Unified Systems Approach for Human centered Operations (USAHO)
Vol. 1, No. 1, June 2026, Page 69-82
***e*-ISSN: XXXX-XXXX**

Agentic LLM Architecture For Real-Time Facility Layout Optimization With An Integrated LLMOps Framework

Jhaka Tegar[1]*, & Agung Aditia Nugraha[2]
[1]*Universitas Riau, Indonesia, [2]Universitas Muhammadiyah Sumatera Barat, Indonesia
*Co e-mail: jhakategar@gmail.com[1]

Article Information
Received: September 14, 2026
Revised: September 29, 2026
Online: October 22, 2026

Keywords
Agentic LLM, Digital twin, Facility layout planning, LLMOps, Real-time optimization

ABSTRACT
Facility layout planning is a combinatorial industrial engineering problem involving material handling cost, travel distance, space utilization, safety, and production flow. Conventional optimization methods often require manual formulation, algorithm selection, parameter tuning, and repeated interpretation of operational changes. This study proposes an agentic large language model (LLM) architecture integrated with an organized LLM Operations (LLMOps) framework for facility layout optimization. The architecture assigns planning, optimization, verification, simulation, and critique to specialized agents, while relying on external optimization tools for numerical computations. A synthetic benchmark comprising 40 layout scenarios with demand and flow perturbations was used to compare manual systematic layout planning (SLP), genetic algorithm (GA), single tool-augmented LLM, and the proposed architecture. The proposed method achieved a mean material-handling-cost index of 88.39, outperforming manual SLP (128.54), GA (110.54), and single-LLM (103.83). This represents reductions of 31.2% and 20.0% compared with manual SLP and GA, respectively. Mean logistics distance decreased by 26.8% relative to the manual baseline. Furthermore, 95% of scenarios produced feasible layouts without manual repair. Statistical analysis confirmed a significant improvement over manual SLP ($t=46.86$, $p<0.001$). These findings demonstrate that agentic LLMs can coordinate optimization tools, constraint verification, simulation feedback, and reproducible operations within an auditable workflow.

INTRODUCTION

Facility layout planning (FLP) determines the spatial arrangement of machines, workstations, storage areas, inspection points, transport corridors, and supporting functions within a production system. Because spatial decisions influence travel distance, material handling, congestion, safety, work-in-process, and throughput, FLP is a strategic industrial engineering

DOI: http://doi.org/10.69855/usaho
Available online at website: https://gpijournal.com/index.php/usaho

This work is licensed under a Creative Commons Attribution 4.0 International license

decision rather than a purely geometric exercise. Recent studies have shown that digital-twin and simulation-based approaches can improve layout evaluation by allowing alternative configurations to be tested before physical implementation (Guo et al., 2021; Choi et al., 2024).

At the same time, manufacturing systems are becoming more dynamic. Product mix changes, demand variability, equipment relocation, automated guided vehicle (AGV) routes, and temporary process constraints can make a layout that was previously efficient less suitable for the current operating state. Research on reinforcement learning and adaptive facility layout has therefore emphasized the need to connect layout decisions with production flow and changing operating conditions (Kroemer et al., 2023; Choi et al., 2024). Recent work combining queueing networks with deep reinforcement learning for intelligent manufacturing units (Li et al., 2025), and a 2026 review of machine learning, digital twin, and AR/VR in facility layout planning (Jazi & Ameri, 2026), further illustrate the movement toward adaptive, data-driven layouts.

Large language models provide a different capability. Rather than serving only as numerical optimizers, LLMs can interpret natural-language requirements, translate heterogeneous instructions into structured constraints, select tools, explain alternatives, and coordinate multi-step workflows. Retrieval-augmented generation can connect a model to external knowledge rather than relying exclusively on parameters (Lewis et al., 2020), while ReAct demonstrated the value of interleaving reasoning and tool-oriented actions (Yao et al., 2023). Toolformer similarly showed that language models can learn when and how to call external APIs, highlighting the importance of tool use for tasks that exceed the model's native capabilities (Schick et al., 2023).

The emergence of agentic LLMs extends this paradigm from a single prompt-response interaction to systems that reason, act, and interact over a task horizon. Recent surveys describe agentic systems in terms of reasoning, action, interaction, tool use, collaboration, memory, and governance (Wang et al., 2024; Plaat et al., 2025; Du et al., 2026; Bajoria et al., 2026). For industrial engineering, this distinction is important because layout optimization naturally consists of several linked tasks: extracting facility requirements, constructing a formal representation, selecting or invoking an optimization algorithm, checking constraints, simulating the candidate layout, comparing alternatives, and communicating the result to a human engineer.

