

Learning Air Traffic Conflict Resolution via Adversarial Inverse Reinforcement Learning

Ferran Dalmau* and Ignasi Cos

Facultat de Matemàtiques i Informàtica, Universitat de Barcelona, Barcelona, Spain

*Corresponding author: dalmaucodinaf@gmail.com

Abstract

Air traffic controllers resolve thousands of potential mid-air conflicts daily, yet automation research has struggled to replicate these decisions. Reinforcement learning approaches have achieved limited success because their action spaces (heading and speed changes) do not reflect controller practice, and their hand-crafted reward functions do not capture the complexity of expert decision-making. We present *Tower Parrot*, to our knowledge the first application of Inverse Reinforcement Learning (IRL) to tactical air traffic conflict resolution, learning the reward function directly from data rather than engineering it manually.

Our approach combines Adversarial IRL (AIRL) with a pointer-network policy that selects among flight-plan waypoints—mirroring the “direct to” instructions controllers actually issue. Building on the publicly available Bordeaux ACC deconfliction dataset of Gaume *et al.* [1], we designed a two-stage instruction-recovery algorithm that extracts approximately 8,800 labelled demonstrations from approximately 31,000 encounters, exploiting structure not identified in prior work.

The central finding is that separation-compliant behaviour emerges without any separation term in the objective. The ICAO-mandated 5 NM separation minimum is encoded nowhere in the reward, the shaping prior or the loss, yet the AIRL policy maintains separation above this threshold in 100% of 200 held-out encounters, whereas behavioural cloning violates it in 27% of cases and a rule-based heuristic reading the same features in 5%. Since the split informed design decisions and the rollouts are simulated, we treat the figure as an optimistic bound and rest the claim on the comparison: a reward inferred from the demonstrations yields better separation behaviour than cloning those same demonstrations. Beyond regulatory compliance, the learned policy intervenes earlier, achieves larger separation margins, and produces smoother trajectories—reflecting implicit controller objectives recovered from data rather than imposed through manual tuning. These results indicate that this algorithm can provide operationally safe, controller-aligned resolution strategies can be learned directly from historical data.

Keywords: conflict resolution; inverse reinforcement learning; imitation learning; air traffic control; safety-critical systems

Abbreviations: AIRL: Adversarial Inverse Reinforcement Learning, ADS-B: Automatic Dependent Surveillance–Broadcast, ATC: Air Traffic Control, BC: Behavioural Cloning, CPA: Closest Point of Approach, IRL: Inverse Reinforcement Learning, NM: Nautical Mile

1. Introduction

Air traffic control (ATC) in en-route sectors is a continuous exercise in conflict avoidance. A controller monitors dozens of aircraft simultaneously and, when two are predicted to lose lateral separation below the 5 nautical mile (NM) standard of the International Civil Aviation Organization (ICAO) [2], issues a tactical instruction— often “direct to” a downstream waypoint, or often a heading vector—that resolves the conflict while keeping each aircraft near its planned route. The 5 NM minimum is not a soft guideline: it is a hard regulatory boundary codified in ICAO Doc 4444, whose violation triggers mandatory reporting and potential enforcement action.

Modern surveillance, in the form of Automatic Dependent Surveillance–Broadcast (ADS-B) [3], captures the *outcome* of every controller decision—the aircraft’s actual ground track—but the instruction itself travels by voice and is never logged. Historical surveillance therefore contains thousands of expert decisions and no labels saying what those decisions were. **This is the critical challenge of the field:** the data record what the expert *did*, never what the expert *chose*, and any learning system must reconstruct the choice before learning from it.

Reinforcement learning (RL) formulations sidestep the missing labels by not needing demonstrations at all: they hand-design a reward that penalises separation loss, typically a large negative term when predicted separation drops below 5 NM, and optimise against a simulator [4, 5]. That buys safety by construction at two costs: the action space is usually a continuous heading or speed change, which is not what controllers issue; and the policy reveals nothing about what controllers weigh when choosing among resolutions that would all be legal. Work that does learn from controllers has mostly cloned their actions under supervision [6, 7] rather than recovered the objective behind them. Inverse reinforcement learning (IRL) has reached air traffic management before [8, 9], but not, to the best of our knowledge, tactical conflict resolution over a recovered, controller-native action space.

This paper presents *Tower Parrot*, a pipeline that learns a conflict-resolution policy from the unlabelled Bordeaux ACC deconfliction catalogue of Gaume *et al.* [1] using Adversarial Inverse Reinforcement Learning (AIRL) [10]. Four technical contributions structure the work:

1. **Instruction recovery from geometry.** A two-stage heuristic matches the post-deviation track to the bearings of upcoming flight-plan waypoints and verifies proximity, extracting approximately 8,800 labelled demonstrations (28% of encounters) from approximately 31,000 unlabelled encounters.
2. **Contamination-aware demonstration building.** A manoeuvre-detection module identifies unlabelled instructions to other aircraft in the sector and down-weights WAIT steps whose “do nothing” label is corrupted by them.
3. **A controller-native action space.** A discrete pointer over the deviated aircraft’s upcoming flight-plan waypoints replaces a continuous (Δ heading, duration) regression, which failed because the duration label encodes turn-execution time rather than controller intent.
4. **Stable AIRL training on a 93/7 action imbalance.** A behavioural cloning (BC) warm-start, an entropy bonus and stratified discriminator batches resolve the mode collapse that otherwise pins the policy at always-WAIT, and a soft early-intervention prior encodes “act sooner” without referencing any distance.

