APPENDIX

A PROCESSING CROWD DATASETS

Given our definition of a crowd scenario, $\mathcal{S}$, a single video can contain hundreds of scenarios, in the sense that each frame in a video corresponds to a (slightly) different configuration of pedestrians. We therefore associate each frame with a different crowd scenario. To avoid repetition from nearly identical crowd states a few milliseconds apart, we re-sample any trajectories extracted from the video at 8 Hz.

For each video, we use an automated process to extract crowd scenarios, with the exceptions of obstacles, which are labeled by hand. While the tracking data typically provides (noisy) information about the position and velocity of each person at each time step, it cannot capture the *intended* future trajectories of various people, requiring these values to be inferred for each agent as follows:

Initial Positions and Velocities. After smoothing all trajectories with a Butterworth filter, the smoothed position at each (resampled) time step provides the position for each agent. The corresponding velocities are inferred using finite differences.

Radius. We estimated a typical agent radius for each dataset by finding the typical minimum distance between trajectories (generally between 0.1 m and 0.15 m).

Goal Position. We estimate the instantaneous goal position for each agent as the final observed position along its smoothed trajectory.

Goal Speed. We analyzed the crowd data from Section 3.1 looking at both the minimum and maximum speeds of the pedestrians over the course of their trajectories. The resulting probability density function is showing in Figure 12. There is a strong peak in maximum speeds at 1.3 m/s which we assume represents a typical desired speed, that is, the speed most people would walk if they had no obstacles or other people around. Furthermore, there are two peaks in minimum speeds, one at 0.7 m/s which generally comes from collision avoidance efforts, and one at 0.1 m/s indicating the preference of people to stop.

Based on this analysis, in our crowd scenarios, agents are assumed to have a desired goal speed of 1.3 m/s, with two exceptions. One, if their current speed averaged over 1 s window is too small (less than 0.1 m/s) agents are assumed to be standing still. Two, if an agent is moving faster than 1.3 m/s we assume this faster speed is its desired goal speed. The underlying assumption to this model is that observed speeds lower than 1.3 m/s are typically caused by collision avoidance which should be modeled by the crowd simulation, whereas speeds higher than this value represent an agent's intention to walk faster than average.

By combining the goal and goal speed, we can infer the *goal velocity* of the agent indicating its desired direction of motion and speed. Note, that the inferred goal velocity will vary from agent to agent and can change from time step to time step.

B ENTROPY METRIC STATISTICAL ASSUMPTIONS

Our new entropy metric formulation assumes that the prediction error for each agent, and for each time step can be well modeled by a Gaussian. It is worth emphasizing that we do not assume the agent's position or velocities follow a Gaussian, only the error between the predicted and the true state does. This assumption proves to hold fairly well for the simulation methods and datasets used in this study. Figure 13a shows a typical example of velocity error from a Vision-based agent in the middle of the Berlin-90 dataset. While this is a very complex scenario, the error between the model and the true estimate is well approximated by a Gaussian distribution.

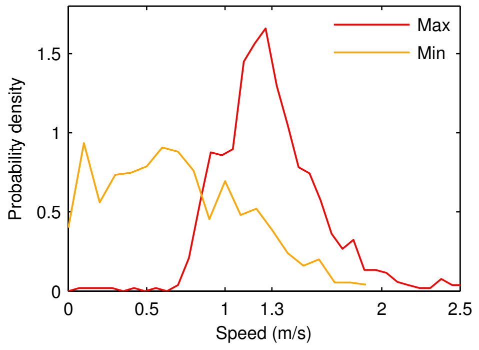

Fig. 12. Distribution of minimum and maximum pedestrian speeds extracted from the ten video datasets described in Section 3.1.

Table 3. The magnitude of the 4D error between predicted and observed states. The reported numbers are the averages over all KS tests, and each error is generated from 1,000 ensemble numbers.

