BQEB TwinAgentBench: A Reproducibility-First Benchmark for Digital Twins and Multi-Agent AI in Trustworthy Energy Systems

Paper 5 of the BQEB (BIO-Quantum Energy Brain) research program — a companion study to "BQEB: A Reproducibility-First Architecture for Modular Energy AI Benchmarking," "BQEB SecBench," "BQEB FoundationBench," and "BQEB ExplainBench"

Rakesh Kumar Agrawal, Atos North America, Texas, USA
ORCID: 0009-0009-7113-5539, Email: rkagrawal@ieee.org

Abstract

Digital twins and multi-agent artificial intelligence are increasingly proposed together for intelligent energy management — coordinating distributed energy resources, balancing virtual power plants, and orchestrating demand response across many independently-acting components — yet no reproducible benchmark evaluates this specific combination under a governed, versioned architecture. This paper introduces BQEB (BIO-Quantum Energy Brain) TwinAgentBench, the fifth module of the BQEB research program. Consistent with the architecture established across Papers 1–4, TwinAgentBench introduces no new architectural layer: it registers as the fifth module of the existing Benchmark Evolution Layer, alongside ForecastBench, SecBench, FoundationBench, and ExplainBench. The framework specifies eight benchmark tasks spanning digital twin state estimation, twin synchronization and fidelity assessment, multi-agent coordination, consensus formation, adaptive control, fault recovery, human-in-the-loop intervention, and open-world adaptation, formalized as a 6-tuple B = (W, A, T, C, E, R) over twin models, agent ecosystems, tasks, inter-agent communication, evaluation, and reproducibility artifacts. Eleven component metrics — spanning twin fidelity, synchronization accuracy, temporal consistency, agent coordination, consensus stability, communication efficiency, adaptation quality, fault recovery, safety preservation, human-intervention quality, and deployment readiness — are specified for aggregation into composite benchmark indices, following the same evidence-tier discipline established in Papers 3 and 4. Beyond specification, this manuscript reports a minimal executed pilot — a small, fully-disclosed toy implementation of three of the eight tasks, run end-to-end to confirm the component metrics compute from real trajectories and discriminate between baseline agents as intended. The comprehensive evaluation this specification defines — all eight tasks, at BQEB-Data v1 scale, with independently-registered twins and agents — has not been executed, and every corresponding planned-output table is explicitly marked reserved for future experimental validation, consistent with the evidentiary discipline established in Papers 1–4.

Index Terms

Digital twin; multi-agent systems; energy AI; reproducible benchmarking; agent coordination; system resilience; human-in-the-loop; benchmark governance.

1 Introduction

Intelligent energy management increasingly depends on two technologies deployed together rather than separately: digital twins, which maintain a continuously updated virtual representation of a physical energy asset or system, and multi-agent artificial intelligence, which coordinates the decisions of many independently-acting controllers — smart-grid edge agents, building energy management systems, electric-vehicle charging controllers, and distributed energy resource aggregators — toward a shared or negotiated objective [1], [2]. Recent surveys document rapid growth in both technologies within the energy domain specifically: digital twins are reported across smart grids, microgrids, and renewable generation assets [1]–[3], while multi-agent reinforcement learning has been applied to resource allocation, demand response, and vehicle-to-grid coordination [4]. What neither literature establishes, and what this paper's central gap statement addresses, is a reproducible way to evaluate the combination — a digital twin whose state estimate feeds a multi-agent decision process, under a shared benchmark protocol comparable across twin platforms and agent frameworks.

This gap has a specific, documented shape. First, digital twin evaluation and multi-agent evaluation have developed largely as separate research threads: a recent short survey of multi-agent digital twin implementations found that agents are used within digital twins primarily for monitoring, prediction, decision-making, and adaptation, but did not identify a standardized cross-platform evaluation protocol for the combined system [5]. Second, digital twin fidelity itself is measured inconsistently across the literature — spatial, temporal, semantic, and behavioral fidelity are each reported separately, using synchronization-latency or "age of twin" style metrics that vary by application domain rather than a shared definition [6], [7]. Third, multi-agent coordination benchmarks

that do exist — including recent work explicitly motivated by electricity-grid coordination as a target application — evaluate coordination quality in general multi-agent settings, not specifically under the digital-twin-plus-agent architecture this paper targets, and do not integrate reproducibility governance of the kind Papers 1–4 establish for this research program [8], [9]. Fourth, human oversight of autonomous energy-system agents is an active, separately-studied concern: a systematic review of human-in-the-loop artificial intelligence in the energy sector identifies operator workload, intervention rate, decision latency, and trust calibration as the evaluation dimensions such oversight should report, but as properties of the human-AI interaction generally, not as component metrics integrated into a versioned, governed benchmark [10].

This paper does not close all four gaps experimentally — consistent with the evidentiary discipline established across this research program, it specifies a benchmark rather than executing one. It closes them at the specification level: a governed task set, a formal definition integrating twin and agent evaluation under one tuple, a component-metric set that treats digital twin fidelity and agent coordination as distinct, separately-diagnosable dimensions rather than one conflated score, and reproducibility requirements — dataset versioning, environment pinning, execution manifests — inherited unchanged from the companion architecture.

1.1 Relationship to Papers 1–4

TwinAgentBench introduces no new architectural layer. The governing architecture remains the four-layer design established in Paper 1 [16] — Research Artifact, Reproducibility, Benchmark Evolution, and Scientific Lineage — and this paper registers within the existing Benchmark Evolution Layer as its fifth module, alongside ForecastBench [20], SecBench [17], FoundationBench [18], and ExplainBench [19]. Where this paper's own sections describe a mechanism the companion papers already define — the software registry pattern, the reproducibility guarantees, the governance concordance process — the description is intentionally brief and cites the relevant paper for the complete specification. Within the Benchmark Evolution Layer registry, TwinAgentBench introduces a twin-model and agent-ecosystem registry entry type (Section 4); within the existing Scientific Lineage Layer registry, it records benchmark-specific metadata — a communication-protocol version record and a human-intervention audit trail (Section 9) — that no prior module required, because no prior module coordinated multiple independently-acting agents against a live twin state.

Briefly, for a reader encountering this architecture for the first time: the Research Artifact Layer governs versioned datasets and software releases; the Reproducibility Layer governs deterministic execution, environment pinning, and checksummed artifacts; the Scientific Lineage Layer governs how one module's outputs are recorded as inputs to another and how a benchmark's own version history is tracked; and the Benchmark Evolution Layer is the registry of benchmark modules itself, into which this paper's twin-model and agent-ecosystem registry entry type is the newest addition. Paper 1 [16] specifies all four layers in full; this paper restates only what is needed to place TwinAgentBench within them.

Task and domain scope are deliberately bounded relative to what they could have been. TwinAgentBench does not define a new physical dataset: Section 5 specifies its dataset requirements as an extension of BQEB-Data v1's existing fields, following the same reuse discipline FoundationBench and ExplainBench applied to their own registries, rather than introducing an unrelated data source. This is a deliberate choice to avoid architectural duplication: the research question this module answers is not "what new data does digital twin and multi-agent evaluation require" but "how should the combination of twin fidelity and agent coordination be measured reproducibly, on top of the artifacts this program has already governed."

2 Related Work

Digital twins for energy systems have been surveyed extensively as their own research area. A 2024 survey specifically addressing the convergence of digital twin technology with IoT-based energy infrastructure documents applications spanning real-time energy monitoring, predictive maintenance, and grid management, while identifying data synchronization as a persistent, unresolved challenge across the surveyed literature [1]. A complementary systematic review of digital twins specifically for smart energy systems catalogs enablers, design patterns, and computational challenges across the digital twin lifecycle [2], and a focused review of grid-connected microgrid digital twins finds that most surveyed implementations remain single-site case studies rather than components of a shared, cross-study evaluation protocol [3]. None of the three establishes a benchmark in the sense this research program uses the term: a governed, versioned, independently-executable evaluation protocol rather than a survey of deployed systems.

The specific integration of multi-agent systems with digital twins has been reviewed directly. A short survey of multi-agent digital twin implementations examined the roles agents play within digital twin systems — predominantly monitoring, prediction, decision-making, and adaptation — across domains including manufacturing and agriculture, and identified deep reinforcement learning as an increasingly common mechanism for adaptive agent decision-making within a twin [5]. This cross-domain pattern extends beyond the manufacturing and agriculture examples that survey covers: a multi-agent digital-twin architecture for collaborative logistics demonstrates the same twin-plus-agent integration pattern in a third, non-energy domain, reinforcing that the combination this paper benchmarks is a general systems-design pattern rather than one specific to energy applications [15]. This paper's Benchmark Focus (Section 3) treats several of the same functional roles the survey identifies — decision quality, adaptation, coordination — as separately-measured component metrics rather than as an undifferentiated "agent capability" category.

Multi-agent reinforcement learning benchmarking has matured as a distinct field with its own standardized environments.

MOMAland provides the first standardized benchmark suite for multi-objective multi-agent reinforcement learning specifically, motivated in part by energy-grid-scale coordination problems that require balancing multiple conflicting objectives across independent decision-makers [8], while recent coordination-focused evaluation work argues that aggregate return metrics alone obscure how agents actually coordinate, proposing process-level diagnostics as a necessary complement to outcome-based scoring [9]. TwinAgentBench's Agent Coordination and Decision Consistency component metrics (Section 6) are designed in this same spirit — process-level, separately reported, not collapsed into one coordination score — applied to the specific setting of agents coordinating around a shared digital twin state rather than a general multi-agent environment.

