal trajectory sampler to algorithmically plan each aircraft's next n steps by sampling candidate trajectories and re-rolling conflicted ones until a conflict-free assignment is found. The long-horizon planner is far more effective. At 400-step horizon, it eliminates crashes in the 512 rollouts and cuts average collision warning time by 71.5% relative to the baseline. In our simulator setting, longer horizons offer limited additional benefit, as a 400-step lookahead already covers the main conflict-prone regions such as downwind entry and base turns.

Discussion The results show our trained policies and algorithms capable of high-quality single-aircraft approach and resolving a majority of conflicts in multi-aircraft settings. However, the main limitations are the simplified deterministic aircraft dynamics and idealized immediate pilot responses. Furthermore, the policy applies per-timestep control for more effective training rather than mimicking the longer vectoring commands used by human controllers and could cause trajectories to wiggle unnecessarily. Future work could also explore the idea of multi-agent reinforcement learning in the multi-aircraft setting for better communications between traffic.

Reinforcement Learning for Terminal-Area Air Traffic Control

Jerry Yin*
Department of Computer Science
Stanford University
jerryyin@stanford.edu

Abstract

We study terminal radar approach control (TRACON) as a long-horizon command-generation problem and build a TRACON-style simulator in which a policy issues controller-style heading, altitude, and speed commands to guide aircraft from a Standard Terminal Arrival Route (STAR) onto final approach. We approach the problem in four phases. First, we start with single-aircraft commands by training a Gaussian Mixture Model policy using behavioral cloning from human demonstrations, DAgger corrections, Flow Matching distillation, and state perturbation. Second, we fine-tune this policy with Proximal Policy Optimization (PPO) under controller-informed dense rewards, reaching a 97.1% single-aircraft landing success rate while preserving realistic approach patterns. Third, we extend to dense multi-aircraft airspace with an additional traffic-aware correction head that observes a radar-style angular density of nearby traffic and reduces loss-of-separation time by 14.7%, though such reactive heuristics do not reliably prevent crashes. Finally, we use the single-aircraft GMM policy as a trajectory sampler in a long-horizon planning algorithm that resolves conflicts over a fixed planning window, eliminating crashes and cutting average collision-warning time by 71.5% relative to the baseline.

1 Introduction

Terminal radar approach control (TRACON) is a high-stakes sequential decision-making problem in air traffic control. In terminal airspace, typically within roughly 30–50 nautical miles of an airport, controllers must continuously issue heading, altitude, speed, and landing-clearance commands to guide aircraft from arrival routes onto final approach. Each command affects not only the current aircraft trajectory, but also future localizer capture and separation from other aircraft. This makes TRACON control a natural reinforcement learning problem: the controller observes the evolving traffic state, chooses commands over a long horizon, and must optimize both landing success and operational safety.

Prior learning-based work in air traffic control often studies important but isolated subproblems such as conflict detection, collision avoidance, arrival sequencing, and trajectory prediction. While these settings capture parts of the controller's job, they do not fully model the full TRACON workflow. In particular, a sequencing model may decide which aircraft should land first without offering radar vectors that make the sequence feasible, while a conflict-resolution model may avoid near-term separation violations without guiding aircraft all the way to a stabilized approach.

In this project, we study a controller-centered formulation of terminal radar approach control that addresses this gap. We build a simulated TRACON environment in which a policy issues controller-style heading, altitude, and speed commands to guide aircraft from arrival routes to final approach. In the single-aircraft setting, policies are evaluated by whether they capture the localizer and glide slope within a fixed time limit while producing approach trajectories that resemble human controller patterns, such as downwind legs followed by base turns onto final. In the multi-aircraft setting, the objective additionally includes reducing collision-warning time and crashes under dense traffic. The project therefore begins with single-aircraft landing as a long-horizon control problem, then extends to multi-aircraft coordination where landing success and separation must be handled jointly.

*Independent work. Claude Code was used to help build the environment and visualize results.

The main contributions of this paper include:

- A TRACON-style simulator that supports Standard Terminal Arrival Route (STAR) procedures, realistic controller commands including heading, altitude, speed, landing clearance, go-around, and holding instructions, ILS approach behavior, and multi-aircraft conflict monitoring.
- A Gaussian Mixture Model (GMM) policy through behavioral cloning and Proximal Policy Optimization (PPO) that commands a single aircraft from spawn to landing and succeeds more than 97% of the time while producing trajectories that resemble human controller behavior.
- An auxiliary GMM correction head that supplements the base GMM policy with traffic-aware command adjustments and reduces collision-warning time by nearly 15% in dense multi-aircraft airspace.
- A long-horizon planning algorithm that leverages probabilistic trajectory sampling from the base GMM policy, drastically reduces collision-warning time, and completely avoids crashes.

2 Related Work and Technical Background

2.1 Machine Learning for Air Traffic Control

Recent learning-based work in air traffic control has studied tactical conflict resolution, autonomous aircraft separation, arrival sequencing, and trajectory prediction. These problems capture important parts of controller decision-making, but they usually isolate one subtask rather than modeling the full terminal radar workflow from arrival route to final approach and landing.

Sui et al. (2023) study tactical conflict resolution with a deep reinforcement learning conflict solver. Their agent operates in a simulated airspace environment and selects finite ATC-like actions, including heading, speed, and altitude adjustments, to resolve conflicts. This is close to the conflict-avoidance part of our setting, but the task is still centered on resolving tactical conflicts rather than producing a full sequence of TRACON vectoring commands from STAR arrival to localizer capture and landing.

Brittain et al. (2020) study autonomous separation assurance with deep multi-agent reinforcement learning in BlueSky. Their method uses distributed aircraft agents to maintain separation in dense airspace, making it relevant to our multi-aircraft setting. However, their goal is separation assurance, not terminal approach control, so the agents are not responsible for landing the aircraft.

Du et al. (2023) study arrival sequencing and scheduling using historical terminal-area operation data. Their method predicts estimated arrival times and optimizes landing sequences and scheduled arrival times. This addresses the question of which aircraft should land when, but not how to issue controller-style commands that make the sequence feasible in the radar environment.

Barratt et al. (2019) learn probabilistic trajectory models of aircraft in terminal airspace from real radar position data. Their work differs from ours in the observation setting. By learning from recorded position tracks, the model reflects a more realistic partial observability of aircraft behavior rather than assuming access to the full simulator state and controller intent and leads to a trajectory-modeling problem to capture and predict realistic aircraft motion from observed data. On the other hand, our work assumes a simplified fully observable simulator state to study the more direct command-generation problem to produce complete approach trajectories.

Overall, prior work addresses important pieces of the ATC problem while our work combines these concerns in a controller-centered TRACON simulator where the agent must operate like a controller and guide aircraft through approach procedures while handling multi-aircraft conflicts.

2.2 Flow Matching

Flow Matching by Lipman et al. (2023) learns a continuous transformation from a simple noise distribution to the expert action distribution. Given an expert action a_1 and a noise sample a_0, the model is trained to predict the velocity along the interpolation path

$$a_\tau = (1-\tau)a_0 + \tau a_1, \qquad \tau \in [0,1].$$

The training loss is

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}\left[\|v_\theta(s, a_\tau, \tau) - (a_1 - a_0)\|_2^2\right].$$

At inference time, the learned velocity field is integrated from noise to action. In this project, Flow Matching is useful because it models multimodal controller actions, but it does not provide a convenient closed-form action likelihood for PPO.

2.3 Gaussian Mixture Model

We use a conditional GMM policy from Viroli and McLachlan (2017) to obtain an explicit action density. Given state s, the policy is

$$p_\theta(a \mid s) = \sum_{k=1}^{K} \pi_k(s) p_k(a \mid s).$$

In our setting, we use separate components for each dimension in aircraft commands. More specifically, we use von Mises distribution modeled heading and Gaussian modeled altitude and speed:

$$p_k(a \mid s) = \mathrm{vM}(\theta; \mu_k, \kappa_k)\mathcal{N}(h; \mu_k^h, (\sigma_k^h)^2)\mathcal{N}(v; \mu_k^v, (\sigma_k^v)^2).$$

The GMM is trained by negative log-likelihood and provides the $\log \pi_\theta(a_t \mid s_t)$ needed for PPO.

