Informative Scene Decomposition for Crowd Analysis, Comparison and Simulation Guidance

FEIXIANG HE, University of Leeds, United Kingdom
YUANHANG XIANG, Xi'an Jiaotong University, China
XI ZHAO*, Xi'an Jiaotong University, China
HE WANG†, University of Leeds, United Kingdom

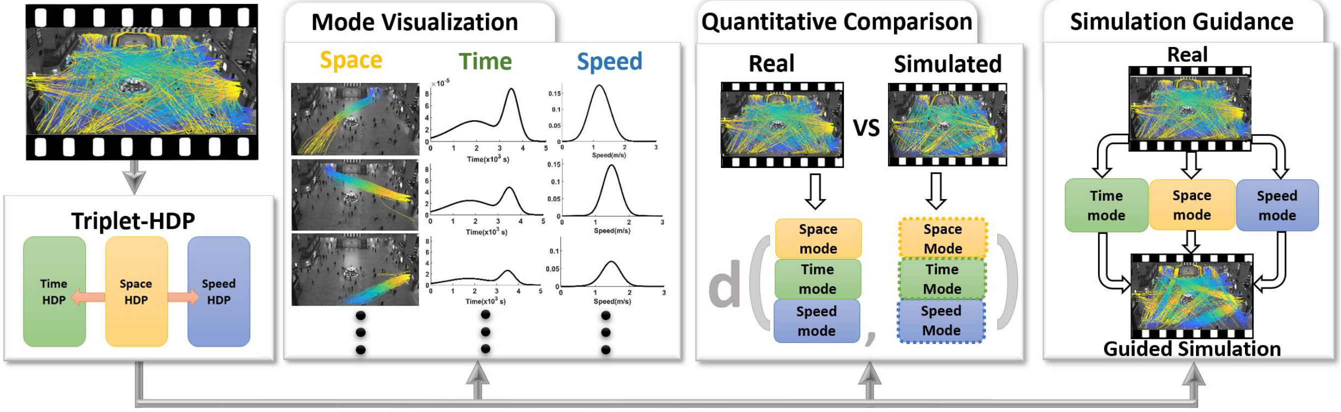

Fig. 1. Overview of our framework.

Crowd simulation is a central topic in several fields including graphics. To achieve high-fidelity simulations, data has been increasingly relied upon for analysis and simulation guidance. However, the information in real-world data is often noisy, mixed and unstructured, making it difficult for effective analysis, therefore has not been fully utilized. With the fast-growing volume of crowd data, such a bottleneck needs to be addressed. In this paper, we propose a new framework which comprehensively tackles this problem. It centers at an unsupervised method for analysis. The method takes as input raw and noisy data with highly mixed multi-dimensional (space, time and dynamics) information, and automatically structure it by learning the correlations among these dimensions. The dimensions together with their correlations fully describe the scene semantics which consists of recurring activity patterns in a scene, manifested as space flows with temporal and dynamics profiles. The effectiveness and robustness of the analysis have been tested on datasets with great variations in volume, duration, environment and crowd dynamics. Based on the analysis, new methods for data visualization, simulation evaluation and simulation guidance are also proposed. Together, our framework establishes a highly automated pipeline from raw data to crowd analysis, comparison and simulation guidance. Extensive experiments and evaluations have been conducted to show the flexibility, versatility and intuitiveness of our framework.

CCS Concepts: • **Computing methodologies** → **Animation**; **Topic modeling**; **Learning in probabilistic graphical models**; **Scene understanding**; **Activity recognition and understanding**; **Multi-agent planning**; • **Mathematics of computing** → **Probabilistic inference problems**; **Nonparametric statistics**.

Additional Key Words and Phrases: Crowd Simulation, Simulation Evaluation, Bayesian Inference

ACM Reference Format:
Feixiang He, Yuanhang Xiang, Xi Zhao, and He Wang. 2020. Informative Scene Decomposition for Crowd Analysis, Comparison and Simulation Guidance. *ACM Trans. Graph.* 39, 4, Article 1 (July 2020), 15 pages. https://doi.org/10.1145/3386569.3392407

1 INTRODUCTION

Crowd simulation has been intensively used in computer animation, as well as other fields such as architectural design and crowd management. The fidelity or realism of simulation has been a long-standing problem. The main complexity arises from its multifaceted nature. It could mean high-level global behaviors [Narain et al. 2009], mid-level flow information [Wang et al. 2016] or low-level individual motions [Guy et al. 2012]. It could also mean perceived realism [Ennis et al. 2011] or numerical accuracy [Wang et al. 2017]. In

*Corresponding author
†Corresponding author

Authors' addresses: Feixiang He, University of Leeds, School of Computing, United Kingdom, fxhe1992@gmail.com; Yuanhang Xiang, Xi'an Jiaotong University, School of Computer Science and Technology, China, xiangyuanhang@icloud.com; Xi Zhao, Xi'an Jiaotong University, School of Computer Science and Technology, China, zhaoxi.jade@gmail.com; He Wang, University of Leeds, School of Computing, United Kingdom, realcrane@gmail.com.

0730-0301/2020/7-ART1 $15.00
https://doi.org/10.1145/3386569.3392407

any case, analyzing real-world data is inevitable for evaluating and guiding simulations.

The main challenges in utilizing real-world data are data complexity, intrinsic motion randomness and the shear volume. The data complexity makes structured analysis difficult. As the most prevalent form of crowd data, trajectories extracted from sensors contain rich but mixed and unstructured information of space, time and dynamics. Although high-level statistics such as density can be used for analysis, they are not well defined and cannot give structural insights [Wang et al. 2017]. Second, trajectories show intrinsic randomness of individual motions [Guy et al. 2012]. The randomness shows heterogeneity between different individuals and groups, and is influenced by internal factors such as state of mind and external factors such as collision avoidance. Hence a single representation is not likely to be able to capture all randomness for all people in a scene. This makes it difficult to guide simulation without systematically considering the randomness. Lastly, with more recording devices being installed and data being shared, the shear volume of data in both space and time, with excessive noise, requires efficient and robust analysis.

Existing methods that use real-world data for purposes such as qualitative and quantitative comparisons [Wang et al. 2016], simulation guidance [Ren et al. 2018] or steering [López et al. 2019], mainly focus on one aspect of data, e.g. space, time or dynamics, and tend to ignore the structural correlations between them. Also during simulation and analysis, motion randomness is often ignored or uniformly modelled for all trajectories [Guy et al. 2012; Helbing et al. 1995]. Ignoring the randomness (e.g. only assuming the least-effort principle) makes simulated agents to walk in straight lines whenever possible, which is rarely observed in real-world data; uniformly modelling the randomness fails to capture the heterogeneity of the data. Besides, most existing methods are not designed to deal with massive data with excessive noise. Many of them require the full trajectories to be available [Wolinski et al. 2014] which cannot be guaranteed in real world, and do not handle data at the scale of tens of thousands of people and several days long.

In this paper, we propose a new framework that addresses the three aforementioned challenges. This framework is centered at an analysis method which automatically decomposes a crowd scene of a large number of trajectories into a series of *modes*. Each mode comprehensively captures a unique pattern of spatial, temporal and dynamics information. Spatially, a mode represents a pedestrian flow which connects subspaces with specific functionalities, e.g. entrance, exit, information desk, etc.; temporally it captures when this flow appears, crescendos, wanes and disappears; dynamically it reveals the speed preferences on this flow. With space, time and dynamics information, each mode represents a unique recurring activity and all modes together describe the *scene semantics*. These modes serve as a highly flexible visualization tool for general and task-specific analysis. Next, they form a natural basis where explicable evaluation metrics can be derived for quantitatively comparing simulated and real crowds, both holistically and dimension-specific (space, time and dynamics). Lastly, they can easily automate simulation guidance, especially in capturing the heterogeneous motion randomness in the data.

The analysis is done by a new *unsupervised* clustering method based on non-parametric Bayesian models, because manual labelling would be extremely laborious. Specifically, Hierarchical Dirichlet Processes (HDP) are used to disentangle the spatial, temporal and dynamics information. Our model consists of three intertwined HDPs and is thus named Triplet HDPs (THDP). The outcome is a (potentially infinite) number of modes with weights. Spatially, each mode is a crowd flow represented by trajectories sharing spatial similarities. Temporally, it is a distribution of when the flow appears, crescendos, peaks, wanes and disappears. Dynamically, it shows the speed distribution of the flow. The whole data is then represented by a weighted combination of all modes. Besides, the power of THDP comes with an increased model complexity, which brings challenges on inference. We therefore propose a new method based on Markov Chain Monte Carlo (MCMC). The method is a major generalization of the Chinese Restaurant Franchise (CRF) method, which was originally developed for HDP. We refer to the new inference method as Chinese Restaurant Franchise League (CRFL). THDP and CRFL are general and effective on datasets with great spatial, temporal and dynamics variations. They provide a versatile base for new methods for visualization, simulation evaluation and simulation guidance.

Formally, we propose the first, to our best knowledge, multi-purpose framework for crowd analysis, visualization, simulation evaluation and simulation guidance, which includes:

(1) a new activity analysis method by unsupervised clustering.
(2) a new visualization tool for highly complex crowd data.
(3) a set of new metrics for comparing simulated and real crowds.
(4) a new approach for automated simulation guidance.

To this end, we have technical contributions which include:

(1) the first, to our best knowledge, non-parametric method that holistically considers space, time and dynamics for crowd analysis, simulation evaluation and simulation guidance.
(2) a new Markov Chain Monte Carlo method which achieves effective inference on intertwined HDPs.

2 RELATED WORK

2.1 Crowd Simulation

Empirical modelling and data-driven methods have been the two mainstreams in simulation. Empirical modelling dominates early research, where observations of crowd motions are abstracted into mathematical equations and deterministic systems. Crowds can be modelled as fields or flows [Narain et al. 2009], or as particle systems [Helbing et al. 1995], or by velocity and geometric optimization [van den Berg et al. 2008]. Social behaviors including queuing and grouping [Lemercier et al. 2012; Ren et al. 2016] have also been pursued. On the other hand, data-driven simulation has also been explored, in using e.g. first-person vision to guide steering behaviors [López et al. 2019] or trajectories to extract features to describe motions [Karamouzas et al. 2018; Lee et al. 2007]. Our research is highly complementary to simulation research in providing analysis, guidance and evaluation metrics. It aims to work with existing steering and global planning methods.

2.2 Crowd Analysis

Crowd analysis has been a trendy topic in computer vision [Wang and O'Sullivan 2016; Wang et al. 2008]. They aim to learn structured latent patterns in data, similar to our analysis method. However, they only consider limited information (e.g. space only or space/time) compared to our method because our method explicitly models space, time, dynamics and their correlations. In contrast, another way of scene analysis is to focus on the anomalies [Charalambous et al. 2014]. Their perspective is different from ours and therefore complementary to our approach. Trajectory analysis also plays an important role in modern sports analysis [Sha et al. 2018, 2017], but they do not deal with a large number of trajectories as our method does. Recently, deep learning has been used for crowd analysis in trajectory prediction [Xu et al. 2018], people counting [Wang et al. 2019], scene understanding [Lu et al. 2019] and anomaly detection [Sabokrou et al. 2017]. However, they either do not model low-level behaviors or can only do short-horizon prediction (seconds). Our research is orthogonal to theirs by focusing on the analysis and its applications in simulations.

Besides computer vision, crowd analysis has also been investigated in physics. In [Ali and Shah 2007], Lagrangian Particle Dynamics is exploited for the segmentation of high-density crowd flows and detection of flow instabilities, where the target was similar to our analysis. But they only consider space when separating flows, while our research explicitly models more comprehensive information, including space, time and dynamic. Physics-inspired approaches have also been applied in abnormal trajectory detection for surveillance [Chaker et al. 2017; Mehran et al. 2009]. An approach based on social force model [Mehran et al. 2009] is introduced to describe individual movement in microscopic by placing a grid particle over the image. A local and global social network are built by constructing a set of spatio-temporal cuboids in [Chaker et al. 2017] to detect anomalies. Compared with these methods, our anomaly detection is more informative and versatile in providing what attributes contribute to the abnormality.

2.3 Simulation Evaluation

How to evaluate simulations is a long-standing problem. One major approach is to compare simulated and real crowds. There are qualitative and quantitative methods. Qualitative methods include visual comparison [Lemercier et al. 2012] and perceptual experiments [Ennis et al. 2011]. Quantitative methods fall into model-based methods [Golas et al. 2013] and data-driven methods [Guy et al. 2012; Lerner et al. 2009; Wang et al. 2016, 2017]. Individual behaviors can be directly compared between simulation and reference data [Lerner et al. 2009]. However, it requires full trajectories to be available which is difficult in practice. Our comparison is based on the latent behavioral patterns instead of individual behaviors and does not require full trajectories. The methods in [Wang et al. 2016, 2017] are similar to ours where only space is considered. In contrast, our approach is more comprehensive by considering space, time and dynamics. Different combinations of these factors result in different metrics focusing on comparing different aspects of the data. The comparisons can be spatially focused or temporally focused. They can also be comparing general situations or specific modes. Overall, our method provides greater flexibility and more intuitive results.