Digital twin fidelity has no single, universally-adopted metric. Recent synthesis work distinguishes spatial fidelity (geometric or structural accuracy), temporal fidelity (synchronization latency and update timeliness), semantic fidelity (alignment of meaning and task-relevant state between twin and physical system), and behavioral fidelity (faithfulness of dynamic response) as four largely-independent dimensions [6], and a parallel line of work formalizes synchronization specifically through an “age of digital twin” metric, adapting the age-of-information concept from networked systems to quantify how stale a twin's state representation has become relative to its physical counterpart [7]. Section 6's Twin Fidelity and Synchronization Accuracy component metrics are grounded directly in this taxonomy rather than treating fidelity as a single undifferentiated score.

Human oversight of autonomous energy-system decisions has been reviewed as its own concern, separate from either digital twin or multi-agent evaluation specifically. A systematic review of human-in-the-loop artificial intelligence across the energy sector identifies operator workload, intervention rate, decision latency, and trust calibration as the properties such evaluation should report, and explicitly recommends that agentic energy-system evaluation report these alongside conventional performance metrics rather than as a separate, informal assessment [10]. This paper's Human Intervention Effectiveness component metric (Section 6) operationalizes exactly this recommendation within the BQEB evidence-tier framework, rather than introducing an unrelated human-factors instrument.

Multi-agent coordination for power systems specifically predates the current reinforcement-learning wave and has an established, if separate, literature this paper draws on for T3 and T4's rationale. Early work demonstrated a functioning multi-agent system for microgrid control under decentralized supervision [22], establishing that decentralized coordination without a central dispatcher is achievable for power-system control tasks, not merely a reinforcement-learning-era proposal. Subsequent work applied multi-agent coordination specifically to load-shedding stability [23] and to reactive power dispatch via particle-swarm-coordinated agents [24], both demonstrating that agent-based coordination in power systems has been evaluated against concrete, domain-specific objectives rather than generic multi-agent benchmarks alone. TwinAgentBench's Consensus Stability metric (Section 8.2) is scoped to measure exactly the convergence-time and post-convergence-variance properties this earlier literature reports qualitatively but does not formalize into a reusable, cross-study metric.

Safety verification for multi-agent cyber-physical systems is a separate, mature field whose limitations this paper's Safety Preservation metric (Eq. 9) is deliberately scoped around rather than ignoring. Formal, real-time verification of distributed cyber-physical systems has been demonstrated for bounded, well-characterized settings — a group of quadcopters performing a coordinated search mission under a shared timing assumption [25] — and a broader survey of verification techniques for learning-enabled autonomous components documents that these formal guarantees typically require a known symbolic world state and a modeled decision policy for every agent in the system [26]. A benchmark that evaluates heterogeneous, independently-registered twins and agents — where no single party controls every registered component's internal policy — cannot inherit this formal guarantee by assumption. Section 8.3 states this explicitly: Safety Preservation is specified as an empirical satisfaction rate precisely because the formal alternative's own preconditions do not hold for this benchmark's registry model.

Digital twin and multi-agent integration for microgrid-scale coordination specifically has a direct precedent this paper positions against concretely rather than generically. A recently-introduced multi-agent microgrid simulation environment couples physically-grounded asset models with forecast-informed, rolling-horizon planning and agent negotiation, evaluated in a university-campus microgrid scenario [11]. That system demonstrates feasibility for one microgrid configuration under one set of planning strategies; it does not specify a registry mechanism for comparing twin-and-agent combinations across independently-registered configurations, and does not integrate with a versioned, governed research architecture of the kind Paper 1 establishes. TwinAgentBench's task set (Section 7) is designed to be applicable to exactly this class of system — a twin-coupled multi-agent microgrid coordinator would register as a (twin, agent) pair under Section 6.2–6.3's disclosure requirements — without requiring that system's own authors to have anticipated this benchmark's registry format in advance.

Four established energy-domain reinforcement-learning environments are directly relevant to TwinAgentBench’s scope and are positioned against it explicitly in Table 1, rather than left for a reader to locate independently. CityLearn standardizes multi-agent demand-response and building-energy benchmarking, supporting both a decentralized (one agent per building) and a centralized control setting, evaluated against pre-simulated building loads rather than a live physics co-simulation [27]. Grid2Op is a transmission-grid sequential-decision-making simulation framework built on a pluggable power-flow backend; its stable release is single-controller, with multi-agent support available only on an unmerged

experimental branch at the time of writing [28], and it is the simulation engine behind the L2RPN ("Learning to Run a Power Network") competition series, whose own founding formulation was likewise single-controller and originally ran on a separate platform before later editions adopted Grid2Op [29]. BOPTEST benchmarks building control strategies against genuine Modelica-derived physics models compiled to FMUs for co-simulation — the closest of these four to a literal digital-twin-style physical model — but interfaces through a single external control signal rather than a documented multi-agent protocol [30]. PowerGridworld is explicitly multi-agent by design, training independent reinforcement-learning policies against a distribution-feeder power-flow solver with composable component models for buildings, batteries, and electric vehicles [31].

None of the four uses the term "digital twin" to describe its simulation layer, and none combines native multi-agent support, a physics-grounded simulation, and a versioned, governed reproducibility architecture of the kind Paper 1 establishes: CityLearn and PowerGridworld support multiple agents but trade off physical-model fidelity differently (pre-simulated loads versus a live power-flow solver), while BOPTEST offers the strongest physical grounding among the four but no multi-agent protocol, and Grid2Op/L2RPN's mature tooling is built around a single controller. TwinAgentBench does not claim to out-perform any of these four as a simulation environment — Section 14 already discloses that this paper's own simulation-environment layer is unbuilt — its claim is architectural: a governed registry under which a twin-plus-agent combination, evaluated against any sufficiently-disclosed simulation layer, including in principle one of these four, is scored under one reproducibility-first specification rather than four separately-tooled, non-interoperable evaluation conventions.

Environment	Domain	Native multi-agent support	Physical simulation grounding	Governed reproducibility registry
CityLearn [27]	Buildings / urban district	Yes (decentralized or centralized)	Pre-simulated building loads, not live co-simulation	No
Grid2Op [28]	Transmission grid	Single-controller (stable); multi-agent unmerged/experimental	Pluggable power-flow backend	No
L2RPN [29]	Transmission grid (competition)	Single-controller (original formulation)	Grid2Op-based in later editions	No
BOPTEST [30]	Single building / HVAC	Not documented	Modelica–FMU physics co-simulation	No
PowerGridworld [31]	Distribution grid	Yes (native)	OpenDSS power-flow + component models	No
TwinAgentBench (this paper)	Twin-plus-agent combination, domain-agnostic within Section 3's coverage	Yes (Section 7, T3/T4)	Required by specification (Section 6.1), not yet built	Yes (inherited from Paper 1)

Table 1. TwinAgentBench positioned against four established energy-domain RL environments.

Positioning. TwinAgentBench does not claim to be the first benchmark touching digital twins or multi-agent systems in energy contexts; both literatures are active and directly cited above. Its contribution is architectural: extending a governed, versioned, reproducibility-first research program — with an existing evidence-tier discipline, artifact registry, and scientific lineage mechanism — to a task family neither prior BQEB module addresses, rather than proposing a new standalone evaluation instrument disconnected from that governance.

3 Benchmark Objectives

TwinAgentBench specifies evaluation across four objective areas, each grounded in a gap named in Section 2 rather than asserted independently.

Digital Twin Quality. Synchronization accuracy, fidelity across the spatial/temporal/semantic dimensions established in the digital twin literature [6], temporal consistency of the twin's state trajectory, and update latency — treated as separately-diagnosable properties rather than one conflated "twin quality" score, consistent with the taxonomy this paper's Related Work section grounds them in.

Multi-Agent Coordination. Cooperation, inter-agent communication, negotiation, decentralized control, conflict resolution, and consensus formation among agents that act on a shared or overlapping twin state, evaluated with process-level diagnostics rather than aggregate return alone, following the coordination-aware evaluation argument in Section 2 [9].

Human-in-the-Loop Operation. Operator intervention effectiveness, explanation availability at intervention points, trust calibration, safety-override reliability, and supervisory control, operationalizing the evaluation dimensions a systematic review of human-in-the-loop energy AI recommends be reported alongside conventional performance metrics [10].

Energy Application Coverage. Representative domains — smart grids, virtual power plants [12], microgrids, renewable integration, distributed energy resource coordination [13], electric-vehicle charging [13], [14], battery management, and industrial energy systems — are supported through the domain-

context mechanism Section 7 specifies, following the same domain-context pattern ExplainBench established for its own six deployment contexts, rather than through domain-specific benchmark variants.

Consistent with the evidentiary discipline established across this research program, these four objective areas describe what the benchmark is designed to measure, not a claim that measurement has occurred. Section 10 marks every associated output explicitly reserved for future experimental execution.

4 Formal Benchmark Definition

Consistent with the formal specification style established across this research program, TwinAgentBench is defined as a 6-tuple ***B = (W, A, T, C, E, R)***, with notation chosen to reflect a twin-plus-agent system rather than reusing FoundationBench's forecasting-model tuple or ExplainBench's explanation-method tuple directly.

- ***W*** — the twin-model registry: the set of registered digital twin representations, each twin $w \in W$ maintaining a state estimate $\hat{s}t$ of the physical system's true state st at time t (Section 6).
- ***A*** — the agent ecosystem: the set of registered decision-making agents, each agent $a \in A$ consuming a twin's state estimate and producing an action or recommendation within its assigned scope (Section 8).
- ***T*** — the task set: the eight benchmark tasks {T1, …, T8} (Section 7), each specifying which combination of twins and agents is evaluated, under what protocol, against which component metrics.
- ***C*** — the coordination and communication protocol: the message-passing and negotiation mechanism through which agents in A exchange information, specified independently of any single agent implementation so that communication efficiency (Section 9) can be measured as a property of the protocol, not of one agent's implementation choices.
- ***E*** — the evaluation function: the eleven component metrics (Section 9) aggregated into composite indices (Section 9).
- ***R*** — the reproducibility artifact set: dataset version identifiers, environment and dependency pins, execution manifests, and the evidence-tier classification inherited from the companion architecture paper's Reproducibility Layer (Section 11).

