Trace and Pace: Controllable Pedestrian Animation via Guided Trajectory Diffusion

Davis Rempe[*,1,2] Zhengyi Luo[*,1,3] Xue Bin Peng[1,4] Ye Yuan[1] Kris Kitani[3]
Karsten Kreis[1] Sanja Fidler[1,5,6] Or Litany[1]

[1]NVIDIA [2]Stanford University [3]Carnegie Mellon University [4]Simon Fraser University
[5]University of Toronto [6]Vector Institute

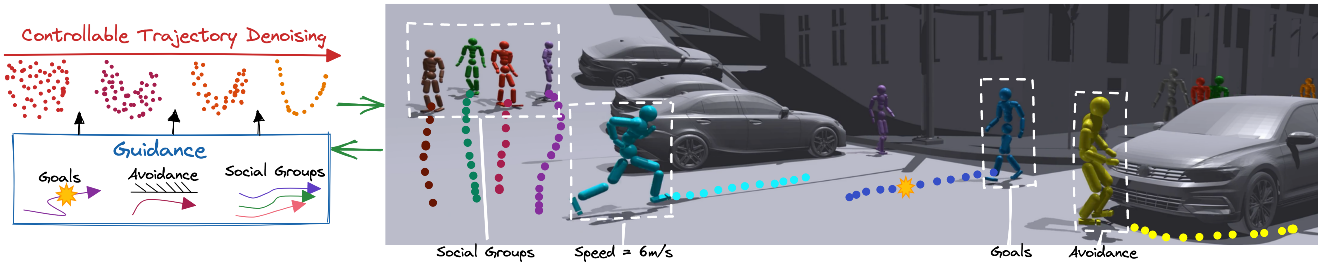

Figure 1. (Left) We propose TRACE, a trajectory diffusion model that enables user control through test-time guidance. (Right) Generated trajectories are passed to a novel physics-based humanoid controller (PACER), forming a closed-loop pedestrian animation system.

Abstract

We introduce a method for generating realistic pedestrian trajectories and full-body animations that can be controlled to meet user-defined goals. We draw on recent advances in guided diffusion modeling to achieve test-time controllability of trajectories, which is normally only associated with rule-based systems. Our guided diffusion model allows users to constrain trajectories through target waypoints, speed, and specified social groups while accounting for the surrounding environment context. This trajectory diffusion model is integrated with a novel physics-based humanoid controller to form a closed-loop, full-body pedestrian animation system capable of placing large crowds in a simulated environment with varying terrains. We further propose utilizing the value function learned during RL training of the animation controller to guide diffusion to produce trajectories better suited for particular scenarios such as collision avoidance and traversing uneven terrain. Video results are available on the `project page`.

1. Introduction

Synthesizing high-level human behavior, in the form of 2D positional trajectories, is at the core of modeling pedestrians for applications like autonomous vehicles, urban planning, and architectural design. An important feature of such synthesis is *controllability* – generating trajectories that meet user-defined objectives, edits, or constraints. For example, a user may place specific waypoints for characters to follow, specify social groups for pedestrians to travel in, or define a social distance to maintain.

Attaining controllability is straightforward for algorithmic or rule-based models of human behavior, since they have built-in objectives. In the simplest case, human trajectories can be determined by the shortest paths between control points [11], but more sophisticated heuristics have also been developed for pedestrians [2, 14], crowds [22, 46], and traffic [29, 53]. Unfortunately, algorithmically generated trajectories often appear unnatural. Learning-based approaches, on the other hand, can improve naturalness by mimicking real-world data. These methods often focus on short-term trajectory prediction using a single forward pass of a neural network [1, 10, 49, 61]. However, the ability to *control* these models is limited to sampling from an output trajectory distribution [34, 58] or using an expensive latent space traversal [45]. As a result, learning-based methods can predict implausible motions such as collisions with obstacles or between pedestrians. This motivates another notion of *controllability* – maintaining realistic trajectories during agent-agent and agent-environment interactions.

In this work, we are particularly interested in using controllable pedestrian trajectory models for character animation. We envision a simple interface where a user provides high-level objectives, such as waypoints and social groups, and a system converts them to *physics-based* full-body human motion. Compared to existing kinematic motion mod-

[*] Equal contribution

els [19, 27, 42], physics-based methods have the potential to produce high-quality motion with realistic subtle behaviors during transitions, obstacle avoidance, traversing uneven terrains, *etc*. Although there exist physics-based animation models [12, 27, 39–41, 57], controlling their behavior requires using task-specific planners that need to be retrained for new tasks, terrains, and character body shapes.

We develop a generative model of trajectories that is data driven, controllable, and tightly integrated with a physics-based animation system for full-body pedestrian simulation (Fig. 1). Our method enables generating pedestrian trajectories that are realistic and amenable to user-defined objectives at test time. We use this trajectory generator as a planner for a physics-based pedestrian controller, resulting in a closed-loop controllable pedestrian animation system.

For trajectory generation, we introduce a **TRA**jectory Diffusion Model for **C**ontrollable **PE**destrians (TRACE). Inspired by recent successes in generating trajectories through denoising [9, 20, 64], TRACE generates the future trajectory for each pedestrian in a scene and accounts for the surrounding context through a spatial grid of learned map features that is queried locally during denoising. We leverage classifier-free sampling [17] to allow training on mixed annotations (*e.g*., with and without a semantic map), which improves controllability at test time by trading off sample diversity with compliance to conditioning. User-controlled sampling from TRACE is achieved through test-time *guidance* [7, 17, 18], which perturbs the output at each step of denoising towards the desired objective. We extend prior work [20] by introducing several analytical loss functions for pedestrians and re-formulating trajectory guidance to operate on clean trajectory outputs from the model [18], improving sample quality and adherence to user objectives.

For character animation, we develop a general-purpose **P**edestrian **A**nimation **C**ontroll**ER** (PACER) capable of driving physics-simulated humanoids with diverse body types to follow trajectories from a high-level planner. We focus on (1) motion quality: PACER learns from a small motion database to create natural and realistic locomotion through adversarial motion learning [40, 41]; (2) terrain and social awareness: trained in diverse terrains with other humanoids, PACER learns to move through stairs, slopes, uneven surfaces, and to avoid obstacles and other pedestrians; (3) diverse body shapes: by training on different body types, PACER draws on years of simulation experience to control a wide range of characters; (4) compatibility with high-level planners: PACER accepts 2D waypoints and can be a plug-in model for any 2D trajectory planner.

We demonstrate a controllable pedestrian animation system using TRACE as a high-level planner for PACER, the low-level animator. The planner and controller operate in a closed loop through frequent re-planning according to simulation results. We deepen their connection by guiding TRACE with the value function learned during RL training of PACER to improve animation quality in varying tasks. We evaluate TRACE on synthetic [2] and real-world pedestrian data [3, 26, 38], demonstrating its flexibility to user-specified and plausibility objectives while synthesizing realistic motion. Furthermore, we show that our animation system is capable and robust with a variety of tasks, terrains, and characters. In summary, we contribute (1) a diffusion model for pedestrian trajectories that is readily controlled at test time through guidance, (2) a general-purpose pedestrian animation controller for diverse body types and terrains, and (3) a pedestrian animation system that integrates the two to drive simulated characters in a controllable way.

