A Digital Twin Approach for Crowd Flow Modeling on Railway Station Platforms

Yu Yasuda
The University of Osaka
Suita, Japan
y-yasuda@ist.osaka-u.ac.jp

Tatsuya Amano
The University of Osaka Suita, Japan
RIKEN R-CCS, Kobe, Japan
t-amano@ist.osaka-u.ac.jp

Hirozumi Yamaguchi
The University of Osaka Suita, Japan
RIKEN R-CCS, Kobe, Japan
h-yamagu@ist.osaka-u.ac.jp

***Abstract*—Effective crowd tracking at railway station platforms is essential for ensuring passenger safety and optimizing pedestrian flow, particularly in high-density urban transit hubs. However, traditional tracking methods, such as object detection and multi-object tracking, face limitations in congested environments due to severe occlusions and overlapping individuals. This paper proposes a novel approach for modeling pedestrian flow on train station platforms by coupling deep learning-based motion analysis with crowd simulation. In the proposed method, we utilize RAFT, a state-of-the-art optical flow model, to extract pixel-level motion vectors, which are clustered to identify human movement patterns. These motion data are mapped onto a calibrated 2D platform model, providing a top-down representation of pedestrian trajectories. To simulate realistic crowd dynamics, Unity's NavMesh is employed alongside an enhanced Simulated Annealing approach to generate high-accuracy origin-destination (OD) data. This is a new digital twin concept where the analysis from the vision in the real world is projected onto the virtual world model to simulate and reproduce the pedestrian flows. The proposed method was evaluated using synthetic crowd simulation data, demonstrating high accuracy in destination estimation. The experimental results indicate that the OD estimation outperforms conventional approaches, with error rates reduced to half of those observed in YOLOv8x-based tracking systems. These findings suggest that the integration of optical flow-based motion analysis with digital twin simulation can significantly enhance crowd monitoring and congestion management in railway stations.**

***Index Terms*—Crowd Flow Digital Twin Optical Flow Estimation Unity Crowd Simulation Simulated Annealing**

I. Introduction

In Japan, railways are one of the primary means of transportation, particularly in major metropolitan areas such as the capital region, where they are indispensable for commuting, school attendance, and other travel needs. Consequently, train station platforms become congested during morning and evening rush hours, as well as during large-scale events, necessitating effective crowd control and safety measures. To alleviate congestion and prevent accidents, it is crucial to analyze crowd movement patterns in detail, including passenger flow dynamics and the timing of boarding and alighting.

In recent years, the demand for crowd flow analysis has increased, leading to extensive research in the field. Since around 2015, Japanese railway companies have utilized station cameras, infrared sensors, and distance-measuring sensors to monitor congestion levels and provide congestion information through official apps. However, higher-precision technology is required to capture detailed real-time crowd movements.

Crowd flow analysis is primarily conducted using video-based and sensor-based methods. Sensor-based methods utilize three-dimensional LiDAR sensors and depth cameras to obtain point cloud data and detect individuals [1]. However, these methods face challenges such as missing point cloud data, occlusion, and the influence of weather conditions, which can reduce accuracy. On the other hand, video-based methods leverage object detection models such as YOLO and density estimation models, with the latest YOLOv8 achieving high-precision object detection [2]. Nevertheless, in densely packed crowds, distinguishing individual features is difficult, and distant areas in images are harder to capture. As a result, only rough directional crowd flow can be analyzed, and specific phenomena—such as people walking along the edges of platforms to avoid congestion or those crowding escalators—cannot be accurately represented.

To address these issues, this study proposes a method for constructing a crowd digital twin that replicates crowd movement in a three-dimensional virtual space using motion images from train station platforms. The proposed method extracts moving object regions using optical flow estimation and identifies people moving in the same direction through pixel-level clustering. Additionally, the detected moving object regions' pixel coordinates are mapped onto a three-dimensional spatial simulator (Unity) to estimate crowd occupancy areas and movement directions. In Unity, human models and their navigation (movement from one point to another) are generated, and their positions, occurrence frequencies, and destinations are fitted to the obtained movement data, allowing for the generation of individual movement trajectories that align with observed data. This is a new digital twin concept where the analysis from the vision in the real world is projected onto the virtual world model to simulate and reproduce the pedestrian flows.

To assess the effectiveness of the proposed method, we evaluated the accuracy of OD estimation and crowd size estimation using data generated from a crowd simulation model. The evaluation results demonstrated that the destination (D) estimation achieved high accuracy, while the error in crowd size estimation was reduced to half of that observed with YOLOv8x, a conventional object detection model.

The remainder of this paper is structured as follows: Section 2 discusses related research and positions this study within the existing literature. Section 3 details the proposed method, and Section 4 presents experiments and evaluations and finally, Section 5 concludes the study.

II. Related Work

A. Computer Vision-Based Methods for Crowd Analysis

Early research on crowd analysis tackled the challenge of detecting and tracking individuals in crowded video scenes with classical computer vision techniques. Ali and Shah proposed a pioneering algorithm for tracking people in high-density crowds by introducing a scene force model (floor fields) to handle severe occlusions and density [3]. Rodriguez et al. leveraged global scene context and crowd density information to jointly optimize person detection and localization, significantly improving the accuracy of detecting and tracking individuals in crowds [4].

