Supplementary Materials for Laying the Foundations of Deep Long-Term Crowd Flow Prediction

We provide additional analysis on CAGE with a comparative study of CAGE and several traditional image resampling techniques. Then, we introduce the particulars of the synthetic datasets that will be released and the details of our proposed procedural floorpan generator. Finally, we present a summary of the models, their standard deviation from quantitative evaluation, and more analysis on our results on the Stanford Crowd Flow dataset [1], complimentary to the Manuscript. Two other materials have been provided: a video explaining the CAGE-encoding procedure (Fig. 1.II in the Manuscript) and a sample of our dataset.

1 Optimality of CAGE

We define the objective of the compression as a complete segmentation of the environment into the minimal number of regions, where each region has consistency (i.e., preserves horizontal and vertical lines of sight) and adjacency to at most one other region for each cardinal direction. Consistency ensures that adjacent regions are navigable between each other and adjacency ensures that the segmentation can be encoded in a 2D matrix. We claim that for axis-aligned environments, CAGE achieves optimality by utilizing $\mathbf{V}^{\mathbf{x},\mathbf{y}}$. By construction, the segmentation that $\mathbf{V}^{\mathbf{x},\mathbf{y}}$ provides has consistency and adjacency, because it groups adjacent cells that share the same lines of sight along both the horizontal and vertical axes. Let us consider the $\mathbf{V}^{\mathbf{x},\mathbf{y}}$ segmentation as a set of edges that delimit the segments. If any one of these edges was removed or relocated, there would be at least one new region formed, which contains a change in the line of sight, violating both consistency and adjacency. By representing this new region as a single cell, the geometry within it becomes uncertain and it is no longer aligned with its adjacent regions. Therefore, no consistent and adjacent segmentation exists with a smaller number of regions or a different set of regions than the $\mathbf{V}^{\mathbf{x},\mathbf{y}}$ segmentation.

No new edge can be added to $\mathbf{V}^{\mathbf{x},\mathbf{y}}$ that breaks a line of sight, meaning that consistency is maintained by all supersets of the $\mathbf{V}^{\mathbf{x},\mathbf{y}}$ edges. However, this does not guarantee adjacency. If the new edge also satisfies adjacency, a valid segmentation will be made, but with a larger number of regions than before. Since there can be no smaller segmentation of the environment that maintains consistency and adjacency, CAGE achieves optimal compression.

2 Comparative Analysis on CAGE

Synthetic Environments. In order to understand the benefits of CAGE, we compare its efficacy to traditional image resampling techniques, e.g., nearest-neighbor (NN), OpenCV's area-based, bilinear, bicubic, and Lanczos interpolation, all of which are provided by the OpenCV library [2]. We evaluate the error that results from taking a raw environment, compressing it to 112×112, and decompressing it back to its original size (Fig. 1).

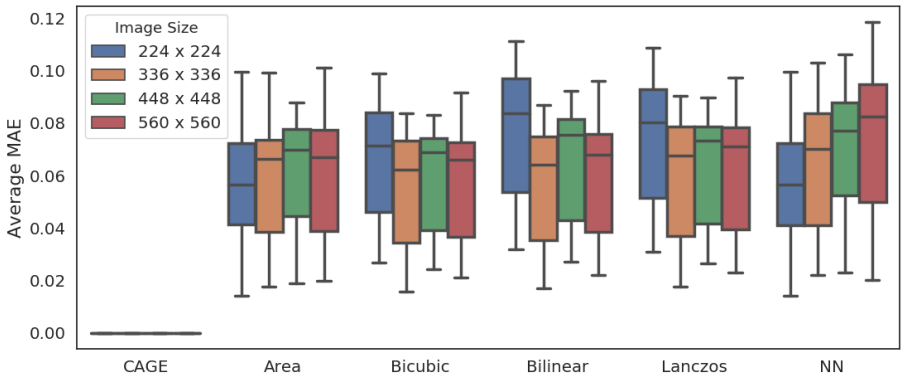

Fig. 1: Average Mean Absolute Error (MAE) of CAGE and traditional image resampling techniques over 36 randomly generated environments for each size. Thirty-six environments were chosen per compression rate, sampling three for each of twelve types of floorplans. The error results from compression and then decompression of a raw environment. All traditional resampling methods lose information as the environments are compressed and decompressed, while CAGE shows lossless resampling performance.

The environments that were used in this evaluation had rooms and corridors of different sizes in order to represent real environments. All resampling methods, except for CAGE, lose information as the environments are compressed and decompressed. They typically generate artifacts at the borders of navigable and non-navigable areas. CAGE's success can be attributed to the fact that it was designed specifically to handle axis-aligned, rectangular environments. The particular reason behind this performance is that CAGE compresses larger features (e.g., rooms and corridors) more than it does smaller features, and only features that can be compressed with perfect accuracy. The errors resulting from the compression and decompression process are visualized in Fig. 2, which shows the difference between the decompressed environment and the original environment. Falsely assigned navigable space is shown in red, and falsely assigned non-navigable space is shown in blue.

