A Visualization Approach for Monitoring Order Processing in E-Commerce Warehouse

Junxiu Tang, Yuhua Zhou, Tan Tang, Di Weng, Boyang Xie, Lingyun Yu, Huaqiang Zhang, and Yingcai Wu

Abstract— The efficiency of warehouses is vital to e-commerce. Fast order processing at the warehouses ensures timely deliveries and improves customer satisfaction. However, monitoring, analyzing, and manipulating order processing in the warehouses in real time are challenging for traditional methods due to the sheer volume of incoming orders, the fuzzy definition of delayed order patterns, and the complex decision-making of order handling priorities. In this paper, we adopt a data-driven approach and propose OrderMonitor, a visual analytics system that assists warehouse managers in analyzing and improving order processing efficiency in real time based on streaming warehouse event data. Specifically, the order processing pipeline is visualized with a novel pipeline design based on the sedimentation metaphor to facilitate real-time order monitoring and suggest potentially abnormal orders. We also design a novel visualization that depicts order timelines based on the Gantt charts and Marey's graphs. Such a visualization helps the managers gain insights into the performance of order processing and find major blockers for delayed orders. Furthermore, an evaluating view is provided to assist users in inspecting order details and assigning priorities to improve the processing performance. The effectiveness of OrderMonitor is evaluated with two case studies on a real-world warehouse dataset.

Index Terms—Streaming data, time-series data, e-commerce warehouse, order processing

1 Introduction

E-commerce, especially business-to-consumer services, such as Amazon and Alibaba, are booming and has become a daily part of modern life owing to its convenience [19, 33, 45, 53]. After consumers purchase goods online, retailers will send orders to the warehouse, where the orders are processed on a tight schedule for the promised delivery due date [11]. At the beginning of the logistics from retailers to consumers, order processing in the warehouse has a great influence on the e-commerce delivery efficiency. Order processing in modern e-commerce warehouse systems mainly depends on the assembly line that includes automated mechanical facilities and human operators [11, 12]. The assembly line generates a mass of streaming data in real time, among which orders that cost more time than the processing schedule exist owing to unpredictable situations, such as order picking faults. Monitoring, analyzing, and evaluating the delay issues are the key problems in order processing, which is difficult due to (a) the rapidly incoming order volume, (b) unclear delayed threshold, and (c) difficult decision making on handling priority.

Prior work in operational research applied algorithmic approaches to increase the efficiency of order processing by optimizing routing [13, 17, 47] and order batching [27, 41, 71]. However, most of the algorithms are not designed for real-time scenarios, and the optimized results are not intuitive enough to trigger further actions from experts. Recent studies propose a few visual analytics methods to solve the optimization problems in warehousing, such as assembly line monitoring [68] and manufacturing schedule visualization [29]. However, these approaches are insufficient for monitoring and analyzing the event data of e-commerce warehouse order processing in real time, as these data comprise irregular temporal patterns and hierarchical structures due to the complexity of the pipeline.

- *J. Tang, Y. Zhou, T. Tang, D. Weng, B. Xie, and Y. Wu are with State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China. E-mail: tangjunxiu, zhouyuhua, tangtan, dweng, xboyang, ycwu@zju.edu.cn. Y. Wu is the corresponding author.*
- *L. Yu is with Department of Computing, Xi'an Jiaotong-Liverpool University, Suzhou, China. E-mail: lingyun.yu@xjtlu.edu.cn.*
- *H. Zhang is with Alibaba Group, Hangzhou, China. E-mail: huaqiang.zhq@cainiao.com.*

Manuscript received 21 Mar. 2021; revised 13 June 2021; accepted 8 Aug. 2021. Date of publication 1 Oct. 2021; date of current version 22 Dec. 2021. Digital Object Identifier no. 10.1109/TVCG.2021.3114878

By collaborating with the experts from the warehouse management industry, we attempt to propose a visual analytics solution that helps them better manage the delay issues in order processing. The following three challenges, summarized from the monitoring, analyzing, and evaluating stages, arise in developing such a solution.

Real-time monitoring of incoming orders. Warehouse managers can be aware of delayed orders through intuitive monitoring. However, the warehouse faces rapidly incoming orders from online users in real time, which brings an uncertain workload to monitoring the status of order processing. Moreover, the streaming data of order processing in e-commerce warehouses have complex features. First, orders have hierarchy data structures. For example, one order can include multiple stock keeping units (SKUs) to pick and a few packages to pack. Second, different orders are processed simultaneously in parallel operation stations. These features bring challenges to the existing visual analytics approach [68] on monitoring assembly lines in smart factories, which has limitations in presenting parallel tasks and hierarchy data structures. Thus, we need a new visual design to monitor complex incoming order data in real-time.

Deep analysis of delayed orders. Practical operations in warehouses such as order-picking and packing generate irregular temporal data, resulting in no clear border between normal and abnormal patterns of order processing, thereby making distinguishing the delay patterns based on fixed thresholds difficult. For example, warehouse managers tend to accumulate and batch a certain number of orders with the same SKUs to save picking costs, which results in an uncertain waiting time for incoming orders. Another example is that fragile products need careful packing; but cost more time than other products, which should not be regarded as delay cases. The existing approach [26, 68] can diagnose anomalies based on algorithms but lack comprehensive considerations of individual attributes (such as order SKUs) and environmental factors (such as practical producing abilities). Thus, we need a new approach to integrating human's knowledge to analyze the processing orders and identify delayed orders for efficient subsequent handling.

Quantified evaluation of delay-handling priority. When faced with different kinds of delayed orders, warehouse managers have difficulty deciding which one needs to be handled first. Under limited producing abilities and a tight delivery schedule, reasonably batching delayed orders based on priority can minimize handling time cost. However, various criteria must be evaluated, for example, delivery schedule and handling complexity, which are also difficult to quantify. How to rank the handling priority based on quantitative criteria remains an unresolved problem. Hence, we need a new method to evaluate the handling priority of delayed orders.

For the first challenge, we propose a novel visual design that can present complex streaming data and a monitoring view to check the order processing status in real time. For the second challenge, we integrate Marey's graph with the Gantt chart in the analyzing view to support inspection and explorations on the processing records of possibly delayed orders. For the third challenge, we develop an evaluating view to inspect order details and interactively evaluate the handling priority. The contributions of this paper are summarized as follows:

- We characterize the design requirements and domain problems in monitoring, analyzing, and evaluating order-delaying issues in e-commerce warehouses.
- We develop a comprehensive visual analytics system based on the design requirements. The system provides real-time monitoring, deep analysis and further decision-making.
- We propose two tailored visualization designs for presenting parallel tasks and hierarchy data structures of process data. One is to illustrate streaming processing data, and the other is to visualize the historical processing records.
- We evaluate our system with real-world order processing data by using two case studies and three expert interviews.

2 Related Work

To the best of our knowledge, few approaches focus on the delay issues of the streaming warehouse order processing data. First, previous work of streaming data visualization lacks considerations in the domain of e-commerce order. Moreover, the existing visualization approach to monitor industrial processes is not suitable for the warehouse scenario. In this section, we introduce related work about streaming data visualization and process visualization.

2.1 Streaming Data Visualization

Streaming data have high updating velocity and volatility, which widely exist in many domains, such as air-traffic control, social media, and cyber threat mitigation [18, 31]. Several visualization methods were proposed to analyze streaming data for event detection [50], situation awareness [21], and dimensionality reduction [22]. Huron et al. [28] introduced an innovative visual metaphor called visual sedimentation to visualize streaming data and smooth the incoming transition of continuously updating data. Other metaphors such as dial [20] and storyline [52] were also proposed for efficient visualizations. Krstajic et al. [30] developed CloudLines, a novel visualization technique that can present event episodes in online news streams. Researchers also developed novel visualization systems to analyze continuously updating information in text data [38, 63] and spatiotemporal data [14, 15].

However, few studies tackle e-commerce order data. This kind of data has high updating speed, non-equidistant generating time intervals, and different data features from streaming data that have been researched before, such as hierarchy content and phased processing. We analyze the domain requirements of e-commerce warehouse order processing and propose a tailored visualization method based on the drop chart design, one of visual sedimentation visualizations [28]. Specifically, our proposed designs can fulfill the situation awareness of delay by presenting the status of orders in each processing procedure.

2.2 Process Visualization

Process visualization is a sub-field of information visualization, which has been used to monitor instrument conditions and producing processes since the industrial age [40]. With the recent development of digital instruments and smart facilities in industrial factories, many visualization methods are proposed to visualize the manufacturing process data and support specific analyzing requirements of different domains. Matković et al. [40] integrated levels of detail and foucs+context technique for virtual instrument process monitoring. Aigner et al. [6] introduced a novel glyph design called *PlanningLines* that is extended from *LifeLines* [46] to represent temporal uncertainties in project management and medical treatment planning. Researchers have explored several types of visualization to present process data, for example, Pathway Waterfall [32], the Gantt chart [6, 29] and Marey's graph [44, 68]. LiveGantt [29] is a novel schedule visualization tool that is integrated with a layout algorithm and interactions to improve the scalability of conventional Gantt charts. Marey's graph is a classical visualization for illustrating public transportation schedules [55]. Inspired by Marey's graph, Palomo et al. [44] introduced an innovative visual analytics system called TR-EX, which applied kernel density estimation for presenting a mass of transportation data at multiple scales. Moreover, based on Marey's graph, Xu et al. [68] proposed ViDX, a novel visual analytics system to diagnose abnormal events of assembly lines in smart factories. They applied a time-aware, outlier-preserving visual aggregation technique on the graph to reveal the historical anomalies and designed a radial graph to monitor the real-time processes. However, the system lacked details of parallel working stations and faced scalability problems. Besides visualizing the temporal processes, researchers also proposed approaches for visualizing processes in monitoring facility conditions [62, 72], simulating fluid [8], analyzing sports [66, 69], and exploring abnormal patterns [67, 70].