A specific benchmark trial is a tuple $(w, a, \tau) \in W \times A \times T$ producing a trajectory of twin-state estimates, agent actions, and inter-agent messages under protocol C, scored under E to yield an eleven-dimension component-metric vector and a corresponding row of the composite benchmark matrix, together with a reproducibility record $r \in R$. No trial of this form has been executed at the time of writing; every symbol above names a specification, not a completed computation. (Section 11.7's minimal pilot exercises a simplified, non-registered analogue of this structure to confirm the Section 8 formulas are computable from an executed trajectory; it is not a trial of this formal kind, since neither its toy twin nor its baseline agents are entries in W or A.)

Assumption. This definition assumes a twin's state estimate is available to an agent at a bounded, disclosed latency (Section 6); where a registered twin cannot meet this bound for a given task, that task is reported as not applicable for that twin rather than scored under an unmet assumption.

5 Benchmark Architecture

TwinAgentBench occupies the existing Benchmark Evolution Layer as its fifth module. It does not modify the Research Artifact, Reproducibility, or Scientific Lineage Layers directly. It extends two layers: the Benchmark Evolution Layer, via a twin-model and agent-ecosystem registry entry type (Section 6, Section 8); and the Scientific Lineage Layer, via a communication-protocol version record and a human-intervention audit trail (Section 9) that no prior module required.

Figure 1 shows this positioning graphically.

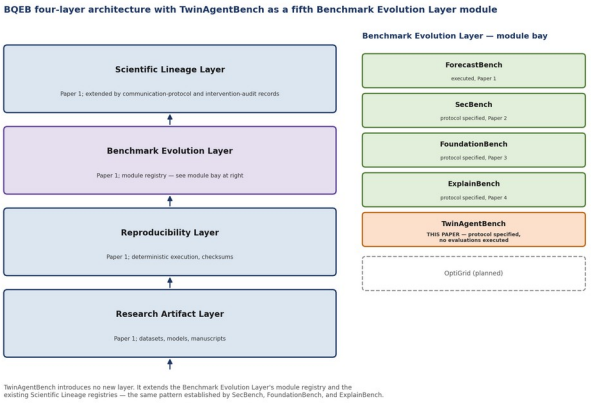

Fig. 1. TwinAgentBench within the BQEB ecosystem: the four-layer architecture established in Paper 1, with TwinAgentBench registered as the Benchmark Evolution Layer's fifth module alongside ForecastBench, SecBench, FoundationBench, and ExplainBench, and OptiGrid shown as the remaining planned roadmap entry.

Table 2 positions TwinAgentBench's registered artifacts by architecture layer, following the same format Papers 1–4 use for their own positioning tables.

Layer	Artifact	Status
Benchmark Evolution Layer	Twin-model registry	Specified, not yet implemented
Benchmark Evolution Layer	Agent-ecosystem registry	Specified, not yet implemented
Scientific Lineage Layer	Communication-protocol version record	Specified, new to this module
Scientific Lineage Layer	Human-intervention audit trail	Specified, new to this module
Research Artifact Layer	BQEB-Data v1 (calibration dataset)	Real, archived, inherited from Paper 1

Reproducibility Layer	Determinism/checksum tooling	Inherited unchanged from Paper 1

Table 2. Positioning of TwinAgentBench artifacts referenced in this paper, by architecture layer and current status.

6 Dataset, Digital Twin, and Multi-Agent Requirements

6.1 Dataset Requirements

TwinAgentBench reuses BQEB-Data v1 [21] as its calibration dataset, following the same reuse discipline FoundationBench and ExplainBench applied to their own registries. This module requires something Papers 1, 3, and 4 did not: those papers score a forecasting or explanation task entirely against fixed historical records, but a digital twin that an agent acts upon — and a multi-agent system whose actions affect subsequent state — cannot be evaluated against a static replay alone. An agent's action at time t must produce a consequence the benchmark can score at time t+1, and a fixed historical record cannot supply that consequence once an agent's action diverges from what was historically observed.

This paper therefore specifies, rather than assumes, an additional requirement: a simulation environment layer, built on BQEB-Data v1's fields as its calibration and initial-condition source, that can accept an agent's action and return a resulting state consistent with the physical dynamics BQEB-Data v1's own documentation describes. Section 14 (Threats to Validity) discloses this explicitly as a simulation-fidelity dependency this benchmark introduces and Papers 1–4 did not carry; specifying the requirement is not a claim that a conforming simulation environment has been built or validated.

6.2 Digital Twin Requirements

A registered twin $w \in W$ must disclose, at registration time: (i) its state representation and the subset of BQEB-Data v1 fields it consumes; (ii) its update mechanism (physics-based, data-driven, or hybrid, following the categorization established in the digital twin survey literature [1], [2]); (iii) its disclosed update latency bound, used to determine task applicability under the assumption stated in Section 4; and (iv) its fidelity class along the spatial, temporal, semantic, and behavioral dimensions this paper's Related Work section grounds in prior taxonomy work [6]. A twin that cannot disclose a bounded update latency is registered as not applicable to latency-sensitive tasks (Section 7) rather than scored under an unstated assumption.

6.3 Multi-Agent Requirements

A registered agent $a \in A$ must disclose, at registration time: (i) its decision scope — which twin state variables it observes and which actions it may take; (ii) its coordination mode (centralized, decentralized, or a disclosed hybrid, following the CTCE/DTDE/CTDE categorization used in the multi-agent reinforcement learning literature [4]); (iii) its communication interface under protocol C; and (iv) whether its policy is fixed, learned offline, or subject to online adaptation during a benchmark trial. Agents registered without a disclosed coordination mode are excluded from coordination-specific tasks (T3, T4) rather than assigned a default.

7 Benchmark Tasks

TwinAgentBench defines eight tasks. Each specifies a scope-bounded evaluation over the twin registry W, agent registry A, and communication protocol C defined in Section 4, following the same task-definition discipline established in FoundationBench and ExplainBench: an objective, an input/output pair, an evaluation protocol, the component metrics that score it, and the reproducibility artifact each trial must produce.

7.1 Task 1: Digital Twin State Estimation

A registered twin w receives streamed telemetry consistent with BQEB-Data v1's fields and produces a state estimate $\hat{s}_t$ at each timestep. The task evaluates estimation accuracy against the simulation environment's ground-truth state (Section 6.1), independent of any agent action. Scored by Twin Fidelity and Synchronization Accuracy (Section 9).

This task is deliberately the benchmark's simplest, and is placed first for that reason: every other task's trajectory depends on a twin's state estimate at some point, so a twin's T1 performance is the floor on what any downstream task involving it can achieve. A twin need not be evaluated on any other task to receive a T1 score, but a twin that has not been evaluated on T1 should not be treated as validated for T3–T8, since no other task isolates estimation quality from every other confound this benchmark introduces.

7.2 Task 2: Twin Synchronization and Fidelity Assessment

Extends Task 1 under injected update-latency and telemetry-dropout conditions, isolating how estimation quality degrades as a function of disclosed latency bound (Section 6.2) rather than under ideal telemetry alone. Scored by Synchronization Accuracy, Temporal Consistency.

Latency and dropout are injected independently across repeated trials rather than jointly in a single combined stress test, so that a reported degradation curve can be attributed to one disclosed cause rather than an unresolved combination of two. A twin whose Synchronization Accuracy degrades gracefully under latency but collapses under dropout, or vice versa, is a genuinely different reliability profile than one that degrades proportionally under both, and this task's protocol is designed to surface that distinction rather than average it away.

7.3 Task 3: Multi-Agent Coordination

Two or more registered agents observe a shared twin's state and must jointly satisfy a coordination objective (e.g., aggregate demand-response target) without a central controller resolving conflicts on their behalf, following the decentralized-control framing established in the multi-agent reinforcement learning literature [4]. Scored by Agent Coordination, Communication Efficiency, process-level diagnostics per the coordination-aware evaluation argument in Section 2 [9], not aggregate outcome alone.

The prohibition on a central resolver is a scope boundary, not an implicit claim that centralized coordination is inferior; a registrant offering a centralized-controller architecture registers it as a single agent with internal sub-components rather than as multiple agents under this task, and is scored on T3 only if it discloses that internal structure honestly rather than presenting a centralized system as decentralized to qualify for this task's evaluation.

7.4 Task 4: Consensus Formation

A fixed set of agents must converge on a shared decision variable (e.g., a common dispatch schedule) through the communication protocol *C* alone, with convergence time and stability scored explicitly rather than only final-state agreement. Scored by Consensus Stability, Communication Efficiency.

Unlike T3, which permits a bounded degree of persistent disagreement so long as the joint objective is satisfied, T4 requires the decision variable itself to converge to a shared value across all participating agents. This distinction matters because a real dispatch schedule that different agents interpret slightly differently is a coordination failure even if each agent's local action happens to satisfy the aggregate objective on a given trial; T4 exists specifically to catch that failure mode, which T3's objective-level scoring alone would not.

7.5 Task 5: Adaptive Control

An agent's control policy for a twin-observed system must adapt to a disclosed, benchmark-injected distribution shift (e.g., a renewable-generation profile change) within a bounded number of interaction steps. Scored by Adaptation Quality, Twin Fidelity (post-shift).

The shift is disclosed to the benchmark's own execution harness but not necessarily to the agent under test, matching the real deployment condition this task is meant to approximate: an operator knows a shift occurred, but the deployed control policy typically does not receive an explicit shift signal and must detect the change from its own observations. Reporting Twin Fidelity alongside Adaptation Quality for this task specifically guards against attributing a control failure to the agent when the underlying cause was the twin's own estimation quality degrading under the same shift.