2.4 Proximal Policy Optimization

PPO from Schulman et al. (2017) is an actor-critic reinforcement learning method that updates the policy while limiting how far the new policy moves from the rollout policy. The core update uses the probability ratio

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t \mid s_t)}$$

and the clipped surrogate objective

$$\mathcal{L}_{\mathrm{actor}} = -\mathbb{E}_t \left[\min \left(r_t \hat{A}_t, \mathrm{clip}(r_t, 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right].$$

A critic estimates values for advantage computation, and an entropy bonus is added to encourage exploration. PPO is useful in our setting because simulator rollouts are cheap while human demonstrations are costly.

3 Simulator and Problem Setting

3.1 TRACON Simulator

We build a TRACON-style radar simulator (Fig.1) for terminal approach control. Aircraft enter the terminal area on one of six simulated STAR procedures (NORTH1-3 and SOUTH1-3, where procedure 1's spawn on downwind, 2's joining downwind, and 3's entering straight in) and continue following the published route unless the policy issues vectoring commands. The simulator supports controller-style instructions including target heading, altitude, and speed commands.

The simulator is deterministic given the current state and command, so each transition follows

$$(s_t, a_t) \mapsto s_{t+1}$$

assuming immediate pilot response to issued commands and uses a uniform aircraft dynamics model across all aircraft. Thus, the simulator captures the command-response structure of terminal control, but does not model stochastic pilot delay or aircraft-specific dynamics such as differences between heavy and light aircraft.

The simulator also enforces instrument landing system (ILS) constraints. A trajectory is considered successful only if the aircraft captures the localizer within an acceptable heading range and captures the glide slope from below. Failed outcomes include timeout, defined as failing to capture the localizer within 1200 steps for longer STAR procedures or 500 steps for straight-in approaches; improper exit, where the aircraft leaves the radar screen; localizer capture above the glide slope; and collision in the multi-aircraft setting.

3.2 State and Action Space

For each aircraft, the policy observes a seven-dimensional state:

$$s_t = (a_{\mathrm{nm}}, c_{\mathrm{nm}}, d_{\mathrm{thr}}, h_t/1000, (v_t - 200)/100, \sin\theta_t, \cos\theta_t).$$

The first three dimensions are runway-relative geometry: along-track position, cross-track deviation from the runway centerline, and distance to threshold. The remaining dimensions describe current altitude, speed, and heading after normalization.

The reinforcement learning action is a controller-style command:

$$a_t = (\theta_t^{\mathrm{cmd}}, h_t^{\mathrm{cmd}}, v_t^{\mathrm{cmd}}).$$

Note that we are using sine-cosine encoding avoids discontinuities in heading near 0° and 360°.

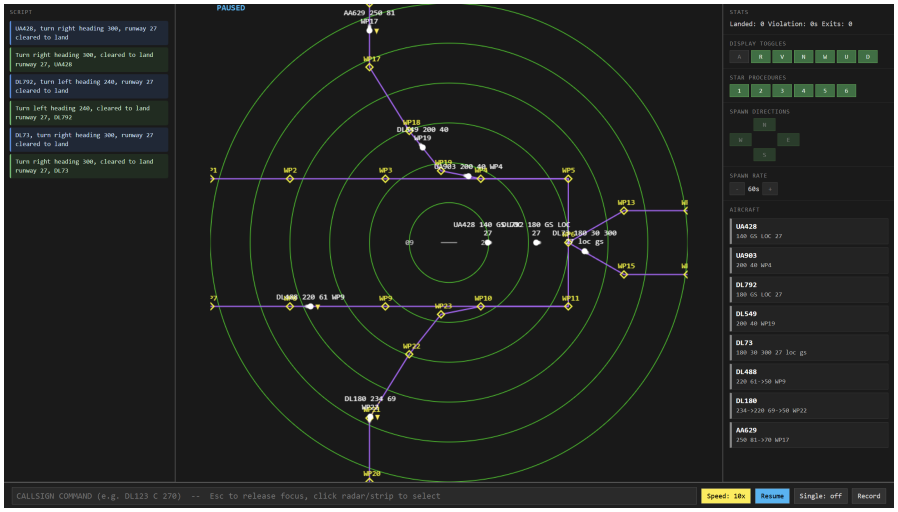


Figure 1: Simulator Screenshot

4 Phase I: Behavioral Cloning from Augmented Data

We first train a Gaussian Mixture Model policy with behavioral cloning. The goal of this phase is not to fully solve the single-aircraft landing task, but to initialize reinforcement learning with a controller-like policy that can already produce some successful landing trajectories.

4.1 Data Collection

We train behavioral cloning policies from three data sources. The initial dataset contains 157 expert human aircraft trajectories collected by one controller, totaling 134,602 state-action pairs. During iterative model development, we add 97 DAgger correction trajectories with 35,166 rows. We also use a trained Flow Matching policy to generate 72 additional successful trajectories with 63,352 rows for distillation. The final shipped BC model is trained on all three sources.

4.2 Methodology

The BC policy maps the 7-D aircraft state to the supervised command target

$$y_t = (\sin\theta_t^{\mathrm{cmd}}, \cos\theta_t^{\mathrm{cmd}}, h_t^{\mathrm{cmd}}/1000, (v_t^{\mathrm{cmd}} - 200)/100).$$

Due to the human trajectories covering a narrow subset of the 7-D state space, a small drift in aircraft state can place the aircraft in states rarely seen during training (see figure 5). DAgger helps with some off-trajectory positions, but cannot cover the continuous combinations of aircraft-state deviations.

We therefore use two additional data augmentation mechanisms. First, we train a 3-D Flow Matching policy on only runway-relative position,

$$(a_{\mathrm{nm}}, c_{\mathrm{nm}}, d_{\mathrm{thr}}),$$

and use its successful rollouts as additional distilled data. Second, we perturb aircraft-state inputs during GMM training:

$$\epsilon_\psi \sim \mathcal{N}(0, 15^\circ), \quad \epsilon_h \sim \mathcal{N}(0, 200\ \mathrm{ft}), \quad \epsilon_v \sim \mathcal{N}(0, 10\ \mathrm{kt}),$$
$$\psi \leftarrow \psi + \epsilon_\psi, \qquad h \leftarrow h + \epsilon_h, \qquad v \leftarrow v + \epsilon_v.$$

The target command y_t is unchanged, so the model learns recovery behavior around the demonstrated trajectories.

All GMM models use a two-layer MLP encoder with hidden size 64 and a $K = 4$ mixture head:

$$p(a \mid s) = \sum_{k=1}^{4} \pi_k(s)\mathrm{vM}(\theta; \mu_k, \kappa_k)\mathcal{N}(h; \mu_k^h, (\sigma_k^h)^2)\mathcal{N}(v; \mu_k^v, (\sigma_k^v)^2).$$

We train with negative log-likelihood, state perturbation, input dropout of 0.1, and a $200\times$ dataset repeat. The final recipe uses 25 epochs, batch size 1024, learning rate 3×10^{-4}, and gradient clipping at 1.

4.3 Experiments and Results

We evaluate each BC model on 200 rollouts per STAR, for 1200 total single-aircraft cases. A rollout succeeds only if the aircraft captures the localizer at or below the glide slope. We report success rate, percentage of trajectories staying inside the "green zone", percentage of trajectories containing loops (lower the better), and improper-exit percentage (lower the better). Note that green zone is a STAR-specific corridor based on expert trajectories; higher green-zone percentage indicates trajectories closer to human-controller behavior. We also qualitatively examine the trajectories for any unreasonable behaviors such as wiggling.

