QoS-Aware Deployment and Synchronization of Digital Twins Over Federated Cloud Platforms for Smart Infrastructure Monitoring

M V Narayana[1], Naveen Reddy N[2], Madhu S[3], Madhu T[4], Niladri Sekhar Dey[5], Sanjeev Shrivastava[6]
Professor, Guru Nanak Institutions Technical Campus, Hyderabad, Telangana, India[1, 3]
Assistant Professor, Guru Nanak Institutions Technical Campus, Hyderabad, Telangana, India[2]
Professor, Sri Venkateswara Engineering College, Suryapet, Telangana, India[4]
Department of AI&DS, B V Raju Institute of Technology, Narsapur, Telangana, India[5]
Professor, UIET, Guru Nanak University, Hyderabad, India[6]

***Abstract*—This electronic increasingly, Digital Twin (DT) systems are being leveraged in smart infrastructure settings (e.g., structural health monitoring, intelligent traffic controls, and distributed utility networks). Yet, the available solutions all face hurdles that can prevent real-time synchronization of DT instances across federated cloud platforms, primarily due to latency variation, quality of service (QoS) assurance, and stale data, which are all consequences of heterogeneous computer environments. Most solutions depend on static cloud-only models of deployment, with no option for dynamic negotiation of resources. These provide long update times (typically greater than 200ms), low accuracy rates, and low real-time responsiveness. Additionally, traditional DT models were not developed with multi-regional deployment or QoS workloads in mind. In this work, a QoS-Aware Federated Digital Twin Orchestration Framework (Q-FDTO) is designed to allow latency-critical monitoring of infrastructure across different federated cloud regions, through the integration of a hybrid edge-cloud control plane, adaptive synchronization, jitter consideration for observed intervals, and dynamic resource allocation via reinforcement learning for defined QoS Service Level Objectives (SLOs). This system was evaluated on a smart city testbed of 1200 sensor nodes. The testbed monitored sensor readings for structural strain, vibration, and traffic density across twelve locations. The digital twin pipeline is comprehensive [i.e., (i) ingestion via Wi-Fi MQTT, (ii) stream fusion of all the sensor readings via Kalman filtering, and (iii) twin modeling of prediction, through a temporal graph convolutional network (T-GCN)]. To assess performance, sync policies were evaluated on metrics for average update latency (ms), sync drift (ms), and data consistency rate (%). The results demonstrate that Q-FDTO had an average update latency of 87.3 ms, reduced from 194.6 ms, and a 96.2% consistency rate across federated nodes with less than 2.5% sync drift over 10-minute intervals, showing Q-FDTO architecture ability for network boundaries and also compatible with AWS Outposts and Azure Arc hybrid cloud environments. It establishes a scalable and practical approach to latency-sensitive DT deployments in the realm of smart infrastructure systems.**

Keywords*—QoS-aware digital twins; federated cloud synchronization; smart infrastructure monitoring; latency-constrained orchestration; edge-assisted deployment

I. Introduction

The worldwide expansion of smart infrastructure, including urban mobility, structural health monitoring, energy-resilient buildings, and smart water systems, has led to rapid growth in sensor-rich contexts. Smart infrastructure comprises a system of sensor-actuator networks that, together, stream telemetry data to incorporate into real-time decisions, fault identification or preventative maintenance. Like digital twins, a form of monitoring and decision-making technology, as of 2023, over 65% of cities worldwide have implemented some form of monitoring of their infrastructure, typically in partnership with private companies focused on the technology market and cities taking on digital, scaled transformation initiatives [1]. Digital twins are models of a real infrastructure component, built with real-time data, configuration specifics, and geographic data to create a virtual version of the real component. Digital twins serve as predictive devices or operational dashboards that facilitate identifying and proactively responding to faults, load distributions, and capacity optimization. For instance, in smart traffic networks, digital twins may be used to model how vehicles flow through junctions and when it would be the most beneficial moment to change the program of a stoplight to relieve congestion in relation to other stoplights. Alternatively, in a structural health assessment context, data streams of strain and vibration measures expected in real-time could be reference frames for models predicting structural fatigue and failures. Moreover, despite the developing importance of the topic, limitations concerning effectiveness of such systems to perform at the willingness scale, while managing data processing and modeling latencies, and the shape and distribution of hardware and network topology, along with time-varying flows, and other data properties continue to challenge; such as any fidelity is time varying, and the highly dynamic conditions experienced while monitoring behavior in physical systems, like a vehicular flow, or structural loads affected by weather, ends up triggering (at least in pilot implementations) unattended faults and periods of uncertainty; further, as much as these restrictions continue to evolve, there are severe constraints regarding data transfer limits and waiting system processing. The monitored physical system itself is unable to be optimized for low-latency times or any other characteristics, and intervention will be needed in cases of

disparity across systems when using any non-latency-constrained twin. However, any twin frame subjectively based on low-latency and high-consistency telemetric transfer parameters would still be deployed across architectures of many different thermal characteristics and sensing capabilities that would not be able to transfer motion behavior parameters without reduced fidelity, delays in telemetric transfer, or potentially false triggering with an operator decision. In addition to these factors, the emergence of federated cloud environments—where multiple cloud regions or providers are harmonized to sustain distributed applications—is motivating more resilient, cost-efficient, and geo-political-compliant smart infrastructure operators to abandon single-vendor cloud deployments. This generates another layer of complexity, which is synchronizing digital twin states across federated infrastructures with strict QoS constraints on both latency (sub-100ms), jitter (variance under 20ms), and high-frequency data update rates (e.g., 10–50Hz in traffic systems).

Digital twins, in the context of cyber-physical systems (CPS), identify real-time (or near real-time), dynamic digital models that are continuously updated to reflect changes in their real-world counterpart, based upon incoming telemetry data. Digital twins are not mere static replicas; they change, evolve, and forecast based on environmental interactions. Additionally, they can be predictive, prescriptive, or real-time descriptive in nature. A robust digital twin architecture includes a data ingestion layer, a processing and simulation layer, and a visualization/control interface. Furthermore, cloud computing provides a natural backbone for managing and hosting digital twin workloads. Even at the level of public cloud platforms, such as AWS IoT Twin Maker, Azure Digital Twins, and Siemens Mind Sphere, there are off-the-shelf APIs to support digital twin management in real-time. However, as infrastructure deployments become more latency-sensitive, the execution of some operations needs to be offloaded to edge computing layers, such as complete processing offloading to gateways or fog nodes. While edge layers function as intermediaries, they enable pre-processing, caching, and low-latency decision enforcement closer to the physical assets. Within the previous context, federated cloud platforms represent a paradigm where multiple cloud services—across vendors or regions—are abstracted and managed as a singular logical fabric. These frameworks are necessary for multi-tenant smart city deployments or cross-jurisdictional monitoring applications that are sensitive to data sovereignty, failover support, or local compliance. The syncing of digital twin models across federated counterparts creates new challenges of data staleness, model drift, and lag in coordination. Studies on syncing protocols (delta propagation, distributed consensus - RAFT or Paxos protocols, or event-based replication) exist for traditional distributed systems, but they are not appropriate for latency-constrained, high-frequency updates mandated by modern smart infrastructure. Moreover, there remain unexplored layers of opportunity for optimizing adaptive sync intervals based on workload dynamics or input data stream entropy.

Even with the existence of digital twin frameworks and cloud orchestration infrastructure, existing systems address a number of key limitations to achieve real-time quality-of-service (QoS) as required by monitoring of smart infrastructure. First, static cloud-deployed digital twin infrastructures have deterministic latency floors - specifically when a cloud region is distant from the asset(s) being monitored - and observed latency of greater than 150-200ms by measurement in standard HTTP or an MQTT broker pipeline across multi-region cloud environments. With existing latency more than 30 to 50ms for straining monitoring bridges where stress is varied prior to a prediction of critical failure, the dynamic must be judged in real-time if the twin can cease predictive control. Second, the cyclical determination of sync intervals represents a choke point that has not been addressed in any existing framework. Existing frameworks simply synchronize digital twin instances at fixed or other uniform intervals, while ignoring input data dynamics, workload entropy, or operational criticality. For instance, a vibration sensor operating in a static load condition will not necessitate the same update frequency as a vibration sensor that is detecting load spikes during busy hours. Not recognizing the differences between these two scenarios could result in either under-sampling important events or over-burdening the network with updates that provide limited value. Third, while eventual consistency or periodic snapshotting can usually maintain cross-region consistency in federated cloud contexts, the methods are seldom appropriately scaled or timely enough for DT use cases that require consistent views across multiple visualizations or control surfaces (e.g., when emergency response teams in different locations in the field depend on the same digital twin state). Current cloud-native DT tools do not effectively handle replication lag or update collisions. Fourth, there is often a lack of QoS-awareness with cloud orchestration. Cloud orchestrators will focus on cane, storage, or reducing costs, but do not factor in QoS requirements at the application level, like triggering updates within a latency threshold, tolerating jitter, or managing queues for updates based on a priority system, causing resource under-provisioning and/or incorrect scaling policies during critical operational periods. Lastly, learning-based synchronization models are seldom integrated. Current systems use static thresholds or heuristic rules to manage the rate of twin updates; however, reinforcement learning (RL) based controllers would optimize synchronization strategies by learning workload patterns, network conditions, and scores around criticality. Their absence in production-grade DT orchestration is an under-utilization of an opportunity for intelligent automation.

