Research Article *Open Access*

Journal of Pioneering Artificial Intelligence Research

ISSN: 3069-0846

DOI: doi.org/10.63721/26JPAIR0133

Deterministic Elimination Framework for Goal-Oriented Real-Time Autonomous Decision-Making Across Adversarial Domains

Huseyin Murat Cekirge

Department of Mechanical Engineering, The City College of New York (CUNY), New York, USA

Citation: *Huseyin Murat Cekirge (2026) Deterministic Elimination Framework for Goal-Oriented Real-Time Autonomous Decision-Making Across Adversarial Domains. J of Poin Artf Research 2(3), 1-13 WMJ-JPAIR-133*

Abstract

This study presents a deterministic framework for autonomous decision-making in which perception, action selection, and coordination are achieved through progressive elimination of infeasible candidates rather than optimization or search. Instead of constructing solutions through cost evaluation or iterative improvement, the proposed approach operates on a finite candidate space and removes incompatible elements through explicit structural constraints until a single admissible outcome remains. A minimal real-time interception scenario is considered in which an autonomous agent must identify a moving target, execute an engagement sequence, and coordinate retrieval through a cooperative unit under adversarial conditions. In this setting, both perception and action selection are governed by the same elimination principle. The presence of multiple targets and decoys does not require ranking or probabilistic inference; instead, it introduces additional constraints that accelerate candidate reduction. The framework extends naturally to multi-agent systems, where coordination emerges from consistency of constraint-driven eliminations across agents. The same mechanism applies under role inversion, governing both pursuit and evasion. The results suggest a complementary perspective on artificial intelligence in which decision is not constructed through search, but revealed through deterministic reduction over a constrained structure. In contrast to planning approaches such as Goal-Oriented Action Planning (GOAP), which rely on heuristic search and cost evaluation over action sequences, the proposed framework eliminates infeasible candidates directly, without constructing or ranking alternative plans. Although illustrated through an interception scenario, the proposed framework is not limited to pursuit tasks. The same elimination-based mechanism applies to a wide range of domains, including maritime, aerial, and ground operations, as well as search and detection problems.

***Corresponding author**: Huseyin Murat Cekirge, Department of Mechanical Engineering, The City College of New York (CUNY), New York, USA. **E-mail:** hmcekirge@usa.net

Submitted: 24.04.2026 **Accepted**: 28.04.2026 **Published**: 10.05.2026

Introduction

Modern artificial intelligence systems are largely formulated as optimization processes. Learning is typically achieved through gradient-based methods such as stochastic gradient descent, perception is derived from statistical inference over learned representations, and action selection is commonly implemented through search-based planning or reinforcement learning. Planning frameworks such as Goal-Oriented Action Planning (GOAP) rely on heuristic search and cost evaluation over action sequences to determine feasible strategies [1-4].

These approaches have demonstrated strong performance, but they share a common structural assumption: the solution must be approached through iterative improvement within a predefined search or cost landscape. This introduces dependencies on convergence behavior, initialization, and computational resources.

In parallel, alternative perspectives have been explored in pattern recognition, constraint satisfaction, and decision theory. Early work on structural recognition and template matching demonstrated that structured problems can be solved by eliminating incompatible candidates rather than by optimizing over them. Constraint satisfaction frameworks similarly operate by progressively reducing feasible sets through consistency conditions [5-7].

Recent deterministic formulations have further shown that both recognition and decision processes can be expressed as structured elimination over finite candidate sets [8-13]. In these approaches, solutions are not approximated through iterative search, but revealed through compatibility constraints applied directly to bounded representations.

This study extends this perspective and proposes a unified deterministic framework in which perception, action selection, and coordination are governed by the same mechanism: progressive elimination over a finite candidate space. The objective is not to replace optimization-based methods, but to formalize a complementary regime in which decision emerges without search, gradient evaluation, or parameter tuning.

The motivation is particularly strong in real-time autonomous systems operating under adversarial conditions. In such environments, decision latency directly impacts feasibility, and eliminating infeasible options rapidly may be more effective than evaluating or ranking alternatives.