Simulation Method	Error
Social Forces	.0730 ± .0493
Vision-based	.0635 ± .0450
ORCA	.0785 ± .0552

As discussed in Section 4.2, to quantify the goodness-of-fit of a Gaussian, we used the Kolmogorov-Smirnov test . This test measures if we can reject the hypothesis that the error is well modeled by a Gaussian distribution. Our analysis was performed on the error distributions of 10 randomly selected agents extracted from Zara1, Students, Berlin-90, and One-way, yielding 2,000 distributions (one per-agent, per-time step) each consisting of 1,000 ensemble members. Each agent's time step sample has its own separate KS test where the empirical Gaussian distribution was used as reference. Figure 14 summarizes all these independent tests for several simulation methods across all four dimensions of the error. Across all cases, very few distributions are found to be non-Gaussian at the α=.5 significance level, suggesting the vast majority of errors can be well represented by a Gaussian. In addition, over the 2,000 analyzed cases, the Gaussians are near zero-meaned, suggesting an unbiased Q. Figure 13b, for example, shows the the evolution over time of the magnitude of the mean error for the Berlin-90 agent, while the average magnitude of the mean error for all 2,000 cases is reported in Table 3.

C NON-DETERMINISM IN HUMAN MOTION

When people walk towards a head-on collision, they may avoid each other either by passing each other on the right or on the left side.

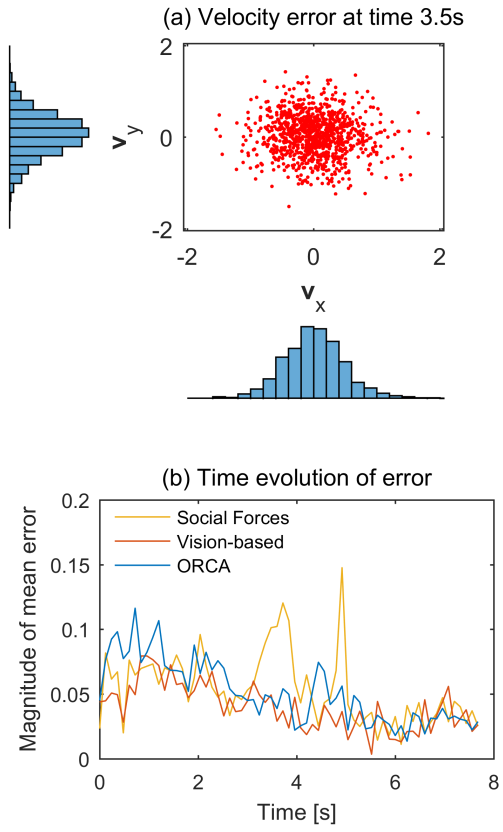

Fig. 13. (a) Velocity prediction error for a Vision-based agent for one time step in the Berlin-90 dataset. Despite the complexity of the scenario, the error distribution is approximately Gaussian. (b) Time evolution of the magnitude of the 4D error for the same scenario using different simulation methods.

Here, we set up an experiment to test the robustness of our entropy metric to such stochastic/non-deterministic behavior as compared to a distance-based error metric and and a neighbor-based error metric (see Section 4.3 for details).

In our setup, we used as reference data trajectories of two pedestrians exactly swapping positions. These initial trajectories were generated by ORCA resulting in two synthetic pedestrians passing each other on the right side. By exactly *mirroring* the velocities of the pedestrians, we generated a different, but equally plausible, set of trajectories for the same start-goal pair (now the pedestrians pass each other on the left).