2.4 Simulation Guidance

Quantitative simulation guidance has been investigated before, through user control or real-world data. In the former, trajectory-based user control signals can be converted into guiding trajectories for simulation [Shen et al. 2018]. Predefined crowd motion 'patches' can be used to compose heterogeneous crowd motions [Jordao et al. 2014]. The purpose of this kind of guidance is to give the user the full control to 'sculpture' crowd motions. The latter is to guide simulations using real-world data to mimic real crowd motions. Given data and a parameterized simulation model, optimizations are used to fit the model on the data [Wolinski et al. 2014]. Alternatively, features can be extracted and compared for different simulations, so that predictions can be made about different steering methods on a simulation task [Karamouzas et al. 2018]. Our approach also heavily relies on data and is thus similar to the latter. But instead of anchoring on the modelling of individual motions, it focuses on the analysis of scene semantics/activities. It also considers intrinsic motion randomness in a structured and principled way.

3 METHODOLOGY OVERVIEW

The overview of our framework is in Fig. 1. Without loss of generality, we assume that the input is raw trajectories/tracklets which can be extracted from videos by existing trackers, where we can estimate the temporal and velocity information. Naively modelling the trajectories/tracklets, e.g. by simple descriptive statistics such as average speed, will average out useful information and cannot capture the data heterogeneity. To capture the heterogeneity in the presence of noise and randomness, we seek an underlying invariant as the scene descriptor. Based on empirical observations, steady space flows, characterized by groups of geometrically similar trajectories, can be observed in many crowd scenes. Each flow is a recurring activity connecting subspaces with designated functionalities, e.g. a flow from the front entrance to the ticket office then to a platform in a train station. Further, this flow reveals certain semantic information, i.e. people buying tickets before going to the platforms. Overall, all flows in a scene form a good basis to describe the crowd activities and the basis is an underlying invariant. How to compute this basis is therefore vital in analysis.

However, computing such a basis is challenging. Naive statistics of trajectories are not descriptive enough because the basis consists of many flows, and is therefore highly heterogeneous and multi-modal. Further the number of flows is not known *a priori*. Since the flows are formed by groups of geometrically similar trajectories/tracklets, a natural solution is to cluster them [Bian et al. 2018]. In this specific research context, unsupervised clustering is needed due to that the shear data volume prohibits human labelling. In unsupervised clustering, popular methods such as K-means and Gaussian Mixture Models [Bishop 2007] require a pre-defined cluster number which is hard to know in advance. Hierarchical Agglomerative Clustering [Kauffman and Rousseeuw 2005] does not require a predefined cluster number, but the user must decide when to stop merging, which is similarly problematic. Spectral-based clustering methods [Shi and

Malik 2000] solve this problem, but require the computation of a similarity matrix whose space complexity is $O(n^2)$ on the number of trajectories. Too much memory is needed for large datasets and performance degrades quickly with increasing matrix size. Due to the afore-mentioned limitations, non-parametric Bayesian approaches were proposed [Wang et al. 2016, 2017]. However, a new approach is still needed because the previous approaches only consider space, and therefore cannot be reused or adapted for our purposes.

We propose a new non-parametric Bayesian method to cluster the trajectories with the time and velocity information in an *unsupervised* fashion, which requires neither manual labelling nor the prior knowledge of cluster number. The outcome of clustering is a series of modes, each being a unique distribution over space, time and speed. Then we propose new methods for data visualization, simulation evaluation and automated simulation guidance.

We first introduce the background of one family of non-parametric Bayesian models, Dirichlet Processes (DPs), and Hierarchical Dirichlet Processes (HDP) (Sec. 4.1). We then introduce our new model Triplet HDPs (Sec. 4.2) and new inference method Chinese Restaurant Franchise League (Sec. 5). Finally new methods are proposed for visualization (Sec. 6.1), comparison (Sec. 6.2) and simulation guidance (Sec. 6.3).

4 OUR METHOD

4.1 Background

Dirichlet Process. To understand DP, imagine there is a multi-modal 1D dataset with five high-density areas (modes). Then a classic five-component Gaussian Mixture Model (GMM) can fit the data via Expectation-Minimization [Bishop 2007]. Now further generalize the problem by assuming that there are an unknown number of high-density areas. In this case, an ideal solution would be to impose a prior distribution which can represent an infinite number of Gaussians, so that the number of Gaussians needed, their means and covariances can be automatically learnt. DP is such a prior.

A DP(γ, H) is a probabilistic measure on measures [Ferguson 1973], with a scaling parameter $\gamma > 0$ and a base probability measure H. A draw from DP, $G \sim DP(\gamma, H)$ is: $G = \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k}$, where $\beta_k \in \boldsymbol{\beta}$ is random and dependent on γ. $\phi_k \in \boldsymbol{\phi}$ is a variable distributed according to H, $\phi_k \sim H$. δ_{ϕ_k} is called an atom at ϕ_k. Specifically for the example problem above, we can define H to be a Normal-Inverse-Gamma (NIG) so that any draw, ϕ_k, from H is a Gaussian, then G becomes an Infinite Gaussian Mixture Model (IGMM) [Rasmussen 1999]. In practice, k is finite and computed during inference.

Hierarchical DPs. Now imagine that the multi-modal dataset in the example problem is observed in separate data groups. Although all the modes can be observed from the whole dataset, only a subset of the modes can be observed in any particular data group. To model this phenomenon, a parent DP is used to capture all the modes with a child DP modelling the modes in each group:

$$G_j \sim DP(\alpha_j, G) \text{ } or \text{ } G_j = \sum_{i=1}^{\infty} \beta_{ji}\delta_{\psi_{ji}} \quad \text{where} \quad G = \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k} \tag{1}$$

where G_j is the modes in the jth data group. α_j is the scaling factor and G is its based distribution. β_{ji} is the weight and $\delta_{\psi_{ji}}$ is the atom. Now we have the Hierarchical DPs, or HDP [Teh et al. 2006] (Fig. 2

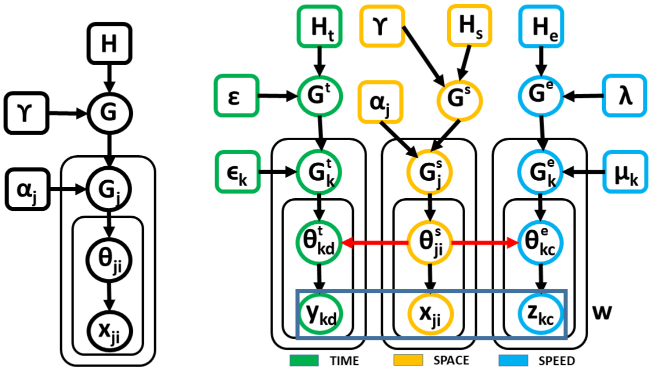

Fig. 2. Left: HDP. Right: Triplet HDP.

Left). At the top level, the modes are captured by $G \sim DP(\gamma, H)$. In each data group j, the modes are captured by G_j which is dependent on α_j and G. This way, the modes, G_j, in every data group come from the common set of modes G, i.e. $\psi_{ji} \in \{\phi_1, \phi_2, ..., \phi_k\}$. In Fig. 2 Left, there is also a variable θ_{ji} called *factor* which indicates with which mode (ψ_{ji} or equally ϕ_k) the data sample x_{ji} is associated. Finally, if H is again a NIG prior, then the HDP becomes Hierarchical Infinite Gaussian Mixture Model (HIGMM).

4.2 Triplet-HDPs (THDP)

We now introduce THDP (Fig. 2 Right). There are three HDPs in THDP, to model space, time and speed. We name them Time-HDP (Green), Space-HDP (Yellow) and Speed-HDP (Blue). Space-HDP is to compute space modes. Time-HDP and Speed-HDP are to compute the time and speed modes associated with each space mode, which requires the three HDPs to be linked. The modeling choice of the links will be explained later. The only observed variable in THDP is w, an observation of a person in a frame. It includes a location-orientation (x_{ji}), timestamp (y_{kd}) and speed (z_{kc}). θ^s_{ji}, θ^t_{kd} and θ^e_{kc} are their factor variables. Given a single observation denoted as w, we denote one trajectory as $\bar{w}$, a group of trajectories as $\check{w}$ and the whole data set as $\boldsymbol{w}$. Our final goal is to compute the space, time and speed modes, given $\boldsymbol{w}$:

$$G^s = \sum_{k=1}^{\infty} \beta_k \delta_{\phi^s_k} \qquad G^t = \sum_{l=1}^{\infty} \zeta_l \delta_{\phi^t_l} \qquad G^e = \sum_{q=1}^{\infty} \rho_q \delta_{\phi^e_q} \tag{2}$$

In THDP, a space mode is defined to be a group of geometrically similar trajectories $\check{w}$. Since these trajectories form a flow, we also refer to it as a space flow. A space flow's timestamps (y_{kd}s) and speed (z_{kc}s) are both 1D data and can be modelled in similar ways. We first introduce the Time-HDP. One space flow $\check{w}$ might appear, crescendo, peak, wane and disappear several times. If a Gaussian distribution is used to represent one time peak on the timeline, multiple Gaussians are needed. Naturally IGMM is used to model the $y_{kd} \in \check{w}$. A possible alternative is to use Poisson Processes to model the entry time. But IGMM is chosen due to its ability to fit complex multi-modal distributions. It can also model a flow for the entire duration. Next, since there are many space flows and the y_{kd}s of each space flow form a timestamp data group, we therefore assume that there is a common set of time peaks shared by all space

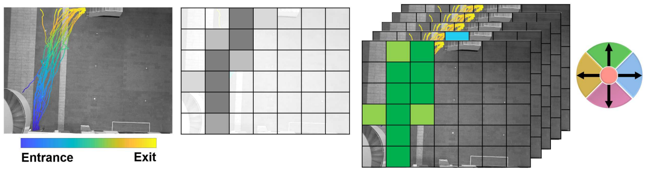

Fig. 3. From left to right: 1. A space flow. 2. Discretization and flow cell occupancy, darker means more occupants. 3. Codebook with normalized occupancy as probabilities indicated by color intensities. 4. Five colored orientation subdomains (Pink indicates static).

flows and each space flow shares only a subset. This way, we use a DP to represent all the time peaks and a child DP below the first DP to represent the peaks in each space flow. This is a HIGMM (for the Time-HDP) where the H_t is a NIG. Similarly for the speed, $z_{kc} \in \check{w}$ can also have multiple peaks on the speed axis, so we use IGMM for this. Further, there are many space flows. We again assume that there is a common set of speed peaks and each space flow only has a subset of these peaks and use another HIGMM for the Speed-TDP.

After Time-HDP and Speed-HDP, we introduce the Space-HDP. The Space-HDP is different because, unlike time and speed, space data (x_{ji}s) is 4D (2D location + 2D orientation), which means its modes are also multi-dimensional. In contrast to time and speed, a 4D Gaussian cannot represent a group of similar trajectories well. So we need to use a different distribution. Similar to [Wang et al. 2017], we discretize the image domain (Fig. 3: 1) into a m × n grid (Fig. 3: 2). The discretization serves three purposes: 1. the cell occupancy serves as a good feature for a flow, since a space flow occupies a fixed group of cells. 2. it removes noises caused by frequent turns and tracking errors. 3. it eliminates the dependence on full trajectories. As long as instantaneous positions and velocities can be estimated, THDP can cluster observations. This is crucial in dealing with real-world data where full trajectories cannot be guaranteed. Next, since there is no orientation information so that the representation cannot distinguish between flows from A-to-B and flows from B-to-A, we discretize the instantaneous orientation into 5 cardinal subdomains (Fig. 3: 4). This makes the grid m × n × 5 (Fig. 3: 3), which now becomes a *codebook* and every 4D x_{ji} can be converted into a cell occupancy. Note although the grid resolution is problem-specific, it does not affect the validity of our method.

Next, since the cell occupancy on the grid (after normalization) can be seen as a Multinomial distribution, we use Multinomials to represent space flows. This way, a space flow has high probabilities in some cells and low probabilities in others (Fig. 3:3). Further, we assume the data is observed in groups and any group could contain multiple flows. We use a DP to model all the space flows of the whole dataset with child DPs representing the flows in individual data groups, e.g. video clips. This is a HDP (Space-HDP) with H_s being a Dirichlet distribution.

After the three HDPs introduced separately, we need to link them, which is the key of THDP. For a space flow $\check{w}_1$, all $x_{ji} \in \check{w}_1$ are associated with the same space mode, denoted by ϕ^s_1, and all $y_{kd} \in \check{w}_1$ are associated with the time modes $\{\phi^t_1\}$ which forms a temporal profile of ϕ^s_1. This indicates that y_{kd}'s time mode association is dependent on x_{ji}'s space mode association. In other words, if $x^1_{ji} \in \check{w}_1$ (ϕ^s_1) and $x^2_{ji} \in \check{w}_2$ (ϕ^s_2), where $x^1_{ji} = x^2_{ji}$ but $\check{w}_1 \neq \check{w}_2$ (two flows can partially overlap), then their corresponding $y^1_{kd} \in \check{w}_1$ and $y^2_{kd} \in \check{w}_2$ should be associated with $\{\phi^t_1\}$ and $\{\phi^t_2\}$ where $\{\phi^t_1\} \neq \{\phi^t_2\}$ when $\check{w}_1$ and $\check{w}_2$ have different temporal profiles. We therefore condition θ^t_{kd} on θ^s_{ji} (The left red arrow in Fig. 2 Right) so that y_{kd}'s time mode association is dependent on x_{ji}'s space mode association. Similarly, a conditioning is also added to θ^e_{kc} on θ^s_{ji}. This way, w's associations to space, time and speed modes are linked. This is the biggest feature that distinguishes THDP from just a simple collection of HDPs, which would otherwise require doing analysis on space, time and dynamics separately, instead of holistically.