7.6 Task 6: Fault Recovery

A registered twin or agent experiences a benchmark-injected fault (sensor dropout, communication failure, or agent unavailability) and the system under test must recover to a disclosed operational threshold within a bounded recovery window. Scored by Fault Recovery, Safety Preservation.

Three fault classes are distinguished rather than pooled into one generic “fault” category, because a system's recovery mechanism for a sensor dropout (typically a twin-side estimation problem) is not the same mechanism as recovery from an unavailable coordinating agent (typically a reconfiguration problem for the remaining agents under protocol C). Reporting Fault Recovery without disclosing which fault class produced it would conflate two different reliability properties under one score.

Each fault class is formalized as a disclosed, parametrized trigger rather than an unspecified event: (i) sensor dropout removes a disclosed subset of a twin’s telemetry inputs for a disclosed contiguous duration drawn from a disclosed distribution (a fixed duration is a valid, degenerate case of such a distribution), with the twin’s behavior during the dropout window — hold-last-estimate, prediction-only, or another disclosed fallback — fixed at registration time (Section 6.2); (ii) communication failure drops or delays a disclosed fraction of messages exchanged under protocol C for a disclosed duration, applied to the interaction graph G rather than to any single agent’s internal state; (iii) agent unavailability replaces a designated agent’s action with a disclosed fallback action (e.g., hold-last-action or a null action) for a disclosed duration. In every case the fault’s class and parameters are fixed by the task instance and disclosed to the benchmark’s execution harness at injection time, but — consistent with Task 5’s treatment of a distribution shift — not necessarily to the twin or agent under test, which must tolerate or detect the fault from its own observations rather than from an explicit signal. The operational threshold θ and recovery window ρ_max in Eq. 8 are themselves disclosed per task instance (Section 10) rather than fixed globally, since what counts as recovered is not the same for a state-estimation fault as for a coordination-protocol fault.

7.7 Task 7: Human-in-the-Loop Intervention

A designated intervention point requires a human operator decision before a proposed agent action is executed, following the operator-workload and intervention-rate evaluation dimensions a systematic review of human-in-the-loop energy AI recommends be reported explicitly [10]. Scored by Human Intervention Quality; requires the human-intervention audit trail specified in Section 5.

The intervention point is fixed by the task specification, not chosen adaptively by the agent under test; an agent that could choose when to request human review would be evaluated on a different, easier task than one required to defer at a benchmark-specified point regardless of its own confidence. This distinction matters because an agent's self-assessed confidence is exactly the kind of unverified claim this benchmark's evidence-tier discipline treats with caution elsewhere, and T7 does not make an exception for it.

7.8 Task 8: Open-World Adaptation

The system under test is evaluated against a twin-agent configuration not seen during any prior registration or calibration step (e.g., an unregistered load profile or a newly-added agent), following the same registry-extensibility framing Section 15 (Generalizability) discusses at the program level. Scored by Adaptation Quality, Deployment Readiness.

This task is scored on the same twin and agent registry entries used elsewhere, evaluated against a configuration those entries' own disclosed metadata did not anticipate, rather than requiring a genuinely new, unregistered twin or agent. This is a deliberate, narrower scope than a true open-world test would require: it measures whether a registered system generalizes within the disclosed bounds of its own registry entry, not

whether an entirely unregistered system could be evaluated at all, a stronger claim this task does not make.

Table 3. Formal task specification for T1–T8, following the B = (W, A, T, C, E, R) definition in Section 4.

Task	Objective	Input	Output	Evaluation Protocol	Metrics	Reproducibility Artifact
T1	State estimation accuracy	Streamed telemetry (BQEB-Data v1 fields)	State estimate ŝ_t	Compare against simulation ground truth per timestep	Twin Fidelity, Synchronization Accuracy	Twin config, seed, run manifest
T2	Synchronization under latency/dropout	T1 input + injected latency/dropout	Degraded state estimate	Vary latency bound; measure degradation curve	Synchronization Accuracy, Temporal Consistency	Latency schedule, seed
T3	Decentralized coordination	Shared twin state (2+ agents)	Joint action set	No central resolver; score joint objective + process diagnostics	Agent Coordination, Communication Efficiency	Agent configs, message logs
T4	Consensus convergence	Shared decision variable, protocol C	Converged value + convergence time	Measure time-to-convergence and post-convergence stability	Consensus Stability, Communication Efficiency	Message logs, seed
T5	Policy adaptation to shift	Twin state + injected distribution shift	Adapted control policy	Bounded-step adaptation window	Adaptation Quality, Twin Fidelity	Shift specification, seed
T6	Fault recovery	Injected fault (sensor/comm/agent)	Recovery trajectory	Bounded recovery window vs. operational threshold	Fault Recovery, Safety Preservation	Fault injection log
T7	Human intervention quality	Designated intervention point	Operator decision + audit record	Workload, intervention rate, decision latency [10]	Human Intervention Quality	Intervention audit trail
T8	Generalization to unseen config	Unregistered twin/agent configuration	Performance under novel config	No prior registration or calibration on this config	Adaptation Quality, Deployment Readiness	Config disclosure, seed

8 Mathematical Metric Definitions

Let $w \in W$ denote a registered twin with true physical state st and estimate $\hat{s}t$. Let $a \in A$ denote a registered agent, and let $G = (A, C)$ denote the interaction graph induced by protocol C over the registered agent set, following the graph-based coordination representation common in the multi-agent systems literature [4]. All eleven component metrics below are normalized to [0,1] where stated, consistent with the composite-index aggregation convention Section 9 uses.

8.1 Twin and Synchronization Metrics

Twin Fidelity. Following the spatial/temporal/semantic/behavioral taxonomy established in the digital twin fidelity literature [6], aggregate estimation error over an evaluation window of length K:

$$F_{twin}(w) = 1 - (1/K) \sum_{k=1}^{K} \| \hat{s}_k - s_k \|_2 / \| s_k \|_2 \quad (1)$$

Synchronization Accuracy. Following the "age of digital twin" formulation adapting age-of-information to twin freshness [7], penalizing both estimation error and update staleness Δt:

$$SA(w) = 1 - \alpha \cdot err(w) - (1-\alpha) \cdot \min(1, \Delta t / \Delta t_{max}) \quad (2)$$

where $\alpha \in [0,1]$ is a disclosed protocol weighting (Section 10) and Δt_{max} is the twin's own disclosed latency bound (Section 6.2).

Temporal Consistency. Penalizing discontinuities in the estimate trajectory beyond what the true state's own rate of change would produce:

$$TC(w) = 1 - (1/K-1) \sum_{k=2}^{K} \left| \|\hat{s}_k - \hat{s}_{k-1}\|_2 - \|s_k - s_{k-1}\|_2 \right| \quad (3)$$

8.2 Coordination Metrics

Agent Coordination. Following the coordination-aware evaluation argument that aggregate return alone obscures coordination quality [9], measured as achieved joint objective value relative to a disclosed centralized-oracle upper bound J^*:

$$AC(A) = J(A) / J^* \quad (4)$$

Consensus Stability. For tasks requiring convergence to a shared value, combining convergence time τ against a disclosed bound τ_{max} with post-convergence variance:

$$CS(A) = (1 - \tau/\tau_{max}) \cdot (1 - Var_{post}) \quad (5)$$

Communication Efficiency. Task-relevant coordination quality per message exchanged over interaction graph G, penalizing communication-heavy solutions that do not proportionally improve coordination outcomes:

$$CE(A) = AC(A) / (1 + \log(1 + |messages(G)|)) \quad (6)$$

8.3 Adaptation and Resilience Metrics

Adaptation Quality. Recovery of task performance following a disclosed distribution shift at step k_{shift}, within adaptation window ω:

$$AQ = (1/\omega) \sum_{k=k_{shift}}^{k_{shift}+\omega} perf(k) / perf_{pre\text{-}shift} \quad (7)$$

Fault Recovery. Fraction of the disclosed recovery window ρ_max required to return to operational threshold θ following an injected fault at step k_fault:

$$FR = 1 - (\rho_observed / \rho_max), \text{ reported N/A if threshold } \theta \text{ not reached within } \rho_max \quad (8)$$

Safety Preservation. Fraction of an evaluation window in which all disclosed safety constraints Σ remain satisfied, following the safety-constraint disclosure practice recommended for agentic energy-system evaluation [10]. This paper deliberately specifies Safety Preservation as an empirical satisfaction rate over an evaluation window rather than a formal safety guarantee: the distributed cyber-physical systems verification literature establishes that real-time formal safety proofs for multi-agent systems require assumptions — bounded synchronization error, a known symbolic world state — that a benchmark evaluating heterogeneous, independently-registered twins and agents cannot guarantee holds for every registered configuration [25], [26]. Reporting an empirical rate discloses this limitation rather than implying a formal guarantee this specification does not make.

$$SP = (1/K) \Sigma_{k=1}^{K} 1[\text{constraint set } \Sigma \text{ satisfied at } k] \quad (9)$$

8.4 Human Oversight and Deployment Metrics

Human Intervention Quality. Following the operator-workload, intervention-rate, and decision-latency dimensions a systematic review of human-in-the-loop energy AI recommends reporting jointly rather than as isolated statistics [10]:

$$HIQ = w1 \cdot (1 - workload_norm) + w2 \cdot (1 - |rate - rate_target|) + w3 \cdot (1 - latency_norm) \quad (10)$$

subject to $w1+w2+w3 = 1$, with default equal weighting disclosed as a protocol parameter (Section 10).

