BlueCollar: Optimizing Worker Paths on Factory Shop Floors with Visual Analytics

Dominik Herr[1,2], Sebastian Grund[1], Thomas Ertl[1]
[1]Institute for Visualization and Interactive Systems
[2]Graduate School of Excellence advanced Manufacturing Engineering
University of Stuttgart
{firstname.lastname}@vis.uni-stuttgart.de

Abstract

The optimization of a factory's productivity regarding quality and efficiency is an important task in the manufacturing domain. To optimize the productivity, production lines are optimized to have short transportation paths and short processing times at the stations that process intermediate components or the final product. A factory's layout is a key factor in this optimization aspect. This optimization mostly comprises the machine tools' positions with respect to places where supply goods are being delivered and other tools are stationed, often neglecting the paths that workers need to take at the shop floor. This impairs a factory's productivity, as machines may need to wait for workers, who operated another machine and are still on the way due to the long distance between the machines. In this work, we present BlueCollar, a visual analytics approach that supports layout planners to explore and optimize existing factory layouts regarding the paths taken by workers. Planners can visually inspect the paths that workers need to take based on their work schedule and the factory's layout. An estimation of distribution algorithm supports them in choosing which layout elements, e.g., shared tool caches, to relocate. Its intermediate and final results are used to provide visual cues for suitable relocation areas, and to suggest new layouts automatically. We demonstrate our approach through an application scenario based on a realistic prototype layout provided by an external company.

1. Introduction

A factory's efficiency can be optimized in several aspects, e.g., the optimization of processes to improve the products' quality and reduce product rejection rate, as well improving a factory's layout regarding the positions of its processing stations. One important aspect for a layout's effectiveness is the productivity of the workers that operate and maintain machine tools. The productivity depends on many aspects, such as ergonomics at a workstation [1] and the work schedule they need to complete [2]. Additionally, the planning of the paths taken by the workers to complete their work schedule is important, e.g., in case they need to operate multiple machines or they need to share special equipment at a tool container [3].

Often, pathing problems for workers are hard to account for during the layout planning phase. Exemplary reasons are changes in a production line over time (e.g., machinery replacement), the addition of new production lines, or changing work schedules. There are numerous approaches to support the workers' pathing during production, for example, visual cues the on floor or presenting the best path on the nearest machine terminal. It is also possible to optimize the positions of movable parts, such as shared tool caches. However, this optimization is challenging [4]. On the one hand, an automatic optimization is expensive to calculate due to the large number of possible solutions. Further, its results are prone to errors due to unmodeled constraints (e.g., availability of adequate power supply) or implicit constraints known by experts, e.g., workers avoid being close to loud machinery. On the other hand, manual optimization by experts is challenging, as the path optimization is dependent on various parameters, such as the produced goods.

This paper contributes a visual analytics approach to support layout planning experts to optimize the layout of planned or existing production lines. The work provides a visual analysis approach for the manufacturing domain that uses an estimation of distribution algorithm (EDA) [5] to provide

- an overview of a specific layout's performance regarding the workers' pathing,
- visual cues, which parts of the layout are most promising for optimization, and
- visual feedback for the most suitable areas for relocation with a heat map visualization.

To evaluate our approach, we implemented a prototypical system called *BlueCollar*. We present an application scenario based on an experimental production line layout provided by a production optimization company to demonstrate the applicability of our approach.

URI: https://hdl.handle.net/10125/59598
ISBN: 978-0-9981331-2-6

2. Related Work

With an increasing degree of digitalization in the manufacturing domain, data become available for process monitoring, analysis, and optimization. There are many areas where visualizations and visual analytics are used to improve a factory's overall performance. For example, to find bottlenecks [6] and anomalies [7] in production line processes, as well as by gaining general process insights [8]. Further, interactive visualizations can be used to optimize work schedules [2]. In the following, we present previous approaches for layout planning and visualization support based on optimization and machine learning algorithms to clarify our approach's novelty.

2.1. Layout Planning

Many approaches exist that apply optimization rules and algorithms to find a satisfying layout [9]. Often, layout optimizations and layout simulations are used together to improve the outcome of the layout planning phase [10]. Modern layout planning approaches also make use of computer-aided design models combined with visualization to make use of human experts' domain knowledge in the planning process [11] or rely on automatic approaches [12].

2.2. Visualization Support based on Optimization and Machine Learning Algorithms

Recently, deep neural networks become increasingly popular for classification or recommendation systems, although their internal mechanisms are often difficult or impossible to understand [13, 14]. Further, such approaches need a lot of training data (that may not be available) and they are unresponsive during the training, as their intermediate results often cannot be used.

Alternatively, evolutionary algorithms (EA) are popular to optimize black box functions [15]. They provide intermediate results and allow experts to manipulate the results during runtime. EAs are based on the biological principle of population recombination and their mutation. The principle of recombination implies that combining two well-performing population members has a good chance to yield a better result. In many scenarios, this approach achieves satisfying results, but it is unsuitable in case good parameter sets may be unrelated (e.g., in case of multiple local optima).

Other optimization approaches, such as simulated annealing [16], use physical models to control the search space used to find a global optimum for a black-box function. They are easy to understand and provide intermediate results. However, they are heavily reliant on their starting configuration and they are unable to represent multiple areas with similar function values.