Table 1: Behavioral cloning ablation under the same 1200-case single-aircraft evaluation

Model	SR (%)	Green (%)	Loop (%)	Improper (%)
Flow Matching teacher	38.3	75.5	0.0	0.0
GMM, human only	51.0	61.4	5.6	6.2
GMM, human + DAgger	47.2	58.1	3.5	3.2
GMM, human + DAgger + Distillation	38.0	64.4	0.2	0.0

The trajectories are visualized under the Appendix Fig. 6.

4.4 Evaluation

The ablation study results show that both DAgger and distillation are useful in regularizing the trajectories for our training recipe, even though they hurt the overall success rate. DAgger provides correction trajectories that help the model recover from off-demonstration positions and reduces loops and improper exits, while adding flow matching further improves trajectory cleanliness. The all-data GMM used as the base model for further training has almost no loops, and show realistic behaviors of turning base at various downwind positions. Even though it has a lower overall success rate fully improvable by PPO, it does provide higher quality success trajectories and more stable approaches.

5 Phase II: Reinforcement Learning for Single-Aircraft Control

We further fine-tune the GMM behavioral cloning policy using PPO to improve landing success rate while preserving the reasonable controller-like trajectory structure learned during behavioral cloning.

5.1 Method

5.1.1 Reward Shaping

Terminal rewards alone provide weak credit assignment because episodes can exceed 1000 steps. In addition, the GMM initialization has low success rate, so early PPO can accidentally reinforce rare successful but undesirable behaviors such as loops, shortcuts, or unstable final approaches. We therefore add controller-informed dense rewards.

The reward consists of terminal success/failure rewards, a localizer-intercept heading reward, a one-shot base-to-final turn reward, a loop penalty, time pressure, and a hard out-of-zone termination. If an aircraft remains outside the green zone for too many consecutive policy steps, the rollout is also terminated as a failure.

Let

$$R_{\text{base}} = R_{\text{term}} + R_{\text{hdg}} + R_{\text{final}},$$

where R_{term} is $+10$ for success and -10 for failure, $R_{\text{hdg}} \in [-2, 1]$ rewards desirable localizer-capture headings, and $R_{\text{final}} \in [0.1, 0.5]$ rewards plausible base-to-final turn commands. The final run uses two phases where Phase 1 uses a stricter out-of-zone threshold to force the policy into the expert corridor. Phase 2 relaxes this threshold to allow more varied but still acceptable trajectories, and adds per-STAR success scaling so that the policy does not sacrifice harder STARs while improving easier ones. Full reward specifications are given in Appendix B.1.

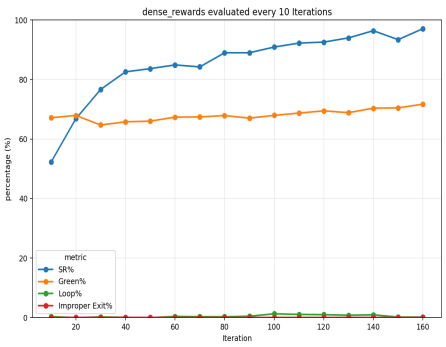
5.1.2 PPO Training

The actor is initialized from the 7-D GMM behavioral cloning policy and remains trainable. The critic is an independent network with two 64-dimensional hidden layers for scalar value estimation.

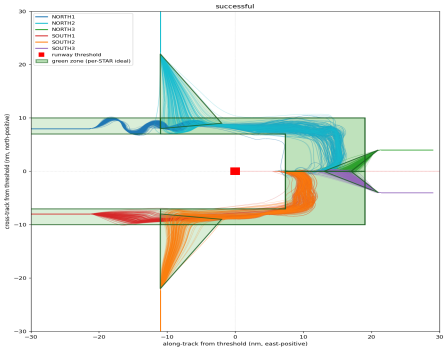
We train following the PPO loss specified in 2.4 with discount $\gamma = 0.999$, GAE parameter $\lambda = 0.95$, PPO clip $\epsilon = 0.2$, 512 complete rollouts per iteration, 4 update epochs, minibatch size 256, target KL 0.01, value coefficient $c_v = 0.5$, and max gradient norm 0.5. The actor and critic use AdamW optimizers with learning rates 10^{-6} and 3×10^{-6}. The entropy coefficient is 0.05 in Phase 1 and 0.10 in Phase 2. We keep the best performing checkpoint at iteration 160.

5.2 Results and Evaluation

We evaluate the final PPO policy using 200 rollouts per STAR, for 1200 single-aircraft evaluation cases. We report success rate, green-zone percentage, loop percentage, and improper-exit percentage.



Training metrics



PPO trajectories

Metric	Value
SR	97.1%
Green	71.7%
Loop	0.2%
Improper	0.0%

Metrics

Figure 2: Single-aircraft PPO Training, Trajectories, and Evaluation

The final PPO policy reaches 97.1% success while keeping loop rate and improper exits near zero. The green-zone metric remains stable during training, indicating that the policy does not improve success by deviating from expert demonstrations. Qualitatively, even though there are some wiggles during downwind legs, the trajectories show clear downwind patterns and varied timing in turning base. The full reward-component ablation study is included in Appendix B, and a contrasting high success rate, low pattern retainment policy is visualized in Fig. 8.

6 Phase III: Traffic-Aware Reinforcement Learning for Multi-Aircraft Control

6.1 Correction Head Method

The single-aircraft policy controls each plane in isolation, with no notion of surrounding traffic. We freeze it and train a lightweight correction head that observes nearby traffic and nudges the GMM command. Traffic is encoded as a radar-style angular density in the ego body frame (0° forward, +90° right): the bearing circle is split into $N = 36$ bins of 10° with centers $c_i \in \{-175°, \ldots, 175°\}$, and a neighbor at relative bearing β, lateral distance $d \le 10$ nm contributes a proximity-weighted Gaussian smear, max-aggregated per bin:

$$\rho_i = \max_{\text{neighbors}} \underbrace{\max\!\left(0,\, 1 - \tfrac{d}{10}\right)}_{\text{proximity}} \cdot \underbrace{\exp\!\left(-\tfrac{(c_i - \beta)^2}{2\sigma^2}\right)}_{\sigma = 12°} \in [0, 1], \qquad \Delta\rho = \rho_t - \rho_{t-1}, \tag{1}$$

where $c_i - \beta$ is taken modulo 360° and $\Delta\rho$ is a temporal channel to encode if the neighboring aircraft is flying towards or away the ego aircraft. A visualized radar head can be seen in Fig.7. The full state stacks ego state with the two density channels:

$$s_{\text{full}} = [\, s_{\text{ego}} \in \mathbb{R}^7,\ \rho \in \mathbb{R}^{36},\ \Delta\rho \in \mathbb{R}^{36}\,] \in \mathbb{R}^{79}. \tag{2}$$

The head is a two-hidden-layer MLP ($79 \to 64 \to 64$) mapping s_{full} to a 3-D diagonal Gaussian over command deltas $\Delta = (\Delta\text{hdg}, \Delta\text{alt}, \Delta\text{spd})$, with mean $\tanh$-bounded to $\pm(30°, 1\,\text{kft}, 30\,\text{kt})$. The sampled correction is added to the frozen GMM mode a_{GMM} to form the applied command:

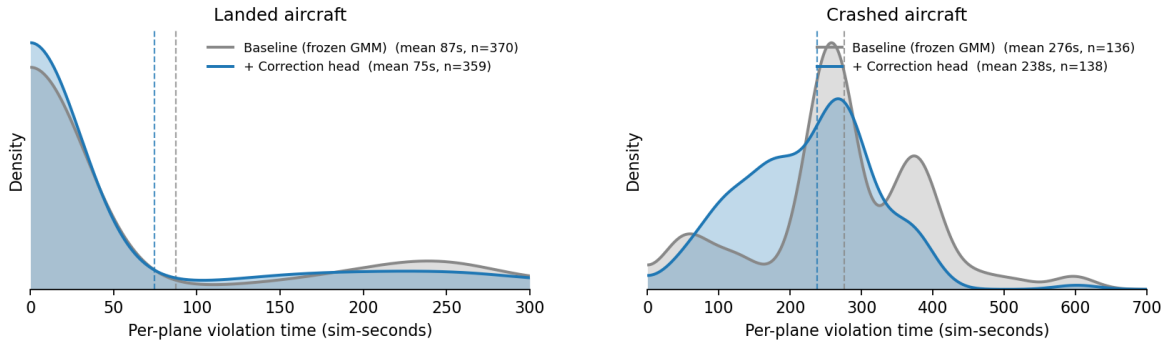
$$\Delta \sim \mathcal{N}\big(\mu(s_{\text{full}}),\, \sigma(s_{\text{full}})^2\big), \qquad a = a_{\text{GMM}} + \Delta. \tag{3}$$

We zero-initialize the mean head so that the training departs from the exact GMM baseline; only the head and a fresh critic ($79 \to 64 \to 64 \to 1$) are trained.

Training is done in two stages. Stage 1 uses single-aircraft scenes with the radar input zeroed, keeping the head near identity before it sees traffic; Stage 2 fine-tunes in multi-aircraft scenes under the Phase II reward plus a

Table 2: Multi-aircraft evaluation over 512 trajectories

	SR	Crashes	Avg. violation (s)	Violation (s) std.
Baseline (frozen GMM)	72.3%	136	137.7	151.3
+ Correction head	70.1%	138	117.5	139.8



Landed aircraft Crashed aircraft

Figure 3: Density of average collision warning time.

per-plane penalty of 0.3 per second inside the 2 nm / 1000 ft warning envelope. PPO reuses the Phase II settings, except entropy is lowered to 0.002 and the learning rate raised to 10^{-5}. We evaluate in one continuously-running simulator spawning an aircraft every 90 s until 512 trajectories terminate, reporting per-plane success rate, crash count, and average violation in seconds.

6.2 Results and Evaluation

The head does not prevent crashes but cuts loss of separation: with comparable policy success rate and crash rate, the average violation-seconds fall 14.7% (Table 2) across the outcomes for both landed and crashed aircraft (Fig. 3).

In the qualitative example shown by Fig. 7, the new head is able to identify the traffic as it turns base and successfully sequence DL316 onto final, whereas a replay using the baseline model fails to do so and creates a conflict. However, in most cases the new model is only able to prevent the aircraft from flying into traffic rather than resequencing it in previous steps such as extending the downwind. This shows that human separation heuristics do not always transfer cleanly: the head meets its training objective and lowers violation-seconds, yet reducing separation loss does not prevent crashes, but performs last-moment avoidance. In a dense airspace, the lack of long-horizon planning limits the effectiveness of the model.

7 Phase IV: Long-Horizon Flight Planning for Sequencing

In this phase, we further leverage the single-aircraft mixture model's capability to sample varied trajectories and the costless simulator roll-outs to plan each plane's next n steps ahead to avoid conflicts. Rather than training a reactive policy, this algorithmically evaluates multi-aircraft interactions.

7.1 Methodology

In this algorithm, we maintain a flight plan for each aircraft. Whenever any aircraft's plan is depleted (or entering the airspace without the flight plan), we extend each aircraft's flight plan to the next n steps. If there exists conflicts in the horizon, we re-roll the conflicted trajectories until the conflict is resolved or the budget (9 re-rolls maximum in our scenario, 3 in a batch) is reached. In case the budget is reached without conflicts resolved, each affected aircraft would pick the trajectory that maximizes the minimum separation. Algorithm 1 summarizes the procedure and the visualization is shown by Fig. 4, where the blue lines show flight plans up to 400 steps.

We evaluate under the same multi-aircraft protocol as stated in 6.1 testing the algorithm on horizons $n \in \{100, 200, 300, 400, 500\}$, plus a *Full* setting in which each candidate is rolled all the way to landing and only successful (landed) trajectories are accepted as plans.

Horizon	SR	Crashes	Avg. viol. (s)	Viol. std.
100	96.9%	12	118.2	134.6
200	97.7%	8	75.8	126.5
300	98.0%	**0**	59.2	114.1
400	97.9%	**0**	**39.3**	100.5
500	97.3%	2	59.9	120.5
Full	**98.8%**	2	52.5	110.5

Table 3: Multi-aircraft evaluation across planning horizons

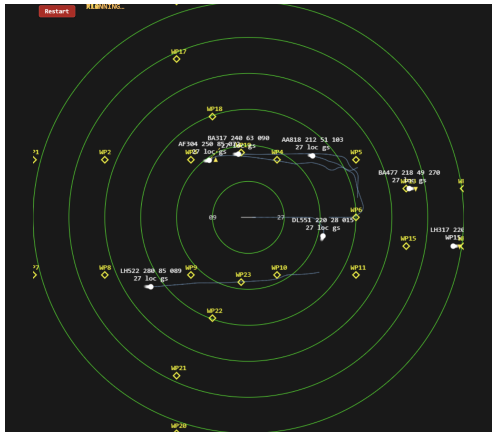


Figure 4: Long-Horizon Planning (400 Steps)

7.2 Results and Evaluation

Planning is highly effective at reducing collisions and loss of separation (Table 3) with no crashes and cutting the average violation-seconds by 71.5% compared to the baseline. This reflects that the GMM's action distribution is not mode-collapsed and the same policy could yield genuinely different trajectories.

Furthermore, longer horizons could have diminishing returns. A 300–400-step plan already spans the decisions that matter, so the gains are minimal beyond it. *Full* even performs worse off since committing every plane to a one-shot landing in dense airspace forces plans that still cross others. The shorter horizon also accidentally exploited the GMM model into occasional holding loops (rare failure trajectories) that happen to keep a plane clear of traffic.

8 Conclusion

This project studies terminal-area air traffic control as a command-generation problem in a custom TRACON-style simulator. We first train a GMM policy from human demonstrations and show the effectiveness of augmentation methods such as DAgger corrections, Flow Matching distillation, and state augmentation. Then, the model is fine-tuned using PPO with dense heuristic rewards to effectively produce high-success single-aircraft control while preserving realistic approach patterns.

In the multi-aircraft setting, we train a reactive radar correction head to reduce loss of separation, but such heuristics do not reliably translate to crash prevention. In contrast, long-horizon flight-plan search utilizes the GMM policy as a trajectory sampler and achieves the strongest multi-aircraft performance that prevents collisions and minimizes loss of separation.

9 Discussion

The main limitation is the trade-off between modeling simplicity and realism. We simplify the pilot's behavior by immediately responding to the commands, and the dynamics are set for every aircraft regardless of their type. This helps formulate a deterministic transition $s' \leftarrow (s, a)$ for easier modeling and training. In addition, we treat STAR commands and human vectoring commands in the same way due to actual human commands, mostly taking over from the flight plan for base and final turns, making up less than 1% of total commands. Even though this provides dense supervision every time-step, the work does not represent realistic controller-style long vectors and can cause trajectories to wiggle at every step. Lastly, for simplicity and ease of modeling, the action setup also assumes landing clearance given to all traffic in the beginning and does not enforce any going around actions.

For future work, the multi-aircraft phases show that in order for the extended model to avoid collisions, it needs a better description of the state of other traffic similar to how long-horizon planning algorithmically resolves all flight plans. Therefore, more investigations could be done on using multi-agent reinforcement learning or a centralized model as an addition to single-aircraft policies for better sequencing and collision prevention.