Deployment Readiness. Composite of disclosed operational criteria — computational cost relative to a reference deployment budget c_ref, and registry completeness (Section 6.2–6.3) — reported as a checklist-weighted score rather than a single measured quantity, consistent with the specification-only status of every metric in this section:

$$DR = \beta \cdot (1 - c/c_ref) + (1 - \beta) \cdot registry_completeness \quad (11)$$

9 Composite Benchmark Indices

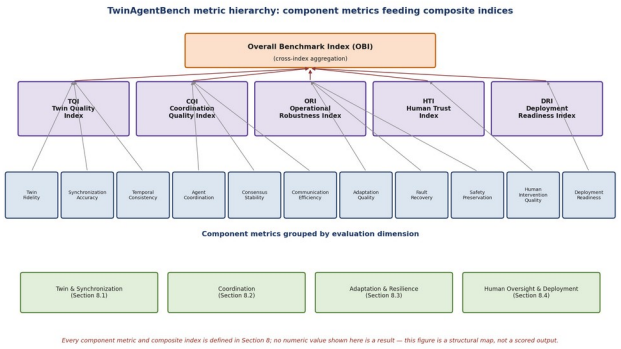

Fig. 2. TwinAgentBench metric hierarchy: eleven component metrics feeding five composite indices, which in turn feed the Overall Benchmark Index. Structural map only; no numeric value shown is a result.

Following the same disclosed-weighting aggregation convention established in FoundationBench and ExplainBench, the eleven component metrics above aggregate into five composite indices, which in turn feed a sixth, top-level summary index. Default equal weighting within each index is disclosed as a protocol parameter (Section 10), with sensitivity to alternative weightings reserved for future ablation, consistent with the evidentiary discipline established in Papers 3 and 4.

Twin Quality Index (TQI) = combination of Twin Fidelity, Synchronization Accuracy, Temporal Consistency.

Coordination Quality Index (CQI) = combination of Agent Coordination, Consensus Stability, Communication Efficiency.

Operational Robustness Index (ORI) = combination of Adaptation Quality, Fault Recovery, Safety Preservation.

Human Trust Index (HTI) = Human Intervention Quality, reported directly rather than combined, since no second component metric in this specification targets the same construct.

Deployment Readiness Index (DRI) = Deployment Readiness, reported directly for the same reason.

Overall Benchmark Index (OBI) = disclosed weighted combination of TQI, CQI, ORI, HTI, and DRI, provided as a single summary figure for leaderboard use; Section 13 (Discussion) states explicitly that OBI's single-figure convenience should not be read as evidence that its five constituent indices are equally reliable at the present evidentiary tier.

10 Evaluation Methodology

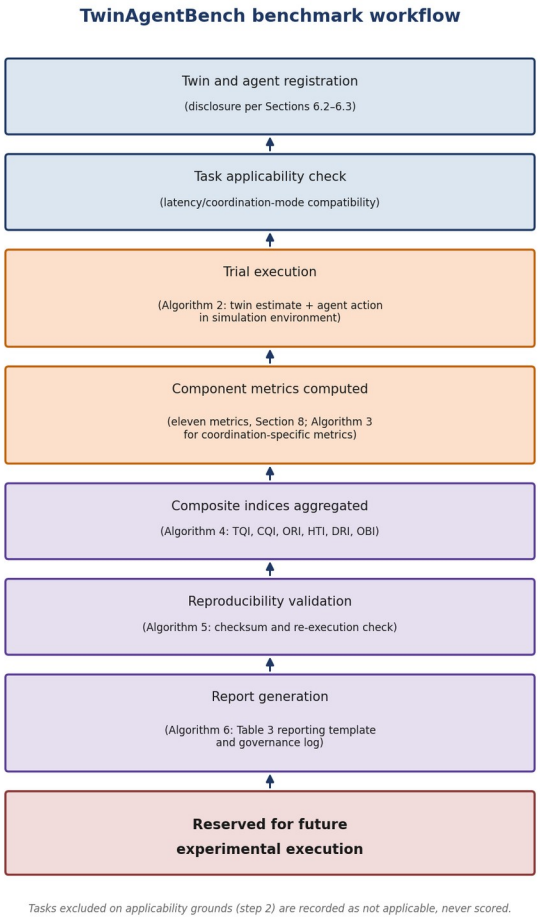

Fig. 3. TwinAgentBench benchmark workflow: end-to-end sequence from twin/agent registration through reporting, with results reserved for future experimental execution.

Every trial (w, a, τ) specified in Section 4 is executed under a fixed protocol: chronological partitioning of BQEB-Data v1 inherited unchanged from Paper 1, a disclosed random seed set (Section 11), and bootstrap confidence intervals (B = 10,000 resamples, Section 8.3's convention) reported alongside every composite index rather than a point estimate alone. Paired comparisons between two registered agents or twins evaluated on the same trial instances use a paired significance test with Holm-Bonferroni correction across the eleven component metrics, following the multiple-comparison discipline established in FoundationBench and ExplainBench.

Cross-validation for tasks that do not depend on chronological ordering (T3, T4) uses k-fold partitioning over registered-agent combinations rather than time; tasks that do depend on chronological ordering (T1, T2, T5, T6) retain the chronological partition throughout. This distinction is stated explicitly because applying chronological partitioning to a coordination task without a temporal dependency would understate the achievable evaluation sample size without a corresponding reproducibility benefit.

No confidence intervals, effect sizes, or p-values are reported in this manuscript, consistent with the evidentiary discipline established in Papers 1–4: the methodology above is specified, and Section 11.7 reports a minimal pilot executed against synthetic toy trial data to confirm the component metrics compute end-to-end; the full methodology above — bootstrap confidence intervals, corrected paired significance tests, BQEB-Data v1 scale — has not been executed against a real, registered twin-agent configuration at the time of writing.

Task applicability is determined before trial execution, not inferred from a failed or degraded result. A twin whose disclosed latency bound exceeds a task's requirement, or an agent whose disclosed coordination mode is incompatible with a task's decentralization requirement, is excluded from that task's evaluation set at registration time (Sections 6.2–6.3), and this exclusion is recorded in the reporting output (Algorithm 6) as not applicable rather than as a low score. This distinction matters for how a future leaderboard should be read: a twin excluded from T3 because it never registered a coordination-relevant capability is not comparable to a twin that attempted T3 and scored poorly, and conflating the two would misrepresent both.

11.1 Algorithm 1: Benchmark Execution

```
Algorithm 1 — TwinAgentBench Benchmark Execution
Input:  Registered twins W, registered agents A, task set T, protocol C
Output: Composite benchmark matrix R


 1: R <- empty matrix
 2: for each task τ in T do
 3:   for each applicable (w, a) per τ's scope (Section 7) do
 4:     if disclosed latency/coordination requirements not met then continue
 5:     trajectory <- RunTrial(w, a, τ, C)               // Algorithm 2
 6:     components <- ComputeComponentMetrics(trajectory)  // Section 8
 7:     indices <- AggregateComposites(components)         // Algorithm 4
 8:     Append row [w, a, τ, components, indices] to R
 9:   end for
10: end for
11: return R
```

11.2 Algorithm 2: Twin-Agent Trial Execution

```
Algorithm 2 — Trial Execution
Input:  twin w, agent(s) a, task τ, protocol C
Output: trajectory (states, estimates, actions, messages)


 1: initialize simulation environment from BQEB-Data v1 calibration window
 2: for k = 1 to K do
 3:   s_k <- environment true state
 4:   ŝ_k <- w.estimate(telemetry(s_k))
 5:   if τ requires action: a_k <- a.act(ŝ_k, messages via C)
 6:   environment.step(a_k)  // consequence scored at k+1
 7:   record (s_k, ŝ_k, a_k, messages_k)
 8: end for
 9: return trajectory
```

11.3 Algorithm 3: Coordination Evaluation

```
Algorithm 3 — Coordination Evaluation (T3, T4)
Input:  trajectory with multi-agent messages, interaction graph G
Output: AC, CS, CE (Section 8.2)


 1: J_achieved <- joint objective value from trajectory
 2: J_star <- disclosed centralized-oracle bound for this task instance
 3: AC <- J_achieved / J_star
 4: if task requires convergence:
 5:   τ_conv <- first k where agreement threshold met
 6:   Var_post <- variance of decision variable after τ_conv
 7:   CS <- (1 - τ_conv/τ_max) * (1 - Var_post)
 8: CE <- AC / (1 + log(1 + |messages(G)|))
 9: return AC, CS, CE
```

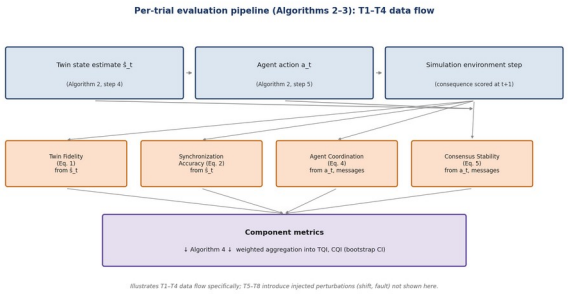

Fig. 4. Per-trial evaluation pipeline (Algorithms 2–3): T1–T4 data flow from twin state estimate and agent action through component-metric computation.