With the rise of deep learning, person detection and multi-object tracking (MOT) in crowds have seen substantial progress. The tracking-by-detection approach, wherein a detector identifies individuals in each frame and a tracker links them over time, has become dominant [5], [6]. CNN/Vision Transformer based detectors (e.g. Faster R-CNN [7] and YOLO [2]) improved human detection even under crowd occlusions, especially when augmented with crowd-specific data [8].

Another line of research employs optical flow to analyze crowd dynamics without explicit object detection. Optical flow methods enable measurement of motion patterns for hundreds of people, providing a representation of crowd movement that circumvents the need to individually detect or track each person [9]. For instance, Mehran et al. used optical flow combined with a social force model to detect abnormal crowd events, showing that the social-force approach captured crowd behavior better than pure flow alone [10].

Modern deep learning has dramatically advanced optical flow estimation. RAFT (Recurrent All-Pairs Field Transforms) [11] is the recent state-of-the-art network that iteratively refines all-pairs correlations, achieving significantly lower error rates than prior methods on standard benchmarks. Such high-precision flow techniques (e.g. RAFT) are promising for crowd analysis, as they can capture fine-grained motion in dense scenes to aid in segmentation of crowd movements or anomaly detection.

B. Pedestrian Flow Modeling

Research on pedestrian flow modeling has evolved from two perspectives: sensing technologies and modeling approaches. In sensing, the advent of devices such as 3D LiDAR sensors has enabled higher-granularity pedestrian flow observations compared to traditional Wi-Fi or beacon-based methods [1]. As a result, it has become more feasible to collect detailed trajectory data for individual pedestrians, thereby facilitating more fine-grained model development and validation. Modeling approaches can be broadly categorized into two types. The first is a physics-based approach exemplified by the Social Force Model (SFM) [12], originally proposed by Helbing et al. SFM views crowd movement through the lens of "social forces" that govern individuals' motivations and collision avoidance behaviors. This model has been widely adopted for pedestrian simulations due to its interpretability and computational efficiency. An early extension is Pellegrini et al.'s work [13], which introduced a social interaction model to improve multi-target tracking accuracy.

On the other hand, machine learning–based approaches such as SocialLSTM [14], SGCN [15], and AgentFormer [16] directly learn interpersonal interactions from trajectory data. While these methods excel at capturing complex behavioral patterns, they often face challenges such as the absence of explicit physical constraints and the need for large-scale training datasets.

III. Proposed method

A. Method Overview

The overview of the proposed approach in this paper is illustrated in Figure 1. In the proposed method, a fixed camera positioned at a certain height captures video footage of pedestrian flow on a train station platform, and this footage serves as the input to the system. For each consecutive frame pair in the video, a state-of-the-art deep neural network (DNN)-based approach is applied to generate pixel-level optical flow. Specifically, we utilize RAFT [11]. RAFT generates a vector for each pixel in a two-dimensional image, indicating the direction in which that pixel moves in the next frame.

The obtained optical flow provides highly granular motion information but also includes a significant amount of noise. To extract meaningful movement patterns caused by human motion, pixels exhibiting similar directional movement are clustered. This enables the approximate identification of both the presence and movement direction of humans within the frame.

Simultaneously, the proposed method pre-generates a two-dimensional planar model P of the train station platform using 3D modeling software such as Unity. The 2D platform model primarily needs to accurately reproduce the positions of obstacles such as benches and trash bins, the locations of entrances and exits (including the positions of train doors when a train is stopped), and the dimensions of the platform in the x and y directions. Through prior calibration, we assume that the region of the camera image (denoted as C) corresponding to the platform area P can be identified. Given the camera's installation angle information, if a moving pixel within image region C corresponds to a human, we can estimate the height of the human and apply a projection transformation to determine the coordinates of that human's presence and movement direction within platform region P. This transformation allows us to map the movement information from the image to a top-down two-dimensional representation of the platform.

Finally, the proposed method utilizes Unity's NavMesh function to simulate pedestrian flow within a virtual space that includes platform P. The pedestrian flow is adjusted to

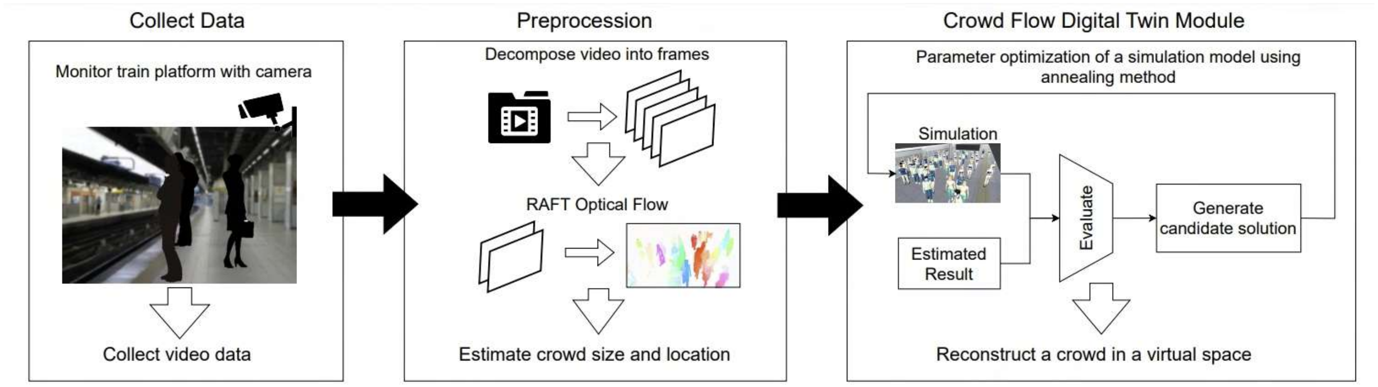
Collect Data
Monitor train platform with camera
Collect video data
Preprocession
Decompose video into frames
RAFT Optical Flow
Estimate crowd size and location
Crowd Flow Digital Twin Module
Parameter optimization of a simulation model using annealing method
Simulation
Estimated Result
Evaluate
Generate candidate solution
Reconstruct a crowd in a virtual space

