Near-Optimal Graph-Based Routing for Manual Picker-to-Parts Warehouses: A Case Study of an Apulian Distribution Center

Federico Signorile, Raffaele Carli, Michele Gorgoglione, and Mariagrazia Dotoli

***Abstract*— Order picking is one of the most costly activities in manual picker-to-parts warehouses, and routing quality has a direct impact on travel effort and operational efficiency. However, commercial warehouse management systems (WMSs) still often rely on simple rule-based policies because exact optimization is difficult to reconcile with real-time execution requirements. This paper presents a WMS-compatible graph-based routing method for manual warehouses based on mission-dependent graph reduction, shortest-path computation, and sequencing optimization over mission-relevant locations. Starting from the physical warehouse graph, the proposed method builds a compact reduced representation that preserves shortest-path distances while significantly decreasing the online computational burden, enabling seamless integration into existing WMSs and real-time operation. The method is validated on a real household-goods distribution center and compared with both practical rule-based routing policies typically adopted in commercial WMSs and exact optimization benchmarks. Results on both synthetic missions and real warehouse orders show substantial travel-distance reductions with runtimes fully compatible with online warehouse operation.**

I. Introduction

Order picking is widely recognized as one of the most labor-intensive and costly warehouse activities, and in manual picker-to-parts systems travel typically represents a major share of the operational effort. Consequently, routing decisions have a direct impact on productivity, lead time, and operating cost, making picker routing a central problem in warehouse operations research [1]–[3]. In practice, however, commercial warehouse management systems (WMSs) still often rely on simple rule-based routing policies [4], [5] because they are easy to implement, computationally inexpensive, and operationally robust. Although attractive in real settings, especially in manual warehouses, these policies may remain far from the best achievable route, and their performance strongly depends on the order profile, pick density, and layout structure [6]. At the same time, the literature has shown that picker routing can be formulated as a shortest-tour problem over the required picking locations, with exact and TSP-based approaches providing a principled path toward optimal or near-optimal routes [6]–[8]. Yet a practical gap remains between routing quality and operational deployability. Exact optimization is difficult to sustain under the response-time requirements of warehouse execution, whereas lightweight heuristics preserve speed at the expense of route quality [9]. For small and medium-sized companies using conventional picker-to-parts warehouses, the relevant question is therefore not whether routing can be optimized in theory, but whether intelligent routing can be integrated into an existing WMS workflow without disrupting upstream processes or violating real-time constraints.

This paper addresses this deployment gap through a real case study from a household-goods distribution center. We consider the realistic setting in which a commercial WMS already generates feasible picking missions, while routing is handled downstream under strict response-time constraints. Starting from the physical warehouse graph, we build for each mission a compact reduced representation that preserves shortest-path distances among mission-relevant locations, so that routing can be optimized without repeatedly operating on the full layout. On this reduced representation, shortest-path information is computed and the visit sequence is optimized through a TSP-based solver, after which the final route is reconstructed on the original warehouse graph. The resulting method provides a practical way to embed near-optimal routing into an existing WMS workflow without modifying upstream batching and feasibility logic. The approach is validated on a real warehouse and compared with both rule-based policies typically used in practice and exact optimization benchmarks, showing substantial travel-distance savings with runtimes compatible with online operation.

The work of F. Signorile was supported by the European Union Next-GenerationEU (National Recovery and Resilience Plan (NRRP), Mission 4 Component 1 Investments 3.3 – Italian Ministry of University and Research Decree no. 630 of 24/04/2024 - code D93D24000270003) within the Italian National Ph.D. Program in Autonomous Systems (DAuSy). The work of R. Carli, M. Gorgoglione, and M. Dotoli was supported by the Apulia Region (Italy) under the Small Medium Enterprise Development Program (P.O. PUGLIA FESR 2014-2020 Asse I – Obiettivi specifici 1a Azione 1.1 Sub-Azione 1.1.c Regolamento Regionale del 30 settembre 2014, n. 17 e s.m.i. – Titolo II – Capo 2 – Art. 27 PROGRAMMI INTEGRATI DI AGEVOLAZIONE – PIA PICCOLE IMPRESE), Project LOGISTIC 4.0 IN PUGLIA, Project no. B66I23000100007.

