DAG-Based QoS-Aware Dynamic Task Placement for Networked Multi-Stage Control Pipelines

Thien Tran*, Jonathan Kua*, Thuong Hoang*, Minh Tran†, Yuemin Ding‡, and Jiong Jin§
*Deakin University, Australia; †RMIT University, Vietnam; ‡University of Navarra, Spain;
§Swinburne University of Technology, Australia
{peter.tran, jonathan.kua, thuong.hoang}@deakin.edu.au; minh.tranquang@rmit.edu.vn;
yueminding@tecnun.es; jiongjin@swin.edu.au

***Abstract*—Current Physical AI (PAI) relies heavily on closed-loop visual-servoing pipelines, whose perception and planning stages may become computationally intensive onboard due to complex models embedded on robots. In practice, offloading the perception task to on-site edges statically is inappropriate for latency-sensitive, precise industrial settings over a standardized industrial network. This emphasizes the importance of *Control–Communication–Computing (3C) co-design* in industrial automation: monolithic local execution saturates AI-accelerated machine and robot hardware, while static edge offloading exposes the control loop to network jitter. Existing adaptive task placement (ATP) controllers can partially address the gap by relocating a single pipeline stage on binary threshold rules, without a multi-stage model and an explicit cost on placement switching. In this Work-in-Progress (WiP) paper, we propose a directed acyclic graph (DAG) based quality-of-service (QoS)-aware dynamic task placement (DTP) framework for sensing–perception–planning–control pipelines in networked robotics. This pipeline is formalized as a DAG with task-level and node-level attributes for compute cost, communication delay, and feasible placement sets; over a small interpretable candidate set (fully local, static offload, hybrid), a window-based cost function combines tail end-to-end latency, deadline violation rate, hardware utilization, and a Hamming-distance switching penalty, and a DTP algorithm with hysteresis and a minimum dwell-time bounds placement chatter. Our WiP paper presents the theoretical framework, a structured qualitative analysis, and a two-phase simulation-plus-hardware-in-the-loop validation roadmap.**

***Index Terms*—Directed Acyclic Graphs (DAG), QoS-Aware Dynamic Task Placement (DTP), Networked Robotics Systems**

I. Introduction

The compute envelope of factory robotics is being reshaped by Physical AI (PAI) workloads. World Foundation Models (WFMs) for surrounding cognitive and agentic visual servoing demand inference costs that exceed the on-board envelope of typical industrial robots, while LLM-based task dispatchers exhibit cold-start latencies of sub-20 seconds, orders of magnitude beyond the sub-millisecond deadlines typical of industrial contact-rich control loops within Vision-Language-Action (VLA) models embedded on PAI [1], [2]. This mismatch demands a unified networked orchestration framework that binds heterogeneous AI capabilities to the robot–edge compute fabric within the closed-loop timing budget.

The Cloud-Fog Automation (CFA) reference architecture [3], [4] articulates a unified vision for networked industrial collaboration across the cloud–fog–edge continuum, while deterministic substrates such as Time-Sensitive Networking (TSN) and 5G-URLLC [5] provide the underlying real-time fabric. Together, they make co-located edge servers a viable extension of the PAI compute envelope, provided placement decisions respect the closed-loop timing budget. However, the layer specifies the runtime decision rule: under workload and network variability that no design-time profile can predict, the deployment question shifts from "*whether*" to offload to "*which*" pipeline stages run, where by being decided online.

A single static placement cannot serve this regime: both perception/planning compute demand and link delay are non-stationary, varying with scene complexity and model branching, and the cost of a wrong decision is asymmetric; a missed deadline propagates into actuation jitter that can violate safety envelopes. Empirical studies on industrial testbeds report deadline-violation rates above 40% under monolithic local deployment of visual-servoing workloads [6]. Adaptive placement is, therefore, a deployment necessity; the open question is what such a rule should observe what it should optimize, and how it should be stabilized for industrial deployment.

DAG-based task offloading has been studied extensively in mobile edge computing (MEC) [7], [8], but these models target "batch workflows" and optimize average latency or energy rather than deadline violation rate (V_D) under closed-loop control. Our previous Adaptive task placement (ATP) on physical robot testbeds demonstrates the value of dynamic offloading, but each system relocates only a "single" pipeline stage using binary threshold rules on individual fixed metrics, providing no formal DAG model, no multi-objective cost function, and no analytical resistance to task placement oscillation.