In this work, we chose to use an estimation of distribution algorithms to provide recommendations. The goal of EDAs is similar to evolutionary algorithms as they also iteratively build sample populations and optimize a cost function of the population members. Unlike evolutionary algorithms, they create a high-dimensional probability space (dependent on the number of parameters) to pick new population members. This is easier to understand compared to the recombination effects used by evolutionary algorithms. Further, the high-dimensional probability space can be used to visualize intermediate results. There are various alternatives to visualize such high-dimensional data, for example, scatterplot matrices [17], parallel coordinate plots [18], glyphs [19], or projection techniques [20], which can be presented as scatter plots or heat maps.

Other layout planning approaches often either show a simulation of a layout's performance or provide multiple possible solutions to optimize a solution [21]. In most cases, experts are only part of this optimization process as decision makers to choose between the available results, after the algorithm completed.

Our approach combines the advantage of EDAs, which are easy to comprehend and their intermediate results can be visualized, with visual analytics concepts that aim to include human experts in an analytics process through interaction with visualizations. To the best of our knowledge, there are no visual analytics approaches that combine EDAs with visualization techniques to provide interactive visual recommendations to optimize the planning of factory layouts.

3. Approach

Targeting layout planning experts, our visual analytics approach supports experts to improve the efficiency of factory layouts by optimizing the paths workers have to take to complete a work schedule. In the following, we first present the requirements and the resulting workflow of our approach. Afterward, we present the different components and the underlying estimation of distribution algorithm (EDA) [5], which our approach uses to provide optimization recommendations, in detail.

3.1. Requirements & Workflow

We identified three system requirements for interactively optimizing factory layouts regarding the workers' pathing. The requirements were derived in previous informal expert interviews.

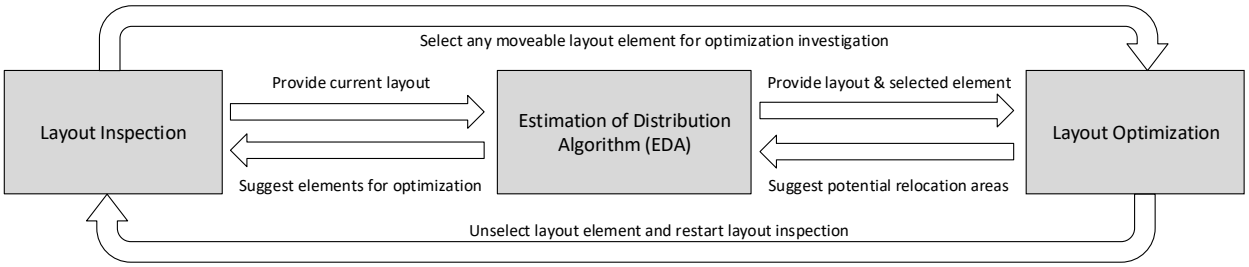

Figure 1: Our approach comprises two stages that use an estimation of distribution algorithm (EDA) to provide visual optimization recommendations. The layout inspection provides an overview of the layout and recommends layout elements that have high potential to improve the layout performance. The layout optimization recommends suitable areas to relocate specific elements to.

> Requirement 1: **Current Performance** (R_1)
> Provide information about the current layout's performance regarding the taken paths of workers.

> Requirement 2: **Overview Guidance** (R_2)
> Present visual feedback, which layout elements can be optimized and how high their optimization potential is.

> Requirement 3: **Optimization Guidance** (R_3)
> Visualize suitable relocation areas for specific layout elements and provide information about the impact of the relocation on the layout's performance.

Based on these requirements, we developed a visual analytics approach that is composed of two stages (see Figure 1). Both stages support experts to decide, how to continue the optimization through EDA-based recommendations. Initially, planning experts can inspect the layout and get a first overview about the positions of the layout elements (e.g., machine tools or tool caches). At this point, BlueCollar provides information about the most likely taken paths of workers, the current layout's performance regarding the workers' pathing, and which elements have the highest optimization potential. Once experts decide, which element's position to optimize, the EDA automatically recommends new layouts. Further, BlueCollar visualizes suitable areas for the relocation of the selected layout element through a progressively rendering heat map visualization. Based on these recommendations, experts can manually modify the layout and continue the optimization.

3.2. Layout Inspection

To optimize a layout, experts first need to get an overview of the status quo. BlueCollar provides an overview of the current layout and enables users to inspect the performance either for the entire workforce or by selecting individual or groups of workers. The analysis starts with an already existing factory layout and a planned work schedule (see Figure 2 Ⓐ). A work schedule describes all of the steps needed to complete the production of certain goods. As our approach targets the optimization of the workers' paths, we restrict the work schedule to tasks that require workers to walk to other machinery.

Layout Performance Overview. To get an overview of the current layout's performance, BlueCollar presents the current layout as a 2D plan view. Every layout element is represented by its rectangular bounding box and an additional icon to represent its function. A legend provides detailed information about the icons' meanings (see Figure 2 Ⓔ). Further, the most likely taken paths of the workers (based on the shortest walking distance, which we calculate using the A* path-finding algorithm [22]) are indicated through semi-transparent polylines. It is possible that some paths are taken multiple times, by either a single worker or multiple workers. Therefore, the line segments that were used multiple times are less transparent than segments only used once. This provides layout planners with an overview of the current layout's performance, which meets requirement (R_1).

Data Filter. Layout planners can also view the performance of individual workers or specific groups of workers, e.g., the group of assemblers or machine operators. The paths are shown as lines in the layout. They can be filtered by selecting them individually or based on their task in the side panel (see Figure 2 Ⓑ).