F. Signorile, R. Carli, and M. Dotoli are with the Department of Electrical and Information Engineering of the Polytechnic of Bari, Italy (email: f.signorile2@phd.poliba.it, {raffaele.carli, mariagrazia.dotoli}@poliba.it). M. Gorgoglione is with the Department of Mechanics, Mathematics, and Management of the Polytechnic of Bari, Italy (e-mail: michele.gorgoglione@poliba.it).

II. Operational Setting and Routing Problem

We consider a manual picker-to-parts warehouse system, where operators move within a storage area to collect items associated with customer orders. The warehouse layout is modeled as a graph $G = (V, E)$ (Fig. 1c), where nodes V represent accessible locations (e.g., aisle intersections or picking positions), and edges E represent feasible movements between locations. Each edge $(i, j) \in E$ is associated with a non-negative traversal cost $c_{ij} \in \mathbb{R}_{\geq 0}$, representing distance or travel time. A distinguished node $v_0 \in V$

represents the depot, i.e., the starting and ending point of each picking tour.

Remark 1. *We assume that the order batching process is performed upstream by the WMS, which generates picking missions that are feasible with respect to operational constraints such as capacity, weight, volumetric limits or due date constraints.*

Consistently with Remark 1, the routing problem is defined over a given picking mission, consisting of a set of locations to be visited. Let $R = \{v_1, v_2, \dots, v_n\} \subseteq V$ denote the set of picking locations associated with a mission. We define the set of relevant nodes as $V_R = \{v_0\} \cup R$ and denote its cardinality as $m = |V_R|$. The routing problem consists in determining a minimum-cost closed route on the physical warehouse graph G that starts from the depot, visits all required picking locations in R, and returns to the depot.

Remark 2. *The routing problem is defined on the physical warehouse graph G, so the final solution must be an executable route in the actual warehouse layout. Accordingly, while each required picking node must be visited, intermediate transit nodes of G may be traversed multiple times in the reconstructed route.*

From a modeling viewpoint, the routing problem on G comprises two coupled components: a path-realization component, which captures the physical travel cost between mission-relevant nodes through feasible paths in the warehouse graph, and a sequencing component, which determines the order in which such nodes are visited. This viewpoint suggests separating the geometric complexity of the warehouse layout from the combinatorial complexity of the sequencing problem.

Accordingly, for any pair of relevant nodes $i, j \in V_R$, let $d_{ij} \in \mathbb{R}_{\geq 0}$ denote the shortest-path cost between i and j in the physical warehouse graph, namely

$$d_{ij} = \min_{p \in \mathcal{P}_{ij}} \sum_{(u,v) \in p} c_{uv}, \tag{1}$$

where $\mathcal{P}_{ij}$ is the set of feasible paths connecting i and j in G.

Based on these pairwise shortest-path costs, the sequencing layer of the routing problem can be formulated on the complete directed graph

$$G_R = (V_R, E_R), \qquad E_R = \{(i,j) \in V_R \times V_R : i \neq j\},$$

where each arc $(i,j) \in E_R$ is weighted by d_{ij}. In other words, G_R (Fig. 1e) is the metric closure induced by the physical warehouse graph over the mission-relevant nodes.

The sequencing layer of the routing problem can therefore be modeled as a traveling salesman problem (TSP) on G_R, where the objective is to determine a minimum-cost tour that starts from the depot, visits each relevant node in R exactly once, and returns to the depot.

More formally, let $y_{ij} \in \mathbb{B}$, $(i,j) \in E_R$, be a binary decision variable such that $y_{ij} = 1$ if node j is visited immediately after node i, and $y_{ij} = 0$ otherwise. Furthermore, let $u_i \in \mathbb{Z}$ be an auxiliary variable representing the visiting order of node $i \in V_R$ along the tour. Let the vector of decision variables be $\mathbf{x} = col\left(\begin{bmatrix} y_{ij} \\ u_i \end{bmatrix}\right)$. The TSP on the complete graph $G_R = (V_R, E_R)$ can be formulated as follows. The objective is to minimize the total traversal cost:

$$J = \sum_{(i,j) \in E_R} d_{ij}\, y_{ij}. \tag{2}$$

To ensure that each required node is visited exactly once, the following degree constraints are imposed:

$$\sum_{j \in V_R \setminus \{i\}} y_{ij} = 1, \qquad \forall i \in V_R, \tag{3a}$$

$$\sum_{i \in V_R \setminus \{j\}} y_{ij} = 1, \qquad \forall j \in V_R. \tag{3b}$$

