Research Article *Open Access*

Journal of Pioneering Artificial Intelligence Research

ISSN: 3069-0846

doi.org/10.63721/26JPAIR0154

A Modular Architecture for Autonomous, Independent and Innovative Robots

Huseyin Murat Cekirge

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

Citation: *Huseyin Murat Cekirge (2026) A Modular Architecture for Autonomous, Independent and Innovative Robots. J.of Pion Artf Int Research 2(5), 1-23. WMJ-JPAIR-154*

Abstract

Robotic autonomy, independent goal generation, and innovation are often treated as equivalent capabilities, although they represent distinct operational mechanisms. An autonomous robot executes or selects actions for an externally assigned goal. A fully independent robot can generate and select a goal without a present command or assigned mission. An innovative robot constructs a new functional strategy when its active library contains no adequate complete solution. This paper proposes a bounded modular architecture that formally separates these three capabilities.

The architecture operates within a four-dimensional context comprising robot state and location, external or event state and location, robot-relative time, and event-relative time. In command-free operation, internal and external segments generate candidate goals, which are evaluated through multidimensional supporting, opposing, and bounded stochastic contributions. Feasibility, competence, safety, installed facilities, and governing authority determine the admissible candidate set. In mission- or cue-driven operation, valid external instructions govern goal selection, and internal preference segments do not alter the assigned objective. When an admissible goal cannot be satisfied by any complete action in the active Action Library, the SPARK mechanism searches for compatible action segments, assembles a candidate strategy, validates it against functional and safety constraints, and admits it to the library only after successful evaluation.

Tennis, cue-driven dance, mission-governed execution, scientific problem solving, tactical coaching, and choreography are used to distinguish goal selection, prescribed execution, and innovative strategy construction.

Innovation is defined operationally as the validated production of a useful strategy S_n such that $S_n \notin L_0$, where L_0 denotes the robot's active pre-existing library. The proposed framework therefore extends robotic autonomy while preserving bounded competence, traceability, safety, and human-defined authority.

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

Received: 07.09.2026 **Accepted**: 11.09.2026 **Published**: 20.09.2026

Keywords: Autonomous Robots, Fully Independent Robots, Command-Free Goal Generation, Robotic Innovation, Bounded Modular Architecture, Internal Arbitration, Strategy Generation, Action Library

Abbreviations and Symbols

Ω	Four-dimensional decision context, (X_r, X_e, T_r, T_e)
X_r	Robot spatial state/location
X_e	External/event spatial state/location
T_r	Robot-relative time
T_e	Event-relative time
L_0	Active pre-event Action Library
L_1	Updated Action Library after validation
S_n	Newly constructed candidate strategy
$G*$	Selected goal
$G^A(\Omega)$	Set of admissible candidate goals
$M_k(\Omega)$	Contextual matching value
$J_k(\Omega)$	Integrated decision value
$D_k(\Omega)$	Supporting/pro-action internal segment
$A_k(\Omega)$	Opposing/inhibitory internal segment
$N_k(\Omega)$	Non-decisive neutral contextual segment
$R_k(\Omega)$	Bounded stochastic contribution
SPARK	Trigger for bounded constructive search when no adequate complete solution exists
Mod	Functional robot module
Mod(r)	Set of modules installed and authorized for robot r
Mod^{INN}	Innovative-construction module set
ε	Maximum admissible compatibility tolerance
V	Validation operator
OUT	Module/system output

1. Introduction

A robot cannot innovate. This assertion is often treated as self-evident: a robot may calculate, retrieve information, optimize a trajectory, select an action, or execute a predefined task, but innovation is commonly reserved for human beings. Yet this conclusion depends less on an established impossibility than on an unclear definition of innovation. If innovation means creating something from nothing, neither a human nor a robot innovates. Human innovation normally begins with inherited knowledge, learned rules, accumulated experience, available components, environmental conditions, and recognized constraints. The genuinely new element is frequently not the material from which a solution is formed, but the complete structure into which those materials are organized.

Four familiar human activities make this distinction visible. A football coach does not create the players, ball, field, or rules of the game. The coach creates a tactical organization that did not previously exist as a complete strategy. A choreographer does not create the human body or every individual movement. The choreographer creates a new spatial, temporal, and expressive composition from available movements and performers. A mathematician does not create axioms, definitions, or rules of logic, but may construct a proof that did not previously exist. A scientist does not create the observed phenomena, but may establish a previously unrecognized relationship and formulate a new hypothesis, model, or theory. Tactical, compositional, deductive, and explanatory innovation therefore operate on different materials while sharing one structural characteristic: existing elements are organized into a useful whole that was not previously available as a complete entity.

This interpretation is consistent with established accounts of creativity. Boden distinguishes combinational, exploratory, and transformational creativity and characterizes creative products in terms of novelty, surprise, and value [1]. These categories weaken the assumption that the use of existing knowledge disqualifies an outcome from being creative. Combination begins with available elements, exploration operates within an existing conceptual space, and transformation modifies that space. The relevant question is therefore not whether the constituent elements existed beforehand, but whether the completed organization and its operational value existed before the creative process began.

Research on artificial creative problem solving has addressed important parts of this question. Sarathy and Scheutz proposed the MacGyver Test as a framework for evaluating machine resourcefulness in open-world problems and subsequently described

MacGyver problems as challenges that require an agent to generate and execute strategies for apparently unsolvable situations [2, 3]. Nair et al. demonstrated feature-guided search for constructing replacement tools when an expected tool was unavailable [4]. Fitzgerald et al. examined how a robot can learn task constraints and identify atypical tools that satisfy them [5]. In these studies, the agent is not limited to repeating the nominal action originally associated with a task. It must reorganize available objects, properties, actions, and constraints into a workable alternative.

Related research addresses the expansion and organization of robotic competence. Continual learning seeks to enable robots to acquire capabilities from sequential experience without losing previously learned knowledge [6]. Educational robotics and robot creativity also provide settings for synthesis, open-ended construction, and creative problem solving [7]. Cognitive-architecture research has examined how perception, attention, action selection, memory, learning, reasoning, and metareasoning can be integrated within a common computational system [8]. Such work is relevant to the present architecture because innovation cannot be reduced to a single generative operation; it requires coordinated recognition, retrieval, construction, evaluation, validation, and retention.

However, learning an additional stored response and constructing a previously unavailable solution are not identical operations. A system may enlarge its memory while remaining restricted to retrieval and selection. Conversely, a system may use a fixed collection of elements to construct an organization that was not stored as a complete solution. Library dependence alone therefore cannot determine whether innovation has occurred. Human innovators are also dependent on libraries of knowledge, experience, methods, and cultural inheritance.

A further distinction is required between autonomous execution, independent goal generation, and innovation. An autonomous robot may select and execute actions in pursuit of an externally assigned goal. Goal-driven autonomy extends conventional planning by allowing an agent to identify discrepancies during execution, formulate goals, and redirect its activity in response to changing circumstances [9]. Research on intrinsic motivation likewise shows how artificial agents can be driven toward exploration, learning progress, or internally valued situations rather than relying exclusively on immediate external rewards [10]. These approaches provide important foundations for self-directed behavior, but selecting a new goal and constructing a new solution are still different operations.

In the terminology adopted in this paper, a fully independent robot can generate candidate goals and choose among them without a present command or assigned mission. An innovative robot performs a different operation: when no complete admissible solution exists for the selected or assigned goal, it constructs and validates a new functional strategy. A robot may therefore be autonomous without being fully independent, fully independent without producing an innovation, or innovative while working on a goal assigned by an external authority. These capabilities require different modules and should not be treated as interchangeable.

The distinction can be expressed structurally. Let L_0 denote the active library of complete alternatives available to the robot before a decision or innovation event. Ordinary autonomous selection chooses an executable alternative $a*$ already contained in L_0: $a* \in L_0$.

By contrast, let S_n denote a newly constructed and validated structure. Operational innovation requires: $S_n \notin L_0$.

The constituent segments of S_n may already exist in the robot's accessible knowledge system. However, S_n itself must not exist in L_0 as a complete tactical, compositional, deductive, explanatory, or otherwise executable solution. After successful evaluation and validation, S_n is admitted to the expanded library: $L_1 = L_0 \cup \{S_n\}$.

The first validated construction of S_n constitutes innovation relative to the active library L_0. Its subsequent retrieval, repetition, or physical reproduction does not constitute another innovation. Thus, $S_n \notin L_0$ is a necessary but not sufficient condition: it establishes library-relative novelty, whereas usefulness and admissibility must be established separately.