However, an LLM-based workflow can also create new risks. A language model may hallucinate a dimension, overlook a hard constraint, invoke an unsuitable optimization tool, or produce a plausible explanation for an infeasible layout. Therefore, an industrial architecture should not allow an LLM to act as an unrestricted numerical decision-maker. Instead, the LLM should coordinate deterministic or validated tools, while an explicit verification layer prevents infeasible solutions from reaching the decision stage. This principle is consistent with recent work connecting LLMs, external function calling, and specialized agents for facility layout planning (Ma et al., 2026; Guo et al., 2026).

The research gap addressed in this study is therefore architectural rather than algorithmic. Existing FLP research has demonstrated the value of metaheuristics, reinforcement learning, digital twins, and simulation, while recent LLM research has demonstrated reasoning and tool-use

This work is licensed under a Creative Commons Attribution 4.0 International license

capabilities. What is less developed is an organized industrial workflow that uses an LLM agent system to coordinate those established methods in real time while preserving reproducibility, model/version tracking, validation, and rollback. The proposed architecture addresses this gap through an LLMOps control plane and specialized agents.

The term *real-time* requires an explicit operational meaning. In manufacturing, real-time decision support depends on a complete data chain: shop-floor events are captured by Internet of Things (IoT) devices, passed through a message broker or stream-processing layer, stored in a real-time database, and consumed by a manufacturing execution system (MES) and a simulation model (Santos et al., 2019; Feldt et al., 2020). Distributed stream-processing frameworks are designed to handle unbounded event streams with low latency and fault tolerance (Isah et al., 2019), whereas studies on MES–digital twin integration show that the achievable refresh rate is often limited by data preparation and synchronization rather than by computation alone (Lanzini et al., 2024). Accordingly, this study uses real-time in the soft or near-real-time sense: an accepted layout recommendation is produced within seconds after a change in the material-flow matrix. It does not refer to hard real-time control with guaranteed deadlines, and only the optimization step of the data chain is measured experimentally.

Building on this gap, the study addresses the following research questions (RQ):

RQ1. To what extent does an agentic LLM architecture, in which numerical search is delegated to external optimization tools, reduce material handling cost (MHC) and logistics distance relative to a manual systematic-layout-planning (SLP) heuristic, a genetic algorithm (GA), and a single tool-augmented LLM when all methods share the same objective function and constraints across 40 perturbed layout scenarios?

RQ2. To what extent does the explicit verification and critique loop increase the share of scenarios for which a feasible layout (no hard-constraint violation) is produced without manual repair?

RQ3. What response latency does the additional planning, verification, simulation, and critique impose compared with a single-LLM workflow and a GA, and is this latency compatible with near-real-time layout re-planning?

RQ4. Which LLMOps controls (model, prompt, tool, data, and objective-weight versioning, monitoring, and rollback) are required so that each recommended layout is traceable, reproducible, and reversible?

The objectives of the study, which correspond to RQ1–RQ4, are: (1) to design an agentic LLM architecture for dynamic facility-layout optimization; (2) to define an organized LLMOps workflow covering data, prompts, models, tools, experiments, monitoring, and rollback; (3) to evaluate the architecture using a reproducible 40-scenario benchmark; and (4) to compare cost, logistics distance, feasibility, and computational response against conventional and single-LLM baselines.

The central hypothesis is that a tool-augmented multi-agent architecture will outperform a manual layout heuristic, a conventional genetic algorithm, and a single LLM workflow on layout quality and constraint compliance because task specialization reduces the burden on any one model

This work is licensed under a Creative Commons Attribution 4.0 International license

and creates explicit opportunities for verification and repair. The LLM configuration used by the single-LLM and agentic variants is specified in the Methods section.

Classical FLP formulations typically minimize material handling cost by combining flow between departments with inter-departmental distance. A simplified objective can be expressed as MHC = Σ_i Σ_j f_ij d_ij c_ij, where f_ij denotes material flow, d_ij denotes distance, and c_ij denotes the unit handling cost. Real factories add hard constraints such as non-overlap, aisle width, safety separation, fixed-location departments, orientation, access, and emergency routes. Consequently, a solution with a low numerical objective may still be operationally unacceptable. Recent digital-twin work emphasizes the value of evaluating layouts in a virtual representation before physical implementation (Guo et al., 2021).