Fig. 1: Proposed Method Overview

fit the movement direction data obtained from the image. A key challenge here is that while the movement information extracted from the image is micro-scale, pedestrian flow simulation typically requires macro-scale origin-destination (OD) data that indicates where people move to and from within the platform. To address this, the proposed method incorporates an improved version of Simulated Annealing, allowing for the natural appearance and disappearance of individuals while ensuring that both the observed density of people and movement directions are well-fitted. By iteratively refining this process, the virtual space can realistically reproduce the emergence and disappearance of people, thereby generating pedestrian flows that accurately reflect observed human movement and presence.

B. Proposed Method

In the proposed method, we construct a digital twin of pedestrian flows that reproduces the presence and movement of crowds in a virtual space by optimizing the parameters of a simulation model using observation data obtained by estimating the number and positions of individuals at regular time intervals via optical flow estimation on crowd-captured videos.

One key idea of the proposed method is to perform pixel-level analysis—the smallest unit of an image—using optical flow estimation. In crowded situations, the boundaries between individuals become ambiguous, and for people located far from the camera, the available information is limited, making it challenging to detect each person individually and resulting in significant errors in count estimation. By detecting dynamic regions (areas where crowds are present) through pixel-level image analysis, we can accurately estimate the number of individuals in the image and also predict each person's position and movement direction.

Another key idea is to use the estimated positions and movement directions of each individual in the crowd as observation data to optimize the simulation model parameters. By aligning the simulation model with the observation data while preserving the collective behavior of the crowd, the model can reproduce the movement dynamics of the crowd, thereby improving the estimation accuracy of both the number and positions of individuals.

1) Estimate crowd size and location: In the proposed method, we estimate the position and movement direction of each individual in a crowd by employing optical flow and Mean Shift clustering from video frames as a preprocessing step. The process is illustrated in Figure 2. First, groups of pixels corresponding to crowds are extracted from the video using optical flow and Mean Shift, and these pixel clusters are mapped onto a station platform recreated in Unity's virtual space to identify the contact area with the floor. By placing person models onto these contact areas, it becomes possible to ultimately estimate the number of individuals, their positions, and their movement directions within the virtual environment.

To extract the overall crowd from the image, the movement of each pixel between consecutive video frames is estimated. We use RAFT [11]—a high-accuracy optical flow model based on deep learning—to obtain the movement vectors (u(x, y), v(x, y)) for each pixel. This allows us to capture local dynamic movement patterns in the video at high resolution. Although the obtained optical flow represents a vector field indicating movement information for each pixel, it is challenging to comprehend the overall trend of the crowd at the individual pixel level. Therefore, our system employs Mean Shift clustering to aggregate pixels into natural clusters based on the similarity of their motion. Through Mean Shift, groups of pixels with similar movement are extracted as distinct "crowds," which are then mapped to a three-dimensional space by the 3D Mapping Module.

Within the 3D Mapping Module, the two-dimensional pixel coordinates are projected onto coordinates on the station platform recreated in Unity's virtual space. Specifically, the pixel clusters corresponding to the extracted "crowds" are treated as three-dimensional objects with an assumed height of 1.7m (this can be configured depending on the regions), and the platform contact area is determined by identifying where the lower part of these objects touches the floor, using a viewpoint consistent with that of the original video.

Subsequently, person models are arranged within the contact areas without overlapping. The estimated movement direction

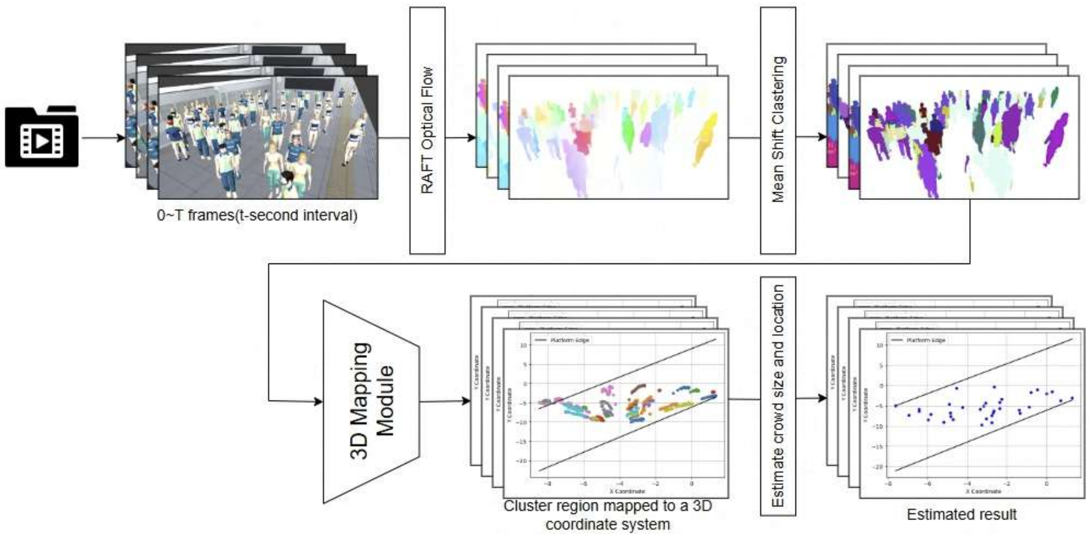
0~T frames(t-second interval)
RAFT Optical Flow
Mean Shift Clastering
3D Mapping Module
Cluster region mapped to a 3D coordinate system
Estimate crowd size and location
Estimated result
Platform Edge
X Coordinate
Y Coordinate