In this Work-in-Progress (WiP) paper, we propose a DAG-based QoS-aware DTP framework, the formal multi-objective extension of the empirically validated edge-based QoS-aware ATP controller from previous work. The system architectural framework is shown in Fig. 1. We present the complete theoretical framework, DAG model, cost function, and DTP algorithm, with the structured qualitative analysis and the validation roadmap. This work contributes the DAG pipeline model that distinguishes hard-anchored sensing and control stages from the relocatable perception and planning adaptation space, agnostic to the LAN/TSN/5G-URLLC substrate. On this model, we define the multi-objective QoS cost function and the window-based DTP algorithm with hysteresis and min-

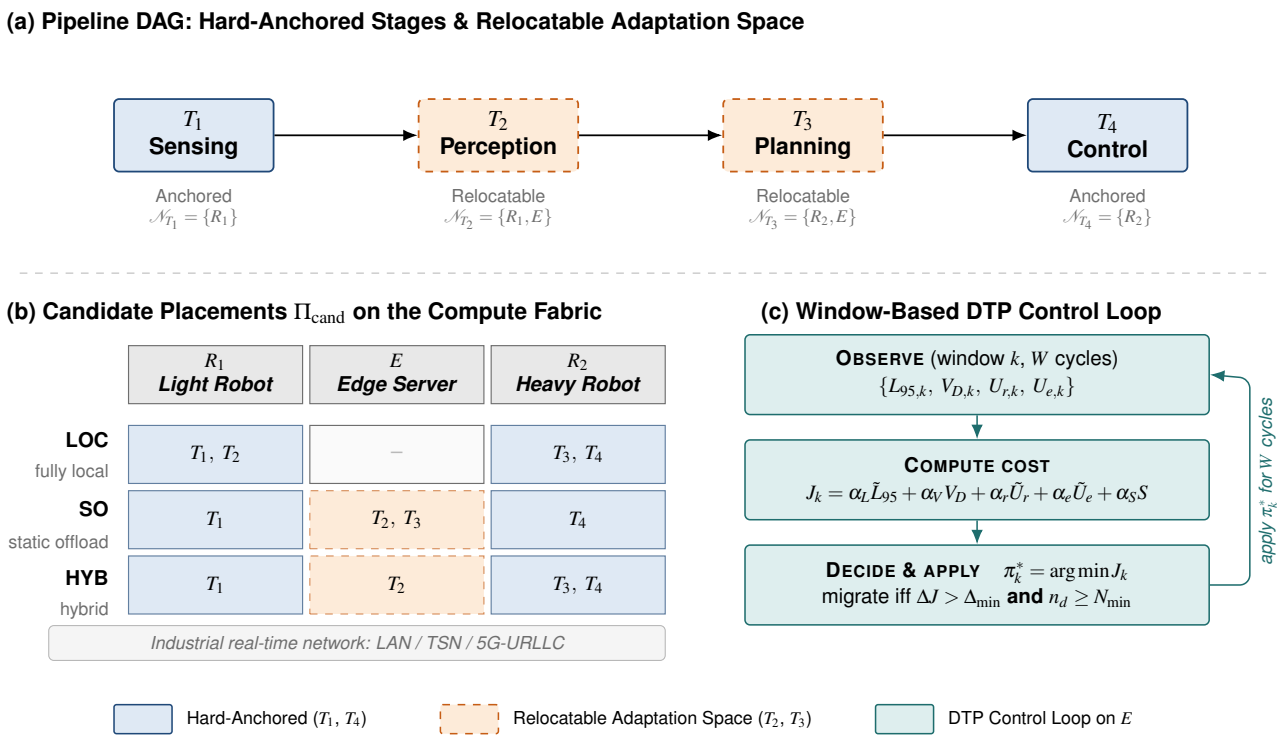

Fig. 1. DAG-QoS DTP system architecture. **(a)** The four-stage sensing–perception–planning–control pipeline as a DAG: T_1, T_4 are hard-anchored; T_2, T_3 form the relocatable adaptation space. **(b)** Three candidate placements (LOC, SO, HYB) on the $R_1/E/R_2$ compute fabric. **(c)** Window-based DTP loop on E: per-window QoS observation, cost J_k, and decision π_k^* with hysteresis $\Delta_{\min}$ and dwell-time $N_{\min}$, maintaining $V_D \leq 5\%$.

imum dwell-time that bounds placement chatter, formalizing the 3C principles within the single per-window optimization.