Non-Axis-Aligned Environments Using the previously described CAGE encoding technique, the least compressible environment is an isosceles right triangle, which causes each cell in the environment to have a unique visibility $\mathbf{V}^{\mathbf{x},\mathbf{y}}$

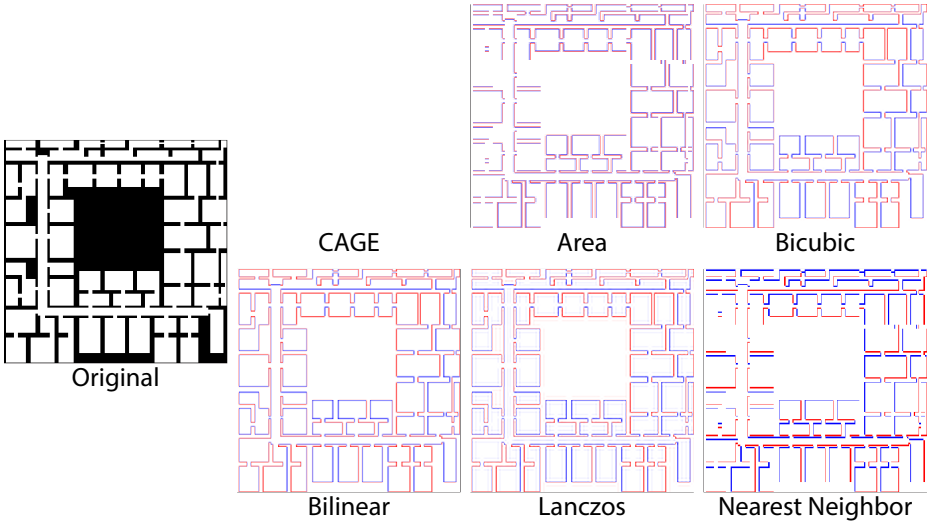

Fig. 2: Qualitative comparison between `CAGE` and traditional image resampling techniques. The right-side 6 images show the signed differences that highlight where the resampling generates false navigable areas (red) and false non-navigable areas (blue).

(Fig. 3 (a)). Thereby, none of the visibility values can be compressed. As the angle gets more parallel to the x- or y-axes, the incompressibility reduces.

In our NARF dataset, we have real-world floorplans from the University of Economics Vienna's Library and Learning Centre, King's Cross Station (beside the tracks), Prentice Women's Hospital, and Tate Britain. Among these environments, which are showcased in Fig. 4, a majority is non-axis-aligned, meaning that the original `CAGE` encoding technique would not significantly compress the environment if losslessness is maintained.

However, it is possible to perform a lossy compression on non-axis-aligned environments using an alternative, more simplistic approach. Instead of compressing an environment using its rectangular regions that have the same visibility, an environment can be compressed based on its entire rows and columns. For example, in Fig. 3 (a), this means that instead of processing each visibility region (of which there are 15 incompressible ones), the entire 5-cell column can be looked at. For an axis-aligned environment, lossless compression can be achieved by combining adjacent rows and columns that share the same environment values in $\mathbf{E}$ (e.g., all of the rows and columns in Fig. 3 (b)). For a non-axis-aligned environment, this perfect-match criterion can be relaxed to achieve a lossy compression. Between the two leftmost columns [0] and [1] in Fig. 3 (a), we can compute the similarity as the mean absolute error between their $\mathbf{E}$ values (which would be equal to 1/5). We can then set an error threshold θ such that if the MAE is less than or equal to it, the two vectors (either rows or columns) can be combined into the one. When $\theta = 0.2$, the columns of Fig. 3 (a) can

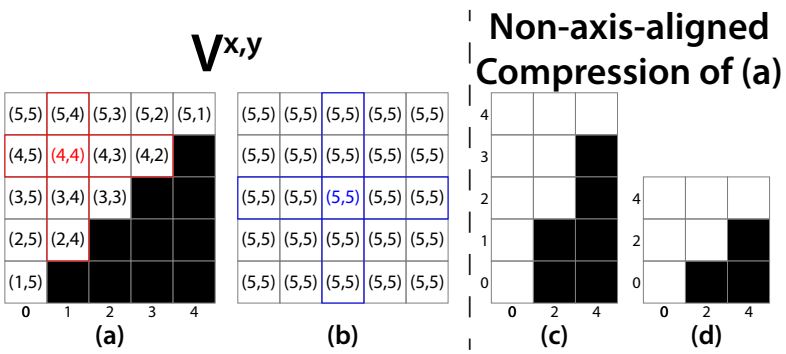

Fig. 3: Matrix (a) shows an environment that is maximally incompressible, while matrix (b) shows one that is maximally compressible. For each matrix, the environments are compressible when visibility $\mathbf{V}^{\mathbf{x},\mathbf{y}}$ is shared between adjacent cells. The white cells indicate where $\mathbf{E}$ is navigable, while black cells denote where $\mathbf{E}$ is non-navigable. (c) shows the lossy compression of (a) along the x-axis and (d) shows the subsequent compression of (a) along the y-axis.