5 INFERENCE

Given data $\boldsymbol{w}$, the goal is to compute the posterior distribution $p(\boldsymbol{\beta}$, $\boldsymbol{\phi^s}$, ζ, $\boldsymbol{\phi^t}$, $\boldsymbol{\rho}$, $\boldsymbol{\phi^e} \mid \mathbf{w})$. Existing inference methods for DPs include MCMC [Teh et al. 2006], variational inference [Hoffman et al. 2013] and geometric optimization [Yurochkin and Nguyen 2016]. However, they are designed for simpler models (e.g. a single HDP). Further, both variational inference and geometric optimization suffer from local minimum. We therefore propose a new MCMC method for THDP. The method is a major generalization of Chinese Restaurant Franchise (CRF). Next, we first give the background of CRF, then introduce our method.

5.1 Chinese Restaurant Franchise (CRF)

A single DP has a Chinese Restaurant Process (CRP) representation. CRF is its extension onto HDPs. We refer the readers to [Teh et al. 2006] for details on CRP. Here we directly follow the CRF metaphor on HDP (Eq. 1, Fig. 2 Left) to compute the posterior distribution $p(\boldsymbol{\beta}$, $\boldsymbol{\phi} \mid \mathbf{x})$. In CRF, each observation x_{ji} is called a *customer*. Each data group is called a *restaurant*. Finally, since a customer is associated with a mode (indicated by θ_{ji}), the mode is called a *dish* and is to be learned, as if the customer ordered this dish. CRF dictates that, in every restaurant, there is a potentially infinite number of tables, each with only one dish and many customers sharing that dish. There can be multiple tables serving the same dish. All dishes are on a global menu shared by all restaurants. The global menu can also contain an infinite number of dishes. In summary, we have multiple restaurants with many tables where customers order dishes from a common menu.

CRF is a Gibbs sampling approach. The sampling process is conducted at both customer and table level alternatively. At the customer level, each customer is treated, in turn, as a new customer, given all the other customers sitting at their tables. Then she needs to choose a table in her restaurant. There are two criteria influencing her decision: 1. how many customers are already at the table (*table popularity*) and 2. how much she likes the dish on that table (*dish preference*). If she decides to not sit at any existing table, she can create a new table then order a dish. This dish can be from the menu or she can create a new dish and add it to the menu. Next, at the table-level, for each table, all the customers sitting at that table are treated as a new group of customers, and are asked to choose a dish together. Their collective dish preference and how frequently the dish is ordered in all restaurants (dish popularity) will influence their choice. They can choose a dish from the menu or create a new

```
ALGORITHM 1: Chinese Restaurant Franchise
  Result: $\boldsymbol{\beta}$, $\boldsymbol{\phi}$ (Eq. 1)
1 Input: $\boldsymbol{x}$ ;
2 while Not converged do
3   for every restaurant j do
4     for every customer $x_{ji}$ do
5       Sample a table $t_{ji}$ (Eq. 11, Appx. A);
6       if a new table is chosen then
7         Sample a dish or create a new dish (Eq. 12, Appx. A)
8       end
9     end
10    for every table and its customers $\boldsymbol{x_{jt}}$ do
11      Sample a new dish (Eq. 13, Appx. A)
12    end
13  end
14  Sample hyper-parameters [Teh et al. 2006]
15 end
```

one and add it to the menu. We give the algorithm in Algorithm 1 and refer the readers to Appx. A for more details.

5.2 Chinese Restaurant Franchise League (CRFL)

We generalize CRF by proposing a new method called Chinese Restaurant Franchise League. We first change the naming convention by adding prefixes space-, time- and speed- to customers, restaurant and dishes to distinguish between corresponding variables in the three HDPs. For instance, an observation w now contains a space-customer x_{ji}, a time-customer y_{kd} and a speed-customer z_{kc}. CRFL is a Gibbs sampling scheme, shown in Algorithm 2. The differences between CRF and CRFL are on two levels. At the top level, CRFL generalizes CRF by running CRF alternatively on three HDPs. This makes use of the conditional independence between the Time-HDP and the Speed-HDP given the Space-HDP fixed. At the bottom level, there are **three** major differences in the sampling, between Eq. 11 and Eq. 3, Eq. 12 and Eq. 4, Eq. 13 and Eq. 5.

```
ALGORITHM 2: Chinese Restaurant Franchise League
  Result: $\boldsymbol{\beta}$, $\boldsymbol{\phi^s}$, $\zeta$, $\boldsymbol{\phi^t}$, $\boldsymbol{\rho}$, $\boldsymbol{\phi^e}$ (Eq. 2)
1 Input: $\boldsymbol{w}$ ;
2 while Not converged do
3   Fix all variables in Space-HDP;
4   Do one CRF iteration (line 3-13, Algorithm 1) on Time-HDP;
5   Do one CRF iteration (line 3-13, Algorithm 1) on Speed-HDP;
6   for every space-restaurant j in Space-HDP do
7     for every space-customer $x_{ji}$ do
8       Sample a table $t_{ji}$ (Eq. 3);
9       if a new table is chosen then
10        Sample a dish or create a new dish (Eq. 4);
11      end
12    end
13    for every table and its space-customers $\boldsymbol{x_{jt}}$ do
14      Sample a new space-dish (Eq. 5);
15    end
16  end
17  Sample hyper-parameters (Appx. B.3);
18 end
```

The first difference is when we do customer-level sampling (line 8 in Algorithm 2), the left side of Eq. 11 in CRF becomes:

$$p(t_{ji} = t, x_{ji}, y_{kd}, z_{kc} | \mathbf{x}^{-\mathbf{ji}}, \mathbf{t}^{-\mathbf{ji}}, \mathbf{k}, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l}, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q}) \quad (3)$$

where t_{ji} is the new table for space-customer x_{ji}. y_{kd} and z_{kc} are the time and speed customer. $\mathbf{x}^{-\mathbf{ji}}$ and $\mathbf{t}^{-\mathbf{ji}}$ are the other customers (excluding x_{ji}) in the jth space-restaurant and their choices of tables. $\mathbf{k}$ is the space dishes. Correspondingly, $\mathbf{y}^{-\mathbf{kd}}$ and $\mathbf{o}^{-\mathbf{kd}}$ are the other time-customers (excluding y_{kd}) in the kth time-restaurant and their choices of tables. $\mathbf{l}$ is the time dishes. Similarly, $\mathbf{z}^{-\mathbf{kc}}$ and $\mathbf{p}^{-\mathbf{kc}}$ are the other speed-customers (excluding z_{kc}) in the kth speed-restaurant and their choices of tables. $\mathbf{q}$ is the speed-dishes. The intuitive interpretation of the differences between Eq. 3 and Eq. 11 is: when a space-customer x_{ji} chooses a table, the popularity and preference are not the only criteria anymore. She has to also consider the preferences of her associated time-customer y_{kd} and speed-customer z_{kc}. This is because when x_{ji} orders a different space-dish, y_{kd} and z_{kc} will be placed into a different time-restaurant and speed-restaurant, due to that the organizations of time- and speed-restaurants are dependent on the space-dishes (the dependence of θ^t_{kd} and θ^e_{kc} on θ^s_{ji}). Each space-dish corresponds to a time-restaurant and a speed-restaurant (see Sec. 4.2). Since a space-customer's choice of space-dish can change during CRFL, the organization of time- and speed-restaurants becomes dynamic! This is why CRF cannot be directly applied to THDP.

The second difference is when we need to sample a dish (line 10 in Algorithm 2), the left side of Eq. 12 in CRF becomes:

$$\begin{aligned} p(k_{jt^{new}} = k, x_{ji}, y_{kd}, z_{kc} | \mathbf{k}^{-\mathbf{jt}^{\mathbf{new}}}, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \\ \mathbf{l}, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q}) \propto \\ \begin{cases} m_{\cdot k} p(x_{ji}|\cdots) p(y_{kd}|\cdots) p(z_{kc}|\cdots) \\ \gamma p(x_{ji}|\cdots) p(y_{kd}|\cdots) p(z_{kc}|\cdots) \end{cases} \end{aligned} \quad (4)$$

where $k_{jt^{new}}$ is the new dish for customer x_{ji}. $\cdots$ represents all the conditional variables for simplicity. $p(y_{kd}|\cdots)$ and $p(z_{kc|...})$ are the major differences. We refer the readers to Appx. B regarding the computation of Eq. 3 and Eq. 4.

The last difference is when we do the table-level sampling (line 14 in Algorithm 2), the left side of Eq. 13 in CRF changes to:

$$\begin{aligned} p(k_{jt} = k, \mathbf{x_{jt}}, \mathbf{y_{kd_{jt}}}, \mathbf{z_{kc_{jt}}} | \mathbf{k}^{-\mathbf{jt}}, \mathbf{y}^{-\mathbf{kd_{jt}}}, \mathbf{o}^{-\mathbf{kd_{jt}}}, \\ \mathbf{l}^{-\mathbf{ko}}, \mathbf{z}^{-\mathbf{kc_{jt}}}, \mathbf{p}^{-\mathbf{kc_{jt}}}, \mathbf{q}^{-\mathbf{kp}}) \propto \\ \begin{cases} m^{-jt}_{\cdot k} p(\mathbf{x_{jt}}|\cdots) p(\mathbf{y_{kd_{jt}}}|\cdots) p(\mathbf{z_{kc_{jt}}}|\cdots) \\ \gamma p(\mathbf{x_{jt}}|\cdots) p(\mathbf{y_{kd_{jt}}}|\cdots) p(\mathbf{z_{kc_{jt}}}|\cdots) \end{cases} \end{aligned} \quad (5)$$

where $\mathbf{x_{jt}}$ is the space-customers at the tth table, $\mathbf{y_{kd_{jt}}}$ and $\mathbf{z_{kc_{jt}}}$ are the associated time- and speed-customers. $\mathbf{k}^{-\mathbf{jt}}$, $\mathbf{y}^{-\mathbf{kd_{jt}}}$, $\mathbf{o}^{-\mathbf{kd_{jt}}}$, $\mathbf{l}^{-\mathbf{ko}}$, $\mathbf{z}^{-\mathbf{kc_{jt}}}$, $\mathbf{p}^{-\mathbf{kc_{jt}}}$, $\mathbf{q}^{-\mathbf{kp}}$ are the rest and their table and dish choices in three HDPs. $\cdots$ represents all the conditional variables for simplicity. $p(\mathbf{x_{jt}}|\cdots)$ is the Multinomial f as in Eq. 13. Unlike Eq. 4, $p(\mathbf{y_{kd_{jt}}}|\cdots)$ and $p(\mathbf{z_{kc_{jt}}}|\cdots)$ cannot be easily computed and needs special treatment. We refer the readers to Appx. B for details.

Now we have fully derived CRFL. Given a data set $\mathbf{w}$, we can compute the posterior distribution $p(\boldsymbol{\beta}, \boldsymbol{\phi^s}, \boldsymbol{\zeta}, \boldsymbol{\phi^t}, \boldsymbol{\rho}, \boldsymbol{\phi^e} \mid \mathbf{w})$ where $\boldsymbol{\beta}$, $\boldsymbol{\zeta}$ and $\boldsymbol{\rho}$ are the weights of the space, time and speed dishes, $\boldsymbol{\phi^s}$, $\boldsymbol{\phi^t}$ and $\boldsymbol{\phi^e}$ respectively. $\boldsymbol{\phi^s}$ are Multinomials. $\boldsymbol{\phi^t}$ and $\boldsymbol{\phi^e}$ are Gaussians. As mentioned in Sec. 5.1, the number of $\boldsymbol{\phi^s}$, is automatically learnt, so we do not need to know the space dish number in advance. Neither do we need it for $\boldsymbol{\phi^t}$ and $\boldsymbol{\phi^e}$. This makes THDP non-parametric. Further, since one ϕ^s could be associated with potentially an infinite number of ϕ^ts and ϕ^es and vice versa, the many-to-many associations are also automatically learnt.

5.3 Time Complexity of CRFL

For each sampling iteration in Algorithm 2, the time complexities of sampling on time-HDP, speed-HDP and space-HDP are $O[W(N+L)+KNL]$, $O[W(A+Q)+KAQ]$ and $O[W(M+K)+2W(K+1)\eta+JMK]$ respectively, where $\eta = N + L + A + Q$. W is the total observation number. K, L and Q are the dish numbers of space, time and speed. J is the number of space-restaurants. M, N and A are the average table numbers in space-, time- and speed-restaurants respectively. Note that K appears in all three time complexities because the number of space-dishes is also the number of time- and space-restaurants.

The time complexity of CRFL is $O[W(N + L) + KNL] + O[W(A + Q)+KAQ]+O[W(M+K)+2W(K+1)\eta+JMK]$. This time complexity is not high in practice. W can be large, depending on the dataset, over which a sampling could be used to reduce the observation number. In addition, K is normally smaller than 50 even for highly complex datasets. L and Q are even smaller. J is decided by the user and in the range of 10-30. M, N and A are not large either due to the high aggregation property of DPs, i.e. each table tends to be chosen by many customers, so the table number is low.

6 VISUALIZATION, METRICS AND SIMULATION GUIDANCE BASED ON THDP

THDP provides a powerful and versatile base for new tools. In this section, we present three tools for structured visualization, quantitative comparison and simulation guidance.