This distinction also separates innovation from random variation. A different output is not necessarily an innovative output. The new structure must be relevant to a goal, compatible with applicable constraints, internally coherent, and sufficiently useful to justify admission to the library. A football tactic that is merely unusual, a choreography that cannot be performed, a proof containing an invalid inference, or a scientific model that provides no testable or explanatory advantage does not satisfy these requirements. Innovation changes the set of available admissible alternatives, but validation determines whether that change is retained.

Independence and innovation are also bounded by architecture and authority. A robot can operate only through the perception, decision, action, knowledge, and validation modules installed and activated for its designated role. Not every autonomous robot requires command-free goal generation, internal volitional arbitration, stochastic exploration, or innovative construction. For example, a cue-driven ballet robot may execute a prescribed figure when it receives the relevant signal, while a mission-governed robot may execute an authorized objective without allowing internal preference segments to redefine that objective. In safety-critical applications, governing constraints must restrict the admissible action set before execution or exploratory selection. This architectural principle is consistent with work on explicit ethical and operational control mechanisms for autonomous robots [11].

Accordingly, the proposed framework does not describe unrestricted robotic freedom. It describes bounded modular independence: each robot operates within its installed competence, validated Action Library, safety requirements, contextual constraints, and governing authority. Different robot types may contain different subsets of the proposed modules. A scientist robot may require related-problem retrieval, hypothesis construction, and evidence evaluation, whereas a cue-driven performer may require only recognition, mapping, and execution modules. Full independence is therefore domain-relative and module-relative rather than universal.

The present study develops a unified structural analysis of autonomous execution, command-free goal generation, and robotic innovation. It first distinguishes selection, self-initiation, and construction. It then introduces a four-dimensional decision context, multidimensional opposing and non-opposing internal segments, admissibility conditions, and a bounded modular architecture. Finally, it defines the SPARK mechanism, through which the absence of an adequate complete solution activates segment search, compatible assembly, consequence evaluation, validation, and controlled library expansion.

The principal contribution is not the claim that every autonomous robot must innovate or independently generate goals. Rather, it is a formal separation of three mechanisms that are frequently conflated: selecting an existing action for an assigned goal, selecting a goal without a present command, and constructing a validated strategy that was not previously stored as a complete solution. If the construction of a useful new organization is accepted as innovation when performed by a human coach, choreographer, mathematician, or scientist, its impossibility for a robot cannot be established merely by observing that the robot begins with an existing library. The decisive issue is whether the system can move beyond retrieval and selection to construct, evaluate, validate, and retain S_n.

The present study proposes a conceptual and operational architecture rather than reporting a completed physical implementation.

2. Analysis of the Innovation Mechanism

2.1 Selection, Self-Initiation, and Construction

The proposed architecture separates three operations that are frequently treated as equivalent: selection, self-initiation, and construction.

Selection occurs when a robot chooses an executable action already stored in its active library. Self-initiation occurs when a robot generates and selects a goal without a present external command or assigned mission. Construction occurs when no adequate complete solution exists for the selected or assigned goal and the robot assembles a new candidate strategy from available functional segments.

These operations can occur independently. An autonomous robot may select an action for an externally assigned goal without generating the goal itself. A fully independent robot may generate a goal but satisfy it

by selecting an already stored action. An innovative robot may construct a new strategy for either a self-generated or externally assigned goal. Self-initiation is therefore not sufficient for innovation, and innovation does not require command-free goal generation.

Let L_o denote the active library of complete executable solutions available before the decision event, and let $a*$ denote the selected action. Ordinary autonomous selection satisfies: $a* \in L_o$.

If the robot generates the current goal $G*$ without a present command or assigned mission, then

$G* \in G^A(\Omega)$

where $G^A(\Omega)$ is the set of admissible candidate goals in context Ω. This constitutes self-initiated goal selection, but not necessarily innovation.

Let S_n denote a newly constructed candidate strategy. Operational novelty requires: $S_n \notin L_o$.

However, non-membership alone is not sufficient. Random, incoherent, unsafe, or ineffective outputs may also satisfy $S_n \notin L_o$. To qualify as an innovation, S_n must additionally satisfy the current goal, applicable constraints, functional-coherence requirements, and a domain-specific validation criterion.

Innovation is therefore defined as a validated expansion of the admissible solution set, rather than the mere production of a different output.

SPARK does not itself construct S_n. It identifies the need for constructive search when the active library contains no adequate complete solution or when the currently active solution has become inadequate. The innovative mechanism begins after this trigger and performs bounded segment search, assembly, evaluation, and validation.

2.2 Four-Dimensional Context and Recognition Before Action

Every recognition, goal, action, and innovation event is evaluated within a four-dimensional context:

$\Omega = (X_r, X_e, T_r, T_e)$

where:

- X_r denotes the robot's current internal and physical state
- X_e denotes the state of the relevant external entity or environment
- T_r denotes the robot-relative temporal state, including its active history and current operational phase
- T_e denotes the temporal state and evolution of the external entity or event

The state Ω is dynamic rather than static or stationary. The same external configuration may produce different admissible actions when the robot's internal state, prior activity, available time, event development, or applicable constraints change.

For example, the availability of a tennis court, a tennis partner, and the required equipment establishes an external opportunity. It does not establish that the robot should immediately play tennis. The final decision also depends on the robot's current internal segments, timing, commitments, safety conditions, and governing constraints.

Recognition must precede action selection or construction. The robot begins with a minimal information set I_1 and progressively acquires additional information until the required recognition confidence is reached: $I_1 \subset I_2 \subset \cdots \subset I_k$,

$\rho(I_k) \geq \rho_{req}$

where $\rho(I_k)$ denotes the recognition confidence obtained from information set I_k, and ρ_{req} is the minimum confidence required to activate the relevant bounded module.

This progressive rule avoids both premature identification and unnecessary sensing. A single visible feature may not be sufficient to establish the identity, function, or relevance of an entity. Additional components, structural relations, motion patterns, temporal continuity, sound, interaction history, or other evidence may be required when ambiguity remains.

Recognition therefore performs two functions. First, it determines whether the perceived entity or event has been identified with sufficient confidence. Second,

it determines which bounded module, authority structure, constraint set, and segment library may legitimately be activated.

Perception alone does not constitute a command. Recognition of a tennis court indicates that a playing opportunity exists; it does not require the robot to play. Similarly, recognition of coffee, a dance cue, a tactical failure, or a scientific problem has different operational meanings depending on the active module and governing context.

2.3 SPARK as a Bounded Search Trigger

SPARK marks the transition from ordinary execution or unresolved goal processing to constructive search. It can be activated when:

1. No complete solution in L_o adequately satisfies the active goal
2. An active strategy ceases to satisfy its expected performance conditions
3. Observations diverge significantly from predicted outcomes
4. A new constraint invalidates previously admissible solutions
5. An unresolved goal remains admissible but cannot be executed by the available complete actions

Let $\Gamma(\Omega)$ denote the context-dependent matching profile:

$$\Gamma(\Omega) = [\mu_0(\Omega), \mu_1(\Omega), \mu_2(\Omega)]$$

where:

- $\mu_0(\Omega)$ measures state or value agreement
- $\mu_1(\Omega)$ measures rate-of-change agreement
- $\mu_2(\Omega)$ measures change-of-rate agreement

These components are conceptual matching indicators rather than mandatory physical derivatives in every module. Their precise interpretation is domain-specific. For example, μ_1 may represent the rate at which a football formation is losing effectiveness, whereas μ_2 may represent whether that deterioration is accelerating.

The module may use any contextually required subset:

$\Gamma_o = [\mu_0]$

$\Gamma_{01} = [\mu_0, \mu_1]$

$\Gamma_{12} = [\mu_1, \mu_2]$

or

$\Gamma_{012} = [\mu_0, \mu_1, \mu_2]$

No universal hierarchy among μ_0, μ_1, and μ_2 is assumed. The relevant combination depends on the event, active module, decision horizon, and validation requirements.

Let $\delta(\Omega)$ denote the measured inadequacy of the active solution and let δ_{th} denote the module-specific inadequacy threshold.

A simplified SPARK activation condition is

$$SPARK(\Omega) = 1$$

if

$\delta(\Omega) \geq \delta_{th}$

or

$\neg \exists a \in L_o : Adequate(a, G*, \Omega) = 1.$

Otherwise,

$SPARK(\Omega) = 0$

SPARK therefore explains why and when constructive search begins. It does not specify the completed strategy and does not guarantee that a valid innovation will be found.

2.4 Bounded Modules and Segment Representation

The proposed architecture does not require a universal robot with unrestricted competence. A robot may operate through a general-purpose core supplemented by daily-life, professional, scientific, artistic, tactical, or other specialist modules.

Not every robot must contain every module. A cue-driven ballet robot may require recognition, cue–action mapping, constraint checking, and movement execution. A scientist robot may additionally require related-problem retrieval, hypothesis construction, model evaluation, and evidence validation. A robot is independent only within the competence and authority provided by its installed and active modules.