2. Related Work

Pedestrian Trajectory Prediction. Modeling high-level pedestrian behavior has been extensively studied in the context of motion prediction (forecasting). Approaches range from physics and planning-based [13, 14, 55] to recent learned methods [1,5,24,49,61]. We refer the reader to the thorough survey by Rudenko *et al*. [47] for an overview, and focus this discussion on controllability. Most forecasting work is motivated by planning for autonomous vehicles (AVs) or social robots [10] rather than *controllability* or longer-term synthesis. Rule-based models for pedestrians [2, 22, 46] and vehicle traffic [29, 53] can easily incorporate user constraints [25] making them amenable to control. However, the trajectories of these approaches are not always human-like; methods have even been developed to choose the best simulation method and tune parameters to make crowd scenarios more realistic [21].

Data-driven methods produce human-like motions, but neural network-based approaches are difficult to explicitly *control*. Some works decompose forecasting into goal prediction followed by trajectory prediction based on goals [6, 34]. These models offer limited control by selecting goal locations near a target or that minimizes an objective (*e.g*. collisions) [58]. Synthesized pedestrian behavior can also be controlled by strategically choosing a starting location [43]. STRIVE [45] showed that a VAE trajectory model can be controlled through test-time optimization in the learned latent space. Reinforcement learning (RL) agents can be controlled in crowd simulations by incorporating tasks into reward functions for training [23]. By varying the weights of different rewards, the characters can be controlled to exhibit one of several behaviors at test time [37]. Our method, TRACE, trains to mimic trajectories from data and is agnostic to any task: all controls are defined at test time, allowing flexibility to new controls after training. Instead of lengthy test-time optimization, we use guidance for control.

Controllable Character Animation. Full-body pedestrian animation typically involves a high-level task (*e.g*. trajectory following, obstacle avoidance) and low-level body control. Some methods solve both with a single network that implicitly uses high-level planning and low-level animation. GAMMA [63] trains a kinematic model to go to waypoints, while PFNN [19] follows gamepad inputs. Physics-based humanoid controllers such as AMP [41] train different models for each task, limiting their general applicability.

Two-stage methods split the task into separate high-level planning and low-level character control, where task information is only used by the planner. Planning can be done with traditional A* [11], using learned trajectory prediction [4], searching in a pre-trained latent space [27, 40, 44, 57], or using hierarchical RL [12, 39, 40, 42, 57]. DeepLoco [39], Haworth *et al.* [12], and ASE [40] utilize hierarchical RL to achieve impressive dynamic control for various tasks. They require lengthy training for both low-level and high-level controllers and often jointly train as a final step. They must also train different planners for different tasks.

Our approach follows the two-stage paradigm, with the distinction that both our high-level (TRACE) and low-level (PACER) models consume task information for pedestrian navigation: through test-time guidance and map-conditioned path following, respectively. TRACE and PACER are unaware of each other at training time, yet can be tightly integrated in a closed loop: trace-pace-retrace.

Diffusion Models and Guidance. Diffusion models have shown success in generating images [16, 36, 54], videos [15], and point clouds [62]. Guidance has been used for test-time control in several ways: classifier [7] and classifier-free [17] guidance reinforce input conditioning, while reconstruction guidance [18] has been used for coherent video generation. Gu *et al.* [9] adapt the diffusion framework for short-term pedestrian trajectory forecasting conditioned on past trajectories. Diffuser [20] generates trajectories for planning and control in robotics applications with test-time guidance. Closest to ours is the concurrent work of CTG [64], which builds on Diffuser to develop a controllable vehicle traffic model, focusing on following formalized traffic rules like speed limits. Our method contains several key differences: we encode map conditioning into an expressive feature grid queried in denoising, we use classifier-free sampling to enable multi-dataset training and test-time flexibility, we re-formulate guidance to operate on clean model outputs, and we link with a low-level animation model using value function guidance.

3. Method

To model high-level pedestrian behavior, we first introduce the controllable trajectory diffusion model (TRACE). In Sec. 3.2, we detail our low-level physics-based pedestrian controller, PACER, and in Sec. 3.3 how they can be combined into an end-to-end animation system.

3.1. Controllable Trajectory Diffusion

Problem Setting. Our goal is to learn high-level pedestrian behavior in a way that can be *controlled* at test time. For pedestrian animation, we focus on two types of control: (1) user specification, *e.g.*, goal waypoints, social distance, and social groups, and (2) physical plausibility, *e.g.*, avoiding collisions with obstacles or between pedestrians.

We formulate synthesizing pedestrian behavior as an agent-centric trajectory forecasting problem. At each time step, the model outputs a future trajectory plan for a target *ego* agent conditioned on that agent's past, the past trajectories of all neighboring agents, and the semantic map context. Formally, at timestep t we want the future state trajectory $\boldsymbol{\tau}_s = [\mathbf{s}_{t+1} \quad \mathbf{s}_{t+2} \quad \dots \quad \mathbf{s}_{t+T_f}]$ over the next T_f steps where the state $\mathbf{s} = [x \quad y \quad \theta \quad v]^T$ includes the 2D position (x, y), heading angle θ, and speed v. We assume this state trajectory is actually the result of a sequence of actions [64] defined as $\boldsymbol{\tau}_a = [\mathbf{a}_{t+1} \quad \mathbf{a}_{t+2} \quad \dots \quad \mathbf{a}_{t+T_f}]$ where each action $\mathbf{a} = [\dot{v} \quad \dot{\theta}]^T$ contains the acceleration $\dot{v}$ and yaw rate $\dot{\theta}$. The state trajectory can be recovered from the initial state and action trajectory as $\boldsymbol{\tau}_s = f(\mathbf{s}_t, \boldsymbol{\tau}_a)$ using a given dynamics model f. The full state-action trajectory is then denoted as $\boldsymbol{\tau} = [\boldsymbol{\tau}_s; \boldsymbol{\tau}_a]$. To predict the future trajectory, the model receives as input the past state trajectory of the ego pedestrian $\mathbf{x}^{\text{ego}} = [\mathbf{s}_{t-T_p} \quad \dots \quad \mathbf{s}_t]$ along with the past trajectories of N neighboring pedestrians $X^{\text{neigh}} = \{\mathbf{x}^{\text{i}}\}_{i=1}^{N}$. It also gets a crop of the rasterized semantic map $\boldsymbol{\mathcal{M}} \in \mathbb{R}^{H \times W \times C}$ in the local frame of the ego pedestrian at time t. These inputs are summarized as the *conditioning* context $C = \{\mathbf{x}^{\text{ego}}, X^{\text{neigh}}, \boldsymbol{\mathcal{M}}\}$.