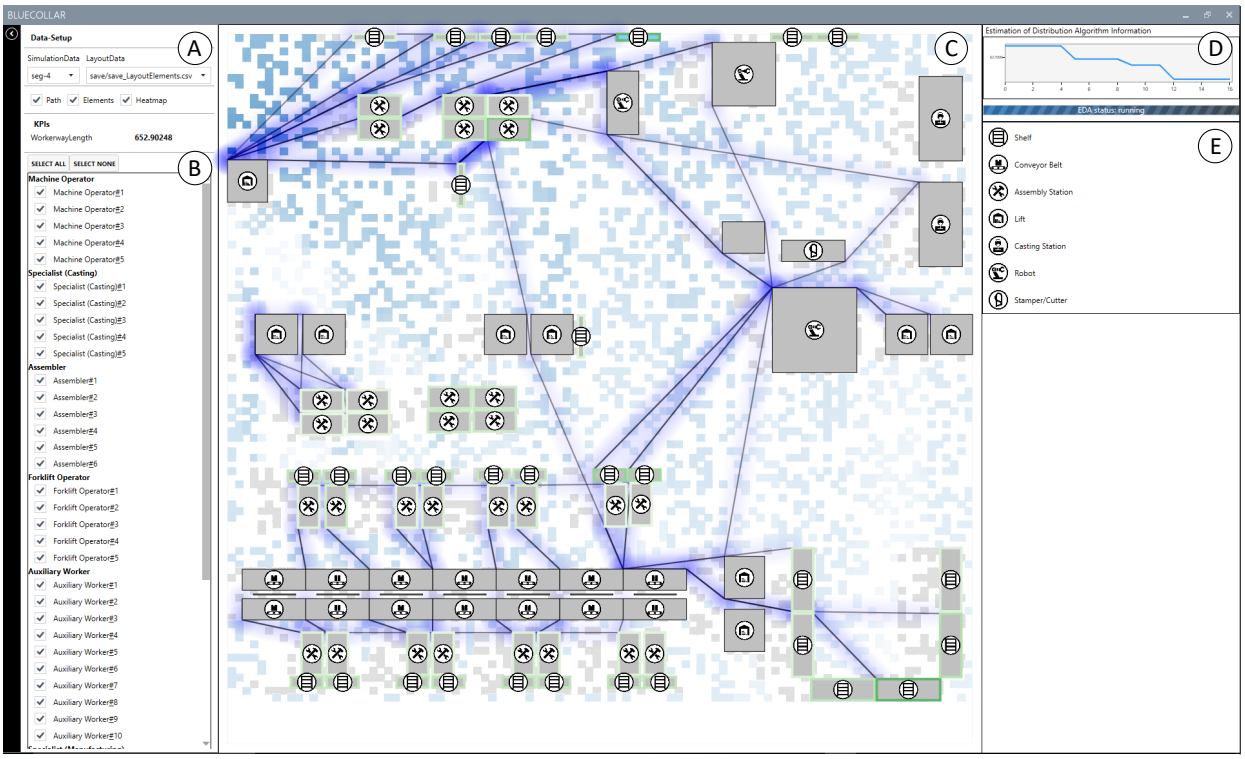

Figure 2: BlueCollar requires layout planning experts to load a layout and work schedule Ⓐ and choose which workers' schedule data Ⓑ to include in the layout optimization. The factory's layout Ⓒ provides a first overview of the current layout and the taken worker paths. Further, it recommends suitable layout elements for optimization through a color coding of their border's color and presents fitting relocation areas as a heat map. The elements' types are indicated by icons, which are explained in a legend on the right Ⓔ. A line chart Ⓓ shows the progress of the optimization algorithm.

To get a better overview of high traffic zones, BlueCollar provides a heat map that encodes the worker density in the layout view. This information can be used to get first insights about possible current or future bottlenecks, where workers may collide or have to take detours. To indicate such potentially problematic zones, the heat map ranges from being entirely transparent (meaning no traffic) to blue (much traffic). In contrast to the path lines, which emphasize the taken paths, the heat map emphasizes areas with highly frequented crossings.

3.3. Visual Optimization Guidance

BlueCollar guides experts during the optimization process in two stages: it recommends elements that have high optimization potential and it suggests better positions for these elements. In the following, we present the visual guidance in more detail.

Layout element recommendation. BlueCollar indicates the optimization potential of movable layout elements, such as machines or tool caches by color coding their borders. A light green border indicates very small changes. The greener the border becomes, the higher the layout element relocation potential is. The possibility of a worse performance can be discarded, as leaving the layout element at its current position is always a possibility and therefore marks the worst case for the optimization. To compensate for the non-uniform distribution of the values, we use a non-linear color saturation mapping following the mapping used by Liu et al. [23]. The optimization potential indication iteratively refines the optimization potential in the background. Figure 3 shows an example, where BlueCollar provides the varying optimization potential of multiple layout elements. Section 3.4–*workload distribution to assess multiple layout elements* details how the optimization algorithm handles the evaluation of multiple layout elements. The layout element recommendation meets requirement (R_2).

Relocation recommendation. Once planners select a specific layout element for optimization, the element is highlighted with a light blue background and the EDA will only optimize the position of the selected element. To give the experts an overview of the optimization's progress, a line chart (see Figure 2 Ⓓ) shows the best layout scores after each iteration. The x-axis shows the iteration and the y-axis the cost. If the cost value decreases, a better relocation position was found.