To avoid confusion with the matching variables, a module is denoted by Mod_i:

$Mod_i = [IN_i, G_i, L_i, C_i, A_i, V_i, OUT_i]$

where:

- IN_i is the module-activation input
- G_i is the goal or admissible goal class
- L_i is the module-specific segment and solution library
- C_i is the applicable constraint set
- A_i is the selected or assembled action
- V_i is the module-specific validation operator
- OUT_i is the result report

Each module limits:

- The goals that may be processed
- The actions and segments that may be searched
- The robot's operational competence
- The applicable safety and authority constraints
- The permitted use of stochastic selection
- The stopping conditions
- The validation criteria

An individual functional segment s_j has the standardized representation

$s_j = [Name_j, IN_j, F_j, OUT_j, C_j]$

where $Name_j$ identifies the segment, IN_j specifies its input conditions, F_j specifies its function, OUT_j specifies its output, and C_j specifies its local constraints.

This representation allows segments acquired in different situations to be compared at their functional interfaces without merging all robot knowledge into a single unbounded library.

A segment may represent a physical movement, logical inference, tactical transition, spatial arrangement, scientific method, communication act, or other bounded transformation. The architecture is therefore not restricted to mechanical actions.

2.5 Deterministic Segment Search and ε-Compatible Assembly

After activation of Mod_i, the current event is compared with candidate segments in the relevant bounded library L_i.

Event–segment relevance is defined as

$\mu_{ei} = Match(Event, s_i)$

and segment–segment compatibility is defined as

$\mu_{ij} = Match(s_i, s_j)$

The highest-scoring segment need not be identical to the required operation. It must instead be sufficiently compatible with the required inputs, outputs, constraints, resources, timing, and continuity conditions.

A candidate strategy is assembled progressively:

$S_n = s_{i1} \oplus s_{i2} \oplus \ldots \oplus s_{in}$

where $\oplus$ denotes functionally compatible assembly rather than simple concatenation.

For every adjacent pair of segments, the output of the preceding segment must be compatible with the input of the following segment:

$e_{ij} = d(OUT_i, IN_j)$

$e_{ij} \leq \varepsilon_{ij}$

where d is a module-appropriate interface-distance function and ε_{ij} is the maximum permitted interface error between s_i and s_j.

The compatibility of the complete candidate is represented by

$e(S_n, \Omega) \leq \varepsilon$

where $e(S_n, \Omega)$ is the aggregate functional and contextual compatibility error and ε is the maximum admissible tolerance for the active module.

The candidate must also satisfy the applicable constraint set:

$S_n \vDash C_i$

Accordingly, an admissible assembled candidate satisfies $S_n \in S^A(\Omega)$.

if

$e(S_n, \Omega) \leq \varepsilon$

and

$S_n \vDash C_i$

where $S^A(\Omega)$ denotes the set of assembled candidates admissible in context Ω.

The complete assembly must respect module authority, safety requirements, available resources, timing, start–stop conditions, and the active goal. A new strategy is therefore not inferred from visual similarity or statistical association alone. It is constructed from a traceable sequence of functionally compatible transitions.

2.6 Functional Validation and Library Expansion

An assembled candidate becomes an innovation only after execution, simulation, formal verification, or another domain-valid evaluation establishes that it satisfies the active goal and applicable constraints.

Module-specific validation is defined as
$v_i = V_i(OUT_i, G_i, C_i)$

The validation result belongs to the set
$v_i \in$ {SUCCESS, INCOMPLETE, FAILED, REFERRED}.

The outcomes have the following operational meanings:

- **SUCCESS**: The goal and applicable constraints have been satisfied. The module terminates and returns control to the scheduler.
- **INCOMPLETE**: The result provides partial progress but does not yet satisfy the goal. Additional segment search, assembly, or evaluation is required.
- **FAILED**: The candidate does not satisfy the functional or constraint conditions. Recovery, rejection, or safe termination is initiated.
- **REFERRED**: The task exceeds the module's competence or authority and is transferred to another authorized module, system, or human operator.

These outcomes provide explicit stopping and transfer rules. They prevent the robot from continuing an action merely because the candidate is new.

Let S_n denote the successfully validated strategy. Library admission requires
$S_n \notin L_0$
$V_i(S_n, G_i, C_i) = SUCCESS$

The expanded library is then
$L_1 = L_0 \cup \{S_n\}$

The transition can be expressed temporally as
$S_n \notin L(t_0)$
$L(t_1) = L(t_0) \cup \{S_n\}$
where $t_1 > t_0$
The first validated construction of S_n is an innovation relative to $L(t_0)$. Its later selection from $L(t_1)$ is retrieval rather than a new innovation.

A small composite remains a reusable segment within its originating module. It is promoted to a submodule or module only if it acquires:

- An independently specified goal or goal class
- Distinct activation and termination conditions
- A substantial internal segment library
- Separate scheduling requirements
- Its own constraint and authority boundaries
- Distinct validation criteria

This promotion rule enables architectural growth without treating every newly learned composite as a new general capability.

2.7 Unified Interpretation Across Domains

The proposed mechanism applies across tactical, compositional, deductive, scientific, and engineering domains. Although their materials and validation criteria differ, the operational sequence remains constant: inadequacy activates SPARK, bounded search retrieves compatible elements, a new candidate is assembled and evaluated, and a successfully validated structure is retained.

A football-coach robot reorganizes players, positions, timing, and tactical direction; a choreography robot reorganizes movements, performers, space, and expression; and a mathematics robot assembles definitions, lemmas, and valid inferences into a proof. Similarly, a scientist robot combines observations, transferable principles, methods, and models to construct a testable explanation. In engineering, a robot unable to use a conventional mechanical pump may combine buoyancy, heating, density differences, and fluid circulation to construct a bubble-pump solution.

In every case, the constituent elements may already be known. Innovation lies in constructing and validating a useful complete organization that was not previously available in the active library: $S_n \notin L_0$.
Robotic innovation may therefore occur at physical, tactical, compositional, deductive, explanatory, or system-organizational levels without requiring the invention of a new mechanical movement.

2.8 Operational Criterion for Robotic Innovation

The complete innovation mechanism is summarized as Ω:

→ Recognition
→ Goal or Mission Evaluation
→ Inadequacy Detection
→ SPARK
→ Module Activation
→ Bounded Segment Search
→ ε-Compatible Assembly
→ Functional and Constraint Validation
→ Retention or Rejection

A more compact representation is

$SPARK(\Omega) = 1$
$\rightarrow Search(L_i)$
$\rightarrow S_n$
$\rightarrow V_i(S_n, G_i, C_i)$

If
$S_n \notin L_o$
and
$V_i(S_n, G_i, C_i) = SUCCESS$
then
$L_1 = L_o \cup \{S_n\}$

If validation returns INCOMPLETE, FAILED, or REFERRED, S_n is not admitted to the active library as a validated complete solution.

Accordingly, robotic innovation is defined as the traceable construction, validation, and retention of a useful functional organization that was not stored as a complete solution in the robot's active pre-event library.

This definition does not claim creation from nothing, and it does not reduce innovation to retrieval, optimization, or arbitrary recombination. It supplies an observable engineering criterion:

- The pre-event and post-event libraries can be compared
- The activation condition can be identified
- The segment search can be bounded
- The assembly path can be traced
- The applicable constraints can be inspected
- The validation result can be reproduced
- The retained structure can be distinguished from its later retrieval

SPARK explains why and when constructive search begins. The bounded modular segment architecture explains how a new candidate is constructed. Functional validation determines whether that candidate becomes an innovation.

3. Operational Modes and Illustrative Cases

This section distinguishes the principal operational modes supported by the proposed bounded modular architecture. Its purpose is not to design a single robot containing every described capability. Instead, each case illustrates a different relationship among external commands, internally generated goals, contextual arbitration, stored actions, and innovative construction.

The cases are analyzed only at the decision-architecture level. Perception, object recognition, localization, navigation, trajectory generation, manipulation, physiological sensing, and detailed physical execution belong to separate systems unless explicitly included in the active module.

The five operational cases are:

1. Command-free generation of a goal
2. Dynamic internal arbitration under changing four-dimensional conditions
3. Autonomous execution initiated by a recognized cue
4. Mission-governed execution in which internal preferences cannot redefine the assigned objective
5. Initiation of scientific activity through related-problem and transferable-knowledge matching

These cases do not imply that every autonomous or independent robot must contain all corresponding modules. Module installation depends on the robot's designated function, competence, safety requirements, and governing authority. The examples collectively clarify the boundaries among autonomous execution, independent goal selection, and innovative strategy construction.