Our key idea is to train a diffusion model to conditionally generate trajectories, which can be *guided* at test time to enable controllability. For simplicity, the following formulation uses the full trajectory notation $\boldsymbol{\tau}$, but in practice, the state trajectory is always a result of actions, *i.e.*, diffusion/denoising are on $\boldsymbol{\tau}_a$ which determines the states through f. Next, we summarize our diffusion framework, leaving the details to the supplementary material.

3.1.1 Trajectory Diffusion Model

We build on Diffuser [20] and generate trajectories through iterative denoising, which is learned as the reverse of a predefined *diffusion* process [16, 51]. Starting from a clean future trajectory $\boldsymbol{\tau}^0 \sim q(\boldsymbol{\tau}^0)$ sampled from the data distribution, the forward noising process produces a sequence of progressively noisier trajectories $(\boldsymbol{\tau}^1, \dots, \boldsymbol{\tau}^k, \dots, \boldsymbol{\tau}^K)$ by adding Gaussian noise at each process step k:

$$\begin{aligned} q(\boldsymbol{\tau}^{1:K} \mid \boldsymbol{\tau}^0) &:= \prod_{k=1}^{K} q(\boldsymbol{\tau}^k \mid \boldsymbol{\tau}^{k-1}) \\ q(\boldsymbol{\tau}^k \mid \boldsymbol{\tau}^{k-1}) &:= \mathcal{N}(\boldsymbol{\tau}^k; \sqrt{1-\beta_k}\boldsymbol{\tau}^{k-1}, \beta_k \mathbf{I}) \end{aligned} \tag{1}$$

where β_k is the variance at each step of a fixed schedule, and with a large enough K we get $q(\boldsymbol{\tau}^K) \approx \mathcal{N}(\boldsymbol{\tau}^K; \mathbf{0}, \mathbf{I})$.

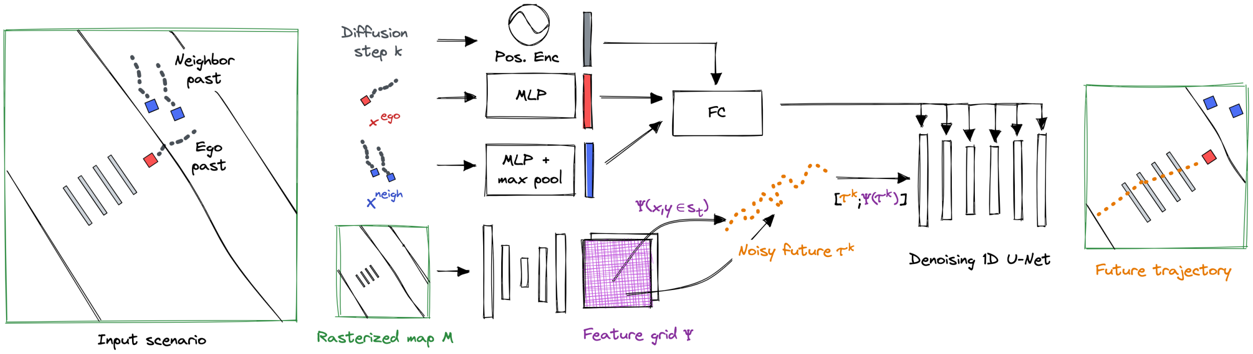

Figure 2. Trajectory diffusion model (TRACE). Future trajectory denoising is conditioned on past and neighbor motion by adding processed features to intermediate U-Net features. Map conditioning is provided through a feature grid queried along the noisy input trajectory.

TRACE learns the reverse of this process so that the sampled noise can be *denoised* into plausible trajectories. Each step of this reverse process is conditioned on C:

$$p_\phi(\boldsymbol{\tau}^{k-1} \mid \boldsymbol{\tau}^k, C) := \mathcal{N}(\boldsymbol{\tau}^{k-1}; \boldsymbol{\mu}_\phi(\boldsymbol{\tau}^k, k, C), \boldsymbol{\Sigma}_k) \quad (2)$$

where ϕ are model parameters and $\boldsymbol{\Sigma}_k$ is from a fixed schedule. TRACE learns to parameterize the mean of the Gaussian distribution at each step of the denoising process.

Training and Classifier-Free Sampling. Importantly for guidance, the network does *not* directly output $\boldsymbol{\mu}$. Instead, at every step it learns to predict the final clean trajectory $\boldsymbol{\tau}^0$, which is then used to compute $\boldsymbol{\mu}$ [36]. Training supervises this network output $\hat{\boldsymbol{\tau}}^0$ with ground truth future trajectories (*i.e.* denoising score matching [16, 52, 56]):

$$L = \mathbb{E}_{\boldsymbol{\epsilon}, k, \boldsymbol{\tau}^0, C} \left[||\boldsymbol{\tau}^0 - \hat{\boldsymbol{\tau}}^0||^2 \right] \quad (3)$$

where $\boldsymbol{\tau}^0$ and C are sampled from the training dataset, $k \sim \mathcal{U}\{1, 2, \ldots, K\}$ is the step index, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is used to corrupt $\boldsymbol{\tau}^0$ to give the noisy input trajectory $\boldsymbol{\tau}^k$.

Our training procedure allows the use of *classifier-free sampling*[1] at test time, which has been shown to improve compliance to conditioning in diffusion models [17]. We simultaneously train both a conditional model $\boldsymbol{\mu}_\phi(\boldsymbol{\tau}^k, k, C)$ and unconditional model $\boldsymbol{\mu}_\phi(\boldsymbol{\tau}^k, k)$ by randomly dropping out conditioning during training. At test time, predictions from both models are combined with weight w as:

$$\tilde{\boldsymbol{\epsilon}}_\phi = \boldsymbol{\epsilon}_\phi(\boldsymbol{\tau}^k, k, C) + w \left(\boldsymbol{\epsilon}_\phi(\boldsymbol{\tau}^k, k, C) - \boldsymbol{\epsilon}_\phi(\boldsymbol{\tau}^k, k) \right) \quad (4)$$

where $\boldsymbol{\epsilon}_\phi$ is the model's prediction of how much noise was added to the clean trajectory to produce the input $\boldsymbol{\tau}^k$; it is straightforward to compute from $\boldsymbol{\mu}_\phi$ [36].

Note that $w{>}0$ and $w{<}0$ increase and decrease the effect of conditioning, respectively, while $w{=}0$ and $w{=}{-}1$ result in the purely conditional or unconditional model, respectively. This flexibility allows a user to trade off respecting conditioning with trajectory diversity, which benefits controllability (see Sec. 4.2). This approach also enables training on multiple distinct datasets with varying annotations: conditioning is already being dropped out randomly, so it is easy to use mixed data with subsets of the full conditioning. Since there are pedestrian datasets with diverse motions but no semantic maps [26, 38], and others with limited motions but detailed maps [3], we find mixed training is beneficial to boost diversity and controllability (see Sec. 4.2).

