Active Inference for Digital Twins: Predicting and Optimizing IoT Processing Service Performance

Elena Pretel
University of Castilla-La Mancha
Albacete, Spain
mariaelena.pretel@uclm.es

Boris Sedlak
Vienna University of Technology
Vienna, Austria
b.sedlak@dsg.tuwien.ac.at

Víctor Casamayor-Pujol
Universitat Pompeu Fabra
Barcelona, Catalonia, Spain
victor.casamayor@upf.edu

Elena Navarro
University of Castilla-La Mancha
Albacete, Spain
Elena.Navarro@uclm.es

Víctor López-Jaquero
University of Castilla-La Mancha
Albacete, Spain
VictorManuel.Lopez@uclm.es

Pascual González
University of Castilla-La Mancha
Albacete, Spain
Pascual.Gonzalez@uclm.es

Schahram Dustdar
Vienna University of Technology
Vienna, Austria
Universitat Pompeu Fabra
Barcelona, Spain
dustdar@dsg.tuwien.ac.at

Abstract

Digital Twins (DTs) are emerging as enablers for real-time monitoring and control in IoT systems. In this work, we propose a DT enabled with Active Inference (AIF), a framework rooted in the Free Energy Principle, to endow DTs with predictive and decision-making capabilities. Our approach is evaluated on a realistic edge computing scenario where two co-located video processing services—a computer vision pipeline using YOLOv8 and a QR code reader—compete for limited CPU resources. The DT continuously infers physical twin state, predicts future performance, and selects actions to meet Service Level Objectives (SLOs). In a series of experiments, we validate the predictive accuracy and control effectiveness of the AIF-enabled DT. Notably, the DT achieves a cumulative SLO fulfillment above 80% for both services, and predicted throughput trajectories show high alignment with real observations, confirmed through statistical testing.

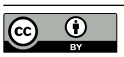
CCS Concepts

• **Computer systems organization → Real-time system architecture**.

Keywords

Digital Twins, Predictability, Active Inference, Internet of Things, IoT

ACM Reference Format:
Elena Pretel, Boris Sedlak, Víctor Casamayor-Pujol, Elena Navarro, Víctor López-Jaquero, Pascual González, and Schahram Dustdar. 2025. Active Inference for Digital Twins: Predicting and Optimizing IoT Processing Service Performance. In *The 15th International Conference on the Internet of Things (IOT 2025), November 18–21, 2025, Vienna, Austria.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3770501.3770527



IOT 2025, Vienna, Austria

ACM ISBN 979-8-4007-1595-2/25/11
https://doi.org/10.1145/3770501.3770527

1 Introduction

The fast advancement of technology over the last few decades has transformed the way systems are monitored, controlled, and optimized in diverse domains: from manufacturing and logistics to smart cities and healthcare. The rise of smart devices, Internet of Things (IoT), Artificial Intelligence (AI), and Cloud Computing have enabled the emergence of Digital Twins (DTs).

In large-scale and distributed IoT environments, such as smart manufacturing lines, intelligent transportation systems, or e-health care infrastructures, managing the complexity of the system becomes a fundamental challenge [13]. DTs provide a suitable way to monitor, model and manage these IoT systems by creating virtual replicas of physical entities operating under real-time data flows enabling proactive and adaptive decision making [7].

The DT concept was originally coined by Grieves and Vicker in 2003 defined as the coexistence of three elements: (1) a physical space, (2) a virtual space, and (3) a bidirectional connection between both spaces allowing their convergence [4]. Since then, the concept has evolved, currently, DTs are understood as dynamic virtual representations of physical entities that allow real-time monitoring, simulation, prediction, and control of the modelled entities' behaviour [8]. One of the main properties of a DT is the *predictability* [12], which refers to the ability of the DT to anticipate and improve the future behaviour of its physical twin [8]. This property is crucial to enable intelligent decision-making within the DT, in particular when leveraging AI-based techniques.

This property is deeply related to *Active Inference* (AIF), an algorithm rooted in Bayesian brain theory and the Free Energy Principle (FEP) [1]. The AIF algorithm operates on the premise that an agent

minimizes uncertainty about its environment by reducing "surprise", that is, the discrepancy between predicted and observed outcomes. By using probabilistic generative models, AIF allows a DT to anticipate future states and plan optimal responses. For instance, an AIF-enabled DT can simulate multiple hypothetical or counterfactual scenarios, such as the impact of varying resource allocations on system performance, and select actions that align with its goals while minimizing the uncertainty.

AIF enables DTs to not only simulate future behaviour but also to continuously refine their predictions by actively interacting with their corresponding physical twins. By combining perception, prediction, and action, AIF enables DTs to dynamically interpret inputs, simulate future states, and execute deliberate actions aimed at achieving desired outcomes. This ensures that the DT maintains an accurate and evolving representation of its physical twin. This approach contrasts with traditional deterministic or machine learning models, which often lack the adaptability required to handle dynamic and uncertain scenarios.

Despite the potential of the AIF-enabled DT, its application is a challenge. To date, there are no known proposals in the literature (source: Scopus, date: July 2025) that use AIF for creating DTs. This is a significant research gap that is addressed in this work. Our hypothesis is that integrating Active Inference into the architecture of Digital Twins enables them to effectively predict the future behaviour of their physical twins and autonomously make decisions to ensure that their performance objectives are met. Based on this hypothesis, we formulate the following research questions (RQs): **RQ1**: Can an AIF-enabled DT accurately predict the performance of its physical twin? and **RQ2**: Can an AIF-enabled DT select actions that ensure the achievement of performance objectives under dynamic and uncertain conditions?