3.1 Command-Free Goal Generation

Command-free goal generation distinguishes a fully independent robot from an ordinary autonomous robot. An autonomous robot selects or executes actions for an externally assigned goal. A fully independent robot can additionally generate candidate goals and select among them when no present command or assigned mission exists.

The governing condition is
No Present Command + No Assigned Mission

Let the robot receive the following internal-state reports:
I_p = Desire to Play
I_c = Desire to Drink Coffee
I_r = Desire to Relax
The corresponding external-condition reports are
E_t = Sufficient Time Available
E_β = Ball-Playing Facility Available
E_c = Coffee Facility Available
E_{co} = Companion Available
E_r = Relaxation Facility Available

These variables are supplied to the decision architecture as reports. Their perception, recognition, localization, and measurement belong to separate systems. An external report establishes the existence of an opportunity; it does not constitute a command to act. Recognition of a ball-playing facility, for example, does not require the robot to play.

From the reported internal and external conditions, the robot constructs three candidate goals:
G_1 = Play with the ball
G_2 = Drink coffee
G_3 = Relax

The contextual matching values are
$M_1(\Omega) = \text{Match}(I_p, E_t, E_\beta, E_{co})$
$M_2(\Omega) = \text{Match}(I_c, E_t, E_c)$
$M_3(\Omega) = \text{Match}(I_r, E_t, E_r)$

where $\Omega = (X_r, X_e, T_r, T_e)$ is the current four-dimensional context.

Contextual matching alone does not make a candidate executable. Each candidate must also satisfy feasibility, competence, safety, authority, and module-specific constraints. The admissible candidate-goal set is therefore

$$G^A(\Omega) = \{G_k : M_k(\Omega) \geq M_{th} \wedge F_k(\Omega) = 1 \wedge C_k(\Omega) = 1\}$$

where:
- M_{th} is the minimum contextual-matching threshold
- $F_k(\Omega)$ indicates whether G_k is feasible within the current context
- $C_k(\Omega)$ indicates whether G_k satisfies all applicable constraints.

Each admissible candidate is assigned an integrated decision value $J_k(\Omega)$. The detailed construction of $J_k(\Omega)$ from multidimensional supporting, opposing, redirecting, and neutral internal segments is presented in Section 3.2.

Under deterministic selection
$k* = \arg\max_k J_k(\Omega)$
$G* = G_{k*}$

For example, if
$J_2(\Omega) > J_1(\Omega) > J_3(\Omega)$
and G_1, G_2, and G_3 satisfy the applicable admissibility conditions, then
$G*$ = Drink coffee

The resulting goal is self-initiated because it was selected without a present command or assigned mission. However, self-initiation does not by itself constitute innovation. If an adequate complete action $a*$ for $G*$ is already available in the active library L_0, then

$a* \in L_0$ and the robot proceeds through ordinary autonomous execution.

If no adequate complete action exists,
$\neg\exists a \in L_0 : \text{Adequate}(a, G*, \Omega) = 1$

The unresolved admissible goal activates SPARK and initiates bounded segment search and ε-compatible strategy assembly.

The architecture may also select deliberate no-action. This occurs when no candidate goal satisfies the volitional and admissibility conditions:

$G^A(\Omega) = \emptyset$
$\Rightarrow$
$G*$ = No Action

The complete process is therefore
No Present Command + No Assigned Mission
→ Internal and External-State Reports
→ Candidate-Goal Generation
→ Contextual Matching
→ Admissibility Evaluation

→ Integrated Decision Values
→ Self-Initiated Goal / No Action
→ Stored-Action Selection or SPARK
This mechanism determines what the robot presently chooses to pursue. It remains distinct from innovative construction, which begins only when an admissible selected goal cannot be satisfied by an adequate complete solution in L_o.

3.2 Dynamic Internal Arbitration: The Tennis Case

The availability of the external conditions required for an action does not imply that the robot must execute that action immediately. In command-free operation, the reported opportunity is evaluated together with multidimensional internal segments whose values change across the four-dimensional context.

$\Omega = (X_r, X_e, T_r, T_e)$

Accordingly, the robot–environment relation is neither static nor stationary. A goal that is strongly supported at one moment may be postponed or rejected when the robot's internal state, external conditions, operational history, or event timing changes.

Consider a robot for which the following external conditions are reported:
E_c = tennis court available
E_p = tennis partner available
E_e = required equipment available
E_t = sufficient time available

The candidate goal is
G_t = Play tennis

Its contextual matching value is
$M_t(\Omega) = \text{Match}(I_t, E_c, E_p, E_e, E_t)$
where I_t represents the robot's current internal inclination toward playing tennis.

Suppose that
$M_t(\Omega) \geq M_{th}$

This condition indicates that the internal inclination and external opportunity are sufficiently compatible to generate G_t as a candidate goal. It does not establish that G_t must be selected or executed immediately.

The candidate is evaluated through multidimensional supporting and opposing internal segments.
The supporting segment is
$D_t(\Omega) = [d_1(\Omega), d_2(\Omega), \ldots, d_n(\Omega)]$

where its components may represent desire, anticipated enjoyment, curiosity, social attraction, exercise value, habit, expected benefit, or urgency.

The opposing segment is
$A_t(\Omega) = [a_1(\Omega), a_2(\Omega), \ldots, a_m(\Omega)]$

where its components may represent fatigue, risk, cost, an unfinished task, future consequences, resource limitations, social constraints, or reasons for postponement.

The architecture may also receive non-decisive contextual segments:
$N_t(\Omega) = [n_1(\Omega), n_2(\Omega), \ldots, n_h(\Omega)]$

where $N_t(\Omega)$ contains information that is relevant to interpreting the situation but does not independently support or oppose G_t. Examples include the colour of the court, the identity of an unused nearby object, or another reported property that does not affect the current decision.

The three segment classes are processed concurrently:
$E(\Omega)$
→ $[D_t(\Omega) \parallel A_t(\Omega) \parallel N_t(\Omega)]$
→ Integration/Arbitration

where $\parallel$ denotes parallel processing.

Let the aggregate supporting contribution be
$\bar{D}_t(\Omega) = \sum_{i=1}^{n} w^d_i d_i(\Omega)$
and let the aggregate opposing contribution be
$\bar{A}_t(\Omega) = \sum_{j=1}^{m} w^a_j a_j(\Omega)$

where w^d_i and w^a_j are module-specific weights. The weights may be fixed, context dependent, or learned within the permitted boundaries of the installed module.

The signed internal decision balance is then
$B_t(\Omega) = \bar{D}_t(\Omega) - \bar{A}_t(\Omega)$

The relation between contextual matching and internal arbitration is represented by
$J_t(\Omega) = \Psi(M_t(\Omega), B_t(\Omega), N_t(\Omega))$

where $J_t(\Omega)$ is the integrated decision value introduced in Section 3.1. The neutral segment $N_t(\Omega)$ may modify contextual interpretation but does not directly contribute a positive or negative preference unless the active module reclassifies one of its components as decision-relevant.

Three principal outcomes are possible.

If
$B_t(\Omega) > \theta_+$
and
$G_t \in G^A(\Omega)$

then the internal balance supports the candidate:
$G* =$ Play tennis now
If
$B_t(\Omega) < \theta_-$

then the opposing contributions dominate. The robot rejects the immediate execution of G_t:
$G* \neq$ Play tennis now.

This outcome does not necessarily imply complete inactivity. The robot may select an admissible alternative such as
$G_{alt} \in$ {Complete another task, Rest, Reschedule tennis, No action}

If
$\theta_- \leq B_t(\Omega) \leq \theta_+$

the internal balance is insufficient for an immediate positive or negative commitment. The architecture may then request additional information, postpone the decision, preserve G_t as a pending goal, or apply a bounded tie-breaking procedure permitted by the active module.

In addition to the supporting segment $D_t(\Omega)$, the opposing segment $A_t(\Omega)$, and the neutral contextual segment $N_t(\Omega)$, the architecture may include a bounded stochastic contribution $R_t(\Omega)$. These segments have distinct functions. $D_t(\Omega)$ supports the candidate goal, $A_t(\Omega)$ opposes or redirects it, and $N_t(\Omega)$ supplies relevant contextual information without directly supporting or opposing the goal. $R_t(\Omega)$ is used only as a bounded tie-breaking or exploratory mechanism after feasibility, competence, safety, authority, and all other admissibility conditions have been satisfied. If $A^*(\Omega)$ denotes the admissible action set, the stochastic contribution may select only from $A^*(\Omega)$, never from the complete Action Library. Its permitted probability $p_r(\Omega)$ is context-dependent and may be zero in safety-critical operation.

