NASH EQUILIBRIUM-INSPIRED OPPONENT POPULATIONS FOR BAYESIAN FPS SKILL DIAGNOSTICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Estimating new player skill is a critical cold-start problem every multiplayer online game faces to deliver fair, competitive, and engaging gameplay. Current approaches over-compress skill into a single transitive rating based on a handful of matches. However, player abilities are multi-dimensional and often non-transitive: a player who dominates one aspect of gameplay may collapse on others. Onboarding bots, a popular solution, are static, easily recognizable by players, and largely uninformative. We introduce GAMBIT, a framework for Bayesian skill diagnostics that uses a calibrated population of autonomous opponents and an adaptive testing loop. Rather than relying on a single transitive ladder, GAMBIT draws on Nash-equilibrium and population-based game-theoretic methods to bootstrap the population with hand-designed archetypes and progressively adds learned specialists to exploit weaknesses. This strategic diversity enables different opponents to probe different aspects of player behavior, becoming simultaneously both the challenge and the diagnostic instrument. GAMBIT operates in two stages. First, scripted and reinforcement-learned opponents are calibrated in a bot-versus-bot tournament to obtain relative difficulty estimates and uncertainty. Second, a new player competes against a short sequence of bot opponents, where after each encounter a skill posterior is updated and the next opponent is adaptively selected to maximize expected information gain. This ensures the framework can be used against a continually evolving corpus of autonomous bots. In simulation, our results demonstrate 56.2% reduction in posterior uncertainty after just 6 opponents, and a final mean rating within 1.4% error margin from the calibrated reference. Furthermore, we analyzed the transferability and generalizability of these bots. Across 64 zero-shot transfers to unseen maps, the full policy stack completed without failure. Together, these results support strategically diverse autonomous opponents for uncertainty-aware cold-start skill estimation.

1 INTRODUCTION

As per the industry estimates, global video-game revenues in 2025 were $188.8 billion, representing 3.4% year-on-year growth (Buijsman, 2025). Yet retaining new players remains challenging: a study reported that 70–90% of new players stopped playing within ten days (Kim et al., 2017), while a US analysis of 2023–2024 mobile-game data reported average Day-30 retention of just 3.2% on Android (Mistplay, 2025). For a more competitive or engaging experience in those games, skill-estimation is the central problem that needs addressing. The problem arises from the fact that this skill is not an absolute or reproducible measure; it depends a lot on the testing environment, player's prior experience, and the relative strength of the opponents against whom the player competes (Walsh & Bokhove, 2021). This creates a problem called "cold-start problem", where new players with no prior history cannot be graded accurately until they acquire some experience on the platform (Zhang et al., 2022). It has also been demonstrated that different rating algorithms differ substantially in both predictive performance and data efficiency (Bober-Irizar et al., 2024), hence in our opinion, they are not measuring the complete profile of a player. Machine learning has tried to address this problem, and methods like QuickSkill demonstrate the value of extracting information before a long match history is available (Zhang et al., 2022). Bayesian and probabilistic rating systems represent skill rating as a distribution whose uncertainty reflects the amount of evidence available (Glickman, 1999; Herbrich et al., 2006; Joshy, 2024). This uncertainty represents a measurement

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

problem, where match outcomes are stochastic, performance can fluctuate, and different opponents may expose different strengths or weaknesses. Further, the player-modeling methods frequently rely on game-specific assumptions or manually selected features, which limits their portability to other interactive environments (Krishnan et al., 2021). We therefore ask whether gameplay itself can serve as the assessment (Demediuk et al., 2018), and benefit from the computerized adaptive testing (CAT) methodology (Klinkenberg et al., 2011). CAT selects exam artifacts sequentially according to the current estimate of ability, allowing tests to concentrate measurements where they are most informative (Zhuang et al., 2024). Bayesian adaptive-testing methods naturally propagate uncertainty in the latent ability estimate and can select future observations according to expected information or posterior uncertainty reduction (van der Linden & Ren, 2020; Su et al., 2025; Yang et al., 2024). Recent developments suggest a useful reinterpretation of the First Person Shooter (FPS) games' skill assessment for the same (Li et al., 2025; Lee & Li, 2026).

The challenge that an FPS-CAT examiner faces is that the examining "item" must be a skillful player of the genre. First-person environments have long served as challenging reinforcement-learning domains because they combine partial observability, continuous or high-dimensional control, navigation, targeting, and delayed rewards (Kempka et al., 2016; Lample & Chaplot, 2016; Jaderberg et al., 2018). At larger scales, systems such as OpenAI Five demonstrate that prolonged self-play can produce extremely strong policies in complex games although at a substantial computational cost (Berner et al., 2019). However, skill assessment requires more than just raw competitive strength, because exploiting privileged state or producing highly unnatural movement may make a strong game-playing agent but a poor measurement instrument. Likewise, a computationally expensive vision-based policy may be impractical when multiple opponent policies must run interactively, at scale, alongside a commercial game. Recent work on tactical-shooter agents has therefore emphasized pixel-free perception using ray-based sensors. With supervised human trajectory learning, it can produce competent and perceptually human-like play with modest inference requirements (Justesen et al., 2025). These considerations suggest that an assessment-oriented agent should be sufficiently strong to challenge the intended player, sufficiently reliable for repeatable diagnostic encounters, sufficiently fair, and sufficiently diverse so that the system does not measure only one narrow style of play. Such strategic diversity introduces an important issue of transitivity, *i.e.* there may be no single ordering such that if policy A defeats B and B defeats C, then A defeats C. Such non-transitivity has motivated population-based training methods that retain multiple strategies rather than a continuous replacement with the single latest policy (Zhang et al., 2024), *e.g.* Policy-Space Response Oracles (PSRO) (Lanctot et al., 2017; Bighashdel et al., 2024). PSRO constructs restricted empirical games from policy populations and iteratively adds responses to strategically relevant opponent mixtures. Work on policy-space diversity further shows that maintaining strategically distinct policies can improve coverage of non-transitive games (Yao et al., 2023).

We propose that a strategically diverse opponent population can act as a set of complementary diagnostic probes. This view is consistent with prior player-modeling results showing that different behavioral features can carry different predictive information about player expertise (Dehpanah et al., 2021). Rather than requiring all opponents to lie on a perfectly transitive ladder, we therefore treat a population of agents as an interactive item bank where opponents may differ both in aggregate difficulty and in the behaviors they probe. In the longer term, this formulation naturally extends toward multidimensional skill models, in which different aspects of the gameplay are treated as distinct latent factors rather than collapsed into a single ability variable (Lee & Li, 2026).

PSRO solves for a meta-strategy over the restricted policy population, and iteratively adds an approximate best response (Lanctot et al., 2017). In a first-person shooter, however, the underlying policy space is extremely large and strongly conditioned by environment geometry, visibility, tactical context and so on. Exhaustively covering all variation through equilibrium-oriented population expansion is therefore computationally impractical. Instead, we use a Nash equilibrium-inspired approximation where we bootstrap the population with a hand-enumerated library of behaviorally distinct opponent archetypes, then progressively add learned specialists that target weaknesses exposed by tournament play.

Thus, we introduce GAMBIT, a framework for uncertainty-aware skill estimation using calibrated autonomous opponents. In the first stage, a heterogeneous library of parametric and learned FPS agents is evaluated through controlled bot-versus-bot tournaments. These tournaments provide a relative difficulty scale and associated uncertainty for the available opponent bank. In the second stage, a new player completes a short sequence of independent player-versus-bot encounters. Following

each encounter, GAMBIT updates a posterior distribution over the player's latent skill and selects the next opponent according to the current posterior. This offers two practical advantages, i) the opponent population can evolve independently of the scoring procedure, ii) the scoring architecture need not assume that exact bot difficulty transfers unchanged between maps, games, or mechanics.

While prior work in player modeling has highlighted the difficulty of transferring game-specific behavioral models across domains (Krishnan et al., 2021), GAMBIT instead aims to transfer the measurement structure of constructing a calibrated set of interactive challenges, representing both item and player uncertainty, and adaptively selecting observations that reduce uncertainty.

2 Theory

2.1 Bayesian Skill and Uncertainty Model

GAMBIT follows the relative-skill intuition of the Elo rating system, originally developed for competitive chess (Elo, 1978). In Elo-style systems, the expected outcome of a match depends on the difference between the ratings of the two competitors rather than on either rating in isolation, *i.e.*

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}}$$

where E_A is the player A's expected score against player B. The numerical factor of 400 defines the scale of chess ratings, and is an arbitrary measure. The updated Elo score for player A is then $R'_A = R_A + K(S_A - E_A)$ where $S_A \in \{1,\ 0.5,\ 0\}$ is the score for winning, drawing, and losing the game. This provides a natural starting point for FPS skill diagnostics, where a player is assessed by repeatedly confronting opponents of known relative difficulty.

For a new player p, latent skill is represented as $R_p \sim \mathcal{N}(\mu_p, \sigma_p^2)$ (Glickman, 1999; Herbrich et al., 2006), with the initial prior $\mu_p^{(0)} = 1000$ and $\sigma_p^{(0)} = \sigma_0$. The value 1000 is a convenient origin for the rating scale rather than an absolute definition of ability. The parameter σ_0 represents the large initial uncertainty associated with an unseen player. As diagnostic encounters accumulate, both the posterior mean and uncertainty are updated.

Opponent difficulty is estimated independently during the preceding bot-versus-bot calibration stage. Each diagnostic bot b is therefore associated with $R_b \sim \mathcal{N}(\mu_b, \sigma_b^2)$, where μ_b denotes its calibrated relative difficulty and σ_b represents uncertainty in that calibration. This separation is important: the bot population defines the reference scale before a new player is assessed. Following Elo formulation, the probability of the player defeating an opponent can first be written as a monotonic function of their rating difference,

$$P(W \mid R_p, R_b) = \frac{1}{1 + 10^{(R_b - R_p)/\kappa}},$$

where κ controls how strongly a given rating difference changes the expected outcome (Bradley & Terry, 1952). In the current work, κ is set to 400.