To eliminate subtours, we enforce:

$$u_i - u_j + m\, y_{ij} \leq m - 1, \qquad \begin{array}{c} \forall (i,j) \in E_R, \\ i \neq v_0\ , j \neq v_0, \end{array} \tag{4a}$$

$$u_{v_0} = 0, \tag{4b}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall (i,j) \in E_R, \tag{4c}$$

$$u_i \in \{1, \dots, m-1\}, \qquad \forall i \in V_R \setminus \{v_0\}. \tag{4d}$$

Finally, the TSP can be formulated as

$$\underset{\mathbf{x}}{\text{minimize}}\ \ J(\mathbf{x})\ \ \text{subject to}\ \ (3a) - (4d). \tag{5}$$

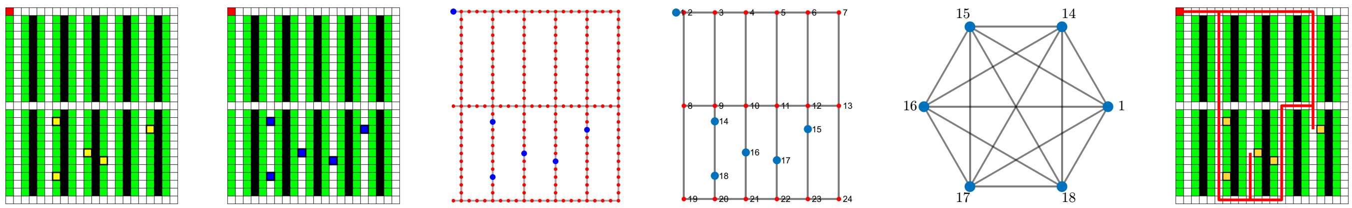
Problem (5) is a standard integer linear programming (ILP) formulation of the TSP on the complete graph G_R. Its solution defines a Hamiltonian cycle over V_R [10], namely a closed tour that starts from the depot, visits each required node exactly once at the sequencing level, and returns to the depot.

While problem (5) provides a compact formulation of the sequencing layer, its practical use in warehouse operations still raises key challenges. Constructing G_R requires computing the pairwise shortest-path costs d_{ij} from the physical warehouse graph G, while returning an executable picker route also requires path-level information for route reconstruction. Moreover, problem (5) is NP-hard [11], so solving it to optimality becomes impractical as the mission size grows, which conflicts with real-time WMS requirements.

III. Proposed WMS-Compatible Routing Method

Starting from the physical warehouse graph G, the depot v_0, and the set of required picking locations R, we propose a routing method that combines mission-dependent graph reduction, shortest-path computation on a compact warehouse representation, sequence optimization, and physical route reconstruction. Rather than repeatedly operating on the full warehouse graph G, the proposed method constructs, for each mission, a reduced graph that preserves the shortest-path distances among relevant nodes while significantly decreasing the number of nodes and edges involved in online computations.

An overview of the method is illustrated in Fig. 1.



(a) Mission layout (b) Access nodes (c) Graph G (d) Graph G' (e) Graph G_R (f) Picking route

Fig. 1. Illustration of the proposed routing method on an example grid-like warehouse graph. (a) A picking mission on the warehouse layout: the red cell denotes the depot, yellow cells denote the required storage locations, white cells are accessible locations, green cells are storage locations, and black cells represent obstacles. (b) Mission-relevant access nodes associated with the required picks, including the depot. (c) Physical warehouse graph G, where blue nodes denote mission-relevant locations and red nodes denote the remaining graph nodes. (d) Reduced graph G', obtained by retaining relevant and structural nodes and contracting corridor segments. (e) Complete graph G_R used for sequencing, whose edge weights represent shortest-path distances between mission-relevant nodes. (f) Physical reconstruction of the final picking route on the warehouse layout.