6.1 Flexible and Structured Crowd Data Visualization

After inference, the highly rich but originally mixed and unstructured data is now structured. This is vital for visualization. It is immediately easy to visualize the time and speed modes as they are mixtures of univariate Gaussians. The space modes require further treatments because they are m×n×5 Multinomials and hard to visualize. We therefore propose to use them as classifiers to classify trajectories. After classification, we select representative trajectories for a clear and intuitive visualization of flows. Given a trajectory $\bar{w}$, we compute a *softmax* function:

$$p_k(\bar{w}) = \frac{e^{p_k(\bar{w})}}{\sum_{k=1}^{K} e^{p_k(\bar{w})}} \; k \in [1, \mathrm{K}] \qquad (6)$$

where $p_k(\bar{w}) = p(\bar{w}|\beta_k, \phi^s_k, \boldsymbol{\zeta_k}, \boldsymbol{\phi^t}, \boldsymbol{\rho_k}, \boldsymbol{\phi^e})$. ϕ^s_k and β_k are the kth space mode and its weight. The others are the associated time and speed modes. The time and speed modes ($\boldsymbol{\phi^t}$ and $\boldsymbol{\phi^e}$) are associated with space flow ϕ^s_k, with weights, $\boldsymbol{\zeta_k}$ and $\boldsymbol{\rho_k}$. K is the total number of space flows. This way, we classify every trajectory into a space flow. Then we can visualize representative trajectories with high probabilities, or show anomaly trajectories with low probabilities.

In addition, since THDP captures all space, time and dynamics, there is a variety of visualization. A period of time can be represented by a weighted combination of time modes $\{\phi^t\}$. Assuming that the user wants to see what space flows are prominent during this period, we can visualize trajectories based on $\int_{\boldsymbol{\rho},\boldsymbol{\phi^e}} p(\boldsymbol{\beta}, \boldsymbol{\phi^s}|\{\phi^t\})$, which gives the space flows with weights. This is very useful if for instance $\{\phi^t\}$ is rush hours, $\int_{\boldsymbol{\rho},\boldsymbol{\phi^e}} p(\boldsymbol{\beta}, \boldsymbol{\phi^s}|\{\phi^t\})$ shows us what flows are prominent and their relative importance during the rush hours. Similarly, if we visualize data based on $\int_{\zeta,\boldsymbol{\phi^t}} p(\boldsymbol{\rho}, \boldsymbol{\phi^e}|\phi^s)$, it will tell us if people walk fast/slowly on the space flow ϕ^s. A more complex visualization is $p(\boldsymbol{\zeta}, \boldsymbol{\phi^t}, \boldsymbol{\rho}, \boldsymbol{\phi^e}|\phi^s)$ where the time-speed distribution is given for a space flow ϕ^s. This gives the speed change against time of this space flow, which could reveal congestion at times.

Through marginalizing and conditioning on different variables (as above), there are many possible ways of visualizing crowd data and each of them reveals a certain aspect of the data. We do not enumerate all the possibilities for simplicity but it is very obvious that THDP can provide highly flexible and insightful visualizations.

6.2 New Quantitative Evaluation Metrics

Being able to quantitatively compare simulated and real crowds is vital in evaluating the quality of crowd simulation. Trajectory-based [Guy et al. 2012] and flow-based [Wang et al. 2016] methods have been proposed. The first flow-based metrics are proposed in [Wang et al. 2016] which is similar to our approach. In their work, the two metrics proposed were: average likelihood (AL) and distribution-pair distance (DPD) based on Kullback-Leibler (KL) divergence. The underlying idea is that a good simulation does not have to strictly reproduce the data but should have statistical similarities with the data. However, they only considered space. We show that THDP is a major generalization of their work and provides much more flexibility with a set of new AL and DPD metrics.

6.2.1 AL Metrics. Given a simulation data set, $\hat{\boldsymbol{w}} = (\hat{\boldsymbol{x}}_{\boldsymbol{ji}}, \hat{\boldsymbol{y}}_{\boldsymbol{kd}}, \hat{\boldsymbol{z}}_{\boldsymbol{kc}})$ and $p(\boldsymbol{\beta}, \boldsymbol{\phi^s}, \boldsymbol{\zeta}, \boldsymbol{\phi^t}, \boldsymbol{\rho}, \boldsymbol{\phi^e} \mid \mathbf{w})$ inferred from real-world data $\boldsymbol{w}$, we can compute the AL metric based on space only, essentially computing the average space likelihood while marginalizing time and speed:

$$\frac{1}{|\hat{\mathbf{w}}|}\sum_{j,i}\sum_{k=1}^{K}\beta_k \int_z \int_y p(\hat{x}_{ji}|\phi^s_k, \hat{y}_{kd}, \hat{z}_{kc})\; p(\hat{y}_{kd})p(\hat{z}_{kc})dydz \qquad (7)$$

where $|\hat{\boldsymbol{w}}|$ is the number of observations in $\hat{\boldsymbol{w}}$. The dependence on $\boldsymbol{\beta}$, $\boldsymbol{\phi^s}$, $\boldsymbol{\zeta}$, $\boldsymbol{\phi^t}$, $\boldsymbol{\rho}$, $\boldsymbol{\phi^e}$ are omitted for simplicity. If we completely discard time and speed, Eq. 7 changes to the AL metric in [Wang et al. 2017], $\frac{1}{|\hat{\mathbf{w}}|}\sum_{j,i}\sum_k \beta_k p(\hat{x}_{ji}|\phi^s_k)$. However, the metric is just a special case of THDP. We give a list of AL metrics in Table 1, which all have similar forms as Eq. 7.

6.2.2 DPD Metrics. AL metrics are based on average likelihoods, summarizing the differences between two data sets into one number. To give more flexibility, we also propose distribution-pair metrics. We first learn two posterior distributions $p(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\phi}}^{\boldsymbol{s}}, \hat{\boldsymbol{\zeta}}, \hat{\boldsymbol{\phi}}^{\boldsymbol{t}}, \hat{\boldsymbol{\rho}}, \hat{\boldsymbol{\phi}}^{\boldsymbol{e}} \mid \hat{\mathbf{w}})$ and $p(\boldsymbol{\beta}, \boldsymbol{\phi^s}, \boldsymbol{\zeta}, \boldsymbol{\phi^t}, \boldsymbol{\rho}, \boldsymbol{\phi^e} \mid \mathbf{w})$. Then we can compare individual

Metric	To compare
1. $\frac{1}{\lvert\hat{\boldsymbol{w}}\rvert} \sum p(\hat{x}_{ji}, \hat{y}_{kd}, \hat{z}_{kc} \vert \bullet)$	overall similarity
2. $\frac{1}{\lvert\hat{\boldsymbol{w}}\rvert} \sum p(\hat{x}_{ji}, \hat{y}_{kd} \vert \bullet)$	space&time ignoring speed
3. $\frac{1}{\lvert\hat{\boldsymbol{w}}\rvert} \sum p(\hat{x}_{ji}, \hat{z}_{kc} \vert \bullet)$	space&speed ignoring time
4. $\frac{1}{\lvert\hat{\boldsymbol{w}}\rvert} \sum p(\hat{y}_{kd}, \hat{z}_{kc} \vert \bullet)$	time&speed ignoring space
5. $\frac{1}{\lvert\hat{\boldsymbol{w}}\rvert} \sum p(\hat{x}_{ij} \vert \bullet)$	space ignoring time & speed
6. $\frac{1}{\lvert\hat{\boldsymbol{w}}\rvert} \sum p(\hat{y}_{kd} \vert \bullet)$	time ignoring space & speed
7. $\frac{1}{\lvert\hat{\boldsymbol{w}}\rvert} \sum \ p(\hat{z}_{kc} \vert \bullet)$	speed ignoring space & time

Table 1. AL Metrics, • represents $\{\boldsymbol{\beta}, \boldsymbol{\phi}^{\boldsymbol{s}}, \boldsymbol{\zeta}, \boldsymbol{\phi}^{\boldsymbol{t}}, \boldsymbol{\rho}, \boldsymbol{\phi}^{\boldsymbol{e}}\}$.

pairs of ϕ^s and $\hat{\phi}^s$, ϕ^t and $\hat{\phi}^t$, ϕ^e and $\hat{\phi}^e$. Since all space, time and speed modes are probability distributions, we propose to use Jensen-Shannon divergence, as oppose to KL divergence [Wang et al. 2017] due to KL's asymmetry:

$$JSD(P||Q) = \frac{1}{2}D(P||M) + \frac{1}{2}D(Q||M) \tag{8}$$

where D is KL divergence and $M = \frac{1}{2}(P + Q)$. P and Q are probability distributions. Again, in the DPD comparison, THDP provides many options, similar to the AL metrics in Table 1. We only give several examples here. Given two space flows, ϕ^s and $\hat{\phi}^s$, JSD(ϕ^s || $\hat{\phi}^s$) directly compares two space flows. Further, P and Q can be conditional distributions. If we compute JSD($p(\boldsymbol{\phi}^{\boldsymbol{t}} \mid \phi^s)$ || p($\hat{\boldsymbol{\phi}}^{\boldsymbol{t}} \mid \hat{\phi}^s$)) where $\boldsymbol{\phi}^{\boldsymbol{t}}$ and $\hat{\boldsymbol{\phi}}^{\boldsymbol{t}}$ are the associated time modes of ϕ^s and $\hat{\phi}^s$ respectively. This is to compare the two temporal profiles. This is very useful when ϕ^s and $\hat{\phi}^s$ are two spatially similar flows but we want to compare the temporal similarity. Similarly, we can also compare their speed profiles JSD($p(\boldsymbol{\phi}^{\boldsymbol{e}} \mid \phi^s)$ || p($\hat{\boldsymbol{\phi}}^{\boldsymbol{e}} \mid \hat{\phi}^s$)) or their time-speed profiles JSD($p(\boldsymbol{\phi}^{\boldsymbol{t}}, \boldsymbol{\phi}^{\boldsymbol{e}} \mid \phi^s)$ || p($\hat{\boldsymbol{\phi}}^{\boldsymbol{t}}, \hat{\boldsymbol{\phi}}^{\boldsymbol{e}} \mid \hat{\phi}^s$)). In summary, similar to AL metrics, different conditioning and marginalization choices result in different DPD metrics.

6.3 Simulation Guidance

We propose a new method to automate simulation guidance with real-world data, which works with existing simulators including steering and global planning methods. Assuming that we want to simulate crowds in a given environment based on data, there are still several key parameters which need to be estimated including, starting/destination positions, the entry timing and the desired speed. After inferring, we use GMM to model both starting and destination regions for every space flow. This way, we completely eliminate the need for manual labelling, which is difficult in spaces with no designated entrances/exits (e.g. a square). Also, we removed the one-to-one mapping requirement of the agents in simulation and data. We can sample any number of agents based on space flow weights ($\boldsymbol{\beta}$) and still keep similar agent proportions on different flows to the data. In addition, since each flow comes with a temporal and speed profile, we sample the entry timing and desired speed for each agent, to mimic the randomness in these parameters. It is difficult to manually set the timing when the duration is long and sampling the speed is necessary to capture the speed variety within a flow caused by latent factors such as different physical conditions.

Next, even with the right setting of all the afore-mentioned parameters, existing simulators tend to simulate straight lines whenever possible while the real data shows otherwise. This is due to that no intrinsic motion randomness is introduced. Intrinsic motion randomness can be observed in that people rarely walk in straight lines and they generate slightly different trajectories even when asked to walk several times between the same starting position and destination [Wang et al. 2017]. This is related to the state of the person as well as external factors such as collision avoidance. Individual motion randomness can be modelled by assuming the randomness is Gaussian-distributed [Guy et al. 2012]. Here, we do not assume that all people have the same distribution. Instead, we propose to do a structured modelling. We observe that people on different space flows show different dynamics but share similar dynamics within the same flow. This is because people on the same flow share the same starting/destination regions and walk through the same part of the environment. In other words, they started in similar positions, had similar goals and made similar navigation decisions. Although individual motion randomness still exists, their randomness is likely to be similarly distributed. However, this is not necessarily true across different flows. We therefore assume that each space flow can be seen as generated by a unique dynamic system which captures the within-group motion randomness which implicitly considers factors such as collision avoidance. Given a trajectory, $\bar{w}$, from a flow $\check{w}$, we assume that there is an underlying dynamic system:

$$\begin{aligned} x_t^{\bar{w}} &= As_t + \omega_t \quad \omega \sim N(0, \Omega) \\ s_t &= Bs_{t-1} + \lambda_t \quad \lambda \sim N(0, \Lambda) \end{aligned} \tag{9}$$

where $x_t^{\bar{w}}$ is the observed location of a person at time t on trajectory $\bar{w}$. s_t is the latent state of the dynamic system at time t. ω_t and λ_t are the observational and dynamics randomness. Both are white Gaussian noises. A and B are transition matrices. We assume that Ω is a known diagonal covariance matrix because it is intrinsic to the device (e.g. a camera) and can be trivially estimated. We also assume that A is an identity matrix so that there is no systematic bias and the observation is only subject to the state s_t and noise ω_t. The dynamic system then becomes: $x_t^{\bar{w}} \sim N(Is_t, \Omega)$ and $s_t \sim N(Bs_{t-1}, \Lambda)$, where we need to estimate s_t, B and Λ. Given the U trajectories in $\check{w}$, the total likelihood is:

$$\begin{aligned} &p(\check{w}) = \Pi_{i=1}^{U} p(\bar{w}_i) \quad \text{where} \\ &p(\bar{w}_i) = \Pi_{t=2}^{T_i - 1} p(x_t^i | s_t) P(s_t | s_{t-1}) \quad s_1 = x_1^i, s_T = x_{T_i}^i \end{aligned} \tag{10}$$

where T_i is the length of trajectory $\bar{w}_i$. We maximize $log\ P(\check{w})$ via Expectation-Maximization [Bishop 2007]. Details can be found in the Appx. C. After learning the dynamic system for a space flow and given a starting and destination location, s_1 and s_T, we can sample diversified trajectories while obeying the flow dynamics. During simulation guidance, one target trajectory is sampled for each agent and this trajectory reflects the motion randomness.