The previous studies about manufacturing process visualization [29, 68] are highly related to our work. However, these studies assume that the dealing workload and producing schedule are stable. The situation in e-commerce warehousing is different because the individual attributes (such as order SKUs) of online orders are unpredictable, which brings an uncertain processing workload. These uncertain workloads further result in a dynamic producing schedule due to specific order-picking operational strategies [11]. All these uncertainties make it hard to directly judge delaying events based on the spent time interval [29] and the proposed time-aware outlier-preserving visual aggregation technique [68]. Inspired by these studies, we propose new visual designs and develop a novel system for monitoring, analyzing, and evaluating order delaying issues in e-commerce warehouses.

3 Background

In this section, we introduce the order processing workflow in the e-commerce warehouse and the related data format. We also summarize major requirements to guide the design of a visualization system that enables warehouse managers to monitor, analyze, and evaluate delay issues in order processing pipelines.

3.1 E-commerce Warehouse Order Processing

Warehousing research has a large scope, including storage management [24], batching optimization [41], picking strategy [27, 45, 71], and routing design [13]. Among these heterogeneous warehousing system designs, the customers that a warehouse serves categorize the warehouse type, such as retail warehouse and e-commerce warehouse, which further leads to special warehousing strategies [9]. E-commerce warehouses face (a) *small orders* that contain very few goods in each order, (b) *large assortment* including various SKUs and even niche goods, (c) *tight delivery deadline* promised to consumers as part of e-commerce service, and (d) *varying workloads* that need scalable processing capacities, especially during sales promotion [11]. These features distinguish the order processing workflow in e-commerce warehouses from manufacturing warehousing [51]. In this work, we abstract the order processing workflow applied in our collaborated e-commerce warehouses, which also has been applied in many e-commerce warehouses according to the domain experts' interview and the literature review [11]. The order processing pipeline is illustrated in Fig. 1 (a), which mainly consists of the following procedures:

Preprocessing: After the e-commerce warehouse receives an online **order request**, the warehouse storage system checks the inventory of all the **goods** included in the **order**. If the inventory of the included goods is not sufficient, the order will be blocked in this procedure until the inventory is sufficient. Orders containing goods that do not meet the inventory are blocked in this procedure until inventory suffices.

Aggregating: Following different order-picking strategies [11, 25], operators aggregate orders that meet the picking conditions into a **picking list**. A picking list contains multiple individual orders. For example, in Fig. 1(b), the first three orders include similar goods (five t-shirts and two headphones in total), which can be batched in one picking operation. Aggregating can save order pickers' travel distances and picking time, which is crucial to the efficiency of the e-commerce

warehouse. However, due to the varieties of incoming online orders, the aggregating result is dynamic, which leads to an uncertain processing time for different picking lists.

Picking & Sorting: According to picking lists, **operators** go to related shelves and pick the ordered goods. In the picking operation, operators bring all the ordered goods in the picking list as total, which needs subsequent sorting into isolated individual orders. In Fig. 1(b), the first three orders that include three blue t-shirts, two orange t-shirts and two headphones, are first picked together, and then sorted into three groups according to different users' orders.

Packing & Weighting: Operators check the quality of the goods (for example, whether the goods are damaged) and compare the SKUs of actual picked goods with each order request. Orders with damaged goods or incorrect SKUs are blocked in this procedure and await a **Reviewing**. If everything is as requested, operators pack the goods of one order together into **packages**. In this procedure, the operating time is also uncertain in each order. First, fragile goods such as glass products need special attention for safe delivery. Therefore, operators need to add customized cushioning carefully, which takes more time than non-fragile goods. Second, orders that include too many goods are packed into multiple packages due to the limited package size, which costs more packing time than smaller orders. After packing, packages are weighted automatically in a conveyor to comply with the requirements of the express.

Outbound: Through the above procedures, an order request is finally ready for delivery. The package(s) is(are) passed to the express and then tagged as outbound, which means that the order processing in the warehouse is finished.

Reviewing: Operators double check the faulty orders and address the fault to fulfill the order request. For example, damaged goods are replaced with sound ones, or the correct number of goods is supplemented in accordance with user-required SKUs.

Resetting: If the goods have already been picked or packed but the orders are cancelled by customers, the processing of these orders is intercepted. Operators place the goods back on the shelves.

In practice, leading logistics companies have deployed systems for order processing monitoring in e-commerce warehouses [43, 49]. For example, Amazon has a system that monitors the status of conveyor belts [43], while Cainiao deployed a cloud-based video-monitoring system to identify abnormalities through computer vision technology [49]. Our collaborated warehouse currently adopts a table-based interface for monitoring order processing, which presents the sum of orders in each procedure at different time intervals as shown in (Fig. 3 (a)). However, these approaches are insufficient to monitor order processing, which involves review analysis and priority evaluation. Visual analytics techniques can help address these issues by presenting the real-time processing status, historical records, and priority ranking on the level of orders or picking lists.

The literature review (Sect. 2.2) indicates that the existing approach for process visualization cannot be directly applied in e-commerce warehouse order processing because of several unique features in this domain. First, each e-commerce order has individual goods SKU requirements rather than the same producing workpiece in factory assembly lines. Second, cancellations are initiated by the online user during every procedure in order processing, for example, the third order in Fig. 1(b). In this condition, even though no abnormal event occurs in the pipeline, the cancelled order should stop being processed and wait for a reset in the storage. Third, the order processing in e-commerce is more uncertain due to the irregular order arrival patterns, such as variable order details (goods' types and numbers) and unpredictable volumes, making the overall temporal sequence of each order processing different and irregular [11, 33, 45]. Fourth, the operation on one procedure does not take place immediately after the previous one. For example, newly arrived orders need to wait for aggregation, where the processing priority depends on specific order picking strategies. Moreover, large order volumes put pressure on the limited producing power, thereby leading to blocks on the processing pipeline.

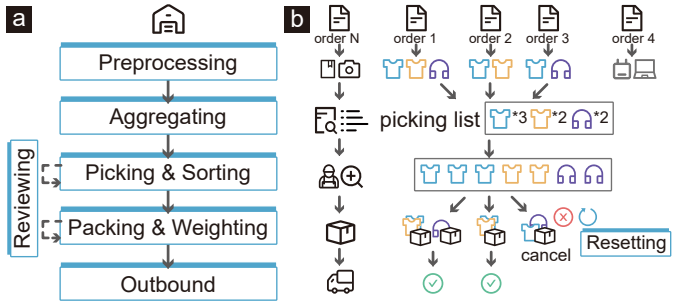

Fig. 1. (a) A typical e-commerce warehouse order processing pipeline. (b) An example for processing orders. Order1, 2, and 3 contain similar goods, which can be batched in one picking list. The picked goods are sorted into different orders and then packed as packages for delivery.

3.2 Data Abstraction

The data in order processing include temporal processing events and details of processed objects, which are logged and stored in a central computer system.

Temporal processing events. The whole order processing pipeline in the e-commerce warehouse is a kind of an assembly line [68], which can be regarded as a DAG (directed acyclic graph). Each order needs to pass through a series of procedures in the DAG to finish the processing. The start or end of a procedure is regarded as one event in order processing. Each event records the operator ID, through which we can access the operator information. The event sequence depicts the processing progress of the order. The start and end timestamp $\{t_i^s, t_i^e\}$ of each procedure P_i is recorded through the radio frequency identification (RFID) technique when operators use their portable RFID devices or the RFID scanners in work stations [34]. These data are synchronized to the cloud database. The temporal order processing event sequence E_i consists of timestamps $\{t_1^s, t_1^e, t_2^s, ..., t_n^e\}$. Through these data and the current timestamp t^c, we can calculate the **processing time** T_i^{pg}, **waiting time** T_i^{wg}, **processed time** T_i^{pd}, and **blocked time** T_i^{bd} of each procedure as follows:

$$T_i^{pg} = t^c - t_i^s, T_i^{wg} = t^c - t_i^e, T_i^{pd} = t_i^e - t_i^s, T_i^{bd} = t_{i+1}^s - t_i^e$$

The **processing time** and **waiting time** are used for real-time monitoring, while the **processed time** and **blocked time** are used for historical record analysis. Specifically, the order that is being processed has the **processing time** T_i^{pg}, which is the time since the order starts being processed. The order that has been processed but not transferred to the following procedure has the **waiting time** T_i^{wg}. The **waiting time** refers to the time an order waits for being processed in the following procedure. **Processed time** T_i^{pd} is the total time one order was processed. **Blocked time** T_i^{bd} is the total time one processed order spent on waiting for being transferred to the following procedure. Warehouse managers can compare these time data with a series of time interval thresholds for initial delay detection, thereby warning of the possible delay issues. The threshold series in each procedure is defined in multiple templates for different situations due to the unpredictable order volumes and tight delivery schedules.