Figure 15 shows the simulation error results for the original and flipped cases obtained with the Vision-based model (blue and red plots, respectively). As the Vision-based simulation leads to slightly different trajectories than the original reference ones, the distance-based metric is able to capture such difference. However, when the flipped trajectories are used as reference, the distance-based error significantly increases during the interaction period of the agents due to the change in the passing side; given the same start-goal pair the Vision-model results in the same pair of trajectories as before, but now the reference trajectories have changed. In contrast, in both original and flipped cases, the time-varying entropy scores are very similar and show nearly perfect correlation. This is due to the fact that our metric does not evaluate the Vision agents at the trajectory level, but rather treats the reference data as a single sample from the true distribution of plausible agent responses. In this scenario, the two error plots generated by the neighbor-based metric are very similar as well (the small difference is due to the fact that the reference data is not perfectly symmetrical, and hence the time-varying distance between the two reference pedestrians is slightly different in the original and flipped data). However, the correlation weakens in scenarios with multiple agents. We refer the reader to Table 1 in the main text for the corresponding results along with additional experiments.

D APPLICATION: PER-AGENT TUNING

The per-agent, per-time step nature of our modified entropy metric allows it to be used in a variety of crowd simulation analysis tasks in a much more flexible manner than the [Guy et al. 2012] formulation. For example, per-agent metrics allow for outlier detection [Charalambous et al. 2014] or per-agent performance tuning [Berseth et al. 2014; Wolinski et al. 2014]. In Figure 16, we show results based on 2-person interaction data from [Pettré et al. 2009]. Here, we graph the effect of tuning each agent's maximum speed and acceleration to better match that of the corresponding person. The entropy improvement varies between agents and across different points of the interaction.

E DIMENSIONALITY REDUCTION ANALYSIS

Section 5.2 discusses the performance of various dimensionality reduction techniques we considered for creating the Crowd Space manifold. One form of analysis of the quality of the low-D manifold is to compare the correspondence between distances of MPD descriptors in the original 60D space and the corresponding distances of the same samples in the embedded low dimensional space [Venna et al. 2010]; for high-quality projections the distances between the two spaces should be highly correlated. To estimate this, we computed pairwise distances for a randomly sampled subset of 1,000 MPD descriptors extracted from the six stratified MPD datasets shown in Figure 1, resulting in 10,000 distances. Table 2a reports this correlation as measured by Spearman's rank correlation coefficient for a varying number of output dimensions per dimensionality reduction method. Examples of corresponding scatterplots, known as Shepard diagrams [Shepard 1980] are shown in Figure 17 for the techniques depicted in Figure 18. Additionally, the generalization errors of the embedded spaces are presented in Table 2b. Here, we used all 15,000 MPD samples, and trained a 1-nearest neighbor classifier on the low-dimensional data to determine the video dataset of origin for each sample. As can be seen from both tables, PCA exhibits the best performance across all the techniques regardless of the dimensionality of the embedded dataset.