Unlike a standard point-rating implementation, neither player nor bot skill is assumed to be known exactly (see also van der Linden & Ren, 2020; Ren et al., 2020). For a candidate player ability r, the likelihood of an observed result against bot b marginalizes over uncertainty in the bot rating,

$$P(W \mid r, b) = \int P(W \mid r, R_b)\, p(R_b)\, dR_b.$$

Writing $q(r, b) = P(W \mid r, b)$, losses use the factor $1 - q(r, b)$, while draws use the fractional-score factor $\sqrt{q(r, b)[1 - q(r, b)]}$, corresponding to a score of $1/2$. The draw update is a fractional-likelihood convention, not a separately normalized three-outcome probability model. Thus, an encounter against a bot with a narrowly calibrated difficulty can provide stronger evidence than an otherwise similar encounter against a bot whose difficulty remains uncertain (Glickman, 1999; Fink et al., 2025). As the same formulation can also be used against other players, GAMBIT is intended as a cold-start initialization layer, after which the same probabilistic rating formulation can continue to update player skill from ordinary player-versus-player matches.

After encounter t, producing outcome y_t, the player posterior is updated through Bayes' rule,

$$p(R_p \mid y_{1:t}) \propto p(y_t \mid R_p, b_t)\, p(R_p \mid y_{1:t-1}).$$

Algorithm 1: Bayesian Player Rating Update

Input: Player posterior $p_{t-1}(r)$, bot parameters (μ_b, σ_b), outcome $y_t \in \{\text{win}, \text{draw}, \text{loss}\}$, rating scale κ

Construct player-rating grid $\mathcal{R} = \{r_1, \ldots, r_N\}$ Construct bot-rating grid $\mathcal{B} = \{b_1, \ldots, b_K\}$

Compute normalized bot weights $w_k \propto \mathcal{N}(b_k; \mu_b, \sigma_b^2)$

foreach $r_i \in \mathcal{R}$ **do**

Compute $q_{ik} = \left(1 + 10^{(b_k - r_i)/\kappa}\right)^{-1}$

Marginalize over bot uncertainty $\bar{q}_i = \sum_k w_k q_{ik}$

Set update factor

$$\ell_i = \begin{cases} \bar{q}_i, & y_t = \text{win}, \\ 1 - \bar{q}_i, & y_t = \text{loss}, \\ \sqrt{\bar{q}_i(1 - \bar{q}_i)}, & y_t = \text{draw} \end{cases}$$

$\tilde{p}_i \leftarrow \ell_i \, p_{t-1}(r_i)$

Normalize

$$p_t(r_i) = \frac{\tilde{p}_i}{\sum_j \tilde{p}_j}$$

Compute $\mu_t = \sum_i r_i p_t(r_i)$

Compute $\sigma_t = \sqrt{\sum_i (r_i - \mu_t)^2 p_t(r_i)}$

return $p_t(r), \mu_t, \sigma_t$

The displayed player score and uncertainty are then obtained from the posterior,

$$\text{Rating}_t = \mu_p^{(t)}, \qquad \text{Uncertainty}_t = \sigma_p^{(t)}.$$

It is useful to distinguish rating uncertainty from ordinary performance variability. A player may perform unusually well or poorly in a particular encounter. Such variation belongs to the encounter likelihood or performance-noise model. By contrast, σ_p describes epistemic uncertainty *i.e.* how precisely GAMBIT currently knows the player's underlying relative ability (Herbrich et al., 2006). Repeated consistent observations therefore reduce σ_p, even though individual match outcomes remain stochastic.

The assessment terminates either after a fixed maximum number of encounters $T_{\max}$, or when the player's posterior uncertainty falls below a predefined threshold, $\sigma_p^{(t)} \leq \sigma_{\text{stop}}$(Zhuang et al., 2024). Algorithm 1 summarizes the full Bayesian posterior update sequence.

2.2 Nash Equilibrium-Inspired Opponent Populations

Conventional scalar ratings assume that opponents can be placed on a one-dimensional difficulty axis. In a strategic game this assumption is imperfect. Let

$$\Pi = \{\pi_1, \pi_2, \ldots, \pi_M\}$$

denote a population of opponent policies. We define the expected payoff between two policies as

$$A_{ij} = \mathbb{E}\left[u(\pi_i, \pi_j)\right],$$

where u denotes the terminal payoff from the perspective of π_i: $+1$ for a win, 0 for a draw, and -1 for a loss. Thus, $A_{ij} = P(\pi_i \text{ wins against } \pi_j) - P(\pi_i \text{ loses to } \pi_j)$, and $A_{ij} > 0$ indicates a pairwise advantage. The opponent receives payoff $-u$, so each encounter is zero-sum. We estimate A_{ij} by averaging tournament outcomes.

In general, this payoff structure need not be transitive (Yao et al., 2023). For three policies, it is possible to observe a situation where, $A_{12} > 0$, $A_{23} > 0$, and $A_{31} > 0$. A classical game-theoretic treatment is to search for a mixed strategy over policies. Let $x \in \Delta^M$ denote a probability distribution over the population, where Δ^M is the probability simplex. In a zero-sum restricted game, an equilibrium meta-strategy seeks a distribution whose expected payoff cannot be substantially improved by unilateral deviation. Conceptually,

Algorithm 2: Nash Equilibrium-Inspired Opponent Population Construction

Input: Initial behavioral archetypes Π_0, evaluation scenarios $\mathcal{S}$, maximum generations G, admission criteria $\mathcal{C}$

for $g \leftarrow 1$ **to** G **do**
- Evaluate all relevant policy pairs in Π over $\mathcal{S}$
- Construct empirical payoff matrix $A_{ij} \leftarrow \mathbb{E}[u(\pi_i, \pi_j)]$
- Identify recurrent losses and hard-negative matchup clusters $\mathcal{H}_g$
- **foreach** *high-priority weakness* $h \in \mathcal{H}_g$ **do**
  - Select a competent parent policy $\pi_{\text{parent}} \in \Pi$
  - Collect successful expert and hard-negative trajectories for h
  - Warm-start candidate π_h using BC/DAgger or offline RL when available
  - Fine-tune π_h against the targeted weakness using conservative RL
  - Evaluate π_h against the frozen population and retention suite
  - **if** π_h *satisfies admission criteria* $\mathcal{C}$ **then**
    - $\Pi \leftarrow \Pi \cup \{\pi_h\}$
- Remove or archive redundant policies that add no diagnostic or strategic coverage
- **if** *no useful policy is admitted* **then**
  - **break**;

Calibrate retained policies to obtain (μ_b, σ_b) for each $b \in \Pi$

return Π

$$x^* \approx \arg \max_{x \in \Delta^M} \min_{y \in \Delta^M} x^\top A y.$$

Methods such as Policy-Space Response Oracles (PSRO) approximate this process by constructing an empirical game over a restricted population, solving for a meta-strategy, and iteratively adding approximate best responses to strategically relevant opponent mixtures (Lanctot et al., 2017; Bighashdel et al., 2024). Population-diversity methods similarly emphasize retaining policies with distinct strategic responses and do not repeatedly replace the population with a single latest policy (Yao et al., 2023).

However, our objective is not to compute the Nash equilibrium of the complete underlying game. We therefore use a Nash equilibrium-inspired population construction rather than a full equilibrium solver. The initial restricted population is bootstrapped using hand-enumerated behavioral archetypes representing strategically distinct forms of interaction, such as `rush`, `strafe`, `hold-angle`, `hunt`, `kite`, `generalist`. These archetypes provide an explicit prior over strategically useful regions of policy space instead of requiring self-play to discover them from scratch. Bot-versus-bot tournaments are then used to expose systematic weaknesses and non-transitive matchups. Learned specialists are progressively added when they provide new strategic coverage or counter existing policies. In GAMBIT, the diversity has an additional diagnostic role because different policies act as complementary test items that may expose different aspects of player performance. The basic population generation loop is outlined in algorithm 2.

2.3 Adaptive Opponent Selection

Once the player posterior and calibrated opponent population have been established, the remaining problem is to determine which opponent should be presented next. GAMBIT treats this as an adaptive-testing problem. The current implementation uses expected information gain (EIG) to select the opponent whose possible outcomes are expected to reduce uncertainty in the player-skill estimate most strongly.

Details of opponent selection, and the implementation of underlying methods, can be accessed in the Appendix A.

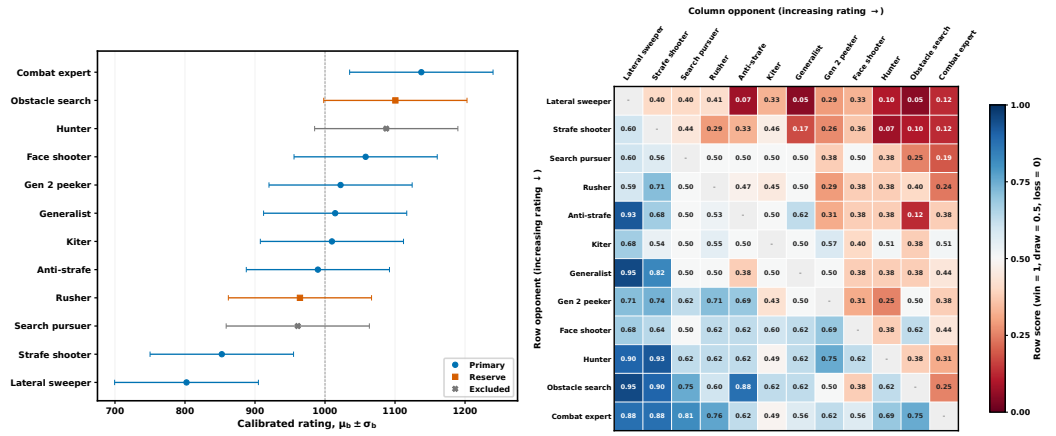


Figure 1: Calibration and population-level matchup structure of the twelve candidate opponents. Left: fitted rating mean and uncertainty. Right: empirical pairwise payoff matrix ($A_{ij} = 2S_{ij} - 1$), illustrating deviations from a purely scalar ordering of opponent strength.

3 Results and Discussion

3.1 Examiner Calibration and Population Diversity

The twelve candidate opponents covered a broad calibrated difficulty range, from $\mu_b = 802.2$ for the broad lateral sweeper to $\mu_b = 1137.6$ for the visible-combat expert. The corresponding rating uncertainties were relatively uniform, ranging from approximately 102.25 to 102.68 rating points. Two candidates were omitted from the final bank, excluded by the population subset-selection objective. Figure 1 shows the calibrated ratings together with the complete pairwise payoff matrix. A three-policy, "rock-paper-scissor-like cycle", was also observed where Kiter > Combat Expert > Obstacle Search > Kiter, with corresponding pairwise scores of 0.5125, 0.75, and 0.625. Policies with similar scalar ratings could also exhibit substantially different payoff profiles. For example, the generalist and Gen 2 peeker differ by only 7.7 rating points but have a common-opponent payoff-profile RMSE of 0.163.