Details of processed objects. As Fig. 1 shows, an order consists of several goods, and a picking list contains several orders. The information of goods is registered when the goods are put in the warehouse storage. Online consumers pay various orders with specific SKUs. Orders and goods' SKUs are sent to warehouses after orders have been made by consumers. The picking lists, including goods to be picked, orders in batches, and assigned operators, are created by managers based on several picking strategies. The details of goods, orders, and picking lists are stored in the database in tables with goods ID, order ID, and picking list ID as the primary keys.

3.3 Design Requirement

Following the nine-stage design study methodology proposed by Sedlmair et al. [48], we collaborated closely with two warehousing experts from a large e-commerce corporation. Both of them have worked in e-commerce warehousing for more than four years. Specifically, the collaboration started with characterizing domain problems for e-commerce warehouses in the *Precondition Phase*. We iteratively experienced the *Learn* and *Winnow* stages at this phase, in which we arranged multiple interviews with experts, conducted field inspections at collaborated warehouses, and surveyed related literature and commercial applications. Initial requirements were derived to monitor order processing status and solve delayed orders. Then, in the *Core Phase*, we held weekly meetings for four months with each other to *discover* specific requirements and tasks, and *design* prototypes to collect experts' feedback for improvements. The final requirements and visual design were derived in the iterative user-centric procedure including interviewing, observing, brainstorming, designing, and prototyping. We formulate design requirements in three aspects, namely, monitoring, analyzing, and evaluating delayed orders.

Monitoring real-time incoming orders:

- M1 **Facilitate the inspection of delayed orders.** Orders that have been processed or hung up for a long time should be preliminarily highlighted for users to efficiently locate the possibly abnormal procedure and delayed orders. Thus, the proposed system should be able to highlight the possible delayed orders. Furthermore, the highlighted orders need to be more evident for a progressive warning based on more cost duration.
- M2 **Scale with the unpredictable volumes of incoming orders.** The system should be capable of an efficient scalability to present the incoming data, due to the uncertain volumes of streaming data. Moreover, the presentations of real-time data change need smooth transitions to ensure readability.
- M3 **Present parallel operation status.** As a part of order processing data in an e-commerce warehouse, the status of parallel task operators is crucial for managers to know the operation efficiency and engagement. Thus, it is necessary to provide a design to present these parallel task data in the system.
- M4 **Support the visualization of hierarchical order processing data.** To support a comprehensive presentation of the order processing data content, such as the picking list, the system should consider the hierarchical data structures and enable the real-time presentations of these data, including individual orders and picking lists.

Analyzing order processing timelines:

- A1 **Visualize the historical processing records of orders.** By checking the processing timeline, users can judge whether an order is delayed from the comprehensive historical overview, including the time cost on processing and waiting, and the relationship of picking lists with the aggregated orders. Therefore, the system should support the visualization of these historical records for in-depth analysis.
- A2 **Provide related information about processing history.** Related information is necessary for analysis. For example, parallel task density can assist warehouse managers in accessing the processing load degree. Statistical information such as each procedures' time cost distribution can also make users aware of the normal and abnormal time cost. The system should also provide flexible interactions such as selecting or filtering specific orders for further analysis and priority evaluations.

Evaluating delayed orders.

- E1 **Present handling priority ranking.** Several factors decide the handling priorities. For example, orders close to the delivery deadline have a high priority to be handled. Thus, the system should integrate a ranking method considering the trade-off between all kinds of delay-handling factors. Moreover, users should be allowed to adjust the weight of these factors for a proper ranking result under practical real-time conditions.
- E2 **Show order details.** To facilitate the further exploration of selected orders, users need to know the details of the orders, such as the goods SKUs, goods attributes, and order ids. So the system should present the detail information on users' demand.
- E3 **Enable interactive identification of delayed orders.** The initial delay detection results according to the time interval thresholds are rough, within which, experts may find several incorrectly detected delayed orders or potential delayed orders based on visual exploration and their domain knowledge, Under this condition, the system should support smooth user interactions to identify and tag delayed orders.

4 E-commerce Warehouse Order Processing Visual Analytics System

4.1 System Overview

On the basis of the aforementioned design requirements, we propose OrderMonitor, a visual analytics system for monitoring e-commerce warehouse order processing. The architecture of the system is illustrated in Fig. 2. It contains three main modules, namely, data storage, data processing, and visualization.

The data storage module (Fig. 2 (a)) consists of several tables in the cloud dataset, including details of goods, orders, pickling lists, and operation events. The goods table records multiple attributes of goods, including fragility, weight, and volume. The order table contains the users' requested goods SKUs. The picking list table includes aggregated orders. The operation event table logs events of order processing in the warehouse, including start and end timestamps of each procedure and operator information. The data storage system will update data in all these tables synchronously when new orders come, thereby generating incoming data streams.

The data-processing module has three main parts: pipeline modeling, delay detection, and solution ranking (Fig. 2 (b)). First, we formulate the tabular data into objects indexed by order IDs. Within these objects, several keys and values depict the related picking list, processing events, and order details on the basis of the processing pipeline. Second, in accordance with temporal thresholds, delayed orders are initially detected and tagged in real time. However, in the shadow of various uncertain order requests, uniform but correct thresholds are difficult to be decided. Therefore, the preliminarily detected results may have faults, such as wrong identifications of delayed orders, which need further exploration by the visualization module. Third, the dealing priorities of delayed orders are ranked based on ranking rules, which can be set by users.

In the visualization module (Fig. 2 (c)), we deploy a visualization system with linked views to facilitate the real-time monitoring, history analysis, and priority evaluation of delayed orders. In the monitoring view (Fig. 2 (c1)), users can inspect the real-time order processing status in each procedure and access the initial result about delayed orders. Through the analyzing view (Fig. 2 (c2)), users can explore the historical processing records of these orders through flexible interactions

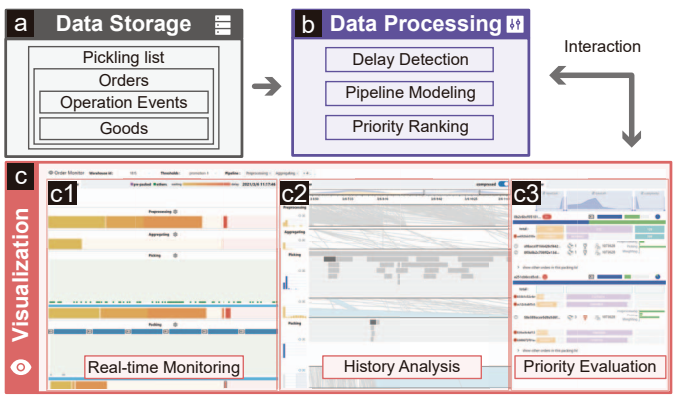

Fig. 2. The architecture of the system. OrderMonitor comprises three main modules, namely data storage (a), data processing (b), and visualization (c).

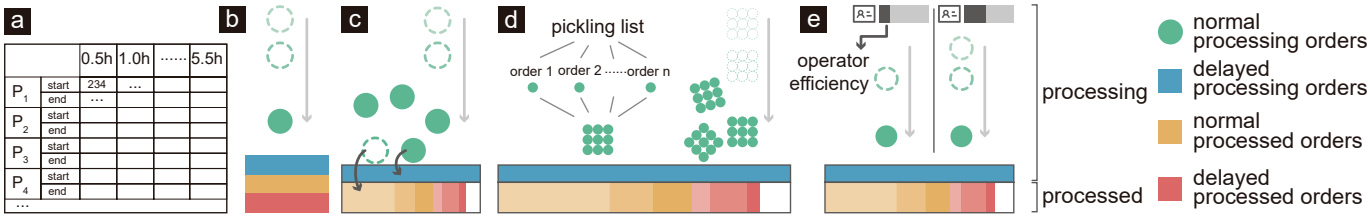

Fig. 3. (a) Our collaborated warehouse currently adopts a table-based interface for monitoring order processing. (b) An alternative of order sedimentation. (c) Our order sedimentation design. (d) In the *Picking* procedure, multiple orders are attached as a picking list. (e) In the *Packing* procedure, parallel operators are presented with an efficiency bar chart individually.

and further identify the delaying issues. In the evaluating view (Fig. 2 (c3)), users can obtain the ranking list on these delayed orders based on solving priorities and specific order details on demand. We introduce the design and implementation of these views in the following sections.

4.2 Monitoring View

The monitoring view (Fig. 2 (c1)) is motivated by the requirements for real-time delay detection (M1) and unpredictable data stream (M2). This view presents the processing status of incoming online orders updated in real time.

Visual Sedimentation metaphor, inspired by the physical process of sedimentation, is proposed to present the accumulation of streaming data [28]. Based on the sedimentation metaphor, we improve the initial tabular design in the existing system (Fig. 3 (a)). The tabular approach is not intuitive for users to distinguish the orders that are being processed from those that have been processed. It also lacks information, such as order types, about individual orders. *Drop chart* is one kind of visual sedimentation that allows tokens to escape through holes in containers' bottom. [28]. A sequence of drop charts can be used to visualize a series of continuous states, which is applicable to present the real-time status of order processing. Thus, we visualize the order-processing pipeline in multiple layers with processing objects accumulating and dropping, namely, order sedimentation, as displayed in Fig. 3 (c). Each layer comprises four types of orders:

- *Normal processing orders* are orders in processing. These orders enter from the previous procedures, falling with time going by.
- *Delayed processing orders* are orders in processing, but have a longer lasting time than the threshold.
- *Normal processed orders* are orders that have been processed in the current procedure and are waiting to be transferred into the next procedure. Their lasting time intervals in this procedure are in a safe range, under the delay threshold. These orders will be aggregated in the waiting sediment layer.
- *Delayed processed orders* are orders that have been processed but blocked in this procedure for a long time and are beyond the delay threshold. These orders will be aggregated in the delay sediment layer.