1.1 Central Finding

The central question of the paper is what such a policy does about separation when nothing in its objective mentions separation. The 5 NM standard is encoded nowhere in the reward, the shaping prior or the loss—by design. If the learned policy maintains separation, that success must come either from imitation of the controller or from information the observation already carries; separating those two sources is what Section 7 is for. The answer: on the 200 held-out encounters of the development split, the AIRL policy produced **no trajectory below 5 NM**, against 27% for BC and 5% for a rule-based heuristic reading the same features—an optimistic bound, since the split informed design, the rollouts run under the same straight-line kinematics that generate the policy’s own features, and the AIRL row comes from a single seed. What the comparison establishes is an ordering between methods evaluated under identical conditions: the inferred reward produces better separation behaviour than cloning the demonstrations it was inferred from. We read the picture as follows: imitation does not supply the boundary, which the observation already encodes in units of the separation standard; what it appears to supply is the *valuation* placed on that information—how much

margin is enough, and how early to spend route efficiency to get it.

The remainder of the paper is organised as follows. Section 2 surveys related work. Section 3 describes the dataset and the instruction-recovery pipeline. Section 4 presents the problem formulation, the pointer-network policy and the AIRL training procedure. Section 5 defines the evaluation protocol. Section 6 reports the experimental results. Section 7 discusses what the results do and do not show, and Section 8 concludes.

2. Related Work

Inverse reinforcement learning. Ng and Russell [11] formalised IRL as the recovery of a reward function under which observed expert behaviour is optimal. Early algorithms reduced the problem to feature-expectation matching [12], generalised by the maximum-entropy principle to resolve the inherent reward ambiguity [13]. The adversarial reformulation—first as Generative Adversarial Imitation Learning (GAIL) [14], then with reward recovery as AIRL [10]—replaces the inner-loop RL problem with a discriminator that separates expert from policy transitions. AIRL constructs the discriminator as $D_\psi = \sigma(f_\psi - \log \pi)$ with $f_\psi(s, a, s') = g_\psi(s, a) + \gamma h_\psi(s') - h_\psi(s)$, which under mild assumptions disentangles the reward g_ψ from potential-based shaping h_ψ ; the disentangled reward can be transferred to new environments—precisely the property we exploit, since our aim is to recover not just expert actions but the valuation that explains *why* experts act safely.

Behavioural cloning. BC—supervised learning over (state, expert action) pairs—dates to ALVINN [15] and is well known to suffer from covariate shift, addressed by DAGger and related interactive corrections [16]. Importantly, BC learns to predict actions but not to understand why certain actions are preferred—a distinction that becomes critical for safety, as Section 6 shows. BC plays two roles here: a baseline establishing whether the recovered demonstrations carry a learnable signal at all, and a warm-start for AIRL that turns adversarial training into fine-tuning of an already-competent policy.

Pointer networks. The action space in Tower Parrot is variable in size: each encounter offers between one and six “direct to” waypoint candidates. Pointer networks [17] use a content-based attention mechanism [18] to produce a probability distribution over a variable-length set of input elements, with a sinusoidal positional encoding [19] breaking the symmetry between waypoints at different route depths.

Learning-based ATC. Prior work applies deep RL to ATC sequencing and separation with hand-designed reward functions that explicitly encode safety constraints [4, 5]. Such systems achieve safety by construction but reveal nothing about what human controllers actually optimise. Data-driven imitation of controllers has been pursued with behaviour cloning [6] and with supervised models of controller policy trained on real-world conflict data [7]. IRL has been applied to ground delay program decisions [8], and Tolstaya *et al.* [9] recovered arrival routings and pairwise spacing behaviour from recorded traffic as cost functions for search-based planning. Tower Parrot differs fundamentally: the reward is *learned* from controller demonstrations rather than designed, safety constraints are *recovered* rather than imposed, and the action space is the controller’s own direct-to instruction over the filed flight plan.

3. Data and Instruction Recovery

Data come from the catalogue of historical deconfliction actions assembled by Gaume *et al.* [1] from ADS-B observations over the Bordeaux Area Control Centre (ACC), AIRAC cycle 2201 (January 2022): approximately 31,000 encounters (approximately 24 GB). Each encounter describes one situation in which two aircraft were predicted to lose lateral separation, and contains the deviated

aircraft’s planned route, the ADS-B tracks of the sector aircraft, closest-point-of-approach (CPA) predictions, and the start_deviation/stop_deviation timestamps bracketing the window during which the deviated aircraft was off its plan. **The critical challenge:** what the files *do not* contain is the controller’s instruction—no voice-command transcript, no target-waypoint label, no instruction-type tag. The dataset records when an aircraft began to deviate and where it went; the instruction that caused the deviation must be inferred from geometry. This single fact dictates the architecture of the preprocessing pipeline.

3.1 Two-stage instruction identification

During the resolution in en-route sectors “direct to waypoint k ”, the controller selects one of the aircraft’s upcoming flight-plan waypoints and instructs the pilot to fly straight to it, bypassing intermediate fixes. This is the only instruction class whose *target* can be recovered from post-deviation geometry alone, by comparing the aircraft’s track to the bearings from its position to each upcoming waypoint. The recovery algorithm proceeds in two stages (Figure 1):

Stage 1 (heading filter). The mean ADS-B track angle over the 120 s following deviation onset is compared to the bearing from the aircraft’s position at t_{dev} to each upcoming waypoint WP_k ; candidates whose angular difference exceeds 5° are discarded.

Stage 2 (proximity check). The heading filter alone breaks down on the colinear routes common in en-route traffic, where consecutive waypoints lie at bearings differing by less than 5° . For each surviving candidate, taken furthest to nearest, we check whether the aircraft’s ADS-B trail actually passes close to WP_k . The selected waypoint is the furthest one the aircraft demonstrably flew near. If no candidate passes, the instruction is classified as a free-heading vector and the encounter is excluded from training.