A. Instance-Dependent Graph Reduction

The proposed method begins by constructing a compact mission-dependent representation of the physical warehouse graph. We first describe this reduction for corridor-based layouts, where linear transit segments can be contracted without affecting the shortest-path structure relevant to the routing problem. The same principle can be extended to more irregular layouts, as discussed later in this section. The key idea is to retain only mission-relevant nodes and structural branching points, while contracting linear corridor segments of non-relevant transit nodes into single weighted edges. In this way, the reduced graph preserves the shortest-path distances needed by the routing problem while significantly decreasing the number of nodes and edges involved in the subsequent online computations.

In the following, we assume that G is a connected undirected weighted graph. For any node $v \in V$, let $\mathcal{N}(v)$ denote its set of adjacent nodes and let $\deg(v) := |\mathcal{N}(v)|$ denote its degree. The reduction is built with respect to the mission-relevant node set $V_R \subseteq V$.

Definition 1. *A* corridor *is a maximal simple path* $P = (v_1, \dots, v_k)$ *in* G *such that all internal nodes satisfy* $\deg(v_i) = 2$ *for* $i = 2, \dots, k-1$, *while the endpoints satisfy* $\deg(v_1) \neq 2$ *and* $\deg(v_k) \neq 2$.

Accordingly, the reduced graph $G' = (V', E')$ (Fig. 1d) is constructed by retaining all relevant nodes and all structural nodes, namely $V' \leftarrow V_R \cup \{v \in V : \deg(v) \neq 2\}$, and by contracting each maximal path of non-relevant degree-2 nodes into a single weighted edge between its endpoints. The weight of each reduced edge is equal to the cumulative cost of the corresponding path in the original graph. Under this construction, every node $w \in V \setminus V'$ is non-relevant and satisfies $\deg(w) = 2$. Therefore, if such a node is reached while traversing a corridor, there exists a unique successor different from the predecessor node. This property ensures that the corridor can be followed unambiguously until another retained node in V' is encountered.

Algorithm 1 summarizes the mission-dependent reduction procedure. Starting from the retained node set V', the algorithm builds the reduced edge set E' by traversing corridor segments from each retained node, while maintaining a set $\mathcal{E}_{\text{proc}}$ of processed original edges to avoid duplicate traversals. For each retained node $u \in V'$, the algorithm explores all adjacent directions by considering each neighbor $z \in \mathcal{N}(u)$ (lines 2-3). If the original edge $\{u, z\}$ has already been processed, the algorithm skips it. Otherwise, it starts traversing the corresponding corridor segment from u toward z. To this end, the current node is initialized as $w \leftarrow z$, the predecessor as $p \leftarrow u$, the cumulative length as $\ell \leftarrow c_{uz}$, and the set $\mathcal{E}_{\text{curr}}$ of edges belonging to the currently explored corridor is initialized with $\{u, z\}$ (lines 5-6). The corridor is then followed until another retained node is reached. Since every internal node $w \notin V'$ is non-relevant and has degree two, the next step is uniquely determined by the successor different from the predecessor p. The algorithm therefore updates ℓ, records the traversed edge in $\mathcal{E}_{\text{curr}}$, and advances until a node in V' is reached (lines 7-12). At that point, if $u \neq w$, the whole corridor is replaced in G' by a single reduced edge $\{u, w\}$ with weight ℓ. Finally, all original edges collected in $\mathcal{E}_{\text{curr}}$ are added to $\mathcal{E}_{\text{proc}}$, and the procedure continues from the next unexplored direction (lines 13-16).

Under a standard adjacency-list representation of G, the construction of V' requires a linear scan of the node set, while each original edge is traversed only when its corridor is explored and then marked as processed. Hence, no corridor segment is explored more than once, and the overall complexity of Algorithm 1 is $O(|V| + |E|)$.

Proposition 1. The reduction preserves the shortest-path distances between all pairs of nodes in V_R.