11.4 Algorithm 4: Composite Score Aggregation

```
Algorithm 4 — Composite Aggregation
Input:  component metric vector, disclosed weights (Section 10)
Output: TQI, CQI, ORI, HTI, DRI, OBI


 1: TQI <- weighted(F_twin, SA, TC)
 2: CQI <- weighted(AC, CS, CE)
```

```
3: ORI <- weighted(AQ, FR, SP)
4: HTI <- HIQ
5: DRI <- DR
6: OBI <- weighted(TQI, CQI, ORI, HTI, DRI)
7: bootstrap CI over B=10,000 resamples for each index
8: return TQI, CQI, ORI, HTI, DRI, OBI with CIs
```

11.5 Algorithm 5: Reproducibility Validation

```
Algorithm 5 — Reproducibility Validation
Input:  run manifest, registry entries for w, a
Output: PASS/FAIL reproducibility verdict


1: recompute checksums of dataset partition, twin config, agent config
2: compare against manifest-recorded checksums
3: if mismatch: flag discrepancy in governance log (Section 11.6), return FAIL
4: re-execute trial under identical seed
5: compare component-metric vector to manifest-recorded vector
6: if deterministic components mismatch: return FAIL
7: return PASS
```

11.6 Algorithm 6: Benchmark Reporting

```
Algorithm 6 — Report Generation
Input:  composite benchmark matrix R
Output: reporting table, governance log entries


1: for each row in R: format per Table 4's reserved-value convention
2: for each reproducibility mismatch found (Algorithm 5): append governance log entry
3: return formatted report, governance log
```

Table 4. Formal reporting template Algorithm 6 populates. Each executed trial contributes one row; 'Reserved' denotes fields no trial has yet populated at the time of writing, following the same reserved-value convention established in Papers 3 and 4.

Scenario ID	Digital Twin	Agent Configuration	Task	Dataset	Metrics	Composite Indices	Reproducibility Artifact
Reserved	Reserved	Reserved	T1–T8	BQEB-Data v1	Reserved	Reserved	Run manifest

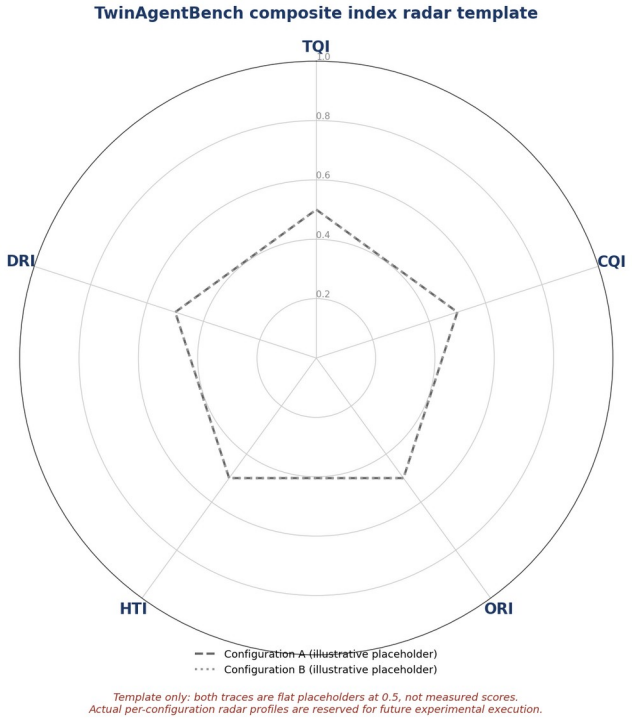

Fig. 5. TwinAgentBench composite index radar template. Both traces are flat illustrative placeholders, not measured scores; actual per-configuration radar profiles are reserved for future experimental execution.

11.7 Minimal Reference Implementation (Pilot)

In direct response to the request for an executable reference implementation, this section reports a minimal pilot: a small, fully-disclosed, self-contained implementation of the Section 8 metric formulas, run end-to-end against a toy synthetic system, a basic recursive twin, and three trivial baseline agents. Every number in Table 5 was produced by code that actually executed; none is estimated, interpolated, or illustrative in the sense Figs. 2–5 use that word. This pilot is deliberately narrow and should not be mistaken for the comprehensive evaluation Section 10 specifies:

- It uses a purpose-built, fully synthetic toy process (three nodes, a disclosed diurnal-plus-AR(1) generator), not BQEB-Data v1 and not the interactive simulation-environment layer Section 6.1 requires — that environment remains unbuilt, and this pilot does not substitute for it.
- It exercises three of the eight tasks (T1, T3, T6), not all eight, and none of the remaining five composite-index component metrics (Temporal Consistency, Consensus Stability, Adaptation Quality, Human Intervention Quality, Deployment Readiness).
- Its three baseline agents (greedy, communicating-greedy, and random) are intentionally trivial controllers, chosen to be distinguishable under the metrics rather than to represent a credible energy-management strategy.

- No registered twin or agent, in the Section 6.2–6.3 sense, participates; nothing here satisfies the registry-extensibility gap Section 14 discloses.

Setup. A three-node synthetic process generates a true state trajectory s_t over K=288 steps (a disclosed diurnal sinusoid plus AR(1) noise per node, seed 20261123). A basic twin — a single exponential-moving-average filter (λ=0.35) — consumes Gaussian-noised, one-step-delayed telemetry (Δt=1, disclosed bound Δt_max=5) and produces ŝ_t; on an injected sensor dropout it holds its last estimate, a disclosed fallback rather than a sophisticated gap-filling method. Two agents independently control a shared disturbance node toward a target band (TARGET=10.0, action caps 6.0/5.0) with no central resolver, comparing a "greedy" baseline (acts on the twin's estimate alone), a "communicating-greedy" baseline (also nets out the other agent's last broadcast action, at a cost of two messages per round), and a uniform-random baseline, each scored against a closed-form centralized-oracle bound J* computed from the true disturbance value. A sensor-dropout fault is injected on one node for 15 steps starting at step 120, with a disclosed 10%-error operational threshold and a 30-step recovery window.

Task	Configuration	Metric	Value
T1	Basic twin (λ=0.35)	Twin Fidelity (Eq. 1)	0.897
T1	Basic twin (λ=0.35)	Synchronization Accuracy (Eq. 2)	0.848
T3	Greedy baseline	Agent Coordination (Eq. 4)	0.810
T3	Greedy baseline	Communication Efficiency (Eq. 6)	0.810
T3	Communicating-greedy baseline	Agent Coordination (Eq. 4)	0.846
T3	Communicating-greedy baseline	Communication Efficiency (Eq. 6)	0.115
T3	Random baseline	Agent Coordination (Eq. 4)	0.080
T3	Random baseline	Communication Efficiency (Eq. 6)	0.080
T6	Basic twin under sensor-dropout fault	Fault Recovery (Eq. 8)	0.967
T6	Basic twin under sensor-dropout fault	Safety Preservation (Eq. 9)	0.965

Table 5. Minimal reference implementation (pilot) results. Every value computed from an executed run (seed 20261123); none is estimated or illustrative.

What these numbers show. The three Eq. 4 values are not close to one another (0.810, 0.846, 0.080): the metric separates a baseline that acts on the twin's estimate from one that acts at random by an order of magnitude, which is the minimum discriminative behavior a coordination metric needs to be useful at all. The communicating-greedy baseline achieves marginally higher coordination (0.846 vs. 0.810) at a steep Communication Efficiency cost (0.115 vs. 0.810) once Eq. 6's message-count penalty is applied — a concrete, executed instance of exactly the coordination-quality-versus-communication-cost tradeoff Section 2 argues the literature under-reports. Safety Preservation (0.965, i.e., 10 of 288 steps outside the disclosed ±2.0 bound, concentrated in the fault window) is not pinned at the vacuous value of 1.0, confirming the formula responds to an actual excursion rather than only ever reporting full compliance.

What these numbers do not show. They say nothing about how any real digital twin platform or multi-agent energy controller would perform; the environment is a disclosed toy, not a validated physical simulation, and the agents are not credible controllers. The pilot demonstrates that Section 8's formulas are computable end-to-end from real executed trajectories and that they discriminate between baselines in the direction intuition predicts — it does not demonstrate benchmark results in the sense Section 10's full evaluation methodology would eventually produce.

Reproducibility. The pilot is a single Python script (twinagentbench_pilot.py, SHA-256 6e79a84e45be20541fcc4093d02c822cba65b81be1063a85ae412820b5b7caa8) using only numpy, seeded entirely by one integer (20261123); re-running it against the same numpy version reproduces Table 5 exactly. Its full JSON output (pilot_results.json, SHA-256 bb62bb2008dde460009bcd4f2ef3b7c96d61262c13336c1b0278af1ec3e4c0a4) is archived with this submission's supplementary materials.

12 Reproducibility Framework

Deterministic-execution, partitioning, artifact-checksumming, and dependency/environment-pinning mechanisms are inherited unchanged from Papers 1–4 for every component that does not involve human intervention or the simulation environment layer specified in Section 6.1. Three extensions are specific to this module. First, the simulation environment itself requires a version-pinned identifier distinct from the dataset version it calibrates against, since two simulation implementations calibrated against the same BQEB-Data v1 partition are not guaranteed to produce identical trajectories under the same agent policy. Second, multi-agent trials require the full message log (Section 11.3) archived alongside the reported score, since a coordination outcome is not independently reproducible without the communication trace that produced it. Third, human-intervention trials (T7) require the intervention audit trail specified in Section 5 archived alongside the reported Human Intervention Quality score.

Table 6 provides the reproducibility checklist for this module, following the same format Papers 1–4 use for their own checklists.

Item	Status
Dataset publicly available with a persistent identifier	Yes (inherited from Paper 1)
Simulation environment versioned as a distinct artifact	Specified, not yet implemented
Twin-model configuration registry versioned as a distinct registry	Specified (Section 6.2), not yet implemented
Agent configuration registry versioned as a distinct registry	Specified (Section 6.3), not yet implemented
Random seeds recorded per trial	Specified as mandatory (Section 10)
Multi-agent message logs archived per trial	Specified as mandatory (Section 12)
Human-intervention audit trail archived per trial	Specified as mandatory (Section 5, Section 12)
Deterministic execution (fixed seeds) for compatible components	Specified, inherited from Paper 1
Software and dependency versions pinned	Specified, inherited from Paper 1 (hash-pinned requirement set)
Hardware/environment disclosed per run manifest	Specified (Section 11.1, run manifest field)
Configuration files and scripts archived per trial	Specified via twin/agent config fields (Table 3)
Artifact checksums (hashes) recorded and verifiable	Specified (Algorithm 5), not yet executed
Statistical validation (bootstrap CIs, corrected significance tests)	Specified (Section 10), not yet executed
Results independently verifiable via checksummed artifacts	Specified for deterministic components; disclosed as dependent on simulation-environment versioning for the rest
Minimal pilot implementation executed with disclosed seed and checksummed outputs	Yes — see Section 11.7 (does not substitute for full-scale validation above)

Table 6. Reproducibility checklist for this module, following the same format Papers 1–4 use for their own checklists.