References

Shane T. Barratt, Mykel J. Kochenderfer, and Stephen P. Boyd. 2019. Learning Probabilistic Trajectory Models of Aircraft in Terminal Airspace From Position Data. *IEEE Transactions on Intelligent Transportation Systems* 20, 9 (2019), 3536–3545. doi:10.1109/TITS.2018.2877572

Marc Brittain, Xuxi Yang, and Peng Wei. 2020. A Deep Multi-Agent Reinforcement Learning Approach to Autonomous Separation Assurance. arXiv:2003.08353 [cs.LG] https://arxiv.org/abs/2003.08353

Zhuoming Du, Junfeng Zhang, and Bo Kang. 2023. A Data-Driven Method for Arrival Sequencing and Scheduling Problem. *Aerospace* 10, 1 (2023). doi:10.3390/aerospace10010062

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. 2023. Flow Matching for Generative Modeling. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=PqvMRDCJT9t

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG] https://arxiv.org/abs/1707.06347

Dong Sui, Chenyu Ma, and Chunjie Wei. 2023. Tactical Conflict Solver Assisting Air Traffic Controllers Using Deep Reinforcement Learning. *Aerospace* 10, 2 (2023). doi:10.3390/aerospace10020182

Cinzia Viroli and Geoffrey J. McLachlan. 2017. Deep Gaussian Mixture Models. arXiv:1711.06929 [stat.ML] https://arxiv.org/abs/1711.06929

A Additional Visualizations

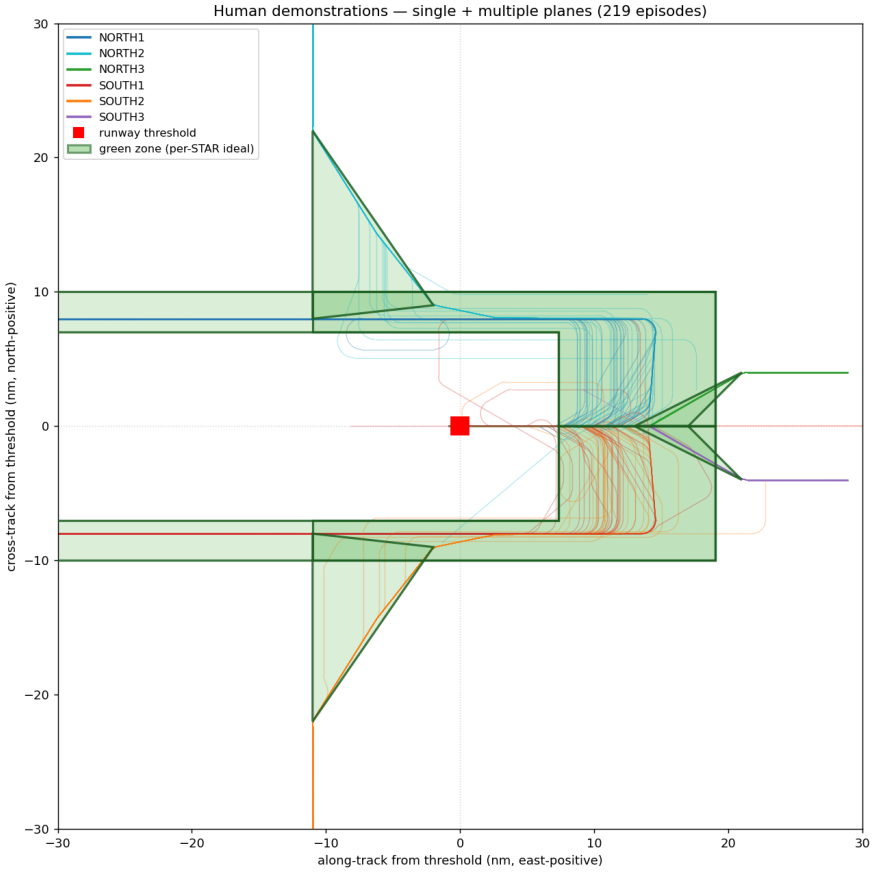
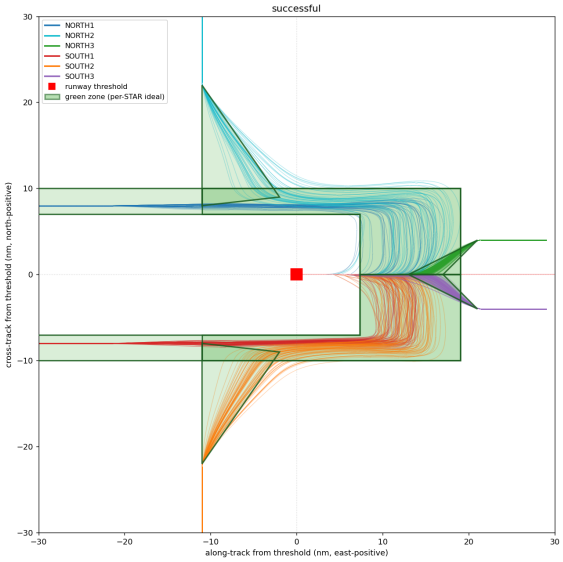
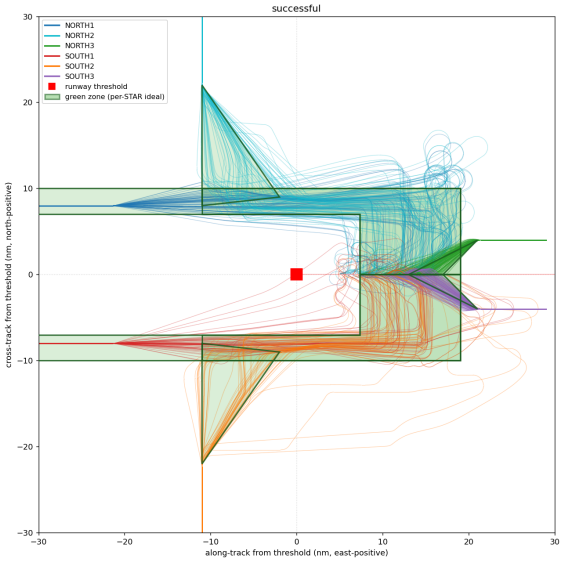


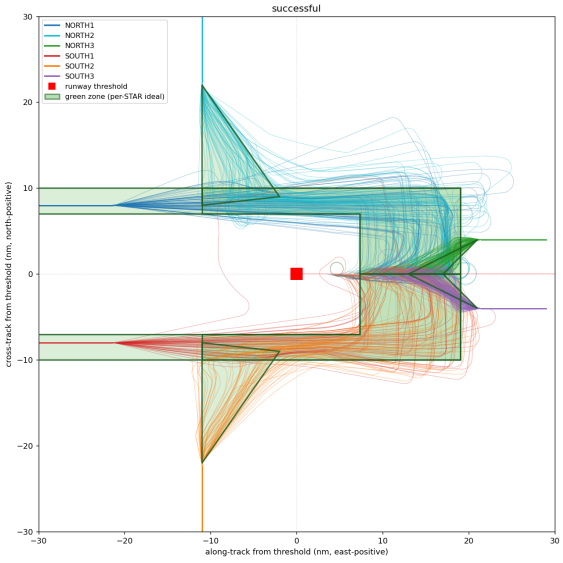
Figure 5: Collected Human Trajectories



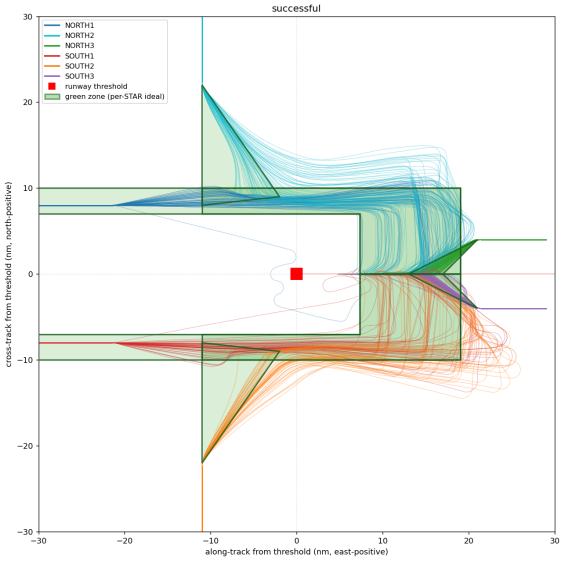
(a) Flow Matching: clean and efficient success trajectories



