Résumé of *SCTP: Scene Compliant Trajectory Prediction Using Diffusion and Point Clouds*

For this thesis, we set out to improve upon the drawbacks of using Denoising Diffusion Probabilistic Models (DDPMs) to do scene-compliant, pedestrian trajectory prediction. The primary drawback of DDPMs is the notoriously slow inference time, as the model is repeatedly evaluated for every timestep; a critical aspect in some areas of trajectory prediction. Previous work have applied DDPMs in this domain, both using trajectory-on-scene and traditional trajectory generation approaches, but with inference speed remaining a critical drawback in either. Our goal was then to strike a balance between achieving a speedup in a DDPM-based and maintaining or improving upon the performance of the model.

We present four contributions on the subject, making substantial strides towards our goal; (1) we propose a DDPM-based model, SCTP, to do stochastic, scene-compliant trajectories, (2) we propose using a point cloud representation of the environment, incorporating only relevant parts of the scene thus improving inference time by decreasing the model size and consequently the computational burden, (3) to further improve inference time we utilise a leapfrogging stage during inference, skipping 50% of the diffusion timesteps and effectively increasing inference performance by 100%, and (4) we evaluate the model on the PFSD benchmark dataset, demonstrating the models ability to outperform state-of-the-art (SOTA) models on the minimum Average Displacement Error ($minADE$) and Environment Collision-Free Likelihood ($ECFL$) metrics, and remain competitive on the minimum Final Displacement Error ($minFDE$) metric. Our qualitative results show five different cases, each in a different environment, further detailing the performance of SCTP. Our model predicted 50 possible future trajectories for each case, and the general consensus across all is its ability to generate natural trajectories that align with the environment. They showcase the model's ability to generate multiple possible directions when presented with an environment containing a fork while showing a clear understanding of navigating in an enclosed space containing only a single exit, short of turning around and backtracking its steps.

In summary, we have successfully achieved a much faster inference time compared to previous DDPM-based work focused on scene-compliance, while *improving* upon the performance metrics $minADE$, $minFDE$, and $ECFL$ to achieve SOTA performance.

SCTP: Scene Compliant Trajectory Prediction Using Diffusion and Point Clouds

Mads Bach Andersen   Simon Andreas Hansen
Aalborg University, Denmark
{madsan18, saha18}@student.aau.dk

***Abstract*—The unpredictable nature of human behaviour is a critical aspect in the domain of trajectory prediction, especially when viewed in the context of autonomous vehicles. Generational models such as Conditional Variational Autoencoders (CVAEs) or Generative Adversarial Networks (GANs) have been shown to produce state-of-the-art results, however, there is still room for improvements in areas such as the ability to generate natural, collision-free predictions. Denoising Diffusion Probabilistic Models (DDPMs) are a recent type of generative model that has seen substantial adaptation across fields such as image generation, audio synthesis, and time series forecasting, but has yet to infiltrate trajectory prediction thoroughly. In this paper, we make further advancements in this domain, demonstrating DDPMs' ability to generate natural, scene-compliant trajectories, capable of competing with state-of-the-art methods through our proposed model, SCTP. We address the slow inference time of DDPMs - a critical aspect of trajectory prediction when viewed in the context of autonomous vehicles - by adopting the leapfrogging technique and a point cloud representation of the map to decrease the inference speed by a factor of more than two when compared to traditional approaches. We also demonstrate that closed-loop predictions can perform as well as state-of-the-art approaches, which commonly generate the trajectories in a one-shot approach, allowing the model to evaluate the map repeatedly, contributing to its superior performance when generating collision-free trajectories.**

***Index Terms*—Trajectory Prediction, Denoising Diffusion Probabilistic Model, Environmental alignment, Point cloud**

1. Introduction

Trajectory prediction is a subset of timeseries prediction in which the goal is to predict the next sequence of timesteps for a given trajectory. This task has broad applications in fields such as autonomous driving [1, 2], Robotics [3, 4, 5], Video Surveillance [6, 7], Sports Analytics [8, 9], Air traffic control [10]. The requirements of a trajectory prediction model vary across the fields, however, autonomous driving sets a particularly high bar for the predictions, both in terms of precision and speed. In this field, trajectory predictions are often used in critical scenarios, such as navigating through an environment containing pedestrians or other moving agents [1, 2, 11, 12, 13]. The predictions thus have a safety-critical aspect and need to be computed in real-time to allow other components of the autonomous vehicle to leverage them. A prediction may either be of the vehicle itself, or nearby pedestrians, the latter of which is the subject of this research. To effectively leverage predicted trajectories in the system of an autonomous vehicle, the predictions need to be plausible and adhere to environmental constraints, and thus predictions which cross obstacles or collide with the environment are unnatural and cannot be considered viable. We illustrate the effects of disregarding scene constraints in Fig. 1a, where an autonomous vehicle is forced to come to a stop, as the prediction puts it on a collision course with the pedestrian. In Fig. 1b the constraints are respected, allowing the vehicle to continue as normal. Furthermore, the intent of nearby pedestrians, i.e. their destination, is an unknown variable, which introduces a stochastic element into trajectory predictions of this kind. This is illustrated in Fig. 2a, where a single trajectory is predicted, disregarding the possibility that a pedestrian might use the crosswalk, whereas Fig. 2b addresses this issue with multiple predictions, allowing the autonomous vehicle to consider more than a single path.