7 EXPERIMENTS

In this section, we first introduce the datasets, then show our highly informative and flexible visualization tool. Next, we give quantitative comparison results between simulated and real crowds by the newly proposed metrics. Finally, we show that our automated simulation

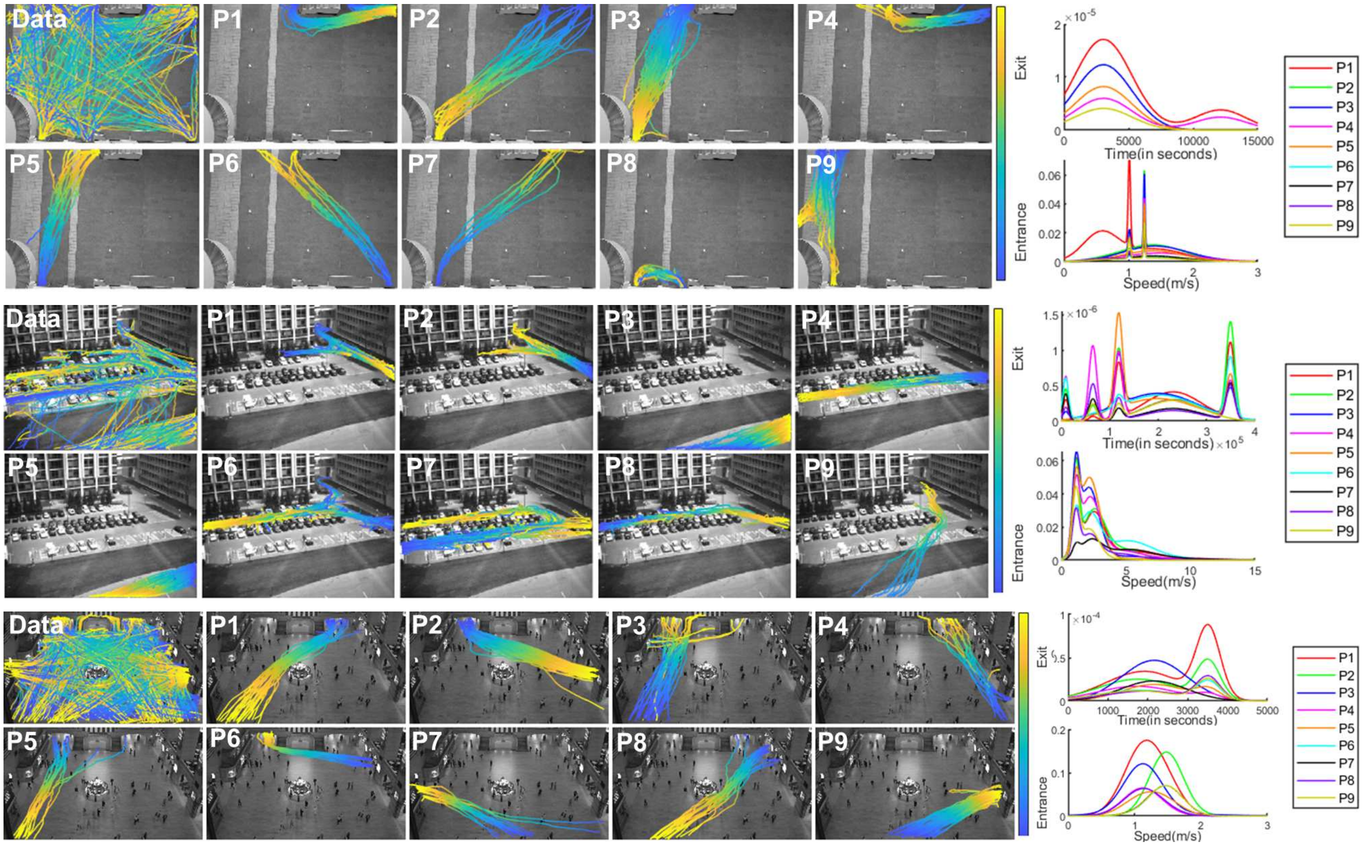

Fig. 4. Forum (top), CarPark (Middle) and TrainStation (Bottom) dataset. In each dataset, Top left: original data; P1-P9: the top 9 space modes; Top right: the time modes of P1-P9; Bottom right: the speed modes of P1-P9. Both time and speed profiles are scaled by their respective space model weights, with the y axis indicating the likelihood.

guidance with high semantic fidelity. We only show representative results in the paper and refer the readers to the supplementary video and materials for details.

7.1 Datasets

We choose three publicly available datasets: **Forum** [Majecka 2009], **CarPark** [Wang et al. 2008] and **TrainStation** [Yi et al. 2015], to cover different data volumes, durations, environments and crowd dynamics. Forum is an indoor environment in a school building, recorded by a top-down camera, containing 664 trajectories and lasting for 4.68 hours. Only people are tracked and they are mostly slow and casual. CarPark consists of videos of an outdoor car park with mixed pedestrians and cars, by a far-distance camera and contains totally 40,453 trajectories over five days. TrainStation is a big indoor environment with pedestrians and designated sub-spaces. It is from New York Central Terminal and contains totally 120,000 frames with 12,684 pedestrians within approximately 45 minutes. The speed varies among pedestrians.

7.2 Visualization Results

We first show a general, full-mode visualization in Fig. 4. Due to the space limit, we only show the top 9 space modes and their corresponding time and speed profiles. Overall, THDP is effective in decomposing highly mixed and unstructured data into structured results across different data sets. The top 9 space modes (with time and speed) are the main activities. With the environment information (e.g. where the doors/lifts/rooms are), the semantic meanings of the activities can be inferred. In addition, the time and dynamics are captured well. One peak of a space flow (indicated by color) in the time profiles indicates that this flow is likely to appear around that time. Correspondingly, one peak of a space flow in the speed profile indicates a major speed preference of the people on that flow. Multiple space flows can peak near one point in both the time and speed profiles. The speed profiles of Forum and TrainStation are slightly different, with most of the former distributed in a smaller region. This is understandable because people in TrainStation in general walk faster. The speed profile of CarPark is quite different in that it ranges more widely, up to 10m/s. This is because both pedestrians and vehicles were recorded.

Besides, we show conditioned visualization. Suppose that the user is interested in a period (e.g. rush hours) or speed range (e.g.

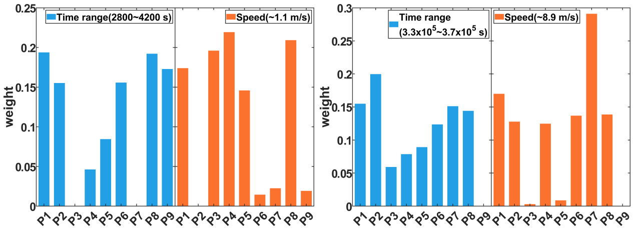

Fig. 5. Left: TrainStation, Right: CarPark. The space flow prominence (indicated by bar heights) of P1-P9 in Fig. 4 respectively given a time period (blue bars) or speed range (orange bars). The higher the bar, the more prominent the space flow is.

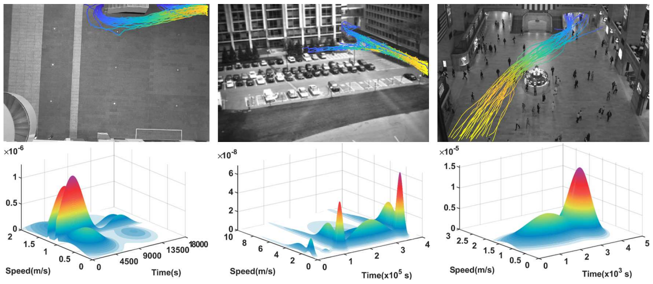

Fig. 6. Space flows from Forum, CarPark and TrainStation and their time-speed distributions. The y (up) axis is likelihood. The x and z axes are time and speed. The redder, the higher the likelihood is.

to see where people generally walk fast/slowly), the associated flow weights can be visualized (Fig. 5). This allows users to see which space flows are prominent in the chosen period or speed range. Conversely, given a space flow in interest, we can visualize the time-speed distribution (Fig. 6), showing how the speed changes along time, which could help identify congestion on that flow at times.

Last but not least, we can identify anomaly trajectories and show unusual activities. The anomalies here refer to statistical anomalies. Although they are not necessarily suspicious behaviors or events, they can help the user to quickly reduce the number of cases needed to be investigated. Note that the anomaly is not only the spacial anomaly. It is possible that a spatially normal trajectory that is abnormal in time and/or speed. To distinguish between them, we first compute the probabilities of all trajectories and select anomalies. Then for each anomaly trajectory, we compute its relative probabilities (its probability divided by the maximal trajectory probability) in space, time and speed, resulting in three probabilities in [0, 1]. Then we use them (after normalization) as the bary-centric coordinates of a point inside of a colored triangle. This way, we can visualize what contributes to their abnormality (Fig. 7). Take T1 for example. It has a normal spacial pattern, and therefore is close to the 'space' vertex. It is far away from both 'time' and 'speed' vertex, indicating T1's time and speed patterns are very different from the others'. THDP can be used as a versatile and discriminative anomaly detector.

Non-parametric Bayesian approaches have been used for crowd analysis [Wang et al. 2016, 2017]. However, existing methods can be seen as variants of the Space-HDP and cannot decompose information in time and dynamics. Consequently, they cannot show any results related to time & speed, as opposed to Fig. 4-7. A naive alternative would be to use the methods in [Wang et al. 2016, 2017] to first cluster data regardless time and dynamics, then do per-cluster time and dynamics analysis, equivalent to using the Space-HDP first, then the time-HDP & Speed-HDP subsequently. However, this kind of sequential analysis has failed due to one limitation: the spatial-only HDP misclassifies observations in the overlapped areas of flows [Wang and O'Sullivan 2016]. The following time and dynamics analysis would be based on wrong clustering. The simultaneity of considering all three types of information, accomplished by the links (red arrows in Fig. 2 Right) among three HDPs in THDP, is therefore essential.

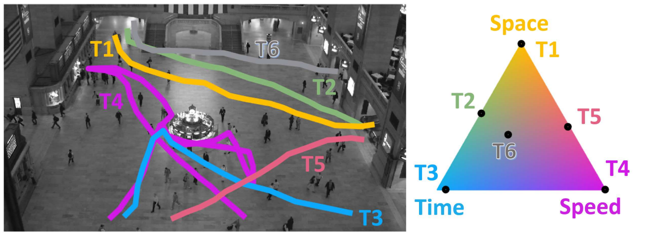

Fig. 7. Representative anomaly trajectories. Every trajectory has a corresponding location in the triangle on the right, indicating what factors contribute more in its abnormality. For instance, T1 is close to the space vertex, it means its spatial probability is relatively high and the main abnormality contribution comes from its time and speed. For T2, the contribution mainly comes from its speed.

7.3 Compare Real and Simulated Crowds

To compare simulated and real crowds, we ask participants (Master and PhD students whose expertise is in crowd analysis and simulation) to simulate crowds in Forum and TrainStation. We left CarPark out because its excessively long duration makes it extremely difficult for participants to observe. We built a simple UI for setting up simulation parameters including starting/destination locations, the entry timing and the desired speed for every agent. For simulator, our approach is agnostic about simulation methods. We chose ORCA in Menge [Curtis et al. 2016] for our experiments but other simulation methods would work equally well. Initially, we provide the participants with only videos and ask them to do their best to replicate the crowd motions. They found it difficult because they had to watch the videos and tried to remember a lot of information, which is also a real-world problem of simulation engineers. This suggests that different levels of detail of the information are needed to set up simulations. The information includes variables such as entry timings and start/end positions, which are readily available, or descriptive statistics such as average speed, which can be relatively easily computed. We systematically investigate their roles in producing scene semantics. After several trials, we identified a set of key parameters including starting/ending positions, entry timing and desired speed. Different simulation methods require different parameters, but these are the key parameters shared by all. We also identified four typical settings where we gradually provide more and more information about these parameters. This design helps us to identify the qualitative and quantitative importance of the key parameters for the purpose of reproducing the scene semantics.

The first setting, denoted as Random, is where only the starting/destination regions are given. The participants have to estimate

Information / Setting	Random	SDR	SDRT	SDRTS
Starting/Dest. Areas	✓	✓	✓	✓
Exact Starting/Dest. Positions	×	✓	✓	✓
Trajectory Entry Timing	×	×	✓	✓
Trajectory Average Speed	×	×	×	✓

Table 2. Different simulation settings and the information provided.

