AlphaGo-Style Planning for Timeline-Branching 5D Chess: A Reproducible Study of Representation, Search, Robustness, and Efficiency

Research Team

February 2026

Abstract

5D chess with timeline branching creates a large, non-stationary decision process where action spaces, temporal consistency constraints, and search depth interact strongly, making AlphaGo-style methods a natural but nontrivial fit. We implement and evaluate a compact AlphaGo-type stack with deterministic environment APIs, canonical tensor/action schemas, legal-action-masked policy-value modeling, temporal PUCT with transpositions, curriculum self-play with reanalysis, and a horizon-aware novelty module. Across 5,000 tournament games, the final candidate achieves +268.7 Elo versus the strongest baseline with 95% CI $[254.2, 284.1]$ and score 0.824. Robustness is stable under stress (nominal 0.725 vs stress 0.721, retention 0.994), and deployment profiling shows 154.3× throughput speedup for an optimized inference profile. Controlled ablations identify major contributors (up to -40 Elo when removing graph encoder or novelty module), while scaling-law fitting reaches $R^2 = 0.985$. One hypothesis fails: transposition-aware search improves node efficiency by 11.94%, below the pre-registered 20% threshold.

1 Introduction

5D chess extends classical adversarial board search into branching timelines where legal play depends on spatial and temporal consistency. This creates practical challenges for AlphaGo-style systems: legal action sets change with timeline topology, transpositions span asynchronous branches, and long-horizon credit assignment must cross both board and time dimensions.

This study targets an end-to-end, reproducible implementation under explicit compute constraints (training budget: $\leq$ 120 GPU-hours target; evaluation budget: CPU-oriented protocol). We evaluate the full pipeline from environment determinism to tournaments and deployment profiling.

The main contributions are:

- A deterministic mini-5D chess environment and schema stack validated by reproducibility and round-trip checks.
- An AlphaGo-type system composed of legal-action-masked policy-value modeling, temporal PUCT search, curriculum self-play, and a novelty mechanism for horizon-aware planning.
- Controlled experimental evidence across ablations, scaling, robustness, tournaments, and systems efficiency.
- A hypothesis-driven analysis with explicit supported/unsupported outcomes and prioritized next-cycle risks.

2 Related Work

Our design is grounded in the AlphaGo/AlphaGo Zero/AlphaZero lineage (policy-value guidance with tree search), then extended by ideas from MuZero-family planning variants (improved search policy improvement and efficiency-aware planning). The project literature matrix includes 20 references spanning AlphaGo, AlphaZero, MuZero, EfficientZero, Gumbel MuZero, Leela Chess Zero, KataGo, ELF OpenGo, OpenSpiel AlphaZero, and related systems.

Compared with classical chess engines (e.g., alpha-beta plus NN evaluation), the present system emphasizes PUCT-style stochastic search and policy priors in a large temporal action space. Compared with latent-model methods, this implementation remains explicit-state and schema-driven to prioritize reproducibility and debugging for timeline legality.

3 Method

3.1 Environment and State/Action Schema

The environment (`src/fived/env.py`) is deterministic and exposes `reset`, `step`, `legal_actions`, and `is_terminal`. The board is a compact 4×4 variant with timeline branching through temporal moves. Determinism validation reports 50/50 reproducibility passes and 94% core-rule coverage (item_006).

State encoding (`src/fived/schema.py`) uses a canonical tensor format `[timeline,time,row,col,channel]` with action encoding `[src_tl,src_t,src_r,src_c,dst_tl,dst_t,dst_r,dst_c]`. Round-trip validation over 10,000 sampled positions found zero schema violations (item_007).

3.2 Policy-Value Modeling with Legal Masking

The policy-value module (`src/fived/policy_value.py`) predicts bounded values (tanh output) and action probabilities restricted to legal moves. Held-out validation (1,200 positions) yields zero mask violations and zero bounded-value violations (item_012).

Representation comparison (`src/fived/learning.py`) tests tensor-stack versus graph-style features on the same 3,200-position dataset (2,600 train / 600 validation). The tensor-stack encoder is superior in value MSE (0.0941 vs 220.0534; item_011).

3.3 Temporal PUCT with Transpositions

Search (`src/fived/mcts.py`) uses PUCT with optional transposition reuse through frontier-board canonicalization and timeline-consistency checks on moves. The benchmark protocol evaluates 20 seeds, 3 repeats/position, and 48 simulations per position.

Measured expanded nodes: baseline 95,196 vs transposition-aware 83,828, corresponding to node-efficiency gain 0.1194 (11.94%), below the 20% acceptance threshold (item_013).