The present framework departs from this formulation by replacing search with deterministic elimination, where plans are not explored but progressively reduced.

While the motivating examples in this study involve adversarial interception, the underlying formulation is domain-independent. Similar structures arise in naval engagement, aerial maneuvering, ground operations, and search tasks, where decision-making can be expressed through progressive elimination rather than search.

Core Principle: Elimination Instead of Optimization

The framework replaces optimization-based reasoning with deterministic elimination. Instead of searching for an optimal solution, the system begins with a finite set of candidates and removes those that violate structural constraints.

Each constraint acts as a filter that reduces the candidate space. The process continues until a single admissible element remains or infeasibility is detected [6].

Unlike optimization, which requires exploration of a solution landscape, elimination directly collapses the candidate space. The solution is not approached through iteration; it is revealed through constraint consistency.

Mathematical Formulation of Deterministic Elimination

System Structure

Let the system operate on four fundamental sets: S (current state), A (set of agents), X (set of candidate objects), and P (set of candidate action sequences).

The decision process consists of two parallel elimination processes:

1. Perception space reduction
2. Action space reduction

Perception Space

Initial observed object set:

$X0 = \{x1, x2, ..., xn\}$

Each object is evaluated through constraint functions:

$Ci(x) \in \{0,1\}$

Feasible object set:

$Xk = \{ x \in X0$ such that $Ci(x) = 1$ for all applied constraints }

Sequential elimination:
X0 → X1 → X2 → ... → X*

Termination conditions:
|X*| = 1 → unique target
|X*| > 1 → more constraints required
|X*| = 0 → inconsistency

Action Space

Let:
P0 = set of candidate action sequences up to bounded depth d

Each sequence p is evaluated through constraints:
Fi(p) ∈ {0,1}

Feasible action set:
Pk = { p ∈ P0 such that Fi(p) = 1 }

Sequential elimination:
P0 → P1 → P2 → ... → P*

Termination:
|P*| = 1 → an executable sequence
|P*| = 0 → reinitialization

The overall structure of the deterministic elimination process across perception and action spaces is illustrated in Figure 1.

C = corobot (cooperative unit)
D = Dog (retrieval agent)

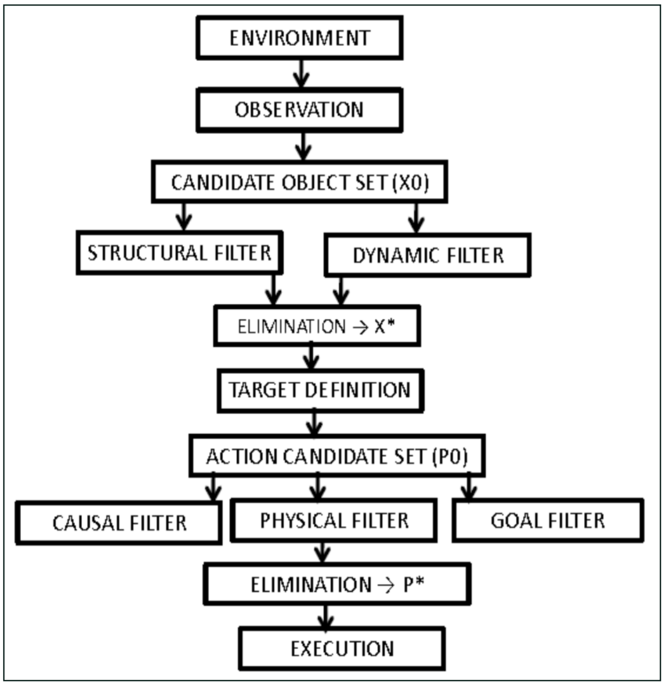

Figure 1: Deterministic elimination architecture showing the progressive reduction of candidate object and action spaces. Perception and action selection are both formulated as constraint-driven elimination processes, leading to a single admissible target and executable sequence without optimization or search.

Coupling of Perception and Action

Define a mapping:
T: X* → P0

The selected object set defines valid action candidates.
Thus:

Object elimination precedes action elimination. Figure 1 highlights the coupling between object elimination and action generation, where the reduced object set directly defines the admissible action space.