In our sedimentation design, the physical force is the time it takes for the order to be processed, making processed orders fall with time going by. Each procedure is visualized as a sedimentation layer. Different from the *drop chart*, each sedimentation layer consists of two parts, namely, the processing area and the processed area as depicted in Fig. 3 (c). The tokens in the processing area represent the processing orders or picking lists. The color of tokens encodes the types, such as ordinary or promoted. The tokens that have not been processed for a long time that is beyond the threshold will sediment in the bottom of processing area. The height of the sediment part encodes the number of orders that are still being processed. On the contrary, those tokens that have been processed will fade out through seamless transitions [54] and sediment in the processed area. In the processed area, normal processed orders are aggregated in the left sediment part, while delayed processed orders are collected in the right sediment part. In each sediment part, the orders are divided into several slots by the **processed time** T^{pd} of these orders. The slots are set based on integer multiples of the threshold, which increase from left to right. There is a boundary between these slots, indicating the border of normal and delayed processed orders. Within each slot, the darkness of color encodes the time cost and the width of the bar represents the number of orders in the slot (M1). An alternative sedimentation design is to arrange the normal and delayed processed orders from top to bottom, as shown in Fig. 3 (b). However, this design indicates that all orders ought to sediment through both the normal and delayed processed areas, which brings misunderstandings. It also requires more vertical screen spaces.

Considering that the order-processing data have a hierarchical structure (M3) and parallel tasks (M4), we propose a tailored design for these data features. Specially, for such procedures as *Picking & Sorting* and *Packing*, extra visual designs are integrated into the order sedimentation metaphor. In the *Picking & Sorting* procedure, the falling objects change from individual orders to picking lists because the processed objects have changed. Within this procedure, multiple tokens are attached to represent orders in one picking list (Fig. 3 (d)). The attached design illustrates the hierarchical structure of aggregated individual orders and the picking list item. As for the *Packing & Weighting* procedure, we present parallel tasks in the parallel drop charts (Fig. 3 (e)). Each drop chart represents one individual working operator or workstation, within which order items fall and accumulate independently. In the procedures of *Preprocessing*, *Aggregating*, and *Weighting*, the **processed time**s are nearly zero because operations in these procedures are executed by the computer system. Therefore, we only leave processed layers in these procedures. In the *Outbound* procedure, the processed orders are outbound, such that no waiting or delay issues occur, which makes weighting the last procedure in the monitoring view.

4.3 Analyzing View

The goal of the analyzing view (Fig. 2 (c2)) is to allow users to check the historical processing record of delayed orders detected by time thresholds (A1, A2). This kind of checking enables users to know the source of delayed orders to solve the delay problems efficiently.

To find the proper visualization for efficient analysis, we explored several design alternatives, including the extended Marey's graph and the Gantt chart, during the iterative design process, as shown in Fig. 4 (a-b). First, we follow the encoding of the extended Marey's graph in ViDX [68] by presenting individual orders as single lines (Fig. 4 (a)). However, different from fully automatic assembly lines, order processing pipelines involve human workers, whose uncertain working status leads to irregular temporal patterns in order-processing event data. Therefore, as shown in Fig. 4 (a), the time-aware outlier-preserving visual aggregation technique [68] is challenging to use to show the outliers in our problem. Moreover, the parallel tasks in Marey's graph are overlaid on the same graph, which leads to an unclear status of each operator or workstation. Marey's graph also lacks the capacity to present the hierarchical structures among picking lists and orders. The second alternative is a Gantt chart, as displayed in Fig. 4 (b). In the Gantt chart, each row represents one order and each procedure is encoded by colors [29]. In this way, the Gantt chart can present orders as individuals in a parallel form, thereby visualizing the meantime processing orders. Nonetheless, the Gantt chart lacks the information of each operator and workstation within the meantime processing.

Our final design, named order processing lifeline, is built on the combination of Marey's graph and a Gantt chart (Fig. 4 (d)), with the advantages of both. In each procedure, the horizontal axis encodes time, which increases from left to right. We employ a component consisting of a Gantt chart unit visualizing the **processed time** and a Marey's graph unit presenting the **blocked time** (A1). In the Gantt chart

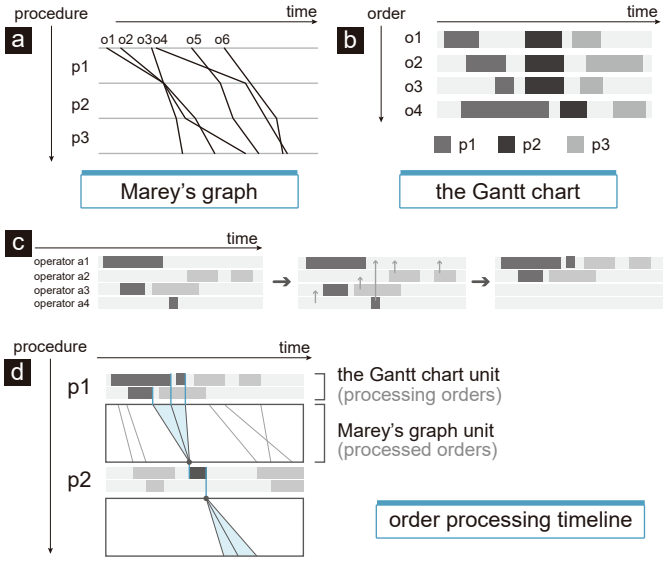

Fig. 4. (a-b) Two design alternatives for the visualization of orders' historical processing records: Marey's graph (a) and the Gantt chart (b). (c) Layout optimization for the overview of parallel tasks. (d) The order processing lifeline in the analyzing view. The Gantt chart unit visualizes the parallel **processed time**s, and the Marey's graph unit shows the processed orders' **blocked time**s. The triangle illustrates the orders and the related picking list.

unit, we make each row represent parallel operators or workstations, where individual processing tasks are shown in a temporal sequence. However, this way is unscalable when many operators exist. Also, we find real-world data are sparse among individual operators. Therefore, we provide an overview of the parallel task status at the same time by compressing the Gantt chart units in the dimension of individual operators (Fig. 4 (c)). The overview is obtained through a dynamic programming algorithm, which makes the rectangles in the Gantt chart arranged in as few rows as possible. Through the overview, users can access the parallel task density in a specific time.

In the Marey's graph unit, each line represents the processing objects in each procedure. To present the information of the hierarchical processing data structures, we improve the extended Marey's graph by aggregating the orders in one picking list into a triangle (Fig. 4 (d)). In the *Aggregating* procedure, the inverted triangle (Fig. 4 (d)) contains a group of aggregated orders, where the left edge represents the earliest order in the picking list while the right edge represents the latest one. The bottom angle of the triangle connects the picking list in the *Picking* procedure. In the *Packing & Weighting* procedure, the triangle indicates that a picking list is sorted in several individual orders. To avoid overlapping, we show the triangles when users interact with orders or picking lists. When users select on order, all the parts of the selected order in Gantt chart units and Marey's graph units in each procedure will be highlighted.

There are also some other visualizations in the Monitoring view to assist users in analysis. The area chart on the top of the analyzing view acts as the overall distribution of processed orders and processing orders. The histogram on the left of each procedure presents the distribution of **processed time** or **blocked time** in this procedure. Users can brush the histogram to filter orders with a specific range of **processed time** or **blocked time**. From these visualizations, users can access the overview of order processing lifelines.

4.4 Evaluating View

To help users decide which delayed orders should be handled first, we provide an evaluating view. In this view, users can set a series of weights on the factors that determine the dealing priority of delayed orders (E1) and access details about each order, including operator information and goods details (E2). Through the above information, users can identify whether orders should be handled (E3).

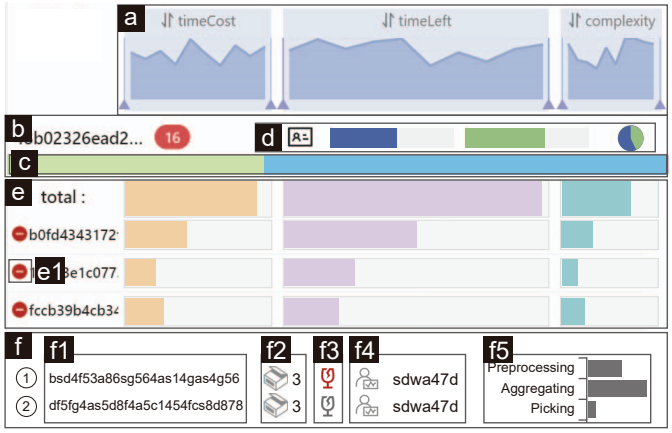

Fig. 5. The evaluating view. (a) An area chart indicates the factor value distribution of all the delayed orders. (b) We aggregate the orders in one pickling list, and present (c) goods category distribution and (d) operator efficiency. (e) Orders are ranked by ranking factors and weights. (f) The order detail panel shows details about the selected orders in the ranking component, including SKUs, quantity, fragility, retailer, and the time distribution of the processed procedures (f1-f5).

We discuss with domain experts on delay handling methods and formulate the following three ranking factors and their initial weights.