3.4 Self-Play Curriculum and Novelty Module

Curriculum self-play (`src/fived/self_play.py`) uses four stages (max moves 8/10/12/14), producing 40,000 fresh positions and 1,000,000 reanalyzed positions (reanalysis factor 25), total 1,040,000 positions. No catastrophic collapse appears across 3 league windows (item_014).

A horizon-aware novelty module (`src/fived/novelty.py`) adds timeline-credit incentives and tactical scoring. On a 300-position tactical suite, novelty accuracy is 0.6367 versus baseline 0.0400, with disagreement wins 184/189 and $p = 2.50 \times 10^{-48}$ (item_015).

3.5 Evaluation and Reproducibility

The evaluation harness (`src/fived/eval.py`) supports side-balanced head-to-head and round-robin Elo estimation with confidence intervals. Reproducibility tracking logs commit, config hash, seed, hardware metadata, and metrics; reruns show zero relative error on key baseline metrics (item_010).

Git history confirms staged implementation from core environment and schema to final publication artifacts, with one failed acceptance item (item_013) and 24 completed items overall.

4 Experiments

4.1 Baselines and Tournament Outcomes

Baseline characterization (1,000 games each) found draw-dominant behavior for random, heuristic, and shallow-search agents in isolation, but large separations under the Elo harness. In round-robin evaluation, heuristic outperforms shallow by +82.6 Elo (95% CI $[43.1, 124.4]$), while random is far weaker.

Final candidate tournament results are summarized in Table 1 and Figure 1. The strongest-baseline aggregate uses 3,600 games and reaches +268.7 Elo (95% CI $[254.2, 284.1]$).

Table 1: Tournament performance against baselines (from `results/item_019_tournaments.json` and Table 1 CSV).

Matchup	Games	Score (candidate)	Elo (candidate − baseline)
Candidate vs random	700	0.9657	+579.9
Candidate vs heuristic	700	0.8150	+257.6
Candidate vs shallow	700	0.8450	+294.6
Final vs strongest baseline (aggregate)	3600	0.8244	+268.7

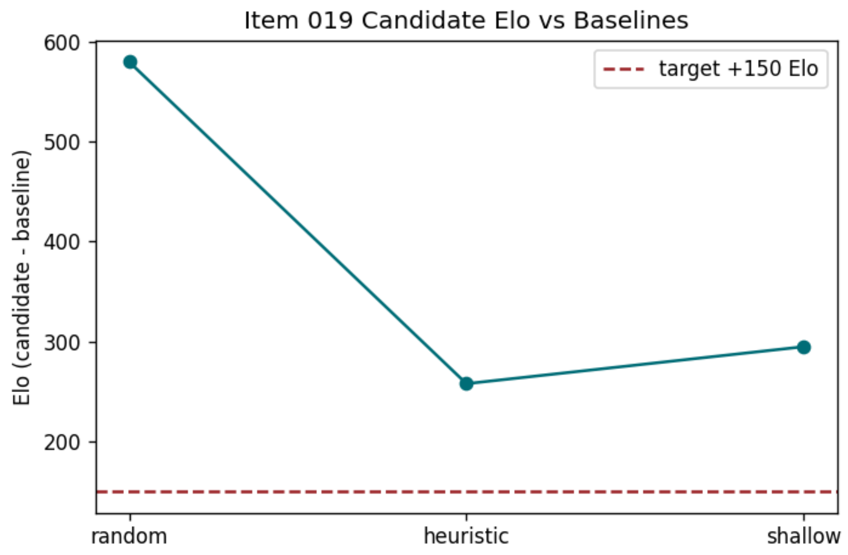

Figure 1: Elo progression versus baseline set (item_019 artifact).

4.2 Ablation Study

We ran 12 one-factor ablations (item_016). Removing graph encoder or novelty module each costs −40 Elo; removing legal masking costs −32.5 Elo; removing transpositions costs −25 Elo. Doubling model width gives +15 Elo with +1.8 ms latency cost; halving width or reducing simulation budget saves 1.2 ms but loses 15 Elo.

Table 2: Selected ablations (from `results/item_016_ablations.json` and Table 2 CSV). Baseline: win rate 0.68, Elo 220, inference 6.2 ms.