To answer these questions, we designed and executed a series of experiments using a case study that involved real-world video processing services deployed on a resource-constrained edge device. The AIF-enabled DT is tasked with predicting throughput values and making decisions to meet predefined service objectives. We first evaluate the impact of enabling model learning and calibrate the optimal learning rate for the AIF agent's transition model. Then, we assess the DT's predictive accuracy by comparing its estimated throughput with the real values observed on the physical system, using statistical tests to quantify alignment. Finally, we test whether the DT can effectively guide the system to fulfill its performance objectives over time. The results show statistical evidence that an AIF-enabled DT can both provide an accurate prediction of future behaviour and meet the stated performance objectives.

The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 introduces the theoretical foundations of AIF and its relevance to the predictability DT property. Section 4 presents the case study used to validate the proposed approach. Section 5 shows the DT architecture proposed in this work. Section 6 details the methodology for implementing the predictability property using AIF. Section 7 describes the experimental setup and presents the experiments and the results. Finally, Section 8 shows the discussion and Section 9 concludes the work.

2 Related Work

According to Minerva et al. [8], the predictability property of a Digital Twin (DT) refers to its ability to anticipate and improve the future behaviour of its physical counterpart. This capability is achieved by embedding artificial intelligence algorithms that use historical and real-time data to generate simulations of future states. Predictability has thus become a central objective in the evolution of DTs, particularly with the integration of AI techniques [18], enabling decision-making and performance optimisation across physical assets and complex systems such as cities, factories, and logistics networks [8]. Industrial applications have already explored this property: DTs have been used in Virtual Testbeds to carry out controlled experiments [16], while large-scale organisations such as NASA apply them in aerospace projects for system validation and forecasting [16]. Findings in the IoT domain also reveal a strong interest in exploiting simulation and prediction to enhance the adaptability and resilience of interconnected infrastructures [6].

Despite these advances, achieving effective predictability remains a challenge, particularly in dynamic and uncertain environments where most approaches still rely on reactive models with limited ability to generalise or adapt in real time [21]. To overcome this, DTs require AI methods capable of reasoning under uncertainty, updating internal models continuously, and selecting actions that not only predict but also shape outcomes. In this context, Active Inference (AIF) [2] stands out as a promising paradigm, offering a unified framework for perception, prediction, and action that pushes DTs beyond passive forecasting toward active control of complex systems.

3 Active Inference for Digital Twins

Active Inference (AIF) is an algorithm rooted in cognitive neuroscience and based on the Free Energy Principle (FEP), which posits that agents must reduce uncertainty about their environment in order to remain viable [1]. This is achieved by minimizing a quantity known as *variational free energy*, which allows the agent to resist entropy by keeping its sensory states within the expected bounds, thus preserving homeostasis and adaptability over time [2].

AIF distinguishes between two key concepts: the *generative process* and the *generative model* [3]. The generative process refers to the actual, but hidden, dynamics of the environment that produce observations. Since the states are not directly observable, the agent must infer them from the observations. In contrast, the generative model is an internal probabilistic representation maintained by the agent to explain and predict sensory input. It encodes the agent's beliefs about how observations arise from hidden states and how its own actions affect future states and outcomes. If the model's predictions diverge significantly from actual observations, producing *surprise*, the agent updates its model to reduce this discrepancy.

AIF unifies three core processes [3]: *perception*: updating beliefs about hidden states; *action*: selecting behaviours that are expected to reduce future surprise; and *learning*: adjusting the generative model based on accumulated experience. These three processes work together to minimise the *expected free energy*, thereby enabling the agent to adaptively align its internal model with its environment.

The expected free energy is composed of two values [17]: (1) the *epistemic value*, which is the expected information gain or reduction

in uncertainty about hidden states, and (2) the *extrinsic value*, which aligns with preferred outcomes, reflecting goal-directed behaviour. By balancing exploration (epistemic value) and exploitation (extrinsic value), AIF promotes behaviours that reduce uncertainty.

3.1 AIF Principles for Predictability DT property

The *predictability* property of DTs is deeply rooted in the inferential processes of AIF. Below, we present our view of the intersection between AIF and predictability, highlighting how AIF concepts are relevant to the development of this DT property:

- **Adaptive perception**: the DT not only receives data from the physical twin, but actively interprets its meaning, generating probabilistic beliefs about its actual state. This involves continuously updating its internal generative model to reflect the most likely explanations for incoming sensory data, while accounting for uncertainty. This ability to dynamically interpret and contextualize environmental inputs allows the DT to maintain an accurate and evolving representation of its physical twin, ensuring that its decisions are grounded in a coherent and up-to-date understanding of reality.
- **Active simulation and prediction**: using an internal generative model, the DT can simulate possible future states and evaluate their consequences. This active simulation process enables the DT to anticipate relevant events and plan appropriate responses in advance. For example, the DT can explore multiple hypothetical scenarios, such as the impact of different actions on the physical twin's state or the likelihood of certain environmental changes. By predicting future outcomes, the DT can identify optimal strategies that align with its goals while minimizing uncertainty.
- **Deliberate action**: thanks to the AIF, the actions of the DT are not reactive responses but strategies aimed at reducing the discrepancy between its predictions and observations, adjusting both its beliefs and its behaviour. For instance, if the DT observes a deviation from its expected state, it may take corrective actions to bring the physical twin back into alignment with its predictions.
- **Continuous learning**: The generative model underlying the DT is not static but evolves over time through continuous learning. This learning process involves updating the model according to experience, allowing the DT to progressively refine its understanding of the physical twin and its own limitations or capabilities. For example, as the DT interacts with its physical twin, it accumulates evidence that either confirms or challenges its existing beliefs, prompting adjustments to its internal representations.