These results support the use of population-level payoff diversity in addition to scalar difficulty when constructing the examiner bank. Physical-side effects were also non-negligible, with a median absolute pairwise side-score difference of 0.25 and a maximum of 0.75. This observation supports the use of paired role swaps during calibration and the inclusion of side asymmetry in the examiner-selection penalty.

3.2 Bayesian Skill-Estimation Demonstration

We next evaluated the complete grading pipeline using a frozen neural proxy player over six adaptive diagnostic encounters. The proxy began with the broad discrete prior (described in Appendix B.6), with initial posterior mean and standard deviation of 1000.0 ± 343.2. As match outcomes were observed, the posterior mean moved according to the strengths of the selected opponents and the corresponding win/loss outcomes, while posterior uncertainty progressively decreased. After six diagnostic encounters, the estimate was

$$1000.0 \pm 343.2 \quad \longrightarrow \quad 1000.6 \pm 150.2.$$

This corresponds to a 56.2% reduction in posterior standard deviation relative to the initial discrete posterior. The proxy's independently fitted value within the twelve-bot calibration was $\mu = 1014.5$, placing the final diagnostic posterior mean 13.9 rating points from this reference value. The posterior trajectory is shown in Figure 2. Importantly, this experiment is a systems demonstration using a neural proxy rather than a human validation study, and the proxy reference rating is derived from the same calibrated population rather than from an independent ground-truth skill measure. Nevertheless, the experiment demonstrates that a short sequence of adaptively selected encounters can substantially

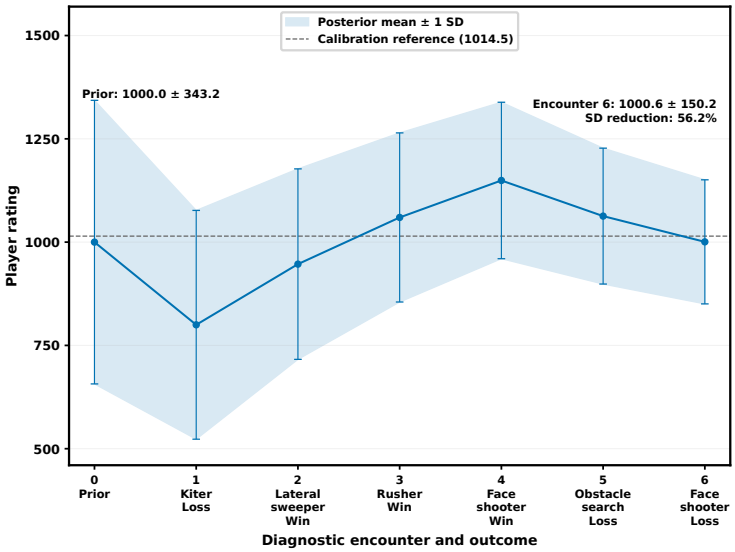


Figure 2: Evolution of the Bayesian player-skill estimate during the six-encounter neural-proxy demonstration. The line shows posterior mean rating and the uncertainty interval shows posterior standard deviation after each diagnostic encounter.

Table 1: Six-encounter backend replay comparison of opponent-selection strategies.

Strategy	**Final σ_p**	**Error**	**EIG**	**Brier**
Active EIG	148.34	50.91	0.753	0.331
Random	153.54	86.64	0.681	0.334
Fixed easy-to-hard	155.24	60.81	0.727	0.331
Nearest rating	151.46	184.38	0.752	0.355

reduce rating uncertainty while maintaining a posterior estimate close to the proxy's calibrated reference value.

3.3 Adaptive Opponent Selection

We compared the EIG-based opponent selector (as a separate frozen-outcome replay session) with three simpler alternatives: random selection, a fixed easy-to-hard ordering, and selection of the examiner nearest to the current posterior mean. All four strategies were evaluated through six-encounter replays of the frozen round-robin outcomes using the same session seed. Table 1 summarizes the resulting posterior uncertainty, error relative to the proxy's calibrated rating, cumulative expected information gain, and predictive Brier score. In this replay, active EIG produced the smallest final posterior uncertainty and the largest cumulative expected information gain. It also produced the smallest absolute error relative to the proxy's calibrated reference rating. The nearest-rating selector achieved a similar cumulative information gain but produced substantially larger endpoint rating error, suggesting that selecting an opponent solely by proximity to the current skill estimate does not necessarily provide the most informative sequence of matchups. The fixed easy-to-hard strategy produced a predictive Brier score nearly identical to active EIG, while random selection gave both lower cumulative information gain and higher posterior uncertainty.

3.4 Qualitative Navigation and Systems Behavior

In addition to the quantitative calibration and grading experiments, we visually inspected autonomous bot-vs-bot rollouts to examine whether the learned navigation policies produced coherent search, movement, and engagement behavior. Representative sequences showed navigation around local geometry, movement along obstacle boundaries, search after loss of contact, and transitions from navigation to visible combat after LOS was re-established. Figure 3 shows representative frames from

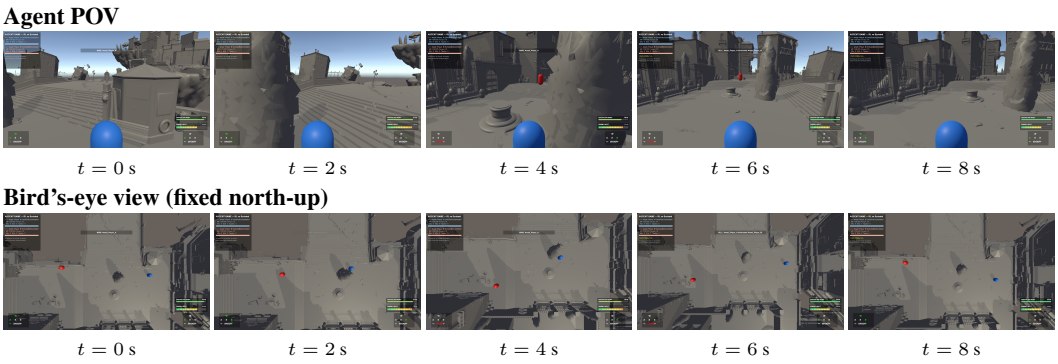


Figure 3: Representative qualitative behavior of a navigation-enabled Gen 2 opponent during autonomous play. The sequence illustrates local navigation, search, reacquisition, and transition to visible combat. These examples are qualitative and are not used as a quantitative measure of navigation generalization.

these autonomous rollouts. These behaviors arise from the local actor231 observation and recurrent navigation policy together with the runtime search, safety, and tactical-control mechanisms (described in Appendix B.1). We therefore interpret the examples as qualitative evidence of coherent autonomous navigation rather than as isolated demonstrations of a particular learned mechanism. The navigation-enabled agents were also executed on a second authored arena without modifying the deployed observation or action interface. Successful execution on this second map demonstrates runtime and interface transfer beyond the primary environment configuration. However, these recordings alone do not establish quantitative behavioral generalization across maps, and we therefore treat them as a systems-level demonstration rather than a formal transfer benchmark.

3.5 Limitations and Conclusion

The results demonstrate that a heterogeneous FPS opponent population can be calibrated into a useful diagnostic bank while retaining matchup structure that is not captured by a single scalar ranking. In particular, the observed payoff cycles and differences between similarly rated policies support the use of strategically diverse opponents as complementary diagnostic probes rather than as a conventional difficulty ladder. The Bayesian assessment demonstration further shows that a short sequence of controlled encounters can substantially reduce uncertainty in an initially broad player-skill estimate.

However, the present study should be interpreted primarily as a systems and methodology demonstration. The player-assessment experiment uses a frozen neural proxy rather than human participants, and the proxy reference rating is derived from the same calibrated population rather than from an independent measure of player skill. Consequently, the current results establish internal consistency of the grading procedure but do not yet establish validity against human rank, matchmaking rating, or long-term competitive performance. The comparison between active EIG, random selection, fixed ordering, and nearest-rating selection is likewise based on a single frozen replay configuration. Although active EIG produced lower posterior uncertainty and lower endpoint error in this comparison, repeated trials across multiple players, seeds, and skill levels are required before drawing stronger conclusions about selector performance. Calibration was performed on a limited procedural validation environment, and the observed physical-side effects indicate that encounter geometry can influence measured outcomes. Paired role swaps reduce this bias, but future calibration should include additional maps, spawn configurations, and environmental conditions to determine how stable examiner ratings remain across environments. Also our assessment treats uncertainty in each bot's rating separately, although some uncertainty is shared across bots. We have not yet tested how accounting for this shared uncertainty would affect the reported reduction in uncertainty about player skill. The examiner uncertainties are obtained from a local Laplace approximation to the regularized Bradley–Terry model and therefore represent approximate calibration uncertainty rather than an exact posterior distribution. In addition, the player model ultimately represents skill using a single scalar rating. This simplification is useful for matchmaking and compact assessment, but FPS competence is inherently multidimensional, and players with similar overall ratings may differ substantially in

tracking, positioning, search, aggression, or movement skill. The present examiner population partly exposes such differences through heterogeneous matchup profiles, but the current scoring model does not yet convert them into separate player-skill dimensions. The population itself is also finite and was selected using a hand-designed utility balancing payoff diversity, rating coverage, reliability, and side asymmetry. Future work can replace or augment this fixed-bank construction with iterative population expansion, including PSRO-like procedures, larger end-to-end policies, and specialized vision-based agents.

Finally, broader validation should include controlled human studies, repeated assessments, independent skill references, and held-out competitive matches. Such experiments would allow the framework to be evaluated not only by posterior contraction, but also by test–retest reliability, predictive validity, and correspondence with established measures of player skill.

AI USE STATEMENT

Generative AI tools were used for the following:

- To revise the manuscript prose for clarity, brevity, and grammar. They also assisted with LaTeX formatting and adaptation to the ICLR 2027 submission template.
- Large portions of supplementary information, especially the literal architecture enumerations and LaTeX tables for vector elements, were created using AI.
- For preparing the playable Unity Arena from the `fbx` maps provided, and also for assistance with code cleanup and documentation.
- For a literary informed suggestions on bootstrapping hyperparameter settings (*e.g.* learning rates, PPO clipping parameters, etc). These suggestions served as starting points for author consideration rather than as evidence of optimality or community consensus.