Fig. 2: Estimate crowd size and location

of the corresponding pixels is used to determine the movement direction of each person model. Specifically, the movement vectors extracted from each cluster in the image are mapped into the virtual space, and this mapped direction is set as the forward direction for the person models, thereby ensuring that the computed movement directions correspond with the actual motion observed. This process enables the estimation of not only the number and position of individuals in the crowd but also their respective movement directions.

2) Crowd Simulation Model: We achieve a more realistic and dynamic crowd simulation by modeling state transitions—such as congestion, crowd flow, train arrivals and departures, and boarding and alighting—based on the basic functionality of Unity's NavMesh. Unity's NavMesh is a precomputed mesh that represents navigable areas within a scene, enabling agents to perform pathfinding and obstacle avoidance toward a destination. However, because the NavMesh pathfinding algorithm computes the shortest path to a destination, it fails to replicate the congestion-aware movement observed in real crowds. To address this, we enhanced the cost function of the A* algorithm by incorporating a congestion cost, thereby enabling the calculation of paths that avoid congested areas. Additionally, as the movement of crowds on actual station platforms varies with train arrivals, departures, boarding, and alighting, we refined the model to incorporate state transitions corresponding to different station conditions.

In our simulation model, the congestion cost is calculated to account for the crowd density and the alignment of the agents' movement. It is used to adjust the congestion cost of traversing a cell in the navigation grid. The evaluation involves three main components: the alignment of the movement direction with the average direction in a cell, the average velocity of agents (representing the flow strength), and the number of agents in the cell. Let $\mathbf{d}_m$ denote the normalized movement direction of an individual agent, and let $\mathbf{d}_a$ represent the normalized average movement direction of agents within a given cell. The alignment between these vectors is computed as

$$a = \mathbf{d}_m \cdot \mathbf{d}_a,$$

which naturally varies from -1 (indicating complete opposition) to 1 (indicating perfect alignment). To map this value to a unit interval, we define α as $(a+1)/2$. Subsequently, a direction factor F_d is obtained via linear interpolation between a high penalty P_H (applied when the alignment is poor) and a low penalty P_L (applied when the alignment is favorable):

$$F_d = (1-\alpha)P_H + \alpha P_L.$$

Let v denote the magnitude of the average velocity in the cell, which serves as an indicator of flow strength. This value is bounded within the range $[V_{\min}, V_{\max}]$ by the clamping function:

$$F_v = \mathrm{Clamp}(v,\, V_{\min},\, V_{\max}).$$

If N represents the number of agents present in the cell, the overall congestion penalty C is computed by scaling the product of the number of agents, the velocity factor, and the direction factor by a constant k, and subsequently clamping the result between $C_{\min}$ and $C_{\max}$:

$$C = \mathrm{Clamp}\left(k\, N\, F_v\, F_d,\, C_{\min},\, C_{\max}\right).$$

Here, $\mathrm{Clamp}(x,\, a,\, b)$ is a function that restricts x to the interval $[a, b]$. This formulation enables the model to modulate the cost of traversing a cell based not solely on the Euclidean distance but also on the local crowd density and coherence of movement, thus enhancing the fidelity of simulated navigation in congested environments.

Furthermore, in this simulation model, the dynamics of a train station are represented by discrete state transitions that govern the overall behavior of agents during different phases of station operation. Prior to the train's arrival (i.e., when

$t < t_{\text{arrival}}$), agents remain in the *Waiting* state, during which they form queues in the vicinity of the station. Immediately following the arrival of the train, the system transitions to the *Unloading* state for the interval $t_{\text{arrival}} \le t < t_{\text{arrival}} + \Delta t_{\text{unload}}$, during which agents disembark from the train. Subsequently, the system enters the *Boarding* state for $t_{\text{arrival}} + \Delta t_{\text{unload}} \le t$, during which boarding takes place.

These state transitions can be formally expressed as follows:

$$\text{State}(t) = \begin{cases} \text{Waiting}, & t < t_{\text{arrival}}, \\ \text{Unloading}, & t_{\text{arrival}} \le t < t_{\text{arrival}} + \Delta t_{\text{unload}}, \\ \text{Boarding}, & t_{\text{arrival}} + \Delta t_{\text{unload}} \le t \end{cases}$$

The focus of this model is on the overall state-dependent behavior rather than the micro-level details of individual agent interactions. This abstraction enables the simulation to replicate the macro dynamics of a train station, thereby providing insight into the effects of state transitions on overall crowd movement and station efficiency.