4 Problem Statement

This work focuses on developing the *predictability* property of DTs using Active Inference (AIF). Therefore, we investigate whether an AIF-enabled DT can serve as a predictive and decision-making controller capable of predicting the physical twin behaviour (**RQ1**) and acting according to predefined performance objectives (**RQ2**). To evaluate this, we implement a case study involving two services deployed on a multi-core edge device with limited CPU resources. In IoT systems, efficiently managing computational resources is essential to maintain optimal performance under dynamic workloads and constrained environments [10].

The physical twin of our case study corresponds to an edge device equipped with 8 CPU cores, hosting two concurrent services whose runtime behaviour and resource usage can be dynamically reconfigured. The two services deployed are:

(1) **Video Stream Inference Service**. This service is a computer vision (CV) pipeline that processes a continuous stream of video frames using YOLOv8 object detection model, a deep neural network from Ultralytics [20]. This service detects objects in real time. The configuration parameters include *quality* which determines the ingested video resolution, *model size* which describes the Yolo model, and the *number of CPU cores* which determines the maximum resources allocated. The service *throughput* is measured in frames per second (FPS).
(2) **QR Code Reading Service**. This service is implemented using OpenCV2 [9] and scans each individual video frames for the presence of QR codes. The configuration parameters here include the *quality* and the *number of CPU cores* allocated. Unlike the CV service, the QR reader does not use a specific model size. Similar to the previous one, the service *throughput* is measured in FPS too.

Both services are designed to be dynamically configurable in terms of the number of cores allocated, the level of processing quality, and the size of the model used (in the case of CV). This flexibility is key to intervene at runtime with resource reallocation decisions. On the other hand, the physical twin provides observable data such as throughput, CPU usage, quality and model size that the DT monitors.

This setup provides a realistic and challenging scenario to evaluate the effectiveness of an AIF-enabled DT, particularly its ability to predict the future behaviour and proactively adapting configurations to meet the Service Level Objectives (SLOs).

5 Architecture of the AIF-enabled DT

Figure 1 depicts the main components of the architecture of the AIF-enabled DT proposed in this work. The physical twin is an edge device that hosts two concurrent services whose resource allocation can be modified at runtime. The DT continuously monitors the physical twin through a Prometheus Database that collects key runtime metrics of the physical twin. These time-series data are used by the DT to maintain an up-to-date internal representation of the two services, as described in the previous Section. At the core of the DT lies an AIF agent, which applies variational Bayesian inference to minimize expected free energy and infer the most suitable configuration actions to take on the physical twin. These decisions are applied through a control interface that enables the DT to directly influence the two services. This closed-loop interaction enables the DT to perceive, reason, and act upon its physical twin in real time. The AIF-enabled DT proposed addresses all the five properties of a DT described in Section 1:

The DT-enabled with AIF fulfils the three essential properties defined by Minerva et al. [8], as well as two additional properties

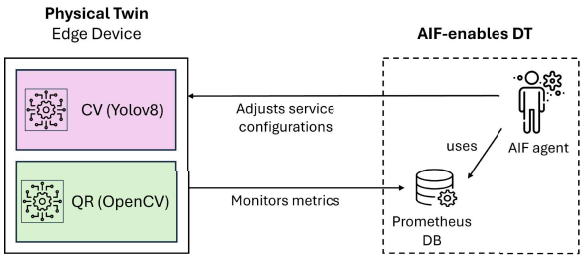


Figure 1: Digital Twin architecture with integrated AIF agent, Prometheus monitoring, and an edge device as physical twin.

that add value. The following explains each property and how it is achieved in our design:

(1) **Representativeness and Contextualisation**: This property requires that the DT captures only the attributes of the physical twin that are relevant to its intended purpose. In our approach, the DT focuses on the variables of the CV and QR services shown in Table ?? and Table ??.
(2) **Reflection**: This property ensures that the value of each attribute is properly reflected in the DT. The AIF-enabled DT continuously collects data from the physical services, which ensures that the DT accurately reflects the current state of the physical twin.
(3) **Entanglement**: This property requires a bidirectional flow of information between the physical and digital twins. The AIF-enabled DT has two communication channels: (1) a monitoring channel for collecting real-time data, and (2) an action channel for applying decisions made by the AIF agent to the physical services.
(4) **Memorization**: This property refers to the ability of the DT to store both current and historical data from the physical twin. This is achieved by persistently storing data in Prometheus database [14].
(5) **Predictability**: As previously discussed, this property refers to the DT's ability to anticipate and improve the future behaviour of its physical twin. In our architecture, an embedded AIF agent is responsible for predicting future states of the services and selecting optimal actions to meet SLOs. The development of this property through AIF is further discussed in the following Section.

6 Proposed methodology for developing predictability of DT using AIF

This Section presents the methodology for implementing the predictability property of an AIF-enables DT. The methodology is structured into three main steps which form a generalizable pipeline applicable across different DT use cases. In what follows, we detail each step and describe its concrete application to the case study described in Section 4.

6.1 Relevant data identification

The first step in the development of the AIF agent, which is integrated into the DT architecture, is to identify the key elements that structure the agent's reasoning and behaviour. In the context of our case study, this is instantiated as follows:

- **Goal elicitation and Preferences**: The DT has specific performance objectives G_{DT}. These objectives are formalised as preferences over observations. For example, in our DT, the CV service prefers a throughput ≥ 3 while the QR reading service prioritises a throughput ≥ 40 (see Table ?? and Table ??). Neither service has any preferences regarding CPU usage.
- **States and Observations identification**: The state space S_{DT} and observation space O_{DT} are defined around the configurable parameters and performance metrics of CV and QR service (see Table ?? and Table ??). In this work, we assume perfect sensing, so S_{DT} and O_{DT} are considered equivalent and include the next variables: allocated CPU cores of CV and QR, video resolution quality of CV and QR, model size of CV, and throughput of CV and QR.
- **Actions identification**: *Actions* (U_{DT}) represent possible reconfigurations the DT can apply, that is: increasing, staying or decreasing core allocation for CV and QR; increasing, staying or decreasing quality for CV and QR; and increasing, staying or decreasing model for CV (see Table ?? and Table ??).