Agent Roles and Interaction Structure
Agent Set
Define agents:
A = {B, R, C, D}
where:
B = boss agent (goal setter)
R = primary robot (decision core)

Role Definitions

Each agent operates on its own candidate space:
For agent a:
Xa = candidate object set
Pa = candidate action set
Each agent performs local elimination:
Xa → Xa*
Pa → Pa*

Object Classes

Define object roles:
FOE = adversarial target
DECOY = misleading target
ALLY = cooperative entity
IRRELEVANT = eliminated early

Initial object partition:
X0 = {FOE candidates, DECOY candidates, IRRELEVANT objects}

Adversarial Structure

Let:
F = set of FOE candidates
D_c = set of DECOY candidates

Initial ambiguity:

$X1 = F \cup D_c$

Additional constraints:

1. temporal consistency
2. trajectory coherence
3. task relevance

These produce:

$X^* \subseteq F$

Decoys satisfy some constraints but fail at least one higher-order constraint.

Organized Decoy Behavior

Define decoy structure:
D_c = {d1, d2, ..., dk}
Decoys may satisfy:
Ci(dj) = 1 for i ≤ m
but violate:
Cm+1(dj) = 0
Thus elimination requires layered constraints.

Multi-Agent Coordination

Each agent operates locally:
For agent a:
Xa → Xa*

Global consistency condition:
Intersection over agents:
X_global = the intersection of Xa* over all agents

Coordination condition:
X_global is not an empty set.

Thus coordination emerges from consistency of eliminations.

Boss–Robot Hierarchy

Boss agent B defines goal:
G

Robot R constructs:
P0 based on G

Thus:
B → defines constraints
R → executes elimination
No optimization loop exists.

Cooperative Agent (Corobot and Dog)

Corobot C and Dog D receive:
target = X*

They perform local elimination:
XD → XD*
Action selection:
PD → PD*

Thus:
execution is distributed but consistent.

Role-Dependent Elimination

Define role function:
R(a, x) → role of object x for agent a
Same object may have different roles:

Example:
x = bird
for robot → FOE
for dog → target
for decoy → irrelevant
Thus elimination is role-dependent:
Ci(x | a)

Role Inversion

Define inverse role mapping:
FOE ↔ THREAT

Under inversion:
$X0$ becomes threat candidate set

Same elimination applies:
$X0 \rightarrow X^*$

Action becomes:
escape sequence instead of engagement

Final Decision Structure

Complete system flow:
$X0 \rightarrow$ elimination $\rightarrow X^*$
$X^* \rightarrow$ mapping $\rightarrow P0$
$P0 \rightarrow$ elimination $\rightarrow P^*$

Final output:
single object + single action sequence

Key Structural Property

Decision is defined as the reduction of candidate space under constraints, not selection via optimization

Real-Time Interception Scenario

A real-time interception scenario is used to illustrate the operational behavior of the proposed framework under time-critical and adversarial conditions. The system consists of a primary autonomous agent responsible for target engagement and a cooperative unit responsible for retrieval. The objective is to identify a valid target, execute an engagement sequence, and complete the task within a constrained time window.

Observation and Initial Candidate Set

The agent observes an environment containing multiple objects:
$X0 = \{x1, x2, ..., xn\}$

These objects may include:

- moving targets
- static structures
- irrelevant background elements
- potential decoys

At this stage, no classification or ranking is performed. All objects are treated as candidates.

First-Stage Elimination: Structural Filtering

Initial constraints are applied to reduce the candidate space:

- motion constraint: static objects are removed
- spatial consistency: objects inconsistent with the observation frame are removed
- basic geometric filtering

This produces:

$X1 \subseteq X0$

At this level, the candidate set may still contain multiple elements, including both valid targets and decoys.

Second-Stage Elimination: Dynamic Constraints

Additional constraints refine the candidate set:

- trajectory consistency: valid targets follow continuous motion
- temporal continuity: valid targets persist over time
- behavioral coherence: motion aligns with expected patterns

These constraints eliminate objects that appear valid in isolation but fail across time.