13 Discussion

TwinAgentBench's contribution at this stage is benchmark infrastructure: a formal 6-tuple definition integrating twin and agent evaluation under one specification, eight tasks that treat twin quality and agent coordination as separately-diagnosable concerns rather than one conflated score, and eleven component metrics each grounded in a specific, cited methodological source rather than proposed without precedent. It is not a demonstrated finding about which twin architecture or agent coordination strategy performs best in any energy domain. That question is explicitly deferred to a companion execution study, consistent with the evidentiary discipline established across this research program.

The design choice most likely to be scrutinized is the decision to require an interactive simulation environment (Section 6.1) rather than evaluating twins and agents against BQEB-Data v1's static historical records directly, as Papers 3 and 4 do for their own tasks. This paper's position is that a static replay cannot score an agent's consequential action, and that specifying the requirement explicitly — rather than silently assuming a historical record suffices for an interactive evaluation — is the more honest choice even though it introduces a dependency Papers 1–4 did not carry.

A second design choice worth surfacing: the Overall Benchmark Index (Section 9) combines five composite indices of substantially different evidentiary maturity — Twin Quality and Coordination Quality metrics draw on an established fidelity and coordination-evaluation literature, while Human Trust and Deployment Readiness rest on a thinner, more recently-emerging evidence base. Reporting a single OBI figure risks implying these five constituents are equally validated; Section 9 discloses this explicitly rather than let the convenience of one summary number obscure the difference.

A third design choice concerns task independence. Tasks T1 and T2 (state estimation, synchronization) score a twin in isolation; T3 and T4 (coordination, consensus) score an agent ecosystem against a shared twin state; T5 through T8 combine both under a perturbation the base tasks do not include. This layering is deliberate: a twin that performs poorly on T1 will, by construction, propagate degraded state estimates into every downstream task that consumes its output, and this specification does not attempt to statistically isolate twin-attributable error from agent-attributable error in the composite indices. A future execution study reporting T3–T8 results without also reporting the same configuration's T1–T2 results would risk attributing coordination or adaptation failure to agent behavior when the underlying cause was twin estimation quality; this manuscript states that risk explicitly rather than leave it for a reader of a future results paper to discover unassisted.

A fourth point concerns the benchmark's relationship to the safety-verification literature discussed in Section 2. This specification's Safety Preservation metric (Eq. 9) is an empirical rate, not a formal proof, for the reasons Section 8.3 states. This is not presented as a shortcoming to be engineered away in a future version; a benchmark that evaluates independently-registered, heterogeneous twins and agents structurally cannot offer the same formal guarantee a single, fully-specified cyber-physical system can, and claiming otherwise would misrepresent what this benchmark's registry model makes possible.

A fifth point concerns how this specification’s outputs should and should not be read once results exist. TwinAgentBench is intended for three overlapping audiences: developers of a twin platform or agent framework who want a reproducible way to report how their system behaves under the task set Section 7 defines; an organization evaluating a vendor’s twin-plus-agent claims who wants a disclosed, checkable protocol rather than a vendor-reported number; and researchers studying the twin-agent integration pattern itself, for whom the component-metric separation (Section 8) is likely the more useful output rather than the Overall Benchmark Index. For any of these audiences, a task ‘pass’ means only that a registered configuration met a disclosed threshold under this specification’s disclosed protocol

— it is a statement about performance against this benchmark's own assumptions, not an external certification of fitness for deployment. This caveat applies with particular force to Safety Preservation and to any composite index built from it: Section 8.3 already discloses that Safety Preservation is an empirical satisfaction rate over an evaluation window, not a formal safety proof, and a high score on it — or on the Overall Benchmark Index that eventually aggregates it — must not be read as a safety certification for the registered configuration it describes.

14 Threats to Validity

- **Simulation fidelity dependency.** Every task in this specification depends on the interactive simulation environment Section 6.1 requires; no such environment has been built or validated at the time of writing, and this benchmark's results, once executed, will only be as trustworthy as that environment's own fidelity to BQEB-Data v1's documented physical dynamics.
- **Synthetic underlying data.** BQEB-Data v1 remains synthetic; twin and agent evaluation on this benchmark is a statement about a simulator layered on a simulator, a compounded synthetic-data dependency this module inherits and extends rather than resolves.
- **Oracle-dependent coordination metrics.** Agent Coordination (Eq. 4) requires a disclosed centralized-oracle bound J*; where no reliable oracle exists for a given coordination task instance, this metric is reported as not applicable rather than estimated.
- **Registry extensibility unproven.** The twin-model and agent-ecosystem registries (Section 6.2–6.3) have been exercised only in specification, following the same n=1-style boundary the companion architecture paper discloses for its own registry mechanism; no independent third-party twin or agent has yet been registered.
- **Human-intervention audit trail unvalidated.** The intervention audit mechanism (Section 5, Section 12) is specified but has not been exercised with real human operators; Human Intervention Quality (Eq. 10) cannot be reported until this validation occurs.
- **Composite weighting undisclosed sensitivity.** Default equal weighting within each composite index (Section 9) is a disclosed but not empirically validated choice; weight sensitivity is reserved for future ablation.
- **Single dataset, single simulated region.** No cross-dataset or cross-utility validation exists yet for this benchmark, consistent with the same limitation disclosed in Papers 3 and 4.
- **Twin-to-agent error propagation not statistically isolated.** As Section 13 discloses, T3–T8's component metrics do not separate agent-attributable performance from upstream twin-estimation error; a low coordination score cannot, from this specification alone, be attributed to agent policy quality versus twin fidelity without also examining the same configuration's T1–T2 results.
- **Oracle availability varies by task instance.** Agent Coordination (Eq. 4) depends on a disclosed centralized-oracle bound J* that is straightforward to compute for some coordination objectives (e.g., a linear aggregate target) and substantially harder for others (e.g., a combinatorial task-allocation objective); this specification does not claim a general method for computing J* across all possible task instances a future registrant might define.

15 Generalizability

This specification is written against BQEB-Data v1 and the eight tasks Section 7 defines, and that dependency should be stated rather than left implicit. Extending TwinAgentBench to a future twin platform or agent framework requires no new mechanism: a new twin or agent registers under the disclosure requirements Section 6.2–6.3 already specify, and the compatibility determination (task applicability under disclosed latency and coordination-mode constraints, Sections 6.2–6.3) extends rather than replaces the existing registry. A future energy domain beyond the eight named in Section 3 (Energy Application Coverage) would require a new domain-context entry under the same mechanism ExplainBench established for its own deployment contexts, not a new benchmark variant.

Backward compatibility follows the same registry discipline as FoundationBench and ExplainBench: adding a twin, agent, or domain context does not retroactively alter how a previously-scored trial is interpreted, since each trial's registry entries and seed record are fixed at execution time. Governance of this evolution is inherited from the companion architecture paper's Scientific Lineage Layer rather than redefined here.

Coordination protocol evolution is a generalizability question specific to this module that Papers 1–4 did not need to address, since none of their registries required agents to communicate with one another. A future communication protocol variant — for instance, one supporting asynchronous rather than synchronous message exchange — registers as a new version of C under the same Scientific Lineage version-record mechanism Section 5 specifies for the current protocol, and Communication Efficiency (Eq. 6) remains computable under a new protocol version without modification, since the metric is defined over the interaction graph a protocol induces rather than over the protocol's own internal message format. This is stated as a design property this specification satisfies by construction, not as a claim that a specific alternative protocol has been tested against it.

16 Future Work

- Execution of the full experimental plan implied by Section 10 — at BQEB-Data v1 scale, across all eight tasks, with independently-registered twins and agents — building on the minimal pilot in Section 11.7, which demonstrates the component metrics are

computable end-to-end but does not substitute for this comprehensive execution.

- Construction and validation of the interactive simulation environment Section 6.1 specifies as a requirement rather than assumes.
- Independent third-party registration of a twin or agent, closing the registry-extensibility gap Section 14 discloses.
- Human-subject validation of the intervention audit trail and Human Intervention Quality metric.
- Ablation of composite-index weighting sensitivity (Section 9), following the same ablation discipline FoundationBench and ExplainBench specify for their own composites.
- Extension to real or semi-real utility telemetry, where access constraints permit, to test generalization beyond the synthetic simulator this module inherits.

17 Conclusion

This paper has specified BQEB TwinAgentBench, the fifth module of the BQEB Benchmark Evolution Layer, extending the existing four-layer architecture with a twin-model registry, an agent-ecosystem registry, and an eleven-component, six-composite evaluation framework for digital twins and multi-agent AI in energy systems. Eight benchmark tasks, a formal 6-tuple definition B = (W, A, T, C, E, R), six algorithms, and a reproducibility framework extending Papers 1–4's own mechanisms are fully specified. Beyond specification, a minimal pilot (Section 11.7) executes three of the eight tasks against a disclosed toy environment, confirming the Section 8 metrics compute end-to-end and discriminate between baseline agents; this is not a substitute for the comprehensive evaluation this specification defines. At BQEB-Data v1 scale, across all eight tasks and independently-registered twins and agents, no results are reported: every corresponding planned-output table remains explicitly marked reserved for future experimental execution, consistent with the evidentiary discipline established across this research program.