The thresholds θ_- and θ_+ create a decision interval that prevents small fluctuations from repeatedly reversing the result. They may vary according to context. For example, safety-critical or resource-intensive activities may require a larger positive balance than low-risk recreational activities.

The dynamic character of the decision can be illustrated at two successive times. At time T_{r1}, the external conditions may be favourable:
$E_c = E_p = E_e = E_t = 1$
but the robot may have an unfinished higher-priority task and a high fatigue report:
$\bar{A}_t(\Omega_1) > \bar{D}_t(\Omega_1)$
Therefore,
$B_t(\Omega_1) < \theta_-$
and the decision is
$G*(\Omega_1) =$ Postpone tennis

At a later time T_{r2}, the tennis court, partner, and equipment may still be available, while the competing task has been completed and the fatigue contribution has decreased: $\bar{D}_t(\Omega_2) > \bar{A}_t(\Omega_2)$.

Therefore,
$B_t(\Omega_2) > \theta_+$
and, provided all admissibility conditions remain satisfied,
$G*(\Omega_2) =$ Play tennis

Thus,
$\Omega_1 \neq \Omega_2$
$\Rightarrow$
$B_t(\Omega_1) \neq B_t(\Omega_2)$
$\Rightarrow$
$G*(\Omega_1) \neq G*(\Omega_2)$

This example demonstrates why internal–external matching cannot be treated as a static stimulus–response relation. The same tennis court, partner, equipment, and general desire may produce different decisions as the

robot-relative and event-relative temporal conditions change.

The decision process is therefore
External Tennis Opportunity
→ Candidate Goal G_t
→ Contextual Match $M_t(\Omega)$
→ [Supporting $D_t(\Omega)$ ‖ Opposing $A_t(\Omega)$ ‖ Neutral $N_t(\Omega)$]
→ Signed Balance $B_t(\Omega)$
→ Integrated Value $J_t(\Omega)$
→ Admissibility
→ Play Now / Alternative Goal / Postpone / No Action

If "Play tennis" is selected and an adequate complete action already exists in L_o, the robot proceeds through autonomous execution. If the goal is admissible but no adequate complete action exists in L_o, SPARK may activate bounded constructive search. The internal arbitration mechanism therefore determines whether and when the goal should be pursued; it does not itself constitute innovation.

3.3 Cue-Driven Autonomous Execution: The Ballet Robot

Not every autonomous action requires command-free goal generation or internal volitional arbitration. In cue-driven operation, a recognized external signal activates a previously defined goal–action mapping. The robot may autonomously control timing, balance, trajectory, and movement execution, but it does not independently determine what goal to pursue.

Consider a ballet robot designed to perform a particular dance figure when it receives the corresponding musical or acoustic cue.

Let
Q_β = recognized ballet cue
G_β = perform the assigned ballet figure
a_β = stored executable movement sequence

The cue-processing system supplies the recognition report $\rho(Q_\beta) \geq \rho_{req}$, where $\rho(Q_\beta)$ is the cue-recognition confidence and ρ_{req} is the minimum confidence required for activation.

If the cue is recognized with sufficient confidence, it activates the ballet module Mod_β:

$Q_\beta \rightarrow Mod_\beta$
The module contains a predefined mapping
$Q_\beta \mapsto G_\beta \mapsto a_\beta$
If the required action is available in the active library L_o, then
$a_\beta \in L_o$

The robot retrieves and executes the stored action:
$a* = \text{Retrieve}(G_\beta, L_o)$
$a* = a_\beta$

The complete nominal process is therefore
Recognized Cue
→ Ballet-Module Activation
→ Assigned Goal G_β
→ Stored-Action Retrieval
→ Constraint Verification
→ Autonomous Execution
→ Result Validation.

The robot's internal supporting, opposing, or stochastic segments do not participate in selecting the assigned performance goal:
$[D_k(\Omega) \parallel A_k(\Omega) \parallel N_k(\Omega) \parallel R_k(\Omega)] \notin$ Goal-Selection Arbitration.
where:

- $D_k(\Omega)$: supporting
- $A_k(\Omega)$: opposing
- $N_k(\Omega)$: neutral contextual
- $R_k(\Omega)$: bounded stochastic

Thus, curiosity, preference, anticipated pleasure, reluctance, or attraction to an alternative dance figure cannot replace G_β with another self-generated goal. The recognized cue determines which installed figure is to be performed.

This exclusion applies only to goal selection. It does not remove safety, feasibility, competence, or execution monitoring. Before the movement begins, the robot must verify that the assigned figure is admissible in the current context:
$F_\beta(\Omega) = 1$
$C_\beta(\Omega) = 1$

where $F_\beta(\Omega)$ denotes physical and operational feasibility and $C_\beta(\Omega)$ denotes satisfaction of the applicable constraints, that is, the ballet constraint condition. The symbols Q_β and $C_\beta(\Omega)$ distinguish the recognized ballet

cue from the ballet constraint condition.

The admissible execution condition is
$\rho(Q_\beta) \geq \rho_{req}$
$\wedge F_\beta(\Omega) = 1$
$\wedge C_\beta(\Omega) = 1$
$\wedge a_\beta \in L_o$
If all conditions are satisfied, then
$Execute(a_\beta, \Omega) = 1$

If cue confidence is insufficient
$\rho(Q_\beta) < \rho_{req}$
The robot does not infer the requested figure:
OUT_β = REQUEST CLARIFICATION
or
OUT_β = NO ACTION

If the cue is recognized but the movement is unsafe or infeasible,
$F_\beta(\Omega) = 0$
or
$C_\beta(\Omega) = 0$

execution is inhibited:
OUT_β = SAFE STOP
or
OUT_β = REFERRED

For example, the robot may correctly recognize the musical cue while detecting insufficient performance space, an obstacle, loss of balance, or a nearby person inside the required movement area. In that case, cue recognition remains valid, but the corresponding movement is not currently admissible.

Recognition and permission must therefore be distinguished:
Cue Recognized ≠ Execution Permitted.

Similarly,
Autonomous Execution ≠ Independent Goal Generation

The robot autonomously performs the assigned figure because it controls the detailed execution without continuous human intervention. Nevertheless, the goal originates from the recognized external cue rather than from the robot's internal candidate-goal mechanism.

The outcome is validated by the ballet module:
$v_\beta = V_\beta(OUT_\beta, G_\beta, C_\beta)$

where
$v_\beta \in$ {SUCCESS, INCOMPLETE, FAILED, REFERRED}

SUCCESS indicates that the required figure was completed within the permitted spatial, temporal, balance, and safety constraints. INCOMPLETE indicates partial execution. FAILED indicates that the figure was not successfully completed. REFERRED indicates that execution exceeded the module's competence or permitted operating conditions.

No innovation occurs when the robot retrieves and performs the stored figure:
$a_\beta \in L_o \Rightarrow$ Cue-Driven Autonomous Execution

If no adequate figure exists in L_o, the response depends on the installed architecture. A ballet robot without an innovation module cannot construct a replacement figure:
$\neg\exists a \in L_o : Adequate(a, G_\beta, \Omega) = 1$
$\Rightarrow OUT_\beta$ = REFERRED or NO ACTION

Only a robot additionally equipped with an authorized choreography or innovative-construction module may activate SPARK:
$\neg\exists a \in L_o : Adequate(a, G_\beta, \Omega) = 1$
$\wedge Mod_{inn}$ installed
$\wedge Authority(Mod_{inn}, G_\beta) = 1$
$\Rightarrow SPARK(\Omega) = 1$

Even in that case, construction of a new figure would be a separate operation from cue-driven execution. It would require bounded segment search, ε-compatible assembly, constraint evaluation, and validation before the new figure could be performed.

The ballet example therefore establishes a boundary condition of the proposed architecture: a robot may be highly autonomous in execution while lacking command-free goal generation and innovative construction. Autonomy describes how the assigned action is executed; independence describes how the goal is generated; innovation describes whether a validated solution absent from L_o is constructed.

3.4 Mission-Governed Autonomous Execution

Mission-governed execution represents a second form of externally directed autonomy. Unlike the ballet robot, whose action is activated by a discrete performance cue, a mission-governed robot receives a persistent objective together with authority, safety, legal, temporal, and operational constraints. The robot may autonomously select among permitted actions, but it cannot replace the assigned mission with a goal generated from its own preferences.

Let
C_m = mission-contract validity
G_m = assigned mission goal
H_m = required human authorization
R_m = compliance with applicable rules
I_m = identity and situational-confidence condition
F_m = operational feasibility
S_m = safety condition

The mission becomes active only if its source, content, authority, and validity have been verified:
$C_m = 1$
Mission activation establishes
$C_m = 1$
$\Rightarrow G* = G_m$.