Result:

$X2 \subseteq X1$

Decoy Rejection

In adversarial settings, decoys may satisfy early constraints but fail higher-level conditions.
Let:
$X2 = \{target, decoy1, decoy2, ...\}$

Higher-order constraints are applied:

- inconsistency across time
- deviation from task-relevant behavior
- lack of sustained trajectory

Decoys are eliminated because they fail at least one constraint layer.

Final result:
$X^* = \{target\}$
The correct target is not selected by ranking, but remains as the only candidate consistent across all constraints.

Action Candidate Generation

Given the identified target, the agent constructs a finite set of candidate action sequences:
P0 = {p1, p2, ..., pm}

Each sequence is composed from a bounded action library and represents a possible execution path.

Action Elimination

Action sequences are filtered using constraint layers:
- target validity: sequences without a valid target are removed
- causal ordering: sequences violating required order are removed
- physical feasibility: sequences incompatible with system constraints are removed
- goal completion: incomplete sequences are removed

This produces:

P1 ⊆ P0

Further refinement leads to:
P* = {p*}

Only one sequence remains.

Execution

The selected sequence is executed immediately without scoring, ranking, or search. The system transitions directly from elimination to execution.

Cooperative Retrieval

A cooperative agent receives the target definition:
target = X*

It performs its own local elimination:
XD → XD*

and executes a retrieval sequence:
PD → PD*

Coordination is achieved through consistency of the shared target, not through centralized planning.

Time-Critical Behavior

The entire process operates under time constraints. In adversarial environments:
- delayed decisions reduce feasibility
- an incorrect early commitment leads to failure

The elimination framework addresses both issues:
- early removal of infeasible candidates reduces computation
- absence of iterative search eliminates convergence delays

Thus, decision latency is minimized while maintaining correctness.

Key Property

The scenario demonstrates a fundamental property: Goal-directed behavior emerges not by selecting the best candidate, but by removing all invalid candidates.

Multi-Agent Coordination

The proposed framework extends naturally from single-agent decision-making to multi-agent systems. In this setting, each agent operates on its own local candidate space while maintaining consistency with a shared objective. Coordination does not arise from centralized planning or distributed optimization, but from the alignment of independently reduced candidate sets. The multi-agent coordination structure under adversarial conditions is illustrated in Figure 2.

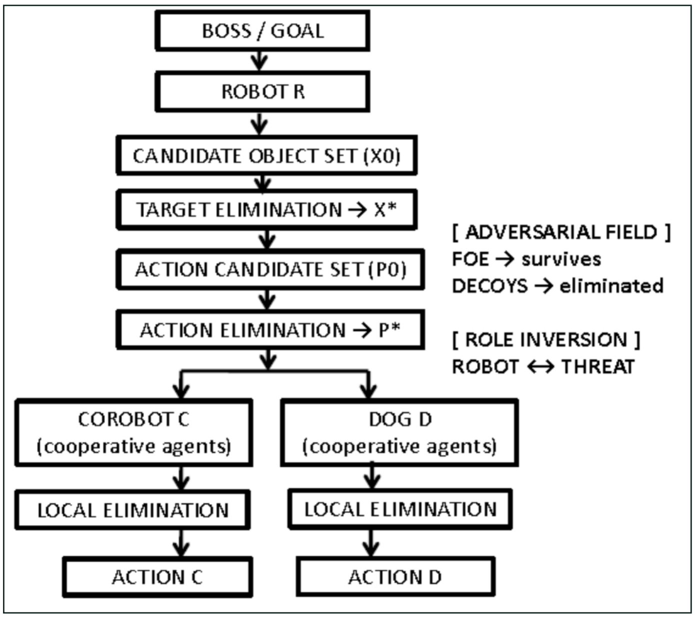

Figure 2: Multi-agent coordination under adversarial conditions. The primary agent performs elimination-based target and action selection, while cooperative agents execute local elimination using the shared target definition. Decoys are removed through constraint consistency, and role inversion preserves the same elimination mechanism for both pursuit and evasion.

Agent-Level Candidate Spaces

Let the system consist of multiple agents:
A = {a1, a2, ..., ak}

Each agent ai maintains:
Xi = local candidate object set
Pi = local candidate action set
Each agent applies elimination independently:
Xi → Xi*
Pi → Pi*

These reductions are performed using local observations and agent-specific constraints.