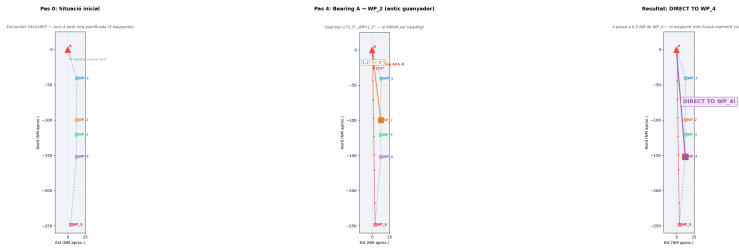


Figure 1. Two-stage instruction identification on one encounter. *Left:* at the moment of deviation, the aircraft’s predicted route includes several upcoming waypoints at slightly different bearings. *Middle:* the post-deviation ADS-B track is compared to each candidate bearing; multiple candidates pass the 5° heading filter on a colinear route. *Right:* the proximity check disambiguates by selecting the furthest waypoint the aircraft actually flew near, yielding the target k^* .

On the full catalogue this procedure extracts approximately 8,800 usable “direct to waypoint” demonstrations from approximately 31,000 encounters (28%), yielding approximately 152,000 step-level tuples of which 93.1% are WAIT. The excluded majority is not a random sample: it contains the encounters resolved by heading vector or with no skip-a-waypoint option, which arise disproportionately when time-to-CPA is short or the geometry is an overtake. The consequences of this selection are analysed in Section 7.

Three thresholds govern the recovery: the 5° heading tolerance of Stage 1, the proximity radius of Stage 2, and the $5^\circ/\text{min}$ heading-change rate that marks a segment as manoeuvring. The first two trade recall against label noise, and we set them conservatively, which is part of why the yield is 28% rather than higher. We did not characterise how sensitive the downstream policy is to these choices, nor to ADS-B position noise, which propagates through the bearing estimates into both the

recovered labels and the CPA features the policy reads; we flag both as open rather than settled.

3.2 Why the continuous action space failed

The discrete pointer formulation is the result of a concrete failure. An earlier version modelled the controller's decision as a continuous tuple—whether to act, by how many degrees, and for how long—with the simulator executing “fly pre_heading + Δ heading for duration seconds, then return to pre_heading”. This did not converge, for two reasons invisible from training curves alone. First, the duration label was meaningless for the dominant case: in approximately 73% of encounters the aircraft never returns to its original heading within the trajectory window—the recorded stop_deviation timestamp captures merely the turn-execution time (typically 30–100 s), not anything the controller chose. Second, because the simulator returned the aircraft to its original heading, rollouts re-entered the original conflict in approximately 73% of cases, despite this re-conflict never appearing in the expert data; the discriminator was trained on *fake negatives*—transitions the environment generated but the world never produced. The fix was not better RL but an action space that matches the controller's actual decision: “fly direct to waypoint k ”.

3.3 WAIT-label contamination

A naïve demonstration builder records a WAIT action for every 30-second step before the labelled instruction, but a fraction of these labels are fictional: the catalogue labels only one instruction per file, while each encounter contains dozens of sector aircraft, many of which are themselves manoeuvring—receiving real controller instructions visible in their ADS-B tracks—around the labelled event. An audit identified approximately 1,950 such unlabelled instructions and flagged 48.2% of demonstration steps as contaminated to some degree. Two consequences follow: WAIT steps coinciding with unlabelled instructions teach the model an incorrect prior about controller workload, and the Markov assumption that aircraft B flies straight breaks exactly when B is itself manoeuvring, so the features computed from the simulated trajectory diverge from reality.

The mitigation, implemented as a manoeuvre-detection module, has three components: a detector marks ADS-B segments whose heading-change rate exceeds $5^\circ/\text{min}$ as manoeuvring; each WAIT step receives a contamination score (1.0 when aircraft B is manoeuvring, proportional to the fraction of other manoeuvring aircraft otherwise, 0.0 when clean); and the scores feed two complementary strategies—hard filtering removes WAIT steps where B itself is manoeuvring, and soft weighting down-weights partially contaminated steps to 0.1 of a clean step in the loss. DIRECT steps are never filtered or down-weighted.

4. Method

4.1 Problem formulation

The conflict-resolution problem is a semi-Markov decision process: the controller monitors an evolving conflict over many time steps and acts at most once per encounter. Time advances in discrete steps of $\Delta t = 30$ s. At each step the action is either WAIT—both aircraft propagate under straight-line kinematics for Δt , executing any route turn at the next waypoint—or DIRECT_k for $k \geq 1$, which sets the deviated aircraft's heading to the bearing toward waypoint WP_k and updates its route to begin there. After one DIRECT is issued, only WAIT remains valid. Episodes terminate when the straight-line CPA exceeds twice the separation minimum and the predicted time-to-CPA exceeds a 600 s horizon, or when either aircraft reaches the end of its observed route; there is no artificial episode length and no step counter in the state.

4.2 State representation

The state $s_t = (\mathbf{c}_t, R_t)$ comprises a fixed-size *context* block $\mathbf{c}_t \in \mathbb{R}^{14}$ encoding the pairwise conflict geometry, and a variable-size *route* block $R_t \in \mathbb{R}^{(N-1) \times 6}$ with one slot per DIRECT_k candidate. The state is computed in the deviated aircraft’s heading-aligned frame—its current position is the translation origin, its current heading the angular reference—so the representation is translation- and rotation-invariant by construction, with bearings encoded as (cos, sin) pairs and distances as dimensionless ratios.