6.2 Generative model construction

The next step is to construct the generative model. An AIF agent can be formally modelled as Partially Observable Markov Decision Processes (POMDPs) [15], which is represented by the tuple (S,U,B,O,A,C,D), where S is the set of states of the system, U is the set of available actions$U$ is the set of available actions, B encodes the state transition probabilities, O is the set of possible observations, A is the observation likelihood model, C defines preferences over observations and D is the prior distribution over initial state

State space (S). The state space S is defined over seven factors $S_i \in S$ that describe the configuration and performance of both services. In order to make the inference computationally feasible, each configuration factor S_i is defined over a finite set of discrete values. The subscript i identifies the factor and the superscript denotes the cardinality, that is, the number of discrete values it can take. Each factor is defined as follows: CV throughput (S_0^5), CV quality (S_1^8), CV model size (S_2^5), CV cores assigned (S_3^7), QR throughput (S_4^6), QR quality (S_5^8) and QR cores assigned (S_6^7). The values of each factor can be seen in Table ?? and Table ??. The complete state space $\boldsymbol{S}$ is defined as the Cartesian product of all individual factors:

$$S = S_0 \times S_1 \times S_2 \times S_3 \times S_4 \times S_5 \times S_6$$

Therefore, the state of the agent is a tuple $\mathbf{s} = (s_0, s_1, s_2, s_3, s_4, s_5, s_6)$, where $\forall s_i \in S_i$ encodes the current value of the corresponding factor.

Action space (U). Each service has its own independent action set composed of discrete operations that adjust its configuration parameters by one step. This design limits the exponential growth of the joint action space. The **CV service actions (U_{CV})** include 7 discrete options: no-op, increase/decrease quality, increase/decrease model size, increase/decrease cores. The **QR service actions (U_{QR})** include 5 discrete options: no-op, increase/decrease quality, increase/decrease cores. Both increase and decrease actions shift the current value by one step within the discretized space (e.g., increasing CV quality from 160 to 192). The special *no-op* action maintains the current setting. The overall joint action which allows

simultaneous control of both services is defined as:

$$u = (u^{CV}, u^{QR})$$

Observations (O). The set of observations represents the possible outputs of the physical twin produced after an action u has been taken. In this work we assume perfect sensing, that is, an identity mapping between states and observations. To this end, the observation space O is equivalent to the state space S.

Likelihood model (A). The agent receives one observation per state factor. As explained earlier, since we assume fully observable outputs (i.e., perfect sensors), each observation variable corresponds directly to its associated hidden state factor. Therefore, the Likelihood model A is composed of identity matrices.

Transition model (B). Each state factor S_i is governed by a transition model B_i, which defines the probability distribution over its future values $s_i^{t+1} \in S_i$, conditioned on the current state $s_i^t \in S_i$ and the actions applied. These transition models capture how each factor evolves over time, reflecting both the internal logic of the physical system and its operational constraints.

The transition model B_i is represented as a multidimensional tensor whose entries specify, for each combination of current state and action, the probabilities of transitioning to all possible next states. This representation allows encoding deterministic updates (where a specific outcome always follows a given condition) as well as stochastic behaviour (where multiple future states are possible with varying likelihoods).

For example, B_1 transition model for CV quality is affected by the current value of CV quality state factor (s_1) and by the CV action u^{CV}. The full transition model B_1 is a 3D tensor of dimensions $8 \times 8 \times 3$, where 8 is the number of discrete quality levels, and 3 is the number of actions for the CV quality u^{CV} (increase, maintain, decrease). Each entry of the tensor $B_1[i,j,u] = 1$ indicates a deterministic transition from state $s_1^t = S_1[j]$ to $s_1^{t+1} = S_1[i]$ when applying an action u^{CV}. When the AIF agent chooses the action $u^{CV}_{\text{increase quality}}$, the quality level increases by a fixed step of 32 (see Table ??), unless it is already at the maximum allowed value. This transition tensor is constructed as follows:

$$B_1[s_1^{t+1}, s_1^t, u^{CV}_{\text{increase quality}}] = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Rows represent next values $s_1^{t+1} \in S_1$, and columns represent current values $s_1^t \in S_1$, where: $S_1 = \{128, 160, 192, 224, 256, 288, 320, 352\}$. The last column models saturation: once the quality reaches 352, further increase actions leave the state unchanged.

An additional example is B_0, corresponding to the CV throughput. This factor is influenced by multiple states: CV quality s_1, CV model size s_2, and CV allocated cores s_3, as well as the applied CV action u^{CV}. Another example is B_3, which models the number of cores allocated to the CV service. Its next state depends on the current core allocation, both CV and QR actions, and the current allocation of QR cores (s_6). The transition must satisfy the constraint $s_3 + s_6 \leq 8$. The full matrix B can be seen in [11]

Preferences (C). Preferences over observations reflect the SLOs:

- CV: throughput ≥ 3, high quality, large model
- QR: throughput ≥ 40, high quality

These preferences are encoded as log-probabilities. For instance, the preferences of the QR throughput state factor (S^4) are encoded as:

$$C_4^{QR} = [-5,\ 1.25,\ 2.5,\ 4,\ 4,\ 4]$$

Initial state (D). It encodes the agent's belief over its initial state for each state factor s_i. For example, both services start with 2 cores each, so the initial belief distributions over the state factors S_3 (CV cores) and S_6 (QR cores) are:

$$D_3^{CV} = [0,1,0,0,0,0,0,0], \quad D_6^{QR} = [0,1,0,0,0,0,0,0]$$

6.3 Active Inference Cycle

The DT runs a continuous Active Inference cycle. This cycle enables the DT to perceive, reason, and act under uncertainty while managing the joint behaviour of both services within a shared resource environment. In our case study, the cycle is instantiated as follows:

(1) **Perception (Infer State)**: The AIF agent receives a observation from the physical twin and infers the most probable hidden state configuration.
(2) **Prediction and Planning**: Using its internal generative model, the AIF agent simulates future trajectories of system behaviour under different policies π. Each policy defines a joint configuration of actions for both services. For each candidate policy, the AIF agent computes expected free energy $G(\pi)$, which balances epistemic value (uncertainty reduction) and extrinsic value (goal achievement based on performance preferences).
(3) **Action Selection and Execution**: The AIF agent selects the policy π^* that minimises expected free energy and applies the corresponding configuration to the physical twin.
(4) **Learning**: The AIF agent updates the parameters of its generative model, particularly the **B** matrix, to better reflect observed transitions in the physical system over time.

This AIF cycle allows the AIF-enabled DT to continuously adapt its control strategy based on real-time data and internal beliefs, optimizing resource allocation and service performance in a dynamic, constrained edge computing environment.

7 Evaluation and Results

This section presents the experimental evaluation of the proposed AIF-enabled DT and assesses its ability to predict and optimize the performance of CV and QR services running on a physical edge device.

7.1 Experiment design and setup

To validate the proposed approach, we implemented an AIF pipeline using the `pymdp` library [5], which governs the behaviour of the DT that manages the two services deployed on a physical twin: a video stream inference service and a QR code reading service.

The experimental setup is a realistic edge computing scenario with shared and limited computational resources.

The physical twin is an 8 core edge device, where both services are containerized by using Docker and executed concurrently. Each service exposes configurable parameters, such as video quality, number of CPU cores, and model size (for the CV service), as well as observable outputs like the throughput (frames per second). See Section 4 for more details.

The DT receives real-time observations from both services and uses its generative model of AIF to infer latent states and determine optimal actions. The experiment consists of executions composed of 50 iterations, that is, 50 AIF cycles. At each iteration, the AIF-enabled DT receives an observation, infers the state, evaluates candidate policies, and selects the one that minimizes the expected free energy. The selected actions are then applied to the physical twin, and resulting observations are collected to complete the loop.

All experiments were executed on *Galgo*, a high-performance computing system maintained by the University of Castilla-La Mancha (UCLM) [19]. The experiments were deployed on a selection of 48 nodes within the Galgo supercomputer. Each node provides 64 GB of RAM and two Intel Xeon E5-2650 processors at 2.00 GHz, with 8 cores per node. This supercomputer provided a controlled and reproducible environment for running the experiments, managing the physical twin remotely.

7.2 Tuning parameters

Before proceeding with the main experiments addressing the research questions (RQs), we conducted two preliminary experiments to configure the AIF-enabled DT and understand the impact of model adaptation and learning parameters on performance.

7.2.1 Learning vs. Fixed Transition Model. The objective of this experiment is to assess whether an AIF-enabled DT that dynamically updates its transition model (**B** matrix) in real time achieves better prediction accuracy than one using a fixed model. To this end, two configurations were compared: one with learning enabled, where the DT updates the **B** matrix at each time step based on the observed transitions; and one with learning disabled, where the **B** matrix remains static throughout the episode. The evaluation metric used was the Root Mean Squared Error (RMSE) between the predicted and actual throughput, calculated independently for the CV and QR services.

The average RMSE values obtained across the 50 iterations are as follows:

- *No update B*: CV = 2.2700, QR = 20.9591
- *Update B*: CV = 0.9988, QR = 8.9329

The reduction in prediction error when updating the **B** matrix at each iteration indicates that enabling learning improves the predictive accuracy of AIF-enabled DT in both services. More specifically, the RMSE decreases substantially for both CV and QR. For the CV service, the error drops from 2.27 to 0.99, and for QR, from 20.96 to 8.93. This suggests that learning the **B** matrix allows the model to better approximate the real physical twin behaviour.

7.2.2 Learning Rate Optimisation. The objective of this experiment was to determine the optimal learning rate α for updating the **B** matrix in the AIF-enabled DT. To achieve this, a grid search was performed over a predefined set of candidate values $\alpha \in \{0.1, 0.3, 0.5, 0.8, 1.0\}$. Each configuration was evaluated across 50 iterations, and the learning rate yielding the best overall prediction performance was selected. The evaluation metric used was the RMSE between the predicted and actual throughput for CV and for QR for each learning rate value. The average RMSE values obtained for each α across all iterations are as follows:

- $\alpha = 0.1$: CV = 1.0378, QR = 9.4381
- $\alpha = 0.3$: CV = 0.9107, QR = 8.4020
- $\alpha = 0.5$: CV = 1.0710, QR = 9.9882
- $\alpha = 0.8$: CV = 1.0253, QR = 8.7431
- $\alpha = 1.0$: CV = 1.0009, QR = 9.0860

Since the learning rate $\alpha = 0.3$ produced the lowest prediction error in both services, it was selected as the default setting for the remaining experiments.

7.3 Experiment 1: Prediction Accuracy of Future States

This experiment answers the **RQ1**: *Can an AIF-enabled DT accurately predict the performance of its physical twin?* Accurate prediction of future system states is a fundamental requirement for the predictability property of DT. This experiment assesses the predictive capability of the DT by evaluating how accurately it anticipates the future performance of its physical twin. For each execution, the DT generated throughput predictions for both services over a 50-iteration horizon using its internal generative model. These predicted values were then compared with the actual throughput observed on the physical twin. The primary evaluation metric was the Root Mean Square Error (RMSE) between predicted and real performance values, computed over the last 15 iterations of each run to focus on the post-learning phase. To ensure generalisability, the experiment was repeated with four different random seeds.