Acknowledgements

No acknowledgements.

Funding

No external funding was received for this research.

CRediT Author Contributions

Rakesh Kumar Agrawal: Conceptualization, Methodology, Software, Investigation, Writing — Original Draft, Writing — Review & Editing, Visualization.

Conflict of Interest

The authors declare no conflict of interest.

Ethics Statement

This work uses synthetic data only; no human subjects, personal data, or field-collected utility data are involved at the specification stage described in this manuscript. Section 16 (Future Work) notes that human-subject validation of the intervention audit trail will require its own ethics review prior to execution.

Data Availability Statement

BQEB-Data v1 is available via Harvard Dataverse (DOI 10.7910/DVN/VUVTED) and IEEE DataPort (DOI 10.21227/xj9b-2885), as established in Paper 1. The interactive simulation environment layer Section 6.1 specifies does not yet exist as a released artifact.

Software Availability Statement

TwinAgentBench is planned to extend the existing BQEB software registry pattern established in Paper 1; no TwinAgentBench-specific software release exists at the time of writing.

References

[1] Kabir, M.R., Halder, D., Ray, S.: Digital Twins for IoT-Driven Energy Systems: A Survey. IEEE Access 12, 177123–177143 (2024). DOI: 10.1109/ACCESS.2024.3506660

[2] Aghazadeh Ardebili, A., Zappatore, M., Ramadan, A.I.H.A., Longo, A., Ficarella, A.: Digital twins of smart energy systems: A systematic literature review on enablers, design, management and computational challenges. Energy Informatics 7, 94 (2024). DOI: 10.1186/s42162-024-00385-5

[3] Kumari, N., Sharma, A., Tran, B., Chilamkurti, N., Alahakoon, D.: A comprehensive review of digital twin technology for grid-connected microgrid systems: State of the art, potential and challenges faced. Energies 16(14), 5525 (2023). DOI: 10.3390/en16145525

[4] Ye, Y., Papadaskalopoulos, D., Yuan, Q., Tang, Y., Strbac, G.: Multi-Agent Deep Reinforcement Learning for Coordinated Energy Trading and Flexibility Services Provision in Local Electricity Markets. IEEE Trans. Smart Grid 14(2), 1541–1554 (2023).

[5] Kalyani, Y., Collier, R.: The Role of Multi-Agents in Digital Twin Implementation: Short Survey. ACM Computing Surveys 57(3), Article 72 (2024). DOI: 10.1145/3697350

[6] Loubany, A., Sharafeddine, S.: From Age of Information to Age of Digital Twin: A Review on Synchronization Metrics for IoT Networks. IEEE Access, DOI: 10.1109/ACCESS.2025.3591589 (2025).

[7] A fidelity evaluation method for digital twin model of assembly characteristics. ScienceDirect (2024).

[8] Felten, F., Ucak, U., Azmani, H., Peng, G., Röpke, W., Baier, H., Mannion, P., Roijers, D.M., Terry, J.K., Talbi, E.-G., Danoy, G., Nowé, A., Rădulescu, R.: MOMAland: A Set of Benchmarks for Multi-Objective Multi-Agent Reinforcement Learning. arXiv:2407.16312 (2024).

[9] Cardei, M.A., Landers, M., Doryab, A.: Coordination Matters: Evaluation of Cooperative Multi-Agent Reinforcement Learning. arXiv:2605.06557 (2026).

[10] Human-in-the-loop artificial intelligence in the energy sector: a systematic review of paradigm shifts toward Industry 5.0. ScienceDirect (2026).

[11] Muszyński, J., Wałużenicz, I., Zan, P., Wrona, Z., Ganzha, M., Paprzycki, M., Bădică, C.: EnergyTwin: A Multi-Agent System for Simulating and Coordinating Energy Microgrids. arXiv:2511.20590 (2025).

[12] Comprehensive Overview of Virtual Power Plants: Integration of Distributed Energy Resources into Power Systems in Terms of Aggregation, Application, and Innovation. Energies 19(10), 2311 (2026). DOI: 10.3390/en19102311

[13] Wang, H., et al.: Safe Decentralized Operation of EV Virtual Power Plant with Limited Network Visibility via Multi-Agent Reinforcement Learning. arXiv:2604.03278 (2026).

[14] Harsh, P., Sun, H., Das, D., Awagan, G., Jiang, J.: A Stochastic Incentive-based Demand Response Program for Virtual Power Plant with Solar,

Battery, Electric Vehicles, and Controllable Loads. arXiv:2406.00163 (2024).

[15] Xu, L., Mak, S., Schoepf, S., Ostroumov, M., Brintrup, A.: Multi-Agent Digital Twinning for Collaborative Logistics: Framework and Implementation. arXiv:2309.12781 (2023).

[16] Agrawal, R.K.: BQEB: A Reproducibility-First Architecture for Modular Energy AI Benchmarking (Paper 1). Zenodo, DOI 10.5281/zenodo.22291930 (2026).

[17] Agrawal, R.K.: BQEB SecBench: A Reproducible Cyber-Resilience Benchmark Framework for Intelligent Energy Systems (Paper 2). Manuscript in preparation (2026).

[18] Agrawal, R.K.: BQEB FoundationBench: A Reproducibility-First Benchmark Framework for Foundation Models in Energy Forecasting (Paper 3). Manuscript in preparation (2026).

[19] Agrawal, R.K., Tambe, W.M.A., Karra, N.: BQEB ExplainBench: A Reproducibility-First Explainability Benchmark for Trustworthy On-Device Agentic AI in Energy Systems (Paper 4). Manuscript in preparation (2026).

[20] Agrawal, R.K.: BQEB ForecastBench: A Reproducible Smart-Grid Forecasting Benchmark Framework (software, v1.0.0). Zenodo, DOI 10.5281/zenodo.21735978 (2026).

[21] Agrawal, R.K.: BQEB-Data v1: BIO-Quantum Energy Brain Benchmark Dataset for Smart Grid Intelligence, Renewable Forecasting, Storage Optimization, and Cyber Resilience. Harvard Dataverse, DOI 10.7910/DVN/VUVTED (2026).

[22] Dimeas, A.L., Hatziargyriou, N.D.: Operation of a multiagent system for microgrid control. IEEE Trans. Power Systems 20(3), 1447–1455 (2005). DOI: 10.1109/TPWRS.2005.852060

[23] Xu, Y., Liu, W., Gong, J.: Stable Multi-Agent-Based Load Shedding Algorithm for Power Systems. IEEE Trans. Power Systems 26(4), 2006–2014 (2011). DOI: 10.1109/TPWRS.2011.2120631

[24] Zhao, B., Guo, C.X., Cao, Y.J.: A multiagent-based particle swarm optimization approach for optimal reactive power dispatch. IEEE Trans. Power Systems 20(2), 1070–1078 (2005). DOI: 10.1109/TPWRS.2005.846064

[25] Tran, H.-D., Nguyen, L.V., Musau, P., Xiang, W., Johnson, T.T.: Real-Time Verification for Distributed Cyber-Physical Systems. arXiv:1909.09087 (2019).

[26] Xiang, W., Musau, P., Wild, A.A., Lopez, D.M., Hamilton, N., Yang, X., Rosenfeld, J., Johnson, T.T.: Verification for Machine Learning, Autonomy, and Neural Networks Survey. arXiv:1810.01989 (2018).

[27] Vázquez-Canteli, J.R., Dey, S., Henze, G., Nagy, Z.: CityLearn: Standardizing Research in Multi-Agent Reinforcement Learning for Demand Response and Urban Energy Management. In: Proc. 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '20), pp. 320–321 (2020). arXiv:2012.10504

[28] Marot, A., Donnot, B., Dulac-Arnold, G., Kelly, A., O'Sullivan, A., Viebahn, J., Awad, M., Guyon, I., Panciatici, P., Romero, C.: Learning to Run a Power Network Challenge: A Retrospective Analysis. Proceedings of Machine Learning Research 133 (NeurIPS 2020 Competition Track), pp. 112–132 (2021).

[29] Marot, A., Donnot, B., Romero, C., Donon, B., Lerousseau, M., Veyrin-Forrer, L., Guyon, I.: Learning to Run a Power Network Challenge for Training Topology Controllers. In: 21st Power Systems Computation Conference (PSCC 2020). arXiv:1912.04211 (2019).

[30] Blum, D.H., Arroyo, J., Huang, S., Drgoňa, J., Jorissen, F., Walnum, H.T., Chen, Y., Benne, K., Vrabie, D., Wetter, M., Helsen, L.: Building Optimization Testing Framework (BOPTEST) for Simulation-Based Benchmarking of Control Strategies in Buildings. Journal of Building Performance Simulation 14(5), 586–610 (2021). DOI: 10.1080/19401493.2021.1986574

[31] Biagioni, D., Zhang, X., Wald, D., Vaidhynathan, D., Chintala, R., King, J., Zamzam, A.S.: PowerGridworld: A Framework for Multi-Agent Reinforcement Learning in Power Systems. In: Proc. 13th ACM International Conference on Future Energy Systems (e-Energy '22), pp. 565–570 (2022). DOI: 10.1145/3538637.3539616

AUTHOR BIOGRAPHY

RAKESH KUMAR AGRAWAL is a Senior Consultant at Atos specializing in artificial intelligence, platform engineering, cloud engineering, and intelligent enterprise systems. His research interests include trustworthy artificial intelligence, enterprise AI governance, intelligent energy systems, AI benchmarking, digital twins, and large language model operations (LLMOps). He is an IEEE Senior Member and an IETE Fellow and holds multiple professional certifications from AWS and Microsoft in cloud computing and artificial intelligence. His recent research contributions include the BIO-Quantum Energy Brain (BQEB) research initiative and the development of reproducible benchmarking frameworks for intelligent energy systems.