METHODS

This study uses a design-and-evaluation approach. First, an agentic architecture was specified from the information requirements of FLP. Second, a synthetic benchmark was constructed to emulate a discrete-manufacturing workshop with repeated changes in material flow and operating constraints. Third, four solution approaches were evaluated under identical scenario definitions. The benchmark is intentionally synthetic so that the workflow can be reproduced without exposing confidential industrial information. It is not presented as data collected from an actual factory.

Each scenario contains eight functional areas: raw material receiving (RM), CNC processing (CNC), assembly (ASM), quality assurance (QA), finished-goods staging (FG), work-in-process storage (WIP), warehouse (WH), and packing/shipping (PK). The virtual workshop is represented as a 6×4 grid. Each department occupies one grid cell for the benchmark. Material-flow matrices are perturbed across scenarios to represent changes in product mix and demand. Distance is calculated using rectilinear distance, while feasibility rules require non-overlap, aisle continuity, and preservation of designated access relationships.

For each scenario s, the system minimizes a weighted objective: J_s = w1 MHC_s + w2 D_s + w3 V_s + w4 R_s, where MHC is material handling cost, D is total logistics distance, V is the number of hard-constraint violations, and R is a normalized risk penalty. Hard violations are assigned a sufficiently large penalty and are also checked separately by the Verifier Agent. The weights used for the benchmark were w1 = 0.50, w2 = 0.25, w3 = 0.20, and w4 = 0.05. The benchmark therefore prioritizes material handling cost while preventing the system from treating safety or feasibility as optional.

All four approaches receive the same scenario definition (department list, 6×4 grid, flow matrix, unit-cost matrix, and access relationships), are scored with the same objective function J_s defined above, and are checked by the same deterministic feasibility checker that implements the hard constraints (non-overlap, aisle continuity, and designated access relationships). Hard-constraint violations enter J_s through V_s for every method, so no method is favored by a different

This work is licensed under a Creative Commons Attribution 4.0 International license

notion of cost or feasibility. What differs between methods is how a candidate layout is generated and when the search stops.

The four approaches are implemented as follows. In the manual SLP baseline, layouts were produced by the study author who derived a relationship chart from the scenario flow matrix, ranked department pairs by flow intensity, and placed high-flow pairs in adjacent cells; no automated search is involved. The GA encodes a layout as an assignment of the eight departments to distinct cells of the 6×4 grid, uses J_s as the fitness function, and applies tournament selection, order crossover, and swap mutation with population size 50; it stops at the generation limit or after the stagnation window reported in Table 1. The single-LLM baseline receives the scenario in one prompt and has access to the same deterministic evaluation tool as the other methods, but the optimizer tool is not available; it returns its layout after a single reasoning-and-acting pass without a mandatory repair loop.

In the proposed agentic architecture, the Planner Agent converts the scenario into a structured problem, the Optimizer Agent invokes the GA tool, the Verifier Agent checks the candidate against the hard constraints, the Simulation Agent evaluates it, and the Critic Agent compares it with alternatives and decides whether to request a repair. A candidate is accepted only when the Verifier reports no hard-constraint violation. If the repair limit R is reached without a passing candidate, the best candidate is returned and counted as infeasible. To keep the comparison fair, the GA component of the agentic architecture uses a reduced search budget of 30 individuals and 50 generations, compared with 50 individuals and 100 generations for the stand-alone GA, and the evaluation budget per scenario is reported in Table 2.

Both LLM-based variants use the same general-purpose, instruction-tuned, decoder-only transformer LLM with native function-calling (tool-use) support, accessed through a hosted application programming interface (API): GPT-4o-mini (exact API model snapshot not recorded) from OpenAI. A general-purpose model was chosen deliberately instead of a domain-fine-tuned model so that performance differences can be attributed to the workflow architecture rather than to additional training. No fine-tuning or retrieval corpus is used; domain knowledge is injected through a versioned system prompt that contains the problem schema, the objective function, and the tool descriptions. Inference parameters were fixed for all runs (temperature, maximum output tokens, context window, and random seed were not recorded in the supplied manuscript). In the agentic architecture the same model is instantiated in the Planner, Optimizer, and Critic roles through role-specific prompts, whereas feasibility verification and layout simulation are deterministic code that the LLM can invoke but cannot override. Model-specific effects, such as context-window limits, token cost, and tool-calling reliability, were not isolated and are discussed as a limitation.