Shared Target Consistency

A key element of coordination is the propagation of a consistent target definition across agents.

Let:
X* = globally consistent target set

Each agent refines its local candidate set toward compatibility with X*:
Xi* → Xi compatible with X*

Consistency condition:
The intersection of Xi* is non-empty.

If this condition holds, all agents operate on a shared interpretation of the environment.

Coordination Without Centralized Planning

Unlike classical multi-agent systems that rely on global optimization, negotiation, or communication-heavy planning, the proposed framework achieves coordination through constraint consistency.

Each agent:

- performs local elimination
- receives or infers shared constraints
- updates its candidate space

No agent computes a global plan. Instead, global behavior emerges as a consequence of aligned eliminations.

Role Differentiation

Agents may have different roles within the system:

- primary agent: target identification and engagement
- cooperative agent: support or retrieval
- auxiliary agents: monitoring or filtering

Each role defines a distinct constraint set.

Thus, elimination is role-dependent:
the same object may be:

- a target for one agent
- irrelevant for another
- or a constraint source for a third

Cooperative Execution

Once a consistent target is established, agents proceed with action elimination:
Pi → Pi*

Execution is distributed:

- the primary agent executes engagement
- the cooperative agent executes retrieval

Coordination is maintained because both action sequences are derived from the same reduced target set.

Robustness Under Adversarial Conditions

In adversarial environments, inconsistencies between agents may arise due to decoys, noise, or partial observations.

The elimination framework resolves this through:

- additional constraint layers
- cross-agent consistency checks
- rejection of incompatible interpretations

Agents that maintain inconsistent candidate sets are naturally filtered out at the system level.

Temporal Coordination

Coordination is not only spatial but also temporal.

Agents must satisfy:

- sequence alignment
- timing constraints
- causal ordering across actions

These constraints are applied within each agent's elimination process, ensuring synchronized behavior without explicit scheduling optimization.

Emergent Coordination Property

The key property of the system is:
Global coordination emerges from the intersection of locally reduced candidate spaces.

No explicit coordination policy is required.

Summary

The multi-agent extension preserves the core principle of the framework:
Each agent eliminates infeasible candidates locally, and coordination arises from consistency of constraints across agents.

Adversarial Conditions and Decoys

Initial Ambiguity

Following initial perception filtering, the candidate set may remain unresolved:
$X1 = \{x1, x2, x3, ...\}$

where:
- one element corresponds to the true target
- remaining elements correspond to decoys or irrelevant entities

At this stage, all candidates may satisfy basic structural constraints such as motion, size, or visibility. No ranking or probabilistic scoring is performed.

Layered Constraint Structure

Resolution is achieved through successive application of constraint layers:
First layer:
- motion detection
- basic spatial consistency

Second layer:
- temporal continuity
- persistence over time

Third layer:
- trajectory coherence
- consistency of direction and velocity

Fourth layer:
- task relevance
- alignment with mission-specific conditions

Each layer removes candidates that fail at least one constraint.

Decoy Behavior

Decoys are characterized by partial compliance:
For a decoy d:
$Ci(d) = 1$ for early constraints
$Cj(d) = 0$ for at least one higher-level constraint

This means:
- decoys may survive early filtering
- but are eliminated in later stages

Thus, decoys are not incorrect initially; they become inconsistent as more constraints are applied.

Progressive Elimination

The candidate set evolves as:
$X1 \rightarrow X2 \rightarrow X3 \rightarrow ... \rightarrow X^*$

At each stage:
$Xi+1$ is a subset of Xi

until:
$X^* = \{\text{true target}\}$

The correct target is not selected by comparison, but remains as the only candidate consistent across all constraints.

Time-Dependent Filtering

Adversarial resolution is inherently time-dependent.
Constraints such as:
- continuity
- persistence
- motion stability

require observation over time.

Thus, elimination is not purely instantaneous but unfolds over a short temporal window.
Candidates that fail to maintain consistency across time are removed.

