DATA-DRIVEN RECONSTRUCTION OF PROCESSES FROM PEDESTRIAN TRAJECTORIES

Elena Eftimova[a], Christoph Nellinger[a], and Tobias Koch[a]

[a] Institute for the Protection of Terrestrial Infrastructures, German Aerospace Center (DLR), Sankt Augustin, Germany
{elena.eftimova,christoph.nellinger,tobias.koch}@dlr.de

ABSTRACT

Agent-based simulations can be helpful in understanding the complex dynamics of human behavior. Data-driven approaches for this purpose show to be promising in extracting complex features, without relying on system-specific expert knowledge. This work aims to develop a data-driven approach that enables automatic generation of agent-based pedestrian flow models, by extracting and classifying regions of interest from trajectory data. For validation purposes, synthetic data from a pedestrian movement simulation was used for the method development. We identify stay point areas from the resulting trajectories, classify the processes occurring in these areas, and reconstruct their properties. The relevant areas and types of processes were successfully extracted in four different case scenarios. However, it is necessary to test and subsequently improve these methods by using real data. Ultimately, our methods should be applied for the automatic modeling of pedestrian behavior in critical infrastructures, such as a railway station or an airport.

Keywords: data-driven, agent-based, stay point detection, process analysis

1 INTRODUCTION

Our society increasingly relies on critical infrastructures such as transportation systems, water and energy supply systems and communication networks. It is therefore imperative to understand the processes in these complex and interdependent infrastructures, for supporting maintenance, planning and emergency decision making [1]. To this end, modeling and simulation approaches have become a critical field of study [1, 2].

Particularly in cases where human behavior is involved, agent-based simulations can provide valuable insights into their complex dynamics, by revealing patterns that might emerge under different simulation conditions. Classical approaches for agent-based simulations are typically *knowledge-based*, meaning that they rely on expert knowledge to define the parameters and rules that guide the simulation. While this type of modeling of a system can be effective for predictions and testing the system's working mechanisms, there are several disadvantages that need to be taken into consideration.

Firstly, this approach can be time and energy-consuming, since experts must often improve the simulation in an ad-hoc manner [3]. Furthermore, complex systems involving human workflow processes require the expertise of different fields [4]. This type of modeling also relies heavily on the individual experience and knowledge of the expert, which inevitably introduces a bias in the model construction.

These limitations can be addressed to some extent by using a *data-driven* approach instead. Using real data reduces the modeler bias, because the model is directly informed by the data coming from this system in

Proc. of the 2024 Annual Simulation Conference (ANNSIM'24), May 20-23, 2024, American University, DC, USA
C. Ruiz-Martin, B. Oakes and R. Cárdenas, eds.

action, instead of relying on the theoretical knowledge of an expert [3]. Especially when real time changes of data are concerned, analyzing patterns in the data itself can be beneficial for the purpose of knowledge discovery and movement modeling, also known as computational movement analysis [5]. The increasing availability of geospatial data has led to advancements in this field; however, there is still a need for reproducible research [5].

This work aims to develop a data-driven approach to aid modeling of a pedestrian movement scenario, by extracting and reconstructing the processes in which the pedestrians take part. The methods developed here should be able to extract information about the processes from real-world data in an automated way. In particular, the input data consists of spatio-temporal information about each pedestrian in the scenario, which could eventually be obtained by location acquisition technologies such as GPS or camera tracking. The application domain lies in understanding the processes behind human behavior in critical infrastructures such as a railway station or an airport. Even though the spatial setup in these infrastructures might be known, processes that depend on human behavior can be difficult to predict, unless informed by real data.

In the field of data-driven pedestrian movement modeling, previous work has been successful in accurately generating realistic pedestrian behavior, specifically in simulating uni- and bidirectional movement [6, 7] and crowd behavior [8, 9]. A study by Lui et. al. [10] highlights the need for using real data to learn latent features that reflect the complexity of human movement. They demonstrate the effectiveness of using past data in predicting future pedestrian trajectories in a public building, specifically by discovering information about the destination of a pedestrian. As part of future work, they suggest considering intermittent destinations with which the pedestrians interact (such as a ticket office) and using such regions of interest in the further development of their framework [10].

For these purposes, it is crucial to identify *stay points* in the trajectories, where the pedestrians slow down or stop moving; these points collectively form areas known as *points of interest* (POIs) [11]. Considering that information about congestion areas might be unavailable, or simply change over time, obtaining it directly from trajectory data can be particularly useful [12]. Piekenbrock et. al. [12] successfully discovered POIs by using density-based spatial clustering, validating their method with traffic simulation data. Similarly, Gong et. al. [13] identified important locations from GPS data by combining density-based clustering with support vector machines (SVMs) to classify areas into 'activity' (e.g., work or shopping) and 'non-activity' (e.g., waiting at a green light) stop locations.

In this work, POIs will be identified from trajectory data for a specific pedestrian movement scenario, to uncover the location of important processes in which the pedestrians participate. In addition, the goal is to describe the processes with respect to their individual properties, as well as the interactions between them.