Context block. Bearing and distance from A to B; direction and magnitude of B’s velocity relative to A; closing speed and straight-line CPA distance and time; distance to A’s next route waypoint; bearing of the next turn; relative altitude; and vertical closure rate.

Route block (one slot per candidate waypoint). Bearing and distance to WP_k ; the *simulated* CPA that would result if A flew direct to WP_k ; a temporal ratio; and a route-disruption proxy. The simulated-CPA field is the single most informative feature in the representation, and Section 7 returns to why that matters.

4.3 Pointer-network policy

The policy $\pi_\theta(a|s)$ is a pointer network (Figure 2) with approximately 14,000 parameters at encoder width 64. A context encoder multi-layer perceptron embeds the context block into $\mathbf{c} \in \mathbb{R}^d$; a shared slot encoder embeds each route slot into $\mathbf{h}_k \in \mathbb{R}^d$; a sinusoidal positional encoding breaks the symmetry between waypoints at different route depths. Scaled dot-product attention scores each slot against the context,

$$e_k = \frac{(W_Q \mathbf{c})^\top W_K (\mathbf{h}_k + \text{PE}(k))}{\sqrt{d}}, \quad k = 1, \dots, N-1, \quad (1)$$

and a separate sigmoid head produces the WAIT gate $p_{\text{wait}} = \sigma(\mathbf{w}^\top \mathbf{c})$. The final distribution is

$$\Pr(\text{WAIT}) = p_{\text{wait}}, \quad \Pr(\text{DIRECT}_k) = (1 - p_{\text{wait}}) \cdot \text{softmax}(e)_k. \quad (2)$$

The heading is not an explicit action variable: each DIRECT_k deterministically implies a heading, so the policy never emits an instruction the controller could not issue.

4.4 Behavioural cloning baseline

BC trains the same pointer network under a weighted cross-entropy loss

$$L_{\text{BC}}(\theta) = -\mathbb{E}_{(s,a) \sim \mathcal{D}} \left[w_{\text{sample}}(s) \cdot w_{\text{class}}(a) \cdot \log \pi_\theta(a|s) \right], \quad (3)$$

where w_{sample} is the contamination weight of Section 3 and w_{class} compensates for the 93.1%/6.9% WAIT/DIRECT imbalance through stratified 50/50 batch sampling.

4.5 AIRL training

The AIRL discriminator decomposes its energy as $f_\psi(s, a, s') = g_\psi(s, a) + \gamma h_\psi(s') - h_\psi(s)$, where g_ψ is the learned reward and h_ψ is potential-based shaping [10]. The reward head takes as input the concatenation of the context embedding and the embedding of the chosen action’s route slot, so the recovered reward depends on the geometric quality of the chosen action. Training alternates

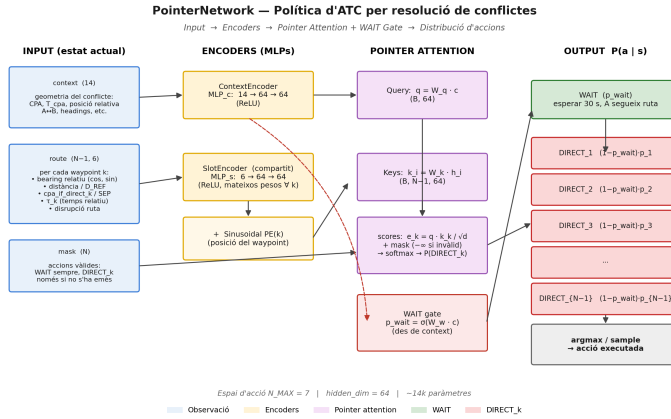


Figure 2. Pointer-network policy architecture. A context encoder embeds the 14-dimensional geometry block; a shared slot encoder embeds each candidate waypoint; scaled dot-product attention scores each slot against the context; a sigmoid head gates WAIT.

discriminator updates with policy updates using REINFORCE [20] on $\hat{r}_t = \log D_\psi - \log(1 - D_\psi)$; Proximal Policy Optimization [21] is the lower-variance upgrade, left as future work.

Mode-collapse mitigations. Trained naively, the policy collapses to always-WAIT: it never samples DIRECT, so the slot parameters receive no gradient, even though the discriminator has already learned the right preference—averaged over the expert set, it assigns a reward of +0.66 to expert DIRECT transitions against +0.28 to expert WAIT transitions. The failure is therefore one of exploration rather than of reward inference. Three mitigations resolve it: (1) *BC warm-start* initialises the policy from a trained BC checkpoint that samples DIRECT; (2) an *entropy bonus* $-\beta H(\pi)$ with $\beta = 0.01$ prevents WAIT-gate saturation; (3) *stratified expert batches* balance the discriminator's expert mini-batches 50/50 between WAIT and DIRECT. Section 6 reports what they contribute to training.

Early-intervention prior. A shaping prior gives every DIRECT a bonus $b_t = \text{coef} \cdot \text{decay}^t$ with $\text{coef} = 0.5$, $\text{decay} = 0.9$, encoding “DIRECT sooner is better” without referencing any distance. It is a prior over *timing*, not safety, and the ablation in Section 6 measures what it contributes.

Critical design choice. The ICAO 5 NM separation minimum is encoded *nowhere* in the system—not in the reward, not in the prior, not in the loss. No threshold, penalty or distance rule appears in what the policy is optimised for, so any safety compliance must emerge from pure imitation of the controller—which makes the central question of the paper an empirical one rather than an assumption.