II. Theoretical System and DAG-based Model

A. Compute Nodes and Workload DAG

Let $\mathcal{N} = \mathcal{R} \cup \mathcal{E} \cup \mathcal{C}$ denote the global compute node set, comprising robots – $\mathcal{R}$, edge servers – $\mathcal{E}$, and (optionally or extensively) cloud nodes – $\mathcal{C}$. The target empirical factory configuration is R_1 (light robot with monocular camera), R_2 (heavy robot driving a manipulator), and E (edge server), interconnected by an industrial real-time network with configurable delay and jitter. The framework is agnostic to the link technology, engineered LAN, TSN, or 5G-URLLC [5], and is parameterized by the per-edge delay distribution $d_{ij}(\cdot)$.

A control pipeline is modeled as a directed acyclic graph – $G = (\mathcal{V}, \mathcal{E}_G)$, with task set $\mathcal{V} = \{T_1, T_2, T_3, T_4\}$ corresponding to four stages, and linear precedence $\mathcal{E}_G = \{(T_1, T_2), (T_2, T_3), (T_3, T_4)\}$. Each task $v \in \mathcal{V}$ carries:

- The feasible placement set $\mathcal{N}_v \subseteq \mathcal{N}$;
- The compute-time function $c_v : \mathcal{N}_v \to \mathbb{R}_{>0}$; and
- The CPU-utilization contribution $u_v : \mathcal{N}_v \to [0, 1]$.

Denote that each precedence edge $(v_i, v_j) \in \mathcal{E}_G$ incurs a communication delay $d_{ij}(n_i, n_j) \geq 0$ depending on link characteristics and payload size of the associated operations.

B. Placement Mapping and Architectural Invariant

A placement mapping $\pi : \mathcal{V} \to \mathcal{N}$ assigns each task to a feasible node. The visual servoing pipeline imposes:

$$\mathcal{N}_{T_1} = \{R_1\}, \qquad \mathcal{N}_{T_4} = \{R_2\}, \tag{1}$$

$$\mathcal{N}_{T_2} = \{R_1, E\}, \qquad \mathcal{N}_{T_3} = \{R_2, E\}. \tag{2}$$

This encodes the key architectural invariant of the multi-PAI System in agentic industrial automation operation:

- *Hard-anchored stages* (T_1, T_4) remain on physical AI-accelerated machines and robots, preserving sub-millisecond local operations regardless of network state.
- *Relocatable adaptation space* (T_2, T_3) migrates across the Cloud–Edge–Robotics continuum (native edge tiers) subject to the QoS cost function defined in Section III.

C. Control Loop and End-to-End (E2E) Latency

The control loop is periodic with period P and deadline $D \leq P$. Under placement π, the nominal E2E latency is:

$$L(\pi) \approx \sum_{v \in \mathcal{V}} c_v(\pi(v)) + \sum_{(v_i, v_j) \in \mathcal{E}_G} d_{ij}(\pi(v_i), \pi(v_j)). \tag{3}$$

Denote that L is stochastic in practice due to workload variability and network jitter during operations; high-percentile values therefore serve as the primary QoS-aware target.

III. QoS-Aware Cost Model

A. Per-Window QoS Metrics

The native DTP algorithm operates over observation windows of W control cycles indexed by k. Per single window, the original four QoS metrics are collected as follows:

- $L_{95,k}$: Empirical 95th-percentile E2E latency, capturing tail behavior from network outliers and bursty compute;
- $V_{D,k} \in [0,1]$: Deadline violation rate (primary SLA);
- $U_{r,k}$, $U_{e,k}$: CPU utilization across robot/edge nodes.