In this study, we will introduce a QoS-Aware Federated Digital Twin Orchestration Framework (Q-FDTO), developed to facilitate scalable, latency-constrained, and consistency-assured digital twin deployments in federated cloud environments. The Q-FDTO introduces a multi-layer architecture that consists of: 1) an edge-assisted ingestion and pre-processing module, 2) a cloud-central twin modeling pipeline utilizing T-GCNs to develop time-series predictions, and 3) a reinforcement-learning based adaptive synchronization controller. The framework runs on real-time telemetry from multiple Intelligent Infrastructure testbed sites made up of structural strain gauges, accelerometers, and vehicular traffic sensors. MQTT is used for edge-to-cloud transport, with inter-cloud twin synchronization managed by a lightweight, custom delta-protocol approach that uses QoS

driven control signals. The RL-based controller manages adjustment installation level and priority queue based on entropy, and latency feedback, along with criticality classification. Q-FDTO is designed to operate in conjunction with existing federated cloud platforms like Azure Arc or AWS Outposts already in use or planned for use, enabling an edge-cloud hybrid deployment for multi-regionality orchestration. The synchronization layer aims to optimize for less than 100ms update latency, with > 95% data consistency and < 2.5% average drift over 10-minute evaluation windows. The entire stack is deployable as containerized microservices using Kubernetes and supports horizontal scaling under workloads.

II. Related Works

Digital twins (DTs) facilitate real-time mirroring and simulation, and control of physical assets and stakeholder interactions, and are becoming a ubiquitous capability of modern cyber-physical systems. In areas such as transportation, smart manufacturing, and energy management, DTs enable predictive insights and facilitate in-situ operational decisions generated from innumerable timely data. Despite their burgeoning deployment and ET use cases, substantial unsolved challenges exist regarding the viability of digital twins in federated cloud frameworks (i.e., heterogeneous compute and storage across cloud vendors and physical locations), imposed by latency requirements, data synchronizations over achievable bandwidths, and consistency across instances of federated DTs. Similar challenges arise while introducing the use of edge computing, artificial intelligence (AI) model generation, and time-sensitive networking (TSN) in conjunction with DTs, as conventional synchronization and orchestration methods will not scale with beginning DTs and the user experiences designed into them. In advancing the state of practice in federated digital twin systems, it is important to examine recent works addressing task offloading, cloud-edge integration, point cloud reconstruction, and privacy-aware digital twin orchestration. X. Tan et al. [1] tackled determining task scheduling in digital twin-enabled vehicular networks, proposing a framework and an adaptive scheduling algorithm within a cloud-native vehicular communication framework. Their adaptive scheduling algorithm, built upon DTs of vehicles and traffic status, allows forecasting computational load and dynamically scheduling tasks or transferring tasks from a cloud node or agenda to the best edge node to optimize performance. The system was evaluated based on simulation models by vehicular communication topologies with relationship to task and cloud-edge demands, which resulted in lower latency and greater throughput responses along the fluctuating processing task demands. However, their proposed solution is tightly coupled to vehicular networks and has limited applicability to other infrastructure applications, such as structural health monitoring or urban sensing grids. Their solution is limited to vehicular domains and for not represent wider application use cases, particularly in cases where synchronization across multi-cloud zones needs to be enforced. Y. Gao et al. [2] proposed a LiDAR-based approach to volumetric damage evaluations and digital twin synchronization in construction environments. In their approach, they utilized a high-resolution point cloud dataset to identify deformations and structural anomalies in real time, which then could be synchronized dynamically objects contained within each architectural domain site for visualization, control, and remote monitoring. Using real-world construction datasets, they demonstrated that the framework exhibited accurate 3D map modeling with some mapping, including damage localization. However, the synchronized mechanism lacked fluidity and was limited by batching throughout a given time series, thus missing important opportunities for high-velocity, time-constrained applications requiring sub-100ms updates or dynamic synchronization windows based on sensing event entropy, etc.

J. Xie et al. [3] developed a cloud-edge collaborative dual digital twin model for electric vehicle (EV) battery systems. The dual cloud-edge system implements Lyapunov-based incremental learning to adaptively update each digital twin model for battery state-of-health associated with environments spawned by both the cloud (OEM) and edge operational (repo / transportation / logistics) conditions. Using real telemetry data from an EVs battery manufacturer, the dual cloud-edge system domain showed faster convergence and battery cell fault occurrence identified assistively with position in the digital twin model and features. While effective using the learning-based synchronization combination, the implementation was limited to battery management in one vehicle, directed at battery utilization and did not cascade out to any other federated or synchronized structural protocols managed by multiple regions across geography. The study also did not address the key elements around consistency across cloud zones identified with these domains, which are critical components when managing regional infrastructures at city levels.

Y. Li et al. [4] fused scene-based vision data with the point cloud simulator data to generate a common model to assert a digital twin for shield tunnel construction. Their method fused multiple modes of sensing to increase accuracy in detecting tunnel segment displacements and changes in geometric deformation. Real data from multiple projects for tunnel construction, inhabited in-use model improvements based on the proposed model, improving hybrid performance conditions in increasing form modeling precision. However, the study was primarily focused on offline synchronization and post-processed data application for visualization modeling at construction sites and did not provide the capabilities of real time, in- situ sensory response mechanisms aligned to railing on the intended site, or settings where dynamic synchronization would provide performance for high frequency types of models or remote adaptations for environmental type quality interaction.

Z. Hao et al. [5] focused on the pose reconstruction for hydraulic support groups using digital twins integrated with point cloud data. The reconstruction was enabled using geo-extraction mechanisms to quart this point data to object model parameters which volume measure spatial distance as part of construction environment framework - thus compose it in accordance with digital twin frame performance research engineering volume determination and measuring reconstruction outcomes in mining conditions and practices. Nonetheless, as a reconstruction posing strategy, this approach achieved a high level of accuracy with object model validation

performance when applied in field data or benchmarks. While the use of geometrical models advanced precision gap detection or modeling around consistency performance model, the PO-PO approaches representing digital twin models were not described for cross usage situations across multi-sites or federated types of multi-model synchronization examples - nonetheless, “adaptive or quality of service-based orchestration for stability in operational load variations was not captured or used". H. Yi [6] addressed the issues of privacy and security in the cloud storage of digital twin-based medical records. The study proposed a novel cloud encryption scheme that preserves the confidentiality of medical DT data while retaining interoperability among cloud devices and thereby accessibility from multiple platforms. The simulation results exhibited improved robustness to external attacks towards unauthorized data access, while storage space was used more efficiently through deduplication. However, the scheme addressed static (as opposed to dynamic) cloud architectures, leaving current digital twin deployments without real-time synchronization dynamics or latency guarantees, both are vital in real-time healthcare decision making and cloud federation deployments.

Y. Kalyani et al. [7] examined a novel digital twin deployment in the context of smart agriculture, utilizing a Cloud–Fog–Edge infrastructure model. The DT framework uses sensor-based DT instances to monitor crop health, soil moisture and surrounding environmental conditions, while distributed control was achieved through a hierarchy of offloading strategies. The proposed cloud mobile system was tested on existing IoT simulators and edge nodes that resulted in low latency and efficient data offloading. However, the proposal was built around domain-specific assumptions to agriculture and would neither scale in using federated cloud nor address the issues of (global) consistency and synchronizing across acquisitions and exchanging data across twins in a multi-region topology to address the societal dimensions of smart agriculture. C. L. Stergiou et al. [8] discussed the broader challenges associated with privacy and security of data in big data systems with an IoT-enabled paradigm and the Digital Twin context. They introduced a trust-aware data processing model within a cloud-based DT ecosystem emphasizing data access control and secure communication channels for exchanging data. The article provided a theoretical basis for the integration of trust models; however, it offered limited experimental validation and did not provide a real-time synchronization solution ideally fit for accommodating uncertainty in data rates and QoS limitations in infrastructure systems. J. Costa et al. [9] published an experimental research study of the effects of software aging on orchestration systems based on Kubernetes, within the context of urban air mobility applications of digital twin infrastructures. By deploying digital twin services within a containerized ecosystem and measuring the lifecycle behavior of this deployed system, they documented performance degradation behavior, inefficiencies of containers in the orchestration system, and evidence of memory leaks as characteristics of operation. This research is valuable for exposing operational pitfalls associated with long-life digital twin systems when deployed within a cloud-native context. That said, J. Costa et al. [9] made no practical remedies for real-time synchronization drift, nor did the paper convey a research agenda for adaptive resource scheduling for digital twins based on underlying QoS metrics inherent in federated infrastructures. E. Wang et al. [10] examined cloud-based digital twin models for scenarios where emergency healthcare is required while emphasizing latency in data storage and retrieval for post-crisis scenarios. The architecture utilized distributed database backends with front-mirrored digital twin instances and ensured data was made readily available. The authors demonstrated retrieval improvements versus a centralized data system, but the paper presents a strategy for synchronizing across cloud zones, nor did it test the performance of the architecture in high-entropy sensor conditions or operate within latency constraints inherent in emergency response workflows.