Response latency is the wall-clock time from the arrival of a scenario (a changed flow matrix) to the return of a layout recommendation, including all LLM calls, tool calls, verification, simulation, and critique steps up to acceptance.

This work is licensed under a Creative Commons Attribution 4.0 International license

RESULTS

1. Overall Benchmark Performance

Overall benchmark performance refers to the overall evaluation of the four facility layout optimization approaches across the benchmark scenarios. It compares the methods based on material handling cost (MHC), logistics distance, feasibility rate, and the percentage reduction in MHC relative to the manual SLP baseline. These indicators are used to assess not only the efficiency of each optimization approach but also its ability to generate feasible layout solutions. A lower MHC and logistics distance indicate better operational efficiency, while a higher feasibility rate indicates stronger compliance with predefined layout constraints. Therefore, the overall benchmark performance provides a comprehensive basis for determining the relative effectiveness of the proposed Agentic LLM architecture compared with conventional and single-LLM approaches.

Table 3. Overall benchmark performance

Method	Mean MHC ± SD	Mean distance	Feasibility	MHC reduction vs manual
Manual SLP	128.54 ± 11.42	415.9	5%	—
Genetic Algorithm	110.54 ± 9.62	363.9	22%	14.0%
Single LLM	103.83 ± 10.00	350.8	57%	19.2%
Proposed Agentic LLM	88.39 ± 8.93	304.5	95%	31.2%

As shown in Table 3, the proposed architecture produced the lowest mean MHC (88.39) and the lowest mean logistics distance (304.5). Compared with manual SLP, the reduction in MHC was 31.2%, while the reduction in distance was 26.8%. Compared with the genetic algorithm, the proposed workflow reduced MHC by 20.0%, and compared with the single LLM by 14.9%. The proposed workflow also produced a feasible layout in 95% of scenarios, against 57% for the single LLM, 22% for the GA, and 5% for manual SLP. These values should be interpreted as benchmark evidence of workflow behavior, not as a claim that the architecture will produce the same percentage improvements in every factory.

2. Scenario-Level Consistency

Scenario-level consistency refers to the ability of each optimization approach to maintain stable performance across different facility-layout scenarios with varying material-flow patterns. This analysis compares the material handling cost (MHC) obtained by each method in individual scenarios and examines whether the proposed Agentic LLM architecture can consistently generate feasible and efficient solutions rather than performing well only under a specific flow configuration. The *Verifier outcome* further indicates whether the resulting layout satisfies the predefined constraints. The benchmark contains 40 scenarios, but, for reasons of space, Table 4 reports Scenarios

This work is licensed under a Creative Commons Attribution 4.0 International license

1–10 as an excerpt; the complete 40-scenario results on which Table 3 is based are provided as a supplementary file to be provided with the full 40-scenario results.

Table 4. Scenario-level consistency (Scenarios 1–10 of 40)

Scenario	Manual	GA	Single LLM	Agentic LLM	Verifier outcome
1	132.3	117.2	110.9	87.2	Feasible
2	113.4	99.7	90.0	78.9	Feasible
3	138.5	115.9	111.6	96.5	Feasible
4	141.2	122.6	113.6	103.2	Feasible
5	100.7	87.0	77.7	72.0	Feasible
6	109.8	95.2	92.1	76.9	Feasible
7	129.8	115.6	104.0	95.3	Feasible
8	123.6	107.2	101.4	80.9	Feasible
9	127.8	112.9	107.2	85.7	Feasible
10	116.1	100.1	97.2	76.9	Feasible
Mean	123.3	107.3	100.6	85.3	10/10 feasible

Within the ten scenarios shown in Table 4, the agentic workflow obtained the lowest MHC in every scenario. Its reduction relative to the GA ranged from 15.8% to 25.6%, and relative to the single LLM from 7.3% to 21.4%, so the magnitude of improvement varies with the flow pattern. The mean MHC of these ten scenarios (manual 123.32, GA 107.34, single LLM 100.57, agentic 85.35) is an excerpt statistic and should not be confused with the 40-scenario means in Table 3. All ten displayed layouts passed the Verifier; because the overall feasibility rate is 95% (38 of 40 scenarios), the two scenarios without a verified feasible layout lie among Scenarios 11–40. The Verifier Agent also creates an explicit boundary between candidate generation and candidate acceptance.