Latency and utilization are normalized by design targets:

$$\tilde{L}_{95,k} = \frac{L_{95,k}}{\bar{L}}, \quad \tilde{U}_{r,k} = \frac{U_{r,k}}{\bar{U}_r}, \quad \tilde{U}_{e,k} = \frac{U_{e,k}}{\bar{U}_e}. \tag{4}$$

$V_{D,k}$ is bounded in $[0,1]$ and requires no normalization.

B. Switching Penalty and Cost Function

To penalize frequent migration, a switching penalty based on the Hamming distance between placements is applied:

$$S(\pi_k, \pi_{k-1}) = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \mathbf{1}\{\pi_k(v) \neq \pi_{k-1}(v)\}. \tag{5}$$

This term suppresses "placement chatter", high-frequency oscillation that degrades network stability and is the infrastructure-level manifestation of trust latency [9]. Given non-negative weights, the QoS-aware cost for window k is:

$$J_k = \alpha_L \tilde{L}_{95,k} + \alpha_V V_{D,k} + \alpha_r \tilde{U}_{r,k} + \alpha_e \tilde{U}_{e,k} + \alpha_S S(\pi_k, \pi_{k-1}). \tag{6}$$

The tension between $\alpha_V V_{D,k}$ (deadline aggression) and $\alpha_S S(\pi_k, \pi_{k-1})$ (migration resistance) formalizes the core control–infrastructure trade-off. In industrial settings, $\alpha_V > \alpha_L > \alpha_S$ reflects prioritizing SLA compliance as the primary, raw latency as the secondary, and stability as a soft constraint.

C. Constrained Placement Problem

To keep per-window evaluation tractable on edge hardware, we originally restrict to a small interpretable candidate set Π_{cand} comprising the following three pilot placements:

- **LOC** (fully local): T_2 on R_1, T_3 on R_2;
- **SO** (static offload): T_2 and T_3 on E;
- **HYB** (hybrid): T_2 on E, T_3 on R_2.

The window-level placement problem is

$$\pi_k^* \in \arg\min_{\pi \in \Pi_{\text{cand}}} J_k(\pi; \pi_{k-1}), \tag{7}$$

subject to $L_{95,k}(\pi) \leq L_{\max}$ and per-node utilisation caps $U_{n,k}(\pi) \leq U_{\max}$ for all $n \in \mathcal{N}$.

IV. Dynamic Task Placement Algorithm

A. Window-Based DTP Procedure

Algorithm 1 describes the DTP procedure running on the edge node – E. Two parameters jointly enforce hysteresis:

- $N_{\min}$: Minimum dwell windows; prevents rapid flapping when metrics fluctuate near thresholds;
- $\Delta_{\min}$: Cost-improvement threshold; ensures only statistically significant improvements trigger migration.

Denote that per-window complexity is $O(|\Pi_{\text{cand}}|)$, which is always lightweight on any edge node if task placements occur.

Algorithm 1 Window-Based Dynamic Task Placement (DTP)

Require: W; $(\alpha_L, \alpha_V, \alpha_r, \alpha_e, \alpha_S)$; $\Delta_{\min}$; $N_{\min}$; Π_{cand}

Init $\pi_0 \in \Pi_{\text{cand}}$; dwell $n_d \leftarrow 0$
for $k = 1, 2, \ldots$ **do**
  Apply π_{k-1} for W cycles; collect $\{L_{95,k}, V_{D,k}, U_{r,k}, U_{e,k}\}$
  Compute $J_k(\pi_{k-1}; \pi_{k-2})$ via (6)
  if $n_d < N_{\min}$ **then**
    $n_d \leftarrow n_d + 1$; $\pi_k \leftarrow \pi_{k-1}$; **continue** {enforce min. dwell}
  end if
  Estimate $J_k(\pi; \pi_{k-1})$ for all $\pi \in \Pi_{\text{cand}}$
  $\pi_k^* \leftarrow \arg\min_\pi J_k(\pi; \pi_{k-1})$
  $\Delta J \leftarrow J_k(\pi_{k-1}; \pi_{k-2}) - J_k(\pi_k^*; \pi_{k-1})$
  if $\Delta J > \Delta_{\min}$ **then**
    $\pi_k \leftarrow \pi_k^*$; $n_d \leftarrow 0$ {migrate}
  else
    $\pi_k \leftarrow \pi_{k-1}$; $n_d \leftarrow n_d + 1$ {hold}
  end if
end for