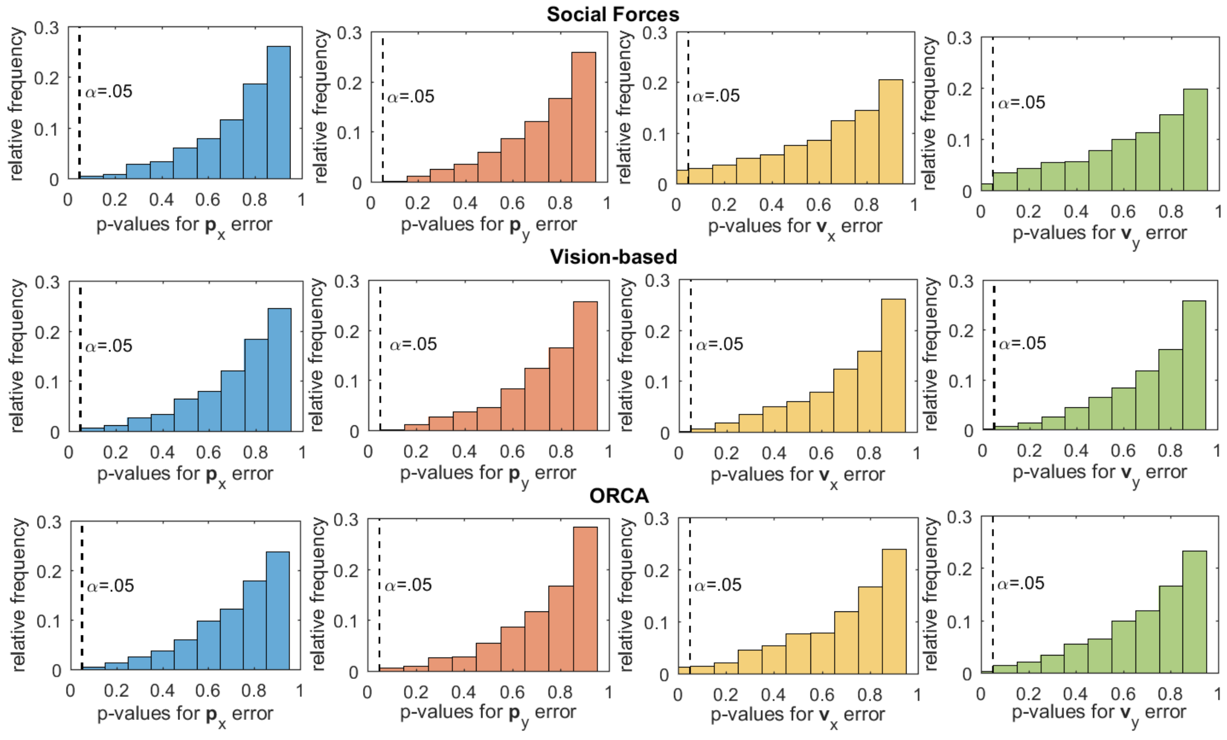

Fig. 14. Results from Kolmogorov-Smirnov test for Gaussian fit for three different simulators. In all cases, the results show very little support for rejecting the Gaussian error model hypothesis.

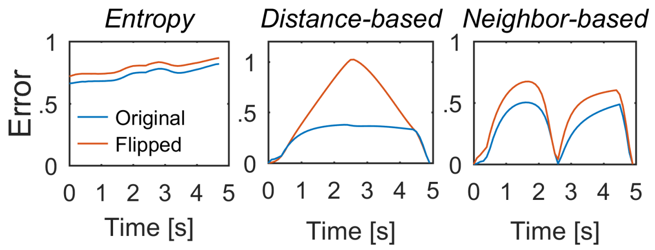

Fig. 15. A comparison of the time-varying error in a Vision-based simulation as measured by our entropy metric, an absolute distanced-based error metric, and a nearest neighbor-based error metric. After mirroring the reference synthetic paths, our entropy metric does not significantly change even though the passing side of the two pedestrians has switched. Importantly, the entropy values are highly correlated with the values derived from the unmodified data.

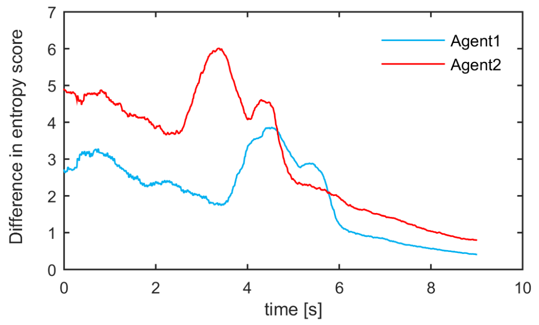

Fig. 16. Our modified entropy metric can capture the per-agent effect of parameter tuning. This graph shows the effect of tuning the velocities of two T-model agents based on interaction data in [Pettré et al. 2009] (as measured by the absolute difference in entropy score).

F HYPERPARAMETER TUNING

For the purpose of tuning hyperparameters (e.g., the percentage of nearby neighbors needed to trigger a vote), we consider two metrics:

(1) W_1: Denotes the percentage of times that we select the highest accuracy simulation methods as winning methods.
(2) L_2: Denotes the percentage of times that we select the one of the worst 2 simulation methods as winning methods.