Ultimately, our goal is to use the results from the process reconstruction to enable a realistic model of pedestrian movement. To this end, the extracted properties from our methods will provide explicit information about the behavior of the pedestrians, in contrast to implicitly learning patterns to model pedestrian trajectories, which was done in most of previous literature in agent-based simulations by using machine learning methods. The contribution of this study lies in developing POI detection specifically for the purposes of pedestrian behavior modeling, by first understanding the processes dependent on human behavior. The integration of the extracted processes and their properties into a simulation model, and subsequent reconstruction of the pedestrian movement scenario, goes beyond the scope of this paper and will be part of future work. Presently, we use synthetic data from a pedestrian simulation to develop the process extraction methods, such that controlled test scenarios could be created to generate different data sets. In addition, by using synthetic data, the success of the methods could be directly evaluated.

This paper is organized as follows: Section 2 describes the methods used in this work, including methods for generating the synthetic data, identification and classification of the POIs where processes occur, and reconstruction of the process properties. Next, Section 3 provides a detailed description of the data and case scenarios used for testing the methods. The results of the analysis are then outlined in Section 4, followed by a discussion and conclusion.

2 METHODS

First, we describe the simulation used to generate the synthetic data used for the analysis. Then, we describe the methods developed for identifying areas of interest, following methods for classifying these areas, and finally, methods for reconstructing the properties of the processes that occur within the areas.

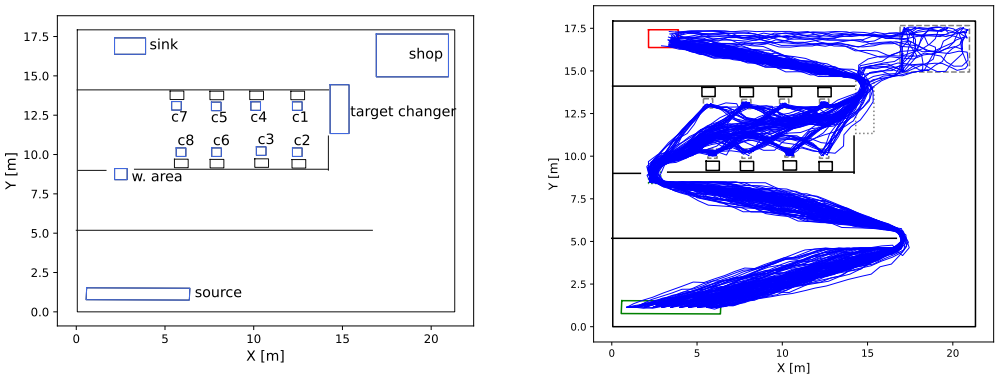
2.1 Synthetic Data Generation

For generating the synthetic data, we use an implementation of a pedestrian simulation based on the optimal steps model described in [14], which performed well in reenacting an evacuation scenario from experimental data. Compared to other methods, this model also incorporates well the social aspect of pedestrian movement and crowd dynamics [14, 15].

In the model used, the moving speed of each pedestrian (also here referred to as an agent) is determined by a truncated normal distribution, with a mean of $1\,\mathrm{m\,s^{-1}}$, a standard deviation of 0.01 and a truncation parameter of 2. The agents move in a given geometry, which can include walls, obstacles and areas where they participate in predefined processes. The processes are characterized by area properties such as waiting time, maximum number of agents allowed at once, a speed adjustment factor, list of next areas and area type. The area type is particularly important, since it implies a special type of process which occurs here. In the simulation used for the purposes of this work, the following area types were defined:

- *Source*: area where the agents appear in the simulation. Input parameters include the spawn rate, which can be either periodic, or normally distributed.
- *Sink*: area where the agents leave the simulation as soon as they enter.
- *Target area*: general area with the possibility of inputting defined waiting time, speed adjustment factor and maximum number of agents. The waiting time can be either fixed, or normally distributed. Sub-types of this area type include *counter*, a generally smaller target area where the maximum number of agents is 1 by definition, and *shop*, where the maximum number can vary.
- *Target changer*: area where there is a probability that the agents change their next area (from the list of input next areas), with each step inside this area. Input parameters include the probability value of changing targets at each step.
- *Waiting area*: area which does not have a predefined waiting time, but the agents are required to wait until a next area (from the list of input next areas) is unoccupied. Only a single agent is allowed at a time; in case the area is occupied, a queue is formed behind it.

The current setup uses these area types, as shown in Figure 1a. The exact input parameters for the processes used to generate the data are discussed later in Section 3. The agents first appear at the source. Then, they pass through the waiting area from which they can go to any available counter, marked c1-8. This waiting area represents the place where the agent that is first in line can wait for their turn to go to a counter. If all counters are full, the agent waits until an area becomes available and then proceeds to it. The counters were



(a) (b)

Figure 1: The spatial setup of the simulation, including the area types described in Section 2.1; The black lines indicate the walls, and the blue lines define the areas. Areas c1 to c8 are counters, a sub-type of target area. The agents appear at the source; refer to text for a detailed description of the agent movement. (a) Trajectories of 120 agents resulting from the simulation (b).

placed in front of *obstacles*, meant to simulate physical counter barriers in front of which the agents take part in a process. After that, the agents proceed to the target changer, where they can go either directly to the sink, or head to the shop. From the shop, the agents can only go to the sink, where they leave the simulation. Resulting trajectories from a single simulation run are shown in Figure 1b. The simulation was run for 4 different case scenarios, with the only changes concerning two parameters: the spawn rate of agents at the source, and the waiting time in the target areas (the counters and the shop). The spawn rate was either periodic or following a normal distribution around a peak time point during the simulation. The objective was to simulate a more realistic scenario, such as a certain time of the day being the busiest, as well as test the robustness of the methods. Similarly, the waiting time at the counters was either fixed, or following a normal distribution, as shown in literature to apply for check-in counters at the airport [16].