- *Time cost*: the time cost since an order is started being processed. The orders in warehouses should be processed on time to make space for newly incoming orders. Therefore, the more time one delayed order has cost, the higher priority the order is assigned.
- *Time left*: the time left from the outbound deadline. Owing to the diverse promised delivery services on different orders, which may be decided by order types, delivery distances or users' special membership service, the latest outbound time will be set for each order. The less time one delayed order remains, the higher dealing priority the order has.
- *Item complexity*: This factor indicates the type and quantity of included items in single orders, which affect the unit working efficiency (processed order numbers per man hour). Delayed orders that have high item complexity need much time to deal with. Thus, the less complexity one delayed order has, the higher priority the order is assigned.

Initial weights are averagely assigned to these factors, but users can change the weights in accordance with actual processing situations in consideration of practical processing power and the number of available operators. The ranking approach is computed with The priority p_i of the order i can be computed with

$$p_i = \sum_{j=1}^{m} w_j v_{ij} \tag{1}$$

where w_j is the weight assigned to the j-th factor and the sum of weights $\sum_j w_j = 1$. v_{ij} is the j-th factor value of the order i. m is the number of ranking factors. In OrderMonitor, m is 3.

There is also a factor named *Similarity*. This factor represents that among all delayed orders, how many orders are in similar conditions or share the same attributes, such as similar goods SKUs, operators, or blocked procedures. According to the experts, the managers tend to deal with orders that are in the same picking list first because these similar orders can be grouped and batched in one handling task assignment, thereby generating a highly efficient solution. We take this experience as the preconceived grouping rules in the evaluating view. Specifically, we first group the orders based on picking lists and then rank the groups by other similarity attributes. Next, we rank the orders within each group based on the priority that the other three factors determine.

Inspired by ValueCharts [16] and LineUp [23] that have been widely used in the visual ranking [36,56,60], we develop a ranking component (Fig. 5) in evaluating view for clear ranking and grouping of the delayed

orders. As shown in Fig. 5 (a), in each factor, an area chart exist at the top of each column, indicating the factor value distribution of all delayed orders. Users can adjust the weight of each factor by dragging interactions. After each weight adjustment, the new ranking result will be updated immediately. Moreover, we aggregate the orders in one pickling list, for batching similar orders expediently. Additional information, such as goods category distribution (Fig. 5 (c)) and operator efficiency (Fig. 5 (d)), is also provided for a comprehensive evaluation within each picking list. After a batch of delayed orders has been handled, these orders will be removed from the evaluating view and submitted to the next procedure in the monitoring view.

The evaluating view also contains detail panels that show detailed information about selected orders in the ranking component (**E2**). By discussing with the domain experts, we select some key attributes that they concern about, including the SKUs, quantity, fragility, and retailer of the goods (Fig. 5 (f1-f4)). For example, if an order contains fragile goods, the operators will do more packing operations than those for the orders with other goods. Moreover, within each order, a bar chart shows the time distribution of the processed procedures (Fig. 5 (f5)). From the details presented, managers can further judge the delaying issues on the selected orders. Moreover, this panel also supports users to untag some wrongly detected orders on demand (**E3**) through a button (Fig. 5 (e1)). The untagged orders will be removed from the ranking list.

4.5 Interaction

The system integrates a series of tailored interactions to support data exploration and analysis.

Overview-to-detail inspections of delayed orders. OrderMonitor comprises four linked views, supporting users in monitoring, analyzing, and evaluating delayed orders. The real-time order processing status are presented in the monitoring view. To facilitate users' inspection about the delayed orders, the system supports users to access the detailed information about processed orders through interactions. When users select one sediment part of the order sedimentation, the bar will be highlighted. The processing records of the orders in this part will be illustrated in the analyzing view. In the analyzing view, users can select several picking lists or orders and then access addition details in the evaluating view. Hence, the overview-to-detail inspection of delayed orders is supported through the coordinated views.

Setting delay detection rules. Dynamically adjusting the planned threshold for detecting delay issues on the basis of actual situations (such as processing capacity) can provide correct detection results. Therefore, OrderMonitor enables users to set the detection rules flexibly. Users can click the setting button (Fig. 7 (b2)) and set thresholds in the popover dialogue. In accordance with experts' domain knowledge, OrderMonitor provides several template rules for users (Fig. 7 (b1)). After the new detection rules are confirmed, the monitoring view will be refreshed and present the new order sedimentation in each procedure.

Scaling the processing timeline. The time intervals of different procedures are not equal, even relatively different. For example, orders in the *Preprocessing* procedure may wait for a long time but be processed for a relatively short time in the *Packing* procedure. Thus, OrderMonitor enables users to scale the timeline for an enhanced readability of the orders' historical records in the analyzing view.

Adjusting handling priority ranking weights. Owing to different judging criteria in diverse situations, the weights of ranking factors will be changed. For example, when near the end of a workday, warehouse managers tend to place the delayed orders that are near the delivery deadline in a high priority, while during other times, managers tend to handle orders with low *item complexity* first. In the evaluating view, users can adjust the weights by changing the width of factor columns.

5 Implementation

OrderMonitor have two separate parts, namely frontend and backend. We implement the frontend in JavaScript by using React.js [5], Matter.js [2], and D³ [10]. The backend is developed in Python, in conjunction with server libraries. Specifically, MongoDB [3] stores the temporal order processing event data and Flask [1] serves the database for the frontend. We deployed the system on a local Node [4] server. The CPU type is Intel(R) Core(TM) i9-9900K with 3.60GHz. The screen size is 3840*2160 pixels. In our iterative design and development progress, we use one real world datasets from one e-commerce warehouse. The dataset include more than 500,000 order processing events, and above 94,000 orders in the warehouse in one day.

6 Evaluation

In this section, inspired by the evaluation methodology [42] and practice of visual analytics studies [57,58,65], we demonstrate the effectiveness of OrderMonitor with two case studies and expert interviews.

6.1 Case Studies

We conducted two case studies with two domain experts EA and EB, who were specialized in e-commerce warehouse order processing, to evaluate the usability of OrderMonitor in real-time monitoring, analyzing, and evaluating the processed orders. EA is a senior engineer who has worked on developing commercial warehouse management products for six years. For better understanding users' requirements, EA has surveyed e-commerce warehouses for a long time and knows well about warehouse management. EB is an expert with rich experience in e-commerce warehouse management. Both experts were familiar with the visual design, interactions, and workflow through our demonstration and introduction. We first presented how OrderMonitor worked and explained the visual encoding in each view. After that, we asked them to use the system.

6.1.1 Inspecting the processing timelines

This case study aims to demonstrate the effectiveness of OrderMonitor in presenting the real-time streaming online order data and assisting domain experts in inspecting the historical order processing records.

EA first selected warehouse ID, pipeline type, and processing threshold template in the header menu of the system (Fig. 6 (a)). Then, the real-time processing status is showed in the monitoring view (Fig. 6 (b)). Most of the picking list consisted of few orders because it was early morning, and orders were not created rapidly at this time. After a while, he found there existed thirteen orders delayed in the *Picking* procedure. To inspect the specific issues of the delayed orders, he clicked the red bar and accessed the thirteen orders' processing timelines.

In the analyzing view, EA noticed that the thirteen orders seemed attached to one picking list (Fig. 6 (d)). He scaled the timeline and found the individual lines of these thirteen orders in the *Preprocessing* procedure gathered into one line in the *Aggregating* procedure, which confirmed his speculation. Moreover, he figured out that the earliest timestamp of this group of orders was earlier than the start working time of warehouse operators (Fig. 6 (c)). So these orders were assigned to the operators even with a small volume. According to EA's experience in management, the early processing and small volume might result in operators' ignorance of the batch of these orders and further delayed them after picking. So, after knowing the processing timelines, he

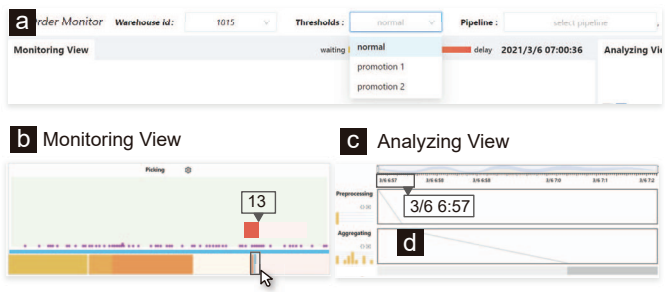

Fig. 6. (a) The header menu supports users choosing the monitored warehouse id, pipeline type, and processing threshold templates. (b) Thirteen orders are initially detected as delayed according to the threshold in the monitoring view. (c) The timestamp indicates that most of the orders' created times are not in the working time. (d) The historical processing timelines of the selected thirteen delayed orders are attached to one picking list.