3. Logistics Distance

Logistics distance refers to the total travel distance required to move materials, work-in-process, and finished products between functional areas within the facility. This metric is an important indicator of facility layout efficiency because shorter travel distances generally reduce material handling effort, transportation time, congestion, and associated operational costs. In this study, logistics distance is used to evaluate how effectively each optimization method arranges departments to minimize unnecessary material movement.

Table 5. Logistics distance

Method	Mean distance	Reduction vs manual
Manual SLP	415.9	Reference
Genetic Algorithm	363.9	12.5%
Single LLM	350.8	15.7%
Proposed Agentic LLM	304.5	26.8%

This work is licensed under a Creative Commons Attribution 4.0 International license

The results show a clear improvement in logistics distance across the evaluated methods. The Manual SLP approach produced the highest mean distance of 415.9, which was used as the reference value. The Genetic Algorithm reduced the mean distance to 363.9, representing a 12.5% reduction compared with the manual approach. The Single LLM achieved a further reduction to 350.8, equivalent to 15.7%. The Proposed Agentic LLM obtained the lowest mean logistics distance of 304.5, corresponding to a 26.8% reduction relative to Manual SLP. These findings indicate that the proposed Agentic LLM architecture provides the most effective arrangement for reducing material travel distance among the evaluated approaches.

4. Response Latency

Response latency refers to the wall-clock time between the arrival of a changed material-flow matrix and the return of a layout recommendation. It is the main operational indicator of whether an architecture can support near-real-time re-planning, because a recommendation that arrives after the production state has changed no longer supports the decision it was meant to inform. For the agentic architecture, latency includes every planner, optimizer, verifier, simulation, and critic step up to acceptance. Latency is reported as the mean per scenario under the measurement conditions listed in Table 2.

Table 6. Response latency

Method	Mean latency	Observation
Genetic Algorithm	8.77 s	Numerical search only
Single LLM	5.93 s	One model call + tools
Proposed Agentic LLM	6.41 s	Multiple agents + verification

As shown in Table 6, the GA required 8.77 s, the single LLM 5.93 s, and the proposed agentic architecture 6.41 s. The agentic architecture was therefore 0.48 s (8.1%) slower than the single-LLM baseline because it performs explicit verification, simulation, and critique, but it was 2.36 s (26.9%) faster than the stand-alone GA. These figures are comparable only under the conditions in Table 2: the same hardware for local tools, the same network path and LLM provider, 3 repetitions per scenario, warm-up runs discarded, and on average not recorded LLM calls and not recorded tool calls per scenario for the agentic workflow against not recorded and not recorded for the single LLM. Because API and network latency fluctuate between runs, a difference of 0.48 s should be read as an indication of the overhead of the verification layer rather than as a fixed constant not recorded in the supplied benchmark results. This result is important: real-time optimization does not necessarily mean minimizing model latency alone. In an industrial setting, the relevant measure is decision latency—the time required to produce an accepted and operationally feasible layout. A faster but infeasible proposal can create greater downstream cost than a slower verified solution. Notably, the agentic architecture invokes the GA as a tool yet is faster than the stand-alone GA, which implies that the GA tool is run with a reduced search budget inside the workflow (30 individuals and 50 generations, versus 50 individuals and 100 generations for the stand-alone GA).

This work is licensed under a Creative Commons Attribution 4.0 International license

DISCUSSION

The principal finding is that the agentic architecture improves layout quality by coordinating specialized tasks rather than by asking an LLM to replace mathematical optimization. This distinction is consistent with the broader literature on tool-using language models: language models are useful for selecting, invoking, and interpreting tools, while deterministic tools remain preferable for exact numerical operations (Schick et al., 2023).

The numerical advantage can be decomposed relative to the manual SLP baseline (mean MHC 128.54). The GA lowered MHC by 18.00 points (14.0%), the single LLM by 24.71 points (19.2%), and the agentic architecture by 40.15 points (31.2%). The agentic architecture therefore improved on the GA by a further 22.15 points (20.0%) and on the single LLM by 15.44 points (14.9%). Because the benchmark was not designed as an ablation, the mechanisms below are interpretations that are consistent with the design and the data, not demonstrated causes.