Architecture. As shown in Fig. 2, TRACE uses a U-Net similar to [20] that has proven effective for trajectories. The input trajectory $\boldsymbol{\tau}_k$ at step k is processed by a sequence of 1D temporal convolutional blocks that progressively down and upsample the sequence in time, leveraging skip connections. A key challenge is how to condition the U-Net on C to predict trajectories that comply with the map and other pedestrians. To incorporate step k, ego past $\mathbf{x}^{\text{ego}}$, and neighbor past X^{neigh}, we use a common approach [18, 20] that extracts a single conditioning feature and adds it to the intermediate trajectory features within each convolutional block. For the map $\boldsymbol{\mathcal{M}}$, we encode with a 2D convolutional network into a feature grid, where each pixel contains a high-dimensional feature. At step k of denoising, each 2D position $(x, y) \in \boldsymbol{\tau}^k$ is queried by interpolating into the grid to give a feature trajectory, which is concatenated to $\boldsymbol{\tau}^k$ and becomes the U-Net input. Intuitively, this allows learning a localized representation that can benefit subtle map interactions such as obstacle avoidance.

3.1.2 Controllability through Clean Guidance

After training TRACE to generate realistic trajectories, controllability is implemented through test-time *guidance*. Intuitively, guidance nudges the sampled trajectory at each step of denoising towards a desired outcome. Let $\mathcal{J}(\boldsymbol{\tau})$ be a guidance loss function measuring how much a trajectory $\boldsymbol{\tau}$ violates a user objective. This may be learned [20] or an analytical differentiable function [64]. Guidance uses the gradient of $\mathcal{J}$ to perturb the predicted mean from the model at each denoising step such that the right side of Eq. (2)

[1] we refer to it as "sampling" instead of the common term "guidance" [17] to avoid confusion with the guidance introduced in Sec. 3.1.2

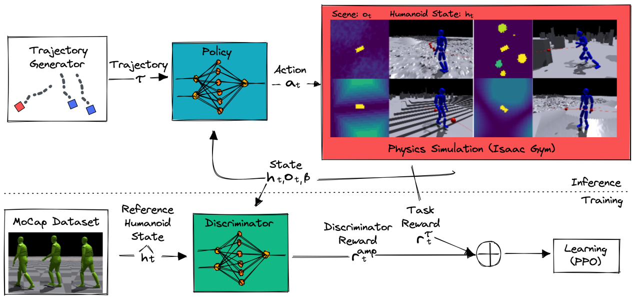

Figure 3. Pipeline: Pedestrian Animation Controller (PACER).

becomes $\mathcal{N}(\boldsymbol{\tau}^{k-1}; \tilde{\boldsymbol{\mu}}_\phi(\boldsymbol{\tau}^k, k, C), \boldsymbol{\Sigma}_k)$ where $\tilde{\boldsymbol{\mu}}$ is the perturbed (guided) mean. Prior work [20, 64] directly perturbs the *noisy* network-predicted mean with

$$\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu} - \alpha \boldsymbol{\Sigma}_k \nabla_{\boldsymbol{\mu}} \mathcal{J}(\boldsymbol{\mu}) \tag{5}$$

where α determines the guidance strength. Note that Eq. (5) evaluates $\mathcal{J}$ at the noisy mean, so learned loss functions must be trained at varying noise levels and analytic loss functions may suffer from numerical issues.

To avoid this, we build upon "reconstruction guidance", which operates on the *clean* model prediction $\hat{\boldsymbol{\tau}}^0$ [18]. We extend the guidance formulation introduced in [18] for temporal video upsampling to work with arbitrary loss functions. At each denoising step with input $\boldsymbol{\tau}^k$, we first perturb the clean trajectory predicted from the network $\hat{\boldsymbol{\tau}}^0$ with

$$\tilde{\boldsymbol{\tau}}^0 = \hat{\boldsymbol{\tau}}^0 - \alpha \boldsymbol{\Sigma}_k \nabla_{\boldsymbol{\tau}^k} \mathcal{J}(\hat{\boldsymbol{\tau}}^0), \tag{6}$$

then compute $\tilde{\boldsymbol{\mu}}$ in the same way as we would in Eq. (2), *i.e.*, as if $\tilde{\boldsymbol{\tau}}^0$ were the output of the network. Note that the gradient is evaluated wrt the noisy input trajectory $\boldsymbol{\tau}^k$ rather than the clean $\hat{\boldsymbol{\tau}}^0$, requiring backpropagation through the denoising model. We formulate several analytical guidance objectives like waypoint reaching, obstacle avoidance, collision avoidance, and social groups (see Sec. 4.1, 4.2). A learned RL value function can also be used (Sec. 4.3).

3.2. Physics-Based Pedestrian Animation

To enable full-body pedestrian simulation, we design the Pedestrian Animation ControllER (PACER) to execute the 2D trajectories generated by TRACE in a physics simulator.

Background: Goal-Conditioned RL. Our framework (Fig. 3) follows the general goal-conditioned reinforcement learning framework, where a goal-conditioned policy π_{PACER} is trained to follow 2D target trajectories specified by $\boldsymbol{\tau}_s$. The task is formulated as a Markov Decision Process (MDP) defined by a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma \rangle$ of states, actions, transition dynamics, reward function, and discount factor. The state $\mathcal{S}$, transition dynamics $\mathcal{T}$, and reward R are calculated by the environment based on the current simulation and goal, while the action $\mathcal{A}$ is computed by the policy π_{PACER}. The policy's objective is to maximize the discounted return $\mathbb{E}\left[\sum_{t=1}^{T} \gamma^{t-1} r_t\right]$ where r_t is the reward per timestep. We utilize Proximal Policy Optimization (PPO) [50] to find the optimal control policy π_{PACER}.

Terrain, Social, and Body Awareness. To create a controller that can simulate crowds in realistic 3D scenes (*e.g.* scans, neural reconstructions, or artist-created meshes (Fig. 1)), our humanoid must be terrain aware, socially aware of other agents, and support diverse body types. We use a humanoid model that conforms to the kinematic structure of SMPL [28], and is automatically generated using a procedure similar to [30, 31, 60]. Our control policy $\pi_{\text{PACER}}(\boldsymbol{a}_t | \boldsymbol{h}_t, \boldsymbol{o}_t, \boldsymbol{\beta}, \boldsymbol{\tau}_s)$ is conditioned on the state of the simulated character $\boldsymbol{h}_t$, environmental features $\boldsymbol{o}_t$, body type $\boldsymbol{\beta}$, and goal trajectory $\boldsymbol{\tau}_s$. The environment input is a rasterized local height and velocity map of size $\boldsymbol{o}_t \in \mathbb{R}^{64\times64\times3}$, which gives agents crucial information about their surroundings. To allow for social awareness, nearby humanoids are represented as a cuboid and rendered on the global height map. In this way, each humanoid views other people as dynamic obstacles to avoid. Obstacle and interpersonal avoidance are learned by using obstacle collision as a termination condition. By conditioning and training with different body parameters $\boldsymbol{\beta}$ our policy learns to adapt to characters with diverse morphologies.