Figure 2 shows a representative execution, comparing predicted and real throughput over time for both the CV and QR services. This figure shows the mean and standard deviation of the 30 executions. The close match between both trajectories visually confirms the predictive capabilities of the AIF-enabled DT. The results show that in the first iterations, it learns the behaviour of the physical twin, then, as it progresses through the iterations, a stable alignment between its internal generative model and the actual behaviour of the physical system is seen.

To statistically evaluate the prediction accuracy of the AIF-enabled DT, we tested whether the two samples (predicted and real) are statistically significant different. The descriptive statistics for CV and QR services are shown in Table 1. We formulated the following hypotheses for each service (CV and QR):

- H_0: There is no statistically significant difference between the distributions of predicted and observed throughput values.
- H_1: There is a statistically significant difference between the distributions of predicted and observed throughput values.

Since the normality assumption was violated (as assessed through Kolmogorov-Smirnov test), we employed a non-parametric alternative: the **Mann–Whitney U test**. All four assumptions required to validly apply the Mann-Whitney U test were assessed and found to be satisfied for the two services. Specifically: (1) the dependent

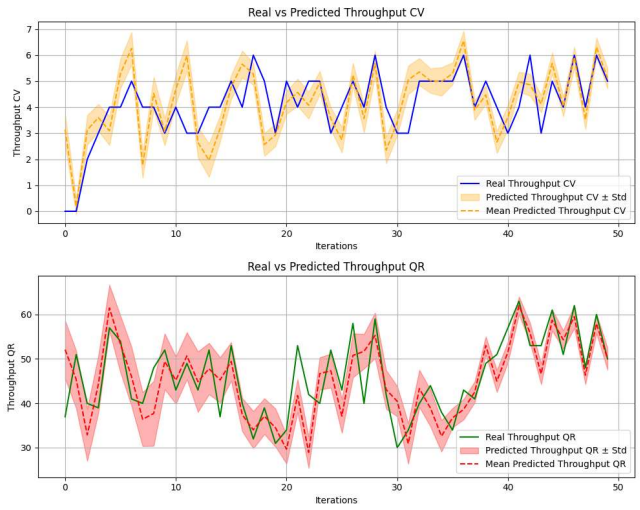


Figure 2: Predicted vs. real throughput for both services.

Table 1: Descriptive statistics for real and predicted throughput values in CV and QR services

Statistic	CV		QR	
	Real	Predicted	Real	Predicted
N	450	450	450	450
Mean	4.66	4.73	52.25	51.74
Median	5.00	4.79	52.00	51.77
Standard Deviation	1.01	1.14	8.07	7.60
Minimum	3.00	2.15	34.00	35.05
Maximum	6.00	7.07	63.00	65.84
Mean Rank	444.57	456.43	465.32	435.67

Table 2: Mann–Whitney U Test Statistics

Statistic	CV	QR
Mann–Whitney U	98580.00	94578.00
Z	-0.688	-1.713
p-value	0.491	0.087

variable was continuous: *throughput value*; (2) the independent variable consisted of two independent and categorical groups: *real* and *predicted*; (3) observations were independent between groups; and (4) the distributions of the two groups exhibited different shapes, therefore, the results will be interpreted as a comparison of mean ranks rather than medians. The test was conducted over the last 15 iterations of 30 executions, comparing the predicted and real values of throughput for both services. The last 15 have been selected since the previous ones are considered for training. The results are summarized below:

For the CV service, as the p-value = 0.491 is higher than 0.05, we fail to reject H_0, indicating that the predicted and real throughput mean rank are statistically indistinguishable. On the other hand, for the QR service, the p-value is 0.087, although this result is close to the 0.05 threshold, it does not provide strong evidence to reject H_0 under the standard significance level, suggesting that the predicted and real values are not significantly different neither. These results support the ability of the AIF-enabled DT to accurately anticipate future system states, addressing RQ1.

Table 3: Descriptive statistics for the cumulative objective fulfilment rate in CV and QR services

Statistic	CV	QR
N	30	30
Mean	0.84	0.88
Median	0.90	0.89
Standard Deviation	0.11	0.06
Minimum	0.66	0.72
Maximum	0.96	0.96

7.4 Experiment 2: Fulfilling the performance service objective

This experiment directly addresses the **RQ2**: *Can an AIF-enabled DT select actions that ensure the achievement of performance objectives under dynamic and uncertain conditions?* Taking actions that modify the configuration of the physical twin to achieve its performance objective is a fundamental requirement for the predictability property of the DT.

The objective of this experiment is to assess whether the AIF-enabled DT can successfully fulfill the predefined performance service-level objectives (SLOs) for both services. Throughout each of the 50 iterations in an execution, we verify whether the throughput exceeds the target threshold, greater than 3 for the CV service and greater than 40 for the QR service. Each iteration in which a condition is met, a counter is incremented. Then, the total count is normalized by the number of iterations to obtain a cumulative SLO fulfillment rate between 0 and 1. This procedure is repeated in 30 independent executions, each initialized with a different random seed, to ensure the robustness and generalizability of the results. The metric used in this experiment is the cumulative SLO fulfillment rate, calculated separately for the CV and QR services.

As shown in Figure 3, which displays 30 independent executions, the AIF-enabled DT consistently meets the defined SLOs. On average, the CV achieves a cumulative fulfillment rate of *0.84*, while the QR service achieves *0.88*. The rest of the descriptive statistics are shown in Table 3. Across all 30 executions, the AIF-enabled DT demonstrates stable and effective behaviour, reliably guiding the physical system toward its operational objectives. To statistically evaluate whether the AIF-enabled DT fulfils the proposed CV and QR performance objective more than 80% of the time, we test whether the median of each service's cumulative SLO fulfilment is equal to this value. We formulated the following hypotheses for each service (CV and QR):

- H_0: There is no statistically significant difference between the median of the observed fulfilment rates and the reference value of 0.8.
- H_1: There is a statistically significant difference between the median of the observed fulfilment rates and the reference value of 0.8, with the median being greater than 0.8.