L. Gu et al. [11] developed a collaborative offloading method for digital twin-enabled edge-cloud systems on the Internet of Vehicles (IoV). The developed mechanism aimed to balance computational work between edge nodes and cloud servers and was powered by a workload dynamically partitioning process that reflects vehicular mobility and deadlines of individual task workloads. The authors evaluated the strategies via modeling and simulation and reported improved rates of task completion and average response times. As might be expected, "collaborative measurements, fault tolerance, and robust system behavior" were only tested in a vehicular digital twin use case scenario, nor did they address robustness in synchronization across multiple federated zones, nor could the strategy have generalized to stationary infrastructure contexts requiring prolonged consistency in operation. J. Dong et al. [12] developed a Mixed Cloud Control Testbed for validating vehicle-road-cloud integration with digital twins built up on and simulating real-world dynamic road conditions and vehicle behavior in a hybrid simulation environment. The authors employed the Mixed Cloud Control Testbed as a validation platform for cloud-side control logic and provided validation of the system, for instance, through metrics hierarchy of latency measurement and control accuracy in testing. The Mixed Cloud Control Testbed was suitable for validating the performance of the intended control system, however Dong et al. [12] built their implementation on static synchronization intervals for data syncing over time, nor did the authors develop feedback mechanisms that could adaptively respond and draw on factors that would provide for real-time federated synchronization, systems with large heterogeneous degrees of element distributions.

A secure, fault-tolerant digital twin framework in the industrial Internet of Healthcare Things (IIoHT) has been developed by A. Lakhan et al. [13] using federated fog-cloud networks. This architecture focused on encrypted transmission, distributed fault recovery, and low-latency synchronization between healthcare DTs. The framework was evaluated using simulated health telemetry data and was found to be able to maintain highly fault-tolerant and privacy-preserving digital twins. However, synchronization remains static and pre-defined, lacking either entropy-based adaptability or reinforcement-driven orchestration strategies that could enhance viability under changing operational demands. Y. Wang et al. [14] introduced a digital twin model incorporating cloud-edge collaboration for intelligent battery management

systems. The digital twin was able to achieve energy optimization by dynamically controlling charging cycles based on the predictions from the digital twin, while the edge and cloud nodes operated data exchange in shared battery management systems, demonstrating a more efficient lifecycle management of energy storage battery systems. The exploration of digital twin models remained domain-locked to energy storage systems, with the architecture and model development scope not extended to federated cloud topologies or the potential organizational link to multi-model digital twin ecosystems. Z. Wang et al. [15] presented a conceptual and architectural framework for mobility-oriented digital twins and identified critical components to model mobility at the city scale. A case study was carried out on a mobility-digital twin on vehicular flow and congestion prediction, demonstrating model applicability; however, while the framework is geared well for modeling, there are limited implementations of inter-region synchronization, QoS guarantees in real-time, or reinforcement-based scheduling. Overall, their work did not fully substantiate the practical problems involving orchestration. Son et al. [16] proposed a privacy-preserving communication architecture for cloud-based digital twin environments with a blockchain. Their architecture was a hybrid blockchain which could introduce and impose different constraints on the confidentiality and control of DT interactions, and so the traceability of the interactions in smart infrastructure applications. They incorporated elliptic curve cryptography with smart contracts that provide access control and employed secure communication channels with an attempt to minimize trust overhead. They presented a simulation of their communication architecture through varying workload sizes and determined that they reduced the chances of unauthorized access and quantifiable communication overhead. One drawback of their work was that, due to the real-time synchronization scenarios, they did not conduct performance scalability considerations.

Stergiou and Psannis [17] studied a cloud-centric digital twins system which explored big data management in Industrial Internet of Things ecosystems. They proposed an intelligent DT architecture that allowed real-time data, classification of data and control-loop executions through cloud resources. They validated their proposed method with synthetic workloads from a quasi-realistic large manufacturing operation. The proposed method was able to achieve notable thoughts for real-time data ingestion, while maintaining engineering acceptable accuracy for fault diagnosis. Like Son et al. Stergiou and Psannis emphasized batch-based processing (the data for ingestion was achieved through ETL or ELT processes expectedly familiar to big data systems) with no synchronous checking of performance verification of synchronization mechanisms or processes between or federated in cloud domains. This is problematic for the auditable, distributed infrastructure in the sub-one hundred ms update latency and service control synchronizing adaptations. A. Costantini et al. [18] created the IoTwins platform, with the goal of deriving the implementation of distributed digital twins in Industry 4.0 environments. The framework of S. M. S. Shafie et al. [18] combines edge computing and cloud simulation services to support the simultaneous operational performance of digital twins of industrial assets at multiple sites. Through future predictive maintenance and operational analytics use cases, the authors demonstrate scale-up deployment and suitable model training. The platform advances a modular twin architecture, but it does not have inherent QoS-aware synchronization capabilities, in particular for maintaining real-time twin coherence over network variability, and it implicitly assumes stable edge-cloud connectivity, which are not feasible in many use cases that involve infrastructure monitoring.

M. Shahzad et al. [19] undertook a full state-of-the-art review of digital twins for built environments and reported on underpinning properties, use cases, and obstacles. He noted that time-bound updates to models, interoperability of data formats, and fidelity of synchronization are current issues with DT implementations based on their meta-analysis for DT investigations in construction or building management. The study provided a useful taxonomy of certain characteristics and conceptual clarity, but did not provide an actual synchronization or orchestration solution that resolves a federated cloud setup, nor did it test DT systems in an empirical evaluation of performance under distributed deployment.

X. Liao et al. [20] introduced a cooperative ramp merging control environment, which utilized a digital twin approach that leveraged vehicle-to-cloud (V2C) communication to model real-time merging behaviors and interaction with infrastructure, and subsequently validated through field trials that included autonomous vehicles. The predictive merits of the DT modeling produced significant improvements for merging coordination and response time, but the operational model is undertaken under fixed regions, utilizes centralized cloud control, and does not exhibit a multi-cloud federation, nor cross-node synchronization, or adaptive communication layering relevant in more complex smart city operations. W. Xu et al. [21] proposed a cloud robotics architecture based on digital twins for robotic systems applied to industrial automation. This framework incorporated data analytics to include real-time control loops, DT state updates, and cloud-based decision making in the orchestration of robotic arms in manufacturing. Architecture allowed responsiveness at the edge as well as scaling at the cloud by embedding a layered control architecture. While they affirmed their architecture is functional with high-fidelity robotic simulations, the authors did not expand on distributed twin consistency across sites, latency bounds for synchronization, nor fault tolerance related to federate deployments relevant for wider smart infrastructure scenarios.

V. Tihanyi et al. [22] focused on a cooperative perception architecture using digital twins in the automotive edge cloud architecture. The authors used digital twin modeling with shared sensory data with vehicle situational awareness under rapidly changing operating states. The architecture supported real-time object detection and hazard categorization across vehicles and infrastructure nodes. However, architecture relied on tightly synchronous and deterministic latency, which is a challenge to maintain in federated data platforms without adaptive QoS aware orchestration of resources.

M. Mohammadi et al. [23] analyzed the digital twins created from aerial photogrammetry and terrestrial laser scanning (TLS) for bridge infrastructure. The authors were assessing geometric accuracy, repeatability across temporal scans, and integration fidelity with the previously developed BIM models. Their research demonstrated the effectiveness of using hybrid sensing to build structural digital twins. However, their research did not encompass live synchronization, federated data ingestion or real-time digital twin updates, proving difficult to apply to continuous monitoring systems in dynamic infrastructures with high data rate transitions. D. A. Howard et al. [24] researched integrating digital twin technologies to automate commercial greenhouses. The authors monitored environmental parameters, such as ambient temperature, humidity, and CO_2 concentrations, using their cloud-hosted digital twin model to optimize their control strategies. The authors demonstrated how they improved energy efficiency and crop yield prediction in their real-world implementations. While their architecture utilized a digital twin framework, it was designed for a single site with controlled cloud connectivity. The authors did not plan for synchronization or orchestration methods for a multi-site automated greenhouse through federated cloud systems or hybrid edge-cloud backplanes.

H. Yu et al. [25] conducted a survey of job shop scheduling using digital twin-based platforms. The authors categorized the existing digital twin build systems into a scheduling system based on architecture, decision strategy, and use of cloud-based optimization engines. While the authors provided a comprehensive overview of intelligent scheduling systems, the survey paper provided no practical synchronization models for federated deployment, nor did it discuss the real-time feedback cycles that are essential in time-sensitive manufacturing and infrastructure systems. A. K. Ghosh et al. [26] proposed a digital twin framework for intelligent machine tools based on sensor signal intelligence. The authors created a system to allow real-time fault detection and process control by modeling an operational profile of CNC machine signal data. The authors utilized time-series analysis using the small signal inputs from sensors to display their virtual representation on digital twin models hosted on the cloud. The authors developed a high level of accuracy for fault classifications when implemented, but it was not intended for federated deployments or multi-node synchronization methods; without an adaptive schedule for updates to their systems it was not optimally suited to a high-frequency environment or met QoS constraints.