REFERENCES

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafał Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d. O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning, 2019. URL https://arxiv.org/abs/1912.06680. arXiv:1912.06680.

Ariyan Bighashdel, Yongzhao Wang, Stephen McAleer, Rahul Savani, and Frans A. Oliehoek. Policy space response oracles: A survey, 2024. URL https://arxiv.org/abs/2403.02227. arXiv:2403.02227.

Mikel Bober-Irizar, Naunidh Dua, and Max McGuinness. Skill issues: An analysis of CS:GO skill rating systems, 2024. URL https://arxiv.org/abs/2410.02831. arXiv:2410.02831.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. doi: 10.1093/biomet/39.3-4.324. URL https://doi.org/10.1093/biomet/39.3-4.324.

Michiel Buijsman. Global games market to hit $189 billion in 2025 as growth shifts to console. Newzoo, September 2025. URL https://newzoo.com/articles/global-games-market-189-billion-2025. September 9.

Arman Dehpanah, Muheeb Faizan Ghori, Jonathan Gemmell, and Bamshad Mobasher. Player modeling using behavioral signals in competitive online games, 2021. URL https://arxiv.org/abs/2112.04379. arXiv:2112.04379.

Simon Demediuk, Marco Tamassia, William L. Raffe, Fabio Zambetta, Florian "Floyd" Mueller, and Xiaodong Li. Measuring player skill using dynamic difficulty adjustment. In *Proceedings of the Australasian Computer Science Week Multiconference*, ACSW '18, pp. 1–7. Association for Computing Machinery, 2018. doi: 10.1145/3167918.3167939. URL https://doi.org/10.1145/3167918.3167939.

Arpad E. Elo. *The Rating of Chessplayers, Past and Present*. Arco Publishing, New York, 1978. ISBN 0-668-04721-6.

Aron Fink, Christoph König, and Andreas Frey. Accounting for item calibration error in computerized adaptive testing. *Behavior Research Methods*, 57(5):126, 2025. doi: 10.3758/s13428-025-02649-8. URL https://doi.org/10.3758/s13428-025-02649-8.

Mark E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 48(3):377–394, 1999. doi: 10.1111/1467-9876.00159. URL https://doi.org/10.1111/1467-9876.00159.

Ralf Herbrich, Tom Minka, and Thore Graepel. TrueSkill: A bayesian skill rating system. In *Advances in Neural Information Processing Systems 19*, 2006. URL https://proceedings.neurips.cc/paper_files/paper/2006/hash/f44ee263952e65b3610b8ba51229d1f9-Abstract.html.

Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, and Thore Graepel. Human-level performance in first-person multiplayer games with population-based deep reinforcement learning, 2018. URL https://arxiv.org/abs/1807.01281. arXiv:1807.01281.

Vivek Joshy. OpenSkill: A faster asymmetric multi-team, multiplayer rating system, 2024. URL https://arxiv.org/abs/2401.05451. arXiv:2401.05451.

Niels Justesen, Maria Kaselimi, Sam Snodgrass, Miruna Vozaru, Matthew Schlegel, Jonas Wingren, Gabriella A. B. Barros, Tobias Mahlmann, Shyam Sudhakaran, Wesley Kerr, Albert Wang, Christoffer Holmgård, Georgios N. Yannakakis, Sebastian Risi, and Julian Togelius. Human-like bots for tactical shooters using compute-efficient sensors, 2025. URL https://arxiv.org/abs/2501.00078. arXiv:2501.00078.

Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. ViZDoom: A Doom-based AI research platform for visual reinforcement learning, 2016. URL https://arxiv.org/abs/1605.02097. arXiv:1605.02097.

Seungwook Kim, Daeyoung Choi, Eunjung Lee, and Wonjong Rhee. Churn prediction of mobile and online casual games using play log data. *PLOS ONE*, 12(7):e0180735, July 2017. doi: 10.1371/journal.pone.0180735. URL https://doi.org/10.1371/journal.pone.0180735.

S. Klinkenberg, M. Straatemeier, and H. L. J. van der Maas. Computer adaptive practice of maths ability using a new item response model for on the fly ability and difficulty estimation. *Computers & Education*, 57(2):1813–1824, 2011. doi: 10.1016/j.compedu.2011.02.003. URL https://www.sciencedirect.com/science/article/pii/S0360131511000418.

Abhijeet Krishnan, Aaron Williams, and Chris Martens. Towards action model learning for player modeling, 2021. URL https://arxiv.org/abs/2103.05682. arXiv:2103.05682.

Guillaume Lample and Devendra Singh Chaplot. Playing FPS games with deep reinforcement learning, 2016. URL https://arxiv.org/abs/1609.05521. arXiv:1609.05521.

Marc Lanctot, Vinicius Zambaldi, Audrūnas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3323fe11e9595c09af38fe67567a9394-Abstract.html.

Seungwon Lee and Xiaoou Li. On the asymptotics of item selection in multidimensional computerized adaptive testing, 2026. URL https://arxiv.org/abs/2608.00201. arXiv:2608.00201.

Jiguang Li, Robert Gibbons, and Veronika Ročková. Deep computerized adaptive testing, 2025. URL https://arxiv.org/abs/2502.19275. arXiv:2502.19275.

Mistplay. Mobile gaming loyalty in 2025: Genres driving monetization, engagement, and LTV. Mistplay, May 2025. URL https://business.mistplay.com/resources/mobile-gaming-trends-2025-report. May 16. Summary of the 2025 Mobile Gaming Loyalty Index, produced in collaboration with AppsFlyer.

Hao Ren, Seung W. Choi, and Wim J. van der Linden. Bayesian adaptive testing with polytomous items. *Behaviormetrika*, 47(2):427–449, 2020. doi: 10.1007/s41237-020-00114-8. URL https://doi.org/10.1007/s41237-020-00114-8.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635. PMLR, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Tina Su, Edison Choe, and Joshua C. Chang. Bayesian model-averaging stochastic item selection for adaptive testing, 2025. URL https://arxiv.org/abs/2504.15543. arXiv:2504.15543.

Wim J. van der Linden and Hao Ren. A fast and simple algorithm for Bayesian adaptive testing. *Journal of Educational and Behavioral Statistics*, 45(1):58–85, 2020. doi: 10.3102/1076998619858970. URL https://doi.org/10.3102/1076998619858970.

Clare Walsh and Christian Bokhove. Targeting data collection in games based assessment. *Computers and Education Open*, 2:100054, 2021. doi: 10.1016/j.caeo.2021.100054. URL https://www.sciencedirect.com/science/article/pii/S2666557321000252.

Hau-Hung Yang, Chia-Min Wei, and Yu-Chang Chen. On the consistency of bayesian adaptive testing under the Rasch model, 2024. URL https://arxiv.org/abs/2412.07170. arXiv:2412.07170.

Jian Yao, Weiming Liu, Haobo Fu, Yaodong Yang, Stephen McAleer, Qiang Fu, and Wei Yang. Policy space diversity for non-transitive games, 2023. URL https://arxiv.org/abs/2306.16884. arXiv:2306.16884.

Chaoyun Zhang, Kai Wang, Hao Chen, Ge Fan, Yingjie Li, Lifang Wu, and Bingchao Zheng. QuickSkill: Novice skill estimation in online multiplayer games, 2022. URL https://arxiv.org/abs/2208.07704. arXiv:2208.07704.

Ruize Zhang, Zelai Xu, Chengdong Ma, Chao Yu, Wei-Wei Tu, Wenhao Tang, Shiyu Huang, Deheng Ye, Wenbo Ding, Yaodong Yang, and Yu Wang. A survey on self-play methods in reinforcement learning, 2024. URL https://arxiv.org/abs/2408.01072. arXiv:2408.01072.

Yan Zhuang, Qi Liu, Haoyang Bi, Zhenya Huang, Weizhe Huang, Jiatong Li, Junhao Yu, Zirui Liu, Zirui Hu, Yuting Hong, Zachary A. Pardos, Haiping Ma, Mengxiao Zhu, Shijin Wang, and Enhong Chen. Survey of computerized adaptive testing: A machine learning perspective, 2024. URL https://arxiv.org/abs/2404.00712. arXiv:2404.00712.

A Adaptive Opponent Selection

After t diagnostic encounters, let $p_t(r)$ denote the current posterior distribution over player rating, and let

$$\mathcal{B} = \{b_1, b_2, \ldots, b_M\}$$

denote the calibrated opponent bank. For each candidate opponent b, GAMBIT considers the possible encounter outcomes $y \in \mathcal{Y}$ with $\mathcal{Y}$ denoting the set of outcomes used by the scoring model, for example win, draw, and loss.

For a candidate opponent b, the predictive probability of outcome y is obtained by averaging the outcome likelihood over the current player posterior:

$$P(y \mid b, D_t) = \int P(y \mid r, b)\, p_t(r)\, dr,$$

where D_t denotes all observations collected before encounter $t+1$. The likelihood $P(y \mid r, b)$ already marginalizes uncertainty in the calibrated opponent rating as described in Section 2.1. If outcome y were subsequently observed against opponent b, the corresponding hypothetical posterior would be

$$p(r \mid D_t, y, b) \propto P(y \mid r, b)\, p_t(r).$$

We quantify uncertainty in the player posterior using Shannon entropy,

$$H[p_t] = -\int p_t(r) \log p_t(r)\, dr.$$

The expected posterior entropy resulting from selecting opponent b is

$$\mathbb{E}\left[H_{t+1} \mid b\right] = \sum_{y \in \mathcal{Y}} P(y \mid b, D_t)\, H[p(r \mid D_t, y, b)]\,.$$

The expected information gain associated with opponent b is therefore

$$\mathrm{IG}(b) = H[p_t] - \mathbb{E}\left[H_{t+1} \mid b\right].$$

GAMBIT selects the next opponent according to

$$b^*_{t+1} = \arg\max_{b \in \mathcal{B}} \mathrm{IG}(b).$$

The criterion has a simple interpretation. An opponent far below the player's current estimated skill is likely to be defeated and therefore usually provides little additional evidence. Likewise, an overwhelmingly difficult opponent is likely to produce a predictable loss. Opponents for which multiple outcomes remain plausible tend to be more informative because the observed result can more strongly reshape the posterior. Unlike a simple nearest-rating rule, however, EIG considers the complete player posterior and the uncertainty associated with each calibrated opponent rather than relying only on the distance between their mean ratings. Algorithm 3 summarizes the opponent-selection procedure used by GAMBIT.