Realistic Motion through Adversarial Learning. To learn the optimal control policy π_{PACER} that (1) follows a 2D trajectory closely and (2) creates realistic pedestrian motions, we follow Adversarial Motion Prior (AMP) [41]. AMP uses a motion discriminator to encourage the policy to generate motions that are similar to the movement patterns contained in a dataset of motion clips recorded by human actors. The discriminator $D(\boldsymbol{h}_{t-10:t}, \boldsymbol{a}_t)$ is then used to specify a motion style reward r_t^{amp} for training the policy. The style reward is combined with a trajectory following reward $r_t^{\mathcal{T}}$ and an energy penalty r_t^{energy} [8] to produce the total reward $r_t = r_t^{\text{amp}} + r_t^{\mathcal{T}} + r_t^{\text{energy}}$. To mitigate artifacts arising from asymmetric gaits, such as limping, we utilize the motion-symmetry loss proposed by [59]:

$$\begin{aligned} L_{\text{sym}}(\theta) = & \| \pi_{\text{PACER}}(\boldsymbol{h}_t, \boldsymbol{o}_t, \boldsymbol{\beta}, \boldsymbol{\tau}_s) - \\ & \Phi_a(\pi_{\text{PACER}}(\Phi_s(\boldsymbol{h}_t, \boldsymbol{o}_t, \boldsymbol{\beta}, \boldsymbol{\tau}_s))) \|^2, \end{aligned} \tag{7}$$

where Φ_s and Φ_a mirror the state and action along the character's sagittal plane. This loss encourages the policy to produce more symmetric motions, leading to natural gaits. During training, random terrains are generated following the procedure used in [48]. We create stairs, slopes, uneven terrains, and obstacles consisting of random polygons. Character morphology is also randomized by sampling a gender and body type from the AMASS dataset [32]. The policy and discriminator are then conditioned on the SMPL gender and body shape $\boldsymbol{\beta}$ parameters. More details are available in the supplementary material.

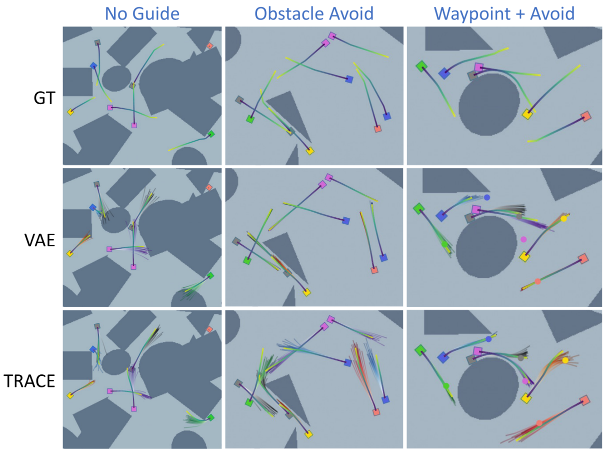

Figure 4. Guidance results on *ORCA-Maps*. For VAE and TRACE, 20 samples are visualized for each pedestrian (the boxes) along with the final trajectory chosen via filtering which is bolded.

3.3. Controllable Pedestrian Animation System

The high-level trajectory planning from TRACE is combined with the low-level character control from PACER to create an end-to-end pedestrian animation system. The two components are trained independently, but at runtime they operate in a closed feedback loop: PACER follows planned trajectories for $2s$ before TRACE re-planning, taking past character motion from PACER as input. By combining terrain and social awareness of PACER with collision avoidance guidance, both high and low-level systems are task-aware and work in tandem to prevent collisions and falls.

Value Function as Guidance. To enable tighter two-way coupling between TRACE and PACER, in Sec. 4.3 we explore using the value function learned during RL training of PACER to guide trajectory diffusion. The value function predicts expected future rewards and is aware of body pose and surrounding terrain and agents. Using the value function to guide denoising encourages TRACE to produce trajectories that are easier to follow and better suited to the current terrain (which TRACE is unaware of otherwise). Unlike Diffuser [20], which requires training a reward function with samples from the diffusion model at varying noise levels, our guidance (Eq. (6)) operates on clean trajectories so we can use the value function directly from RL training.

4. Experiments

We first demonstrate the controllability of TRACE when trained on synthetic (Sec. 4.1) and real-world (Sec. 4.2) pedestrian data. Sec. 4.3 evaluates our full animation system on several tasks and terrains. **Video results** are provided in the supplementary material.

Implementation Details. TRACE is trained to predict $5s$ of future motion from $3s$ of past motion (both at 10Hz), and uses $K{=}100$ diffusion steps. During training, map and neighbor conditioning inputs are independently dropped with 10% probability. At test time, we sample (and guide) multiple future trajectories for each pedestrian in a scene and choose one with the lowest guidance loss, which we refer to as *filtering*. PACER operates at 30Hz; we randomly sample terrain, body type, and procedural 2D trajectories during training and use a dataset of locomotion sequences from AMASS [32]. All physics simulations are performed using NVIDIA's Isaac Gym simulator [33].

Datasets. The *ORCA* dataset (Sec. 4.1) contains synthetic trajectory data from $10s$ scenes generated using the ORCA crowd simulator [2]. Up to 20 agents are placed in a $15m\times15m$ environment with $\leq$20 static primitive obstacles. Agent placement and goal velocity, along with obstacle placement and scale, are randomized per scene. The dataset contains two distinct subsets: *ORCA-Maps* has many obstacles but few agents, while *ORCA-Interact* has no obstacles (*i.e.* no map annotations) but many agents.

For real-world data (Sec. 4.2), we use ETH/UCY and nuScenes. ETH/UCY [26, 38] is a common trajectory forecasting benchmark that contains scenes with dense crowds and interesting pedestrian dynamics but does not have semantic maps. nuScenes [3] contains $20s$ driving scenes in common street settings. We convert the pedestrian bounding-box annotations to 2D trajectories and use them for training and evaluation. Detailed semantic maps are also annotated with layers for roads, crosswalks, and sidewalks.

Metrics. We care about trajectory plausibility and meeting user controls. Controllability is evaluated with a *Guidance Error* that depends on the task: *e.g.*, for avoidance objectives this is collision rate, while the waypoint error measures the minimum distance from the trajectory. *Obstacle and Agent Collision Rates* measure the frequency of collisions. *Realism* is measured at the dataset or trajectory level by (1) computing the Earth Mover's Distance (EMD) between the generated and ground truth test-set histograms of trajectory statistics (*e.g.* velocity, longitudinal/lateral acceleration) [58], or (2) measuring the mean accelerations of each trajectory, assuming pedestrians generally move smoothly.

4.1. Augmenting Crowd Simulation

We first evaluate TRACE trained on *ORCA-Maps* and *ORCA-Interact*. These provide a clean test bed for comparisons since there is a clear definition of correct pedestrian behavior – no obstacle or agent collisions are present in the data. All methods operate in an open loop by predicting a single $5s$ future for each pedestrian. This way, compounding errors inherent to closed-loop operation are not a factor.