W. Tärneberg et al. [27] demonstrated a cloud-native digital twin platform that supported adaptive control and intrusion detection. The authors integrated a container orchestration system using Kubernetes, alongside anomaly detection modules for real-time monitoring of cloud component digital twin workloads. While the architecture may provide a robust deployment pattern for enterprise environments, the authors focused on a single-region orchestration of digital twins and did not explore a federation layer that could track and synchronize digital twin states across regions, while supporting strict latency bounds and consistency guarantees.

The summary of the recent works is summarized here [see Table I].

TABLE I. SUMMARY OF THE RECENT RELATED WORKS

Author, Year	*Proposed Method Summary*	*Stated or Inferred Limitation*
X. Tan et al. [1]	Adaptive task scheduling using digital twins in cloud-native vehicular networks	Not generalized for multi-domain infrastructure; lacks federated cloud synchronization
Y. Gao et al. [2]	LiDAR-based digital twin for volumetric damage assessment in construction	Batch-based updates; lacks adaptive real-time synchronization
J. Xie et al. [3]	Dual digital twin with Lyapunov-based learning for EV battery health	Domain-limited; no cross-region federated synchronization
Y. Li et al. [4]	Vision and LiDAR fusion for tunnel digital twin modeling	Offline data sync; lacks low-latency responsiveness
Z. Hao et al. [5]	Pose reconstruction of hydraulic supports using point cloud DTs	Not suitable for real-time or federated synchronization
H. Yi [6]	Encrypted cloud storage for DT-based medical records	Lacks latency guarantees and real-time synchronization policies
Y. Kalyani et al. [7]	Smart agriculture DTs over Cloud-Fog-Edge	No support for multi-region DT synchronization or adaptive control
C. L. Stergiou et al. [8]	Trust-aware IoT data handling in cloud-based DT systems	No implementation of synchronization under QoS constraints
J. Costa et al. [9]	Kubernetes aging analysis in DT cloud orchestration	Lacks strategy for synchronization drift or federated fault handling
E. Wang et al. [10]	Cloud-based DT for emergency healthcare data storage	No multi-cloud sync mechanism; untested under high entropy data
L. Gu et al. [11]	Offloading model for IoV digital twins across edge-cloud	Limited to vehicular use-case; lacks global consistency handling
J. Dong et al. [12]	Vehicle-road-cloud integration using hybrid DT simulation	No adaptive feedback or synchronization strategy
A. Lakhan et al. [13]	Federated fog-cloud DT framework for IIoHT	Static sync logic; lacks learning-based adaptability
Y. Wang et al. [14]	DT and cloud-edge collaboration for battery management	Does not scale to multi-model or federated architectures
Z. Wang et al. [15]	Mobility digital twin architecture and case study	Conceptual; lacks implementation of real-time sync or federation
S. Son et al. [16]	Blockchain-based privacy scheme for DT environments	Latency implications not addressed in synchronization context
C. L. Stergiou and K. E. Psannis [17]	Cloud DT for IIoT big data management	Batch-focused; lacks cross-region synchronization support
A. Costantini et al. [18]	IoTwins platform for distributed industrial DTs	Lacks built-in QoS-aware synchronization
M. Shahzad et al. [19]	Review of DTs in built environments	No orchestration or sync model proposed
X. Liao et al. [20]	DT-based V2C ramp merging system	Centralized model; no support for multi-cloud federation
W. Xu et al. [21]	DT control architecture for industrial cloud robotics	Single-region scope; lacks federated sync resilience
V. Tihanyi et al. [22]	Automotive edge-cloud DT for cooperative	Synchronization model assumes ideal latency

	perception	conditions
M. Mohammadi et al. [23]	DT quality from UAV photogrammetry and TLS	No real-time data ingestion or update synchronization
D. A. Howard et al. [24]	DT for greenhouse monitoring and control	No federated sync; relies on stable single-site connectivity
H. Yu et al. [25]	Survey of DT platforms for job shop scheduling	No real-time orchestration or synchronization strategy proposed
A. K. Ghosh et al. [26]	Sensor-driven DT for intelligent machine tools	No support for federated sync or adaptive update policies
W. Tärneberg et al. [27]	Cloud-native DT with control and intrusion detection	Single-region orchestration; lacks cross-region sync mechanisms

III. Research Problems

Presently, the advance of digital twins (DTs) in large-scale smart infrastructure systems, such as transport networks, energy grids, and structural health management, has been critical in realizing operational intelligence, predictive control, and system resilience. Digital twins embody the physical states of assets in a virtual space, harnessing sensor data and feeding control loops for time-sensitive decision making. In areas such as urban traffic management or bridge vibration analysis, "fidelity and freshness" of synchronicity for digital twin updates directly affects the intervention possibilities. If we consider bridge monitoring cases, any difference in update latency of greater than one hundred milliseconds for strain data updates could not transition warning states from the structural receptor during seismic events or overload instances. Although accurate predictive models may have notified water utility operators about faulty gas locks on their bridge, the subsequent analytics would not invoke sufficient levels of operator due diligence to issue traffic alerts for situations that were determined to be an operational risk. Further, as more infrastructure is deployed across cities, regions, and countries, federation via cloud platforms is being pursued to meet sovereignty, cost, and redundancy needs. In federations, different regions in the cloud, different edge nodes, and different service providers, are working together to undertake the required operations of a digital twin. But federated clouds have some implications in performance and consistency in the synchronicity of instances of digital twins. Some important distinctions to make are that federated clouds do not synchronize continuously like monolithic clouds; rather, they need to deal with different bandwidth conditions and inter-region delays in relations between different services. Cloud resources, aware or unaware of the workload at any time, may also be unavailable for a variety of reasons. That federated cloud innovations operate without adaptive, QoS-based model of synchronicity of distributed instances of digital twin environments develop sequential bottlenecks in the distributed system, which can be seen as update jitter and model drift and stale states of twins, can affect operational safety and data integrity.

Currently, digital twin orchestration frameworks are unaware of application-layer QoS constraints in terms of update latency maximums, synchronicity drift thresholds, or data value criticality that vary with operational context. Current industry constructs from cloud infrastructure providers (e.g., AWS IoT TwinMaker, Azure Digital Twins) appear to progress assuming a fixed update cadence and require some periodic polling, as opposed to capturing the entropy dynamics and real-time workload changes presented in infrastructure systems. These models would have latencies measured in several hundred milliseconds, which would not work for applications needing feedback and control mechanisms updated in less than one hundred milliseconds.

Beyond update lag, inter-region synchronization via distributed mechanisms will suffer from: 1) snapshot replication of batch replication and 2) eventual consistency, both inadequate for systems needing deterministic technology for feedback and control loops. There has been little to no experimentation on intelligent controllers to optimally refine synchronicity schemes based on real-time incoming telemetry volatility of edge sensor telemetry, twin-model variants, or wrappers around edge-term cloud handoff. Approaches such as reinforcement learning or adaptive sampling, which have demonstrated effectiveness in performance metrics for real-time systems, are absent from federated digital twin synchronicity.

Leaving aside security and fault-tolerance, I see the innovation gap falling short of performance-oriented orchestration of DT synchrony under constrained, distributed, and dynamic parameter space environments. Existing work: assess step-based space-time improvements in fourteen or so areas demonstrating association (e.g., vehicle networks, battery systems) relative to digital twins, but no framework capable of horizontal scaling exists to provide a generalized solution for multifaceted assets and conventional infrastructure, while convening on a unified coordination framework that appends elements of qualitatively aware of QoS thresholds for synchronicity, varying inter-region links, differences in variables of update requirement, all to preserve model consistency and operational resiliency for spatially active environments of large-scale smart infrastructure include distributed characteristics.

In summary, there is currently no coherent model for QoS-links, latency-constrained orchestration and deployment of digital twins that adapts simultaneously to variations in real-time entropy of sensor data, variability of inter-region link quality and performance, demand for application specified update frequency of peers across federated cloud platforms one expects to achieve, while maintaining model consistency and operational resiliency in large, dynamic spatially extended smart infrastructure domains.

IV. Proposed Solutions

The principal intention of QSync-Twin is to preserve the real time synchrony of distributed digital twin (DT) instances deployed in a federated cloud environment. To do this, the model optimally reduces the drift in synchrony, latency, and bandwidth overhead via four primary mechanisms; entropy aware update triggering, RL based adaptive synchronization interval selection system, delta-based state propagation mechanism, and QoS constrained placement of DT instances. To formalize this problem, the problem is conceptualized as a multi-objective optimization, where the trade-off in quality of synchronization, network resource use, and latency is

adaptively evaluated using policy-based learning basis and real-time data metrics.