Since the assumption of normality (assessed by the Shapiro-Wilk tests) is not met in both the CV and QR services, we use a non-parametric alternative: the **one-sample Wilcoxon Signed Rank**

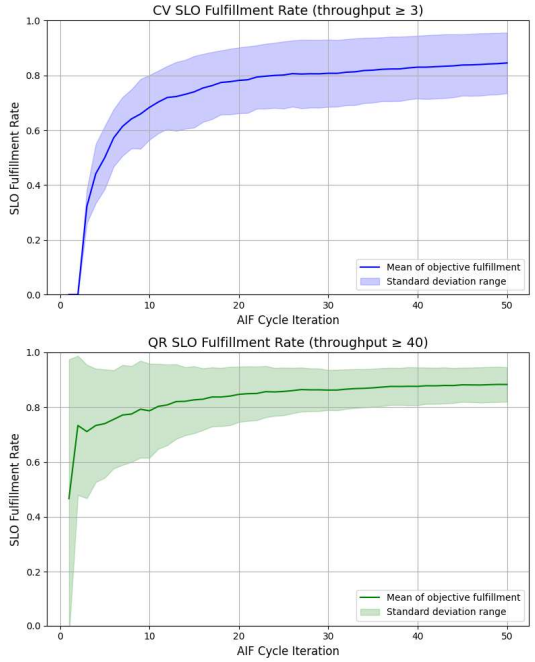


Figure 3: Evolution of cumulative objective fulfilment rate.

Table 4: Results of the Wilcoxon signed-rank test

Statistic	CV	QR
Total N	30	30
Test Statistic	440.500	295.500
Standard Error	48.491	41.366
p-value	<0.001	0.010

test. All four assumptions required to validly apply the one-sample Wilcoxon signed-rank test were assessed and found to be satisfied for the two services. Specifically: (1) the dependent variable is continuous; (2) data are independent between observations; and (3) data is distributed symmetrically around the median. The test was conducted over 30 executions. The results of Table 4 indicate that the null hypothesis can be rejected for both services as both p-values are lesser than the threshold of $\alpha = 0.05$. This suggests that both the median CV and QR fulfilment rates are significantly higher than 0.8.

8 Discussion

This Section analyses how the experiments described above provide empirical support for the theoretical claims made in Section 3, regarding the contribution of AIF to the predictability property of DTs.

- **Adaptive Perception:** Section 3 explains that an AIF-enabled DT interprets sensory inputs probabilistically, updating its internal generative model to infer latent states. This is confirmed by Experiment 1 (Section 7.3), where the DT accurately infers the underlying physical twin state and aligns its predictions with real throughput data. The close statistical match between predicted and observed values supports the presence of dynamic perception.
- **Active Simulation and Prediction:** AIF enables the DT to simulate future outcomes and assess their consequences. Experiment 1 (Section 7.3) demonstrates this through prediction of throughput over 50 iterations and 30 executions using the generative model. The alignment between predicted and real curves show the ability of the AIF-enabled DT to anticipate system behaviour under different configurations.
- **Deliberate Action:** Theoretical principles in Section 3 emphasize that AIF agents act strategically to reduce expected surprise. In the Experiment 2 (Section 7.4), the AIF-enabled DT consistently selects configurations that lead to objective fulfillment for both services (CV and QR), achieving average fulfillment rates of 0.84 and 0.88 respectively. This shows that the DT not only predicts, but also controls the behaviour of two services deliberately.
- **Continuous Learning:** AIF relies on ongoing model refinement. The results from Section 7.2 show that updating the transition model (B matrix) significantly reduces prediction error, especially when using the optimized learning rate $\alpha = 0.3$. This shows how the AIF-enabled DT refines its internal model over time to improve predictive performance.

9 Conclusion and Future Work

This work presented a novel approach to develop the predictability property of DTs through AIF, enabling autonomous prediction and control in dynamic, resource-constrained IoT environments. We proposed an AIF-enabled DT capable of modelling and managing two co-located video processing services deployed on an edge device. The DT continuously inferred latent system states, predicted future performance, and selected optimal actions based on a generative model encoded as a POMDP. A series of experiments validated the proposed approach. The tuning parameters confirmed that learning the transition model (B matrix) significantly improved prediction accuracy. Furthermore, we showed that the DT's predictions closely matched real system behaviour, as supported by statistical tests (RQ1). Finally, the DT demonstrated its ability to meet the SLOs performance with high consistency across multiple executions (RQ2). Finally, with regard to future work, firstly, we plan to compare the AIF-enabled DT with other approaches such as reinforcement learning or deep neural networks. Secondly, we aim to explore how AIF can contribute to the development of other DT properties.

Acknowledgments

This paper has been partially funded by the R+D+i project PID2022-140907OB-I00 funded by MICIU/AEI /10.13039/501100011033 and ERDF, EU and Horizon 2020 (TEADAL, 101070186) and by CNS2023-144359 financed by MICIU/AEI/10.13039/501100011033 and the European Union NextGenerationEU/PRTR. It has also been partially supported by Junta de Comunidades de Castilla-La Mancha/ERDF (SBPLY/24/180501/000020), by the University of Castilla-La Mancha (2025-GRIN-38441) and by "Catedra Ciudad de Albacete" (13585/2025). Elena Pretel holds a FPU21/02679 scholarship from Spanish Ministerio de Educación y Formación Profesional and holds a EST25/00141 scholarship from Spanish Ministerio de Ciencia, Innovación y Universidades.