Metric/Simulations	Random	SDR	SDRT	SDRTS	Ours
Overall ($\times10^{-8}$)	7.11	20.67	37.08	40.55	**57.9**
Space-Only ($\times10^{-3}$)	2.7	5.3	5.3	5.5	5.1
Space-Time ($\times10^{-7}$)	1.23	2.96	5.56	5.77	**6.02**
Space-Speed ($\times10^{-3}$)	1.5	3.6	3.5	4.0	**4.9**
Overall ($\times10^{-7}$)	6.7	11.97	13.96	19.39	**19.89**
Space-Only ($\times10^{-3}$)	3.5	6.8	6.7	6.6	**6.9**
Space-Time ($\times10^{-7}$)	8.02	15.87	19.00	18.84	**20.44**
Space-Speed ($\times10^{-3}$)	2.9	5.0	4.9	6.9	6.7

Table 3. Comparison on Forum (Top) and TrainStation (Bottom) based on AL metrics. **Higher** is better. Numbers should only compared within the same row.)

the rest. Based on Random, we further give the exact starting/ending positions, denoted by SDR. Next, we also give the entry timing for each agent based on SDR, denoted by SDRT. Finally, we give the average speed of each agent based on SDRT, denoted by SDRTS. Random is the least-informed scenario where the users have to estimate many parameters, while SDRTS is the most-informed situation. A comparison between the four settings is shown in Table 2.

We use four AL metrics to compare simulations with data, as they provide detailed and insightful comparisons: Overall (Table 1: 1), Space-Only (Table 1: 5), Space-Time (Table 1: 2) and Space-Speed (Table 1: 3) and show the comparisons in Table 3. In Random, the users had to guess the exact entrance/exit locations, entry timing and speed. It is very difficult to do by just watching videos and thus has the lowest score across the board. When provided with exact entrance/exit locations (SDR), the score is boosted in Overall and Space-Only. But the scores in Space-Time and Space-Speed remain relatively low. As more information is provided (SDRT & SDRTS), the scores generally increase. This shows that our metrics are sensitive to space, time and dynamics information during comparisons. Further, each type of information is isolated out in the comparison. The Space-Only scores are roughly the same between SDR, SDRT and SDRTS. The Space-Time scores do not change much between SDRT and SDRTS. The isolation in comparisons makes our AL metrics ideal for evaluating simulations in different aspects, providing great flexibility which is necessary in practice.

Next, we show that it is possible to do more detailed comparisons using DPD metrics. Due to the space limit, we show one space flow from all simulation settings (Fig. 8), and compare them in space only (DPD-Space), time only (DPD-Time) and time-speed (DPD-TS) in Table 4. In DPD-Space, all settings perform similarly because the space information is provided in all of them. In DPD-Time, SDRT & SDRTS are better because they are both provided with the timing information. What is interesting is that SDRTS is worse than SDRT on the two flows in DPD-TS. Their main difference is that the desired speed in SDRTS is set to be the average speed of that trajectory, while the desired speed in SDRT is randomly drawn from a Gaussian estimated from real data. The latter achieves a slightly better performance on both flows in DPD-TS.

Metric/Simulations	SDR	SDRT	SDRTS	Ours
DPD-Space	0.4751	0.3813	0.4374	**0.2988**
DPD-Time	0.3545	0.0795	0.064	**0.0419**
DPD-TS	1.0	0.8879	1.0	**0.4443**
DPD-Space	0.2753	0.2461	0.2423	**0.1173**
DPD-Time	0.0428	0.0319	0.0295	**0.0213**
DPD-TS	0.9970	0.8157	0.9724	**0.5091**

Table 4. Comparison on space flow P2 in Forum (Top) and space flow P1 in TrainStation (Bottom) based on DPD metrics, both shown in Fig. 4. **Lower** is better.

Quantitative metrics for comparing simulated and real crowds have been proposed before. However, they either only compare individual motions [Guy et al. 2012] or only space patterns [Wang et al. 2016, 2017]. Holistically considering space, time & speed has a combinatorial effect, leading to many explicable metrics evaluating different aspects of crowds (AL & DPD metrics). This makes multi-faceted comparisons possible, which is unachievable in existing methods. Technically, the flexible design of THDP allows for different choices of marginalization, which greatly increases the evaluation versatility. This shows the theoretical superiority of THDP over existing methods.

7.4 Guided Simulations

Our automated simulation guidance proves to be superior to careful manual settings. We first show the AL results in Table 3. Our guided simulation outperforms all other settings that were carefully and manually set up. The superior performance is achieved in the Overall comparisons as well as most dimension-specific comparisons. Next, we show the same space flow of our guided simulation in Fig. 8, in comparison with other settings. Qualitatively, SDR, SDRT and SDRTS generate narrower flows due to straight lines are simulated. In contrast, our simulation shows more realistic intra-flow randomness which led to a wider flow. It is much more similar to the real data. Quantitatively, we show the DPD results in Table 4. Again, our automated guidance outperforms all other settings.

Automated simulation guidance has only been attempted by a few researchers before [Karamouzas et al. 2018; Wolinski et al. 2014]. However, their methods aim to guide simulators to reproduce low-level motions for the overall similarity with the data. Our approach aims to inform simulators with structured scene semantics. Moreover, it gives the freedom to the users so that the full semantics or partial semantics (e.g. the top n flows) can be used to simulate crowds, which no previous method can provide.

7.5 Implementation Details

For space discretization, we divide the image space of Forum, CarPark and TrainStation uniformly into 40 × 40, 40 × 40 and 120 × 120 pixel grids respectively. Since Forum is recorded by a top-down camera, we directly estimate the velocity from two consecutive observations in time. For CarPark and TrainStation, we estimate the velocity by reconstructing a top-down view via perspective projection. THDP also has hyper-parameters such as the scaling factors of every DP

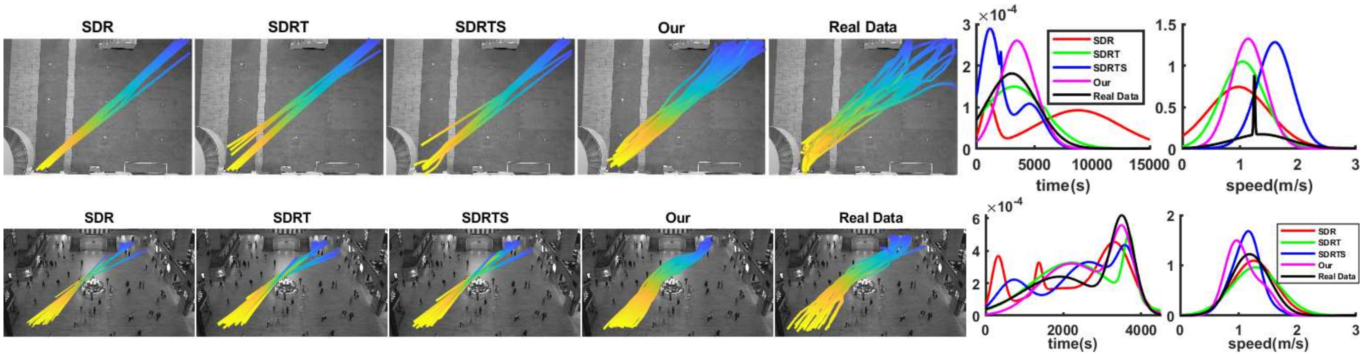

Fig. 8. Space flow P2 in Forum (Top) and P1 in TrainStation (Bottom) in different simulations. The y axes of the time and speed profiles indicate likelihood.

(totally 6 of them). Our inference method is not very sensitive to them because they are also sampled, as part of the CRFL sampling. Please refer to Appx. B.3 for details. In inference, we have a burn-in phase, during which we only use CRF on the Space-HDP and ignore the rest two HDPs. After the burn-in phase, we use CRFL on the full THDP. We found that it can greatly help the convergence of the inference. For crowd simulation, we use ORCA in Menge [Curtis et al. 2016].

We randomly select 664 trajectories in Forum, 1000 trajectories in CarPark and 1000 trajectories in Trainstation for performance tests. In each experiment, we split the data into segments in time domain to mimic fragmented video observations. The number of segments is a user-defined hyper-parameter and depends on the nature of the dataset. We chose the segment number to be 384, 87 and 28, for Forum, CarPark and TrainStation respectively to cover situations where the video is finely or roughly segmented. During training, we first run 5k CRF iterations on the Space-HDP only in the burn-in phase, then do the full CRFL on the whole THDP to speed up the mixing. After training, the numbers of space, time and speed modes are 25, 5 and 7 in Forum; 13, 6 and 6 in CarPark; 16, 3 and 4 in TrainStation. The training took 85.1, 11.5 and 7.8 minutes on Forum, Carpark and TrainStation, on a PC with an Intel i7-6700 3.4GHz CPU and 16GB memory.

8 DISCUSSION

We chose MCMC to avoid the local minimum issue. (Stochastic) Variational Inference (VI) [Hoffman et al. 2013] and Geometric Optimization [Yurochkin and Nguyen 2016] are theoretically faster. However, VI for a single HDP is already prone to local minimum [Wang et al. 2016]. We also found the same issue with geometric optimization. Also, can we use three independent HDPs? Using independent HDPs essentially breaks the many-to-many associations between space, time and speed modes. It can cause mis-clustering due to that the clustering is done on different dimensions separately [Wang and O'Sullivan 2016].

The biggest limitation of our method does not consider the cross-scene transferability. Since the analysis focuses on the semantics in a given scene, it is unclear how the results can inspire simulation settings in unseen environments. In addition, our metrics do not directly reflect visual similarities on the individual level. We deliberately avoid the agent-level one-to-one comparison, to allow greater flexibility in simulation setting while maintaining statistical similarities. Also, we currently do not model high-level behaviors such as grouping, queuing, etc. This is due to that such information can only be obtained through human labelling which would incur massive workload and be therefore impractical on the chosen datasets. We intentionally chose unsupervised learning to deal with large datasets.

9 CONCLUSIONS AND FUTURE WORK

In this paper, we present the first, to our best knowledge, multi-purpose framework for comprehensive crowd analysis, visualization, comparison (between real and simulated crowds) and simulation guidance. To this end, we proposed a new non-parametric Bayesian model called Triplet-HDP and a new inference method called Chinese Restaurant Franchise League. We have shown the effectiveness of our method on datasets varying in volume, duration, environment and crowd dynamics.

In the future, we would like to extend the work to cross-environment prediction. It would be ideal if the modes learnt from given environments can be used to predict crowd behaviors in unseen environments. Preliminary results show that the semantics are tightly coupled with the layout of sub-spaces with designated functionalities. This means a subspace-functionality based semantic transfer is possible. Besides, we will look into using semi-supervised learning to identify and learn high level social behaviors, such as grouping and queuing.

ACKNOWLEDGEMENT

The project is partially supported by EPSRC (Ref:EP/R031193/1), the Fundamental Research Funds for the Central Universities (xzy012019048) and the National Natural Science Foundation of China (61602366).

REFERENCES

Saad Ali and Mubarak Shah. 2007. A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis. In 2007 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 1–6.

Jiang Bian, Dayong Tian, Yuanyan Tang, and Dacheng Tao. 2018. A survey on trajectory clustering analysis. CoRR abs/1802.06971 (2018). arXiv:1802.06971

Christopher Bishop. 2007. Pattern Recognition and Machine Learning. Springer, New York.

Rima Chaker, Zaher Al Aghbari, and Imran N Junejo. 2017. Social network model for crowd anomaly detection and localization. Pattern Recognition 61 (2017), 266–281.

Panayiotis Charalambous, Ioannis Karamouzas, Stephen J Guy, and Yiorgos Chrysanthou. 2014. A data-driven framework for visual crowd analysis. In Computer Graphics Forum, Vol. 33. Wiley Online Library, 41–50.

Sean Curtis, Andrew Best, and Dinesh Manocha. 2016. Menge: A Modular Framework for Simulating Crowd Movement. Collective Dynamics 1, 0 (2016).

Cathy Ennis, Christopher Peters, and Carol O'Sullivan. 2011. Perceptual Effects of Scene Context and Viewpoint for Virtual Pedestrian Crowds. ACM Transaction on Applied Perception 8, 2, Article Article 10 (Feb. 2011), 22 pages.

Thomas S. Ferguson. 1973. A Bayesian Analysis of Some Nonparametric Problems. The Annals of Statistics 1, 2 (1973), 209–230.

Abhinav Golas, Rahul Narain, and Ming Lin. 2013. Hybrid Long-range Collision Avoidance for Crowd Simulation. In ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. 29–36.

Stephen J. Guy, Jur van den Berg, Wenxi Liu, Rynson Lau, Ming C. Lin, and Dinesh Manocha. 2012. A Statistical Similarity Measure for Aggregate Crowd Dynamics. ACM Transaction on Graphics 31, 6 (2012), 190:1–190:11.

Dirk Helbing et al. 1995. Social Force Model for Pedestrian Dynamics. Physical Review E (1995).

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. 2013. Stochastic Variational Inference. Journal of Machine Learning Research 14, 1 (2013), 1303–1347.

Kevin Jordao, Julien Pettré, Marc Christie, and Marie-Paule Cani. 2014. Crowd Sculpting: A Space-time Sculpting Method for Populating Virtual Environments. Computer Graphics Forum (2014).

Ioannis Karamouzas, Nick Sohre, Ran Hu, and Stephen J. Guy. 2018. Crowd Space: A Predictive Crowd Analysis Technique. ACM Transaction on Graphics 37, 6, Article Article 186 (Dec. 2018), 14 pages.