B Methods

B.1 Environment and Agent Interface

Diagnostic opponents run autonomous one-versus-one combat in a Unity FPS with procedural arenas and headless execution. A fixed 0.02 s timestep gives 50 Hz control decisions with no action repeat. The runtime separates navigation/search, using a 231-dimensional observation, from visible combat, using a 45-dimensional observation.

The map-independent, agent-relative navigation vector encodes self motion and state, previous actions, and local obstacle and floor sensing. Torso-, foot-, and head-height radial rays plus downward floor and step probes expose nearby free space, walls, and drops without a global navigation route. Unity scales these features analytically, so the deployed navigator uses an identity normalizer rather than fitted whitening.

Algorithm 3: Expected-Information-Gain Opponent Selection

Input: Current player posterior $p_t(r)$, calibrated opponent bank $\mathcal{B}$
Compute current posterior entropy $H_t \leftarrow H[p_t(r)]$
foreach $b \in \mathcal{B}$ **do**
  $\bar{H}_b \leftarrow 0$
  foreach $y \in \mathcal{Y}$ **do**
    Compute predictive outcome probability $P_y \leftarrow \int P(y \mid r, b)\, p_t(r)\, dr$
    Compute hypothetical posterior $\tilde{p}^{(y,b)}(r) \leftarrow P(y \mid r, b)\, p_t(r)$
    Normalize

$$p^{(y,b)}(r) \leftarrow \frac{\tilde{p}^{(y,b)}(r)}{\int \tilde{p}^{(y,b)}(r')\, dr'}$$

    $\bar{H}_b \leftarrow \bar{H}_b + P_y H[p^{(y,b)}(r)]$
  $\mathrm{IG}(b) \leftarrow H_t - \bar{H}_b$
$b^* \leftarrow \arg\max_{b \in \mathcal{B}} \mathrm{IG}(b)$
return b^*

The visible-opponent block contains LOS, relative bearing and elevation, distance, and relative velocity; all target-specific fields are zero without geometric LOS. Hidden-target search instead uses stale last-seen position, a noisy incoming-damage direction, and a coarse hunt cue. The hunt cue quantizes bearing into eight sectors and distance into five Unity distance unit bins, $[0, 4)$, $[4, 8)$, $[8, 16)$, $[16, 32)$, and $[32, \infty)$; it refreshes at most every 0.5 s with quantization noise and default dropout probability 0.15. Search can therefore continue without exact through-wall coordinates.

Although the 45-dimensional telemetry transport retains precise target-relative quantities under occlusion, the combat network is evaluated only with LOS:

$$(a_t^{\text{combat}}, h_{t+1}^{\text{combat}}) = \begin{cases} \pi_{\text{combat}}\left(o_t^{\text{combat}}, h_t^{\text{combat}}\right), & \text{LOS}_t = 1, \\ (\varnothing, h_t^{\text{combat}}), & \text{LOS}_t = 0, \end{cases}$$

where o_t^{combat} is the combat observation and h_t^{combat} its recurrent state. Without LOS, neither inference nor recurrent-state advancement uses hidden-target telemetry. The observation uses its frozen training-time per-feature affine normalizer. LOS is a Unity raycast that ignores self and trigger collisions and requires the first relevant collider to belong to the opponent; no camera field-of-view restriction is added, reducing computation and sustaining the combat policy's required frame rate.

Agents produce four clipped continuous controls,

$$a_t^c = [m_x, m_z, \ell_{\text{yaw}}, \ell_{\text{pitch}}], \qquad a_t^c \in [-1, 1]^4$$

for lateral movement, forward movement, yaw, and pitch, plus four binary controls,

$$a_t^d = [a_{\text{shoot}}, a_{\text{reload}}, a_{\text{jump}}, a_{\text{crouch}}], \qquad a_t^d \in \{0, 1\}^4.$$

Deterministic execution uses continuous-distribution means and activates binary actions above probability 0.5; stochastic execution samples the corresponding Gaussian and Bernoulli distributions. Final examiner calibration is deterministic to reduce action-sampling variability.

Navigation and combat recurrent states persist within an encounter but are cleared by encounter termination or explicit environment/session reset, preventing intentional carryover between opponents. Loss of LOS freezes rather than resets combat state until visibility resumes. Navigation also receives short- and long-window failed-movement indicators spanning approximately 0.5 and 2.0 s at 50 Hz for collision and stall recovery. Complete index-level definitions and implementation-specific observation details appear in Appendix C.

B.2 Examiner Population Construction

The twelve candidates are autonomous opponent bots, comprising six deterministic parametric anchors and six learned or hybrid policies. The parametric bots represent easily interpretable combat

Table 2: Behavioral composition of the twelve-opponent candidate population.

Class	Candidate behavior	Implementation type
Parametric	Stationary precision shooter	Deterministic parameterized controller
Parametric	Lateral strafe shooter	Deterministic parameterized controller
Parametric	Forward rusher	Deterministic parameterized controller
Parametric	Backpedaling kiter	Deterministic parameterized controller
Parametric	Broad lateral sweeper	Deterministic parameterized controller
Parametric	Search pursuer	Deterministic parameterized controller
Learned/hybrid	Visible-combat specialist	Recurrent learned combat policy
Learned/hybrid	Generalist	Learned combat with fair search and safety
Learned/hybrid	Anti-strafe specialist	Learned combat with fair search and safety
Learned/hybrid	Obstacle-search pursuer	Learned combat with fair search and safety
Learned/hybrid	Map-general hunter	Learned navigation with frozen combat expert
Learned/hybrid	Cover-oriented peeker	Learned navigation with frozen combat expert and tactical adapter

styles: a stationary shooter holds position and aims accurately; a strafe shooter moves repeatedly from side to side while firing; a rusher closes distance aggressively; a kiter retreats while continuing to engage; a sweeper makes broad lateral movements to change its firing position; and a search pursuer actively looks for an opponent after losing sight of them. The learned bots extend these behaviors with more adaptive policies, including a balanced generalist, a specialist trained to counter lateral movement, a high-precision visible-combat expert, an obstacle-aware search policy, a persistent hunter that pursues hidden opponents, and a peeker that attempts to use cover and expose itself only briefly during engagements. These bots were chosen to span different behavioral and matchup profiles rather than to form a simple ladder of increasing strength. This yielded the six behaviors listed in Table 2. They provide repeatable tactical patterns without training a separate neural policy for each behavior.

Learned candidates arise from distinct training stages. Some pair a recurrent combat controller with fair scripted search and safety, while later hybrids combine learned 231-dimensional navigation, a frozen visible-combat expert, and stateful tactical adapters. Thus, diversity spans both policy behavior and controller structure; Table 2 lists every candidate and implementation.

Together, anchors and learned/hybrid agents span scalar difficulty and pairwise matchup behavior: anchors occupy controlled diagnostic regions, while learned agents respond more richly to movement, obstacles, search, reacquisition, and combat context. Selection therefore considers reliability, rating coverage, and payoff-profile diversity rather than scalar rank alone. The frozen bank contains eight primary and two reserve examiners — five learned/hybrid and five parametric (four primary + one reserve each) — and tournament calibration assigns each a difficulty estimate and uncertainty (Section B.5).

Gen 0 denotes deterministic parametric controllers; Gen 1 adds learned combat within scripted search and safety; and Gen 2 adds learned local navigation/search, visibility-gated combat, and tactical adaptation. These generations coexist because simpler agents remain useful controlled probes. Gen 3 is future work and is excluded from the reported experiments.

B.3 Learned Policy Training and Bootstrapping

The learned examiners were developed through a staged training pipeline combining supervised imitation, dataset aggregation, and reinforcement-learning refinement. The final examiner-construction and calibration pipeline itself performs no policy training. Instead, it loads frozen checkpoints inherited from earlier training stages and evaluates them under a common runtime and calibration protocol.

The first learned generation focuses primarily on visible combat. Combat policies are initialized using behavioral cloning (BC) on a dataset of 139,101 labeled transitions from 1,644 gameplay

sessions. We partitioned the data by session, allocating 111,680 transitions to training, 12,712 to validation, and 14,709 to testing. These trajectories contain observation–action sequences collected from human players and provide the demonstration data used to learn the initial mapping from combat observations to aiming, firing, and local movement actions. Behavioral cloning therefore provides an initial policy that imitates human actions in states represented within the demonstration dataset. To improve robustness outside the human demonstration distribution, the policy is subsequently refined using Dataset Aggregation (DAgger) (Ross et al., 2011).

During DAgger, the current learner is allowed to visit states induced by its own policy, after which a scripted Unity teacher provides corrective action labels for those states. The resulting learner-state demonstrations are combined with retained training data and used for further supervised policy updates. For navigator BC/DAgger, teacher-labelled observations were collected along trajectories generated by mixtures of teacher and learner control. Training used 273,020 base samples and 130,549 supplemental samples, with 6,068 samples reserved for validation. Privileged teacher information was used for supervision but excluded from the actor's observation inputs.

After BC and DAgger initialization, selected combat policies are further specialized through PPO-based reinforcement-learning continuations (Schulman et al., 2017). Different PPO continuations produce behaviorally distinct descendants, including the generalist, anti-strafe, visible-combat, and obstacle/search policy families. These descendants are not treated as independent policies trained from scratch. Instead, they share inherited initialization and diverge through different continuation objectives, opponent mixtures, and training stages.

The second learned generation adds a separate navigation and search policy operating on the 231-dimensional observation described in Section B.1. The navigator is initialized using scripted-teacher supervision and DAgger-style learner-state collection: training rollouts mix teacher- and learner-controlled behavior, while the student receives only the deployable actor observation. Privileged information may construct teacher labels or training targets but is excluded from actor inputs. Staged reinforcement-learning continuations then emphasize search, pursuit, reacquisition after loss of contact, obstacle handling, and cover-oriented tactical behavior, producing hunter, reacquirer, and peeker-style descendants. These agents retain the frozen recurrent visible-combat expert, so navigation can improve without retraining the aiming and firing controller.