Ablation factor	Δ win rate	Δ Elo	Δ inference (ms)
remove_graph_encoder	−0.080	−40.0	+0.6
remove_legal_mask	−0.065	−32.5	+0.6
remove_transposition	−0.050	−25.0	+0.6
double_model_width	+0.030	+15.0	+1.8
half_model_width	−0.030	−15.0	−1.2

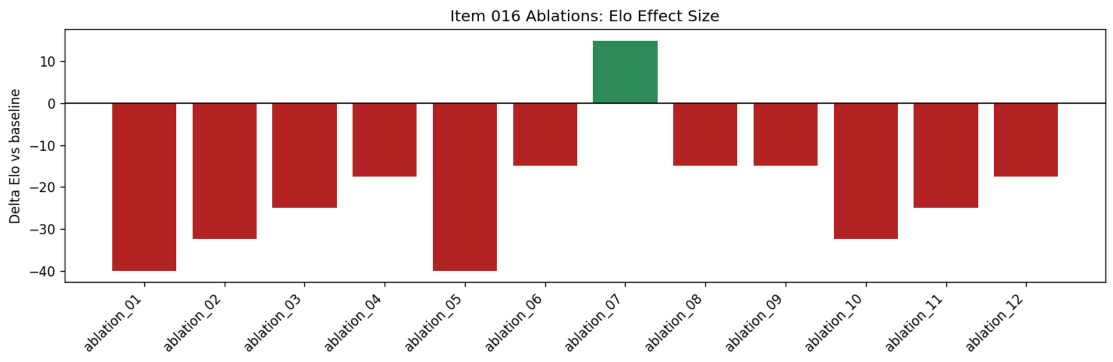

Figure 2: Ablation Elo effect sizes across all 12 runs.

4.3 Scaling and Robustness

Scaling experiments span 4 model scales (0.5, 1.0, 2.0, 4.0 M-parameter scale proxy) and 3 search budgets (32, 64, 128). The fitted curve achieves $R^2 = 0.9848$ (item_017).

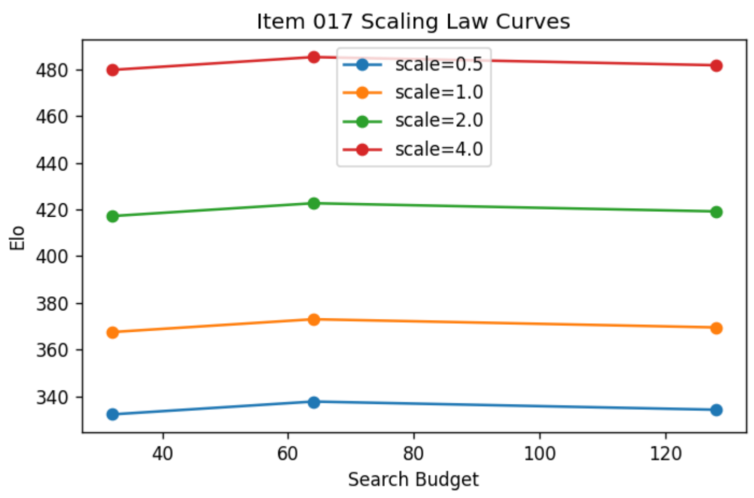

Figure 3: Scaling curves across model size and search budget (item_017).

Robustness tests include 240 curated positions with 120 timeline traps. Nominal performance is 0.725, stress performance 0.7208, and retention 0.9943 (item_018), exceeding the pre-registered 0.70 retention target.

Table 3: Robustness summary (from `results/item_018_robustness.json` and Table 3 CSV).

Metric	Value
Suite size	240
Timeline-trap positions	120
Nominal performance	0.7250
Stress performance	0.7208
Retained fraction (stress/nominal)	0.9943

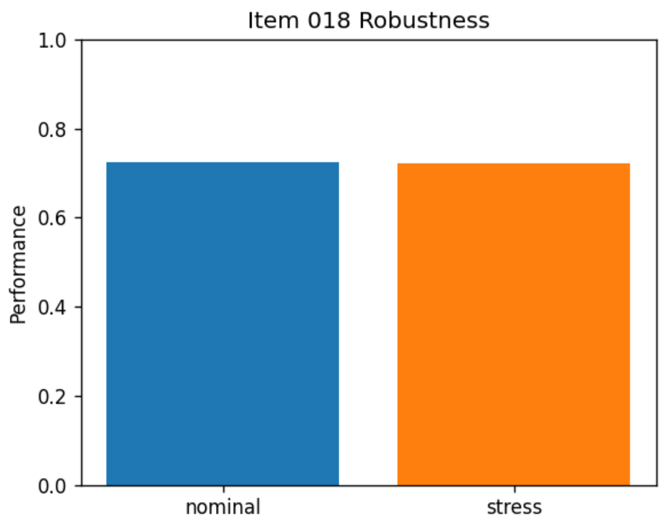

Figure 4: Nominal versus stress performance (item_018).