(b) GMM with human data only: messy with loops

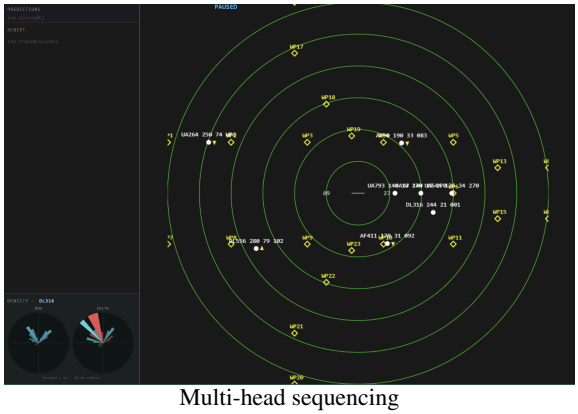


(c) GMM with DAgger: cleaner but still with unnecessary loops

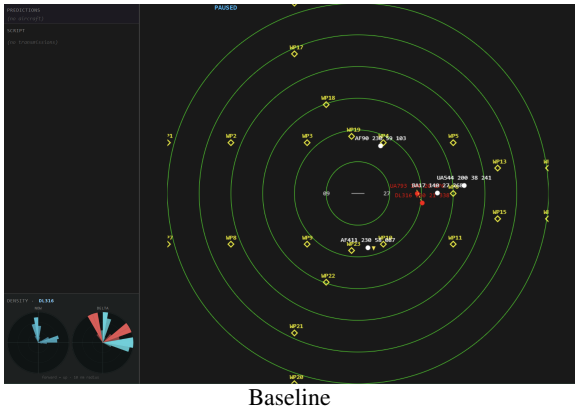


(d) GMM with full data: cleanest GMM trajectories with varied realistic behaviors

Figure 6: Successful trajectory visualizations for Flow Matching and GMM behavioral cloning variants.



Multi-head sequencing



Baseline

Figure 7: Multi-head Sequencing vs. Baseline

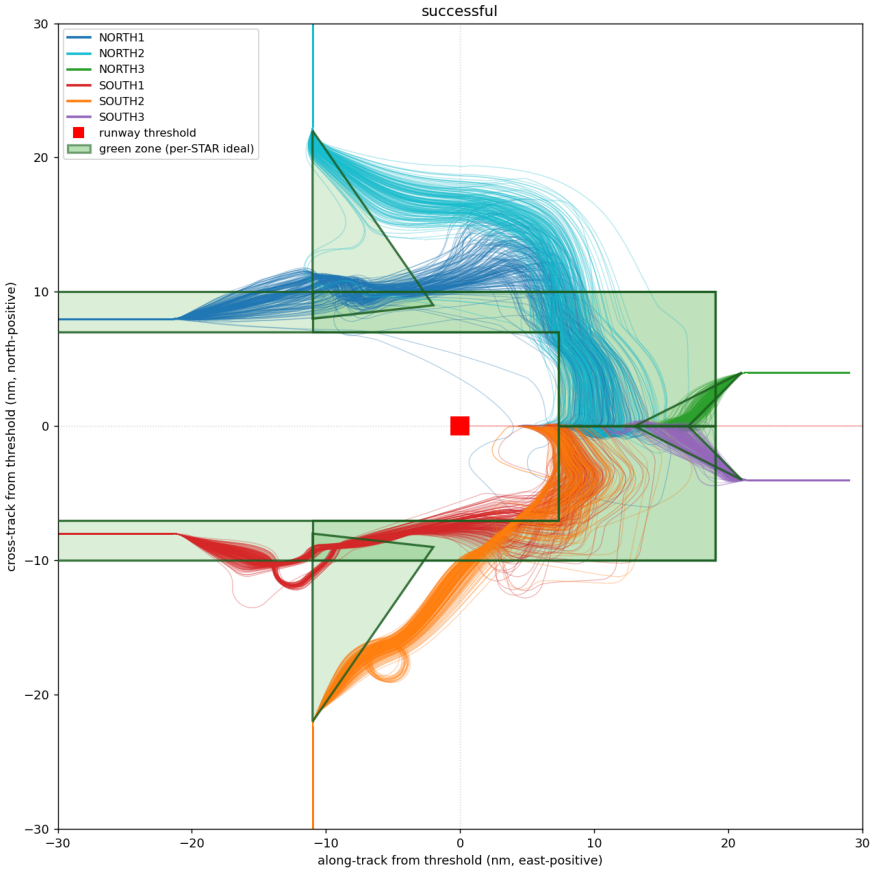


Figure 8: High SR, Low Quality Policy (`continuous_01`) from the ablation studies

B PPO Reward and Ablation Studies

B.1 Single-Aircraft PPO Reward Shaping

Following the state s_t, action a_t, and simulator outcome definitions in 3.2, we restore the aircraft heading from the state as

$$\theta_t = \operatorname{atan2}(\sin\theta_t, \cos\theta_t),$$

and the commanded heading from the action as

$$\theta_t^{\mathrm{cmd}} = \operatorname{atan2}(\sin\theta_t^{\mathrm{cmd}}, \cos\theta_t^{\mathrm{cmd}}).$$

Let $\mathrm{dir}(\mathrm{STAR}) \in \{\mathrm{N}, \mathrm{S}\}$ indicate whether the STAR starts from the north or south. The ideal localizer-intercept headings are

$$\theta^\star = \begin{cases} 240^\circ, & \mathrm{dir}(\mathrm{STAR}) = \mathrm{N}, \\ 300^\circ, & \mathrm{dir}(\mathrm{STAR}) = \mathrm{S}. \end{cases}$$

Let $\mathcal{Z}_{\mathrm{STAR}}$ be the green-zone polygon for each STAR. The counter z_t records the number of consecutive policy steps outside this polygon. STAR-3 procedures are direct-vector approaches and are exempt from zone-based terms.

We define the specifics of each reward component under Table 4. The run we shipped contains a two-phase schedule with

$$R_{\mathrm{base}} = R_{\mathrm{term}} + R_{\mathrm{hdg}} + R_{\mathrm{final}},$$

the shipped single-aircraft PPO run uses

$$\begin{aligned} &\text{Phase 1:} \quad R_t = R_{\mathrm{base}} + R_{\mathrm{every}}(0.001) + R_{\mathrm{loop}}(0.10) + T_{\mathrm{OOZ}}(5), \\ &\text{Phase 2:} \quad R_t = R_{\mathrm{base}} + R_{\mathrm{every}}(0.002) + R_{\mathrm{loop}}(0.10) + T_{\mathrm{OOZ}}(10) + R_{\mathrm{scale}}(0.5). \end{aligned}$$