Results for single and multi-objective guidance on the *ORCA-Maps* test set are shown in Tab. 1. TRACE is compared to a *VAE* baseline [45] adapted to our setup, which achieves controllability through test-time latent optimization. This is a strong baseline that generally works well, but requires expensive optimization at test time. We also

Guide	Method	Guidance Error	Collision Rate Obstacle	Collision Rate Agent	Realism (EMD) Vel	Realism (EMD) Lon Acc	Realism (EMD) Lat Acc
None	VAE [45]	–	0.076	**0.118**	0.038	0.039	0.040
	TRACE	–	**0.050**	0.132	**0.029**	**0.008**	**0.009**
Obstacle Avoid	VAE [45]	0.018	0.018	**0.116**	0.040	0.036	0.039
	TRACE-Filter	0.018	0.018	0.123	**0.019**	**0.011**	**0.015**
	TRACE-Noisy	0.015	0.015	0.125	0.021	0.012	0.017
	TRACE	**0.014**	**0.014**	0.124	0.020	**0.011**	0.017
Agent Avoid	VAE [45]	0.010	0.075	0.010	0.041	0.038	0.039
	TRACE-Filter	0.049	**0.050**	0.049	0.031	0.012	0.013
	TRACE-Noisy	**0.000**	0.056	**0.000**	0.035	0.013	**0.012**
	TRACE	**0.000**	0.058	**0.000**	**0.025**	**0.010**	**0.012**
Waypoint	VAE [45]	**0.078**	0.051	**0.092**	0.070	0.031	0.033
	TRACE-Filter	0.333	**0.046**	0.112	**0.044**	**0.013**	**0.013**
	TRACE-Noisy	0.129	0.052	0.110	0.067	0.038	0.033
	TRACE	0.105	0.048	0.093	0.057	**0.013**	0.014
Waypoint & Obs Avoid & Agt Avoid	VAE [45]	**0.207**	**0.021**	0.015	0.053	0.032	0.032
	TRACE-Filter	0.527	0.023	0.096	**0.025**	0.014	0.016
	TRACE-Noisy	0.236	0.022	0.017	0.057	0.025	0.022
	TRACE	0.211	**0.021**	**0.009**	0.036	**0.007**	**0.009**

Table 1. Guidance evaluation on *ORCA-Maps* dataset. TRACE using full diffusion guidance improves upon VAE latent optimization and selective sampling (*TRACE-Filter*) in terms of meeting objectives, while maintaining strong realism.

Guide	Method	Train Data	w	Guidance Error	Realism (Mean) Lon Acc	Realism (Mean) Lat Acc
Waypoint	VAE [45]	Mixed	–	**0.340**	0.193	0.172
	TRACE	nuScenes	-0.5	0.421	0.177	0.168
		Mixed	0.0	0.551	0.159	0.145
		Mixed	-0.5	0.366	**0.140**	**0.132**
Waypoint perturbed	VAE [45]	Mixed	–	0.962	0.443	0.441
	TRACE	nuScenes	-0.5	0.977	0.239	0.238
		Mixed	0.0	1.129	0.233	0.218
		Mixed	-0.5	**0.802**	**0.212**	**0.204**
Social groups	VAE [45]	Mixed	–	0.297	0.109	0.104
	TRACE	nuScenes	-0.5	0.287	0.155	0.158
		Mixed	0.0	**0.244**	0.110	0.101
		Mixed	-0.5	0.245	**0.094**	**0.087**

Table 2. Guidance evaluation on nuScenes. Training on mixed data and using $w{<}0$ for classifier-free sampling are important to achieve controllability for out-of-distribution objectives.

compare to two ablations: *TRACE-Filter* samples from the diffusion model *without guidance* and chooses the best sample according to the guidance loss (similar to [58]), while *TRACE-Noisy* uses the guidance formulated in Eq. (5) from prior works [20, 64]. Models are trained on the combined dataset of *ORCA-Maps* (with map annotations) and *ORCA-Interact* (no map annotations). The guidance losses are: **None** samples randomly with no guidance; **Obstacle avoid** discourages collisions between map obstacles and pedestrian bounding boxes; **Agent avoid** discourages collisions between pedestrians by denoising all their futures in a scene jointly; **Waypoint** encourages a trajectory to pass through a goal location at any point in the planning horizon. For this experiment, the waypoint is set as the position of each pedestrian at $4s$ into the future in the ground truth data. These are *in-distribution* objectives, since they reinforce behavior already observed in the ground truth data.

In Tab. 1, TRACE successfully achieves all objectives through the proposed guidance. It is competitive or better than the VAE optimization in terms of guidance, while maintaining velocity and acceleration distributions closer to ground truth as indicated by *Realism*. Fig. 4 shows that random samples from the VAE contain collisions, and using latent optimization for controllability gives similar local minima across samples thereby limiting diversity compared to TRACE. Finally, using our proposed clean guidance (Eq. (6)) instead of the noisy version produces consistently better results in guidance and realism.

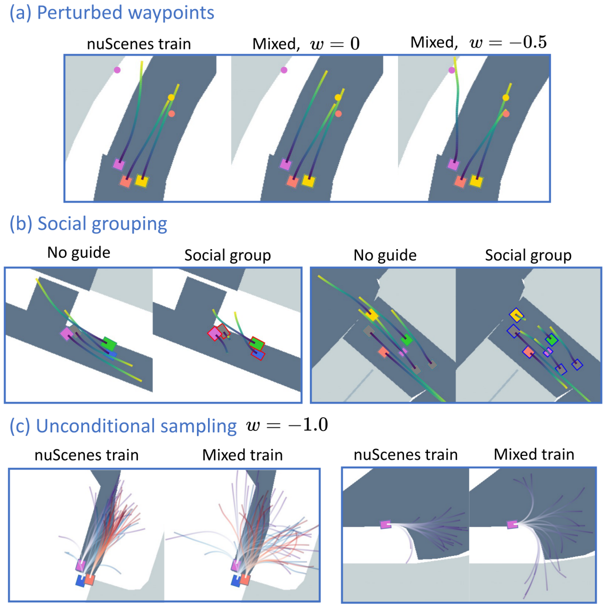

Figure 5. nuScenes results demonstrating flexibility of TRACE. (a) Using mixed training and $w{=}{-}0.5$ is best for noisy waypoints. (b) Social group guidance encourages sets of pedestrians to stay close. (c) Mixed training (ETH/UCY+nuScenes) learns a more diverse distribution as demonstrated by unconditional sampling.

4.2. Real-world Data Evaluation

We next evaluate controllability when trained on real-world data, and focus on *out-of-distribution* (OOD) guidance objectives to emphasize the flexibility of our approach. In this experiment, methods operate in a *closed loop*: pedestrians are rolled out for $10s$ and re-plan at 1Hz. Results on a held out nuScenes split are shown in Tab. 2. We compare TRACE trained on mixed data (ETH/UCY+nuScenes), after training on nuScenes only, and using two different classifier-free sampling weights w. Along with in-distribution **Waypoint** (now at $9s$ into the future), two additional objectives are evaluated: **Waypoint perturbed** uses a noisily perturbed ground truth future position (at $9s$), requiring pedestrians to go off sidewalks or into streets to reach the goal; **Social groups** specifies groups of agents to stay close and travel together. Groups are set heuristically based on spatial proximity and velocity at initialization.