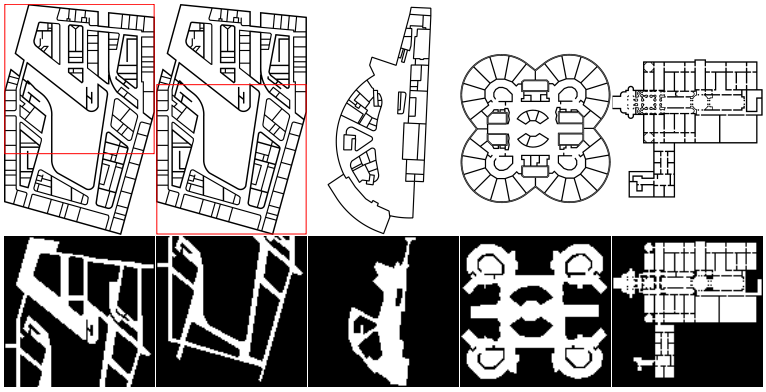

Fig. 4: The 5 floorplan images above (from left to right) are of the University of Economics Vienna's Library and Learning Centre, King's Cross Station, Prentice Women's Hospital, and Tate Britain, which are real-world environments with non-axis-aligned geometries.

be compressed into Fig. 3 (c), which can have its rows further compressed into Fig. 3 (d) using $\theta = 0$.

Although `CAGE` is not always able to losslessly encode non-axis-aligned environments, an alternative method can be used to achieve the same compressed representation. Since our model is ignorant to whichever method is being used, as long as the 5-channel `CAGE` representation is consistent, the model will be able to make a prediction.

Dataset ID	1					2	3	4
LoS	A∼F	A∼F	A∼F	A∼F	A∼F	A∼F	A∼F	A∼F
CR	1	1.25	1.5	1.75	2	1	1	1
Flow	Proxy	Proxy	Proxy	Proxy	Proxy	SF	MGNU	NARF
Train Size	12k	12k	12k	12k	12k	7.2k	4.2k	N/A
Test Size	3k	3k	3k	3k	3k	600	480	120
Total	87,600							

Table 1: Statistics of the datasets to be released. In total of 87,600 examples are included. Note the equal size per LoS and equal amounts of the twelve types of floorplans (generated by our procedural floorplan generator) in each dataset.

3 Datasets to Release

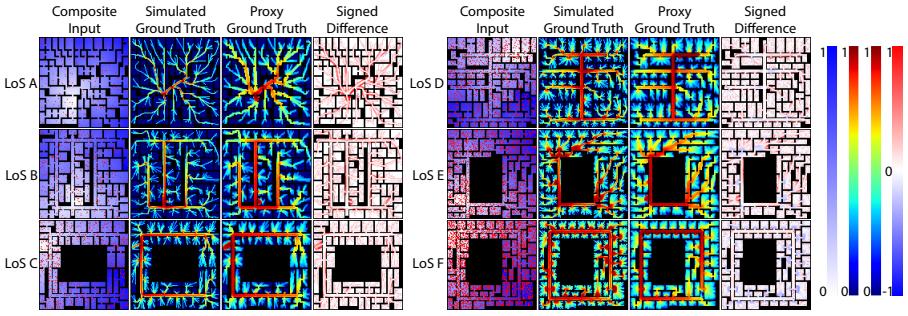

Fig. 5: We show the visual differences between *ground truth* simulated crowd flow and *ground truth* proxy crowd flow for each Level of Service on uncompressed environments of size 112×112. Substantial resemblance could be observed between the two types of crowd flow (especially as density increases), suggesting the proxy crowd flow could capture the major features of simulated crowd flow, and could serve as a computational-efficient alternative for the simulated crowd flow.

Dataset Overview. Tab. 1 shows the statistics of different types of data in our dataset, which we have submitted a sample of. The columns represent data with unique combinations of compression rate (CR) and flow type, and each one has equal amounts of the 12 environment types described in the next section. For a given input-output pair, we provided the input and output channels as separate grayscale images. There are two types of datasets: For data with CR greater than 1 (i.e., requires compression), the raw input (a 3D tensor) of one data point is divided into 3 grayscale images for $\mathbf{A}$, $\mathbf{G}$, and $\mathbf{E}$; its `CAGE` input (a 5D tensor) is divided into 5 grayscale images for $\mathbf{C}^{\mathbf{x}}$, $\mathbf{C}^{\mathbf{y}}$, $\mathbf{A}'$, $\mathbf{G}'$, and $\mathbf{E}'$; and its corresponding output is divided into 2 grayscale images for $\mathbf{Y}'$ and $\mathbf{Y}$. For data with CR equal to 1 (i.e., compression is not necessary), the input (a 3D

tensor) of one data point is divided into 3 grayscale images for **A**, **G**, and **E**; and its corresponding output (a 1D tensor) is another grayscale image for **Y**.

Proxy Versus Simulated Crowd Flow. Fig. 5 shows the differences between proxy and simulated crowd flow for all Levels of Service (LoS). At lower density levels, the proxy crowd flow appears wider than the simulated crowd flow and generally overestimates the crowd flow (as shown by the red areas in the signed difference column). As the density level increases from LoS A to LoS F, the proxy crowd flow transitions from overestimating the width of simulated crowd flow to underestimating its congestion at bottlenecks, which begins midway through at LoS D. However, there is generally a reasonable resemblance between the two types of crowd flow.