Proof sketch. Each contracted corridor segment is replaced by a single edge whose weight equals the total cost of the corresponding original path. Hence, every path between two nodes in V_R admits a cost-equivalent representation in G' and vice versa. Therefore, shortest-path distances between mission-relevant nodes are preserved.

Remark 3 highlights an implementation-level acceleration of Algorithm 1 for static layouts.

Remark 3. *In static layouts, the online construction can be accelerated by precomputing the corridor decomposition offline. In this case, the reduced graph is assembled at corridor level and the online cost scales with the number of activated corridors and mission-relevant breakpoints, rather than with* $|V| + |E|$.

Algorithm 1 Reduced-graph construction

Require: $G = (V, E),\ \ V_R \subseteq V$
Ensure: $G' = (V', E')$
$V' \leftarrow V_R \cup \{v \in V : \deg(v) \neq 2\},\ \ E' \leftarrow \emptyset,\ \ \mathcal{E}_{\text{proc}} \leftarrow \emptyset$
for all $u \in V'$ **do**
  for all $z \in \mathcal{N}(u)$ **do**
    if $\{u, z\} \notin \mathcal{E}_{\text{proc}}$ **then**
      $w \leftarrow z,\ p \leftarrow u,\ \ell \leftarrow c_{uz}$
      $\mathcal{E}_{\text{curr}} \leftarrow \{\{u, z\}\}$
      while $w \notin V'$ **do**
        $x \leftarrow$ the unique node in $\mathcal{N}(w) \setminus \{p\}$
        $\ell \leftarrow \ell + c_{wx}$
        add $\{w, x\}$ to $\mathcal{E}_{\text{curr}}$
        $p \leftarrow w,\ w \leftarrow x$
      end while
      if $u \neq w$ **then**
        add edge $\{u, w\}$ to E' with weight ℓ
      end if
      $\mathcal{E}_{\text{proc}} \leftarrow \mathcal{E}_{\text{proc}} \cup \mathcal{E}_{\text{curr}}$
    end if
  end for
end for

Although the proposed reduction is naturally suited to corridor-based warehouses, the same principle can be extended to more irregular layouts by partitioning the warehouse graph into sectors and retaining sector access nodes together with the mission-relevant picking locations. Moreover, Algorithm 1 is formalized here for connected undirected weighted graphs, but the same reduction principle can be generalized to directed graph representations provided that corridor definitions and contraction rules are adapted to account for arc directionality.

B. Shortest-Path Computation on the Reduced Graph

Once G' has been constructed, the pairwise shortest-path costs d_{ij} between all nodes in V_R are computed on the reduced graph and used to build the complete graph $G_R = (V_R, E_R)$, where each arc $(i, j) \in E_R$ is weighted by the corresponding shortest-path cost d_{ij}.

An efficient way to obtain these distances is to run Dijkstra's algorithm once from each node in V_R. Since $m := |V_R|$, performing this computation directly on the original warehouse graph G would require

$$O(m(|E| + |V|) \log |V|)\,,$$

assuming a binary-heap implementation [12]. By contrast, if the reduced graph G' is used, and the generic reduction of Algorithm 1 is executed online, the overall cost becomes

$$O(|V| + |E|) + O(m(|E'| + |V'|) \log |V'|)\,.$$

The first term accounts for the construction of the reduced graph, while the second term corresponds to the m shortest-path computations on G'. Since warehouse graphs are typically sparse and corridor contraction often yields $|V'| \ll |V|$ and $|E'| \ll |E|$, the proposed reduction substantially decreases the cumulative cost of shortest-path computation in practice.

Under the implementation described in Remark 3, the reduction step can be further accelerated by exploiting precomputed corridor-level information. In that case, the online assembly of G' no longer requires scanning the full graph, and its cost becomes proportional to the mission-activated corridor structure rather than to $|V| + |E|$.

Beyond reducing computational effort, this approach also avoids the need to precompute and maintain full routing tables on the original warehouse graph, which may become memory-intensive when both pairwise distances and route-reconstruction information must be stored.