The cases were therefore defined as follows: 1) periodic spawn rate and a fixed waiting time for the counters and shop 2) spawn rate with a normal distribution and a fixed waiting time 3) periodic spawn rate and normally distributed waiting time 4) normally distributed spawn rate and normally distributed waiting time.

In order to test the robustness of the methods, the simulation was run 20 times for cases 2, 3 and 4, for which the normal distribution of the spawn rate and/or waiting time caused different data to be generated each time. The exact input parameters are included in Section 3.

2.2 Identification of Areas

The first significant areas to be identified are the source and the sink, which is where the agents enter and leave the scenario, respectively. Then, relevant areas of interest are identified using stay point detection.

2.2.1 Source and Sink Detection

To detect the source, the first coordinates of each agent are considered, and then clustered using density-based spatial clustering for applications with noise (DBSCAN) [17]. Clustering is needed for detecting multiple different sources. DBSCAN detects high-density neighborhoods, where the number of minimum

neighbors for a cluster is defined by the input parameter *minPts*, and the neighborhood radius by ε. In our implementation, *minPts* was 4, as recommended for 2-dimensional data [18], and ε was 0.8.

Similarly, for detecting the sink we consider the last recorded coordinates of the agents, which are then also clustered using DBSCAN. Notably, all agents will not have reached the sink, so their last coordinates could be outside of this area; however, DBSCAN will likely detect these coordinates as outliers and they will not be included in the final clusters.

2.2.2 Stay Point Areas Detection

Stay point detection (SPD) was used to identify regions of little or no movement, where the agents either remain stationary, or move with decreased speed for a significant amount of time. This work implements a version of the SPD algorithm described in [19]. We identify stay points by checking whether consecutive points of a single agent lie within a certain distance threshold. Input parameters for this algorithm thus include a *distance threshold*, and a *time threshold* for limiting the number of points to be considered.

The distance threshold here was set to 1 m and the time threshold to 5 s, such that the agents are considered to stay in one place if they are moving with $0.1\,\mathrm{m\,s^{-1}}$, or at least 10 times slower than the assumed velocity.

In order to obtain POIs, here referred to as *stay point areas*, the extracted stay points are clustered using DBSCAN, as described in Section 2.2.1. The values for the input parameters were the same as for clustering the source and the sink points, with *minPts* = 4 and ε = 0.8. Finally, rectangle-shaped areas are defined around the points, with a padding coefficient, here chosen to be 0.1 m (nearly half of the agent radius).

2.3 Classification of Stay Point Areas

The next step is to characterize the discovered areas based on the type of processes that occur inside the area, as well as any interacting processes. Based on the area types that were used as an input to the simulation, we assume the discovered stay point (SP) areas to fall under one of the following categories:

- *Counter*: SP area where a waiting process of a certain duration occurs, and there are as many agents entering as leaving at a time.
- *Shop*: SP area with many agents being allowed at the same time, and in which they spend a certain amount of time, with a reduced speed.
- *Queue*: SP area with a variable waiting time and variable number of agents at a time. It must causally affect and be affected by its targets, since the agents only occupy this area if the targets are unavailable, and them leaving it leads to the targets being occupied again.
- *Waiting Area*: SP area with only a single agent allowed at a time, and must causally affect its targets (i.e., as a dispatching system).

In order to classify the extracted areas into the above-described categories, we must first obtain information about the processes occurring within. Within a single area, we can now look at the agent flow characteristics, such as the number of agents present at each time point, and the time spent by the agents in each area. However, in this case we should distinguish between agents that spend some time in an area, and agents that simply pass by the area without stopping. To find the latter, the average time necessary to cross the distance of the diagonal in each area was calculated and used as a threshold to filter passing-by agents.

Similarly, we can consider the agent flow between areas, by looking at the order in which the areas are visited. In this way, a directed graph can be constructed with the nodes as areas, and the edges representing the agent flow between them. Hence, each area has *targets*, to which agents can continue, and *precursor areas*, from which this current area can be reached. The shortest path of each area to the source is then used to assign area depth; namely, how many other areas (at least) need to be passed to reach it. We can then define areas with the same depth as areas with *parallel processes*.

Using these observations about the number of agents per time frame, the time spent in each area, and the agent-flow graph, we develop several criteria to classify each area. For testing the criteria, the passing-by agents are filtered out, since we are only interested in the agents which take part in the processes in this area.