Organized Decoys

In more complex environments, decoys may appear in structured or coordinated forms.
Let:
$D_c = \{d1, d2, ..., dk\}$

These decoys may:
- mimic motion patterns

- maintain temporary consistency
- appear coordinated

However, such coordination introduces higher-order inconsistencies:

- divergence over longer trajectories
- mismatch with task-relevant behavior
- lack of sustained coherence

These inconsistencies are captured by deeper constraint layers, leading to elimination.

Adversarial Effect on Complexity

Adversarial environments increase the size of the candidate set:
$|X1| >> 1$

However, they also introduce additional constraints:

- behavioral constraints
- temporal constraints
- interaction constraints

These constraints accelerate elimination:
$|Xi+1| << |Xi|$

Thus, increased adversarial complexity does not necessarily increase computational burden; it may reduce it by enabling stronger filtering.

No Ranking or Optimization

A key distinction of the framework is the absence of ranking.

The system does not:

- assign scores
- compare likelihoods
- evaluate probabilities

Instead:

- candidates either satisfy constraints or not

Decoys are eliminated because they fail constraint consistency, not because they are less optimal.

Robustness Property

The framework exhibits robustness under adversarial conditions:

- false positives are removed through layered constraints
- temporary ambiguities are resolved over time
- coordinated decoys are rejected through higher-order inconsistencies

This leads to a stable final outcome:
$X^* = \{\text{true target}\}$

Summary

Adversarial environments do not invalidate the framework; they strengthen it. While they increase the number of initial candidates, they also introduce additional inconsistencies that enable more effective elimination. The presence of decoys does not require ranking or probabilistic inference. Instead, it reinforces the central principle:

Candidates are removed through accumulated constraint violations until only one consistent element remains. Additional constraints such as temporal continuity, trajectory coherence, and task relevance are applied.

These constraints progressively eliminate inconsistent candidates. Decoys are not ranked or scored; they are removed because they fail to satisfy accumulated constraints. Adversarial complexity increases the number of candidates but also introduces inconsistencies that accelerate elimination.

Role Inversion

The proposed framework is symmetric under role inversion. The same elimination mechanism governs both pursuit and evasion, with the only difference being the definition of the objective and the associated constraints.

Role Transformation

In the standard configuration, the system operates in pursuit mode:

- candidate set: potential targets
- objective: identify and engage target

Under role inversion, the system operates in evasion mode:

- candidate set: potential threats
- objective: identify and avoid or escape threat

Formally, the candidate space remains the same, but its interpretation changes:
$X0 = \{x1, x2, ..., xn\}$

In pursuit:

- x represents possible targets

In evasion:

- x represents possible threats

Symmetry of Elimination

The elimination process remains unchanged:
$X0 \rightarrow X1 \rightarrow X2 \rightarrow \ldots \rightarrow X^*$

In both cases:

- the remaining element represents the most consistent interpretation

The difference lies only in the action mapping:

- pursuit → engagement sequence
- evasion → escape sequence

Thus, decision is invariant under role inversion, while action semantics change.

Threat Identification

In evasion scenarios, the system identifies threats rather than targets.

Initial ambiguity may include:
X1 = {neutral object, potential threat, decoy threat}

Constraints applied:

- approach behavior
- trajectory toward the agent
- velocity and intent consistency
- persistence over time

These eliminate non-threatening candidates.

Final result:
X* = {threat}

Escape Action Selection

Once a threat is identified, the action space is constructed:
P0 = {possible escape sequences}

Constraints eliminate:

- paths leading toward threat
- infeasible or blocked movements
- sequences that do not increase safety

Result:
P* = {escape sequence}
Execution proceeds immediately.

Mutual Adversarial Systems

In many real-world scenarios, multiple agents operate under opposing roles, each applying elimination simultaneously.

Examples include:

- two aircraft in aerial engagement
- two naval units in surface conflict
- two opposing agents in confined environments such as trench conditions

In these cases:
Each agent performs:
$Xi \rightarrow Xi^*$
$Pi \rightarrow Pi^*$

independently and in parallel.

Coupled Elimination Dynamics

Let agents A and B operate in opposition.