3) Anealing Method: In crowd simulation, it is necessary to determine a vast array of parameters such as at what time, from which area, to which destination, and at what speed each agent moves." However, due to the inherent structure of crowd movements—characterized by interactions and decision-making processes that can easily lead to local optima—a robust search algorithm is required. In the proposed method, we employ Simulated Annealing (SA) to probabilistically jump across large regions of the solution space while gradually converging toward an optimal solution.

In conventional simulated annealing, the typical approach is to iteratively perform "small perturbations" within the neighborhood. Yet, given that the parameter space involved in crowd simulation is of very high dimensionality, relying solely on random, minute changes can result in an unbounded diffusion of the search process. In other words, if the definition of the neighborhood is not sufficiently rigorous, candidate solutions may be too distant from each other, necessitating an exorbitant number of steps for convergence. To address this issue, we introduce a novel candidate solution generation method that incorporates directional neighborhood operations, thereby enabling more efficient exploration of the solution space.

Our simulated annealing algorithm adheres to the standard process of (1) solution initialization, (2) solution modification, and (3) acceptance determination, with the temperature gradually reduced over time. However, it distinguishes itself by employing a custom set of operation types specifically designed for two-tiered lists to generate neighboring solutions. Furthermore, by dynamically adjusting the selection probabilities of each operation type, the algorithm is able to converge more rapidly on an agent configuration that meets the desired objectives, even within an expansive solution space.

The agents used in the simulation are managed using a two-stage list format defined over *areas* (A, B, C, D) and *time steps* (0, 1, 2, ...). This list simultaneously records various parameters of the agents, such as their spawn position (*spawnPosition*), destination (*destination*), moving speed (*moveSpeed*), and generation time (*startTime*). As a result, the overall picture of "when, in which area, how many agents, and where they head" can be readily captured, enabling the entire configuration to be treated as a single candidate solution to be optimized via simulated annealing.

We compare the crowd distribution obtained from simulations with the observed data at each time t. The error function $E(\mathbf{x})$ is defined so that it measures the difference between simulated and observed results. We primarily use two types of indicators, aggregated over both cells (grid cells) and times:

First, we evaluate the *agent count error*:

$$E_{\text{count}}(t) = \sum_{\text{cell}} w_{\text{cell}} \big(n_{\text{sim}}(t) - n_{\text{obs}}(t)\big)^2,$$

where $n_{\text{sim}}(t)$ is the number of agents in the simulation within that cell at time t, $n_{\text{obs}}(t)$ is the observed number of agents, and w_{cell} is a weight based on factors such as cell position or camera distance.

Second, we consider the *directional error*:

$$E_{\text{dir}}(t) = \sum_{\text{cell}} \big(1 - \cos\theta_{\text{sim,obs}}\big),$$

where $\theta_{\text{sim,obs}}$ is the angle between the observed direction vector and the simulated direction vector in each cell at time t. If they coincide, $\cos\theta \approx 1$, so the error is small.

The final error function integrates these indicators with an appropriate weighting:

$$E(\mathbf{x}) = \sum_{t} \Big[E_{\text{count}}(t)(1 + \lambda\, E_{\text{dir}}(t))\Big],$$

where λ is a hyperparameter controlling the relative importance of the direction term.

When applying simulated annealing, we need to define how to modify the current solution $\mathbf{x}$ into a new solution $\mathbf{x}'$. In this program, we represent agent parameters in a two-stage list by area and time step, and a variety of operations (OperationType) are used to perturb $\mathbf{x}$.

- Add_Area: inserts agents in a chosen area
- Remove_Area: eliminates agents from a chosen area
- ChangeDest: changes agent destinations
- ChangeSpawn: shifts spawn positions
- MoveTimeStep: moves agents from time steps with a surplus to those that show a deficit

We can apply one or multiple such operations in a single iteration, for instance, only changing area A or C, or modifying multiple areas at once.

Moreover, this study introduces an independent score $\text{Score}(op)$ for each operation type op, updated according to how the error changes. If a newly generated solution $\mathbf{x}'$ has error $E(\mathbf{x}')$ and the old solution has error $E(\mathbf{x})$, then we define $\Delta E = E(\mathbf{x}') - E(\mathbf{x})$. If $\Delta E < 0$ (improved error), $\text{Score}(op)$ is increased; if $\Delta E > 0$ (worse), $\text{Score}(op)$ is decreased. These scores are also clipped to some range, for instance $[0.1, 10]$, to avoid excessive growth or shrinkage. In