First, the agentic workflow converts infeasible or poorly scoring candidates into repaired ones. Its feasibility rate (95%) is far higher than that of the single LLM (57%), the GA (22%), and manual SLP (5%). Because J_s places a large weight on hard-constraint violations ($w3 = 0.20$, together with a large penalty), a method that stops at a layout with violations remains at a worse point of the weighted objective, whereas the Verifier–Optimizer repair loop keeps searching until the candidate is admissible. The GA, in contrast, has a single stagnation-based stop rule and no mechanism that reacts to a particular violated constraint. Second, the Planner Agent re-expresses each changed flow matrix as a structured problem before the optimizer is called, so the search starts from an up-to-date problem definition instead of a generic one; this is the most plausible reason why the agentic workflow outperforms the single LLM, which has to interpret the scenario, choose tools, and check constraints within one pass and has no re-planning step when its check fails.

Third, the Critic and Simulation Agents compare several candidates before acceptance, which acts as a selection step: choosing the best of several evaluated candidates lowers the expected MHC even if each candidate is generated by the same optimizer. The scenario excerpt supports a shift of the whole distribution rather than a reduction in extreme cases: agentic MHC follows the difficulty of the scenario (Pearson $r = .93$ with manual MHC across Scenarios 1–10), the relative dispersion is similar across methods (coefficient of variation 8.7–10.1% from Table 3), and the improvement over the GA stays within 16–26%. The high paired t statistic ($t = 46.86$) mainly reflects how consistent these paired differences are in a synthetic benchmark; it is not evidence of external generalization.

Fourth, and most importantly for interpreting the numbers, part of the advantage may come from a larger total evaluation effort rather than from the architecture itself. The agentic workflow performs more evaluations (verification, simulation, critique, and repeated optimizer calls) than the single-LLM baseline and the one-shot GA. The latency results partly mitigate this concern in wall-clock terms (6.41 s versus 8.77 s for the GA), but they do not equalize the number of objective evaluations. An equal-budget GA with restarts and an ablation that removes the Verifier,

This work is licensed under a Creative Commons Attribution 4.0 International license

Simulation, or Critic Agent one at a time are therefore the necessary controls to attribute the 20.0% gain over the GA to the architecture, and they are the first priority for follow-up experiments.

The reduction in material handling cost observed in the benchmark is also consistent with the direction of recent facility-layout research. Digital-twin and reinforcement-learning studies demonstrate that production-flow information can improve layout decisions beyond static geometric criteria (Guo et al., 2021; Choi et al., 2024). The proposed architecture adds a coordination layer: the Planner Agent translates changing operational requirements into a structured problem, the Optimizer Agent executes the approved numerical search, and the Simulation and Verifier Agents determine whether a candidate is actually acceptable.

An important contribution of the architecture is constraint governance. In ordinary conversational LLM use, a model can provide a confident response even when its internal representation is incomplete. In the proposed system, confidence is not treated as a validation signal. A layout becomes eligible for recommendation only after passing deterministic feasibility checks. This design directly addresses a known weakness of LLM applications: fluent generation does not imply factual or numerical correctness.

The LLMOps layer is equally important. Without versioning, the same natural-language instruction could yield different outcomes after a model update, prompt change, or tool-schema modification. An organized LLMOps framework therefore records the model identifier, prompt version, tool version, scenario data, objective weights, verifier result, and final layout. This enables traceability and makes rollback possible when a new configuration performs worse than the approved version.

The latency results reveal a practical trade-off. The proposed method is slower than a single LLM because it executes multiple reasoning and validation steps. Nevertheless, the additional latency is associated with quality assurance rather than redundant generation. For real-time manufacturing, a hierarchical trigger strategy is therefore recommended: minor flow changes can use a fast local repair agent, while major layout changes trigger the full multi-agent optimization loop.

The real-time claim is supported only when the optimization latency is embedded in a complete operational data path. A plausible workflow is as follows. IoT sensors, AGV controllers, and warehouse-management transactions publish events to a message broker; a stream-processing layer aggregates these events into a rolling material-flow matrix and monitors a drift indicator, such as the change in inter-departmental flow over a time window; the MES supplies orders, routings, and resource status, which the Planner Agent converts into constraints; when the drift exceeds a threshold, the agentic loop is triggered (6.41 s in the benchmark); and the accepted layout is evaluated in the digital twin and passed to engineers for approval. Studies on IoT–simulation integration show that a message broker, a real-time database, an MES-generated schedule, and a simulator can be combined to support decisions on live shop-floor data (Santos et al., 2019), and a digital-twin study at a manufacturing company quantified the potential of real-time autonomous decisions for production planning (Feldt et al., 2020). Distributed stream-processing frameworks

This work is licensed under a Creative Commons Attribution 4.0 International license

provide the low-latency, fault-tolerant ingestion required for the aggregation step (Isah et al., 2019). At the same time, MES–digital twin integration shows that data cleaning and synchronization can limit the refresh rate to far less than continuous updating (Lanzini et al., 2024). End-to-end latency is therefore the sum of data ingestion, aggregation, optimization, and approval, and only the optimization component (about 6 s) was measured here. Because physical relocation of equipment occurs on a time scale of hours to days, the appropriate claim is near-real-time re-planning and decision support, not real-time control.