All policy checkpoints and observation normalizers are frozen before examiner screening and final evaluation. No online learning occurs during examiner selection, tournament calibration, or player assessment. The selected learned policies share a common training ancestry but arise from different continuation branches and therefore do not use a single universal PPO configuration. Rollout construction, learning rates, KL constraints, opponent mixtures, and reinforcement-learning objectives vary across policy generations and specialist continuations.

We therefore report the shared training structure in the main text and provide lineage-specific optimization settings and checkpoint provenance in the supplementary material.

Policy architectures. The Gen 1 combat policy processes the 45-dimensional combat observation using three feature branches corresponding to spatial (*where*), view/aim, and temporal/action-rhythm information. The three branches receive 10, 13, and 22 features, respectively, and each produces a 128-dimensional embedding using two fully connected layers with LayerNorm and GELU activations; the rhythm branch additionally applies input dropout of 0.1. The resulting 384-dimensional representation is fused through two fully connected layers to a 512-dimensional representation and passed to a single-layer GRU with hidden size 512. The recurrent state feeds separate heads for four continuous action means, four binary-action logits, and the state-value estimate, together with four learned state-independent continuous-action log-standard deviations. Gen 2 introduces a separate recurrent navigator operating on the 231-dimensional actor observation. The navigator separates 152 local geometry features from 79 contextual features, encodes them independently, fuses them into a 256-dimensional representation, and processes the result using a single-layer GRU with hidden size 128. Its output heads predict movement, look control, tactical mode, action blending, and an auxiliary local-geometry target. During navigator PPO training, value estimation is performed by a separate feed-forward critic receiving a 428-dimensional privileged observation; these privileged features are never supplied to the deployed actor. Gen 2 hunter, reacquirer, and peeker policies combine this navigator with the frozen Gen 1 combat expert and runtime handoff, search, recovery, and tactical-control logic. Figure 4 summarizes the two deployed neural architectures. The actor231

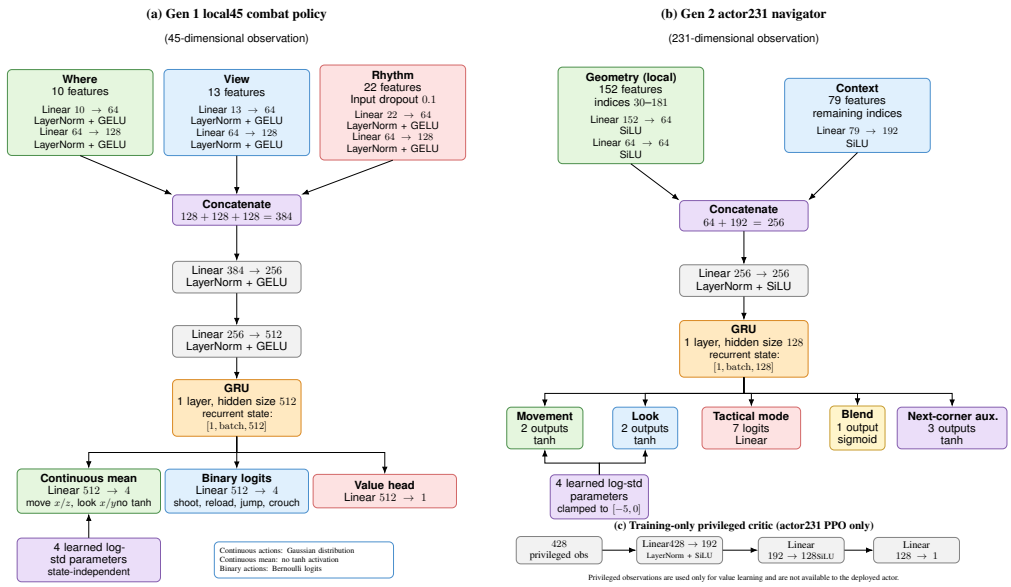


Figure 4: Neural architectures used by the learned opponent policies. The Gen 1 local45 combat network uses separate spatial, view, and rhythm encoders followed by a 512-unit recurrent GRU and continuous, binary, and value heads. The Gen 2 actor231 navigator separately embeds local geometry and contextual information before a 128-unit recurrent GRU and navigation/tactical output heads. During actor231 PPO training, value estimation uses a separate privileged critic that is excluded from the deployed actor.

network does not itself replace the Gen 1 combat action heads; the final Gen 2 bots combine navigator outputs with the frozen local45 combat expert through the runtime handoff and tactical-composition logic described in Section B.1.

B.4 Examiner Screening and Bank Selection

The final examiner bank is selected from the calibrated candidate population using both reliability and population-level behavioral diversity. The objective is not to retain only the strongest opponents, but to construct a compact set that spans a useful range of difficulty while preserving distinct matchup characteristics. Selection therefore uses the tournament ratings and pairwise payoff profiles described in Section B.5. Candidates are first screened for unreliable or degenerate behavior. A candidate is excluded if any of the following conditions hold, i) $\sigma_b > 120$, ii) $T_b > 0.90$, iii) $\max_j s_{bj} < 0.05$, or iv) $\min_j s_{bj} > 0.95$, where σ_b is the calibrated rating uncertainty, T_b is the timeout rate, and s_{bj} is the empirical pairwise score of candidate b against opponent j. The first two conditions remove candidates whose calibrated strength is poorly determined or whose matches are dominated by timeouts. The latter two remove opponents that are nearly ineffective against the entire population or nearly dominant against every opponent, since either extreme provides limited discrimination over much of the intended player-skill range.

For each remaining candidate, we define a payoff profile from its empirical scores against the rest of the calibrated population. Two candidates are considered behaviorally distinct when these payoff profiles differ, even if their scalar ratings are similar. For candidates i and j, payoff-profile distance is measured using the root-mean-square difference between their pairwise score vectors, d_{ij}^{payoff}. This quantity captures differences in who each policy performs well or poorly against, rather than only differences in average strength.

Selection is performed separately within the learned/hybrid and parametric-anchor classes. For a candidate subset $\mathcal{S}$, the implemented utility combines payoff-profile diversity, rating coverage, and a penalty for unreliable or asymmetric behavior, *i.e.*

$$U(\mathcal{S}) = 8D(\mathcal{S}) + \frac{\max_{b\in\mathcal{S}} \mu_b - \min_{b\in\mathcal{S}} \mu_b}{400} - P(\mathcal{S}),$$

where

$$D(\mathcal{S}) = \mathrm{mean}_{i<j,\, i,j\in\mathcal{S}}\, d_{ij}^{\mathrm{payoff}}$$

is the mean pairwise payoff-profile RMS distance, and μ_b is the calibrated rating mean of examiner b. The penalty term is

$$P(\mathcal{S}) = \mathrm{mean}_{b\in\mathcal{S}} \left(A_b + 0.25\, T_b + \frac{\sigma_b}{300} \right),$$

where A_b denotes the maximum observed side-dependent score difference for candidate b, T_b its timeout rate, and σ_b its calibrated rating uncertainty. The first term of U therefore rewards strategically distinct matchup profiles, the second rewards coverage of a broad difficulty interval, and the final term discourages side-sensitive, timeout-prone, or weakly calibrated candidates. The coefficient on payoff-profile diversity is intentionally dominant, reflecting the use of the examiner population as a set of complementary diagnostic probes rather than as a simple strength ladder.

Within each policy class, the procedure selects a five-member subset maximizing the above utility. Four members are designated as primary examiners and one as a reserve, yielding four learned/hybrid primaries and four parametric primaries together with one reserve from each class. The learned-primary set is additionally constrained to contain at least one Gen 2 (`bank_phase5_`) policy so that the deployed bank includes a navigation-enabled examiner. The resulting bank therefore contains eight primary examiners and two reserves, with equal representation from learned/hybrid and parametric policy classes. This procedure deliberately permits a weaker opponent to remain in the bank when its payoff profile contributes useful diagnostic diversity. Conversely, a strong opponent may be excluded if its matchup behavior is redundant with policies already represented in the selected subset. An earlier single-bot screening stage was used during development to identify gross behavioral failures, but its contact, damage, kill-conversion, and timeout thresholds were not used as universal admission criteria for the final examiner bank.

B.5 Tournament Calibration and Bot Rating

All twelve candidate opponents are calibrated jointly before the final examiner subset is selected. Calibration is performed through a symmetric round-robin tournament so that every candidate is evaluated against every other candidate under matched encounter conditions. With $N = 12$ candidates, the tournament contains $\binom{12}{2} = 66$ unordered opponent pairs. For each pair, we evaluate 20 paired random seeds, and each seed is played twice with the physical player roles exchanged. Thus, every unordered pair contributes 40 matches, giving $66 \times 20 \times 2 = 2640$ calibration encounters in total.

The calibration encounters use the procedural validation arena family described in Section B.1, with layouts cycled across paired seeds. All learned policies are executed deterministically during calibration so that continuous actions use the policy mean and binary actions use their deterministic thresholded decisions. Each encounter terminates on a kill/death event or after a maximum duration of 30 simulation seconds. A timeout is treated as a draw during bot calibration.

B.5.1 Pairwise outcome representation.

Let $u_{ij} = u(\pi_i, \pi_j)$ denote an individual terminal payoff for candidate i against candidate j: $+1$ for a win, 0 for a draw, and -1 for a loss. The payoff-matrix entry $A_{ij} = \mathbb{E}[u_{ij}]$ is estimated by averaging these outcomes over tournament encounters. For Bradley–Terry fitting, each terminal payoff u_{ij} is mapped to the fractional score

$$y_{ij} = \frac{u_{ij} + 1}{2} \in \left\{ 0, \frac{1}{2}, 1 \right\}.$$

Thus, wins contribute score 1, losses score 0, and draws contribute the symmetric fractional score $1/2$. Each candidate is assigned a latent rating

$$R_i \sim \mathcal{N}(1000, 350^2),$$

where 1000 defines the center of the rating scale and 350 represents the prior rating uncertainty. For two candidates i and j, the expected probability that i defeats j is modeled using the Elo-scaled Bradley–Terry relation

$$P(i > j) = \frac{1}{1 + 10^{(R_j - R_i)/400}}.$$

Equivalently, defining the centered rating $r_i = R_i - 1000$, the matchup probability can be written as

$$P(i > j) = \sigma\left(\frac{r_i - r_j}{400/\ln 10}\right),$$

where $\sigma(\cdot)$ denotes the logistic sigmoid. The denominator 400 therefore controls the sensitivity of expected outcome probability to rating difference, while the value 1000 only fixes the origin of the rating scale. Let $p_{ij} = P(i > j)$, the calibrated ratings are obtained by maximizing the Gaussian-regularized fractional Bradley–Terry objective

$$\mathcal{L}(\mathbf{r}) = \sum_{(i,j)} [y_{ij} \log p_{ij} + (1 - y_{ij}) \log(1 - p_{ij})] - \sum_i \frac{r_i^2}{2(350)^2}.$$

The fractional score for a draw therefore enters the same pairwise likelihood symmetrically rather than introducing a separate three-outcome draw model. Optimization is performed using Newton updates, and the fitted centered ratings are recentered such that the mean rating across the twelve calibrated candidates remains 1000.