4 Details of Procedural Floorplan Generation

We propose a procedural floorplan generator to produce realistic floorplans of various size and topology for our synthetic datasets. Here, we present the details of the proposed procedural floorplan generator. The proposed procedural floorplan generator is based on the work of Dogan, et al. [3], in which they establish a comprehensive taxonomy of floorplans based on observations of modern architecture. Namely, these observations are of the morphology (i.e., exterior shape) and organization (i.e., interior core) of built environments, which are classified into separate typologies. To elaborate, "core" signifies areas of high centrality in the environment, such as a corridor. The details of the typologies are provided below.

Morphological Typology:

- *Point*: A compact, convex shape.
- *Block*: A convex shape with a hole in its interior.
- *Line*: An elongated, convex shape.

Organizational Typology:

- *Vertical Point*: A core point.
- *Corridor Center*: A core line that divides a space.
- *Corridor Edge*: A core line that borders a space.
- *Compartments*: A complex set of core line segments.

The above typologies are subsets of the ones used in the original paper, because the other types were either not unique or not well-defined. We have re-purposed this taxonomy in order to generate floorplans instead of classifying them. To this end, we first define an empty environment matrix **E** with a desired size. We then mask the exterior of **E** based on a morphology type, making the area non-navigable. *Point* has no mask, *Block* has a rectangular mask in the center of **E**, and *Line* has a mask on the lower part of **E** (Fig. 6 (a)).

Based on the examples that Dogan, et al. [3] provided for their taxonomy, we inferred that for organization, the cores should be oriented along the medial axis of the masked **E**. Since **E** consists of rectangular components after masking, the

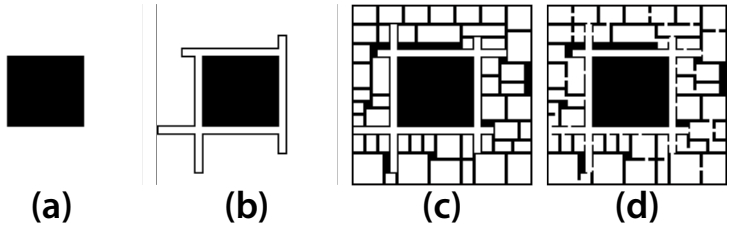

Fig. 6: The above images show (from left to right) the stages of our procedural generation of floorplans.

medial axis can be approximated in segments by bisecting the rectangular components along their longer axes. For each rectangular component, we randomly determine whether it has a core, and if so, we raise a non-navigable perimeter around it. *Vertical Point*'s core is a small square, and *Corridor Center* and *Corridor Edge*'s are long, thin rectangles. The *Compartments* organization is an exception that does not rely on the medial axis (Fig. 6 (b)).

Afterwards, we populate the remaining space with rectangular rooms that have their dimensions randomly sampled within a set range. The rooms are populated from the top of **E** to the bottom, and each room has a probability of sharing the same dimensions as the prior room to emulate the regularity seen in real built environments. If a room's dimensions do not fit in a space, it is removed. Otherwise, it has a perimeter raised like the corridors (Fig. 6 (c)).

What remains is a set of disconnected components (i.e., corridors and rooms), for which we must ensure that the corridors function as cores. To give them high centrality, we create an adjacency graph of the components and use Dijkstra's algorithm to find the shortest paths from the corridors to the rooms. If two components are connected by a shortest path, an entrance is cut out in their perimeters (Fig. 6 (d)). In effect, there becomes a high chance that if an agent is navigating between two random positions in **E**, it will pass through a corridor. Fig. 7 shows each of the 12 main types of floorplans that result from the procedural generator.

5 Miscellaneous

Model Summary. We present a summary of the models with details of their training data in Tab. 2 for easier comparison and reference. During training, all models share the same hyper-parameters which we have described in Sec. 3.3 in the Manuscript.

Standard Deviation of MAE in Manuscript. We provide the standard deviation of Mean Absolute Error for models trained without compression (associated with Tab. 1 in the Manuscript) in Tab. 3. Also, the standard deviation of Mean Absolute Error for models trained with `CAGE` under different compression rates (associated with Tab. 2 in the Manuscript) can be found in Tab. 4.