The proposed architecture also complements rather than replaces human industrial engineers. The engineer remains responsible for defining non-negotiable constraints, approving objective weights, validating simulation assumptions, and authorizing physical implementation. The LLM system acts as an orchestration and decision-support layer. This human-in-the-loop design is particularly important where safety, regulatory compliance, or expensive equipment relocation is involved.

Compared with conventional GA, the agentic approach has a different strength profile. GA is strong at numerical exploration but does not naturally interpret a new textual requirement such as "keep the maintenance access clear, move inspection closer to assembly, and prioritize the new product family." The agentic architecture can translate such requirements into structured constraints and then call the GA as one component of the solution process. Thus, the LLM adds semantic flexibility while the optimizer retains numerical rigor.

The architecture is also compatible with digital twins. A digital twin can supply the current state of the workshop, while the agent system generates candidate changes and requests KPI evaluation. Recent research on adaptive factory layout and twin learning supports this direction, suggesting that future systems can move from periodic redesign toward continuous layout adaptation.

Despite the promising benchmark results, several limitations remain. First, the benchmark is synthetic and simplified to a grid representation. Real factories contain continuous dimensions, obstacles, utilities, safety zones, vertical transport, stochastic processing times, and heterogeneous vehicle behavior. Second, the LLM itself is treated as a generic component; model-specific effects such as context-window limits, token cost, tool-calling reliability, and multilingual robustness were not experimentally isolated. Third, the benchmark does not measure the organizational cost of human approval or physical implementation. Fourth, no ablation or equal-evaluation-budget control was performed, and latency was measured under a single hardware, network, and provider configuration.

Future research should therefore evaluate the architecture using real factory data and a digital-twin environment, compare multiple LLM families, introduce vision-language processing for CAD or floor-plan drawings, and quantify cost-sensitive triggers for re-optimization. A particularly important direction is uncertainty-aware agent behavior, in which the system abstains from autonomous action when input data are incomplete or when the candidate solution lies near a safety boundary.

This work is licensed under a Creative Commons Attribution 4.0 International license

CONCLUSIONS

This study proposed an agentic LLM architecture with an integrated LLMOps framework for near-real-time facility layout optimization and compared it with manual SLP, a GA, and a single tool-augmented LLM on a synthetic 40-scenario benchmark. The architecture achieved the lowest mean material handling cost (88.39, which is 31.2% below manual SLP and 20.0% below the GA), the lowest mean logistics distance (304.5, 26.8% below manual SLP), and the highest feasibility rate (95%), with a mean response latency of 6.41 s, only 0.48 s above the single-LLM baseline.

The study contributes (1) an architecture in which an LLM coordinates validated optimization tools and a deterministic verifier decides whether a layout may be accepted; (2) an LLMOps control layer that makes each recommendation traceable and reversible; and (3) a comparison protocol in which the objective function, constraints, and stop criteria of all methods are stated explicitly. Together, these results position LLMs as coordinators of established optimization methods rather than as replacements for them.

The findings are exploratory because they rest on a synthetic, grid-based benchmark, a single LLM configuration, and a design without ablation or equal-budget controls. Future work should validate the architecture with real factory data streamed from MES and IoT systems into a digital twin, separate the contribution of each agent and of evaluation budget, compare several LLM families, extend the model to continuous geometry and multiple objectives, and study human acceptance and uncertainty-aware abstention before any autonomous deployment.