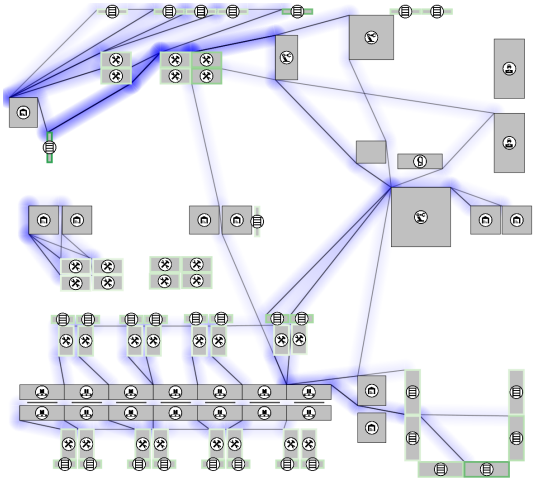

Figure 3: BlueCollar provides a visual indication of the optimization potential for movable components through the border of the elements. The color ranges from light green (low potential) to dark green (high potential).

Additionally, BlueCollar visualizes the optimization progress by mapping already available results onto a heat map visualization overlay on top of the currently viewed layout. This results in a progressively updating heat map, which gives planners an early impression of possible relocation areas. The heat map's color scheme ranges from white (worst performance) to dark blue (best performance) and uses a min-max normalization of the available data. Due to the way EDA works, the surroundings of well-performing relocation areas are more likely to be sampled, which additionally emphasizes these regions. Figure 4 shows the suggested relocation areas for the selected casting station Ⓓ based on a simplified work plan. The legend on the right side of Figure 4 shows the heat map's color coding. The results show that the station should be relocated between the conveyor belt Ⓐ and the robot station Ⓒ. As the heat map includes information about the relocation areas and their optimization potential, they meet analysis requirement (R_3).

Planners can also relocate layout elements manually by dragging them inside the currently viewed layout. If they previously selected the dragged element to get relocation recommendations, the heat map keeps updating until they unselect the element. Otherwise, the heat map is cleared and the EDA is restarted with the currently dragged layout element as the selected element.

3.4. Optimization with an Estimation of Distribution Algorithm

The suggestions of suitable layout elements for relocation (see Section 3.3–*component recommendation*) and the recommendations for suitable relocation areas (see

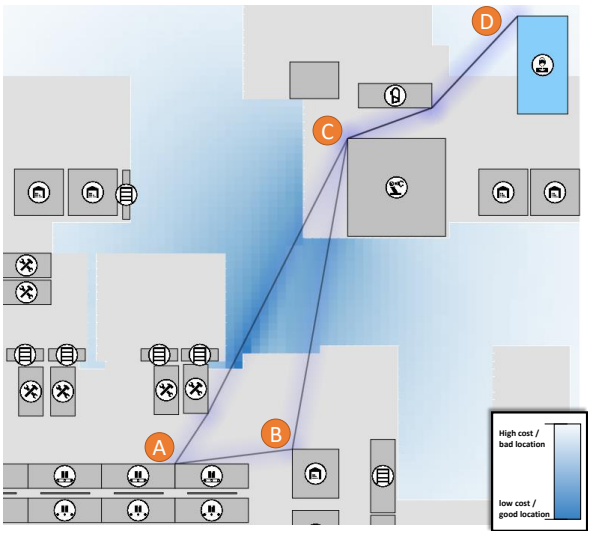

Figure 4: The worker has to walk in a circle (Ⓐ→Ⓑ→Ⓒ→Ⓓ→Ⓐ). The legend on the right shows the heat map's color coding. The heat map indicates that the selected casting station Ⓓ should be relocated between the conveyor belt Ⓐ and the robot station Ⓒ.

Section 3.3–*relocation recommendation*), are based on an estimation of distribution algorithm. In the following, we will briefly introduce the goal of EDAs, their general step-by-step procedure, and a detailed explanation of the steps in the context of our application scenario.

Goal. Estimation of distribution algorithms are a class of optimization algorithms and therefore aim to minimize the output of a given black-box function (usually referred to as cost function) by exploring a (possibly high-dimensional) input parameter space. The cost function itself is usually unknown, but it can be evaluated for any given input parameter set (whereas the parameters' ranges can be defined beforehand). To find an optimal solution, the algorithm builds an n-dimensional probability space, where n is the number of parameters. The probability space defines the likeliness that a certain parameter set is picked for an evaluation.

General step-by-step procedure. Figure 5 shows the general steps of an EDA. It initializes with a uniform probability distribution and then iteratively picks parameter sets for the black-box function based on the probability space. The picked sets are evaluated and the probability space updates depending on the evaluation results.

Generally, there are options for the probability space update: i) rebuild the probability space depending on all previously evaluated input parameter sets (requires more memory); ii) use only the evaluated input parameter sets of the last iteration (requires more computing power, as larger populations need to be picked). We decided on

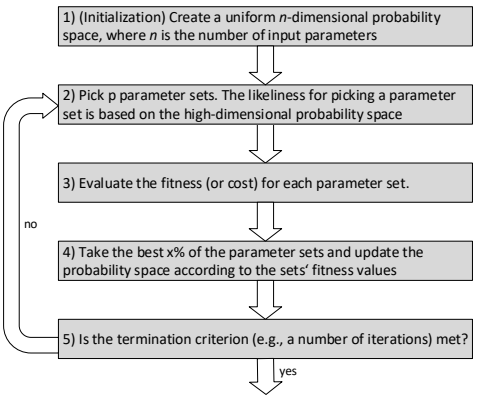

Figure 5: Generally, an EDA comprises five steps [5]. After initializing its probability space uniformly, it iteratively picks parameter sets based on the probability space. Then, the probability space is iteratively updated by evaluating the picked parameter sets. This process repeats until a predefined termination criterion is met.

the first option, as we use the evaluated input parameter sets for our heat map visualization. Although the probability space may converge faster in the latter option, the probability space may change considerably in each iteration step, resulting in quickly changing heat maps, which increases the visualization's cognitive load.