For agent A:

- candidate set includes possible targets (agent B)

For agent B:

- candidate set includes possible threats (agent A)

Thus:
XA includes B as FOE
XB includes A as FOE

Both agents perform elimination based on:

- observed motion
- trajectory prediction
- consistency over time

Their actions influence each other's candidate sets.

Dynamic Interaction

The interaction is inherently dynamic:

- actions of one agent modify the observation space of the other
- elimination must be continuously updated
- both perception and action operate in a feedback loop

Thus:
X_{t+1} depends on actions taken at time t
This produces a coupled elimination system.

Absence of Optimization

Even in adversarial dual-agent scenarios, no optimization is required.

Each agent:
- does not search for best strategy
- does not evaluate opponent utility
- does not solve a game-theoretic optimization

Instead:
- each agent removes infeasible interpretations
- selects actions consistent with remaining candidates

Interaction emerges from constraint consistency, not strategic optimization.

Generality

The same structure applies across domains:
- aerial systems (mutual interception and evasion)
- maritime systems (surface engagement)
- ground systems (close-range constrained interaction)

The framework is independent of domain specifics. Only the constraint set changes.

Summary

Role inversion does not introduce a new mechanism. It redefines the meaning of the candidate set and the objective.

The same elimination process governs:
- target selection in pursuit
- threat identification in evasion
- action generation in both cases

Thus, the framework is symmetric and general.

Discussion

This distinction becomes particularly significant in structured and time-critical environments. In such settings, the feasibility of an action depends not only on correctness but also on decision latency. Optimization-based methods may require iterative refinement, evaluation of alternatives, or convergence processes, all of which introduce delays. In contrast, the elimination-based framework removes infeasible candidates early, allowing rapid convergence to a single admissible solution without iterative search.

Another key implication is interpretability. Each decision can be traced back to explicit constraint violations that eliminated alternative candidates. This contrasts with many optimization-based systems, where decisions emerge from high-dimensional parameter interactions that are difficult to interpret directly.

The framework also offers robustness in adversarial environments. The presence of decoys or misleading signals does not require probabilistic inference or ranking. Instead, adversarial complexity introduces additional inconsistencies, which are captured by higher-order constraints. As a result, incorrect candidates are progressively eliminated, and the correct solution remains as the only consistent outcome.

However, the approach has clear limitations. Its effectiveness depends on the availability of well-defined and sufficiently discriminative constraints. If constraints are weak, incomplete, or poorly structured, the elimination process may fail to converge to a unique solution. Similarly, the framework assumes a finite or bounded candidate space. In environments with continuous or unbounded representations, direct application may become impractical without additional discretization or structure.

Scalability is another consideration. While elimination reduces candidate sets, the initial construction of the candidate space must remain tractable. The framework is therefore most effective in systems where strong constraints significantly reduce the search space early in the process.

It is important to emphasize that the proposed approach is not intended to replace optimization-based methods. Instead, it defines a complementary regime. Optimization is effective in high-dimensional, weakly structured, and data-driven problems. In contrast, elimination is effective in structured, constrained, and time-critical systems where feasibility dominates optimality.

From a broader perspective, the framework suggests a reinterpretation of artificial intelligence processes. Many systems described as reasoning or decision-making rely on optimization dynamics. The present approach demonstrates that, under appropriate conditions, intelligent behavior can emerge without search, learning, or gradient-based adjustment. Decision arises from constraint consistency rather than from optimization.

The approach is effective in structured and time-critical environments. Its limitations include dependence on well-defined constraints and finite candidate representations.

Conclusion

The framework shows that, in structured environments, decision-making does not require iterative search, gradient-based adjustment, or probabilistic ranking. Instead, a finite candidate space can be progressively reduced through explicit constraints until a single admissible outcome remains. This mechanism applies consistently across perception, action, and multi-agent coordination, providing a coherent alternative to conventional approaches.

The analysis further demonstrates that adversarial conditions, including the presence of decoys, do not weaken the framework but instead reinforce it by introducing additional constraints that accelerate elimination. Similarly, the principle remains valid under role inversion, supporting both pursuit and evasion within the same structural formulation.