Leonard Kauffman and Peter J. Rousseeuw. 2005. Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley & Sons.

Kang Hoon Lee, Myung Geol Choi, Qyoun Hong, and Jehee Lee. 2007. Group behavior from video: a data-driven approach to crowd simulation. In Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation. 109–118.

S. Lemercier, A. Jelic, R. Kulpa, J. Hua, J. Fehrenbach, P. Degond, C. Appert-Rolland, S. Donikian, and J. Pettré. 2012. Realistic Following Behaviors for Crowd Simulation. Computer Graphics Forum 31, 2 (2012), 489–498.

Alon Lerner, Yiorgos Chrysanthou, Ariel Shamir, and Daniel Cohen-Or. 2009. Data driven evaluation of crowds. In International Workshop on Motion in Games. Springer, 75–83.

Ning Lu et al. 2019. ADCrowdNet: An Attention-injective Deformable Convolutional Networkfor Crowd Understanding. IEEE Conference on Computer Vision and Pattern Recognition (2019).

A López, F Chaumette, E Marchand, and J Pettré. 2019. Character navigation in dynamic environments based on optical flow. In Proceedings of Eurographics 2019 (Eurographics 2019). Eurographics.

B. Majecka. 2009. Statistical models of pedestrian behaviour in the Forum. MSc Dissertation. School of Informatics, University of Edinburgh, Edinburgh.

Ramin Mehran, Alexis Oyama, and Mubarak Shah. 2009. Abnormal crowd behavior detection using social force model. In 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 935–942.

Rahul Narain, Abhinav Golas, Sean Curtis, and Ming C. Lin. 2009. Aggregate Dynamics for Dense Crowd Simulation. ACM Transaction on Graphics 28, 5 (2009), 122:1–122:8.

Carl Edward Rasmussen. 1999. The Infinite Gaussian Mixture Model. In International Conference on Neural Information Processing Systems (NIPS'99). MIT Press, Cambridge, MA, USA, 554–560.

Jiaping Ren, Wei Xiang, Yangxi Xiao, Ruigang Yang, Dinesh Manocha, and Xiaogang Jin. 2018. Heter-Sim: Heterogeneous multi-agent systems simulation by interactive data-driven optimization. CoRR abs/1812.00307 (2018). arXiv:1812.00307

Zeng Ren, P. Charalambous, J. Bruneau, Q. Peng, and J. Pettré. 2016. Group modelling: A unified velocity-based approach. Computer Graphics Forum (2016).

Mohammad Sabokrou et al. 2017. Deep-cascade:cascading 3D deep neural networks for fast anomaly detection and localization in crowded scenes. IEEE Transaction on Image Processing (2017).

Long Sha, Patrick Lucey, Yisong Yue, Xinyu Wei, Jennifer Hobbs, Charlie Rohlf, and Sridha Sridharan. 2018. Interactive sports analytics: An intelligent interface for utilizing trajectories for interactive sports play retrieval and analytics. ACM Transactions on Computer-Human Interaction (TOCHI) 25, 2 (2018), 1–32.

Long Sha, Patrick Lucey, Stephan Zheng, Taehwan Kim, Yisong Yue, and Sridha Sridharan. 2017. Fine-grained retrieval of sports plays using tree-based alignment of trajectories. (2017). arXiv:1710.02255

Yijun Shen, Joseph Henry, He Wang, Edmond S. L. Ho, Taku Komura, and Hubert P. H. Shum. 2018. Data-Driven Crowd Motion Control With Multi-Touch Gestures. Computer Graphics Forum 37, 6 (2018), 382–394. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13333

Jianbo Shi and J. Malik. 2000. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 8 (2000), 888–905.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical Dirichlet Processes. Journal of American Statistical Association 101, 476 (2006), 1566–1581.

J. van den Berg, Ming C. Lin, and Dinesh Manocha. 2008. Reciprocal Velocity Obstacles for real-time multi-agent navigation. IEEE International Conference on Robotics and Automation (2008).

He Wang, Jan Ondřej, and Carol O'Sullivan. 2016. Path Patterns: Analyzing and Comparing Real and Simulated Crowds. In Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D '16). ACM, New York, NY, USA, 49–57. https://doi.org/10.1145/2856400.2856410

He Wang, Jan Ondřej, and Carol O'Sullivan. 2017. Trending Paths: A New Semantic-level Metric for Comparing Simulated and Real Crowd Data. IEEE Transactions on Visualization and Computer Graphics 23, 5 (2017), 1454–1464.

He Wang and Carol O'Sullivan. 2016. Globally Continuous and Non-Markovian Crowd Activity Analysis from Videos. Springer International Publishing, Cham, 527–544.

Qi Wang et al. 2019. Learning from Synthetic Data for Crowd Counting in the Wild. IEEE Conference on Computer Vision and Pattern Recognition (2019).

Xiaogang Wang, Keng Teck Ma, Gee-Wah Ng, and W. E. L. Grimson. 2008. Trajectory analysis and semantic region modeling using a nonparametric Bayesian model. In IEEE Conference on Computer Vision and Pattern Recognition. 1–8.

David Wolinski, Stephen J. Guy, Anne-Hélène Olivier, Ming C. Lin, Dinesh Manocha, and Julien Pettré. 2014. Parameter estimation and comparative evaluation of crowd simulations. Computer Graphics Forum 33, 2 (2014), 303–312.

Yanyu Xu et al. 2018. Encoding Crowd Interaction with Deep Neural Network for Pedestrian Trajectory Prediction. IEEE Conference on Computer Vision and Pattern Recognition (2018).

S. Yi, H. Li, and X. Wang. 2015. Understanding pedestrian behaviors from stationary crowd groups. In IEEE Conference on Computer Vision and Pattern Recognition. 3488–3496.

Mikhail Yurochkin and XuanLong Nguyen. 2016. Geometric Dirichlet Means Algorithm for topic inference. In International Conference on Neural Information Processing Systems.

A CHINESE RESTAURANT FRANCHISE

To give the mathematical derivation of the sampling process described in Sec. 5.1, we first give meanings to the variables in Fig. 2 Left. θ_{ji} is the dish choice made by x_{ji}, the ith customer in the jth restaurant. G_j is the tables with dishes and the dishes are from the global menu G. Since θ_{ji} indicates the choice of tables and therefore dishes, we use some auxiliary variables to represent the process. We introduce t_{ji} and k_{jt} as the indices of the table and the dish on the table chosen by x_{ji}. We also denote m_{jk} as the number of tables serving the kth dish in restaurant j and n_{jtk} as the number of customers at table t in restaurant j having the kth dish. We also use them to represent accumulative indicators such as $m_{\cdot k}$ representing the total number of tables serving the kth dish. We also use superscript to indicate which customer or table is removed. If customer x_{ji} is removed, then n_{jtk}^{-ji} is the number of customers at the table t in restaurant j having the kth dish without the customer x_{ji}.

Customer-level sampling. To choose a table for x_{ji} (line 5 in Algorithm 1), we sample a table index t_{ji}:

$$p(t_{ji} = t|\mathbf{t}^{\mathbf{-ji}}, \mathbf{k}) \propto \begin{cases} n_{jt\cdot}^{-ji} f_{k_{jt}}^{-x_{ji}}(x_{ji}) \text{ if } t \text{ already exists} \\ \alpha_j p(x_{ji}|\mathbf{t}^{\mathbf{-ji}}, t_{ji} = t^{new}, \mathbf{k}) \text{ if } t = t^{new} \end{cases} \tag{11}$$

where $n_{jt\cdot}^{-ji}$ is the number of customers at table t (table popularity), and $f_{k_{jt}}^{-x_{ji}}(x_{ji})$ is how much x_{ji} likes the k_{jt}th dish, $f_{k_{jt}}$, served on that table (dish preference). $f_{k_{jt}}$ is the dish and thus is a problem-specific probability distribution. $f_{k_{jt}}^{-x_{ji}}(x_{ji})$ is the likelihood of x_{ji} on $f_{k_{jt}}$. In our problem, $f_{k_{jt}}$ is Multinomial if it is the Space-HDP or otherwise Normal. α_j is the parameter in Eq. 1, so it controls how likely x_{ji} will create a new table, after which she needs to choose a dish according to $p(x_{ji}|\mathbf{t}^{\mathbf{-ji}}, t_{ji} = t^{new}, \mathbf{k})$. When a new table is created, $t_{ji} = t^{new}$, we need sampling a dish (line 7 in Algorithm 1), indexed by $k_{jt^{new}}$, according to:

$$p(k_{jt^{new}} = k|\mathbf{t}, \mathbf{k}^{-\mathbf{jt}^{\mathbf{new}}}) \propto \begin{cases} m_{\cdot k} f_k^{-x_{ji}}(x_{ji}) \text{ if } k \text{ already exists} \\ \gamma f_{k^{new}}^{-x_{ji}}(x_{ji}) \text{ if } k = k^{new} \end{cases} \tag{12}$$

where $m_{\cdot k}$ is the total number of tables across all restaurants serving the kth dish (dish popularity). $f_k^{-x_{ji}}(x_{ji})$ is how much x_{ji} like the kth dish, again the likelihood of x_{ji} on f_k. γ is the parameter in Eq. 1, so it controls how likely a new dish will be created.

Table-level sampling. Next we sample a dish for a table (line 11 in Algorithm 1). We denote all customers at the tth table in the jth restaurant as $\mathbf{x_{jt}}$. Then we sample its dish k_{jt} according to:

$$p(k_{jt} = k|\mathbf{t}, \mathbf{k}^{-\mathbf{jt}}) \propto \begin{cases} m_{\cdot k}^{-jt} f_k^{-\mathbf{x_{jt}}}(\mathbf{x_{jt}}) \text{ if } k \text{ already exists} \\ \gamma f_{k^{new}}^{-\mathbf{x_{jt}}}(\mathbf{x_{jt}}) \text{ if } k = k^{new} \end{cases} \tag{13}$$

Similarly, $m_{\cdot k}^{-jt}$ is the total number of tables across all restaurants serving the kth dish, without $\mathbf{x_{jt}}$ (dish popularity). $f_k^{-\mathbf{x_{jt}}}(\mathbf{x_{jt}})$ is how much the group of customers $\mathbf{x_{jt}}$ likes the kth dish (dish preference). This time, $f_k^{-\mathbf{x_{jt}}}(\mathbf{x_{jt}})$ is a joint probability of all $x_{ji} \in \mathbf{x_{jt}}$.

Finally, in both Eq. 12 and Eq. 13, we need to sample a new dish. This is done by sampling a new distribution from the base distribution H, $\phi_k \sim H$. After inference, the weights $\boldsymbol{\beta}$ can be computed as $\boldsymbol{\beta} \sim Dirichlet(m_{\cdot 1}, m_{\cdot 2}, \cdots, m_{\cdot k}, \gamma)$. The choice of H is related to the data. In our metaphor, the dishes of the Space-HDP are flows so we use Dirichlet. In the Time-HDP and Speed-HDP, the dishes are modes of time and speed which are Normals. So we use Normal-Inverse-Gamma for H. The choices are because Dirchlet and Norma-Inverse-Gamma are the *conjugate priors* of Multinomial and Normal respectively. The whole CRF sampling is done by iteratively computing Eq. 11 to Eq. 13. The dish number will dynamically increase/decrease until the sampling mixes. In this way, we do not need to know in advance how many space flows or time modes or speed modes there are because they will be automatically learnt.

B CHINESE RESTAURANT FRANCHISE LEAGUE

B.1 Customer Level Sampling

When we do customer-level sampling to sample a new table (line 8 in Algorithm 2), the left side of Eq. 11 becomes:

$$p(t_{ji} = t, x_{ji}, y_{kd}, z_{kc}|\mathbf{x}^{-\mathbf{ji}}, \mathbf{t}^{-\mathbf{ji}}, \mathbf{k}, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l}, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q}) \tag{14}$$

So whether y_{kd} and z_{kc} like the new restaurants should be taken into consideration. After applying Bayesian rules and factorization on Eq. 14, we have:

$$\begin{aligned} p(t_{ji} = t, x_{ji}, y_{kd}, z_{kc}|\bullet) = p(t_{ji}|\mathbf{t}^{-\mathbf{ji}}, \mathbf{k}) \\ p(x_{ji}|y_{kd}, z_{kc}, t_{ji} = t, k_{jt} = k, \bullet) \\ p(y_{kd}|t_{ji} = t, k_{jt} = k, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l}) \\ p(z_{kc}|t_{ji} = t, k_{jt} = k, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q}) \end{aligned} \tag{15}$$

where $\bullet$ is $\{\mathbf{x}^{-\mathbf{ji}}, \mathbf{t}^{-\mathbf{ji}}, \mathbf{k}, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l}, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q}\}$. The four probabilities on the right-hand side of Eq. 15 have intuitive meanings. $p(t_{ji}|\mathbf{t}^{-\mathbf{ji}}, \mathbf{k})$ and $p(x_{ji}|y_{kd}, z_{kc}, t_{ji} = t, k_{jt} = k, \bullet)$ are the table popularity and dish preference of x_{ji} in the space-HDP:

$$p(t_{ji}|\mathbf{t}^{-\mathbf{ji}}, \mathbf{k}) \propto \begin{cases} n_{jt}^{-ji} \text{ if } t \text{ already exists} \\ \alpha_j \text{ if } t = t^{new} \end{cases} \tag{16}$$

$$p(x_{ji}|y_{kd}, z_{kc}, t_{ji} = t, k_{jt} = k, \bullet) \propto \begin{cases} f_{k_{jt}}^{-x_{ji}}(x_{ji}) \text{ if } t \text{ exists} \\ m_{\cdot k} f_k^{-x_{ji}}(x_{ji}) \text{ else if } k \text{ exists} \\ \gamma f_{k^{new}}^{-x_{ji}}(x_{ji}) \text{ if } k = k^{new} \end{cases} \tag{17}$$

Eq. 16 and Eq. 17 are just re-organization of Eq. 11 and Eq. 12. The remaining $p(y_{kd}|t_{ji} = t, k_{jt} = k, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l})$ and $p(z_{kc}|t_{ji} = t, k_{jt} = k, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q})$ can be seen as how much the time-customer y_{kd} and speed-customer z_{kc} like the kth time and speed restaurant respectively (restaurant preference). This restaurant preference does not appear in single HDPs and thus need special treatment. This is the first major difference between CRFL and CRF. Since we propose the same treatment for both, we only explain the time-restaurant preference treatment here.