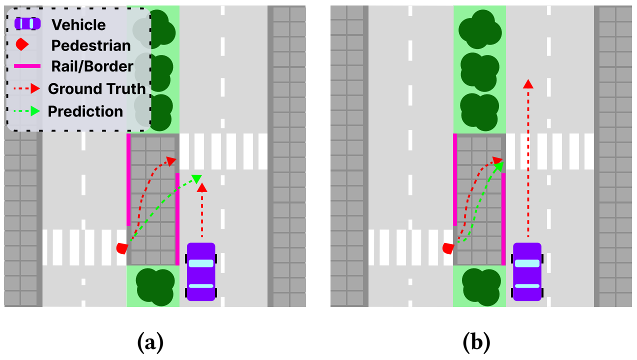
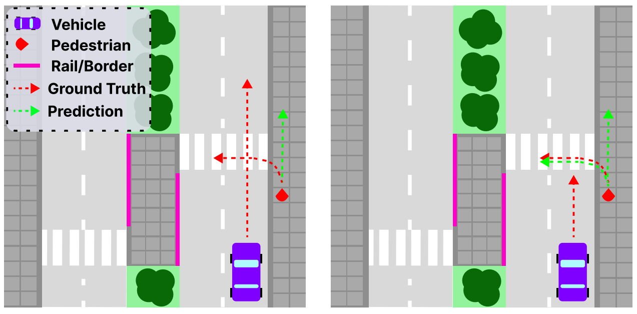


Figure 1: Traffic scenario in which a) the border constrains are violated, putting the trajectory on collision course with the vehicle and forcing a stop, and b) border constraints are respected, allowing the vehicle to pass the crossing.



(a) **(b)**

Figure 2: Traffic scenario in which a) the prediction is deterministic and the crossing is not considered and b) the prediction is stochastic, allowing the vehicle to evaluate multiple options.

Some state-of-the-art (SOTA) models [11, 12, 13, 14] also incorporate social interactions into their predictions, leveraging how the movement and trajectories of agents affect one another in a social context. Models such as these appear to produce plausible trajectories when viewed in this social context, but there is still work to be done when considering environmental constraints. [15] criticises previous work and the real-world datasets they are tested on, making note of the missing assessments of their ability to remain scene compliant and the simple environments of these datasets, which contain few to no scene constraints. [16] compares the SOTA approaches of [12, 17, 18], highlighting their inability to remain scene compliant when the movement of an agent is constricted by a more complex environment.

In much of recent work [12, 16, 17, 19, 20], the stochastic element is captured through Variational Autoencoders (VAEs) and Generational Adversarial Networks (GANs), but recently, a new type of generative model have appeared, namely Denoising Diffusion Probabilistic Models (DDPMs). Originally proposed by Sohl-Dickstein et al. as Diffusion Probabilistic Models to do image synthesis [21], the models were adapted by Ho et al. to birth Denoising Diffusion Probalistic Models[22]. In image synthesis, these models have seen wide adoption [23, 24, 25, 26], and have also found application in numerous other fields, including computer vision [27, 28, 29, 30], audio synthesis [31, 32, 33], and lately, also time series forecasting, imputation, and generation [34, 35, 36]. DDPMs do not suffer from the mode collapse and unstable training prevalent in GANs [26, 37, 38, 39], and have been demonstrated as capable of predicting the natural, scene-compliant trajectories for which VAEs have received critique[14, 40, 41]. However, DDPMs do have one prominent drawback in the context of trajectory prediction - inference time. The methodology of the model is to gradually destroy a sample, e.g. by injecting pure isotropic Gaussian noise, in a series of steps, and then reverse the process by having the model reconstruct, or *denoise* it, in a similar, but reverse, stepwise process. This can be computationally expensive and significantly slow down the inference process, as the number of steps varies across applications, ranging from 100s of steps as seen in [14] to 1000s in works such as [21, 22, 40]. [40] adopts a trajectory-on-scene approach, leveraging the U-net architecture, but this approach imposes a computational burden which makes the model unfeasible for real-time predictions. Likewise, using a CNN to encode a semantic map as seen in [14] still result in inference times which are impractical, and while their trajectories look natural, their experiments are conducted on the datasets critiqued in [15].