Out of the six criteria described below, criterion 1 to 4 use the number of agents per time frame, criterion 5 uses the time spent in each area, and criterion 6 uses the number of SPs per time frame. Using these criteria, we could then build a decision tree, shown in Figure 2. The criteria are summarized as follows:

1. **Alternating derivative**: In order to check whether an agent must first leave an area before other agents can enter it (e.g., a counter), we check if the derivative of the agents per time frame is alternating between negative and positive. A toleration threshold was also set, such that several exceptions were allowed in case of unusual cases, such as agents entering the area accidentally. Here, the toleration threshold was set to half of the simulation time.
2. **Stationarity**: The agents per time frame observations from each area can be checked for stationarity, indicating constant mean and standard deviation. A counter, for instance, is expected to be stationary, since it has a constant number of agents entering and leaving throughout the simulation. To this end, the Augmented Dickey-Fuller (ADF) test was used, from the Python package *statsmodels* [20].
3. **Granger causality to and from target areas**: To determine interdependence between areas, we can check whether the number of agents at a time in one area influences the number of agents in another area within some time lag. For this purpose, a Granger causality (GC) test can be used [21]. In order to uncover dispatching systems, such as the agents leaving a waiting area as soon as any counter is free, we must consider the combined influence to and from its targets. Therefore, for each area, GC was tested between the number of agents per time frame for this area, and the combined number of agents per time frame for all targets from the graph. Notably, areas with the same depth were not considered as part of each other's targets. We check the stationarity assumption for the GC with the aforementioned ADF test. In the case the time series is not stationary, first order differentiation is used to transform the signal before performing the GC test. The maximum lag at which the two series can be compared for considering causality was here chosen to be 20 frames, or 4 s. The GC test was performed using the implementation from the Python package *statsmodels*.
4. **Granger causality from precursor areas**: Areas can also be checked for influence from precursors. This criterion can help characterize a waiting area, by checking for GC by its precursor, a queue.
5. **Gaussian distribution of time spent in area**: We test whether the waiting time in each area follows a Gaussian distribution, by using the Shapiro-Wilk test for normality from the Python package *scipy*. If normality does not hold, this can indicate a queue rather than a shop or counter.
6. **Constant boundary**: For all areas, we can draw a boundary box around the SPs at any time point and check how the area varies over time. For areas such as a queue or a shop where there are more SPs at certain times, it is expected that the size of the boundary is not constant, as opposed to the counters, where only a certain number of agents is allowed at a time.

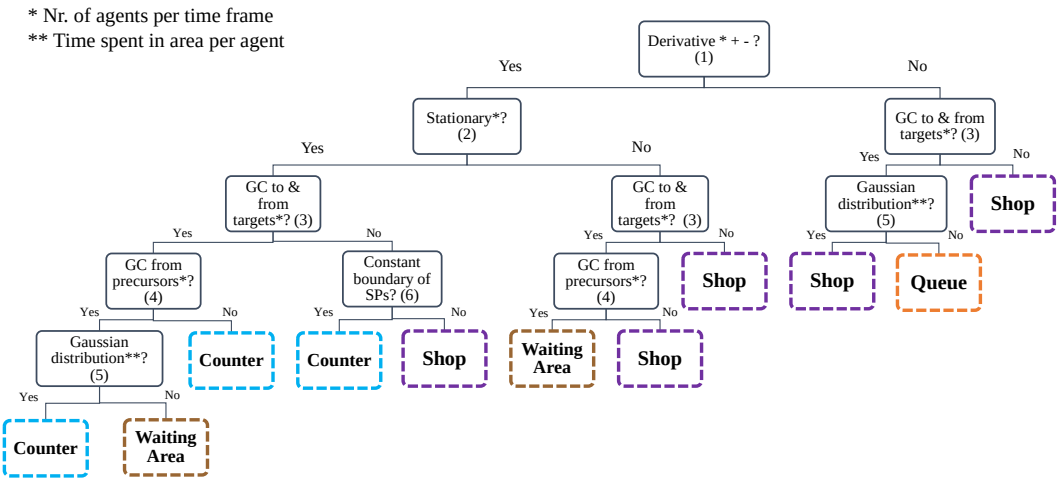


Figure 2: Decision tree used for classifying areas into different types; the criteria 1-6 are described in Section 2.3. The asterisk next to the criteria indicates which time series is used to check the criterion.

For all statistical tests used to check the criteria, a p-value of 0.05 was used.

2.4 Extraction of Process Properties

Apart from the area type and agent flow characteristics, individual properties for each area were also obtained, namely the spawn rate, waiting time, speed adjustment factor and maximum number of agents.

For estimating the spawn rate parameters at the source, we extract the time frames at which the agents were spawned and check whether the difference between them is constant, indicating a periodic spawn rate. For cases 2, 3 and 4, we combine the data from the 20 simulation runs and if the spawn rate is found to be non-periodic, we fit a Gaussian distribution on the combined data to estimate the parameters. The merging of data was done in order to ensure enough data for the Gaussian optimization function.

The reconstruction of the waiting time is important for counters and shops, since the processes in these areas usually take a predictable amount of time. The assumption is that each agent spends a certain amount of time coming from a normal distribution; for this reason, a Gaussian distribution was fitted for the times the agents spent there, and the resulting mean and standard deviation then describe the waiting time property. As described previously for the spawn rate estimation, for each of the cases 2, 3 and 4 we combine the recorded waiting time data from the 20 simulation runs, for the matching areas.

To calculate the speed adjustment factor for an area, the average velocity of agents which stay in this area was calculated. Then, this value was divided by the average moving velocity, which is defined as the average velocity an agent has outside the SP areas. The final speed adjustment was then the average of these values.

Another input parameter for the simulation that could be reconstructed was the maximum number of agents allowed inside an area. This value was simply taken from the data to be the maximum number of agents that appears at the same time, thereby only considering the agents with a stay point in this area.