Where we seek to maximize W_1 while minimizing L_2. In both metrics, we rank the simulation methods based on the percentage of agents that each method is suitable to simulate. If no method is the best for at least half of the agents in a scenario, the scenario is removed from the training dataset.

In order to avoid over-fitting, the W_1 and L_2 metrics were computed using a leave-4-out cross validation scheme, where we randomly select six of the datasets as training data and the remaining

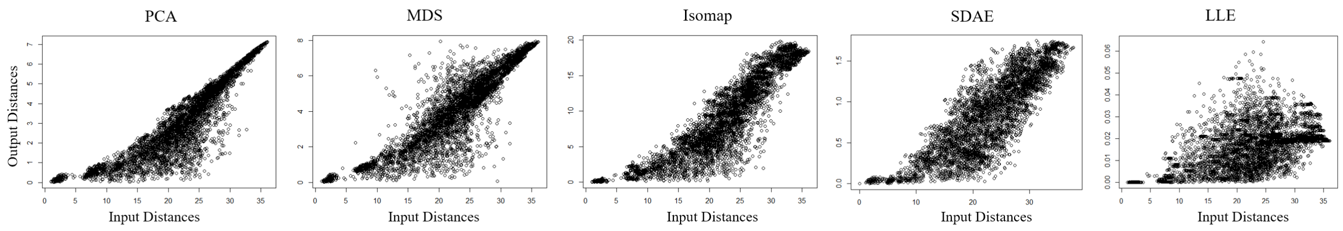

Fig. 17. Shepard plots comparing the 60 dimensional distances between points to the distances between the same points in various 2D projections (c.f. Figure 18). Ideally, these distances would be preserved exactly leading to a one-one-one correspondence (i.e., all points falling on a single straight line). Here, we can see PCA does the best as persevering the true distances with low dimensions for our datasets.

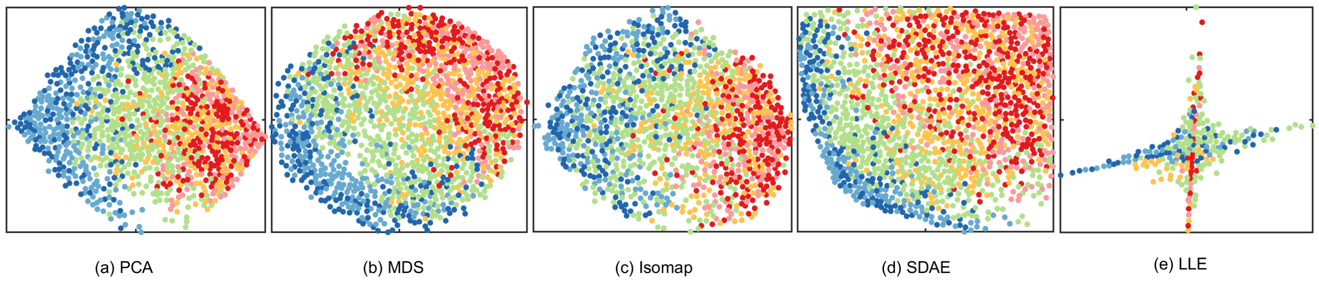

Fig. 18. 2D projection of six MPD datasets obtained using common dimensionality reduction methods. The projections were obtained after subsampling an equal number of agents from each dataset.

four datasets are retained as testing data. To tune our hyperparameters, we used 4 inner cross validation loops. In each loop, we split the training datasets to 4 training and 2 validation datasets, and perform a grid search on the hyperparameters based on the classification performance on the validation set using the following function:

$$\frac{2\,W_1(1-L_2)}{W_1+(1-L_2)} \tag{3}$$

which seeks to balance finding the best winner with reducing the risk of predicting one the worst methods.