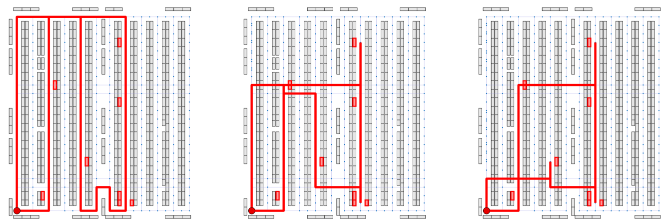

(a) (b) (c)

Fig. 2. Functional excerpt of the graph-based warehouse digital twin used in the case study, with routing paths obtained for one representative real order using: (a) S-Shape, (b) SSP, and (c) the proposed method with LKH sequencing. The figure is not to scale for confidentiality reasons.

C. Sequencing Solution

Once the complete graph G_R has been constructed, the sequencing layer of the routing problem is solved over the mission-relevant node set V_R. This stage is solver-agnostic: in principle, any TSP algorithm can be adopted, since the reduced-graph construction and the shortest-path computations are independent of the specific optimization engine used to determine the visit sequence.

In the present method, the sequencing problem is solved by the Lin–Kernighan–Helsgaun (LKH) heuristic [13]. LKH is an edge-exchange local-search method that iteratively improves a current tour by exploring variable-depth k-opt moves within a restricted candidate set, thereby focusing the search on the most promising edges while keeping the computational effort low. This combination makes it possible to obtain solutions that are typically very close to the optimum at runtimes compatible with online warehouse operations.

D. Physical Route Reconstruction

Once the sequencing problem on G_R has been solved, the final picker route is therefore reconstructed by retrieving, for each pair of consecutive nodes in the sequence, the corresponding shortest path computed as discussed in Section III-B, and by concatenating these path segments in order. This final step reconnects the abstract sequencing solution on G_R with its practical execution in the warehouse (Fig. 1f).

IV. Case Study

The proposed method is assessed on the warehouse of *Consulenza & Logistica Villa Franca S.r.l.*, a household-goods distribution center operating under a manual picker-to-parts policy in Apulia, Italy. The warehouse is managed through a commercial WMS provided by a software vendor and representative of the solutions commonly adopted in medium-sized companies of this type. In the current operating setting, the WMS generates feasible picking missions upstream, whereas route execution is handled downstream according to simple operational rules and operator practice. This makes the case study particularly suitable for evaluating whether near-optimal routing can be embedded into an existing WMS workflow without modifying upstream WMS logic.

A. Simulation Setup

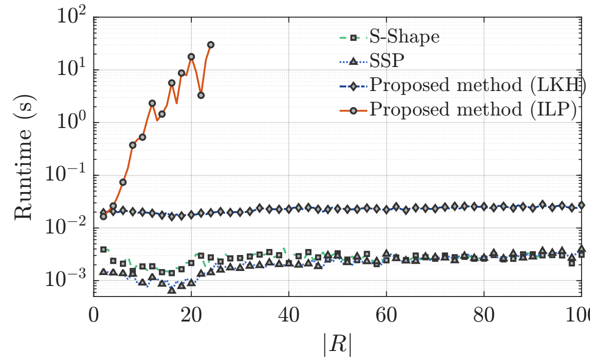
A graph-based digital twin of the warehouse was built from the available AutoCAD layout and WMS location-mapping data, and used to derive the physical graph $G = (V, E)$. For confidentiality reasons, only a functional excerpt of the layout and of its corresponding graph-based representation is shown in Fig. 2. The proposed method is compared against three routing strategies: (i) the WMS-induced S-Shape baseline, based on a predefined serpentine ordering of storage positions; (ii) a sequential shortest-path policy (SSP), which preserves the WMS sequence but computes shortest paths between consecutive picks; and (iii) the complete proposed method, combining graph reduction, shortest-path computation, TSP-based sequencing, and route reconstruction. In the online configuration, sequencing is handled by LKH, while the exact ILP formulation is used only as a benchmark for tractable instances.

All simulations are conducted in MATLAB R2025b. All components of the compared routing pipelines are developed in-house, except for the LKH solver used in the sequencing stage of the proposed online configuration, which relies on the official LKH-3.0 implementation [14] and is invoked through a Python wrapper. The experiments are run on a notebook equipped with an Intel Core i7-12700H CPU (2.30 GHz, 14 cores, 20 threads) and 16 GB DDR4 RAM.