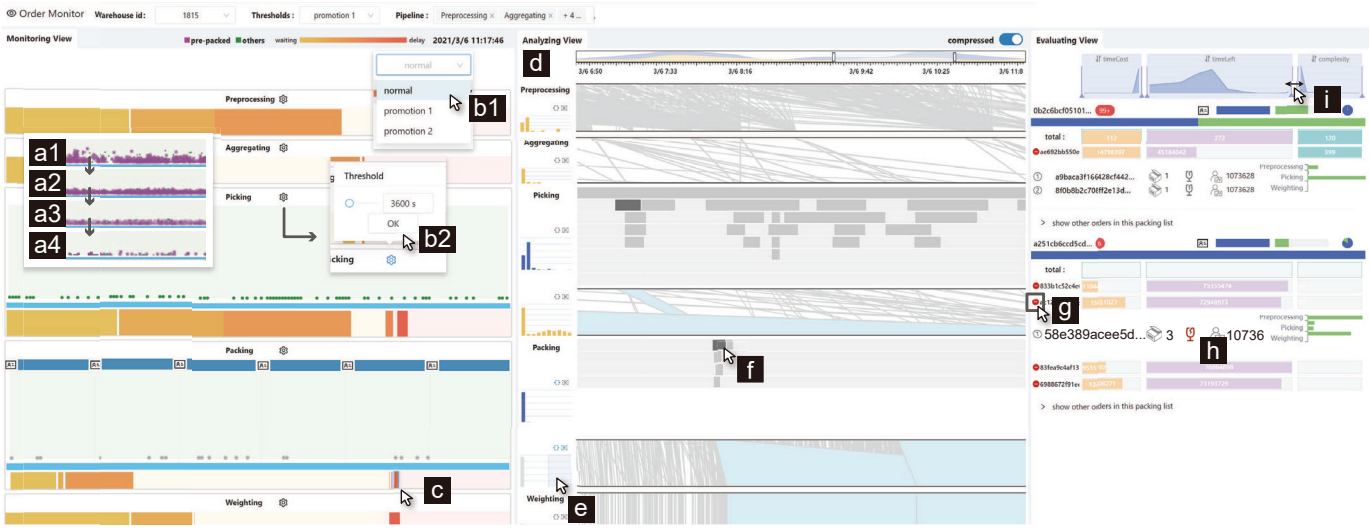

Fig. 7. (a1-a4) A large volume of orders that included on-sale goods entered into the *Picking* procedure. With dropping, the pre-packed orders disappeared simultaneously. EB could set delay detection rules by templates (b1) or individual procedures (b2). In the monitoring view, EB selected delayed orders (c) and accessed the historical processing pipelines of these orders in the analyzing view (d). He brushed the bars in the histogram of **processed time** distribution (e) and then selected some orders (f) in the Gantt chart unit. Then, he checked order details (h) in the evaluating view and canceled the delay label (g). At last, he adjusted the ranking weights (i).

reported these orders to one operator and asked him to check them. The finding in the monitoring view and the analyzing view help EA notice the delayed orders in real time and take action promptly.

6.1.2 Handling delayed orders

This case study aims to demonstrate the effectiveness of OrderMonitor in monitoring real-time order processing, analyzing delayed orders, and evaluating the handling priority in a promotion.

For the sake of more efficient order processing, before the promotion, the retailer will notify the warehouse in advance to pre-pack several packages, which usually contain one unit of promoted goods, for example, one t-shirt or a set of mouthwash. Orders that only include one unit of promoted goods can be quickly processed and outbound, while orders containing the on-sale goods and other goods will be picked and packed as regular orders. At the beginning of the promotion of goods M, a large volume of orders that included goods M was created by online consumers and sent to the warehouse. Most of them were quickly aggregated and passed to the *Picking* procedure. So, at that time (11:05), many orders showed and dropped in the picking panel (Fig. 7 (a1-a4)), where the purple circles represented the pre-packed orders and the green ones were picking lists that contained both promoted goods and other goods. After a short time, the pre-packed orders disappeared simultaneously, indicating that they have been finished processing in the *Picking* procedure. Through the picking panel in the monitoring view, EB noticed that there still existed many light green orders and other kinds of orders, which meant the promotion bring an increase in the processing workload even most of the orders are processed by pre-packing. Taking into account the above finding, he adjusted the delay detection threshold template to a promotion mode. Specifically, he changed the *Picking* and *Packing* procedure threshold to a larger one because the increasing workload should allow more processing time for the limited number of operators. After the adjustment, the bar that presented the delayed order numbers in the *Picking* procedure was updated to a lower value. This demonstrated that OrderMonitor could support users to adjust the detection rule of delayed order based on actual situations.

EB continued checking the processing status in the monitoring view. After a while, several delayed orders appeared in the *Packing* procedure. EB clicked the bar chart that represented delay numbers in the monitoring view (Fig. 7 (c)) for accessing the historical processing pipelines of these orders in the analyzing view (Fig. 7 (d)). He observed the parallel task density through the Gantt chart unit. To further inspect those orders with long packing time, he brushed the bars in the histogram of **processed time** distribution on the side of the *Packing* procedure (Fig. 7 (e)). Then, he clicked some of the filtered Gantt chart units for further handling them in the follow-up evaluations (Fig. 7 (f)). In the evaluating view, he obtained several key details of the selected orders. He checked the order detail panel and noticed that there are multiple and fragile goods in one of the selected orders (Fig. 7 (h)). Given that multiple and fragile goods would need more time to pack, since it had not been much longer than the setting threshold (three times longer than the threshold), EB understood that it should be regarded as a special case which should not be marked as delayed. Therefore, he canceled the delayed labels of orders that had fragile goods (Fig. 7 (g)).

Since it was near noon, it was necessary to process the earlier morning orders as soon as possible to avoid more uncertain factors in case more orders needed to be processed in the afternoon. In addition, the orders in evaluating view no longer contained fragile ones that need special handling. So, EB turned up the *time left* weight and turned down the *item complexity* weight by flexible dragging interactions (Fig. 7 (i)). After the weight adjustment, the system provided him with a ranking list about the handling priority of delayed orders, where the orders were grouped by picking lists. At last, EB followed the ranking list and traced to the source of the delayed orders according to the corresponding operator IDs. So far, the delayed orders could be handled in a proper and efficient sequence.

6.2 Expert Interview

We conducted expert interviews with three domain experts who are specialized in e-commerce warehouse order processing. Two of them are experts in case studies (EA and EB), while the third expert (EC) is a designer who has been designing warehouse management products for 2 years.

Procedure. The expert interview was conducted individually with the experts. We first introduced the visual designs and data structure in each view in OrderMonitor. After the experts were familiar with the system, we connected OrderMonitor with the real-word dataset and presented the interactions and workflow of the system. Then, experts were invited to freely explore the system and give feedback about the overall usability of the system. Finally, we asked several questions about the visual design, readability, system functions, and interface design. The feedback of expert interviews is as follows.

System effectiveness. All the experts responded with positive feedback on our system and think highly of the sedimentation design and the processing pipeline chart. EA said, “Neither of too low nor too high parallel density is expected. I can be aware of the parallel task density

through the pipeline chart. It is useful for a historical process review." EB also confirmed the process pipeline chart's usefulness in inspecting the relationship between picking lists and orders. Besides, he acknowledged the order sedimentation in the monitoring view was intuitive, "...especially when the sheer volume of orders appeared in promotion scenarios." EC mentioned that grouping orders by their similarities was of great assistance. She also pointed that the system interaction workflow is "layered" and helpful for solving problems effectively. EA and EC expressed positive attitudes on the further deployment of the current system functions for different terminal devices, such as portable devices and small screens in workstations.

Usability. Experts generally appreciated that the coordinated views in the system could help monitor, analyze, and evaluate delayed orders. EB said, "I like the linked views, from which I can locate the delayed orders step by step and solve them." Experts also agreed that the learning curve of the system was acceptable and they could easily learn it after a quick demonstration. EA commented that the system's workflow conforms to her previous experience in managing order processing.

Improvement. While the experts acknowledged the system had met domain analytic requirements, they also provided several valuable suggestions on improving the performance and usability of the proposed system. First, EB figured out that, "The system should support the filter on specific procedures because sometimes I might just want to inspect the *Picking* and *Packing* procedure. Only presenting these two procedures can make me more concentrated." EC agreed with EB's opinion and advised to align each procedure in analyzing view with the procedures in the monitoring view, which can make these two views more harmonious visually. We have improved these functions in our system accordingly. Moreover, EA recommended using a machine learning approach to detect delayed orders based on the current visualization system, which is a promising future work [61] but out of the scope of this paper. We further discuss this in Sect. 8.

7 Discussion

In this section, we first present lessons learned from our design study and then discuss the generalization, significance, and limitations.

Lessons Learned. The close collaboration with domain experts provides valuable lessons in developing visualization systems to manage order processing. First, designing an overview for order processing event data need to consider the temporal sparsity of patterns. The sparsity generates due to the long time intervals of task stages. This feature requires the design to efficiently utilize the screen space to show valid data. As such, we designed the compressed Gantt chart unit to present the density of parallel tasks, which allows users to easily obtain the overview and locate patterns of interest [7]. Second, warehouse visual analytics needs to consider the uncertainty introduced by human factors. In our study, the order processing pipeline is not fully automatic, which involves the collaboration of both workers and machines. For example, workers may spend extra effort on special cases, like taking longer time to pack fragile items. This can bring uncertainty for detection and cause "false-delayed" orders. Thus, we provide extra information such as fragility for users to make a comprehensive judgment.

Generalization. OrderMonitor is proposed to monitor, analyze, and manipulate the order processing in e-commerce warehouses. We mainly evaluated the system on one typical processing pipeline in an e-commerce warehouse, which is not unique using scenarios of the proposed approach in our visualization system. Specifically, the two novel visual designs, namely, order sedimentation and processing timeline based on the Gantt chart and Marey's graph, can be applied in other process visualizations. First, order sedimentation that adopts individual aggregations is applicable in streaming data that have a hierarchical data structure. For example, a product is comprised of multiple industrial components in the factory, where manufacturing progress generates a sheer volume of streaming data with a hierarchical data structure. Second, the combination of the Gantt charts and Marey's graphs can be adopted in those processes with parallel tasks, such as task assignment in teamwork or project management, and parallel progress in manufacturing. The Gantt chart unit allows users to access the status of parallel tasks, while Marey's graph unit provides an overview of individual time cost and aggregation patterns through presenting. The combinations of them allow users to inspect both information in the meantime.