In Tab. 2, we observe that OOD flexibility requires (1) training on mixed data, and (2) classifier-free sampling. Since nuScenes data is less diverse (people tend to follow the sidewalk), TRACE trained on just nuScenes struggles

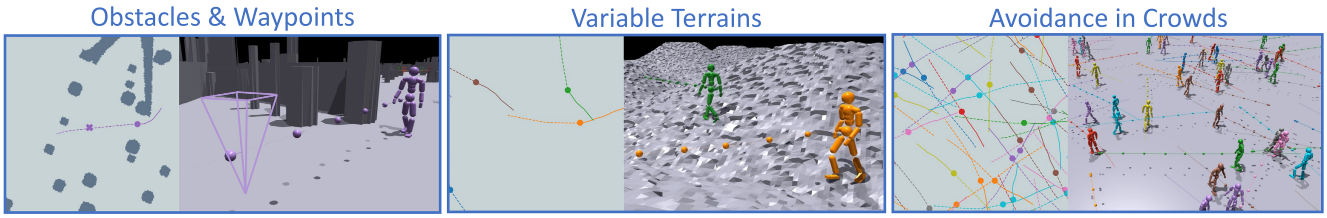

Figure 6. Our animation system enables avoiding obstacles, meeting goals, traversing variable terrains, and large crowds.

Terrain	**Guide**	**Fail** Rate	**Traj Follow** Error	**Discrim** Reward
Random	Procedural	0.133	0.680	1.950
	None	**0.093**	**0.104**	1.887
	Waypoint	0.107	0.111	**2.113**
Obstacles	Procedural	0.307	0.948	2.278
	None	0.125	0.093	2.512
	Obs Avoid	**0.063**	**0.089**	**2.521**
Flat (Crowd)	Procedural	0.127	0.371	2.320
	None	0.087	0.082	2.374
	Agt Avoid	**0.013**	**0.071**	**2.402**

Table 3. Closed-loop animation results. Our system successfully follows waypoints and avoids collisions in a variety of terrains, and additional guidance improves performance.

to hit perturbed waypoints. Though the VAE is trained on mixed data, it struggles to produce diverse dynamics on the nuScenes maps to achieve OOD objectives, even though it uses 200 optimization steps (2× more than the diffusion steps K=100 in TRACE). TRACE reaches OOD objectives using classifier-free sampling with w=-0.5 to downweight the conditioning of the semantic map and leverage diverse trajectories learned from ETH/UCY. The flexibility of TRACE is further highlighted in Fig. 5.

4.3. Controllable Pedestrian Animation

Finally, we demonstrate our full controllable pedestrian animation system. TRACE is trained on *ORCA* and used as a planner for the pre-trained PACER without any finetuning. We evaluate the animations by: *Fail Rate* measures the fraction of agents that fall down or collide with an obstacle or other agent, *Trajectory Following Error* measures the average deviation of the character from TRACE's plan, and *Discriminator Reward* is the mean reward returned by the adversarial motion prior used to train PACER, which measures how human-like a generated motion appears.

Tab. 3 evaluates the animations from our system using TRACE with and without guidance in various settings: *Random* is an assortment of smooth and rough slopes and stairs with varying difficulties, *Obstacles* is a flat terrain with large obstacles, and *Flat* is a flat terrain with pedestrians spawned in a crowd of 30. For each setting, 600 rollouts of 10s are simulated across 30 characters with random bodies from AMASS [32]. To put the difficulty of environments and discriminator rewards in context, we also include metrics when using the (terrain and obstacle unaware) *Procedural* trajectory generation method used to train PACER.

	Guide		**Waypoint**	**Fail**	**Traj Follow**	**Discrim**
Terrain	Waypoint	Value	Error	Rate	Error	Reward
Random	√		0.541	0.107	**0.111**	2.113
	√	√	**0.481**	**0.100**	0.112	**2.162**
Obstacles	√		1.065	0.220	0.138	2.552
	√	√	**0.929**	**0.178**	**0.113**	**2.609**
Flat (Crowd)	√		0.248	0.063	**0.084**	2.555
	√	√	**0.175**	**0.053**	**0.084**	**2.607**

Table 4. Using the value function learned in RL training as guidance improves quality of trajectory following and robustness to varying terrains, obstacles, and other agents.

Our combined system performs well in the physically-simulated environment with TRACE providing easy-to-follow trajectories resulting in high-quality animations from PACER, as evaluated by the discriminator. Diffusion guidance can further improve failure rates, especially for avoiding agent collisions in dense crowds. Fig. 6 shows some qualitative applications of our animation system and we highly encourage viewing the supplementary **video results** to qualitatively evaluate the motion quality. Tab. 4 shows the effect of using the learned value function from training PACER as a guidance loss for TRACE. In each setting, adding value guidance in addition to waypoint guidance makes trajectories easier to follow, reduces failures, and improves the discriminator reward. As a result, waypoint guidance error also improves.

5. Discussion

We have introduced a controllable trajectory diffusion model, a robust physics-based humanoid controller, and an end-to-end animation system that combines the two. This represents an exciting step in being able to control the high-level behavior of learned pedestrian models, and opens several directions for future work. First is improving the efficiency of sampling from trajectory diffusion models to make them real-time: TRACE currently takes 1-3s to sample for a single character, depending on the guidance used (see the supplement for full analysis). Recent work in diffusion model distillation [35] offers a potential solution. In addition to high-level motion controllability, exploring how diffusion models can be extended to low-level full-body character control is an interesting next step.

Acknowledgments. Davis Rempe was supported by an NVIDIA Graduate Fellowship. The authors thank Ziyuan Zhong and the AV Research Group for helpful discussions on trajectory diffusion.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016. 1, 2

[2] Jur van den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics Research: The 14th International Symposium (ISRR)*, pages 3–19. Springer, 2011. 1, 2, 6

[3] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 2, 4, 6

[4] Zhe Cao, Hang Gao, Karttikeya Mangalam, Qi-Zhi Cai, Minh Vo, and Jitendra Malik. Long-term human motion prediction with scene context. In *European Conference on Computer Vision*, pages 387–404. Springer, 2020. 3

[5] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *Conference on Robot Learning (CoRL)*, pages 86–99. PMLR, 2020. 2

[6] Patrick Dendorfer, Aljosa Osep, and Laura Leal-Taixé. Goalgan: Multimodal trajectory prediction based on goal position estimation. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 2

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 2, 3

[8] Zipeng Fu, Xuxin Cheng, and Deepak Pathak. Deep whole-body control: Learning a unified policy for manipulation and locomotion. *ArXiv*, abs/2210.10044, 2022. 5