Modeling the dependency of variables. Simple EDA implementations such as the univariate marginal distribution algorithm (UMDA) [24] assume the independence of all input parameters. However, in many scenarios, including ours, the parameters depend on each other and an independence assumption will produce incorrect results. To represent such dependencies, a joint distribution needs to be modeled, e.g., as a Bayesian network [25].

Layout construction and its cost calculation. We use the combined length of all paths that the workers have to take to complete a work plan as the cost function to measure a layout's performance. To efficiently calculate the workers' paths, we first map the layout to a graph representation. We assume that each layout element has a rectangular bounding box. Each element comprises four corner vertices. In an initialization step, all vertices are connected with each other (which results in a complete graph). After that, edges that intersect any layout element are removed. To calculate a workers path, we split the corresponding work plan into individual tasks. Then, we calculate the optimal path to solve each task that requires the worker to change the location using the A* pathfinding algorithm [22]. Afterward, we reconstruct the worker's total path by concatenating all task-based paths. At last, the path lengths of each worker are summed up.

Optimization of the cost calculation's runtime. As generating the layout graph is the most expensive operation, we optimized the relocation operation to reuse the original graph and perform as few graph changes as possible. To do this, we use the fact that our implementation of EDA only optimizes the position of one layout element at a time. We assume that the changing layout element is known in advance and initialize the complete graph without this element. Then, the element can be added at an arbitrary position. We use a quadtree to check if the newly added layout element collides with any other layout elements or edges. In case the layout element collides with another element, we assume the layout is invalid and increase the layout's cost value accordingly. To prevent the position layout from disproportionally decreasing the picking probability of surrounding configurations (see *updating the probability model* below), we set the cost value of an invalid layout to a plausible value that would indicate a very inefficient layout: $cost_{max} = s \cdot l_{layout}$, where s is the total number of path segments and l_{layout} is the width of the layout.

If the layout element collides with connections, these are temporarily removed from the graph. Afterward, new connections to the other elements are added analogous to the initialization step. Instead, the connection is implicitly replaced by two edges: one from the starting vertex to the inserted element and one from the inserted element to the ending vertex. For further relocations of the element, the only needed adaption is to remove the previously inserted layout element and its edges and add the deleted edges again before adding the layout element elsewhere. This skips the initialization step, which is the most expensive part of the procedure.

To improve the pathfinding's scalability regarding the number of workers, we added a lookup table to the A* algorithm that stores all previously calculated movement tasks. This reduces the calculation time, as each path needs to be calculated only once.

Updating the probability model. The step of updating the probability model to choose the next iteration's parameter sets is dependent on the assumptions of the parameters' relations. In addition to their dependency, we also assume that the evaluated parameter sets' cost is not specific for its own configuration, but it can also be assumed that the neighboring cells have similar values. Consequentially, we do not only add the cost for the recently evaluated parameter configuration, but we also approximate the cost for its neighboring cells. By doing so, the likeliness for a cell to be picked is influenced by the already evaluated neighboring cells. We modeled the influence of neighboring cells as a two-dimensional triangular function, where the center has the highest influence and the values are halved for every step towards the outer

1	2	4	2	1

Figure 6: Exemplary one-dimensional weighting stamp that influences all neighbors with distance ≤ 2.

corner of the neighborhood (see Figure 6 for an example of its 1D counterpart). The size of the stamp depends on the dataset. In the dataset used in Section 4, we use a 100x100 cell grid for the heat map and adapted the stamp size to cover five cells in every direction.

Without further modifications, the algorithm is likely to get stuck in local minima, as the initially picked parameter sets strongly influence the probability space's development. To prevent this, we reserve a base probability of 10% for all elements to be picked (e.g., if there are four elements, each of the elements will have a base probability of $\frac{10\%}{4} = 2.5\%$ to be picked).

Overall runtime complexity. The runtime of each iteration depends on the cost to pick and evaluate the population members and the cost to update the probability space. In our EDA implementation, the runtime cost is $\underbrace{p \cdot \mathcal{O}(d^2 + runtime_{eval})}_{picking} + \underbrace{\mathcal{O}(k^2 + k)}_{picking\ update}$, where d is the size of the grid's dimensions and k is the size of the kernel. The cost of picking new elements mainly depends on the runtime complexity of the evaluation function, which depends on the number of necessary node expansions in A*. The cost of updating the probability space depends in our case on the number of nodes that need to be updated. In the worst case, none of the picked population members were evaluated before and therefore, every element's neighborhood (which is k^2) must be updated, leading to the runtime complexity given above.

Workload distribution to assess multiple layout elements. Until this point, only the position optimization of a previously known layout element was considered. However, the optimization potential recommendation implies that various layout elements were evaluated. To do this, all movable layout elements are first added to a queue. Then, a layout element is taken from the queue and one pass of the estimation of distribution algorithm is calculated with that element. The best layout result's performance is used as the optimization potential value. Afterward, all calculated layouts and their performance values are stored for that layout element and the element is added to the queue again. If the layout element that is taken from the queue already contains calculated positions, these are used to initialize the probability space. The main challenge in this step is to balance the calculation time assigned to each layout element to calculate its optimization potential against the evaluation of as many layout elements in as little time as possible. This means that more time per layout element results in a better indication quality of individual layout elements, while less time means that more layout elements can provide an indication, but with a lower indication quality. Therefore, the algorithm only estimates a limited amount of new locations for each layout element. From our experience, it seems to be a good trade-off to run one EDA iteration that generates $\sqrt{n}$, with n being the number of grid cells, relocation positions for the layout element and then put the layout back to the end of the queue.