Significance. To the best of our knowledge, we are the first to apply the visual analytics technique on e-commerce warehouse order processing, which involves a new kind of streaming data, namely e-commerce orders. Except for the common feature of streaming data like sheer volumes and unpredictable updating speeds, this kind of data has hierarchical data structures and various attributes. Techniques of the internet of things (IoT), such as sensor [39, 59] and RFIDs [34, 35], provide the opportunity to access the streaming progress data, which can be processed and visualized for analysis. Through our visualization approach, warehouse managers can make good use of these data to monitor, analyze and manipulate order processing in e-commerce warehouses and further improve the processing efficiency. Moreover, due to the uncertain workload, high cost of automatic facilities, and the flexibility of human operators [11, 12], most e-commerce warehouses still need human-in-the-loop operations and management. Under this condition, our system bridge the gap between the domain requirements of warehouse managers and practical order processing data, which helps users observe the processing status and control abnormal situations as much as possible.

Limitation. The case studies and expert interviews demonstrate the effectiveness of OrderMonitor. Nevertheless, there are still several limitations of our work. The first limitation exists in the detection method of delayed issues. We apply the threshold-based methods that are widely used in the current warehouse management system and deal with the drawback of rough detection by providing visual analytics approaches. A more intelligent method is to automatically detect the delayed issues with considerations about order details and attributes, such as fragility. However, the automatic methods need ground truth on delayed orders and their attributes, which is still not resolved for now. The second limitation is the lack of coordination with other multidimensional spatio-temporal data [37, 64] in the warehouse, like the shelf layouts and inventory changes, to explore patterns about order processing. Moreover, we have not considered the effect of event correlations in different processing procedures on delayed orders, which needs further modeling and research.

8 Conclusion

In this work, we adopt a visualization approach for monitoring order processing in e-commerce warehouses and propose OrderMonitor, a visualization system that enables warehouse managers to monitor and manipulate order processing in real time based on streaming e-commerce order data. The proposed system integrates a novel order sedimentation design and a newly proposed process timeline with three coordinated views to facilitate the monitoring, analysis, and evaluation of delayed orders in order processing. The case studies and expert interviews demonstrate that the patterns and insights revealed in OrderMonitor can guide warehouse managers in efficiently manipulating order processing in a timely manner.

In future work, we will continue collaborating with domain experts on order processing in e-commerce warehouses. The future work focuses on two major avenues. First, we plan to consider more automatic methods to detect delayed order processing in e-commerce warehouses. The currently proposed visualization system's working logs, especially user identification on delayed orders by labeling interactions in evaluating view, provide the ground truth and data for the machine learning training on delay detection. Second, various sensors in the warehouse record the sheer volume of environmental conditions and events. We plan to explore these data to further analyze order processing performance, such as the effects of process event correlation, and build a more comprehensive warehouse management system.

Acknowledgments

This work was supported by NSFC (62072400) and Zhejiang Provincial Natural Science Foundation (LR18F020001). This work was also supported by Alibaba-Zhejiang University Joint Institute of Frontier Technologies and the Collaborative Innovation Center of Artificial Intelligence by MOE and Zhejiang Provincial Government (ZJU).

REFERENCES

[1] Flask. Website. Retrieved June 28th, 2021 from https://flask.palletsprojects.com/en/2.0.x/.

[2] Matter.js. Website. Retrieved June 28th, 2021 from https://brm.io/matter-js/.

[3] MongoDB. Website. Retrieved June 28th, 2021 from https://www.mongodb.com/.

[4] Node.js. Website. Retrieved June 28th, 2021 from https://nodejs.org/en/.

[5] React.js. Website. Retrieved June 28th, 2021 from https://reactjs.org/.

[6] W. Aigner, S. Miksch, B. Thurnher, and S. Biffl. PlanningLines: novel glyphs for representing temporal uncertainties and their evaluation. In *Proceedings of International Conference on Information Visualisation*, pp. 457–463, 2005.

[7] N. Andrienko, G. Andrienko, S. Miksch, H. Schumann, and S. Wrobel. A Theoretical Model for Pattern Discovery in Visual Analytics. *Visual Informatics*, 5(1):23–42, 2021.

[8] Z. Bai, Y. Tao, and H. Lin. Time-varying Volume Visualization: A Survey. *Journal of Visualization*, 23:745–761, 2021.

[9] J. J. Bartholdi and S. T. Hackman. *Warehouse & Distribution Science: Release 0.89*. Supply Chain and Logistics Institute Atlanta, 2008.

[10] M. Bostock, V. Ogievetsky, and J. Heer. D³ Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.

[11] N. Boysen, R. de Koster, and F. Weidinger. Warehousing in the E-commerce Era: A Survey. *European Journal of Operational Research*, 277:396–411, 2019.

[12] N. Boysen, K. Stephan, and F. Weidinger. Manual Order Consolidation with Put Walls: the Batched Order Bin Sequencing Problem. *EURO Journal on Transportation and Logistics*, 8:169–193, 2019.

[13] T. Bódis and J. Botzheim. Bacterial Memetic Algorithms for Order Picking Routing Problem with Loading Constraints. *Expert Systems with Applications*, 105:196–220, 2018.

[14] N. Cao, C. Lin, Q. Zhu, Y. Lin, X. Teng, and X. Wen. Voila: Visual Anomaly Detection and Monitoring with Streaming Spatiotemporal Data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):23–33, 2018.

[15] N. Cao, Y. Lin, X. Sun, D. Lazer, S. Liu, and H. Qu. Whisper: Tracing the Spatiotemporal Process of Information Diffusion in Real Time. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2649–2658, 2012.

[16] G. Carenini and J. Loyd. ValueCharts: Analyzing Linear Models Expressing Preferences and Evaluations. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, p. 150–157, 2004.

[17] F. Chen, H. Wang, C. Qi, and Y. Xie. An Ant Colony Optimization Routing Algorithm for Two Order Pickers with Congestion Consideration. *Computers & Industrial Engineering*, 66(1):77–85, 2013.

[18] A. Dasgupta, D. L. Arendt, L. R. Franklin, P. C. Wong, and K. A. Cook. Human Factors in Streaming Data Analysis: Challenges and Opportunities for Information Visualization. *Computer Graphics Forum*, 37(1):254–272, 2018.

[19] T. Economist. The Age of Amazon and Alibaba is just Beginning. Website, October 2017. Retrieved March 31st, 2021 from https://www.economist.com/leaders/2017/10/26/the-age-of-amazon-and-alibaba-is-just-beginning.

[20] R. F. Erbacher. Visualization Design for Immediate High-Level Situational Assessment. In *Proceedings of International Symposium on Visualization for Cyber Security*, p. 17–24, 2012.

[21] F. Fischer and D. A. Keim. NStreamAware: Real-Time Visual Analytics for Data Streams to Enhance Situational Awareness. In *Proceedings of Workshop on Visualization for Cyber Security*, p. 65–72, 2014.

[22] T. Fujiwara, J.-K. Chou, S. Shilpika, P. Xu, L. Ren, and K.-L. Ma. An Incremental Dimensionality Reduction Method for Visualizing Streaming Multidimensional Data. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):418–428, 2020.

[23] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit. LineUp: Visual Analysis of Multi-Attribute Rankings. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2277–2286, 2013.

[24] J. Gu, M. Goetschalckx, and L. F. McGinnis. Research on Warehouse Operation: A Comprehensive Review. *European Journal of Operational Research*, 177(1):1–21, 2007.

[25] J. Gu, M. Goetschalckx, and L. F. McGinnis. Research on Warehouse Design and Performance Evaluation: A Comprehensive Review. *European Journal of Operational Research*, 203(3):539–549, 2010.

[26] Y. Guo, S. Guo, Z. Jin, S. Kaul, D. Gotz, and N. Cao. Survey on Visual Analysis of Event Sequence Data. *IEEE Transactions on Visualization and Computer Graphics*, 2021.

[27] S. Henn. Algorithms for On-line Order Batching in an Order Picking Warehouse. *Computers & Operations Research*, 39(11):2549–2563, 2012.

[28] S. Huron, R. Vuillemot, and J. Fekete. Visual Sedimentation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2446–2455, 2013.

[29] J. Jo, J. Huh, J. Park, B. Kim, and J. Seo. LiveGantt: Interactively Visualizing a Large Manufacturing Schedule. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2329–2338, 2014.

[30] M. Krstajić, E. Bertini, and D. Keim. CloudLines: Compact Display of Event Episodes in Multiple Time-Series. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2432–2439, 2011.

[31] K. Kucher, R. M. Martins, C. Paradis, and A. Kerren. StanceVis Prime: Visual Analysis of Sentiment and Stance in Social Media Texts. *Journal of Visualization*, 23(6):1015–1034, 2020.

[32] B. C. Kwon, V. Anand, K. A. Severson, S. Ghosh, Z. Sun, B. I. Frohnert, M. Lundgren, and K. Ng. DPVis: Visual Analytics With Hidden Markov Models for Disease Progression Pathways. *IEEE Transactions on Visualization and Computer Graphics*, 27(9):3685–3700, 2021.