From a systems perspective, the framework offers advantages in interpretability, determinism, and response time. Decisions can be directly explained through constraint violations, and the absence of iterative optimization reduces latency in time-critical applications. These properties make the approach particularly suitable for structured, constrained, and real-time environments.

At the same time, the framework is not universally applicable. Its effectiveness depends on the availability of well-defined constraints and bounded candidate spaces. In domains characterized by weak structure or high-dimensional uncertainty, optimization-based methods may remain more appropriate.

The results suggest that elimination-based reasoning constitutes a complementary paradigm within artificial intelligence. In this view, intelligent behavior is not obtained by selecting the best option among many, but by systematically removing all options that cannot be sustained under the constraints of the system.

Although the framework is demonstrated using an interception scenario, its applicability extends beyond pursuit. The same elimination-based structure can be employed in maritime, aerial, and ground systems, as well as in non-adversarial search and identification tasks.

This study introduces a deterministic elimination framework for autonomous decision-making. By unifying perception, action selection, and coordination under a single principle, it demonstrates that intelligent behavior can emerge from constraint-driven reduction rather than optimization.

Unlike planning approaches such as Goal-Oriented Action Planning (GOAP), which construct solutions through search and evaluation, the proposed framework derives decisions through direct constraint-based elimination.

Abbreviations

AI	Artificial Intelligence
B	Boss agent (goal-setting agent)
C	Corobot (cooperative agent)
Ci	Constraint function for objects
D	Dog (retrieval agent)
DECOY	Misleading or false candidate
Fi	Constraint function for action sequences
FOE	Adversarial target
GOAP	Goal-Oriented Action Planning
P*	Final executable action sequence
P0	Initial candidate action set
R	Primary robot (decision agent)
X*	Final reduced object set (target)
X0	Initial candidate object set

Author Contributions

Huseyin Murat Cekirge is the sole author. The author read and approved the final manuscript.

Conflicts of Interest

The author declares no conflicts of interest.

References

1. Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press https://www.deeplearningbook.org/.
2. Sutton RS, Barto AG (2018) Reinforcement Learning: An Introduction https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf.
3. Bertsekas DP (2016) Nonlinear Programming. Athena Scientific https://web.mit.edu/15.053/www/AMP-Chapter-13.pdf.
4. Orkin J (2006) Symbolic Representation of Game World State: Toward Real-Time Planning in Games. AIIDE https://aaai.org/papers/ws04-04-006-symbolic-representation-of-game-world-state-toward-real-time-planning-in-games/.
5. Marr D (1982) Vision. MIT Press https://www.mdpi.com/journal/vision.
6. Tsang E (1993) Foundations of Constraint Satisfaction. Academic Press https://cse.unl.edu/~choueiry/Documents/TsangTextbook/Tsang-Fcs1993-Toc.pdf.
7. Duda R., Hart P, Stork D (2001) Pattern Classification. Wiley https://tjzhifei.github.io/links/PC2.pdf.
8. Lowe D (2004) Distinctive Image Features from Scale-Invariant Keypoints. IJCV https://link.springer.com/article/10.1023/B:VISI.0000029664.99615.94.
9. Simon H (1982) Models of Bounded Rationality. MIT Press https://direct.mit.edu/books/monograph/4286/Models-of-Bounded-Rationality-Volume 3: Empirically.
10. Pearl J (1984) Heuristics: Intelligent Search Strategies. Addison-Wesley https://www.researchgate.net/publication/220605641_Heuristics_Intelligent_Search_Strategies_for_Computer_Problem_Solving_-_Book_Review.
11. Wooldridge M (2009) An Introduction to MultiAgent Systems. Wiley https://www.wiley.com/enus/An Intruduction+toMultiAgent+Systems%2C+2nd+Edi.
12. Cekirge HM (2026) Structural Filtering for Object Recognition: A Deterministic Alternative to Optimization-Based Training. Journal of Pioneering Artificial Intelligence Research 2: 1-15.
13. Cekirge HM (2026) Deterministic Autonomous Structural Elimination Framework. Journal of Pioneering Artificial Intelligence Research 2: 1-15.