Table 4: Reward components for single-aircraft PPO

Component	Definition
R_{term}	$+10$ on success (intercepting LOC under GS), -10 on failure (o/w)
R_{hdg}	Applied at localizer capture using the realized aircraft heading θ_t to encourage good interception headings: $$R_{\mathrm{hdg}} = \begin{cases} \max(-2,\ 1 - (\theta_t - 240^\circ)), & \mathrm{dir} = \mathrm{N},\ \theta_t \geq 240^\circ, \\ \max(-2,\ 1 - (300^\circ - \theta_t)), & \mathrm{dir} = \mathrm{S},\ \theta_t \leq 300^\circ, \\ -2, & \text{o/w.} \end{cases}$$
R_{final}	Applied once when the aircraft is in the green zone and $\lvert c_{\mathrm{nm}} \rvert \in [1, 2]$ nm. It uses the commanded heading to encourage good turn-final behaviors: $$R_{\mathrm{final}} = \begin{cases} 0.5 - 0.4\,\dfrac{\theta_t^{\mathrm{cmd}} - 240^\circ}{5^\circ}, & \mathrm{dir} = \mathrm{N},\ \theta_t^{\mathrm{cmd}} \in [240^\circ, 245^\circ], \\ 0.5 - 0.4\,\dfrac{300^\circ - \theta_t^{\mathrm{cmd}}}{5^\circ}, & \mathrm{dir} = \mathrm{S},\ \theta_t^{\mathrm{cmd}} \in [295^\circ, 300^\circ], \\ 0, & \text{o/w.} \end{cases}$$ It is identically zero for STAR-3.
$R_{\mathrm{every}}(p)$	$-p$ per simulator step.
$R_{\mathrm{loop}}(q)$	After a rollout, a loop detector flags orbiting behavior from the $(a_{\mathrm{nm}}, c_{\mathrm{nm}})$ trace. A penalty $-q$ is applied to detected loop steps.
$R_{\mathrm{zone}}(\eta, \kappa)$	A counterpart to hard out-of-zone termination: per-step soft out-of-zone penalty. With d_t the distance (nm) of the aircraft outside the green zone: $$R_{\mathrm{zone}} = \begin{cases} -\min(\eta\, d_t,\ \kappa), & d_t > 0, \\ 0, & \text{o/w.} \end{cases}$$
$T_{\mathrm{OOZ}}(c)$	If $z_t \geq c$, terminate the episode as an out-of-zone failure.
$R_{\mathrm{scale}}(\alpha)$	Scale the success terminal by $$1 + \alpha(1 - \mathrm{SR}_{\mathrm{STAR}}),$$ where $\mathrm{SR}_{\mathrm{STAR}}$ is the previous evaluation block's success rate for that STAR to encourage higher SR for failing approaches.

B.2 Ablation configurations and results

Ablation experiments rerun the two-phase schedule of Section B.1 with a different reward component changed. All runs share the actor, critic models, seeds, and the same PPO hyperparameters specified in 5.1.2. We refer to the shipped run as `dense_rewards`; Table 5 lists each run's reward configuration. Note the two exceptions include

`partial_state` run that keeps the full reward exactly but swaps the 7-D actor seed for a 3-D position-only one, isolating state observability rather than reward, and `continuous_01` is using soft out-of-zone penalties and changing the reward weights every 10 iterations based on current model performance. `terminal_only` is a baseline run without intermediate rewards or out-of-zone terminations.

Table 5: Reward configuration per run; / shows the Phase-1/Phase-2 differences; the boldfaced entries are ablation changes relative to `dense_rewards`

Run	R_{zone} (η/κ)	R_{every}	R_{loop}	T_{OOZ}	R_{scale}	R_{hdg}	R_{final}
`dense_rewards`	off	.001/.002	.10	5/10	.5	on	on
`no_loop`	off	.001/.002	**0**	5/10	.5	on	on
`no_everywhere`	off	**0**	.10	5/10	.5	on	on
`no_turn_final`	off	.001/.002	.10	5/10	.5	on	**off**
`no_heading_intercept`	off	.001/.002	.10	5/10	.5	**off**	on
`no_sr_scale`	off	.001/.002	.10	5/10	**0**	on	on
`soft_zone`	**.0005/.005**	.001/.002	.10	**off**	.5	on	on
`no_ooz_term`	off	.001/.002	.10	**off**	.5	on	on
`terminal_only`	off	**0**	**0**	**off**	**0**	**off**	**off**
`continuous_01`	.0005/.005–.02	0–.001	.05–.10	off	off	on	on

Fig. 9 tracks the four metrics we report across training and Table 6 reports every run at that best checkpoint.

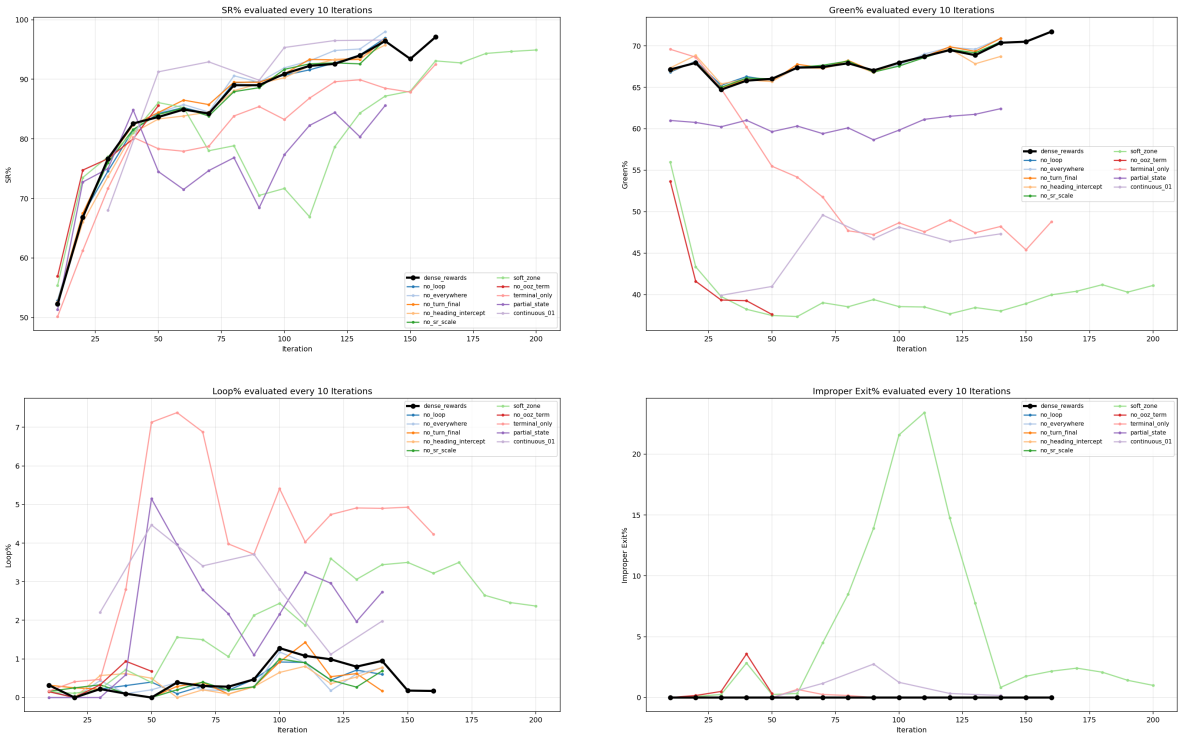


Figure 9: Evaluation metrics every 10 iterations for all runs

B.3 Evaluation

From the ablation results, we show that out-of-zone termination is the component that matters the most. Replacing it with soft per-step penalty holds success rate but does not well enforce regulating trajectories patterns. An example is shown in 8 where the downwind is not clearly maintained and shortcuts are taken for longer trajectories. Removing green-zone regularization would further destabilize training for `no_ooz_term` model, and the STAR NORTH1 specifically collapsed during training.