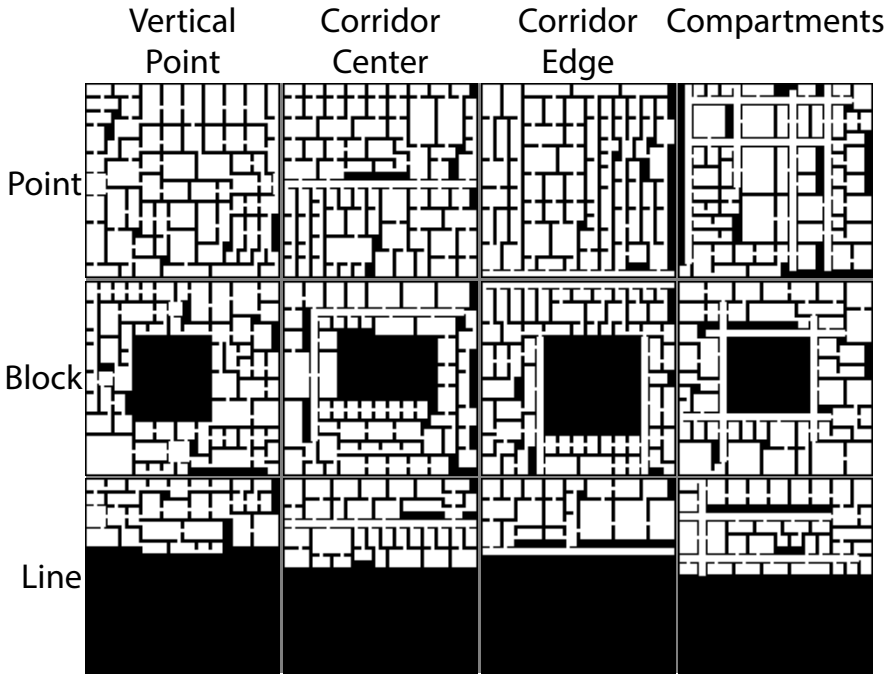

Fig. 7: The 12 floorplans shown above are examples from each combination of morphological and organizational typologies that we use in our procedural generation.

Model	LOS-Train	CR-Train	Flow-Train	#TrainSamples	#TestSamples
L^A	A	1	Proxy	1000	500
L^B	B	1	Proxy	1000	500
L^C	C	1	Proxy	1000	500
L^D	D	1	Proxy	1000	500
L^E	E	1	Proxy	1000	500
L^F	F	1	Proxy	1000	500
$PanL^{PX}$	A∼F	1	Proxy	12000	3000
$CR^{1.25}$	A∼F	1.25	Proxy	12000	3000
$CR^{1.5}$	A∼F	1.5	Proxy	12000	3000
$CR^{1.75}$	A∼F	1.75	Proxy	12000	3000
CR^2	A∼F	2	Proxy	12000	3000
$PanCR$	A∼F	1∼2	Proxy	18000	3000
$PanL^{SF}$	A∼F	1	SF	7200	600
$PanL^{SF+}$	A∼F	1	SF + MGNU	7200	600

Table 2: Overview of Models.

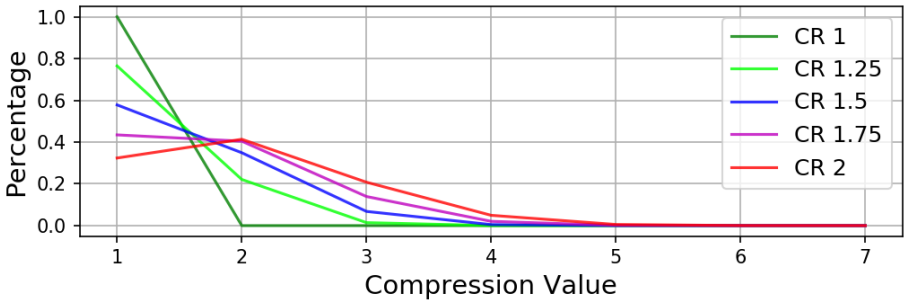

Fig. 8: Distribution of compression values from $\mathbf{C^x}$ and $\mathbf{C^y}$ in the datasets of the models trained under different compression rates. Only CR2.0-trained model does not have a majority of uncompressed cell values in its data.

Additional Analysis of Results on Stanford Data. In the Manuscript, we evaluated the performance of models $PanL^{PX}$, $PanL^{SF}$, and $PanL^{SF+}$ on Stanford real-world human trajectories. Models $PanL^{SF}$ and $PanL^{SF+}$ were not trained on compressed data, because it is prohibitively expensive to simulate, so the real-world environments and crowd flows were scaled down to fit within 112×112. In Tab. 5 and Fig. 9, we convey the results of compressing the Stanford dataset from its proper size of 112×170 to 112×112 using `CAGE` and testing with models $PanL^{PX}$, $CR^{1.25}$, $CR^{1.5}$, $CR^{1.75}$, CR^{2}, and $PanCR$, which were all presented in the Manuscript. The approximate compression rate resulting from the compression is 1.52. It is evident that nearly all models trained on higher compression rates performed better on average than $PanL^{PX}$ (which has only been trained uncompressed data), even though all of the aforementioned models were trained on proxy data, not real data. We attribute this improvement to the learning of higher compression rates during training. This finding is notable, because it suggests that learning compression through proxy data can generalize to real-world crowd flows. Given all prior results, we have reasons to believe that by training a model on compressed simulated crowd flows, we can improve these results even further. This additional analysis on Stanford dataset of real-world crowd flow (which has been presented in the Manuscript) further highlights the effectiveness of our pipeline, including `CAGE`.

We consider the results of CR^2 as an outlier in both the Manuscript and Supplementary Material, because during its training, uncompressed data comprised a minority of the training data, while for other models, a majority of their training data was uncompressed (Fig. 8). We therefore conclude that learning from uncompressed training data is critical for a model to perform well and generalize to higher, potentially unseen compression rates.