[9] Tianpei Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. Stochastic trajectory prediction via motion indeterminacy diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17113–17122, 2022. 2, 3

[10] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2255–2264, 2018. 1, 2

[11] Mohamed Hassan, Duygu Ceylan, Ruben Villegas, Jun Saito, Jimei Yang, Yi Zhou, and Michael J Black. Stochastic scene-aware motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11374–11384, 2021. 1, 3

[12] M. Brandon Haworth, Glen Berseth, Seonghyeon Moon, Petros Faloutsos, and Mubbasir Kapadia. Deep integration of physical humanoid control and crowd navigation. *Proceedings of the 13th ACM SIGGRAPH Conference on Motion, Interaction and Games*, 2020. 2, 3

[13] Dirk Helbing, Illés Farkas, and Tamas Vicsek. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, 2000. 2

[14] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995. 1, 2

[15] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 3

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 3, 4

[17] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 2, 3, 4

[18] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022. 2, 3, 4, 5

[19] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017. 2, 3

[20] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *International Conference on Machine Learning (ICML)*, 2022. 2, 3, 4, 5, 6, 7

[21] Ioannis Karamouzas, Nick Sohre, Ran Hu, and Stephen J Guy. Crowd space: a predictive crowd analysis technique. *ACM Transactions on Graphics (TOG)*, 37(6):1–14, 2018. 2

[22] Jongmin Kim, Yeongho Seol, Taesoo Kwon, and Jehee Lee. Interactive manipulation of large-scale crowd animation. *ACM Transactions on Graphics (TOG)*, 33(4):1–10, 2014. 1, 2

[23] Jaedong Lee, Jungdam Won, and Jehee Lee. Crowd simulation by deep reinforcement learning. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, pages 1–7, 2018. 2

[24] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 336–345, 2017. 2

[25] Marilena Lemonari, Rafael Blanco, Panayiotis Charalambous, Nuria Pelechano, Marios Avraamides, Julien Pettré, and Yiorgos Chrysanthou. Authoring virtual crowds: A survey. In *Computer Graphics Forum*, volume 41, pages 677–701. Wiley Online Library, 2022. 2

[26] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007. 2, 4, 6

[27] Hung Yu Ling, Fabio Zinno, George Cheng, and Michiel Van De Panne. Character controllers using motion vaes. *ACM Transactions on Graphics (TOG)*, 39(4):40–1, 2020. 2, 3

[28] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: a skinned multi-person linear model. *ACM Trans. Graph.*, 34:248:1–248:16, 2015. 5

[29] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 2575–2582. IEEE, 2018. 1, 2

[30] Zhengyi Luo, Ryo Hachiuma, Ye Yuan, and Kris Kitani. Dynamics-regulated kinematic policy for egocentric pose estimation. In *Advances in Neural Information Processing Systems*, 2021. 5

[31] Zhengyi Luo, Shun Iwase, Ye Yuan, and Kris Kitani. Embodied scene-aware human pose estimation. In *Advances in Neural Information Processing Systems*, 2022. 5

[32] Naureen Mahmood, N. Ghorbani, N. Troje, Gerard Pons-Moll, and Michael J. Black. Amass: Archive of motion capture as surface shapes. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5441–5450, 2019. 5, 6, 8

[33] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021. 6

[34] Karttikeya Mangalam, Yang An, Harshayu Girase, and Jitendra Malik. From goals, waypoints & paths to long term human trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15233–15242, 2021. 1, 2

[35] Chenlin Meng, Ruiqi Gao, Diederik P Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. *arXiv preprint arXiv:2210.03142*, 2022. 8

[36] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 3, 4

[37] Andreas Panayiotou, Theodoros Kyriakou, Marilena Lemonari, Yiorgos Chrysanthou, and Panayiotis Charalambous. Ccp: Configurable crowd profiles. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 2

[38] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009. 2, 4, 6

[39] Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel van de Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics (Proc. SIGGRAPH 2017)*, 36(4), 2017. 2, 3

[40] Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Trans. Graph.*, 41(4), July 2022. 2, 3

[41] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph.*, 40(4), July 2021. 2, 3, 5

[42] Maria Priisalu, Ciprian Paduraru, Aleksis Pirinen, and Cristian Sminchisescu. Semantic synthesis of pedestrian locomotion. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 2, 3

[43] Maria Priisalu, Aleksis Pirinen, Ciprian Paduraru, and Cristian Sminchisescu. Generating scenarios with diverse pedestrian behaviors for autonomous vehicle testing. In *Conference on Robot Learning*, pages 1247–1258. PMLR, 2022. 2

[44] Davis Rempe, Tolga Birdal, Aaron Hertzmann, Jimei Yang, Srinath Sridhar, and Leonidas J. Guibas. Humor: 3d human motion model for robust pose estimation. In *International Conference on Computer Vision (ICCV)*, 2021. 3

[45] Davis Rempe, Jonah Philion, Leonidas J. Guibas, Sanja Fidler, and Or Litany. Generating useful accident-prone driving scenarios via a learned traffic prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 6, 7

[46] Zhiguo Ren, Panayiotis Charalambous, Julien Bruneau, Qunsheng Peng, and Julien Pettré. Group modeling: A unified velocity-based approach. In *Computer Graphics Forum*, volume 36, pages 45–56. Wiley Online Library, 2017. 1, 2

[47] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras. Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, 39(8):895–935, 2020. 2

[48] Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning, 2021. 5

[49] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European Conference on Computer Vision*, pages 683–700. Springer, 2020. 1, 2

[50] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017. 5

[51] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 3

[52] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. 4

[53] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000. 1, 2

[54] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Advances in Neural Information Processing Systems*, 34:11287–11302, 2021. 3

[55] Jur Van den Berg, Ming Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE international conference on robotics and automation*, pages 1928–1935. Ieee, 2008. 2

[56] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011. 4

[57] Jungdam Won, Deepak Gopinath, and Jessica Hodgins. Physics-based character controllers using conditional vaes. *ACM Transactions on Graphics (TOG)*, 41(4):1–12, 2022. 2, 3

[58] Danfei Xu, Yuxiao Chen, Boris Ivanovic, and Marco Pavone. Bits: Bi-level imitation for traffic simulation. *arXiv preprint arXiv:2208.12403*, 2022. 1, 2, 6, 7

[59] Wenhao Yu, Greg Turk, and C. Karen Liu. Learning symmetric and low-energy locomotion. *ACM Transactions on Graphics (TOG)*, 37:1 – 12, 2018. 5

[60] Ye Yuan, Shih-En Wei, Tomas Simon, Kris Kitani, and Jason Saragih. Simpoe: Simulated character control for 3d human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 5

[61] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823, 2021. 1, 2

[62] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[63] Yan Zhang and Siyu Tang. The wanderings of odysseus in 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20481–20491, 2022. 3

[64] Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone. Guided conditional diffusion for controllable traffic simulation. *International Conference on Robotics and Automation (ICRA)*, 2023. 2, 3, 4, 5, 7