Table 6: Each run at its highest-SR checkpoint. SR and the failure columns (ABV-GS = LOC-above-GS, BHD = LOC-behind-threshold, T/O = timeout, IMP = improper exit) in percentages of all 1200 cases; loop% is the fraction of landed trajectories with loops detected; green% is the mean in-zone coverage over landed trajectories

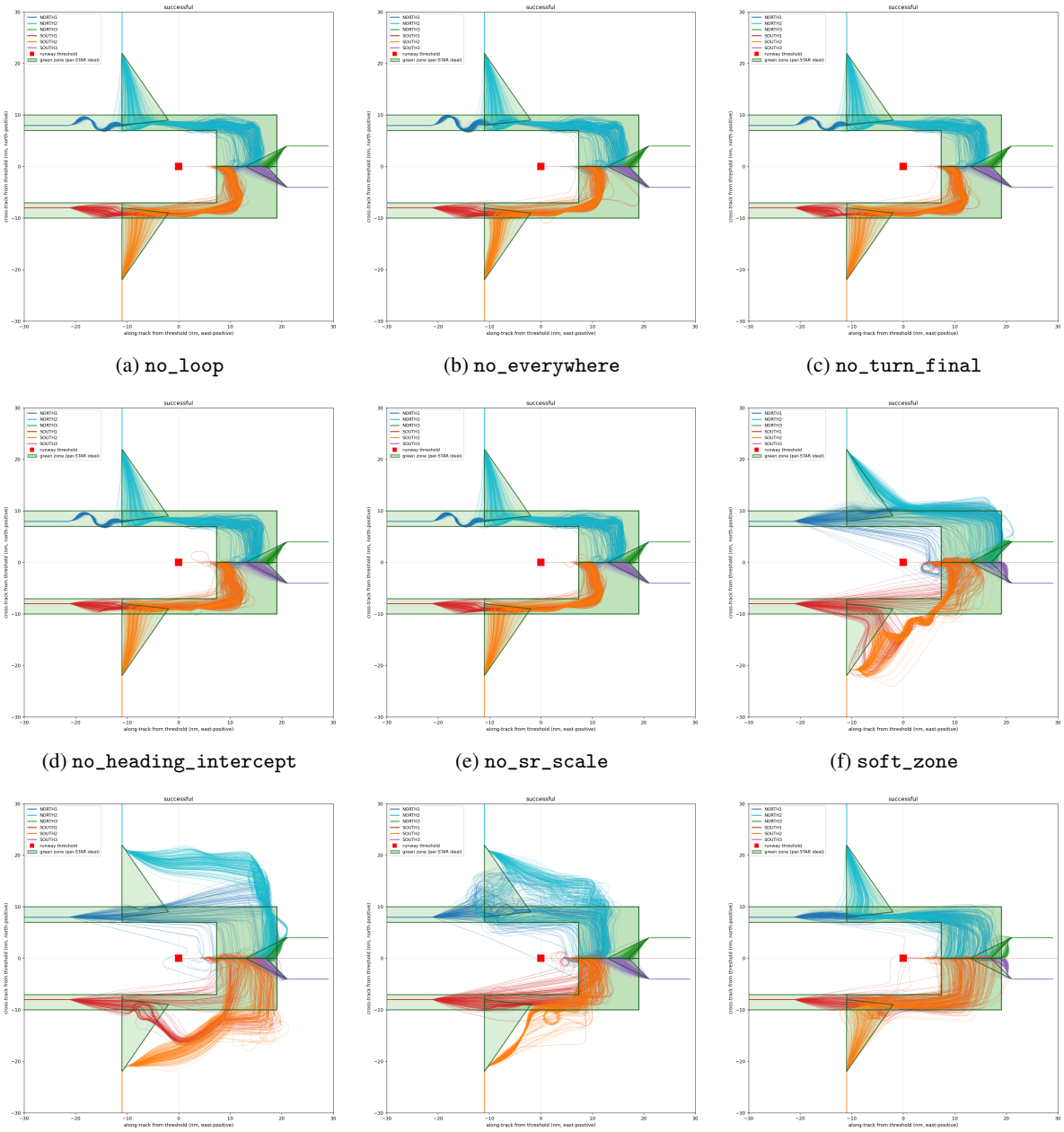
Run	best it.	SR	green	loop	ABV-GS	BHD	T/O	IMP
dense_rewards	160	97.1	71.7	0.2	1.0	0.2	1.8	0.0
no_loop	140	96.9	70.5	0.6	1.8	0.0	1.2	0.0
no_everywhere	140	98.0	70.9	0.8	1.0	0.0	1.0	0.0
no_turn_final	140	96.8	70.9	0.2	1.9	0.0	1.3	0.0
no_heading_intercept	140	95.8	68.7	0.8	1.8	0.0	2.4	0.0
no_sr_scale	140	96.6	70.5	0.7	2.3	0.0	1.1	0.0
soft_zone	200	94.9	41.1	2.4	3.0	0.0	1.1	1.0
no_ooz_term	50	85.6	37.6	0.7	4.3	1.2	8.6	0.3
terminal_only	160	92.5	48.8	4.2	1.7	0.0	5.8	0.0
partial_state	140	85.6	62.4	2.7	4.1	0.0	10.3	0.0
continuous_01	140	96.6	47.3	2.0	1.2	0.0	2.1	0.2

Table 7: Per-STAR SR percentage for ablation models

Model	NORTH1	NORTH2	NORTH3	SOUTH1	SOUTH2	SOUTH3	Overall
dense_rewards	97	98	100	94	95	98.5	97.1
no_loop	97	96.5	100	93	96	99	96.9
no_everywhere	98	98.5	98.5	96.5	98	98.5	98
no_turn_final	96	96.5	100	94	94.5	99.5	96.8
no_heading_intercept	94.5	95.5	99.5	92.5	94	98.5	95.8
no_sr_scale	97.5	97.5	99	93	94	98.5	96.6
soft_zone	93	96.5	99.5	86.5	95.5	98.5	94.9
no_ooz_term	59.5	95.5	97	73.5	89.5	98.5	85.6
terminal_only	76.5	92	98.5	96.5	93	98.5	92.5
continuous_01	95.5	96.5	100	94	94	99.5	96.6

Furthermore, the ablation results on 3D partial state show the necessity of implementing 7D mixture model to capture the complexity of states, even though the performance regresses on behavioral cloning, as shown by the plateau `partial_state` model training.

The other reward components mainly help improving the quality of trajectories instead of helping with success rates. For example, loop penalty would to some extent help model avoid looping, but the effects are minimal when combined with hard out-of-zone termination as most loops would be large enough to exit the green zone completely. It can be shown in runs without using hard OOZ penalty, where loops form without such penalties. Turn final and heading intercept rewards are useful qualitatively in localizer interception, as the shipped model demonstrates high quality trajectories on turning final at right times and clear attempts to amend the target when overshooting the centerline. The everywhere penalty, when used with hard OOZ termination, also shows qualitatively cleaner trajectories preventing traffic from extending downwind to over 15 nm out, which is rarely shown in human demonstrations. Scaling at phase 2 iterations also lightly help with balancing the success rate between different approach STARs, as shown in 7. Note that some differences are subtle; the trajectories are visualized in Fig. 10.



(a) `no_loop` (b) `no_everywhere` (c) `no_turn_final`

(d) `no_heading_intercept` (e) `no_sr_scale` (f) `soft_zone`

(g) `no_ooz_term` (h) `terminal_only` (i) `partial_state`

Figure 10: Successful Trajectory Visualizations for Ablation Runs

C Long Horizon Planning Algorithm

Algorithm 1 Long-horizon planning for sequencing

parameters: horizon n, re-roll budget K, batch size b
$P \leftarrow \emptyset$
for each simulation tick **do**
    for each aircraft entering the airspace **do**
        add it to P with an empty plan
    end for
    if any plan in P is depleted **then**
        extend every plan in P to n steps by rolling the GMM forward
        for $iter = 1$ **to** K **do**
            $C \leftarrow$ planes whose plans conflict over the next n steps
            if $C = \emptyset$ **then break**
            re-roll b fresh GMM candidates for each $a \in C$ ▷ excluding planes on short finals
            $P \leftarrow$ a conflict-free assignment from the candidate pool, if one exists
        end for
        if $C \neq \emptyset$: for each $a \in C$, keep the candidate with the largest min. separation over n
    end if
    advance every plan in P one step
end for