The final rating estimate for candidate i is

$$\mu_i = 1000 + \hat{r}_i.$$

B.5.2 Rating uncertainty.

In addition to the calibrated mean rating, we associate each candidate with an approximate uncertainty derived from the curvature of the regularized Bradley–Terry objective around its optimum. Let $\mathcal{I}(\hat{\mathbf{r}})$ denote the regularized observed information matrix at the fitted solution. A local Gaussian, or Laplace, approximation gives

$$\mathbf{\Sigma} \approx \mathcal{I}(\hat{\mathbf{r}})^{-1},$$

and the uncertainty assigned to candidate i is

$$\sigma_i = \sqrt{\Sigma_{ii}}.$$

We therefore represent each calibrated examiner as

$$R_i \approx \mathcal{N}(\mu_i, \sigma_i^2).$$

The quantity σ_i is an approximate local posterior uncertainty obtained from the inverse Hessian and should not be interpreted as an exact Bayesian posterior standard deviation. The final ten-examiner subset is selected from the jointly calibrated twelve-candidate population and is not recentered after subset selection.

B.6 Online Player Skill Assessment

A new player is initialized with a broad skill prior centered at rating 1000 with standard deviation 350, represented numerically on a discrete rating grid from 0 to 2000. For each encounter, the win probability against examiner b is obtained from the same Elo-scaled logistic model used for bot calibration while marginalizing over the examiner's calibrated uncertainty (μ_b, σ_b) using Gauss–Hermite quadrature. After observing the encounter outcome, the player's rating distribution is updated using Bayes' rule, and its posterior mean and standard deviation are reported as the current skill estimate and uncertainty. Draws are supported during posterior updating through a symmetric fractional likelihood, but are not modeled as a normalized third outcome in the current EIG calculation; opponent selection evaluates only prospective win and loss branches. The next examiner is chosen adaptively by maximizing expected information gain over the current posterior, subject to simple constraints preventing an immediate opponent repeat and, when possible, encouraging a change in behavioral style. This process is repeated for the short diagnostic sequence, progressively concentrating the posterior around the player's estimated skill.

C Supplementary Agent Interface Details

This section provides additional implementation details for the observation, action, visibility, and execution interfaces summarized in Section B.1. The main Methods section defines the deployed observation split and visibility boundary. This supplement records the index-level feature contracts, frozen normalization statistics, geometric constants, and compatibility fields needed for implementation and audit.

C.1 VISIBLE-COMBAT OBSERVATION VECTOR

The visible-combat policy consumes a 45-dimensional vector referred to as `local45`. For each raw component x_i, the deployed combat normalizer applies

$$z_i = \mathrm{clip}\left(\frac{x_i - \mu_i}{s_i}, -10, 10\right),$$

where μ_i and s_i are the frozen per-feature mean and effective standard deviation, respectively. The resulting neural input therefore lies in the interval $[-10, 10]$ for every dimension.

Table 3 lists the implemented fields. Unity spatial quantities are reported in Unity length units because a separate physical-metre calibration was not established.

Table 3: Implemented 45-dimensional visible-combat observation vector. μ and s denote the frozen normalization mean and effective standard deviation.

Idx.	Feature	Units/range	μ	s	Notes
0	opponent relative position x	Unity units	0.3576	0.8207	Self-local target displacement; combat-policy access is LOS-gated.
1	opponent relative position y	Unity units	7.6150	70.7315	Self-local target displacement; combat-policy access is LOS-gated.
2	opponent relative position z	Unity units	252.6452	333.7487	Self-local target displacement; combat-policy access is LOS-gated.
3	self local velocity x	Unity units/s	-0.0269	2.0493	Finite-difference estimate; reset on large teleport.
4	self local velocity y	Unity units/s	-22.3661	21.1903	Finite-difference estimate; reset on large teleport.
5	self local velocity z	Unity units/s	-3.9610	32.2308	Finite-difference estimate; reset on large teleport.
6	self local acceleration x	Unity units/s^2	-0.0787	27.6415	Finite-difference estimate.
7	self local acceleration y	Unity units/s^2	-9.8225	163.3345	Finite-difference estimate.
8	self local acceleration z	Unity units/s^2	-1.8574	135.6932	Finite-difference estimate.
9	opponent distance	Unity units	256.3730	338.4559	Scalar target distance.
10	sin(pitch error)	$[-1, 1]$	-0.0151	0.1444	Target-relative aiming feature.
11	cos(pitch error)	$[-1, 1]$	0.9827	0.1154	Target-relative aiming feature.
12	sin(yaw error)	$[-1, 1]$	0.1147	0.2919	Target-relative aiming feature.
13	cos(yaw error)	$[-1, 1]$	0.9315	0.1845	Target-relative aiming feature.
14	view Euler velocity x	deg/s	-0.2509	19.8297	Finite difference of Unity body Euler angle.
15	view Euler velocity y	deg/s	20.8871	83.8870	Finite difference of Unity body Euler angle.
16	view Euler velocity z	deg/s	3.74×10^{-5}	2.11×10^{-4}	Finite difference of Unity body Euler angle.
17	view Euler acceleration x	deg/s^2	2.1605	293.3884	Angular finite-difference feature.
18	view Euler acceleration y	deg/s^2	-45.7755	1312.7844	Angular finite-difference feature.
19	view Euler acceleration z	deg/s^2	1.41×10^{-4}	0.01683	Angular finite-difference feature.
20	yaw error	degrees	7.9911	21.0007	Signed target-relative yaw error.
21	pitch error	degrees	-1.2003	12.4738	Target-relative pitch error.
22	aim error magnitude	degrees	16.4602	19.7734	$\sqrt{\mathrm{yaw}^2 + \mathrm{pitch}^2}$.

Idx.	Feature	Units/range	μ	s	Notes
23	previous shoot command	$\{0, 1\}$	0.9344	0.2476	Previous shoot command, not necessarily a confirmed weapon discharge.
24	unused action-history slot 1	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
25	unused action-history slot 2	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
26	unused action-history slot 3	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
27	unused action-history slot 4	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
28	unused action-history slot 5	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
29	unused action-history slot 6	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
30	unused action-history slot 7	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
31	unused action-history slot 8	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
32	unused action-history slot 9	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
33	unused action-history slot 10	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
34	unused action-history slot 11	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
35	unused action-history slot 12	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
36	unused action-history slot 13	$\{0\}$	0	10^{-6}	Reserved array position; not written by current implementation.
37	action count total	$\{0, 1\}$	0.9344	0.2476	Only the shoot command currently increments this field.
38	has any action	$\{0, 1\}$	0.9344	0.2476	Reflects the currently tracked shoot action.
39	opponent health fraction	$[0, 1]$	0.4709	0.3391	Opponent health scalar; visible-combat access is LOS-gated.
40	time since action	$[0, 1]$	0.01419	0.11618	Elapsed time clipped to 5 s and divided by 5.
41	time since shot command	$[0, 1]$	0.01419	0.11618	Elapsed time clipped to 5 s and divided by 5.
42	time since reload	$[0, 1]$	1.0000	10^{-6}	Current implementation does not advance the reload timestamp, so this component remains at its sentinel value.

Idx.	Feature	Units/range	μ	s	Notes
43	time since targeted hit	$[0, 1]$	0.7368	0.4220	Elapsed time clipped to 5 s and divided by 5.
44	shot fired this step	$\{0, 1\}$	0.09530	0.29362	Weapon-confirmed shot indicator.

Several entries in this vector therefore have zero or near-zero variance in the frozen normalizer. These dimensions are retained for compatibility with the historical observation contract rather than removed from the deployed checkpoint interface.

C.2 Line-of-Sight and Contact Definition

The ray endpoints are offset approximately 0.7 Unity units above the two player origins.

In the 231-dimensional actor interface, the deployed LOS flag is stored at index 193. Thus,

$$\mathrm{LOS}_t = \mathbb{I}\left[o^{\mathrm{nav}}_{t,193} > 0.5\right].$$

The LOS definition does not additionally impose a camera field-of-view test. For evaluation telemetry, an encounter is said to achieve *contact* if at least one control step contains a positive LOS observation. First-contact latency is the elapsed simulation time until the first such observation. Reacquisition is defined as a subsequent transition from absent to present LOS after an earlier loss of contact.

C.3 Navigation and Search Observation Vector

The index-level contract below uses an agent-relative yaw frame wherever applicable, with local x pointing right and local z pointing forward.

Table 4 summarizes the observation layout. The majority of the navigation observation is devoted to local obstacle and floor geometry. Thirty-two torso-height rays are distributed uniformly around the agent at 11.25° intervals. Each ray has a maximum length of 40 Unity units and contributes two values: normalized hit distance and a binary hit indicator. A miss is represented by distance 1 and hit flag 0.

An additional set of sixteen foot-height and sixteen head-height rays samples the surrounding geometry at 22.5° spacing, using origins approximately 0.25 and 1.45 Unity units above the agent, respectively. These measurements allow the policy to distinguish low and high obstacles without using a global navigation mesh.

Eight additional radial floor probes estimate nearby ground continuity and step-like obstacles. For each direction, a downward ray measures floor distance, while paired low and high level horizontal rays provide a binary obstacle-height heuristic. Together these measurements supply local information about walls, free space, drops, and traversable-looking steps. Indices 193–201 contain the LOS flag and exact visible-target features; when LOS is absent, $o_{t,193} = 0$ and $o_{t,194:201} = \mathbf{0}$. Indices 202–206 store last-seen memory, whose age is normalized over 10 s. Indices 207–210 store noisy damage-direction memory and should not be interpreted as an auditory or footstep sensor. The hunt cue at indices 211–226 applies neighboring-bin noise in addition to the quantization, refresh, and dropout rules reported in Methods.