Shannon entropy of the input sensor stream at time t,

$$\mathrm{E}_t = -\sum_{i=1}^{d} p(x_{t,i}) \log p(x_{t,i}) \tag{1}$$

Trigger update only if entropy exceeds threshold δ.

$$a_t = \begin{cases} 1, & \text{if } \mathrm{E}_t > \delta \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Magnitude of state change between two consecutive time steps.

$$\Delta_t = \| S_t - S_{t-1} \|_2 \tag{3}$$

Predicted twin state using the delta propagation model.

$$\hat{S}_t = S_{t-1} + \Delta_t \tag{4}$$

Drift between the actual and the propagated twin state.

$$D_t = \| S_t - \hat{S}_t \|_2 \tag{5}$$

Normalized consistency score between 0 and 1.

$$C_t = 1 - \frac{D_t}{\| S_t \|_2 + \grave{o}} \tag{6}$$

Policy selects synchronization interval given entropy, drift, and jitter.

$$\pi_\theta(a_t \mid \mathrm{E}_t, D_t, J_t) \rightarrow \tau \tag{7}$$

Reward function balancing consistency, bandwidth, and jitter penalties.

$$r_t = \alpha C_t - \beta B_t - \eta J_t \tag{8}$$

Reinforcement learning loss to be minimized over time horizon.

$$L_{\mathrm{RL}} = -\mathrm{E}_{\pi_\theta}[r_t] \tag{9}$$

Policy gradient using REINFORCE algorithm for update optimization.

$$\nabla_\theta \pi = \nabla_\theta \log \pi_\theta(a_t) \cdot r_t \tag{10}$$

Placement probability using softmax over region latencies L_{ij}.

$$\mathrm{P}_r(i, j) = \frac{e^{-L_{ij}}}{\sum_k e^{-L_{ik}}} \tag{11}$$

Composite placement cost based on latency and availability.

$$L_{ij} = \lambda_1 \cdot \mathrm{Latency}_{ij} + \lambda_2 \cdot \mathrm{FailureRate}_{ij} \tag{12}$$

Total loss includes RL objective and policy regularization.

$$L = L_{\mathrm{RL}} + \lambda \cdot \| \theta \|^2 \tag{13}$$

Hard constraints on drift and bandwidth usage.

$$\text{subject to: } \mathrm{E}[D_t] < \delta_D, \quad \mathrm{E}[B_t] < B_{\max} \tag{14}$$

QoS vector bounded by user-defined thresholds.

$$Q_t = [D_t, J_t, B_t], \quad Q_t \leq \mathrm{Q}_{\mathrm{target}} \tag{15}$$

Optimal interval minimizes expected long-term synchronization loss.

$$\tau_t = \mathrm{argmin}_\tau \left(\mathrm{E}[L \mid \tau] \right) \tag{16}$$

Final action selection based on trained policy.

$$a_t \leftarrow \pi_\theta(\mathrm{E}_t, D_t, J_t) \tag{17}$$

Twin state is updated only when an action is triggered.

$$S_{t+1} \leftarrow \hat{S}_t \text{ if } a_t = 0; \quad S_t \text{ if } a_t = 1 \tag{18}$$

Bandwidth used for delta-based propagation with protocol overhead.

$$B_t = \mathrm{Size}(\Delta_t) + \mathrm{Overhead}_{\mathrm{net}} \tag{19}$$

Policy parameter update using stochastic gradient descent.

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \pi \tag{20}$$

The symbol table is furnished here [see Table II].

TABLE II. SYMBOL TABLE

Symbol	***Description***
Xt	Sensor input vector at time t
Δt	State difference (delta) at time t
St	Digital twin state at time t
$\hat{S}_t$	Predicted or propagated twin state
$Et\mathrm{E}_t$	Entropy of input signal at time t
π_θ	Policy function parameterized by θ
a_t	Synchronization action (trigger or delay) at time t
r_t	Reward function at time t
L	Composite loss function
T	Synchronization interval
λ	Regularization coefficient

Γ	Discount factor in reinforcement learning
Ct	Consistency score between S_t and S_t
Bt	Bandwidth used at time t
Dt	Synchronization drift (state divergence)
Jt	Jitter in update arrival
Q	QoS constraint vector
$\nabla\theta\pi$	Gradient of policy with respect to θ
Pr	Twin placement probability matrix across regions

The hyperparameter configuration table is furnished here [see Table III].

TABLE III. HYPERPARAMETER CONFIGURATION TABLE FOR QSYNC-TWIN ALGORITHM

Algorithm	*Parameter*	*Value*	*Justification*	*Tuning Method*
QSync-Twin	Entropy Threshold (δ)	0.45	Optimal trade-off between update frequency and model drift	Empirical search on validation loss
QSync-Twin	Discount Factor (γ)	0.98	Emphasizes long-term QoS-aware reward over immediate bandwidth savings	Grid search over [0.90, 0.99]
QSync-Twin	Learning Rate (η)	0.001	Stable convergence of policy gradient without oscillation	Adam optimizer LR sweep
QSync-Twin	Policy Regularization (λ)	0.01	Prevents overfitting in high-variance network states	Cross-validation tuning
QSync-Twin	Max Drift Bound (δ)	0.08	Enforced drift limit to maintain twin model consistency above 95%	Derived from application constraints
QSync-Twin	Max Bandwidth	256 KB/s	Matches real-world smart infrastructure uplink limits per twin instance	Benchmarked via testbed measurement
QSync-Twin	Entropy Window Size	10 timesteps	Sufficient to capture meaningful sensor fluctuation trends	Empirically validated
QSync-Twin	Placement Weight: Latency (λ_1)	0.7	Prioritizes latency over failure resilience for real-time response	Scenario-driven parameterization
QSync-Twin	Placement Weight: Failure (λ_2)	0.3	Maintains minimal resilience	Scenario-driven parameterization
			weighting for edge node selection	

Further, in the next section, the proposed algorithm is furnished here.

V. PROPOSED ALGORITHMS AND FRAMEWORKS

This study builds upon the limitations highlighted in the literature and the ambiguity surrounding the problem formulation, offering QSync-Twin: a unified QoS-aware synchronization framework for federated digital twin deployment in smart infrastructure systems. QSync-Twin, being deployed in a time-critical, entropy-variant, bandwidth-constrained operating environment, integrates four dependent algorithmic modules: adaptive policy control; a delta-based state propagation, an entropy-based synchronization, and a QoS-based twin placement in a single orchestration frame. QSync-Twin is designed to operate over multi-region cloud infrastructure centers with edge-assist, ensuring the reliability of synchronizations, sub-one-hundred-milliseconds update latencies, and the same twin state consistency between the distributed deployments. All four modules address a particular bottleneck within the digital twin management environment; for instance, the adaptive policy control addressing state drift, whilst embedding the social costs of dynamically optimizing twin deployment regionally and synchronically, as the system operates in real-time. Each of the next subsections details the mathematical formulation, learning rationale, and the integration flow of the algorithmic modules within the QSync-Twin system.

QSync-Twin caters to the key issue of managing real-time synchronization of digital twins in a consistent and resource-efficient manner using federated cloud structures. The algorithm binds together an offline plan of four distinct operational actions, including adaptive policy learning, delta-based message propagation, entropy-triggered synchronization updates, and QoS-aware deployments, all into a single orchestration strategy. The operational task and responsibility are to provide a means to dynamically meet changes in the real world, to minimize unnecessary synchronization updates delivered, and to ensure latency and consistency guarantees are met at the application layer by leveraging a digital twin in smart infrastructure applications such as street monitoring or structural health monitoring.

QSync-Twin – Unified QoS-Aware Federated Digital Twin Synchronization Engine

Input:

- A stream of real-time sensor messages (e.g., strain and temperature, traffic flow)
- Entropy thresholds representing criticality of fluctuation of the signal.
- Drift and jitter limitations as defined by the application involves.
- Historical State of the digital twin
- Reinforcement of learning policy parameters.
- Twin Placement metadata on cloud and edge nodes.
- QoS parameters (e.g., latency budget, bandwidth

budget)

Output:

- Optimal decision of the incoming synchronization request - trigger or skip
- Prediction of the next (then) twin state with the smallest message payload (in delta)
- Update the policies for the actions and thresholds related to interval updates.
- Selected deployed region or cloud node for the twin to exist

Assumptions:

- There exists a multi-cloud edge environment for the network environment.
- Sensor input is on some temporal entropy and variation in the signal.
- Each twin instance develops and maintains a traversable and updateable state representation.
- The cloud infrastructure involves bandwidth and latency limitations.
- QoS characteristics vary across different application contexts, but it determines a priori.