Generally, the trajectory prediction problem is approached as predicting the entire sequence in one shot [12, 16, 17, 40, 42], as opposed to predicting the points one at a time in a closed loop. Closed-loop predictions have a few inherent issues relating to the warranted stochastic, scene-compliant predictions mentioned previously; error propagation and expensive computation. The trajectory sequences found in evaluation datasets like [16, 43, 44] sample only short time frames (4.8-6 seconds) with 8-12 prediction steps, and while inaccurate initial predictions can lead to error accumulation in subsequent predictions, if a stochastic model adequately represents the data distribution, the impact of error propagation within these time frames is minimal. Even if an initial prediction deviates from the ground truth, it should still be plausible with a sufficiently capable stochastic model. Still, iterative prediction requires more computational resources and time compared to one-shot methods, as previous predictions need to be calculated before subsequent ones. This limitation hampers real-time inference but offers an advantage in terms of scene compliance, as the scene can be reevaluated from the perspective of the agent's next predicted position, and only the parts of the scene which are relevant to the next step; an interesting prospect in the context of DDPMs, as the prominent approach to increase inference speed in production-grade image generation DDPMs[24, 45], is to reduce the dimensionality of the predicted samples, and subsequently upsample them.

Preliminary works on DDPMs regarding our desired, scene-compliant trajectories have shown promising results, but the scarce literature warrants further research on both the subject of scene compliance and inference speed, as the current approaches are impractical. We address these limitations through a series of contributions, leveraging the capabilities of DDPM models in a closed-loop approach to generate natural, scene-compliant trajectories, and incorporate SOTA methods to confront the issues of inference speed. Our contributions can be summarised as follows:

- We propose a stochastic, scene-compliant model, SCTP, based on DDPMs, to address the inability of traditional approaches to generate natural-looking, collision-free trajectories. This approach is much easier to train compared to GANs as it does not suffer from unstable training and mode collapse, and compared to VAE-based approaches, it is capable of generating natural-looking trajectories.
- Leveraging the advantages of closed-loop predictions, we propose using a Euclidean distance, point cloud representation, incorporating only relevant parts of the scene to improve inference time and alleviate the computational burden imposed by including the scene in its entirety; a critical consideration when leveraging DDPMs.
- We make further improvements on the inference time of DDPMs for trajectory prediction, skipping large parts of the reverse diffusion process using a leapfrog technique, increasing inference performance by 100%. This has a negligible impact on the performance metrics and brings DDPM-based approaches much closer to real-time performance.
- We evaluate the performance of our proposed model on the PFSD benchmark dataset and compare it with existing SOTA trajectory prediction methods. The experimental results demonstrate that our model achieves competitive accuracy and outperforms previous models in terms of scene compliance.

2. Related Work

1. Trajectory Predictions

Trajectory prediction in the context of human motion presents a multitude of challenges that need to be addressed, not least of which is alignment with the environment to produce collision-free, realistic trajectories. In this section, we highlight recent work done in the field to address this issue.