Additional Qualitative Analysis of $PanL^{SF+}$ Results. We provide additional visualizations for the test results of model $PanL^{SF+}$ (which is the best-

performing model trained on simulated crowd flow) on the SF (Fig. 10), MGNU (Fig. 11), and NARF datasets (Fig. 12, 13, 14, and 15).

	$L^{A\sim F}$	$PanL^{PX}$			$PanL^{SF}$					$PanL^{SF+}$			
CR-Train	1	1	1	1	1	1	1	1	1	1	1	1	1
Flow-Train	Proxy	Proxy	Proxy	Proxy	SF	SF	SF	SF	SF	SF + MGNU	SF + MGNU	SF + MGNU	SF + MGNU
Flow-Test	Proxy	Proxy	SF	Stanford	Proxy	SF	MGNU	NARF	Stanford	SF	MGNU	NARF	Stanford
LoS A	0.0431	0.0064	0.0225	0.0005	0.0231	0.0078	0.0110	0.0211	0.0015	0.0063	0.0069	0.0161	0.0012
LoS B	0.0146	0.0059	0.0244	0.0010	0.0241	0.0083	0.0123	0.0207	0.0004	0.0069	0.0078	0.0176	0.0009
LoS C	0.0135	0.0056	0.0253	0.0010	0.0245	0.0092	0.0105	0.0210	0.0008	0.0076	0.0077	0.0183	0.0011
LoS D	0.0127	0.0054	0.0246	0.0010	0.0243	0.0092	0.0112	0.0204	0.0007	0.0078	0.0090	0.0193	0.0010
LoS E	0.0118	0.0050	0.0245	N/A	0.0247	0.0089	0.0110	0.0210	N/A	0.0077	0.0102	0.0174	N/A
LoS F	0.0103	0.0052	0.0251	N/A	0.0268	0.0089	0.0110	0.0207	N/A	0.0080	0.0086	0.0159	N/A

Table 3: Standard Deviation of Mean Absolute Error of models trained without compression.

	$PanL^{PX}$	$CR^{1.25}$	$CR^{1.5}$	$CR^{1.75}$	CR^{2}	$PanCR$
CR-Train	1	1.25	1.5	1.75	2	1∼2
Flow-Train	Proxy	Proxy	Proxy	Proxy	Proxy	Proxy
Flow-Test	Proxy	Proxy	Proxy	Proxy	Proxy	Proxy
LoS A	0.0728 ± 0.0401	0.0346 ± 0.0142	0.0358 ± 0.0139	0.0357 ± 0.0124	0.1270 ± 0.0382	**0.0322 ± 0.0131**
LoS B	0.0582 ± 0.0322	0.0312 ± 0.0134	0.0320 ± 0.0132	0.0326 ± 0.0119	0.1147 ± 0.0394	**0.0289 ± 0.0123**
LoS C	0.0509 ± 0.0268	0.0296 ± 0.0120	0.0300 ± 0.0116	0.0308 ± 0.0107	0.1248 ± 0.0439	**0.0269 ± 0.0107**
LoS D	0.0475 ± 0.0240	0.0274 ± 0.0105	0.0279 ± 0.0103	0.0285 ± 0.0091	0.1345 ± 0.0496	**0.0248 ± 0.0092**
LoS E	0.0402 ± 0.0210	0.0232 ± 0.0093	0.0237 ± 0.0096	0.0241 ± 0.0085	0.0997 ± 0.0393	**0.0210 ± 0.0086**
LoS F	0.0389 ± 0.0205	0.0212 ± 0.0081	0.0224 ± 0.0091	0.0221 ± 0.0075	0.0795 ± 0.0296	**0.0211 ± 0.0089**
CR 1.0	**0.0164 ± 0.0058**	0.0190 ± 0.0068	0.0201 ± 0.0073	0.0261 ± 0.0089	0.1011 ± 0.0381	0.0167 ± 0.0059
CR 1.25	0.0433 ± 0.0179	0.0247 ± 0.0093	0.0255 ± 0.0096	0.0280 ± 0.0102	0.1057 ± 0.0397	**0.0236 ± 0.0088**
CR 1.5	0.0577 ± 0.0241	0.0288 ± 0.0112	0.0293 ± 0.0112	0.0291 ± 0.0112	0.1097 ± 0.0410	**0.0273 ± 0.0104**
CR 1.75	0.0667 ± 0.0280	0.0321 ± 0.0127	0.0326 ± 0.0125	0.0301 ± 0.0120	0.1128 ± 0.0426	**0.0298 ± 0.0116**
CR 2.0	0.0728 ± 0.0299	0.0348 ± 0.0136	0.0355 ± 0.0135	**0.0313 ± 0.0126**	0.1415 ± 0.0505	0.0317 ± 0.0123

Table 4: Standard Deviation of Mean Absolute Error of models trained with `CAGE` under different compression rate (CR).