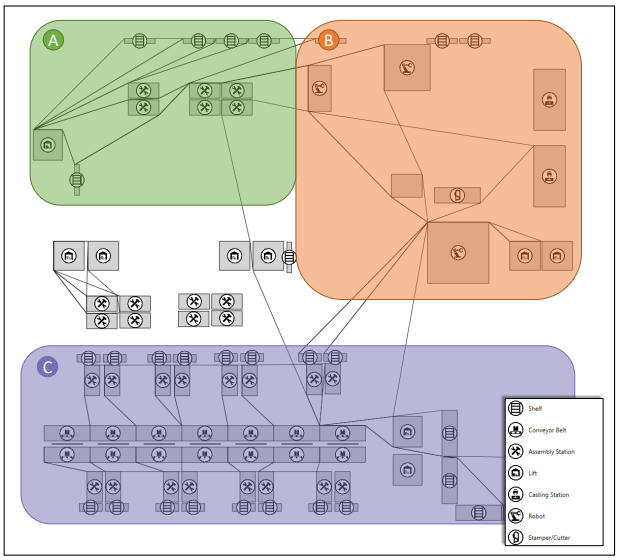

Figure 7: The use case's layout can be subdivided into three areas: The green area Ⓐ comprises assembly stations and multiple shelves that are connected to a lift. The orange area Ⓑ contains mainly robot stations. The purple area Ⓒ contains a conveyor belt, which is connected to multiple assembly stations and a high rack storage.

4. Application Scenario

A software engineering company that offers planning and production simulation software provided us with the layout used in the following application scenario. The production layout was planned for a prototypical production line and focuses on manually operated assembly stations between which the workers have to change. The layout is 35.23 meters wide and 31.5 meters long. We added plausible work schedules that incorporate different tasks in several areas of the factory. It can be subdivided into three areas (see Figure 7). The green area Ⓐ comprises several shelves that are refilled and assembly stations that produce parts for a robot station. Further, one station gets supplies from a separate shelf. Area Ⓑ, highlighted in orange, contains three robot stations. The bottom-most station's goods are continuously delivered to two lifts. The purple area Ⓒ is composed of two conveyor belts that transport goods from an (unmodeled) external supplier. The workers at the assembly stations

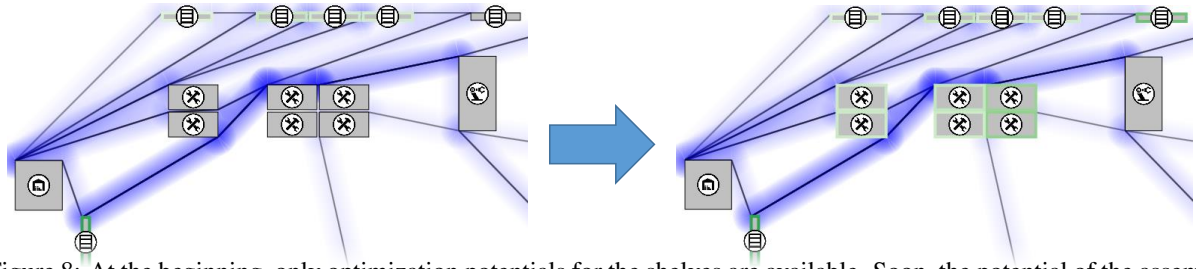

Figure 8: At the beginning, only optimization potentials for the shelves are available. Soon, the potential of the assembly stations becomes available as well, but the shelf at below the lift stays the station with the highest optimization potential.

along the conveyor belts take the goods, process them, and put them back on the belt. At last, the parts are taken off the belt and get stored in a high rack storage area.

Exploration of optimization potential. Layout planning experts would start an analysis with the layout view as presented in Figure 7 (except for the highlighting). It seems clear that the planned layout can be optimized, as there are many long paths to a limited number of stations. One of the areas that seems most viable for optimization is area Ⓐ. The positions of the lift and the robot are fixed, but the shelves at the top, the assembly stations, and the shelf below the lift can be relocated. However, it is unclear, which layout element has the highest optimization potential. Therefore, they base their decision on the optimization suggestion that BlueCollar continuously extends and improves (see Figure 8). The system recommends optimizing the shelf below the lift. With this insight, they manually relocate the shelf towards the block of assembly stations (see Figure 9 Ⓐ).

Optimization of the layout. To further optimize the shelf's position, the experts decide to select it and optimize its position with the support of BlueCollar's relocation heat map. As the analysis progresses, it becomes apparent that the most suitable position for relocation is in the gap between the two groups of assembly stations (see Figure 9 Ⓑ – Ⓓ). Although the best position is plausible, they decide to leave it below the assembly station. Based on total walking distance, this position may be worse (650m vs. 640m), but their placement prevents crowding of workers in a small area.

After the shelf's relocation, the planners go back to the element recommendation mode to confirm that the shelf at the top still has the same optimization potential as before. Once done, they select the shelf to get detailed relocation recommendations (see Figure 2). BlueCollar recommends to place the shelf directly above the lift (reducing the distance to 623m), but the experts opt to place it to the left of the top shelves to keep obstructions at a minimum, still reducing the path length to 632m.