5. Evaluation Protocol

Policies compared. Four policies are evaluated throughout: (i) the *human* row, which replays the controller's recorded instruction sequence through the simulator; (ii) the *AIRL* policy; (iii) the *BC* baseline, warm-started identically; and (iv) a *rule-based* heuristic that issues DIRECT to the waypoint maximising the simulated CPA once the time-to-CPA falls below 600 s, reading the same observation the networks see.

Simulator. Closed-loop rollouts propagate aircraft A according to the policy's DIRECT choice and aircraft B along its recorded ADS-B track under straight-line kinematics. The human row therefore

uses the identical kinematics as the policy rows, so all four rows are scored on the same footing.

Splits. Results are reported on the 200 encounters held out by the 80/20 encounter-level split of the demonstration dataset (seed 42), so no step from a held-out encounter appears in training. The split also guided design decisions and model selection and was not blind to the results reported below; its figures are therefore an optimistic bound, and we read them as a comparison between methods rather than as an absolute compliance rate.

Metric families. Three families of metrics are reported, each answering a different question: *per-step imitation*—replaying the expert trajectory, querying each policy at every expert state: overall accuracy, WAIT accuracy, DIRECT exact-waypoint match and recall, false-alarm rate, and mean negative log-likelihood; *decision-level*—from the closed-loop rollout, one decision per episode: issued rate, waypoint-match rate overall and given issued, and the timing error Δt in 30 s steps relative to the controller’s DIRECT (negative = earlier); and *outcome / safety*—the minimum A–B separation achieved in each rollout against the 5 NM standard: compliance proportions above 5, 7 and 10 NM with Wilson 95% confidence intervals [22], and the distribution of minimum separation (mean, median, 10th percentile). The outcome family is the basis of every comparative claim; Section 6 shows that each family would be misleading alone.

Hyperparameters. Table 1 summarises the training configuration. BC and AIRL share architecture and data; AIRL adds the discriminator, the entropy bonus, and the prior. All experiments use encoder width 64 ($\approx 14,000$ parameters). BC figures are averaged over three seeds; the AIRL checkpoint comes from a single seed (42), and seed-to-seed variation in the adversarial run remains uncharacterised.

Table 1. Training configuration. Both methods share the pointer-network architecture at encoder width 64.

Parameter	BC	AIRL
Policy learning rate	3×10^{-4}	1×10^{-4}
Discriminator learning rate	—	3×10^{-4}
Batch size	64	64
Epochs / iterations	50	500–1000
Encoder width	64	64
Discount γ	—	0.99
Entropy coefficient β	—	0.01
Prior coefficient / decay	—	0.5 / 0.9

6. Results

6.1 What behavioural cloning recovers

BC answers the prior question of whether the recovered demonstrations carry a learnable signal at all, and since it is also the AIRL warm-start, its weaknesses propagate. Its overall WAIT/DIRECT accuracy of $89.9 \pm 0.9\%$ (three seeds) sits *below* the 93.1% obtained by always predicting WAIT—a consequence of the balanced sampling needed to make it predict DIRECT at all; overall accuracy is simply the wrong metric here. On the question that matters—when the controller issued a DIRECT, which waypoint did BC select—it matched $17.4 \pm 7.3\%$ of the time, against $33.7 \pm 17.8\%$ for the rule-based heuristic (which scored $69.3 \pm 2.6\%$ overall accuracy on the same task). With one to six candidates per encounter, uniform guessing scores in the range 17–100%, and the intervals overlap, so we draw no ranking on waypoint agreement. **What the comparison does reveal is BC’s fundamental weakness: it learns *when* to act, but not *which action is best*.** Cloning

recovers the demonstrations poorly even on the metric it was trained to optimise—which is exactly why recovering the objective behind the demonstrations matters.

6.2 AIRL training and mode-collapse resolution

Without the mitigations of Section 4, adversarial training pins the policy at always-WAIT even though the discriminator has already learned the right preference: averaged over the expert set, it assigns a reward of +0.66 to expert DIRECT transitions against +0.28 to expert WAIT transitions. The policy nonetheless never samples a DIRECT, so the slot parameters receive no gradient. The failure is therefore one of exploration rather than of reward inference. With the BC warm-start, the entropy bonus and stratified discriminator batches in place, the discriminator loss falls from 1.08 to 0.95 within 10 iterations, against a movement of only 1.35 → 1.21 over 200 iterations without them—the mitigated run improves further in a twentieth of the iterations. The three mitigations together convert a training failure into a routine optimisation problem.

6.3 Central finding: compliance without a separation term

This is the key result of the paper. The ICAO 5 NM minimum is a hard regulatory boundary rather than a soft target or a quantity to be traded off: a breach triggers mandatory reporting and potential enforcement action. Nothing in the *objective* refers to it: the inferred reward g_ψ contains no distance threshold, no separation penalty, and no clause of the form “if separation falls below 5 NM, penalise”; the early-intervention prior rewards acting sooner and references *time* only, never distance; and the BC and AIRL losses are computed over actions and discriminator logits, never over achieved separation. The qualifier matters: the constant is absent from what the policy is *optimised for*, not from the system it is embedded in.

Table 2 reports the safety outcomes on the 200 held-out development encounters. The AIRL policy produced no rollout below 5 NM, matching the replayed human row; BC breached the minimum in 54 of 200 encounters (27%) and the rule-based heuristic—which reads the same predicted-CPA features and maximises them greedily—in 10 (5%). Both differences are significant under a two-sided Fisher’s exact test ($p = 1.8 \times 10^{-18}$ against BC, $p = 0.0017$ against the rule-based heuristic). The second comparison carries the design argument: access to the CPA features is not by itself enough to stay clear, because the heuristic fires on a fixed trigger and commits to whichever waypoint looks best at that moment, while the learned reward, never told what 5 NM means, did better.