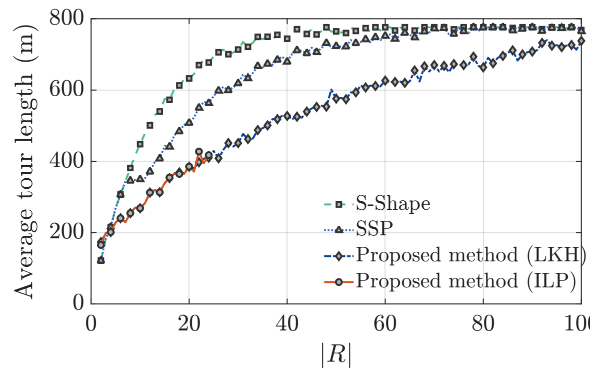
B. Controlled Mission Scenarios on the Real Layout

The first experimental campaign is based on controlled synthetic missions generated on the real warehouse graph. More precisely, random picking instances are generated with increasing cardinality of the required pick set, from $|R| = 2$ up to $|R| = 100$. For each mission size, 25 independent random instances are generated and solved by all the considered routing methods. To reflect online operational requirements, a runtime limit of 30 s is imposed on all algorithms. If the time limit is reached before proving optimality, the best feasible solution found so far is accepted.

The average results of this experiment over the 25 instances are reported in Fig. 3, using runtime and tour length as performance indicators. For methods that do not return a feasible solution within the assigned time limit, the corresponding metrics are averaged only over the solved instances.



(a) Runtime comparison in logarithmic scale.



(b) Comparison of the average tour length.

Fig. 3. Performance comparison of the considered routing methods. (a) Runtime comparison including the exact ILP benchmark in logarithmic scale. (b) Comparison of the average tour length.

The preprocessing stage, including mission-dependent graph reduction and shortest-path computation between relevant nodes, consistently required only a negligible fraction of the total runtime.

As shown in Figure 3, the exact ILP benchmark rapidly reaches the imposed 30 s time limit as the mission size increases, while the other methods remain computationally lightweight. S-Shape and SSP run in milliseconds, and the proposed LKH-based method remains within a few tens of milliseconds, which is compatible with online operation. For the instances solved by the exact benchmark, LKH matches the ILP solutions. In terms of route length, the proposed method consistently provides the best online performance: SSP improves over S-Shape by optimizing physical connections while preserving the WMS sequence, whereas the proposed method further improves the result by optimizing the visit sequence. This advantage is most pronounced for spatially sparse missions, where the order in which distant picking locations are visited strongly affects the total route length. As pick density increases, the gap progressively decreases, since the optimal tour tends to approach a more structured aisle-based traversal and simpler routing policies become comparatively more competitive.

C. Real Order Validation on Warehouse Execution Data

The proposed method was further evaluated on 116 real warehouse orders processed over two operating days by different operators. For each order, the required picking locations were reconstructed from execution data and solved offline using S-Shape, SSP, and the complete proposed routing method using LKH as the sequencing solver. This experiment is intended to assess the practical relevance of the proposed method under realistic operating conditions, where the mission size, item distribution, and storage dispersion are determined by actual warehouse activity rather than by synthetic random generation. In this case, all methods are evaluated on the same real orders, so that performance differences can be directly attributed to the routing strategy.

Figure 2 shows one representative real order, while Table I reports the average route length and computation time over the full dataset. The results show a clear performance gap among the compared routing strategies on real warehouse orders. In this dataset, the required picking locations are often spatially dispersed across distant areas of the layout, i.e., a regime in which the visit sequence plays a major role in the overall route quality. In this setting, SSP already provides a substantial improvement over the conventional S-Shape policy by replacing rule-based aisle traversals with shortest physical connections while preserving the WMS-induced visit order. The proposed approach further strengthens this effect by explicitly optimizing the sequencing layer, yielding the shortest routes among the compared methods. At the same time, the computational times remain in the order of a few milliseconds for S-Shape and SSP and a few tens of milliseconds for the proposed online method using LKH as sequencing engine, which is fully compatible with online warehouse operation. These results support the practical feasibility of the proposed method as a routing module that can be integrated downstream of existing commercial WMSs, without requiring changes to the upstream order management and batching logic, while still providing a substantial improvement over the routing policy currently induced by the management system.