REFERENCES

Bajoria, S., Ranjan, S., Adhitya, M., Suri, V., Hassija, V., Chamola, V., & Hussain, A. (2026). From Language Models to Agentic AI: A Survey of Autonomous, Action-Enabled, and Collaborative LLM Agents. Cognitive Computation, 18, 103. https://doi.org/10.1007/s12559-026-10619-1

Choi, H., Yu, S., Lee, D. H., Noh, S. D., Ji, S., Kim, H., Yoon, H., Kwon, M., & Han, J. (2024). Optimization of the Factory Layout and Production Flow Using Production-Simulation-Based Reinforcement Learning. Machines, 12(6), 390. https://doi.org/10.3390/machines12060390

Du, S., Zhao, J., Shi, J., Xie, Z., Jiang, X., Bai, Y., & He, L. (2026). A Survey on the Optimization of Large Language Model-based Agents. ACM Computing Surveys, 58(9), 223. https://doi.org/10.1145/3789261

Feldt, J., Kourouklis, T., Kontny, H., & Wagenitz, A. (2020). Digital twin: Revealing potentials of real-time autonomous decisions at a manufacturing company. Procedia CIRP, 88, 185–190. https://doi.org/10.1016/j.procir.2020.05.033

Guo, H., Zhu, Y., Zhang, Y., Ren, Y., Chen, M., & Zhang, R. (2021). A digital twin-based layout optimization method for discrete manufacturing workshop. The International Journal of Advanced Manufacturing Technology, 112, 1307–1318. https://doi.org/10.1007/s00170-020-06568-0

This work is licensed under a Creative Commons Attribution 4.0 International license

Guo, X., Huang, Y., Zhang, S., & Ma, G. (2026). Three-dimensional facility layout optimization using large language model-driven A-star and non-dominated sorting genetic Algorithm-II. Expert Systems with Applications, 315, 131745. https://doi.org/10.1016/j.eswa.2026.131745

Isah, H., Abughofa, T., Mahfuz, S., Ajerla, D., Zulkernine, F., & Khan, S. (2019). A Survey of Distributed Data Stream Processing Frameworks. IEEE Access, 7, 154300–154316. https://doi.org/10.1109/ACCESS.2019.2946884

Jazi, A. E., & Ameri, F. (2026). Machine learning, digital twin, and AR/VR in manufacturing facility layout planning: a review. International Journal on Interactive Design and Manufacturing. https://doi.org/10.1007/s12008-026-02629-y

Kroemer, O., Niekum, S., & Konidaris, G. (2023). Deep reinforcement learning for layout planning – An MDP-based approach for the facility layout problem. Manufacturing Letters, 38, 40–43. https://doi.org/10.1016/j.mfglet.2023.09.007

Lanzini, M., Ferretti, I., & Zanoni, S. (2024). Towards the Implementation and Integration of a Digital Twin in a Discrete Manufacturing Context. Processes, 12(11), 2384. https://doi.org/10.3390/pr12112384

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., & Kiela, D. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems, 33. https://doi.org/10.48550/arXiv.2005.11401

Li, H., Zhang, H.-Y., Liao, Y., Tian, J.-M., & Kuang, M.-S. (2025). Facility layout optimisation of an intelligent manufacturing unit based on queueing network and deep reinforcement learning. International Journal of Production Research, 64(4), 1295–1318. https://doi.org/10.1080/00207543.2025.2568181

Ma, D., Lang, S., Kute, S., Reider, R., & Müller, M. (2026). LLMOps for End-to-End Automation in Facility Layout Planning: Methodology and Application. Procedia Computer Science, 277, 1538–1547. https://doi.org/10.1016/j.procs.2026.02.192

Plaat, A., van Duijn, M., van Stein, N., Preuss, M., van der Putten, P., & Batenburg, K. J. (2025). Agentic Large Language Models, a Survey. Journal of Artificial Intelligence Research, 82. https://doi.org/10.1613/JAIR.1.18675

Santos, R., Basto, J. P. T. V., Alcalá, S. G. S., Frazzon, E. M., & Azevedo, A. (2019). Industrial IoT integrated with simulation: A digital twin approach to support real-time decision making. In EU IEOM 2019: Proceedings of the 3rd European International Conference on Industrial Engineering and Operations Management (pp. 816–828). https://doi.org/10.46254/EU03.20190225

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language Models Can Teach Themselves to Use Tools. Advances in Neural Information Processing Systems, 36. https://doi.org/10.48550/arXiv.2302.04761

This work is licensed under a Creative Commons Attribution 4.0 International license

Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., Zhao, W. X., Wei, Z., & Wen, J.-R. (2024). A survey on large language model based autonomous agents. Frontiers of Computer Science, 18(6), 186345. https://doi.org/10.1007/s11704-024-40231-1

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. International Conference on Learning Representations (ICLR 2023). https://doi.org/10.48550/arXiv.2210.03629