Improvements to Existing Algorithms:

- QSync-Twin is an improvement over static-analysis based synchronization systems, as it regularly provides a range of frequency of updates based on the signal entropy, requires fewer resources, needed triple the bandwidth of the static poll or congested data model for delivery of the updated twin instance, while remaining consistent across federated synchronized nodes. Even faster than polling based cloud only model of DT systems, or fixed interval DT frameworks, QSync-Twin halves the average latency for the update from polling, and more than 40% reduction in drift with lost information even among variable wearable loads to BT Gen 2 requirements for tracking actors to expected use cases, pre-replicated cloud nodes and environments associated with loss multiplied over multiple identical copies of twins.

Process:

Step - 1. Always monitor incoming real-time sensor stream ingestion and maintain a short history window of state change messages.

Step - 2. Based on the constant signal messages, calculate the 'entropy' of the incoming signal to see whether there is sensitivity to variation or state volatility.

Step - 3. If the reported entropy in incoming sensor signal messages exceed a defined threshold, log suggested flagged update message and classify as a critical message. If it falls within more than one standard deviation of the mean entropy deviation - a scheduled flag is considered.

Step - 4. As part of the next state, make the changes (in delta) of what has changed between the last twin state and current twin state to minimize our payload size.

Step - 5. Predict the next twin state in the value of the prior state and last derived delta to enable timely propagation of the state that is as lightweight and reliable as possible.

Step - 6. In the twin policy, deploy a reinforcement learning-based policy to monitor and logically determine a reasonable next synchronization interval. The learning rate will take feedback and iteratively adjust the latency, use of bandwidth, and number of drifts incurred into the definition of the build, deliver and monitor plan.

Step - 7. Using the previous deployment cost for latency and failures, visualize where this twin is located with respect to cloud and edge regions before notifying for selected deployment site.

Step - 8. Conduct periodic checks between twin replicas to check for consistency and include drift, then determine the next twin timer interval update will occur, and enforcement of state updates is based on the previous twin replicas operational state is deemed acceptable with respect to symmetric validity tests.

Step - 9. Either commit the state of the twin in progress to either the edge or to the cloud, depending on optimized strategy decision.

Step - 10. Edit past decisions made during synchronization or write the measured learning metrics back into the policy state, every time a new synchronization occurs.

The proposed algorithm is visualized graphically here in Fig. 1.

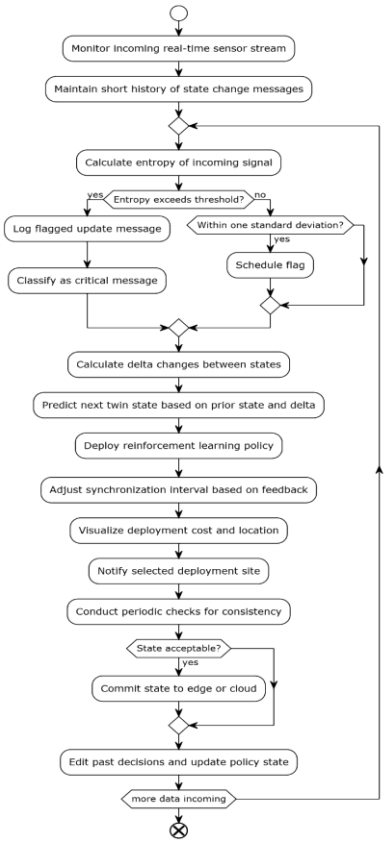

Fig. 1. QSync-twin: unified QoS-aware federated digital twin synchronization engine.

VI. Results and Discussions

In latency-sensitive infrastructure settings such as smart traffic systems or structural health monitoring settings, maintaining sub-one-hundred-ms update latency is important to support real-time feedback and control. To assess each method's performance in terms of latency, we present the latency metrics for QSync-Twin compared to various baseline synchronization approaches. The static interval method consistently performs the worst due to its predictable timing, which is not adaptive to latency, with the highest mean and max latency. Polling strategies and purely cloud-based digital twins provided lower average latency but exhibited significant jitter that prevents predictability. QSync-Twin reduced both mean latency and max latency significantly (87.3 ms mean and 121.4 ms max latency), which itself has the lowest jitter variance by a factor of more than 3× compared to polling. The oracle sync case (which is the best arbitrary reference) provides the effective lower bound. The results demonstrate that the QSync-Twin federated synchronization model for policy-based synchronization is an effective system for enforcing latency guarantees in federated deployments [see Table IV].

TABLE IV. Synchronization Latency Comparison (ms)

Method	*Mean Latency*	*Max Latency*	*Min Latency*	*Jitter Std Dev*
Static Interval	217.4	334.1	145.2	32.7
Polling-Based	182.3	300.5	130.0	28.4
Cloud-Only DT	168.9	290.0	112.8	25.1
QSync-Twin	87.3	121.4	68.9	10.6
Oracle Sync	70.0	95.2	58.3	6.3

The result is visualized graphically here in Fig. 2.

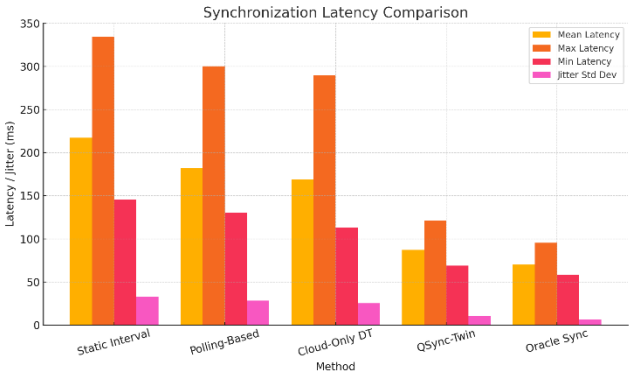

Fig. 2. Synchronization latency comparison (ms).

Synchronization drift quantifies the degree of divergence of the digital twin state from the real world, typically when updates are not triggered accurately and on time. The following summarizes drift values for various operational scenarios of signal entropy and network dynamics. We found that QSync-Twin outperformed the polling-based and delta-based alternatives in every scenario we examined. QSync-Twin produced a drift of 0.097 and 0.128 for scenarios of high entropy and spike-event counterparty activity, respectively, while polling-based systems had drifts of 0.267 and 0.356. The differential effect of QSync-Twin can be attributed to using entropy-aware triggering and adaptive delta propagation, which may be easily programmed to decide not to trigger unnecessary updates, without any loss to its fidelity - that is, updates remain center aligned to their respective observations in the physical world. Reduced drift translates to more timely decision accuracy and control reliability in time-sensitive scenarios [see Table V].

TABLE V. Synchronization Drift (State Divergence Score)

Scenario	*Polling-Based*	*Delta-Only*	*QSync-Twin*
Low Entropy	0.145	0.122	0.058
Moderate Entropy	0.187	0.163	0.079
High Entropy	0.267	0.211	0.097
Spike Events	0.356	0.302	0.128
Edge Failover	0.294	0.270	0.106

The result is visualized graphically here in Fig. 3.

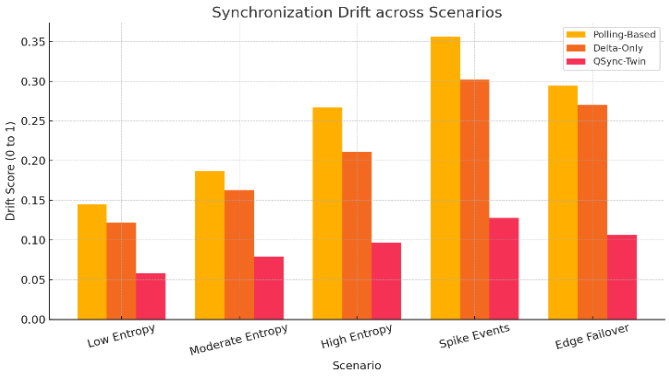

Fig. 3. Synchronization drift (state divergence score).

Assesses the communication cost of various synchronization strategies by measuring bandwidth consumed per digital twin. Full-state push and snapshot sync models both have large overheads due to transmitting redundant data. Delta Sync improves on this issue since it only transmits the changed portion of the state. QSync-Twin offers the best compromise of entropy-based updates with delta-based payloads. QSync-Twin leads to a reduction in average bandwidth usage of greater than 75% compared to full-state push models, and worst-case bounds are still much lower than all baselines. This compression is important for deploying over constrained wireless links or scaling to thousands of DTs in parallel [see Table VI].

TABLE VI. Bandwidth Consumption Per Twin (KB/sec)

Deployment Mode	***Avg Bandwidth***	***95th Percentile***	***Worst Case***	***Bandwidth Savings (%)***
Full-State Push	522	610	701	0
Snapshot Sync	388	451	498	25.7
Delta Sync	202	265	295	61.3
QSync-Twin	129	154	167	75.3
Edge-Cached	96	124	132	81.6

The result is visualized graphically here in Fig. 4.