On the 200 held-out development encounters, the AIRL policy never produces a trajectory in which the minimum separation falls below 5 NM.

Table 2. Development split: proportion of encounters ($n = 200$ held-out) in which minimum separation stayed above each threshold, with Wilson 95% confidence intervals [22] in brackets. All four rows are rollouts under the same straight-line kinematics; the human row replays the recorded instruction.

Method	>5 NM	>7 NM	>10 NM
Human (replayed)	100.0% [98.1–100]	89.0% [83.9–92.6]	62.0% [55.1–68.4]
AIRL	100.0% [98.1–100]	85.0% [79.4–89.3]	58.0% [51.1–64.6]
BC	73.0% [66.5–78.7]	61.0% [54.1–67.5]	42.0% [35.4–48.9]
Rule-based	95.0% [91.0–97.3]	78.0% [71.8–83.2]	51.0% [44.1–57.8]

Three readings follow from the finding, and one tempting reading does not. *Implicit constraint learning*: a constraint need not be programmed if the demonstrations respect it consistently—the adversarial framework recovered a reward whose optimum lies on the compliant side of a boundary that was never named to it. *Human-aligned margins*: the policy does not merely avoid violations; it holds

margins close to those the controllers held (Table 2, 7 NM and 10 NM columns)—carrying not just *what is safe* but *how much margin an expert finds comfortable*. *Transferability*: a learned reward is a function of the geometry rather than an enumeration of cases, though this paper does not test that claim. The reading that does *not* follow is that safety was learned from nothing: the standard is absent from the objective but not from the system, and Section 7 sets out what the observation already encodes.

6.4 Behavioural differences: AIRL versus BC

The two learned policies are trained on the same demonstrations and share an architecture, yet they resolve conflicts differently in three respects, illustrated by Figure 3: AIRL intervenes *earlier*, it frequently selects *different waypoints* that yield larger separation margins than the controller’s own choice, and its separation profiles are *smoother*.

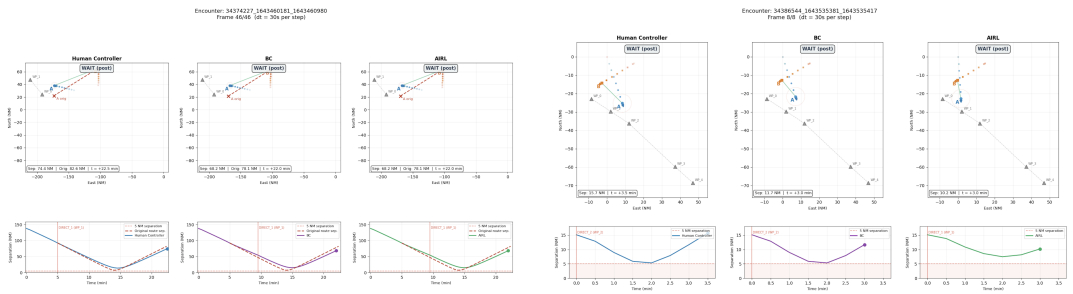


Figure 3. Behavioural differences on held-out development encounters. *Left*: both BC and AIRL concentrate their DIRECT predictions near the controller’s step—both have learned *when* to act—with the AIRL distribution slightly earlier; acting sooner buys the margin, since an earlier turn needs a smaller heading change to open the same CPA. *Right*: separation against time for one encounter under the recorded human resolution, BC and AIRL; BC reproduces the controller’s waypoint and with it a tight 5.3 NM minimum, while AIRL picks a different waypoint and holds 7.5 NM. Illustrative of a behavioural difference; Table 2 carries the aggregate result.

6.4.1 Earlier intervention

The early-intervention prior, combined with the learned reward, shifts the AIRL policy’s DIRECT distribution ahead of BC’s (Figure 3, left).

6.4.2 Different waypoint selection

AIRL frequently selects a waypoint other than the one the controller chose, and the waypoints it prefers yield larger separation margins; BC, trained on those labels, reproduces the controller’s choice—and inherits the controller’s margin. This is the first point at which agreement with the expert and safety of the outcome pull apart. The effect is visible in the rollouts and in the aggregate margins of Table 2.

6.4.3 Smoother separation profiles

The AIRL rollouts trace smoother separation profiles with higher minima, where BC’s show sharp dips toward the threshold before recovering (Figure 3, right).

6.5 Ablation: the early-intervention prior

Removing the early-intervention prior leaves the 5 NM result on the development split unchanged—compliance comes from the learned reward, not the prior—but tightens margins: median minimum separation falls from 7.8 NM to 6.2 NM. **The prior buys margin; the reward buys compliance.** The first half of that claim is the weaker one: with zero violations in both conditions, the ablation

cannot distinguish compliance rates finer than the width of the interval in Table 2, and the shift in median margin is the part of the result the design actually resolves.

6.6 What the metric framework shows

The three metric families jointly explain the headline numbers, and each would be misleading alone. Per-step accuracy is dominated by the 93.1% WAIT floor: a policy that never intervenes scores about 93% overall, so BC's 89.9% cannot be read as a quality measure at all, and any per-step figure quoted for this task is uninterpretable without the WAIT/DIRECT breakdown beside it. Waypoint agreement and safety are not the same objective: the policy that reproduces the controller's recorded waypoint most faithfully is BC, and BC is the policy that breaches the minimum most often, because the margin a waypoint buys matters more than which waypoint the controller happened to choose on the day. We state this as a qualitative ordering rather than a measured gap, since a per-step waypoint-agreement figure for the AIRL policy would require the multi-seed runs listed as future work. A policy that is safer than the controller it imitates is a plausible decision-support tool, but it is no longer straightforwardly an imitator—and that distinction matters for how such a system would be validated.