	$PanL^{PX}$	$CR^{1.25}$	$CR^{1.5}$	$CR^{1.75}$	CR^{2}	$PanCR$
CR-Train	1	1.25	1.5	1.75	2	1∼2
Flow-Train	Proxy	Proxy	Proxy	Proxy	Proxy	Proxy
Flow-Test	Stanford	Stanford	Stanford	Stanford	Stanford	Stanford
LoS A	0.0392 ± 0.0041	0.0322 ± 0.0035	0.0368 ± 0.0028	**0.0308 ± 0.0020**	0.0455 ± 0.0034	0.0315 ± 0.0041
LoS B	0.0407 ± 0.0018	0.0355 ± 0.0019	0.0368 ± 0.0055	**0.0294 ± 0.0019**	0.0488 ± 0.0059	0.0323 ± 0.0050
LoS C	0.0377 ± 0.0040	0.0323 ± 0.0031	0.0362 ± 0.0030	**0.0305 ± 0.0026**	0.0502 ± 0.0052	0.0322 ± 0.0026
LoS D	0.0379 ± 0.0044	0.0326 ± 0.0026	0.0361 ± 0.0015	0.0313 ± 0.0019	0.0518 ± 0.0065	**0.0309 ± 0.0017**
LoS E	N/A	N/A	N/A	N/A	N/A	N/A
LoS F	N/A	N/A	N/A	N/A	N/A	N/A

Table 5: Average and standard deviation of Mean Absolute Error of models trained with `CAGE` on Stanford Crowd Flow dataset.

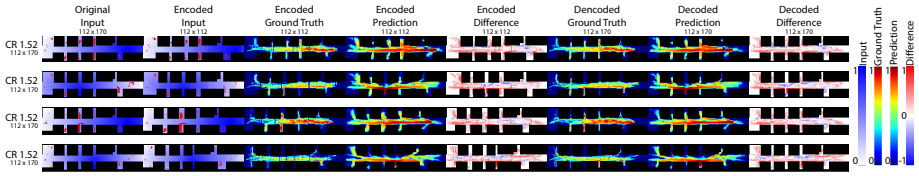

Fig. 9: These results show the predictions of $CR^{1.75}$ on the compressed Stanford dataset.

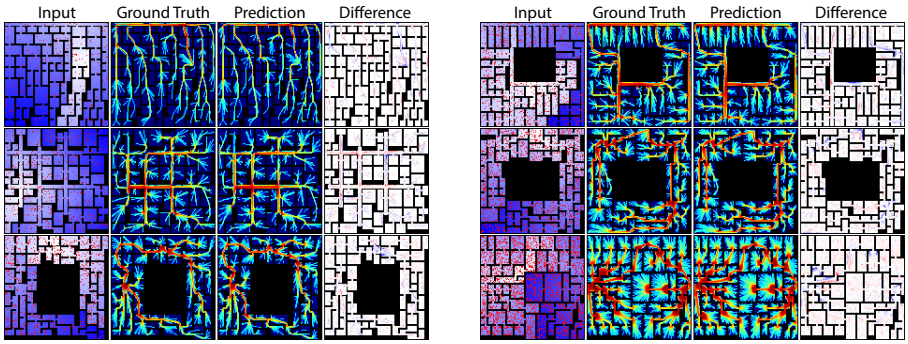

Fig. 10: These results show the predictions of $PanL^{SF+}$ on the SF dataset.

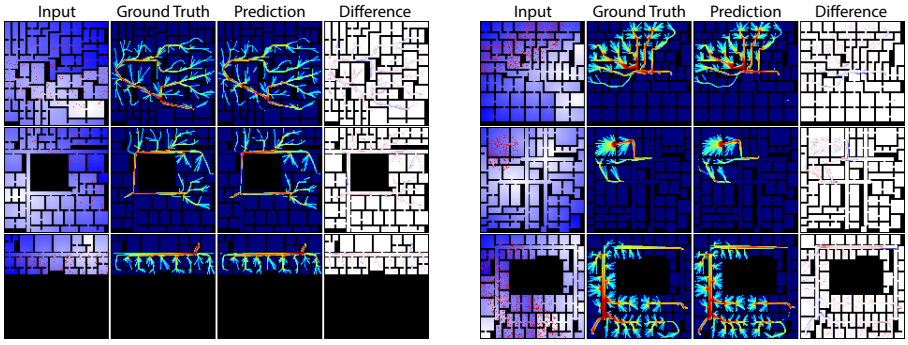

Fig. 11: These results show the predictions of $PanL^{SF+}$ on the MGNU dataset.

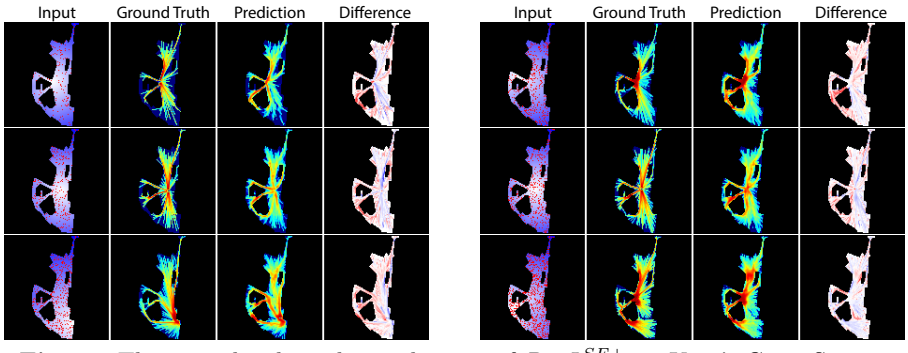

Fig. 12: These results show the predictions of $PanL^{SF+}$ on King's Cross Station.