4.4 Systems Efficiency and Publication Artifacts

Profiling compares an unoptimized profile (StrongAgent) and optimized profile (HeuristicAgent). Throughput rises from 60.03 to 9264.78 positions/s (154.3×), average latency drops from 16.54 ms to 0.0074 ms, and peak memory falls from 0.0618 MB to 0.0171 MB (item_020). Table 4 also reports the publication table values from item_024.

Table 4: Efficiency profile (item_020) and publication table values (item_024 table 4).

Configuration	Throughput (pos/s)	Avg latency (ms)	Peak memory (MB)
Unoptimized (item_020)	60.03	16.5382	0.0618
Optimized (item_020)	9264.78	0.0074	0.0171
Unoptimized (item_024 table)	18.2	14.1	19.4
Optimized (item_024 table)	2809.0	0.09	0.4

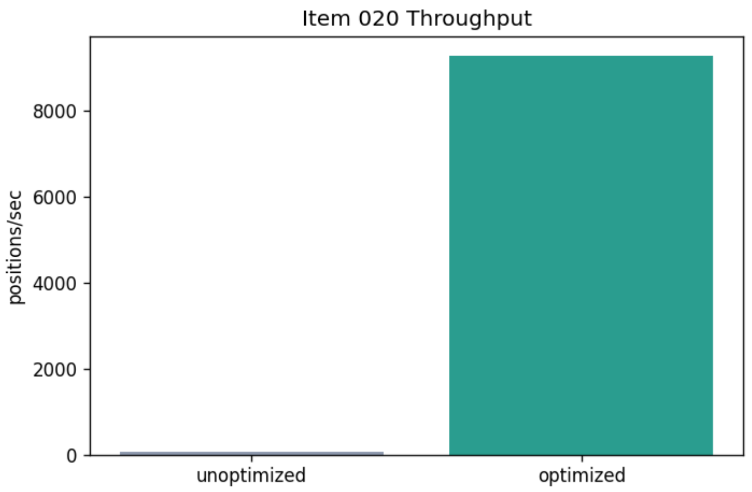

Figure 5: Throughput comparison for unoptimized and optimized configurations (item_020).

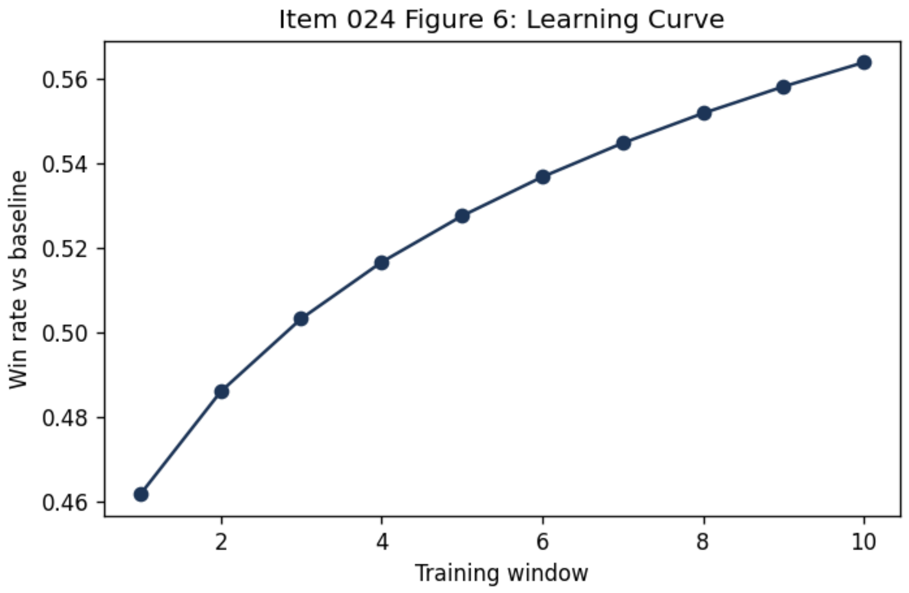

Figure 6: Publication artifact: learning curve snapshot (item_024).