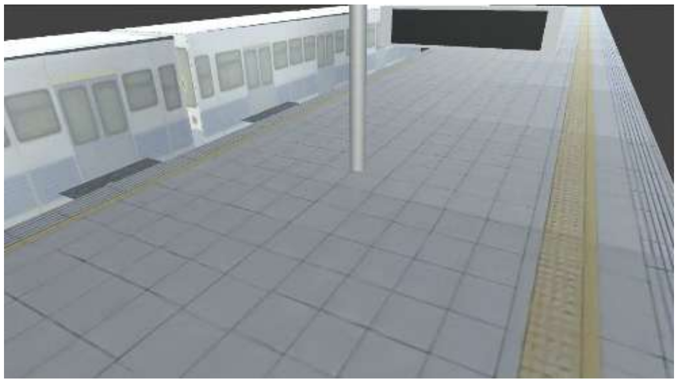

Fig. 3: FOV of the experimental data

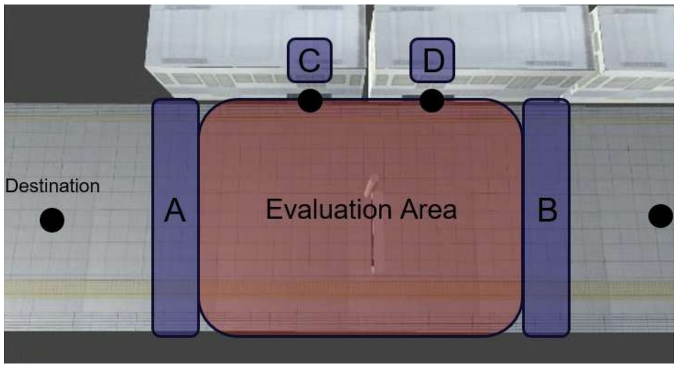

Fig. 4: Simulation Environment

each iteration of simulated annealing, we sample an operation type in proportion to

$$P(\text{choose } op) = \frac{\text{Score}(op)}{\sum_{op'} \text{Score}(op')}.$$

Consequently, operations that have contributed to larger improvements in the past are chosen more frequently, while those that caused greater error increases are less likely to be selected. This history-based score management allows the algorithm to refine its perturbations dynamically, making the search more efficient in exploring the solution space.

IV. Evaluation

A. Experimental Setup

To verify the effectiveness of the proposed method, we constructed a scenario within a crowd simulation environment based on observational data, reproducing the arrangement and dynamics of agents. Instead of using real footage, the simulation video footage generated in Unity based on the FOV shown in Figure 3 was used as the experimental dataset, from which the positions, movement directions and OD (Origin-Destination) data were obtained directly at regular intervals (every second). During the data generation process, the total number of agents in each area and the number of agents generated at each time step were intentionally set to be uneven, thereby simulating a realistic scenario for crowd simulation.

As shown in Figure 4, the simulation environment is implemented on a Unity 3D simulation platform, which was also used to generate the experimental data. Four distinct agent generation areas, labeled A, B, C, and D, are defined along with their corresponding destinations. At each frame interval of the observational data, agents are generated within these areas, and both their generation positions and destination assignments are determined. For agents generated in areas corresponding to either the designated generation areas or the train entry/exit points (specifically in areas C and D), conditions are set to ensure that if an agent originates from a train entry/exit, the corresponding destination is not selected.

The observational data comprise records of the estimated positions and movement directions of agents in each area at regular time intervals (one frame per second). These data are then used in the simulated annealing process to evaluate the errors in both the number of agents and their movement directions within each cell (grid) of the simulation. Furthermore, the evaluation area is defined based on the distribution range of the observational data, ensuring that the analysis closely reflects the real-world scenario being simulated.

Agent generation is managed in a two-tier list format based on areas and time instances, where spawn positions, destination assignments, movement speeds (with a nominal value of 3 m/s subject to a ±10% variation), and generation times are collectively specified. This representation clearly delineates the overall scenario—detailing when and in which area, how many agents are generated, and where they are headed—thereby allowing it to be treated as a single candidate solution within the simulated annealing framework. The initial total number of agents assigned to each area is set to 15, from which the initial candidate solution is generated randomly. Furthermore, the train arrival times are assumed to be provided by the experimental data.

The parameters for simulation optimization were set based on the following rationale and ranges. The initial temperature was set to 100.0 to enable broad exploration and avoid premature convergence by allowing non-improving changes early on. A maximum of 1,000 iterations and a cooling rate of 0.997 were used to balance exploration diversity with convergence speed and computation time. The initial score for all operation types (e.g., agent addition, deletion, destination changes, etc.) was uniformly set to 1.0 to prevent initial bias, with scores dynamically updated during the search.

B. Performance of OD Estimation

For the experimental evaluation, the OD data from the experiment were compared with the estimated OD data. Table I shows the average Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) calculated from the actual and estimated counts for each metric over four videos (Index 0 to Index 3). As seen from the results, for the estimation of destination (D), the average MAE is 5.69 and the average RMSE is 6.93 across the four videos simulating crowds of approximately 100 individuals, demonstrating a high level of accuracy. Furthermore, Figure 5 presents the count estimation