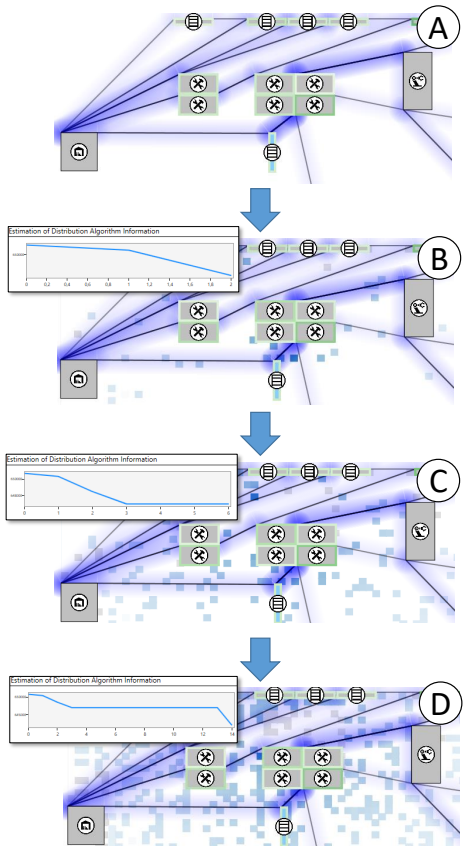

Figure 9: The longer the optimization continues, the more apparent it becomes that the most ideal relocation area is between the two groups of assembly stations.

5. Discussion and Future Work

Our approach aims to support layout planning experts in the production domain to improve layouts. In addition, once a layout was deployed, the approach can be used by technicians if they want to propose layout improvements, as they can use the visualization to verify or falsify the effectiveness of their proposed optimization. Further, this approach could also be applied to other planning fields such as urban planning, traffic concepts for fairs, and many more. In the following, we will discuss our approach regarding performance, visual scalability, its limitations, as well as open challenges.

Our EDA implementation converges quickly towards good results but has difficulties to find the most optimal

result (due to its probabilistic nature). Therefore, we measured, how many iterations our EDA implementation needs to find one of the ten best possible results. For this, we used the simple layout presented in Section 3.3 and the complete plan used in Section 4. The used population size per iteration was 100 and we measured the needed iterations 100 times per layout. In the following, we will give the evaluation results with an annotation of $\varnothing(average) \pm \sigma(standard\ deviation)$. For the simple plan, the EDA needed 5.18 ± 3.04 iterations to find a top 10 value. For the complex plan, it needed 6.12 ± 6.48 iterations for a top 10 value. We benchmarked the values against a random sampling, which resulted in 12.07 ± 10.40 iterations for a top 10 pick. Overall, the EDA does not only find suitable values in fewer iterations than random sampling, but it also guides planners quicker towards areas of interest, as areas that perform better have a higher chance of being evaluated and therefore being highlighted.

Currently, our approach only supports relocation recommendations for one layout element at a time for a comparably small factory layout. We discretized the layout as a 100x100 cell grid for the heat map and the optimization algorithm. For our layout, this corresponds to cells of approximately 0.3 m^2. A larger layout with the same cell size results in a bigger grid, which affects the probability space modeled by EDA. First, this means that more cells need to be evaluated to provide a heat map that shows trends of good relocation areas. Second, the memory needed to store the probability space (or the already calculated fitness values) grows exponentially. This may pose the even more severe problem and the optimization algorithm may need to be adapted to only use the previous sampling generation as explained in Section 3.3–*general step-by-step procedure*.

The scalability regarding the number of layout elements to optimize is more challenging. It may be necessary to move groups of layout elements to achieve an improvement of the layout's overall performance. The optimization of the first shelf in the application scenario (Section 4) is a good example for this, as the optimization will always try to put the shelf beside the top right shelf of the 2x2 shelf block, as workers need to go back and forth between them, no matter where this shelf is located. In such cases, a simple color coding as it currently exists is insufficient, as it would be difficult to visually encode the layout elements' dependencies without additional visual aids, which may result in visual clutter. One possible solution for this issue is the introduction of additional views that propose groups of layout elements that have a high symbiotic potential. These views can be more abstract and interact with the layout view through a brushing & linking approach. We plan to investigate the potential benefit of such an extension in the future. The visualization of the layout optimization is also limited regarding its scalability towards multiple elements. Heat maps are designed to visualize the (possibly weighted) distribution of independent data points and are unfit to visualize relocation areas of position tuples.

The simulation view is mostly independent of the number of simulated workers or an extension of their work schedule, as it mostly visualizes aggregated data. At the same time, the calculation times of the layout's performance are heavily dependent on the speed of the path calculation, which is impacted by the number of workers and their work schedule. However, there are many approaches to increase the scalability issues of A*, e.g., by restricting its memory consumption [26] or by adaption to changing layouts [27].

As we opted to store all evaluated grid cells, the completion time for one EDA iteration increases with every iteration. The main reason for this effect is that we manipulate the neighborhood of the newly evaluated cells, which need to take all already calculated cells in their own neighborhood into account. Despite this effect, our approach is guaranteed to find the global optimal solution eventually and also stabilizes the heat map visualization. Further, the optimization algorithm and its heat map representation already provided first indications about suitable areas within the first five iterations. More iterations slightly improve the results, but often the affected areas are already discoverable before they are evaluated. However, we think that this finding is dependent on the used layout's size and overall complexity and that this topic needs further investigation. Additionally, we plan to investigate the trade-off between our approach that stores all results and the iterative approach that discards all results except for the last generation's. The latter approach may cause issues with the users' mental maps due to the constantly evolving and possibly considerably changing set of evaluated grid cells (which affect the presented heat map).