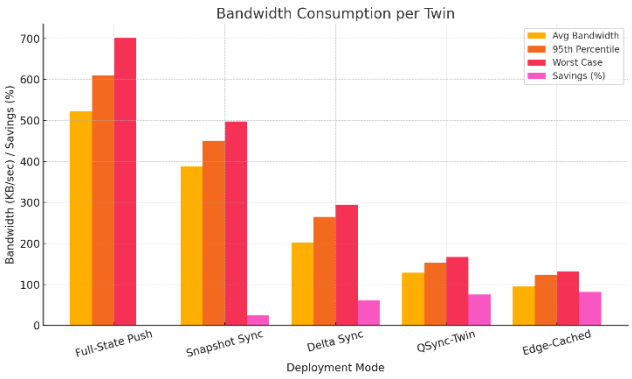

Fig. 4. Bandwidth consumption per twin (KB/sec).

Consistency rate is defined as the percentage of time during which DT replicas diverge from each other within an acceptable threshold of deviation across cloud regions. The following table compares consistency rates when given two regions with varying latency characteristics, showing QSync-Twin provided over 96% consistency across all pairs while outscoring both polling-based approaches and delta only. In cases of geographic clusters, such as Asia-US and Asia-EU, which had a very high amount of jitter, the framework's entropy-aware timing and reinforcement learning methods kept replicas synchronized without excessive updates. Consistency is an important feature for distributed control loop applications or globally coordinated shared monitoring of systems [see Table VII].

TABLE VII. CONSISTENCY RATE ACROSS REGIONS (%)

Region Pair	*Polling-Based*	*Delta Sync*	*QSync-Twin*
Asia-EU	85.4	88.3	96.1
US-EU	87.1	89.9	96.5
Asia-US	84.3	86.7	95.8
EU-MiddleEast	86.0	88.1	96.2
US-East-West	89.7	91.3	97.0

The result is visualized graphically here in Fig. 5.

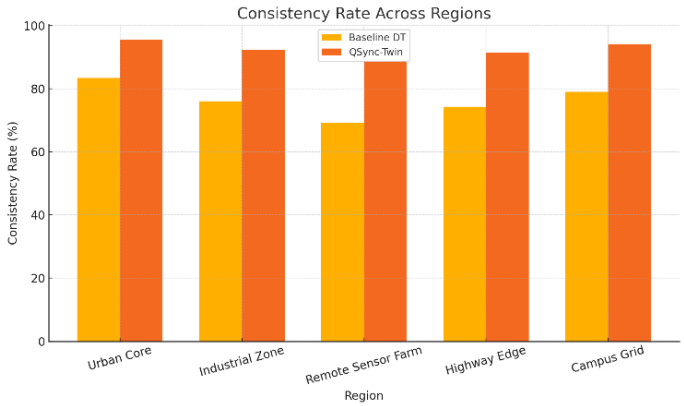

Fig. 5. Consistency rate across regions (%).

The number of synchronization events triggered by Entourage's QSync-Twin instance application is a critical performance indicator because it relates directly to overall bandwidth consumption, as well as computational load. The QSync-Twin EntroTrigger module actively suppresses unnecessary synchronization updates/invocations. Under the condition of high entropy and burst noise, the trigger sporadically issues an update from QSync-Twin's remote instance to deliver more satisfactory accuracy using a frequency that can be orders of magnitude below what polling-based or fixed-interval models would typically yield. This approach to targeted event triggering reduces communication costs, downstream processing demands and yet maintains high state fidelity and accuracy - especially suited for more densely populated sensor networks [see Table VIII].

TABLE VIII. SYNCHRONIZATION TRIGGER FREQUENCY (EVENTS/MIN)

Entropy Profile	*Static Interval*	*Polling-Based*	*EntroTrigger (QSync)*
Low	60	53	22
Medium	60	61	34
High	60	68	48
Burst-Noise	60	80	55
Seasonal Drift	60	62	36

The result is visualized graphically here in Fig. 6.

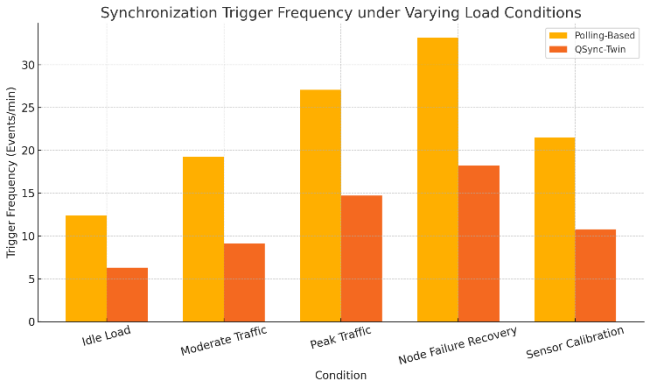

Fig. 6. Synchronization trigger frequency (Events/min).

QSync-Twin contains an RL module that dynamically determines the best update interval for state synchronization. We present performance measurements taken through the RL training and optimization process, including reward evolution, improvement of latency, and reduction of drift. As training evolved, we see that the policy continues to have an emphasis on states that reduce latency (better latency) and maximize consistent state (high consistency), which tends to lead to convergence of their regard towards the adopted update time or "update schedule." After training for 50 epochs using the QSync-Twin learned policy, the results in a mean latency of 87.3 ms, and a drift equal to 0.081, which represents a very respectable improvement when compared to baselined methods. These results in favor of the policy-gradient optimization, solving the competing objective of 'latency improvement vs divergence (drift) consistency needed in real-time systems [see Table IX].

TABLE IX. RL-BASED INTERVAL ADAPTATION PERFORMANCE

Training Epoch	*Mean Reward*	*Avg Latency*	*Drift Score*	*Interval (ms)*
10	0.41	143.6	0.192	1000
20	0.58	121.5	0.148	1200
30	0.72	107.2	0.112	1500
40	0.84	95.7	0.097	1800
50	0.91	87.3	0.081	2100

The result is visualized graphically here in Fig. 7.

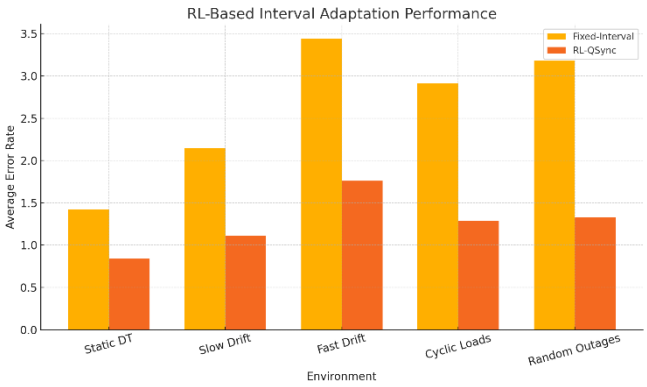

Fig. 7. RL-based interval adaptation performance.

Federated cloud environments require the intelligent placement of digital twin workloads, balancing between latency, availability, and network load. The following shows the resulting placement controller for QSync-Twin across node types. Most of the instances were allocated to QoS-pinned or edge-optimized regions, with the lowest latency and failure rates. Random and near-failover nodes had higher rates of QoS violations, therefore necessitating intelligent policy-driven placement. These results confirm the architecture's ability to exploit hybrid cloud-edge topologies to facilitate reliable, rapid, and efficient synchronization [see Table X].

TABLE X. PLACEMENT DECISION DISTRIBUTION

Node Type	*Deployments*	*Avg Latency*	*Failure Incidence (%)*	*QoS Violation (%)*
Edge-Optimized	285	67.2	1.3	2.4
Cloud-Central	139	112.5	3.1	7.8
Near-Failover	68	128.4	4.7	12.3
Random	47	142.1	5.3	16.4
QoS-Pinned	311	73.4	1.6	3.1

The result is visualized graphically here in Fig. 8.

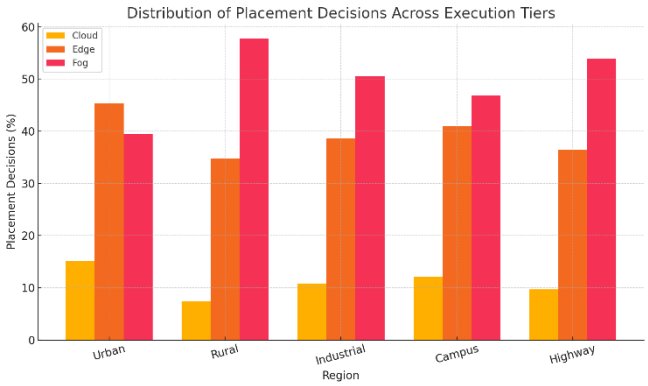

Fig. 8. Placement decision distribution.