B. Estimating QoS for Non-Active Placements

Scoring placements not currently active requires estimating their idle QoS metrics. The three initial complementary mechanisms are supported along the operation loop as follows:

- **Static profiling:** Compute times $c_v(n)$ and communication delays measured offline and stored as lookup tables.
- **Online emulation:** A small fraction of cycles is routed through each alternative in the background to update estimates continuously during the non-placement stage.
- **Conservative scaling:** Upper-bound estimates derived from current measurements or passive monitored feedback and known compute/network ratios.

The original algorithm is agnostic to the choice, provided cost comparisons remain qualitatively consistent for DTP.

V. Qualitative Analysis and Deployment Context

A. Anticipated Placement Trajectories

Table I summarizes the expected steady-state DTP behaviors relative to LOC and SO baselines across four pilot canonical stress scenarios. The policy converges toward LOC under adverse network conditions and toward SO under robot CPU stress; HYB dominates when compute and communication resources are jointly constrained, by offloading the heavier T_2 (Perception) to E while keeping T_3 (Planning) on R_2.

B. Co-Design Implications and the PAI Nervous System

Cast as the CFA 3C co-design principle-based orchestration layer, the DAG-based QoS-aware DTP framework provides three concrete guarantees against the failure modes intrinsic to networked factory control following industrial standards:

- The hard-anchoring of T_1 and T_4 preserves the sub-millisecond reflex loop on local hardware, decoupling actuation safety and dependencies from network state;

- The relocatable adaptation space accommodates the heterogeneous compute demands of T_2 and T_3, including emerging VLA/LLM workloads, without overloading machine hardware or hard-coding offload decisions;
- The Hamming-distance switching penalty in (5) provides a mathematically bounded guarantee against placement chatter, translating directly to operator-observable control-loop stability and to the deterministic timing budgets expected of TSN-class substrates [5].

Technically, a factory automation system that cannot guarantee $V_D \leq 5\%$ across the deployment envelope cannot credibly claim Industry 4.0 production maturity [9] toward industrial agentic automation for Industry 5.0 readiness.

VI. Empirical Strategy and Validation Roadmap

Validation proceeds in two main phases, each with concrete parameters drawn directly from the physical robotic testbed to ensure direct comparability and reflection on the previous empirical LOC, SO, and ATP baselines.

A. Phase 1 – Discrete-Event Simulation

A discrete-event simulator parameterized will validate cost-function sensitivity and hysteresis bounds against the predictions in Table I. Key parameters are:

- Control period $P \in [20, 50]$ ms with deadline $D \leq P$;
- Baseline link RTT of 1–2 ms representative of an engineered industrial LAN, with TSN-class jitter envelopes as a sensitivity axis following standardization [5];
- CPU stress profiles applied to the R_1-equivalent node;
- Gaussian network fault injection (e.g., $\mu = 25$ ms, $\sigma = 5$ ms, $p_{\text{loss}} = 2\%$), plus additional extended verification.

B. Phase 2 – Hardware-in-the-Loop (HIL)

The modular cloud-edge-robotics testbed will provide:

- Measured compute profiles $c_v(n)$ on each physical node;
- Empirical network delay distributions on the R_1–E and E–R_2 links with extended IIoT devices/PLCs;
- Empirical CDFs of L_{95}, V_D, U_r, and U_e for LOC, SO, ATP, and DAG-QoS DTP across all stress scenarios.

Phase 2 enables direct quantitative comparison between the proposed multi-stage DTP and the single-stage ATP baseline.

VII. Conclusions and Future Work

In this WiP paper, we presented the DAG-based QoS-aware DTP, the formal multi-objective extension of our edge-based QoS-aware ATP framework. By formalizing the visual-servoing pipeline as the DAG, introducing the window-based cost function with the Hamming-distance switching penalty, and generalizing the ATP algorithm to concurrent multi-task placement, the extended framework addresses three structural limitations of single-task adaptive placement: limited scope, threshold-based decisions, and the absence of a formal switching cost model. The result is a runtime determinism layer compatible with the CFA-based 3C co-design ethos central to factory communications and mature industrial automation.