If every time we sample a t_{ji}, we compute $p(y_{kd}|t_{ji} = t, k_{jt} = k, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l})$ on every time table in every time-restaurant, it will be prohibitively slow. We therefore marginalize over all the time tables in a time-restaurant, to get a general restaurant preference of y_{kd}:

$$\begin{aligned} p(y_{kd}|t_{ji} = t, k_{jt} = k, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l}) = \\ \sum_{o_{kd}=1}^{h_{k\cdot}} p(o_{kd} = o|t_{ji} = t, k_{jt} = k, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}) \\ p(y_{kd}|o_{kd} = o, l_{ko} = l, \mathbf{l}) \end{aligned} \tag{18}$$

where o_{kd} is the table choice of y_{kd} in the kth time-restaurant. l_{ko} is the time-dish served on the oth table in the kth time-restaurant.$h_{k\cdot}$ is the total number of tables in the kth time-restaurant. Similar to Eq. 16 and Eq. 17:

$$p(o_{kd} = o|t_{ji} = t, k_{jt} = k, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}) \propto \begin{cases} s_{ko}^{-kd} \text{ if } o \text{ exists} \\ \epsilon_k \text{ if } o_{kd} = o^{new} \end{cases} \tag{19}$$

where s_{ko}^{-kd} is the number of time-customers already at the oth table and ϵ_k is the scaling factor.

$$p(y_{kd}|o_{kd} = o, l_{ko} = l, \mathbf{l}) \propto \begin{cases} g_{l_{ko}}^{-y_{kd}}(y_{kd}) \text{ if } o \text{ exists} \\ h_{\cdot l} g_l^{-y_{kd}}(y_{kd}) \text{ else if } l \text{ exists} \\ \varepsilon g_{l^{new}}^{-y_{kd}}(y_{kd}) \text{ if } l = l^{new} \end{cases} \tag{20}$$

where $h_{\cdot l}$ is the total number tables serving time-dish l and g is a posterior predictive distribution of Normal, a Student's t-Distribution. ε controls how likely a new time dish would be needed. Now we have finished deriving the sampling for $p(y_{kd}|t_{ji} = t, k_{jt} = k, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \mathbf{l})$. Similar derivations can be done for $p(z_{kc}|t_{ji} = t, k_{jt} = k, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q})$.

After table sampling, we need to do dish sampling (line 10 in Algorithm 2). The left side of Eq. 12 becomes:

$$
\begin{aligned}
&p(k_{jt^{new}} = k, x_{ji}, y_{kd}, z_{kc}|\mathbf{k}^{-\mathbf{jt}^{\mathbf{new}}}, \mathbf{y}^{-\mathbf{kd}}, \mathbf{o}^{-\mathbf{kd}}, \\
&\qquad\qquad \mathbf{l}, \mathbf{z}^{-\mathbf{kc}}, \mathbf{p}^{-\mathbf{kc}}, \mathbf{q}) \propto \\
&\qquad\qquad \begin{cases} m_{\cdot k}^{-jt} p(x_{ji}|\cdots)p(y_{kd}|\cdots)p(z_{kc}|\cdots) \\ \gamma p(x_{ji}|\cdots)p(y_{kd}|\cdots)p(z_{kc}|\cdots) \end{cases}
\end{aligned} \tag{21}
$$

The differences between Eq. 21 and Eq. 12 are $p(y_{kd}|\cdots)$ and $p(z_{kc}|\cdots)$. Both are Infinite Gaussian Mixture Model so the likelihoods can be easily computed. We therefore have given the whole sampling process for the customer-level sampling (Eq. 14). We still need to deal with the table-level sampling.

B.2 Table Level Sampling

Similarly, when we do the table-level sampling (line 14 in Algorithm 2), the left side of Eq. 13 change to:

$$
\begin{aligned}
&p(k_{jt} = k, \mathbf{x_{jt}}, \mathbf{y_{kd_{jt}}}, \mathbf{z_{kc_{jt}}}|\mathbf{k}^{-\mathbf{jt}}, \mathbf{y}^{-\mathbf{kd_{jt}}}, \mathbf{o}^{-\mathbf{kd_{jt}}}, \\
&\qquad\qquad \mathbf{l}^{-\mathbf{ko}}, \mathbf{z}^{-\mathbf{kc_{jt}}}, \mathbf{p}^{-\mathbf{kc_{jt}}}, \mathbf{q}^{-\mathbf{kp}}) \propto \\
&\qquad\qquad \begin{cases} m_{\cdot k}^{-jt} p(\mathbf{x_{jt}}|\cdots)p(\mathbf{y_{kd_{jt}}}|\cdots)p(\mathbf{z_{kc_{jt}}}|\cdots) \\ \gamma p(\mathbf{x_{jt}}|\cdots)p(\mathbf{y_{kd_{jt}}}|\cdots)p(\mathbf{z_{kc_{jt}}}|\cdots) \end{cases}
\end{aligned} \tag{22}
$$

where $\mathbf{x_{jt}}$ is the space-customers at the table t, $\mathbf{y_{kd_{jt}}}$ and $\mathbf{z_{kc_{jt}}}$ are the associated time and speed customers. $\mathbf{k}^{-\mathbf{jt}}$, $\mathbf{y}^{-\mathbf{kd_{jt}}}$, $\mathbf{o}^{-\mathbf{kd_{jt}}}$, $\mathbf{l}^{-\mathbf{ko}}$, $\mathbf{z}^{-\mathbf{kc_{jt}}}$, $\mathbf{p}^{-\mathbf{kc_{jt}}}$, $\mathbf{q}^{-\mathbf{kp}}$ are the rest customers and their choices of tables and dishes in three HDPs. $\cdots$ represents all the conditional variables for simplicity. $p(\mathbf{x_{jt}}|\cdots)$ is the Multinomial f as in Eq. 13.

$p(\mathbf{y_{kd_{jt}}}|\cdots)$ and $p(\mathbf{z_{kc_{jt}}}|\cdots)$ are not easy to compute. However, they can be treated in the same way so we only explain how to compute $p(\mathbf{y_{kd_{jt}}}|\cdots)$ here. To fully compute $p(\mathbf{y_{kd_{jt}}}|\cdots) = p(\mathbf{y_{kd_{jt}}}|k_{jt} = k, \mathbf{o}^{-\mathbf{kd_{jt}}}, \mathbf{l}^{-\mathbf{ko}})$, one needs to consider it for every $y_{kd_{jt}} \in \mathbf{y_{kd_{jt}}}$ which is extremely expensive. This is because we deal with large datasets and there can easily be thousands, if not more, of customers in $\mathbf{y_{kd_{jt}}}$. In Eq. 15, we already see how y_{kd}'s time-restaurant preference influences the table choice of x_{ji}. Given a group $\mathbf{y_{kd_{jt}}}$, their collective time-restaurant preference, $p(\mathbf{y_{kd_{jt}}}|\cdots)$, will influence the dish choice of $\mathbf{x_{jt}}$. Since the distribution of individual time-restaurant preference is hard to compute analytically, we approximate it. We do a random sampling over $\mathbf{y_{kd_{jt}}}$ to approximate $p(\mathbf{y_{kd_{jt}}}|\cdots)$. This number of samples is a hyper-parameter, referred as *customer selection*. For every single $y \in \mathbf{y_{kd_{jt}}}$ we can compute its probability in the same way as in Eq. 18. So we approximate the $p(\mathbf{y_{kd_{jt}}}|\cdots)$ with the joint probability of the sampled time-customers.

B.3 Sampling for Hyper-parameters

A Dirichlet Process contains two parameters, a base distribution and a concentration parameter. To make THDP more robust to these parameters, we impose a prior, a Gamma distribution onto the concentration parameter $\gamma \sim \Gamma(\alpha, \varpi)$, where α is the shape parameter and ϖ is the rate parameter. There are totally six αs and ϖs for the six DPs in THDP. They are initialized as 0.1. Then they are updated during the optimization using the method in [Teh et al. 2006]. The update is done in every iteration in CRFL, after sampling all the other parameters. The customer selection parameter is set to 1000 across all experiments. Finally, after CRFL, the inference is done for the three distributions in Eq. 2:

$$\phi_k^s \sim H_s, \quad \beta \sim Dirichlet(m_{\cdot 1}, m_{\cdot 2}, \cdots, m_{\cdot k}, \gamma) \tag{23}$$

$$\phi_l^t \sim H_t, \quad \zeta \sim Dirichlet(h_{\cdot 1}, h_{\cdot 2}, \cdots, h_{\cdot l}, \varepsilon) \tag{24}$$

$$\phi_q^e \sim H_e, \quad \rho \sim Dirichlet(a_{\cdot 1}, a_{\cdot 2}, \cdots, a_{\cdot q}, \lambda) \tag{25}$$

where $m_{\cdot k}$ is the total number of space-tables choosing space-dish k; $h_{\cdot l}$ is the total number of time-tables choosing time-dish l; $a_{\cdot q}$ is the total number of speed-tables choosing speed-dish q. γ, ε and λ are the scaling factors of G_s, G_t and G_e.

C SIMULATION GUIDANCE

The dynamics of of one trajectory, $\bar{w}$, is:

$$
\begin{aligned}
x_t^{\bar{w}} &= As_t + \omega_t \quad \omega \sim N(0, \Omega) \\
s_t &= Bs_{t-1} + \lambda_t \quad \lambda \sim N(0, \Lambda)
\end{aligned}
$$

Given the U trajectories, from a space flow $\check{w}$, the total likelihood is:

$$
\begin{aligned}
&p(\check{w}) = \Pi_{i=1}^{U} p(\bar{w}_i) \quad \text{where} \\
&p(\bar{w}_i) = \Pi_{t=2}^{T_i-1} p(x_t^i|s_t)P(s_t|s_{t-1}) \quad s_1 = x_1^i, s_T = x_{T_i}^i
\end{aligned} \tag{26}
$$

where A is an identity matrix and Ω is a known diagonal matrix. T_i is the length of the trajectory i. We use homogeneous coordinates to represent both $x = [x_1, x_2, 1]^{\mathrm{T}}$ and $s = [s_1, s_2, 1]^{\mathrm{T}}$. Consequently, A is a $\mathbf{R}^{3\times3}$ identity matrix. Ω is set to be a $\mathbf{R}^{3\times3}$ diagonal matrix with its non-zeros entries set to 0.001. B is a $\mathbf{R}^{3\times3}$ transition matrix and Λ is $\mathbf{R}^{3\times3}$ covariance matrix, both to be learned.

We apply Expectation-Maximization (EM) [Bishop 2007] to estimate parameters B, Λ and states S by maximizing the log likelihood $log\ P(\boldsymbol{u})$. Each iteration of EM consists of a E-step and a M-step. In the E-step, we fix the parameters and sample states s via the posterior distribution of x. The posterior distribution and the expectation of complete-data likelihood are denoted as

$$
\begin{aligned}
\mathcal{L} &= E_{S|X;\hat{B},\hat{\Lambda}}(logP(S, X; B, \Lambda)) \\
&= \sum_i \tau_i E_{s^i|x^i}\{p(s^i, x^i)\}
\end{aligned} \tag{27}
$$

where τ_i is defined as $\tau_i = \frac{\frac{1}{T_i}\sum_{t=1}^{T_i} p(x_t^i|s_t^i)}{\sum_{i=1}^{U}\frac{1}{T^i}\sum_{t=1}^{T_i} p(x_t^i|s_t^i)}$. In the M-step, we maximize the complete-data likelihood and the model parameters are updated as:

$$B^{new} = \frac{\sum_i \tau_i \sum_{t=2}^{T_i} P_{t,t-1}^i}{\sum_i \tau_i \sum_{t=2}^{T_i} P_{t-1,t-1}^i} \tag{28}$$

$$\Lambda^{new} = \frac{\sum_i \tau_i (\sum_{t=2}^{T_i} P_{t,t}^i - B^{new}\sum_{t=2}^{T_i} P_{t,t-1}^i)}{\sum_i \tau_i (T_i - 2)} \tag{29}$$

$$P_{t,t}^i = E_{s^i|x^i}(s_t s_t^{\mathrm{T}}) \tag{30}$$

$$P_{t,t-1}^i = E_{s^i|x^i}(s_t s_{t-1}^{\mathrm{T}}) \tag{31}$$

During updating, we use $\Lambda = \frac{1}{2}(\Lambda + \Lambda^{\mathrm{T}})$ to ensure its symmetry.