3 DATA

The input parameters to the pedestrian simulation for generating the synthetic data are included in Table 1, which describes the values for all predefined properties for the areas in Figure 1a. The distribution

Table 1: Description of area properties in original setup, matching the areas from Figure 1a. The waiting time is given in seconds. The dash indicates that this parameter is not defined for this area (None).

area	source	w. area	c1, c3, c5, c8	c2, c4, c6, c7	t. changer	shop
spawn rate (cases 1 & 3)	7	-	-	-	-	-
mean, st.d. ratio (cases 2 & 4)	0.5, 0.8	-	-	-	-	-
waiting time	-	-	40	60	-	50
waiting time st.d. (cases 3 & 4)	-	-	8	12	-	10
speed adjustment factor	-	-	0.001	0.1	-	0.1
max nr. agents	-	1	1	1	-	-
probability to change targets	-	-	-	-	0.1	-

parameters for the spawn rate given with *mean, st.d.* refer to the time point at the given fraction of the total simulation time. Here, the peak of the distribution is at halfway of the total time, and the standard deviation is set to 80 % of the total time, resulting in a wide distribution curve. Smaller values for the standard deviation were decided against, since they resulted in large queues which overlapped considerably with the source area itself and were therefore regarded as unrealistic. The total number of agents in each simulation run was 120, each agent with a radius of 0.25 m.

For the waiting time parameters, the fixed value in cases 1 and 2 was used as the mean value for the normal distribution in cases 3 and 4, and the standard deviation was chosen to be a fifth of the mean waiting time for the area in question, in order to account for more variability among individual agents.

The resulting spatio-temporal trajectory data was the only input data for development of the process extraction methods. The available data included an agent identification number, the X and Y coordinates of each agent, and the time stamps when these coordinates were taken, the rate being 5 frames per second.

4 RESULTS

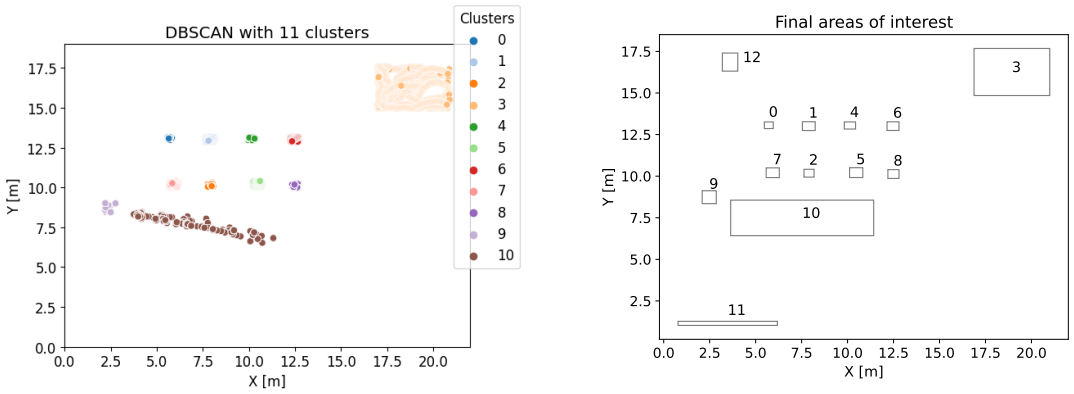
The analysis was run for the resulting data sets from the 4 simulation scenarios described in Section 2.1. For demonstration purposes, we first show the results for the first case (periodic spawn rate and fixed waiting time at counters) in detail, before discussing the results for all cases and validating with the original setup.

Firstly, a single source and a single sink were identified. Next, the SPD algorithm was applied, resulting in 14 741 discovered SPs. The DBSCAN algorithm then identified 11 clusters (Figure 3a), with the final padded areas shown in Figure 3b. Out of the 13 original areas, only the target changer was not recognized, since our approach focused on areas where agent velocity is reduced, which was not a property of this area.

To compare the sizes and location of the extracted areas and the areas from the original setup, the intersection-over-union (IOU) values were calculated for each pair of overlapping areas. The results are shown in Figure 4. The counters, waiting area, and particularly the shop, showed high overlap with the original areas, ranging from 0.57 to 0.95. The source and sink, however, show only localized overlap of under 0.4.

The decision tree shown in Figure 2 used to determine the type of each SP area tests several criteria, for which some properties such as the agent flow between areas were first needed. The corresponding directed graph is shown in Figure 5. Areas 0, 1, 2, 4, 5, 6, 7 and 8 were found to be parallel processes, as well as area 3 and the sink. Using this information, the results from the decision tree are as follows:

- Areas 0, 1, 2, 4, 5, 6, 7, 8 were classified as *Counter* areas.
- Area 3 was classified as *Shop*.
- Area 10 was classified as *Queue*.
- Area 9 was classified as *Waiting Area*.



(a) (b)

Figure 3: Results from the DBSCAN clustering of stay points (a) for case 1 and final areas determined by the clusters, including the source and sink (b).

Considering also that the source and sink were correctly matched to the source and sink in the original setup, 12 out of the 12 areas that matched were correctly classified.