TABLE I
Expected dominant DTP placement per stress scenario.

Scenario	**LOC**	**SO**	**DTP (expected)**
Baseline	Low V_D; high U_r	Low U_r; moderate V_D	Mostly LOC; occasional HYB
Robot CPU stress	High V_D (U_r sat.)	Mitigated; higher L_{95}	Converges to SO
Edge CPU stress	Stable; U_r elevated	High V_D (U_e sat.)	LOC or HYB
Network impairment	Low variance	High V_D via jitter	Converges to LOC

Beyond factory robotic automation, this middle layer will act as one of the key cores in termed the "*PAIs Nervous System*": the runtime infrastructure that lets LLM agents and VLA models operate reliably under real compute and network variability on different PAIs and nearby edges/clouds.

Future work will extend the proposed framework in this WiP towards full deployment through: (i) empirical validation on the physical testbed; (ii) expansion of the candidate placement set beyond LOC/SO/HYB and extension to multi-robot fleets where a single edge serves several DTP loops concurrently; (iii) online cost-weight adaptation of hyperparameters via Bayesian optimization or lightweight meta-learning; (iv) joint compute–network co-scheduling with deterministic substrates, elevating the network from a delay distribution to a co-optimized hardware resource, and considering new QoS dimensions such as Age-of-Information (AoI).

References

[1] R. Li, Y. Zhou, Y. Zhu, K. Chen, J. Wang, S. Wang, K. Hu, M. Yu, B. Jiang, Z. Su, J. Ma, X. He, Y. Shen, Y. Yang, G. Ren, M. Yao, W. Wang, and Y. Mu, "RoboClaw: An agentic framework for scalable long-horizon robotic tasks," 2026.

[2] H. Li, X. Cao, Z. Zeng, Y. Wu, Y. Zhang, and Y. Xia, "OpenGo: An OpenClaw-based robotic dog with real-time skill switching," 2026.

[3] J. Jin, K. Yu, J. Kua, N. Zhang, Z. Pang, and Q.-L. Han, "Cloud-fog automation: Vision, enabling technologies, and future research Directions," *IEEE Trans. Ind. Inform.*, pp. 1–16, 2023.

[4] J. Jin, Z. Pang, J. Kua, Q. Zhu, K. H. Johansson, N. Marchenko, and D. Cavalcanti, "Cloud-fog automation: The new paradigm towards autonomous industrial cyber-physical systems," *IEEE J. Sel. Areas Commun.*, 2025.

[5] T. Zhang, G. Wang, C. Xue, J. Wang, M. Nixon, and S. Han, "Time-Sensitive Networking (TSN) for industrial automation: Current advances and Future directions," *ACM Comput. Surv.*, vol. 57, no. 2, Oct. 2024.

[6] T. Luu, Q. Nguyen, T. Tran, M. Tran, S. Ding, J. Kua, and T. Hoang, "Enhancing real-time robot teleoperation with immersive virtual reality in industrial IoT networks," *Int. J. Adv. Manuf. Technol.*, vol. 139, no. 11, pp. 6233–6257, Aug. 2025.

[7] Z. Tong, J. Deng, J. Mei, Y. Zhang, and K. Li, "Multi-objective DAG task offloading in MEC environment based on federated DQN with automated hyperparameter optimization," *IEEE Trans. Serv. Comput.*, vol. 17, no. 6, pp. 3999–4012, 2024.

[8] J. Liu, J. Ren, Y. Zhang, X. Peng, Y. Zhang, and Y. Yang, "Efficient dependent task offloading for multiple applications in MEC-cloud system," *IEEE Trans. Mobile Comput.*, vol. 22, no. 4, pp. 2147–2162, 2023.

[9] R. Andreoli, R. Mini, P. Skarin, H. Gustafsson, J. Harmatos, L. Abeni, and T. Cucinotta, "A multi-domain survey on time-criticality in cloud computing," *IEEE Trans. Serv. Comput.*, vol. 18, no. 2, pp. 1152–1170, 2025.