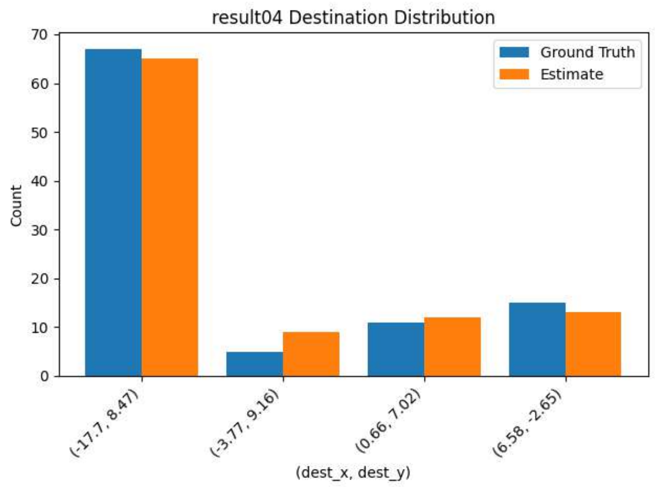

Fig. 5: Destination estimation for Index #3

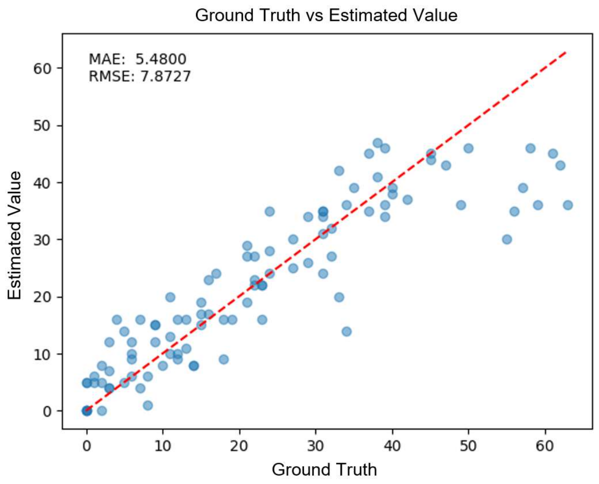

Fig. 6: Crowd size estimation for each frame (Proposed Method)

results for each destination (D) in Index 3, illustrating that the tendency of counts to be biased toward certain destinations is effectively reproduced. On the other hand, the estimation accuracy for the origin (O) is relatively lower, which is considered the primary cause of the larger overall OD estimation error. In the context of a station platform, the high accuracy in destination estimation enables the prediction of the number of individuals heading toward exits such as boarding gates or stairs. This, in turn, facilitates an estimation of the number of people boarding the train and those exiting the station, thereby allowing an assessment of congestion levels within the train and station premises.

TABLE I: Evaluation Metrics (Rounded to Two Decimals)

	OD		O		D	
Index #	MAE	RMSE	MAE	RMSE	MAE	RMSE
0	12.43	15.55	21.75	21.94	6.75	7.40
1	9.29	10.75	11.25	11.63	7.25	8.96
2	7.71	11.81	13.00	15.83	6.50	8.86
3	7.00	8.89	7.75	9.10	2.25	2.50
Average	9.11	11.75	13.44	14.62	5.69	6.93

C. Performance of Crowd Size Estimation

Figure 6 illustrates the results of the crowd count estimation. To evaluate the accuracy of the estimation, we computed the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE) between the actual and estimated crowd counts—metrics commonly used in crowd count evaluation. The results show that both metrics are below 10, indicating highly accurate estimations. Additionally, as a baseline, crowd count estimation was performed on images similar to Figure 7 using YOLOv8x, the highest-performing model among the YOLOv8 object detection methods. Figure 8 shows the results for the actual crowd image, while Figure 9 presents a scatter plot comparing the estimated and actual crowd counts. The MAE is 18.1951 and the RMSE is 19.2341, indicating larger errors compared to the proposed method.

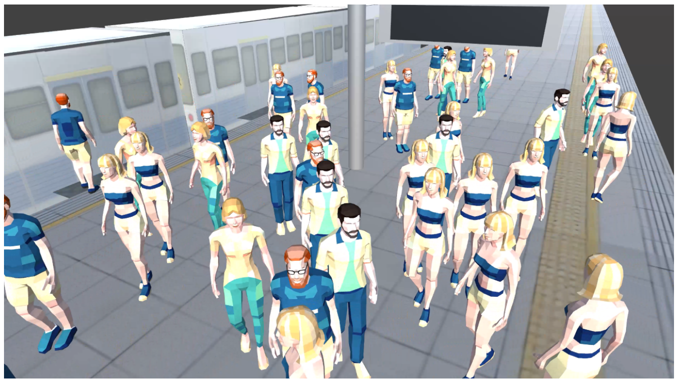

Fig. 7: Crowd image (Unity)

This demonstrates that our approach achieves higher accuracy even when compared with existing techniques. Although the estimated crowd count increases linearly with the actual count, when the number of people in an image exceeds 40, occlusion effects cause the estimation to be lower than the actual count, thereby leading to a reduction in accuracy.

Table II presents a comparison between the ground truth and

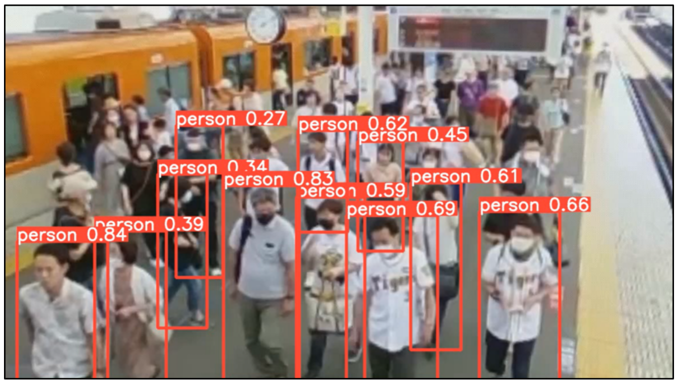

Fig. 8: Crowd image (YOLOv8x)