A requested-movement step is classified as stuck when 3D displacement is below 0.02 Unity units. The short and long histories contain up to 25 and 100 control steps, respectively, corresponding nominally to 0.5 and 2.0 s at 50 Hz. Collision frequency, collision normal, and time since displacement progress provide additional recovery signals.

The final four observation dimensions contain mutually exclusive, deterministically constructed tactical indicators for patrol, investigate, combat, and recovery. These flags are derived from visibility, remembered target information, and stuck history. They are sensor-level features and should not be confused with either the historical recurrent `TemporalExpertSelector` or the EIG opponent-selection algorithm used for adaptive player assessment.

Table 4: Group structure of the 231-dimensional navigation/search observation. Indices are zero based.

Indices	Dim.	Group	Description
0–2	3	Self velocity	Agent velocity expressed in the current yaw-relative coordinate frame.
3–5	3	Angular motion	Finite differences of body Euler angles for pitch, yaw, and roll.
6–7	2	Ground/crouch state	Grounded and physically crouched indicators.
8–11	4	Health and weapon state	Health fraction, magazine fraction, reserved-ammo placeholder, and weapon cooldown.
12–15	4	Recent-damage cue	Validity, noisy incoming-damage bearing, and cue age.
16–23	8	Previous action	Previous movement, look, shoot, reload, jump, and crouch commands.
24–29	6	Progress/stuck history	Speed, displacement, short- and long-window stuck fractions, collision fraction, and time since movement progress.
30–93	64	Torso obstacle rays	Thirty-two horizontal yaw-relative rays, each represented by normalized hit distance and hit-validity flag.
94–157	64	Foot/head obstacle rays	Sixteen foot-height and sixteen head-height radial rays, each represented by distance and hit flag.
158–181	24	Floor/drop/step probes	Eight radial probes containing downward floor distance, floor-validity, and step-obstacle indicators.
182–184	3	Collision normal	Most recent collision normal expressed in the current yaw frame.
185–192	8	Free-space summary	Minimum torso-ray clearance within eight coarse angular sectors.
193–201	9	Visible enemy state	LOS flag, target bearing/elevation, log distance, and relative velocity; all exact target fields are zero when LOS is absent.
202–206	5	Last-seen memory	Validity, remembered opponent position expressed relative to the agent, and memory age.
207–210	4	Damage-derived directional memory	Validity, noisy stored bearing, and age derived from incoming-hit events.
211–226	16	Coarse hunt cue	Eight-way bearing one-hot, five-way distance one-hot, validity, age, and dropout status.
227–230	4	Tactical mode	Deterministic patrol, investigate, combat, and recovery sensor flags.

D Training Lineages and Optimization Details

D.1 Gen 1 Recurrent PPO Configuration

For these PPO continuations, the inherited observation normalizer remains frozen and actions are sampled stochastically during training. Table 5 summarizes the shared optimization configuration.

The recurrent minibatch contains four trajectory fragments of length 16, corresponding to approximately 0.32 s of temporal context at the 50 Hz control frequency. Unlike conventional PPO implementations that perform multiple optimization epochs over all minibatches from a rollout, the inspected Phase 4.4 driver performs one optimizer update using the first recurrent minibatch produced from each rollout. New environment interaction is therefore collected between successive optimizer updates.

D.2 Anti-Drift Motivation for Conservative PPO

The conservative PPO configuration was motivated by earlier warm-start continuation experiments in which repeated PPO updates gradually degraded useful pretrained combat behavior. In an earlier retention study, the warm-start policy initially retained effective aiming behavior but subsequently collapsed after approximately 37 PPO iterations. The degradation occurred despite small instantaneous policy changes, with mean per-iteration KL divergence of approximately 0.003. This indicated

Table 5: Shared recurrent PPO configuration for the Gen 1 combat-policy continuations.

Parameter	Setting
Policy backbone	`local45` encoder + GRU(512)
Optimizer	Adam
PPO clip parameter	0.1
Discount factor γ	0.99
GAE parameter λ	0.95
Value-loss coefficient	0.5
Entropy coefficient	0.003
Nominal rollout	256 steps $\times$ 2 arenas
Learner transitions / rollout	512
Recurrent minibatch	4 sequences $\times$ 16 steps
Optimization / rollout	1 optimizer step
Observation normalizer	Frozen
Training actions	Stochastic

Table 6: Initial Phase 4.4 specialist PPO continuation settings.

Family	Learning rate	KL threshold
Generalist	6.0×10^{-7}	0.0012
Anti-temporal-selector	5.5×10^{-7}	0.0010
Anti-v000 / visible	6.0×10^{-7}	0.0012
Anti-strafe	6.5×10^{-7}	0.0010
Underfire retaliator	6.0×10^{-7}	0.0010
Evasive survivor	5.0×10^{-7}	0.0010
Search / obstacle pursuer	4.5×10^{-7}	0.0010

that individually small updates could accumulate into substantial behavioral drift over a long continuation. A subsequent anti-drift configuration reduced the number of PPO epochs from four to one, strengthened the KL penalty relative to the warm-start reference policy, and tightened the KL stopping threshold. Under the combined configuration, useful combat behavior persisted for approximately 100 iterations rather than approximately 37 iterations. The intervention therefore substantially delayed behavioral collapse, although long-horizon drift was not completely eliminated. These experiments motivated the later use of small learning rates, strong policy-change constraints, frozen observation normalization, and limited optimization during specialist PPO continuations. The reduction in PPO epochs was evaluated together with stronger KL anchoring and a tighter KL threshold; the retention improvement should therefore be attributed to the combined anti-drift configuration rather than to the epoch reduction alone.

D.3 Gen 1 Specialist Continuation Families

The initial Phase 4.4 population was constructed through several specialist PPO continuation families initialized from existing BC/DAgger or residual expert checkpoints. The families shared the recurrent PPO implementation described above but used different learning rates, KL stopping thresholds, opponent mixtures, and continuation objectives. Table 6 reports the initial family-level optimization settings. The initial launcher requested up to 120 PPO updates for each family. Individual training segments could terminate earlier under their corresponding guard conditions. Later repair and population-expansion continuations used modified configurations and are therefore treated as separate continuation stages rather than as part of a single universal PPO recipe. Although several family configuration files contain family-specific entropy coefficients, the inspected GvG PPO driver applies an entropy coefficient of 0.003 for these continuations. Scripted-retention runs follow a separate optimization path and are not assumed to share this behavior.

Table 7: Training ancestry of the learned and hybrid examiners retained in the final bank.

Examiner	Principal training lineage
Generalist	Human-telemetry BC → DAgger refinement → generalist PPO continuation.
Visible-combat expert	Generalist combat checkpoint → conservative PPO repair / retention continuation.
Anti-strafe specialist	Human-telemetry BC → DAgger refinement → anti-strafe PPO continuation.
Obstacle-search pursuer	BC → DAgger with stronger retention → search / obstacle PPO continuations → repair and expansion continuation.
Peeker	Actor231 scripted-teacher BC / DAgger → hunter PPO → reacquisition PPO → peeking / local-geometry PPO → later tactical continuation, combined with a frozen visible-combat expert.

D.4 Selected Learned-Policy Lineages

Table 7 summarizes the principal training ancestry of the learned and hybrid policies retained in the final examiner bank. The table reports conceptual lineage rather than every intermediate checkpoint produced during development.

D.5 Gen 2 Navigation Training and Hybrid Assembly

Gen 2 navigator continuations use a rollout and optimization procedure distinct from the Phase 4.4 recurrent combat-policy driver. Consequently, the Gen 1 PPO hyperparameters in Appendix D.1 should not be assigned to the navigator by assumption.

D.6 Training Stability under Conservative PPO Refinement

The learned combat policies were initialized from pretrained behavior rather than trained from scratch, making retention of existing combat competence an important consideration during PPO refinement. In an earlier warm-start continuation, the policy initially retained effective aiming behavior but progressively drifted away from this solution and reached a persistent zero-hit regime at approximately iteration 37. A subsequent anti-drift configuration reduced PPO epochs from four to one, strengthened the reference-policy KL penalty, and tightened the KL constraint. Under this combined configuration, useful combat behavior was retained substantially longer, with non-zero hit counts still observed near iteration 101 and collapse occurring only after approximately 100 iterations. Thus, the intervention increased the duration of behavioral retention by roughly a factor of 2.7. Importantly, this comparison does not isolate the contribution of any single hyperparameter, since the PPO epoch count, KL anchoring, and KL threshold were modified together. The result instead motivated the conservative optimization strategy used in later specialist continuations, where PPO was treated as a small refinement of an already competent policy rather than as unrestricted policy optimization.

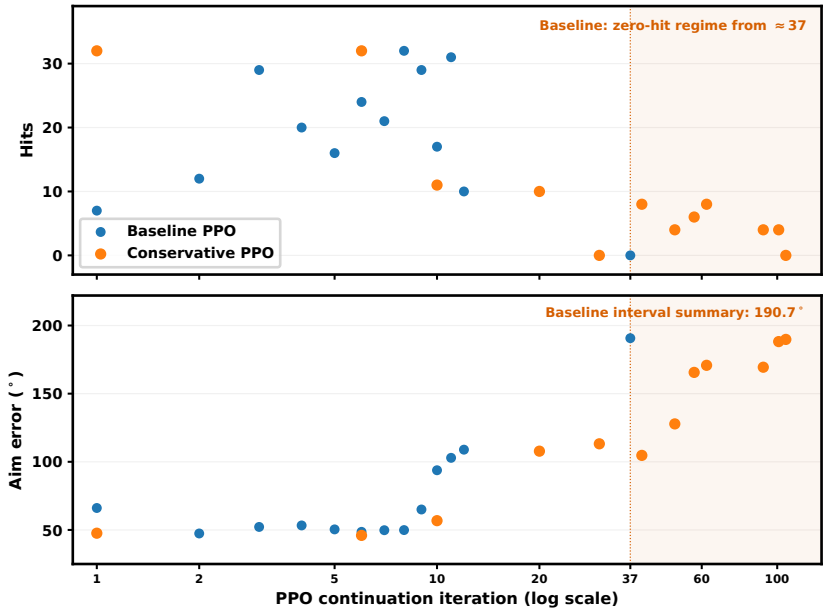


Figure 5: Behavioral retention during two warm-start PPO continuation configurations. The anti-drift configuration combined fewer PPO epochs, stronger reference-policy KL anchoring, and a tighter KL constraint.