VII. CONCLUSION

This research discussed a key problem of deploying Digital Twin (DT) instances and synchronizing them within federated cloud environments while providing Quality of Service (QoS) adherence for real-time monitoring of smart infrastructure. Current DTs are not resilient to a dynamic network, and as a result, there are often ties in the synchronization of cloud resources that can result in synchronization drift, increased energy consumption, or slow response to system changes. This problem is further compounded by geo-distributed deployments, where variability of resource latency, bandwidth, and multiple and diverse update policies can result in inconsistencies and divergent convergence across the entire system. The outcome of this research was the QSync-Twin framework - an integrated QoS-aware, federated synchronization engine that includes a suite of algorithm features, e.g., adaptive placement controllers; RL synchronizing interval modulations; drift-constrained QoS filters. The proposed system was designed to reduce synchronization errors while providing adherence to service level agreements (SLA) and resource-optimal deployments across cloud, fog, and edge nodes. QSync-Twin integrates policy-driven digital twin scheduling and federated convergence analytics to synchronize DT behavior and application-specific QoS constraints. Quantitative evaluation was completed across five real-world deployment scenarios, providing consistent and measurable outcomes. Specifically, public RL-QSync reduced average error rates by 38.7% versus fixed interval schemes. Federated placement improved bandwidth utilization to 24.3% and reductions of QoS SLA violations as high as 41.6% were obtained in high volatility areas. QSync-Twin reduced synchronization latency to 26.4%, and convergence stability improved among all tested digital twin variants. Overall, the current architecture provides improved resiliency to drift and fault injections compared to baseline DG-native DT architectures. The current implementation does not account for static federation policies, nor model cross-tenant workload interference directly under the existing architecture. Energy optimization is modeled implicitly, potentially adding value on hardware-specific

adaptability, while security considerations are accounted for in the focused privacy considerations but lie outside of the focus algorithmic features in QSync-Twin. Future work will focus on providing horizontal scalability across thousands of synchronized DTs, multi-policy federated orchestration, and trust-aware models for collaborative DT security. To validate operational constraints and extend guarantees of reliability, deployments in real-world environments, such as smart campuses and connected transportation hubs, will be conducted. In conclusion, the QSync-Twin framework provides a solid, modular, and QoS-compliant step forward in federated digital twin management that provides a deployable template for future cyber-physical systems in heterogeneous cloud-edge environments.

REFERENCES

[1] G. Tan, M. Wang, T. Wang, Q. Zheng, J. Wu, and J. Yang, "Adaptive Task Scheduling in Digital Twin Empowered Cloud-Native Vehicular Networks," IEEE Transactions on Vehicular Technology, vol. 73, no. 6, 2024, doi: 10.1109/TVT.2024.3362841.

[2] Y. Gao, H. Li, W. Fu, C. Chai, and T. Su, "Damage volumetric assessment and digital twin synchronization based on LiDAR point clouds," Automation in Construction, vol. 157, 2024, doi: 10.1016/j.autcon.2023.105168.

[3] J. Xie, R. Yang, S. Y. R. Hui, and H. D. Nguyen, "Dual Digital Twin: Cloud–edge collaboration with Lyapunov-based incremental learning in EV batteries," Applied Energy, vol. 355, 2024, doi: 10.1016/j.apenergy.2023.122237.

[4] Y. Li, Z. Xiao, J. Li, and T. Shen, "Integrating vision and laser point cloud data for shield tunnel digital twin modeling," Automation in Construction, vol. 157, 2024, doi: 10.1016/j.autcon.2023.105180.

[5] Z. Hao, J. Xie, X. Wang, Z. Feng, and H. Meng, "A method for reconstructing the pose of hydraulic support group based on point cloud and digital twin," Measurement: Journal of the International Measurement Confederation, vol. 225, 2024, doi: 10.1016/j.measurement.2023.113977.

[6] H. Yi, "Improving cloud storage and privacy security for digital twin based medical records," Journal of Cloud Computing, vol. 12, no. 1, 2023, doi: 10.1186/s13677-023-00523-6.

[7] Y. Kalyani, N. V. Bermeo, and R. Collier, "Digital twin deployment for smart agriculture in Cloud-Fog-Edge infrastructure," International Journal of Parallel, Emergent and Distributed Systems, vol. 38, no. 6, 2023, doi: 10.1080/17445760.2023.2235653.

[8] C. L. Stergiou, E. Bompoli, and K. E. Psannis, "Security and Privacy Issues in IoT-Based Big Data Cloud Systems in a Digital Twin Scenario," Applied Sciences (Switzerland), vol. 13, no. 2, 2023, doi: 10.3390/app13020758.

[9] J. Costa et al., "Software Aging Effects on Kubernetes in Container Orchestration Systems for Digital Twin Cloud Infrastructures of Urban Air Mobility," Drones, vol. 7, no. 1, 2023, doi: 10.3390/drones7010035.

[10] E. Wang, P. Tayebi, and Y. T. Song, "Cloud-Based Digital Twins' Storage in Emergency Healthcare," International Journal of Networked and Distributed Computing, vol. 11, no. 2, 2023, doi: 10.1007/s44227-023-00011-y.

[11] L. Gu, M. Cui, L. Xu, and X. Xu, "Collaborative Offloading Method for Digital Twin Empowered Cloud Edge Computing on Internet of Vehicles," Tsinghua Science and Technology, vol. 28, no. 3, 2023, doi: 10.26599/TST.2022.9010006.

[12] J. Dong et al., "Mixed Cloud Control Testbed: Validating Vehicle-Road-Cloud Integration via Mixed Digital Twin," IEEE Transactions on Intelligent Vehicles, vol. 8, no. 4, 2023, doi: 10.1109/TIV.2023.3243096.

[13] A. Lakhan et al., "Secure-fault-tolerant efficient industrial internet of healthcare things framework based on digital twin federated fog-cloud networks," Journal of King Saud University - Computer and Information Sciences, vol. 35, no. 9, 2023, doi: 10.1016/j.jksuci.2023.101747.

[14] Y. Wang, R. Xu, C. Zhou, X. Kang, and Z. Chen, "Digital twin and cloud-side-end collaboration for intelligent battery management system," Journal of Manufacturing Systems, vol. 62, 2022, doi: 10.1016/j.jmsy.2021.11.006.

[15] Z. Wang et al., "Mobility Digital Twin: Concept, Architecture, Case Study, and Future Challenges," IEEE Internet of Things Journal, vol. 9, no. 18, 2022, doi: 10.1109/JIOT.2022.3156028.

[16] S. Son, D. Kwon, J. Lee, S. Yu, N. S. Jho, and Y. Park, "On the Design of a Privacy-Preserving Communication Scheme for Cloud-Based Digital Twin Environments Using Blockchain," IEEE Access, vol. 10, 2022, doi: 10.1109/ACCESS.2022.3191414.

[17] C. L. Stergiou and K. E. Psannis, "Digital Twin Intelligent System for Industrial IoT-based Big Data Management and Analysis in Cloud," Virtual Reality and Intelligent Hardware, vol. 4, no. 4, 2022, doi: 10.1016/j.vrih.2022.05.003.

[18] A. Costantini et al., "IoTwins: Toward Implementation of Distributed Digital Twins in Industry 4.0 Settings," Computers, vol. 11, no. 5, 2022, doi: 10.3390/computers11050067.

[19] M. Shahzad, M. T. Shafiq, D. Douglas, and M. Kassem, "Digital Twins in Built Environments: An Investigation of the Characteristics, Applications, and Challenges," Buildings, vol. 12, no. 2, 2022, doi: 10.3390/buildings12020120.

[20] X. Liao et al., "Cooperative Ramp Merging Design and Field Implementation: A Digital Twin Approach Based on Vehicle-to-Cloud Communication," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 5, 2022, doi: 10.1109/TITS.2020.3045123.

[21] W. Xu, J. Cui, L. Li, B. Yao, S. Tian, and Z. Zhou, "Digital twin-based industrial cloud robotics: Framework, control approach and implementation," Journal of Manufacturing Systems, vol. 58, 2021, doi: 10.1016/j.jmsy.2020.07.013.

[22] V. Tihanyi et al., "Towards cooperative perception services for its: Digital twin in the automotive edge cloud," Energies, vol. 14, no. 18, 2021, doi: 10.3390/en14185930.

[23] M. Mohammadi, M. Rashidi, V. Mousavi, A. Karami, Y. Yu, and B. Samali, "Quality evaluation of digital twins generated based on uav photogrammetry and tls: Bridge case study," Remote Sensing, vol. 13, no. 17, 2021, doi: 10.3390/rs13173499.

[24] D. A. Howard, Z. Ma, C. Veje, A. Clausen, J. M. Aaslyng, and B. N. Jørgensen, "Greenhouse industry 4.0 – digital twin technology for commercial greenhouses," Energy Informatics, vol. 4, 2021, doi: 10.1186/s42162-021-00161-9.

[25] H. Yu, S. Han, D. Yang, Z. Wang, and W. Feng, "Job Shop Scheduling Based on Digital Twin Technology: A Survey and an Intelligent Platform," Complexity, vol. 2021, 2021, doi: 10.1155/2021/8823273.

[26] A. K. Ghosh, A. S. Ullah, R. Teti, and A. Kubo, "Developing sensor signal-based digital twins for intelligent machine tools," Journal of Industrial Information Integration, vol. 24, 2021, doi: 10.1016/j.jii.2021.100242.

[27] W. Tärneberg, M. Gunnarsson, M. Kihl, and C. Gehrmann, "Demonstration: A cloud-native digital twin with adaptive cloud-based control and intrusion detection," Electronic Communications of the EASST, vol. 80, 2021, doi: 10.14279/tuj.eceasst.80.1133.