From an operational perspective, the reduction in traveled distance can be interpreted as a proxy for reduced operator travel effort and reduced non-value-added time. Under a constant-speed assumption, route length savings translate directly into a proportional reduction in theoretical travel time. Although this does not fully capture the total order fulfillment time, which also depends on picking actions, local congestion, and operator-specific execution behavior, it still provides a meaningful estimate of the potential operational benefit. Therefore, improvements in routing efficiency can be translated into business-relevant KPIs such as lower travel time per order, reduced cumulative operator workload over a shift, and increased effective picking capacity.

V. Conclusions

This paper presented a WMS-compatible routing method for manual picker-to-parts warehouses, combining mission-dependent graph reduction, shortest-path computation, and TSP-based sequencing. Results on synthetic missions and real warehouse orders show substantial travel-distance reductions with runtimes compatible with online operation, supporting integration downstream of existing WMSs without modifying upstream planning logic. Future work will address joint batching-routing optimization, richer motion-cost models including turning and acceleration effects, and congestion-aware data-driven calibration from execution logs.

TABLE I
Comparison of routing strategies on 116 real warehouse orders.

Method	Avg. tour length (m)	Avg. runtime (ms)
S-Shape	759	8
SSP	631	10
Proposed method (LKH)	412	19

Acknowledgements

The authors thank the Staff of *Consulenza & Logistica Villa Franca S.r.l.*, and particularly Logistic Manager P. Di Castri and Technical Lead A. Ruggieri, for their support in the preparation of this paper.

References

[1] R. De Koster, T. Le-Duc, and K. J. Roodbergen, "Design and control of warehouse order picking: A literature review," *European journal of operational research*, vol. 182, no. 2, pp. 481–501, 2007.

[2] G. Casella, A. Volpi, R. Montanari, L. Tebaldi, and E. Bottani, "Trends in order picking: A 2007–2022 review of the literature," *Production & Manufacturing Research*, vol. 11, no. 1, p. 2191115, 2023.

[3] M. Dotoli, N. Epicoco, M. Falagario, N. Costantino, and B. Turchiano, "An integrated approach for warehouse analysis and optimization: A case study," *Computers in Industry*, vol. 70, pp. 56–69, 2015.

[4] R. W. Hall, "Distance approximations for routing manual pickers in a warehouse," *IIE transactions*, vol. 25, no. 4, pp. 76–87, 1993.

[5] C. G. Petersen, "An evaluation of order picking routeing policies," *International Journal of Operations & Production Management*, vol. 17, no. 11, pp. 1098–1111, 1997.

[6] M. Masae, C. H. Glock, and E. H. Grosse, "Order picker routing in warehouses: A systematic literature review," *International journal of production economics*, vol. 224, p. 107564, 2020.

[7] C. Theys, O. Bräysy, W. Dullaert, and B. Raa, "Using a tsp heuristic for routing order pickers in warehouses," *European Journal of Operational Research*, vol. 200, no. 3, pp. 755–763, 2010.

[8] H. D. Ratliff and A. S. Rosenthal, "Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem," *Operations research*, vol. 31, no. 3, pp. 507–521, 1983.

[9] K. R. Mokarrari, T. Sowlati, J. English, and M. Starkey, "Optimization of warehouse picking to maximize the picked orders considering practical aspects," *Applied mathematical modelling*, vol. 137, p. 115585, 2025.

[10] J. A. Bondy and U. S. R. Murty, *Graph theory*. Springer Publishing Company, Incorporated, 2008.

[11] V. Traub and J. Vygen, *Approximation algorithms for traveling salesman problems*. Cambridge University Press, 2024.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[13] K. Helsgaun, "An effective implementation of the lin–kernighan traveling salesman heuristic," *European journal of operational research*, vol. 126, no. 1, pp. 106–130, 2000.

[14] "Lkh-3," 2017, available: https://webhotel4.ruc.dk/ keld/research/LKH-3/.