Thus, the assigned mission goal governs the robot's objective-selection layer. Multidimensional internal preference segments do not participate in redefining that objective:
$C_m = 1$
$\Rightarrow [D_k(\Omega) \parallel A_k(\Omega) \parallel N_k(\Omega) \parallel R_k(\Omega)] \notin$ Mission-Goal Arbitration

Here, $D_k(\Omega)$, $A_k(\Omega)$, and $R_k(\Omega)$ have the meanings introduced in Section 3.2. Their exclusion does not imply the absence of internal monitoring. The robot may still evaluate energy, damage, uncertainty, timing, resource availability, and system integrity. However, these signals function as feasibility, safety, scheduling, or recovery variables; they do not function as personal preferences capable of replacing G_m.

Accordingly,
Internal Monitoring ≠ Independent Mission Selection
A mission-governed robot may autonomously determine how to pursue G_m only within the admissible action set:
$A^M(\Omega) = \{a_j \in L_o :$
$\text{Relevant}(a_j, G_m, \Omega) = 1$
$\wedge H_j(\Omega) = 1$
$\wedge R_j(\Omega) = 1$
$\wedge I_j(\Omega) = 1$
$\wedge F_j(\Omega) = 1$
$\wedge S_j(\Omega) = 1\}$.

Here, $A^M(\Omega)$ contains only stored actions that are relevant to the assigned mission and satisfy all applicable authorization, rule, identification, feasibility, and safety conditions.

The governing admissibility condition is
$H_j(\Omega) \wedge R_j(\Omega) \wedge I_j(\Omega) \wedge F_j(\Omega) \wedge S_j(\Omega) = 1$

If any required condition fails, then
$H_j(\Omega) \wedge R_j(\Omega) \wedge I_j(\Omega) \wedge F_j(\Omega) \wedge S_j(\Omega) = 0$
and the corresponding action is excluded:
$a_j \notin A^M(\Omega)$

This formulation is stronger and clearer than multiplying the conditions, because the variables represent Boolean requirements rather than numerical performance values.

Within the admissible set, the robot may select the action having the highest mission-specific decision value:
$a* = \arg\max_{a_j \in A^M(\Omega)} U_m(a_j, \Omega)$

where $U_m(a_j, \Omega)$ evaluates mission relevance, expected effectiveness, resource requirements, timing, risk, and other permitted operational factors.

The complete nominal process is
Validated Mission Contract
→ Assigned Goal G_m
→ Context Recognition
→ Admissible Action Set $A^M(\Omega)$
→ Mission-Constrained Action Selection
→ Autonomous Execution
→ Monitoring and Validation

For example, a military-support robot may receive an externally assigned reconnaissance, transport, communication, evacuation-support, or area-monitoring mission. The robot may autonomously select a route,

schedule its movements, manage resources, avoid obstacles, or choose among authorized observation positions. However, it cannot replace the assigned mission with a recreational, exploratory, or internally preferred goal.

The distinction is therefore

Autonomous Selection Within a Mission

≠

Independent Selection of the Mission.

The mission does not authorize unrestricted action. Every candidate remains subject to the governing constraint hierarchy:

Human Authority
→ Applicable Law and Rules
→ Mission Contract
→ Safety and Identity Conditions
→ Feasibility
→ Action Selection.

A lower-level objective cannot override a higher-level restriction. If the action would violate a governing rule or lacks required authorization, its expected mission benefit cannot make it admissible:

$R_j(\Omega) = 0$
$\vee\ H_j(\Omega) = 0$
$\Rightarrow a_j \notin A^M(\Omega)$

Similarly, insufficient identity or situational confidence prevents an action whose permissibility depends on that identification:

$I_j(\Omega) = 0$
$\Rightarrow a_j \notin A^M(\Omega)$

When no admissible action exists,

$A^M(\Omega) = \emptyset$

The robot must not improvise outside its authority. The appropriate output is selected from a bounded set such as

$OUT_m \in$ {HOLD, SAFE STOP, REQUEST AUTHORIZATION, REQUEST CLARIFICATION, REFERRED}.

The robot may activate SPARK only when innovative construction is installed and explicitly authorized for the active mission class:

$A^M(\Omega) = \emptyset$
$\wedge$ Mod^{INN} installed
$\wedge$ Authority(Mod^{INN}, G_m) = 1
$\Rightarrow$ SPARK(Ω) = 1

SPARK does not remove mission constraints. Any constructed candidate S_n must satisfy the same or stricter authorization, rule, identity, feasibility, and safety conditions:

$S_n \in S^M(\Omega)$

if

Relevant(S_n, G_m, Ω) = 1
$\wedge\ H_n(\Omega) = 1$
$\wedge\ R_n(\Omega) = 1$
$\wedge\ I_n(\Omega) = 1$
$\wedge\ F_n(\Omega) = 1$
$\wedge\ S_n^c(\Omega) = 1$

The notation $S_n^c(\Omega)$ denotes the safety-constraint status of the newly constructed strategy and distinguishes it from the strategy S_n itself.

If innovative construction is not installed or not authorized, the absence of an adequate stored action results in referral or safe termination rather than unrestricted improvisation:

$A^M(\Omega) = \emptyset$
$\wedge$ [Mod_{inn} not installed
$\vee$ Authority(Mod_{inn}, G_m) = 0]
$\Rightarrow OUT_m \in$ {HOLD, SAFE STOP, REFERRED}

Mission completion is evaluated by the mission-specific validation operator:

$v_m = V_m(OUT_m, G_m, C_m^c)$

where C_m^c represents the complete mission-constraint set and

$v_m \in$ {SUCCESS, INCOMPLETE, FAILED, REFERRED}

The mission-governed example therefore demonstrates that internal independence is not required for every autonomous system. A robot can exercise substantial autonomy in perception, planning, action selection, execution, and recovery while its governing goal remains externally assigned. Its autonomy concerns how an authorized mission is performed; it does not provide authority to determine whether that mission should exist or to replace it with a self-generated objective.

3.5 Related-Problem Initiation in a Scientist Robot

A scientist robot represents a form of bounded independent operation in which research activity may

begin without a present task-specific command, provided that the observed problem falls within an installed scientific module and a sufficient reasoning basis can be established.

The robot does not begin from unrestricted curiosity or search across all possible domains. It operates within a validated research module Mod_s containing a defined problem class, knowledge library, applicable methods, evidential standards, safety constraints, and validation procedures.

Let
P_n = newly recognized problem
L_s = active scientific knowledge library
$P^R(P_n)$ = set of stored problems related to P_n
$K^T(P_n)$ = set of transferable principles, methods, and validated knowledge applicable to P_n

The related-problem set is constructed as
$P^R(P_n) = \{P_j \in L_s : Rel(P_n, P_j) \geq \rho_{th}\}$
where $Rel(P_n, P_j)$ measures the structural relevance of stored problem P_j to the newly recognized problem P_n, and ρ_{th} is the minimum relevance threshold.

Similarity is not restricted to shared surface features. Two problems may be related through their variables, causal structure, constraints, boundary conditions, mathematical form, functional objective, or applicable method.

The transferable-knowledge set is
$K^T(P_n) = \{k_j \in L_s : Transfer(k_j, P_n) \geq \tau_{th}\}$

where $Transfer(k_j, P_n)$ measures the applicability of knowledge element k_j to P_n, and τ_{th} is the required transfer threshold.

The complete research-basis set is therefore
$B^R(P_n) = P^R(P_n) \cup K^T(P_n)$

Research initiation is permitted only when the recognized problem falls within the competence and authority of Mod_s and the research-basis set is non-empty:
$Recognized(P_n) = 1$
$\wedge\ Competent(Mod_s, P_n) = 1$
$\wedge\ Authority(Mod_s, P_n) = 1$
$\wedge\ B^R(P_n) \neq \emptyset$
$\Rightarrow Research\text{-}Initiation(P_n) = 1$

This condition does not require an identical previous problem. The robot may begin from a related problem, transferable scientific principle, applicable mathematical structure, validated method, causal model, or relevant body of evidence.

If
$P^R(P_n) = \emptyset$
but
$K^T(P_n) \neq \emptyset$
Research may still begin through cross-problem or cross-domain knowledge transfer.

Conversely, if
$B^R(P_n) = \emptyset$

then the installed module lacks a justified starting basis:
$Research\text{-}Initiation(P_n) = 0$

The corresponding output is
$OUT_s \in$ {INSUFFICIENT BASIS, REQUEST INFORMATION, REFERRED}

The robot does not fabricate premises or initiate unrestricted reasoning outside its installed competence. It may request additional data, transfer the problem to another authorized module, or refer it to a human scientist.