The results for the extracted properties for this case, and all case scenarios previously described in Section 3, are shown in Table 2. The table also includes the parameters from the original setup (Table 1) for comparison of matching areas. As previously mentioned, the queue is not predefined in the original setup and was only included here for comparison among the different cases. Table 2 also shows the type outcomes from all data sets; notably, the results for counters with the same properties are summarized. Regarding the waiting time, the values for counters with the same properties were averaged out. The deviation from the original values is also given in percentage. The results for the counters show a maximum percentage difference of 3.875 % from the original waiting time, and a maximum negative 1.66 % difference in the standard deviation. For the shop, the maximum percentage difference is 9.4 % in waiting time mean, and 10 % in the standard deviation.

The speed adjustment factor was also overestimated for the shop, up to 20 %. The same parameter is underestimated for the counters, ranging from 17 % to 40 %. For the spawn rate estimation at the source, the periodic rate for cases 1 and 3 was correctly reconstructed. For the cases where the spawn distribution was normal (cases 2 and 4), the mean was also reconstructed within an error margin of 2 %. However, the standard deviation ratio was found to deviate notably from the original value, with a value of 1.15.

5 DISCUSSION

The results showed that all relevant areas from the original setup could be discovered in all four cases, with the counters, shop and waiting area showing high positional overlap with the original areas. The queue forming behind the waiting area, which was an important functionality of the original pedestrian movement scenario, could also be classified with a high success rate.

The IOU results showed only a small overlap for the source in the middle. This reflects the original design, where agents can only get generated at points where the whole agent fits inside the source. The sink was also found to overlap only partly, on the right side, since this is the closest point of possible entrance. In cases 2, 3 and 4, we see a notable increase of the sink IOU values, since the normal distribution at the spawn rate and/or the waiting time at counters causes many agents to arrive at the same time. This leads to agents being compelled to enter the sink at different angles, thereby covering a greater area compared to case 1.

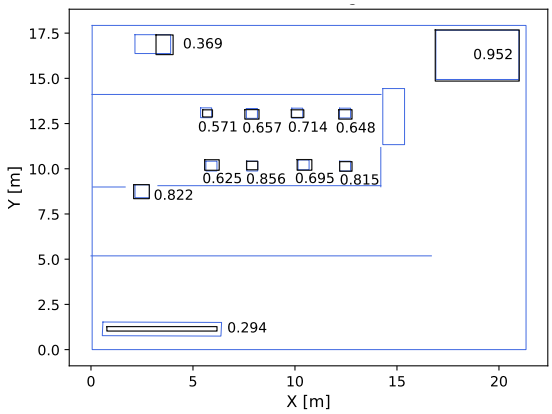


Figure 4: IOU values for the original areas (blue) and extracted areas (black) for case 1.

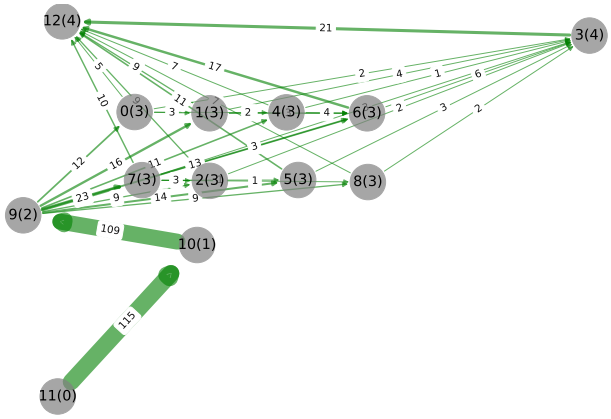


Figure 5: Graph showing agent flow between the SP areas for case 1, with the area numbers matching the cluster labels from Figure 3b. Nodes represent the areas and edges show the nr. of agents going into the next area. The nr. in brackets in each node label shows the shortest distance from the source node ('depth').

Regarding the classification of the areas, the types *Counter* and *Shop* were recognized with a 100 % accuracy for all cases, unlike *Waiting Area* and *Queue*. We found that the waiting area was typecasted as a queue when it was merged with the queue as a single area. This result is not necessarily problematic, since the existence of the waiting area is only representative of the first agent in the queue, and discovering the queue itself was of greater importance. In some cases where few agents had to wait, the queue was wrongly classified as a shop. The waiting times were then approximated to follow a normal distribution, characteristic to that of a shop. This result implies a bias of the methods toward larger rather than smaller queues.

The reconstruction of the properties showed a considerable overestimation of the mean waiting time for the shop. The reason is that we consider the total waiting time inside the area, whereas in the simulation, the value refers to the time stamp when the process ends and the agent is instructed to leave. The results for the shop further support this conclusion, since this area is much larger, so the agents take longer to cross it after the process ends. For the same reason, the speed adjustment factor for the shop was overestimated, as it was calculated for the entire time the agent spent there. The spawn rate property also showed overestimation of the standard deviation, which was due to the distribution already being wide in the original setup.