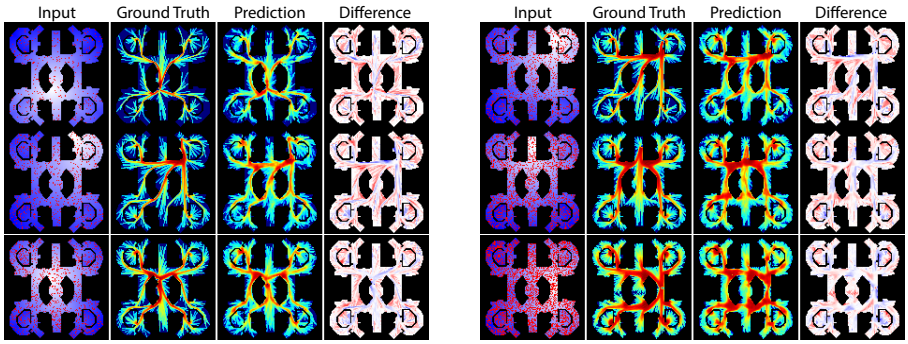

Fig. 13: These results show the predictions of $PanL^{SF+}$ on the Prentice Women's Hospital.

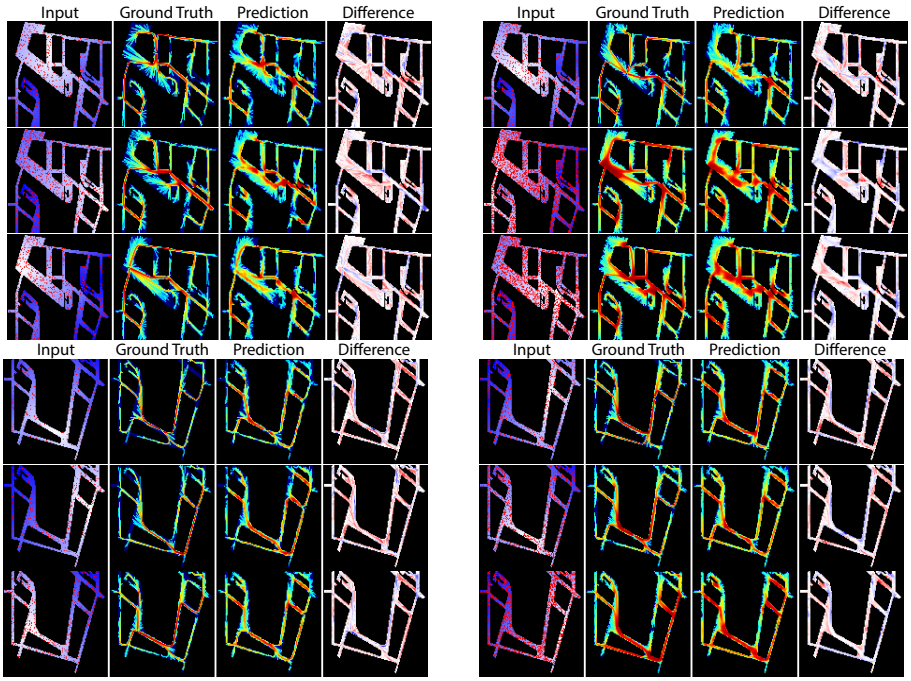

Fig. 14: These results show the predictions of $PanL^{SF+}$ on the University of Economics Vienna's Library and Learning Centre.

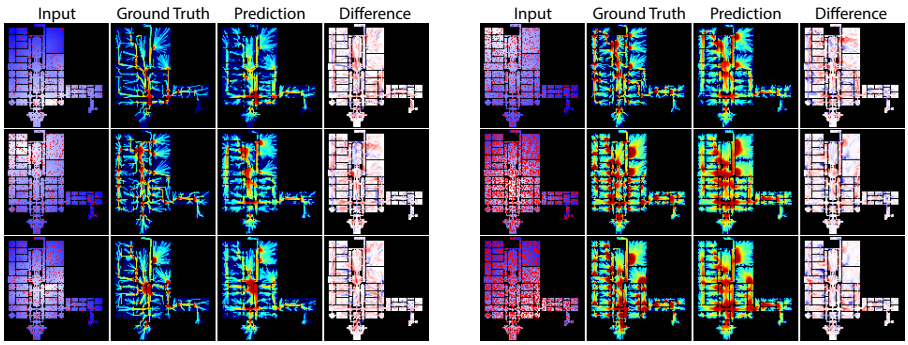

Fig. 15: These results show the predictions of $PanL^{SF+}$ on the Tate Britain.

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social LSTM: Human trajectory prediction in crowded spaces. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 961–971 (2016) 1
2. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000) 2
3. Dogan, T., Saratsis, E., Reinhart, C.: The optimization potential of floor-plan typologies in early design energy modeling (12 2015) 6