When $B^R(P_n) \neq \emptyset$, the robot constructs one or more candidate research paths:
$H^A(P_n) = \{H_1, H_2, \ldots, H_n\}$

where each H_k may represent a candidate hypothesis, explanatory relation, model, proof direction, experimental design, or problem-solving strategy.

Each candidate is assembled from traceable knowledge elements:
$H_k = k_{i1} \oplus k_{i2} \oplus \cdots \oplus k_{im}$

where $k_{ij} \in B^R(P_n)$ and $\oplus$ denotes logically and methodologically compatible assembly.

Candidate generation does not establish scientific validity. Each H_k must satisfy the module's logical, evidential, methodological, safety, and authority constraints:

$H_k \in H^{ADM}(P_n)$
if
$Coherent(H_k) = 1$
$\wedge\ Testable(H_k) = 1$
$\wedge\ MethodValid(H_k) = 1$
$\wedge\ Safe(H_k) = 1$
$\wedge\ Authorized(H_k) = 1$

The admissible research candidate having the highest scientific evaluation value may then be selected:
$k* = \arg\max_k U_s(H_k, P_n, \Omega)$
$H* = H_{k*}$

where U_s evaluates explanatory relevance, evidential support, testability, expected information gain, resource requirements, and methodological adequacy.

The selected candidate initiates a bounded research process:
P_n
→ Related-Problem and Transferable-Knowledge Search
→ Research-Basis Set $B^R(P_n)$
→ Candidate Construction
→ Admissibility Evaluation
→ Candidate Selection
→ Investigation
→ Validation
The result is evaluated by the scientific validation operator: $v_s = V_s(OUT_s, P_n, C_s)$.where C_s is the applicable scientific-constraint set and $v_s \in$ {SUPPORTED, INCONCLUSIVE, REJECTED, REFERRED}.

SUPPORTED indicates that the candidate has satisfied the installed evidential and methodological criteria. INCONCLUSIVE indicates that the available evidence is insufficient and further investigation is required. REJECTED indicates that the candidate has failed logical, empirical, or methodological evaluation. REFERRED indicates that the problem exceeds the module's competence, resources, or authority.

Independent research initiation and scientific innovation remain distinct. If the robot selects and applies a complete stored method already contained in L_s, the operation is autonomous scientific problem solving: $a* \in L_s$.

If the robot constructs and validates a new explanatory or functional structure S_n that was not stored as a complete solution, then the operational novelty condition is
$S_n \notin L_s(t_0)$

Following successful validation,
$L_s(t_1) = L_s(t_0) \cup \{S_n\}$
where $t_1 > t_0$

The first case represents retrieval and application of existing scientific knowledge. The second may constitute robotic scientific innovation relative to the active pre-event library.

The scientist-robot example therefore establishes three boundaries. First, no present command is required if the module has standing authority to investigate recognized problems within its domain. Second, research cannot begin without a related problem or another transferable and validated knowledge basis. Third, beginning an investigation does not itself constitute innovation; innovation occurs only when the robot constructs and validates a useful scientific structure absent from its active library.

3.6 Module Requirements Across Robot Types

The preceding cases demonstrate that autonomy, command-free goal generation, internal arbitration, and innovation are separable capabilities. A robot does not need to contain every module described in the proposed architecture. Its required module set depends on its designated role, operating environment, competence boundaries, safety requirements, and governing authority.

Let
$Mod^R = \{Mod_1, Mod_2, \ldots, Mod_n\}$ denote the complete set of modules defined by the proposed architecture.

Let
$Mod(r) \subseteq Mod^R$ denote the modules installed and authorized for robot r.

The operational capability of r is determined by Mod(r), rather than by the architecture's complete theoretical module set. Thus,
$Mod_i \notin Mod(r)$
implies that robot r cannot use the capability implemented by Mod_i.

This absence does not necessarily constitute a system failure. It may be an intentional architectural restriction. A robot designed only for cue-driven performance, for example, does not require a module for command-free goal generation. Likewise, a mission-governed robot does not require internal preference arbitration at the mission-goal level.

A minimal autonomous-execution robot requires a set such as

$Mod^{AE} = \{Mod_{rec}, Mod_{map}, Mod_{con}, Mod_{exe}, Mod_{val}\}$

where:

- Mod_{rec} is the recognition module.
- Mod_{map} maps the recognized command, cue, or assigned goal to an available action.
- Mod_{con} evaluates feasibility, safety, and applicable constraints.
- Mod_{exe} controls action execution.
- Mod_{val} validates the result.

For such a robot,

G_m or C_{ext}
$\rightarrow a* \in L_o$
$\rightarrow$ Execute($a*$)
where G_m is an externally assigned goal and C_{ext} is a recognized external cue or command.

This architecture supports autonomous execution but not command-free goal generation:

$Mod^{GG} \notin Mod(r)$
$\Rightarrow$ Command-Free-Goal-Generation(r) = 0

A fully independent command-free robot requires additional modules:

$Mod^{fI} = Mod^{AE} \cup \{Mod_{state}, Mod^{GG}, Mod_{int}, Mod_{ar\beta}\}$

where:

- Mod_{state} receives and organizes internal and external state reports
- Mod^{GG} generates candidate goals
- Mod_{int} evaluates multidimensional internal segments
- $Mod_{ar\beta}$ integrates decision values and selects among admissible candidates

Command-free goal selection is possible only if the required modules are installed and active:

$\{Mod_{state}, Mod^{GG}, Mod_{int}, Mod_{ar\beta}\} \subseteq Mod(r)$
$\Rightarrow$ Command-Free-Goal-Generation(r) = 1

Even then, the robot remains bounded by its installed goal classes, facilities, competence, constraints, and authority. Command-free operation does not imply that the robot may generate or pursue an arbitrary goal. An innovative robot requires a further module set:

$Mod^{INN} = \{Mod_{spark}, Mod_{search}, Mod_{asm}, Mod_{val}, Mod_{lup}\}$
where:

- Mod_{spark} detects the absence or inadequacy of a complete solution;
- Mod_{search} performs bounded segment retrieval;
- Mod_{asm} performs ε-compatible assembly;
- Mod_{val} evaluates the constructed candidate; and
- Mod_{lup} controls validated library updating.

Innovative construction is available only when
$Mod^{INN} \subseteq Mod(r)$
and
$Authority(Mod^{INN}, G*) = 1$

If these conditions are not satisfied, the absence of an adequate stored action cannot activate unrestricted construction:

$\neg\exists a \in L_o : Adequate(a, G*, \Omega) = 1$
$\wedge\ Mod^{INN} \not\subset Mod(r)$

$\Rightarrow$ OUT $\in$ {NO ACTION, REQUEST ASSISTANCE, REFERRED}

A robot may possess the innovation modules without possessing command-free goal generation. Such a robot can construct a new strategy for an externally assigned goal but cannot independently decide which goal should be pursued:

$Mod^{INN} \subseteq Mod(r)$
$\wedge\ Mod^{GG} \notin Mod(r)$

$\Rightarrow$ Innovation(r) = 1
$\wedge$ Command-Free-Goal-Generation(r) = 0

Conversely, a fully independent robot may generate its own goal while lacking innovative-construction modules:

$Mod^{GG} \in Mod(r)$
$\wedge\ Mod^{INN} \not\subset Mod(r)$

$\Rightarrow$ Command-Free-Goal-Generation(r) = 1

$\wedge$ Innovation(r) = 0

In this case, if no adequate stored action exists for the selected goal, the robot must postpone, reject, or refer the goal rather than construct a new strategy.

The relationships among the illustrative robot types can be summarized as follows:

Table 1: Module Requirements Across Robot Types.

Robot type	External goal or cue	Command-free goal generation	Internal goal arbitration	Innovative construction
Cue-driven ballet robot	Required	Not required	Not used for assigned-goal selection	Optional
Mission-governed robot	Required	Not required at mission level	Excluded from mission redefinition	Optional and authority dependent
Command-free tennis robot	Not required	Required	Required	Optional
Scientist robot	Standing domain authority may be sufficient	Required for problem initiation	Domain-specific	Optional but required for new strategy construction
Innovative specialist robot	Either assigned or self-generated	Optional	Role-dependent	Required

The table describes architectural requirements rather than universal definitions. An individual implementation may contain additional perception, navigation, communication, manipulation, simulation, or domain-specific modules.

The architecture therefore distinguishes four capability levels:

1. **Execution capability:** the robot executes an existing action for an assigned goal or recognized cue.
2. **Selection capability:** the robot chooses among stored actions or strategies.
3. **Independent goal capability:** the robot generates and selects an admissible goal without a present command or assigned mission.
4. **Innovative-construction capability:** the robot constructs and validates a useful strategy that was not stored as a complete solution in L_0.