7. Discussion

What is and is not absent from the system. The result invites a stronger reading than it supports. No separation threshold, penalty or distance rule appears in the reward, the shaping prior or the loss; the policy is never told that going below 5 NM is bad. The constant is not absent from the system as a whole: the observation expresses both the current predicted CPA and each candidate waypoint's simulated CPA in units of the separation standard, so the regulatory boundary sits at exactly 1.0 in feature space; one context feature switches on whether predicted CPA falls below that standard; and the simulator ends an episode when predicted CPA exceeds twice it. The policy therefore has the information it needs to respect the minimum, and any claim that safety was learned from nothing would be wrong. What imitation supplies is the *valuation* placed on that information—how much margin is enough, and how early to spend route efficiency to secure it—the quantity a hand-designed reward would have to encode by fiat.

How far the development figure can be pushed. The 200 encounters are a held-out but selected population: the same split informed model selection, and the design decisions reported in Sections 3 and 4 were taken with its results in view. A zero-violation count on such a split is evidence of an ordering between methods evaluated identically, not an estimate of the rate at which this policy would respect the minimum on traffic it has never seen. The claim we make is the comparative one; establishing the absolute figure requires a split frozen before training and never consulted during design, which is the first item of future work.

The evaluation is simulated, and partly circular. Only the human row of Table 2 corresponds to a real instruction. The policy rows are rollouts under straight-line kinematics in which the second aircraft does not react and a DIRECT resolves instantly into a new heading. That same kinematic model generates the CPA features the policy reads, so the policy is scored partly by the model it was optimised against; and the termination rule stops episodes once predicted CPA exceeds twice the minimum, so a compliant rollout may not have been followed to the end. Fast-time simulation with realistic dynamics and a responding second aircraft is the necessary next step, and we expect the compliance figures to fall.

Selection bias. All results hold on the 28% of encounters from which a direct-to instruction could be recovered. The excluded majority contains the heading-vector resolutions, which arise disproportionately in the harder cases—short time-to-CPA, no suitable waypoint, overtake geometry. A

two-stage policy adding a small heading head to the pointer network would close this gap.

Statistical strength. The Wilson intervals describe binomial sampling uncertainty over encounters at a *fixed* checkpoint; they say nothing about how much the result would move if the policy were retrained. The AIRL figures come from a single training run (seed 42), and the BC development row, at three seeds, is the only one carrying an estimate of seed-to-seed variation. Adversarial training is the less stable of the two procedures, so the missing variance estimate is missing exactly where it would matter most.

The no-action baseline is missing. Not every predicted conflict needs an instruction: some proportion resolve under the aircraft’s own filed routes, and nothing in the outcome tables measures that proportion. A policy that never issues a DIRECT is the natural floor for the comparison; we expect that floor to sit well below the learned rows, since these encounters entered the catalogue precisely because a controller acted—but an expectation is not a measurement, and it is the first row we would add.

Only one side of the trade-off is measured. Every result here scores separation and nothing else. A direct-to instruction carries a cost—extra track miles, fuel burn, and downstream sequencing consequences—and a policy that buys margin by intervening earlier and routing more aggressively pays that cost even when its separation column looks good; pairing the separation proportions with track-mileage and fuel estimates would make the comparison two-dimensional.

Implications beyond ATC. The lesson generalises even where the domain does not: the difference between cloning expert actions and recovering the valuation behind them is measurable, and in safety-critical settings it is the difference that matters. The Tower Parrot template—preprocessing that recovers the missing labels, IRL that recovers the reward, and a validation regime that confirms constraint compliance—applies wherever expert behaviour is recorded but its rationale is not.

Path to deployment. The path to deployment as a decision-support tool follows the gaps identified here: (1) a two-stage DIRECT/HEADING policy, where the pointer network handles direct-to instructions and a small continuous head handles heading vectors, closing the selection gap; (2) multi-seed and fast-time re-evaluation on a split frozen before training, so the compliance figures become measurements rather than optimistic bounds; (3) multi-sector validation on further ACCs; and (4) human-in-the-loop evaluation in which controllers review AIRL suggestions.

8. Conclusion

Recovering controller instructions from trajectory geometry makes a large body of unlabelled surveillance data usable for imitation learning, and a reward inferred from those demonstrations produces the most separation-compliant learned policy in our comparison without any separation term in its objective. On 200 held-out encounters the AIRL policy produced no trajectory below the ICAO 5 NM standard, against 27% of encounters for behavioural cloning and 5% for a heuristic that greedily maximises predicted CPA from the same inputs (Fisher’s exact test, $p = 1.8 \times 10^{-18}$ and $p = 0.0017$). We do not read that as a compliance rate: the split guided design, the rollouts are simulated, and the policy comes from a single seed. We read it as an ordering between methods scored under identical conditions. Imitation does not supply the boundary, which the observation already encodes in units of the separation standard; it supplies the margin and the timing that controllers treat as acceptable—a recovered valuation rather than a memorised action map.

Tower Parrot demonstrates that the barrier to learning air traffic control—the absence of instruction labels—can be overcome, and that what is learned beyond it is closer to controller judgement than to a memorised action map. With the barrier removed, the path is open to automation that augments

human controllers rather than replacing them, learning from their expertise rather than second-guessing it. The next steps are adversarial rather than confirmatory: multiple seeds, a fast-time simulator the policy was not optimised against, heading-vector coverage, and human-in-the-loop evaluation—the infrastructure for all of it is in place and open.