References

[1] Karl Friston. 2010. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience* 11 (2 2010), 127–138. Issue 2. doi:10.1038/nrn2787

[2] Karl Friston, Lancelot Da Costa, Noor Sajid, Conor Heins, Kai Ueltzhöffer, Grigorios A. Pavliotis, and Thomas Parr. 2023. The free energy principle made simpler but not too simple. *Physics Reports* 1024 (6 2023), 1–29. doi:10.1016/j.physrep.2023.07.001

[3] Karl J Friston, Maxwell JD Ramstead, Alex B Kiefer, Alexander Tschantz, Christopher L Buckley, Mahault Albarracin, Riddhi J Pitliya, Conor Heins, Brennan Klein, Beren Millidge, Dalton AR Sakthivadivel, Toby St Clere Smithe, Magnus Koudahl, Safae Essafi Tremblay, Capm Petersen, Kaiser Fung, Jason G Fox, Steven Swanson, Dan Mapes, and Gabriel René. 2024. Designing ecosystems of intelligence from first principles. *Collective Intelligence* 3 (1 2024). Issue 1. doi:10.1177/26339137231222481

[4] Michael Grieves and John Vickers. 2017. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. 85-113 pages. doi:10.1007/978-3-319-38756-7_4

[5] Conor Heins, Beren Millidge, Daphne Demekas, Brennan Klein, Karl Friston, Iain D. Couzin, and Alexander Tschantz. 2022. pymdp: A Python library for active inference in discrete state spaces. *Journal of Open Source Software* 7 (5 2022), 4098. Issue 73. doi:10.21105/JOSS.04098/STATUS.SVG

[6] Tobias Jung, Nasser Jazdi, and Michael Weyrich. 2017. A survey on dynamic simulation of automation systems and components in the Internet of Things. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. 1–4. doi:10.1109/ETFA.2017.8247770

[7] Maninder Jeet Kaur, Ved P. Mishra, and Piyush Maheshwari. 2020. The Convergence of Digital Twin, IoT, and Machine Learning: Transforming Data into Action. 3-17 pages. doi:10.1007/978-3-030-18732-3_1

[8] Roberto Minerva, Gyu Myoung Lee, and Noel Crespi. 2020. Digital Twin in the IoT Context: A Survey on Technical Features, Scenarios, and Architectural Models. *Proceedings of IEEE* 108 (10 2020), 1785–1824. Issue 10. doi:10.1109/JPROC.2020.2998530

[9] opencv. 2024. Opencv: opencv at 4.9.0. https://github.com/opencv/opencv/tree/4.9.0

[10] Susanna Pirttikangas and Florian Michahelles. 2025. The Power Play: AI Agents, Digital Twins, and the Energy Hustle. *IEEE Pervasive Computing* 24, 1 (2025), 8–9. doi:10.1109/MPRV.2025.3555053

[11] Elena Pretel. 2025. Development of an AIF agent for the predictability DT property. https://github.com/ElenaPretelFdez/DT_elastic-workbench

[12] Elena Pretel, Alejandro Moya, Elena Navarro, Víctor López-Jaquero, and Pascual González. 2024. Analysing the synergies between Multi-agent Systems and Digital Twins: A systematic literature review. *Information and Software Technology* 174 (10 2024), 107503. doi:10.1016/j.infsof.2024.107503

[13] Elena Pretel, Elena Navarro, Víctor Casamayor Pujol, and Schahram Dustdar. 2025. Digital Twins and Artificial Collective Intelligence: Synergies for the Future. *IEEE Internet Computing* 29 (1 2025), 75–85. Issue 1. doi:10.1109/MIC.2024.3521607

[14] Prometheus. 2025. Open source metrics and monitoring for your systems and services. https://prometheus.io/

[15] Victor Casamayor Pujol, Boris Sedlak, Tommaso Salvatori, Karl Friston, and Schahram Dustdar. 2025. Distributed Intelligence in the Computing Continuum with Active Inference. arXiv:2505.24618 [cs.DC]

[16] Michael Schluse and Juergen Rossmann. 2016. From simulation to experimentable digital twins: Simulation-based development and operation of complex technical systems. In *2016 IEEE International Symposium on Systems Engineering (ISSE)*. 1–6. doi:10.1109/SysEng.2016.7753162

[17] Boris Sedlak, Victor Casamayor Pujol, Praveen Kumar Donta, and Schahram Dustdar. 2024. Active Inference on the Edge: A Design Study. In *2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. 550–555. doi:10.1109/PerComWorkshops59983.2024.10502828

[18] Fei Tao, Jiangfeng Cheng, Qinglin Qi, Meng Zhang, He Zhang, and Fangyuan Sui. 2018. Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology* 94 (2 2018), 3563–3576. Issue 9-12. doi:10.1007/s00170-017-0233-1

[19] Universidad de Castilla-La Mancha (UCLM) 2025. *Galgo for dummies: Ubuntu & SLURM*. Universidad de Castilla-La Mancha (UCLM). https://www.i3a.uclm.es/galgo/Galgo_for_dummies_ubuntu_slurm.pdf Documento técnico sobre el supercomputador GALGO y su sistema de gestión SLURM en Ubuntu 22.04.

[20] Rejin Varghese and Sambath M. 2024. YOLOv8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness. *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, 1–6. doi:10.1109/ADICS58448.2024.10533619

[21] Mohammad Yazdi, Tulen Saner, and Mahlagha Darvishmotevali. 2020. Application of an Artificial Intelligence Decision-Making Method for the Selection of Maintenance Strategy. 246-253 pages. doi:10.1007/978-3-030-35249-3_31