Table 2: Comparison of area properties for the original setup vs. extracted from all case scenarios. The counter numbers match cluster numbers from Figure 3b. The waiting time results are given in seconds, also including the difference from the original time in %.

		original	case 1	case 2	case 3	case 4
Counters 0, 2, 4, 8	type	Counter	Counter	Counter (20/20)	Counter (20/20)	Counter (20/20)
	IOU	-	0.74	0.643	0.732	0.726
	w. time	60	61.173 (+2 %)	61.1 (+1.83 %)	60.84 (+1.4 %)	61.6 (+2.66 %)
	w. time st. d.	12	-	-	11.8 (-1.66 %)	12.03 (+0.25 %)
	speed factor	0.001	0.0007 (-30 %)	0.00075 (-25 %)	0.00075 (-25 %)	0.00083 (-17 %)
	max nr. agents	1	1	1	1	1
Counters 7, 1, 5, 6	type	Counter	Counter	Counter (20/20)	Counter (20/20)	Counter (20/20)
	IOU	-	0.66	0.732	0.687	0.684
	w. time	40	41.164 (+3 %)	41.4 (+3.5 %)	41.55 (+3.875 %)	41.29 (+3.225 %)
	w. time st. d.	8	-	-	8.02 (+0.25 %)	7.295 (-0.94 %)
	speed factor	0.1	0.063 (-37 %)	0.07 (-30 %)	0.06 (-40 %)	0.062 (+38 %)
	max nr. agents	1	1	1	1	1
Shop	type	Shop	Shop	Shop (20/20)	Shop (20/20)	Shop (20/20)
	IOU	-	0.952	0.934	0.95	0.95
	w. time	50	54 (+8 %)	54.9 (+9.8 %)	54 (+8 %)	54.7 (+9.4 %)
	w. time st. d.	10	-	-	10	11 (+10 %)
	speed factor	0.1	0.12 (+20 %)	0.118 (+18 %)	0.119 (+19 %)	0.119 (+19 %)
Waiting Area	type	W. Area	W. Area	W. Area(17/20) | Queue(3/20)	W. Area(19/20) | Queue(1/20)	W. Area(15/20) | Queue(3/20) | Shop(1/20) | Counter(1/20)
	IOU	-	0.817	0.64	0.69	0.59
	max nr. agents	1	1	1	1	1
Queue	type	-	Queue	Queue (20/20)	Queue(18/20) | Shop(2/20)	Queue(19/20) | Shop(1/20)
Source	type	Source	Source	Source (20/20)	Source (20/20)	Source (20/20)
	IOU	-	0.3	0.264	0.294	0.264
	spawn rate	7	7	-	7	-
	mean ratio	0.5	-	0.49 (-2 %)	-	0.49 (-2 %)
	st.d. ratio	0.8	-	1.15 (+43.75 %)	-	1.15 (+43.75 %)
Sink	type	Sink	Sink	Sink (20/20)	Sink (20/20)	Sink (20/20)
	IOU	-	0.37	0.55	0.56	0.6

There exist several other general limitations of our methods, the key being the use of synthetic data. Some parameters were chosen depending on the current geometry setup, such as the distance and time threshold for the SPD algorithm. The clustering parameters for DBSCAN were adjusted based on the current data, in addition to following recommendations from literature. Therefore, they might not be appropriate for geometry layouts that differ greatly from the current one. In addition, using synthetic data means that the criteria which classified the areas were guided by properties of only the types present here, and by the subsequent behavior of the agents in these areas. The current approach therefore cannot discern all process types that might exist in real data sets. Furthermore, the current methods only capture a Gaussian distribution of both the waiting time and the spawn rate, whereas pedestrian movement in real life may follow other types of distributions based on the context. However, discovery of other distributions can be easily integrated.

To address the described limitations, it is necessary to test the developed methods with different data sets and scenarios with a different geometry, and subsequently make improvements depending on the patterns that emerge. For example, clustering parameters can be set automatically, based on geometry characteristics such as average distance between physical counters or obstacles. The criteria for area type classification may also be updated according to characteristics of the new data, and different distributions for properties such as waiting time and spawn rate could also be analyzed and integrated into the methods.

6 CONCLUSION AND OUTLOOK

This work builds upon previous research on identifying points of interest solely from a spatio-temporal trajectory, by detection and subsequent clustering of stay points. In addition, the areas were classified according to the type of process, going beyond the work in [13], which could only distinguish activity vs. non-activity areas. Furthermore, whereas previous research in POI detection focused on applications for transportation demand and urban planning, here we develop a framework for POI and process extraction specifically for the purposes of pedestrian behavior models. The methods from this study could potentially be used to inform data-driven pedestrian simulation framework such as the one described in [10].

The extraction of areas of interest and reconstruction of their properties was successfully performed for a specific pedestrian movement scenario. The synthetic data used comprised of spatio-temporal stamps for each pedestrian. The methods were tested for four different case scenarios, distinguished by a parameter change regarding the distribution of agents entering the simulation, or the distribution of waiting time at counters. In all cases, the original areas and their types were reliably discovered, importantly including the functionality of a waiting area as a dispatching system, and the formation of a queue behind this area. These functionalities are also likely to exist in realistic scenarios of our desired application domain, namely processes in a railway station or an airport. In that case, all processes might not match our predefined types; however, our model would be able to reveal possible unexpected congestion areas, as well as reconstruct important properties describing these areas. Nevertheless, given the restrictions of the synthetic data and setup, our methods must be tested and subsequently improved by using real data such as history logs from these infrastructures. As a result, we should be able to characterize additional processes and incorporate them in our methods. Ultimately, our framework will be used to develop a simulation model which can faithfully reproduce the original trajectories showing complex behavior of pedestrians.