Currently, an expert's only possibility to interact with the EDA is by manually changing the layout and therefore forcing a reset under possibly better conditions. We plan to allow experts to directly influence the probability space interactively to guide the optimization algorithm at the beginning of the process. This may lead to a faster convergence of good results. Such a manipulation could be realized by brushing the layout to indicate interesting areas. However, such an interaction faces similar problems as the probability space's visualization, as the visualization and interaction of high-dimensional data pose a separate challenge.

Finally, we intend to evaluate the applicability of our approach with industry experts. Especially compa-

nies that produce many product variants in small batch sizes may benefit from tools that support them in quickly adapting their factory layout to their current needs. An interesting aspect may be the trade-off between the improved performance regarding the layouts efficiency and a possible negative impact if workers need to adapt to the resulting changes in their everyday work.

6. Conclusion

In this paper, we presented BlueCollar, a visual analytics approach that supports factory layout planning experts to optimize layouts regarding the paths taken by workers on the shop floor. An estimation of distribution algorithm continuously updates a probability space that is used to recommend layout elements with high optimization potential. Further, BlueCollar provides suitable areas for the relocation of layout elements selected by experts through a heat map visualization. The experts can interactively manipulate the layout and iteratively improve the layout's performance by doing so. We presented an application scenario to demonstrate the applicability of our approach.

7. Acknowledgments

We thank iFakt GmbH for their kind support and for providing us with the data and scenario used in this work.

References

[1] A.-C. Falck, R. Örtengren, and D. Högberg, "The impact of poor assembly ergonomics on product quality: A cost-benefit analysis in car manufacturing," *Human Factors and Ergonomics in Manufacturing & Service Industries*, vol. 20, no. 1, pp. 24–41, 2010.

[2] 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*, vol. 20, no. 12, pp. 2329–2338, 2014.

[3] J. P. Shewchuk, "Worker allocation in lean u-shaped production lines," *International Journal of Production Research*, vol. 46, no. 13, pp. 3485–3502, 2008.

[4] D. Mourtzis, M. Doukas, and D. Bernidaki, "Simulation in manufacturing: Review and challenges," *Procedia CIRP*, vol. 25, pp. 213–229, 2014.

[5] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111–128, 2011.

[6] M. Wörner and T. Ertl, "Visual analysis of advanced manufacturing simulations," in *International EuroVis Workshop on Visual Analytics*, vol. 2011, pp. 29–32, 2011.

[7] A. Maier, T. Tack, and O. Niggemann, "Visual anomaly detection in production plants," in *ICINCO*, pp. 67–75, 2012.

[8] 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*, vol. 23, no. 1, pp. 291–300, 2017.

[9] A. Drira, H. Pierreval, and S. Hajri-Gabouj, "Facility layout problems: A survey," *Annual Reviews in Control*, vol. 31, no. 2, pp. 255–267, 2007.

[10] E. E. Aleisa and L. Lin, "For effective facilities planning: Layout optimization then simulation, or vice versa?," in *Proceedings of the Winter Simulation Conference*, pp. 5–pp, 2005.

[11] H. M. Osman, M. E. Georgy, and M. E. Ibrahim, "A hybrid cad-based construction site layout planning system using genetic algorithms," *Automation in construction*, vol. 12, no. 6, pp. 749–764, 2003.

[12] R. S. Liggett, "Automated facilities layout: Past, present and future," *Automation in construction*, vol. 9, no. 2, pp. 197–215, 2000.

[13] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards better analysis of deep convolutional neural networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 91–100, 2017.

[14] P. E. Rauber, S. G. Fadel, A. X. Falcao, and A. C. Telea, "Visualizing the hidden activity of artificial neural networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 101–110, 2017.

[15] M. Wörner, *Visual analytics for production and transportation systems*. PhD thesis, Universität Stuttgart, Insitut für Visualisierung und Interaktive Systeme, 2014.

[16] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, *et al.*, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[17] J. A. Hartigan, "Printer graphics for clustering," *Journal of Statistical Computation and Simulation*, vol. 4, no. 3, pp. 187–213, 1975.

[18] A. Inselberg, "The plane with parallel coordinates," *The visual computer*, vol. 1, no. 2, pp. 69–91, 1985.

[19] J. Bertin, *Semiology of graphics: diagrams, networks, maps*. Madison, WI, USA: University of Wis., 1983.

[20] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[21] M. Wörner, M. Metzger, and T. Ertl, "Dataflow-based visual analysis for fault diagnosis and predictive maintenance in manufacturing," in *International EuroVis Workshop on Visual Analytics*, pp. 55–59, 2013.

[22] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[23] Z. Liu, B. Jiang, and J. Heer, "immens: Real-time visual querying of big data," *Computer Graphics Forum*, vol. 32, no. 3pt4, pp. 421–430, 2013.

[24] H. Mühlenbein and G. Paass, "From recombination of genes to the estimation of distributions i. binary parameters," *Parallel problem solving from nature (PPSN IV)*, pp. 178–187, 1996.

[25] J. Pearl, "Bayesian networks: A model of self-activated memory for evidential reasoning," *Proceedings of the 7th Conference of the Cognitive Science Society, 1985*, pp. 329–334, 1985.

[26] S. J. Russell, "Efficient memory-bounded search methods," in *ECAI*, vol. 92, pp. 1–5, 1992.

[27] A. Stentz, "Optimal and efficient path planning for unknown and dynamic environments."