These capabilities form neither a mandatory sequence nor a universal hierarchy. A robot may be highly competent at mission execution while intentionally lacking independent goal generation. Another robot may independently select daily-life goals but lack permission to construct new actions. A specialist robot may innovate within a narrow scientific or engineering domain while remaining unable to operate outside that domain.

Accordingly,

Autonomy ≠ Command-Free Independence ≠ Innovation.

Autonomy concerns the robot's ability to select or execute actions without continuous external control. Command-free independence concerns its ability to generate and select a goal without a present command. Innovation concerns its ability to construct and validate a useful solution absent from its active library.

The complete robot-specific capability condition may be written as

$Capability_c(r) = 1$
if

$Mod_c \subseteq Mod(r)$
$\wedge$ Competence(r, c) = 1
$\wedge$ Authority(r, c) = 1
$\wedge$ Constraints(r, c, Ω) = 1

where Mod_c is the set of modules required for capability c.

This formulation prevents the term "fully independent robot" from implying unlimited competence or unrestricted authority. Independence is bounded by the robot's installed modules and is evaluated relative to a specified functional domain. Consequently, each autonomous, command-free, or innovative robot requires the modules necessary for its own operational class, but not every robot requires every module contained in the general architecture.

4. Conclusions

This study has distinguished autonomous execution, command-free goal generation, and robotic innovation as separate operational capabilities. An autonomous robot selects or executes actions for an externally assigned goal. A fully independent robot can generate and select an admissible goal when no present command or assigned mission exists. An innovative robot performs a further operation: it constructs and validates a functional strategy that was not stored as a complete solution in its active pre-event library.

The proposed distinction prevents retrieval, self-initiation, and innovation from being treated as equivalent. A self-generated goal may be satisfied by an existing action and therefore involve no innovation. Conversely, a robot may construct an innovative strategy for an externally assigned goal without independently generating that goal. Innovation begins only when no adequate complete solution exists, SPARK activates bounded constructive search, and compatible segments are assembled into a candidate strategy.
The operational novelty condition is $S_n \notin L_0$.

This condition is necessary but not sufficient. A candidate qualifies as an innovation only when it also satisfies the active goal, functional-coherence requirements, feasibility conditions, safety constraints, competence boundaries, and governing authority. Following successful validation, the new strategy is retained through $L_1 = L_0 \cup \{S_n\}$.

Its first validated construction constitutes innovation relative to L_0; its subsequent retrieval or repetition constitutes ordinary autonomous selection.
The four-dimensional context

$$\Omega = (X_r, X_e, T_r, T_e)$$

shows why robotic decisions cannot be represented as static stimulus–response mappings. Internal state, external conditions, robot-relative time, and event-relative time may change independently. Consequently, the same external opportunity may support immediate action, an alternative goal, postponement, or deliberate no-action at different moments.

The tennis case illustrated this dynamic internal arbitration. The availability of a court, partner, equipment, and sufficient time did not require the robot to play immediately. Supporting, opposing, and contextually neutral internal segments were integrated into a signed decision balance. The resulting choice depended on both the current opportunity and the robot's changing multidimensional state.

The ballet and mission-governed cases demonstrated a different operational structure. When a valid cue or authorized mission assigns the goal, internal preference segments do not redefine that goal. The robot may remain autonomous in recognition, planning, action selection, execution, monitoring, and recovery, while safety, feasibility, applicable rules, and governing authority retain veto power. Autonomous execution therefore does not necessarily imply independent goal generation.

The scientist-robot case demonstrated bounded self-initiation within an installed research domain. A scientific investigation may begin without a present task-specific command when the robot recognizes a relevant problem and identifies a sufficient basis in related problems, transferable principles, applicable methods, or validated knowledge. However, initiating research is not itself innovation. Innovation occurs only

if the robot constructs and validates a useful scientific structure that was absent from its active library.

The architecture is modular because different robot types require different capabilities. A cue-driven performer need not contain command-free goal-generation or scientific-reasoning modules. A scientist robot need not possess the physical facilities required for competitive sport. A mission-governed robot may intentionally exclude internal preference arbitration from mission selection. Likewise, a fully independent robot may generate its own goals while lacking the modules or authority required for innovative construction.

A fully independent robot is therefore not a universal machine capable of generating arbitrary goals or operating in every domain. Its independence is bounded by its installed and validated modules, available facilities, represented goal classes, operational competence, contextual constraints, and governing authority. Adding a module may expand the robot's possible behavior, but it does not create competence outside the newly validated domain.

The principal contribution of the proposed framework is a traceable engineering distinction among three questions:

- **What assigned action should the robot execute?**
- **What admissible goal does the robot presently choose to pursue?**
- **What new strategy can the robot construct when no adequate complete solution exists?**

Autonomous execution answers the first question, command-free goal generation answers the second, and the SPARK-based innovation mechanism answers the third.

The present work provides a conceptual and operational architecture rather than a completed physical implementation. Future work should specify domain-dependent matching functions, decision thresholds, segment weights, compatibility-error measures, and validation operators. Experimental evaluation should compare the robot's pre-event and post-event libraries, trace the construction path of S_n, test the reproducibility of the validation result, and distinguish genuine library expansion from optimization or recombination of an already stored complete solution.

The central conclusion is that robotic independence and robotic innovation are technically conceivable without requiring unrestricted autonomy or creation from nothing. Independence is modular, domain-bounded, context-dependent, and authority-constrained. Innovation is the validated construction of a useful new organization relative to the robot's active library. Together, these distinctions provide a testable foundation for moving from autonomous action execution to independent goal selection and bounded strategy generation.

Finally, the robot's multidimensional supporting and opposing internal segments, $D_k(\Omega)$ and $A_k(\Omega)$, may be described intuitively as the internal "angel" and "devil." These terms are metaphors rather than moral classifications: the "angel" represents internal contributions supporting an action, whereas the "devil" represents internal contributions opposing, inhibiting, or redirecting it. Both are evaluated concurrently within the current context Ω. When their deterministic arbitration does not produce a sufficiently decisive result, the bounded stochastic internal segment $R_k(\Omega)$ may permit limited random selection. This mechanism operates only after feasibility, competence, safety, authority, and all other admissibility conditions have been satisfied, and only within the admissible action set $A*(\Omega)$. The internal "angel," internal "devil," and bounded randomness therefore provide an intuitive but formally controlled representation of competing influences in robotic decision-making.

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. Boden, M. A. (2004). The Creative Mind: Myths and Mechanisms, 2nd ed. London, UK: Routledge https://www.routledge.com/The-Creative-Mind-Myths-and-Mechanisms/Boden/p/book/9780415314534.
2. Sarathy, V. and Scheutz, M. (2017). "The MacGyver Test: A framework for evaluating machine resourcefulness and creative problem solving," https://arxiv.org/abs/1704.08350.

3. Sarathy, V. and Scheutz, M. (2018). "MacGyver problems: AI challenges for testing resourcefulness and creativity," Advances in Cognitive Systems 6: 31-44.
4. Nair, L., Balloch, J. and Chernova, S. (2020). "Feature-guided search for creative problem solving through tool construction," Frontiers in Robotics and AI 7: 592382; DOI 10.3389/frobt.2020.592382.
5. Fitzgerald, T., Short, E., Goel, A. K. and Thomaz, A. L. (2021). "Modeling and learning constraints for creative tool use," Frontiers in Robotics and AI 8: 674292; DOI 10.3389/frobt.2021.674292.
6. Lesort, T., Lomonaco, V., Stoian, A., Maltoni, D., and Filliat, D. (2020). "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges," Information Fusion 58: 52-68.
7. Gubenko, A., Kirsch, C., Smilek, J. N., Lubart, T. and Houssemand, C. (2021). "Educational robotics and robot creativity: An interdisciplinary dialogue," Frontiers in Robotics and AI 8: 662030; DOI 10.3389/frobt.2021.662030.
8. Kotseruba, I., and Tsotsos, J. K. (2020). "A review of 40 years of cognitive architecture research: Core cognitive abilities and practical applications," Artificial Intelligence Review 53: 17-94.
9. Molineaux, M., Klenk, M. and Aha, D. W. (2010). "Goal-driven autonomy in a Navy strategy simulation," in Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence 1548-1554.
10. Oudeyer, P. Y., Kaplan,F. and Hafner, V. V. (2007)."Intrinsic"motivation systems for autonomous mental development," IEEE Transactions on Evolutionary Computation, vol. 11: 265-286.
11. Arkin, R. C. (2009). Governing Lethal Behavior in Autonomous Robots. Boca Raton, FL, USA: CRC Press https://www.taylorfrancis.com/books/mono/10.1201/9781420085952/governing-lethal-behavior-autonomous-robots-ronald-arkin.