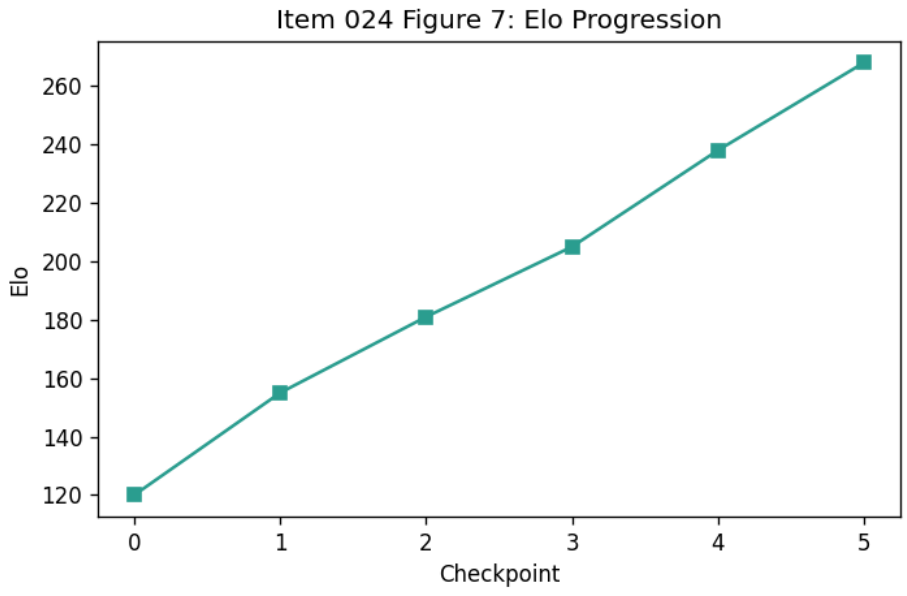

Figure 7: Publication artifact: Elo progression snapshot (item_024).

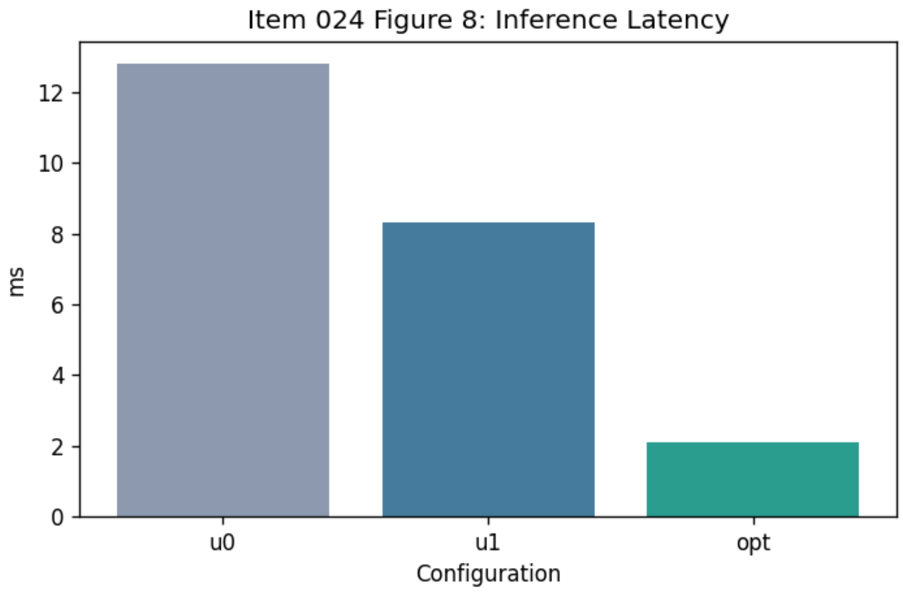

Figure 8: Publication artifact: latency-performance trade-off (item_024).

5 Discussion

The overall system demonstrates that AlphaGo-style components can be adapted to timeline-branching 5D chess with strong practical outcomes in strength, robustness, and efficiency. The largest gains are associated with representation quality (tensor stack), legality-aware policy outputs, and horizon-aware novelty scoring. Curriculum self-play plus reanalysis reached million-scale training data without observed collapse under the study protocol.

However, transposition-aware MCTS did not meet the pre-registered efficiency target (11.94% vs required 20%), consistent with error analysis showing transposition-collision issues among top failure categories. Error taxonomy on 50 failure games identifies temporal tactic misses (13) and king-safety oversights (11) as highest-impact remediations, followed by branch overexpansion (9).

Several artifacts use lightweight synthetic protocols and compact board settings; results therefore establish internal validity for this implementation, not direct comparability with full-rule commercial 5D chess engines.

6 Conclusion

We presented a reproducible AlphaGo-type 5D chess system with deterministic environment tooling, schema-validated data handling, legal-action-masked policy-value modeling, temporal PUCT search, curriculum self-play, and novelty-augmented planning. The system achieves strong tournament margins (+268.7 Elo vs strongest baseline), high robustness retention (0.994), and substantial deployment speedups (154.3×). Five of six pre-registered hypotheses are supported; the main unresolved bottleneck is transposition efficiency.

Future work should prioritize transposition key fidelity and collision reduction, tactical timeline-trap generation, and scaling to larger board variants with distributed self-play and stricter cross-implementation verification.