Acknowledgements

We thank Xavier Olive and Kim Gaume for making the Bordeaux ACC deconfliction dataset publicly available, and the OpenSky Network for the infrastructure that produced the underlying ADS-B data. This work originated in the MSc thesis of the first author at the Universitat de Barcelona under the supervision of the co-authors.

Open Data Statement

The Bordeaux ACC deconfliction dataset is publicly available through the Journal of Open Aviation Science [1]. The model, feature and training code is available at https://github.com/U150401/tower_parrot_public under an MIT licence, including the preprocessing scripts, the training pipelines, the split manifests with checksums, and the evaluation notebooks that produce every table in this paper. Recovered labels, trained checkpoints and the per-encounter rollout records behind Table 2 are released as artefacts.

Reproducibility Statement

All experiments are reproducible from the public repository: (1) download the AIRAC-2201 encounter files of the Bordeaux ACC catalogue; (2) run the split script to reconstruct the train/development manifests (seed 42); (3) train the BC checkpoint and warm-start AIRL from it with the hyperparameters of Table 1; (4) run the evaluation notebooks to reproduce the per-step figures and the safety proportions of Table 2, including the Wilson intervals. The only resources not redistributable are the encounter files, available from the dataset publisher.

References

- [1] Kim Gaume, Xavier Olive, and Luis Basora. “A Catalogue of Deconfliction Actions Extracted from Historical ADS-B Data”. In: *Journal of Open Aviation Science* 1 (2023). DOI: 10.59490/joas.2023.7222.
- [2] International Civil Aviation Organization. *Procedures for Air Navigation Services — Air Traffic Management*. Technical Report Doc 4444. ICAO, 2016.
- [3] Matthias Schäfer, Martin Strohmeier, Vincent Lenders, Ivan Martinovic, and Matthias Wilhelm. “Bringing up OpenSky: A large-scale ADS-B sensor network for research”. In: *IPSN-14 proceedings of the 13th international symposium on information processing in sensor networks*. IEEE, 2014, pp. 83–94.
- [4] Marc Brittain and Peng Wei. “Autonomous Aircraft Sequencing and Separation with Hierarchical Deep Reinforcement Learning”. In: *Proceedings of the International Conference for Research in Air Transportation (ICRAT)*. 2018.
- [5] Duc-Thinh Pham, Ngoc Phu Tran, Sim Kuan Goh, Sameer Alam, and Vu Duong. “A Machine Learning Approach for Conflict Resolution in Dense Traffic Scenarios with Uncertainties”. In: *13th USA/Europe Air Traffic Management Research and Development Seminar (ATM Seminar)*. 2019.

- [6] Yash Guleria, Duc-Thinh Pham, Sameer Alam, Phu N. Tran, and Nicolas Durand. “Towards Conformal Automation in Air Traffic Control: Learning Conflict Resolution Strategies through Behavior Cloning”. In: *Advanced Engineering Informatics* 59 (2024), p. 102273. doi: 10.1016/j.aei.2023.102273. 469 470 471 472
- [7] Alevizos Bastas and George A. Vouros. “Data-Driven Modeling of Air Traffic Controllers’ Policy to Resolve Conflicts”. In: *Aerospace* 10.6 (2023), p. 557. doi: 10.3390/aerospace10060557. 473 474
- [8] Michael Bloem and Nicholas Bambos. “Ground Delay Program Analytics with Behavioral Cloning and Inverse Reinforcement Learning”. In: *Journal of Aerospace Information Systems* 12.3 (2015), pp. 299–313. 475 476 477
- [9] Ekaterina Tolstaya, Alejandro Ribeiro, Vijay Kumar, and Ashish Kapoor. “Inverse Optimal Planning for Air Traffic Control”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019. doi: 10.1109/IROS40897.2019.8968460. arXiv: 1903.10525. 478 479 480
- [10] Justin Fu, Katie Luo, and Sergey Levine. “Learning Robust Rewards with Adversarial Inverse Reinforcement Learning”. In: *International Conference on Learning Representations (ICLR)*. 2018. 481 482
- [11] Andrew Y. Ng and Stuart Russell. “Algorithms for Inverse Reinforcement Learning”. In: *Proceedings of the 17th International Conference on Machine Learning (ICML)*. 2000, pp. 663–670. 483 484
- [12] Pieter Abbeel and Andrew Y. Ng. “Apprenticeship Learning via Inverse Reinforcement Learning”. In: *Proceedings of the 21st International Conference on Machine Learning (ICML)*. 2004. 485 486
- [13] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. “Maximum Entropy Inverse Reinforcement Learning”. In: *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*. 2008, pp. 1433–1438. 487 488 489
- [14] Jonathan Ho and Stefano Ermon. “Generative Adversarial Imitation Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. 2016. 490 491
- [15] Dean A. Pomerleau. “Efficient Training of Artificial Neural Networks for Autonomous Navigation”. In: *Neural Computation* 3.1 (1991), pp. 88–97. 492 493
- [16] Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. “A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning”. In: *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2011, pp. 627–635. 494 495 496
- [17] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. “Pointer Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 28. 2015. 497 498
- [18] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural Machine Translation by Jointly Learning to Align and Translate”. In: *International Conference on Learning Representations (ICLR)*. 2015. 499 500 501
- [19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 30. 2017. 502 503 504
- [20] Ronald J. Williams. “Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning”. In: *Machine Learning* 8.3–4 (1992), pp. 229–256. 505 506
- [21] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. “Proximal Policy Optimization Algorithms”. In: *arXiv preprint arXiv:1707.06347* (2017). 507 508
- [22] Lawrence D. Brown, T. Tony Cai, and Anirban DasGupta. “Interval Estimation for a Binomial Proportion”. In: *Statistical Science* 16.2 (2001), pp. 101–133. 509 510