[33] K. Leung, K. Choy, P. K. Siu, G. Ho, H. Lam, and C. K. Lee. A B2C E-commerce Intelligent System for Re-engineering the E-order Fulfilment Process. *Expert Systems with Applications*, 91:386–401, 2018.

[34] M. K. Lim, W. Bahr, and S. C. Leung. RFID in the Warehouse: A Literature Analysis (1995–2010) of Its Applications, Benefits, Challenges and Future Trends. *International Journal of Production Economics*, 145(1):409–430, 2013.

[35] C. D. G. Linhares, J. R. Ponciano, J. G. S. Paiva, B. A. N. Travençolo, and L. E. C. Rocha. A Comparative Analysis for Visualizing the Temporal Evolution of Contact Networks: A User Study. *Journal of Visualization*, 2021.

[36] D. Liu, D. Weng, Y. Li, J. Bao, Y. Zheng, H. Qu, and Y. Wu. SmartAdP: Visual Analytics of Large-scale Taxi Trajectories for Selecting Billboard Locations. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):1–10, 2017.

[37] D. Liu, P. Xu, and L. Ren. TPFlow: Progressive Partition and Multidimensional Pattern Extraction for Large-Scale Spatio-Temporal Data Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):1–11, 2019.

[38] S. Liu, J. Yin, X. Wang, W. Cui, K. Cao, and J. Pei. Online Visual Analytics of Text Streams. *IEEE Transactions on Visualization and Computer Graphics*, 22(11):2451–2466, 2016.

[39] H. Mansoor, W. Gerych, A. Alajaji, L. Buquicchio, K. Chandrasekaran, E. Agu, and E. Rundensteiner. ARGUS: Interactive Visual Analysis of Disruptions in Smartphone-detected Bio-Behavioral Rhythms. *Visual Informatics*, 2021.

[40] K. Matković, H. Hauser, R. Sainitzer, and M. E. Gröller. Process Visualization with Levels of Detail. In *Proceedings of IEEE Symposium on Information Visualization*, pp. 67–70, 2002.

[41] M. Matusiak, R. de Koster, L. Kroon, and J. Saarinen. A Fast Simulated Annealing Method for Batching Precedence-constrained Customer Orders in a Warehouse. *European Journal of Operational Research*, 236(3):968–977, 2014.

[42] T. Munzner. A Nested Model for Visualization Design and Validation. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):921–928, 2009.

[43] A. of Amazon. Amazon Fulfillment Center Video Tour. Website, August 2020. Retrieved June 28th, 2021 from https://www.youtube.com/watch?v=e3QgE4Vs5Cs.

[44] C. Palomo, Z. Guo, C. T. Silva, and J. Freire. Visually Exploring Transportation Schedules. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):170–179, 2016.

[45] G. Pedrielli, A. Vinsensius, E. P. Chew, L. H. Lee, A. Duri, and Haobin Li. Hybrid Order Picking Strategies for Fashion E-commerce Warehouse Systems. In *Proceedings of Winter Simulation Conference*, pp. 2250–2261, 2016.

[46] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman. LifeLines: Using Visualization to Enhance Navigation and Analysis of Patient Records. In *The Craft of Information Visualization*, pp. 308–312.

2003.

[47] H. D. Ratliff and A. S. Rosenthal. Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem. *Operations Research*, 31(3):507–521, 1983.

[48] M. Sedlmair, M. Meyer, and T. Munzner. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, 2012.

[49] A. Staff. Cainiao Gears Up for 11.11 Shopping Festival with New Logistics Tech. Website, October 2018. Retrieved June 28th, 2021 from `https://www.alibabagroup.com/en/news/article?news=p181030b`.

[50] M. Steiger, J. Bernard, S. Mittelstädt, H. Lücke-Tieke, D. Keim, T. May, and J. Kohlhammer. Visual Analysis of Time-Series Similarities for Anomaly Detection in Sensor Networks. *Computer Graphics Forum*, 33(3):401–410, 2014.

[51] D. Sun, R. Huang, Y. Chen, Y. Wang, J. Zeng, M. Yuan, T. C. Pong, and H. Qu. PlanningVis: A Visual Analytics Approach to Production Planning in Smart Factories. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):579–589, 2020.

[52] Y. Tanahashi, C. Hsueh, and K. Ma. An Efficient Framework for Generating Storyline Visualizations from Streaming Data. *IEEE Transactions on Visualization and Computer Graphics*, 21(6):730–742, 2015.

[53] T. Tang, Y. Wu, L. Yu, Y. Li, and Y. Wu. VideoModerator: A Risk-aware Framework for Multimodal Video Moderation in E-Commerce. *To appear in IEEE Transactions on Visualization and Computer Graphics*, 2022.

[54] C. Tominski, G. Andrienko, N. Andrienko, S. Bleisch, S. I. Fabrikant, E. Mayr, S. Miksch, M. Pohl, and A. Skupin. Toward Flexible Visual Analytics Augmented through Smooth Display Transitions. *Visual Informatics*, 5(3):28–38, 2021.

[55] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, 1986.

[56] E. Wall, S. Das, R. Chawla, B. Kalidindi, E. T. Brown, and A. Endert. Podium: Ranking Data Using Mixed-Initiative Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):288–297, 2018.

[57] J. Wang, J. Wu, A. Cao, Z. Zhou, H. Zhang, and Y. Wu. Tac-Miner: Visual Tactic Mining for Multiple Table Tennis Matches. *IEEE Transactions on Visualization and Computer Graphics*, 27(6):2770–2782, 2021.

[58] Y. Wang, H. Liang, X. Shu, J. Wang, K. Xu, Z. Deng, C. D. Campbell, B. Chen, Y. Wu, and H. Qu. Interactive Visual Exploration of Longitudinal Historical Career Mobility Data. *IEEE Transactions on Visualization and Computer Graphics*, 2021.

[59] D. Wei, C. Li, H. Shao, Z. Tan, Z. Lin, X. Dong, and X. Yuan. SensorAware: Visual Analysis of Both Static and Mobile Sensor Information. *Journal of Visualization*, 24(3):597–613, 2021.

[60] D. Weng, C. Zheng, Z. Deng, M. Ma, J. Bao, Y. Zheng, M. Xu, and Y. Wu. Towards Better Bus Networks: A Visual Analytics Approach. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):817–827, 2021.

[61] A. Wu, Y. Wang, X. Shu, D. Moritz, W. Cui, H. Zhang, D. Zhang, and H. Qu. AI4VIS: Survey on Artificial Intelligence Approaches for Data Visualization. *arXiv preprint arXiv:2102.01330*, 2021.

[62] W. Wu, Y. Zheng, K. Chen, X. Wang, and N. Cao. A Visual Analytics Approach for Equipment Condition Monitoring in Smart Factories of Process Industry. In *Proceedings of IEEE Pacific Visualization Symposium*, pp. 140–149, 2018.

[63] Y. Wu, Z. Chen, G. Sun, X. Xie, N. Cao, S. Liu, and W. Cui. StreamExplorer: A Multi-Stage System for Visually Exploring Events in Social Streams. *IEEE Transactions on Visualization and Computer Graphics*, 24(10):2758–2772, 2018.

[64] Y. Wu, D. Weng, Z. Deng, J. Bao, M. Xu, Z. Wang, Y. Zheng, Z. Ding, and W. Chen. Towards Better Detection and Analysis of Massive Spatiotemporal Co-Occurrence Patterns. *IEEE Transactions on Intelligent Transportation Systems*, 22(6):3387–3402, 2021.

[65] M. Xia, R. P. Velumani, Y. Wang, H. Qu, and X. Ma. QLens: Visual Analytics of MUlti-step Problem-solving Behaviors for Improving Question Design. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):870–880, 2021.

[66] X. Xie, J. Wang, H. Liang, D. Deng, S. Cheng, H. Zhang, W. Chen, and Y. Wu. PassVizor: Toward Better Understanding of the Dynamics of Soccer Passes. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1322–1331, 2021.

[67] K. Xu, Y. Wang, L. Yang, Y. Wang, B. Qiao, S. Qin, Y. Xu, H. Zhang, and H. Qu. CloudDet: Interactive Visual Analysis of Anomalous Performances in Cloud Computing Systems. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1107–1117, 2020.

[68] P. Xu, H. Mei, L. Ren, and W. Chen. ViDX: Visual Diagnostics of Assembly Line Performance in Smart Factories. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):291–300, 2017.

[69] S. Ye, Z. Chen, X. Chu, Y. Wang, S. Fu, L. Shen, K. Zhou, and Y. Wu. ShuttleSpace: Exploring and Analyzing Movement Trajectory in Immersive Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):860–869, 2021.

[70] T. Zhang, Z. Chen, Z. Zhao, X. Luo, W. Zheng, and W. Chen. FaultTracer: Interactive Visual Exploration of Fault Propagation Patterns in Power Grid Simulation Data. *Journal of Visualization*, 2021.

[71] Z. Zhao and P. Yang. Improving Order-picking Performance by Optimizing Order Batching in Multiple-cross-aisle Warehouse Systems: A Case Study from E-commerce in China. In *Proceedings ofInternational Conference on Industrial Engineering and Applications*, pp. 158–162, 2017.

[72] F. Zhou, X. Lin, X. Luo, Y. Zhao, Y. Chen, N. Chen, and W. Gui. Visually Enhanced Situation Awareness for Complex Manufacturing Facility Monitoring in Smart Factories. *Journal of Visual Languages and Computing*, 44:58–69, 2018.