REFERENCES

[1] M. Ouyang, "Review on modeling and simulation of interdependent critical infrastructure systems," *Reliability Engineering I& System Safety*, vol. 121, pp. 43–60, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832013002056

[2] E. Brucherseifer, H. Winter, A. Mentges, M. Mühlhäuser, and M. Hellmann, "Digital twin conceptual framework for improving critical infrastructure resilience," *at - Automatisierungstechnik*, vol. 69, no. 12, pp. 1062–1080, 2021.

[3] N. Keller and X. Hu, "Towards data-driven simulation modeling for mobile agent-based systems," *ACM Trans. Model. Comput. Simul.*, vol. 29, no. 1, feb 2019.

[4] *Using Model Pipelines to Simulate the Processes in and around an Airport through a Web-Interface*, ser. GCMS '13. Vista, CA: Society for Modeling I& Simulation International, 2013.

[5] M. T. Somayeh Dodge, Song Gao and R. Weibel, "Progress in computational movement analysis – towards movement data science," *International Journal of Geographical Information Science*, vol. 34, no. 12, pp. 2395–2400, 2020. [Online]. Available: https://doi.org/10.1080/13658816.2020.1784425

[6] Y. Ma, E. W. Lee, Z. Hu, M. Shi, and R. K. Yuen, "An intelligence-based approach for prediction of microscopic pedestrian walking behavior," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3964–3980, 2019.

[7] W. Wang, J. Rong, Q. Fan, J. Zhang, X. Han, and B. Cong, "Data-driven simulation of pedestrian movement with artificial neural network," *Journal of Advanced Transportation*, vol. 2021, pp. 1–16, 08 2021.

[8] X. Wei, W. Lu, L. Zhu, and W. Xing, "Learning motion rules from real data: Neural network for crowd simulation," *Neurocomputing*, vol. 310, pp. 125–134, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218305599

[9] K. H. Lee, M. G. Choi, Q. Hong, and J. Lee, "Group behavior from video: a data-driven approach to crowd simulation," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '07. Goslar, DEU: Eurographics Association, 2007, p. 109–118.

[10] A. K.-F. Lui, Y.-H. Chan, and M.-F. Leung, "Modelling of destinations for data-driven pedestrian trajectory prediction in public buildings," in *2021 IEEE International Conference on Big Data (Big Data)*, 2021, pp. 1709–1717.

[11] A. Karite, D. B. Ahmed, and E. M. Diaz, "Points of interest identification: A case study in beijing metropolitan area," in *2022 IEEE International Smart Cities Conference (ISC2)*, 2022, pp. 1–7.

[12] M. Piekenbrock and D. Doran, "Intrinsic point of interest discovery from trajectory data," *ArXiv*, vol. abs/1712.05247, 2017.

[13] L. Gong, T. Yamamoto, and T. Morikawa, "Identification of activity stop locations in gps trajectories by dbscan-te method combined with support vector machines," *Transportation Research Procedia*, vol. 32, pp. 146–154, 2018, transport Survey Methods in the era of big data:facing the challenges. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146518301820

[14] M. J. Seitz and G. Köster, "Natural discretization of pedestrian movement in continuous space," *Phys. Rev. E*, vol. 86, p. 046108, Oct 2012.

[15] Y. Wang, M. Kyriakidis, and V. N. Dang, "Incorporating human factors in emergency evacuation – an overview of behavioral factors and models," *International Journal of Disaster Risk Reduction*, vol. 60, p. 102254, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S221242092100220X

[16] M. Schultz, "Entwicklung eines individuenbasierten modells zur abbildung des bewegungsverhaltens von passagieren im flughafenterminal," Ph.D. dissertation, 01 2010.

[17] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "Dbscan revisited, revisited: Why and how you should (still) use dbscan," *ACM Trans. Database Syst.*, vol. 42, no. 3, jul 2017.

[18] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm gdbscan and its applications," *Data Mining and Knowledge Discovery*, vol. 2, pp. 169–194, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:445002

[19] *Mining User Similarity Based on Location History*, ser. GIS '08. New York, NY, USA: Association for Computing Machinery, 2008. [Online]. Available: https://doi.org/10.1145/1463434.1463477

[20] (2023) Stationarity and detrending (adf/kpss). https://www.statsmodels.org/dev/examples/notebooks/generated/stationarity_detrending_adf_kpss.html. Accessed: 2023-20-12.

[21] A. Shojaie and E. B. Fox, "Granger causality: A review and recent advances," *Annual Review of Statistics and Its Application*, vol. 9, pp. 289–319, 2022. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/37840549/

AUTHOR BIOGRAPHIES

ELENA EFTIMOVA is a research associate in the Institute for the Protection of Terrestrial Infrastructures. Her research interests lie in data analysis and automated process modeling, as a contribution to the development of Digital Twins. Contact: elena.eftimova@dlr.de.

CHRISTOPH NELLINGER is a research associate in the Institute for the Protection of Terrestrial Infrastructures. His research interests lie in agent-based simulations, particularly the development of pedestrian movement models for Digital Twins. Contact: christoph.nellinger@dlr.de.

TOBIAS KOCH is a research group leader at the Department for Digital Twins for Infrastructures in the Institute for the Protection of Terrestrial Infrastructures. His research focus is the generation and operation of Digital Twins as a decision support and crisis management tool. Contact: tobias.koch@dlr.de.