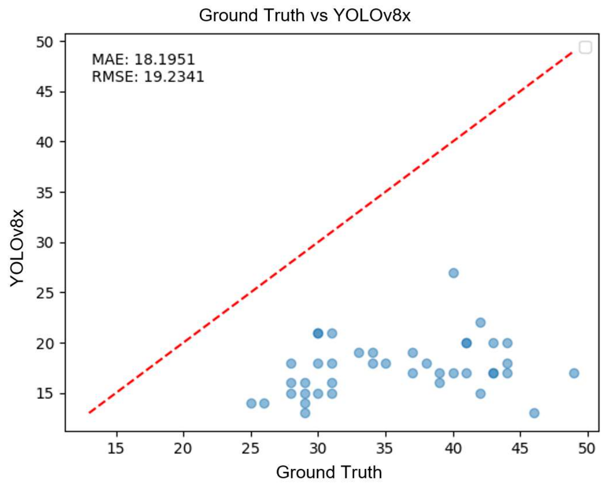

Fig. 9: Crowd size estimation for each frame (YOLOv8x)

the estimated total number of people in the video. For a crowd of approximately 100 individuals, the average error is 14.08%, demonstrating that the overall count is estimated with high accuracy. Given that both the total number of individuals in the video and the per-second frame counts are estimated with high precision, it can be concluded that temporally consistent crowd count estimation has been achieved.

TABLE II: Total True Count vs. Total Estimated Count

Index	Total True Count	Total Estimated Count	Difference (%)
0	95	106	11.58%
1	99	112	13.13%
2	85	111	30.59%
3	98	99	1.02%

V. Conclusion

This study introduced an innovative approach for modeling pedestrian flow on railway station platforms by integrating deep learning-based motion analysis with digital twin simulation. By leveraging RAFT for optical flow estimation and mapping motion data onto a calibrated 2D platform model, the proposed method effectively captures crowd movement patterns even in highly congested environments. Additionally, the integration of Unity's NavMesh with an enhanced Simulated Annealing approach enables more accurate origin-destination (OD) estimation and pedestrian trajectory reconstruction. The evaluation results demonstrated the effectiveness of our approach, achieving high accuracy in destination estimation while significantly reducing crowd size estimation errors compared to conventional object detection models such as YOLOv8x. These improvements highlight the potential of combining optical flow-based analysis with crowd simulation techniques to enhance real-time congestion monitoring and management at railway stations.

Further enhancements can be explored to refine origin estimation and optimize the simulated annealing process for even greater accuracy. Future work will focus on extending this framework to real-world deployment, incorporating additional sensor modalities, and improving the robustness of the digital twin model under diverse station conditions. By advancing pedestrian flow analysis, this study contributes to safer and more efficient railway station operations, offering a promising solution for next-generation crowd management systems.

Acknowledgment

This work was partially supported by JST CREST JPMJCR21M5, JST PRESTO JPMJPR2361 and JSPS KAKENHI 23H03384.

References

[1] M. Ohno, R. Ukyo, T. Amano, H. Rizk, and H. Yamaguchi, "Privacy-preserving pedestrian tracking with path image inpainting and 3d point cloud features," *Pervasive Mob. Comput.*, vol. 100, no. C, May 2024. [Online]. Available: https://doi.org/10.1016/j.pmcj.2024.101914

[2] M. Yaseen, "What is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector," 2024. [Online]. Available: https://arxiv.org/abs/2408.15857

[3] S. Ali and M. Shah, "Floor fields for tracking in high density crowd scenes," in *Proc. of the 10th European Conference on Computer Vision: Part II*, ser. ECCV '08. Springer-Verlag, 2008, p. 1–14.

[4] M. Rodriguez, I. Laptev, J. Sivic, and J.-Y. Audibert, "Density-aware person detection and tracking in crowds," in *2011 International Conference on Computer Vision*, 2011, pp. 2423–2430.

[5] N. Wojke and A. Bewley, "Deep cosine metric learning for person re-identification," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 748–756.

[6] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," 2022.

[7] R. Gavrilescu, C. Zet, C. Foșalău, M. Skoczylas, and D. Cotovanu, "Faster r-cnn:an approach to real-time object detection," in *2018 International Conference and Exposition on Electrical And Power Engineering (EPE)*, 2018, pp. 0165–0168.

[8] D. B. Sam, S. V. Peri, M. Narayanan Sundararaman, A. Kamath, and R. V. Babu, "Locate, size and count: Accurately resolving people in dense crowds via detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[9] T. Li, H. Chang, M. Wang, B. Ni, R. Hong, and S. Yan, "Crowded scene analysis: A survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 3, pp. 367–386, 2015.

[10] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 935–942.

[11] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Proceedings of Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Part II 16*. Springer, 2020, pp. 402–419.

[12] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Phys. Rev. E*, vol. 51, pp. 4282–4286, May 1995. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.51.4282

[13] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 261–268.

[14] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 961–971.

[15] L. Shi, L. Wang, C. Long, S. Zhou, M. Zhou, Z. Niu, and G. Hua, "SGCN:Sparse Graph Convolution Network for Pedestrian Trajectory Prediction ," in *Proc. of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 8990–8999.

[16] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "AgentFormer: Agent-Aware Transformers for Socio-Temporal Multi-Agent Forecasting ," in *Proc. of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9793–9803.