Many works incorporate scene information as rasterised images or graphs [12, 16, 17, 18, 19, 20, 46], but as noted by [15], much of prior work lacks an evaluation of the performance benefits and ability to produce trajectories that respect environmental constraints. They also note that many real-world datasets are simple in nature, with the majority of samples containing little to no agent-to-environment interaction for pedestrian samples, as they usually navigate in open spaces. To combat this, their contributions include the Environment Collision-Free Likelihood (ECFL) metric as a means to evaluate a model's ability to generate collision-free trajectories and the A2X dataset which addresses the limitations of existing real datasets by incorporating simulated crowd egress and complex navigation scenarios, compensating for the absence of agent to environment interactions found in other datasets. The A2X dataset is based on the concept of social force, which was later adopted by [16] to birth the Path Finding Simulation Dataset (PFSD), a complex binary navigation environment, with both open spaces and multiple narrow corridors. [12] proposes a stochastic model which accounts for the dynamic movement of other agents (agent-to-agent interaction), and which is able to incorporate information about the scene specifically to avoid border violations (agent-to-environment interaction), where they encode the map using a CNN, the effect of which is an overall 1% border violation on datasets such as [47, 48], down from 4.6% without the map encoding according to their ablation study. Their work is used as a base of comparison in many recent SOTA models[16, 17, 42, 46]. However, their experiments are conducted on the datasets which received critique in [15]. [17] also uses a convolutional network to encode a map, but provide no quantitative measurements regarding collisions or border violations. [49] introduces the concept of goals under a time constraint to represent the innate desire of pedestrians to reach some predetermined destination. Using particle filters to uniformly model the probability distribution of goals as a discretised grid, they incorporate the map as a prior distribution, to reflect the likelihood that a pedestrian will enter some grid cell if occupied by an obstacle. [18] uses a more complex variant of the convolutional approach above, leveraging the U-net architecture to encode the information about the scene by adopting a trajectory-on-scene approach, in which they process and predict the trajectory in the same space as the map - a rasterised image. However, they too incorporate goals as predicted long-term and intermediate goals to capture the uncertainty of the future trajectories. They modify the U-net architecture to include two decoders - one which produces a goal and way-point heatmap, and one which produces a trajectory conditioned on the goal and waypoint heatmap. Their approach to incorporating goals was later used by [16], who proposed a three-stage model, where each stage predicts first the long-term goal, then the intermediate goals, and lastly the final trajectory. [16] also provide insight into the quantitative performance of [17], [12], and [18] on environmental alignment, showcasing that while they do take the same approach to map encoding, their performances vary across the datasets NuScenes, SDD, and PFSD [16, 43, 44]. Incorporating goals in this way inherently encodes information about the map, as was demonstrated in the ablation studies of [40] and [16]. [19] is another three-stage model, where the first stage proposes an initial set of targets, the second estimate a trajectory for each target, and the last ranks trajectory predictions and outputs a final set of K-predicted trajectories. They take two approaches to incorporate the scene, depending on the dataset. If the map is only available as a rasterised image, they use a convolutional approach as the previous approaches. If a High Definition semantic map is available, they make use of VectorNet[50], encoding the map by extracting features such as road lines, traffic signs, and other relevant information.

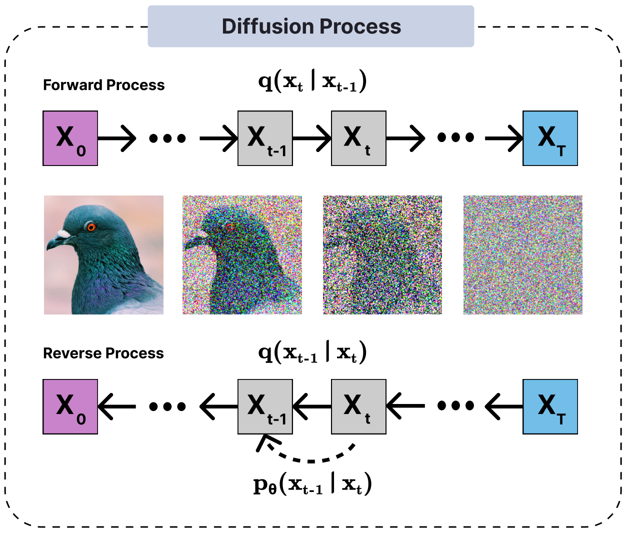


Figure 3: The Markov chains of forward and reverse diffusion: Each transition in the forward diffusion process adds noise to destroy structure in the data. Conversely, each transition in the reverse diffusion process removes noise to create structure in the data.

2. Denoising Diffusion Probabilistic Models

Research on the use of DDPMs in the trajectory prediction domain remains limited, but preliminary results have been favourable [14, 40, 42], however, the slow inference time of DDPMs poses a significant challenge. In production image generation models such as [24, 45], reduction of inference time is accomplished by generating an image of a much smaller dimensionality and subsequently upsampling it using a different technique. [51] and [52] explores a teacher-student approach, progressively distilling a teacher DDPM into a student DDPM, which requires fewer steps to sample from. [53] takes another approach, proposing Denoising Diffusion Implicit Models (DDIMs), which generalises the traditional Markovian process, on which DDPMs are based, to a non-Markovian one. They demonstrate that using non-Markovian diffusion processes can lead to a shorter generative Markov chain, resulting in increased sample efficiency with only a minor impact on sample quality. [42] specifically addresses the issue of inference time in trajectory prediction DDPMs by introducing a neural network which enables skipping a large amount of the initial diffusion steps through a leapfrogging technique, diffusing the first δ steps in one-shot, and thus drastically lowering the inference speed, while maintaining SOTA performance. [14] demonstrates a transformer-based trajectory prediction DDPM able to capture the scene and social context to produce paths comparable with SOTA. Their approach leverages the social encoder of [12] as a conditional framework. However, they also note the slow inference time, with their model being almost 40 times slower than the model on which they base their work[12]. [40] demonstrates another DDPM-based approach, in which they focus on diverse, collision-free trajectories. Their model takes a multi-stage, trajectory-on-scene approach, in which they layer predicted long-term goals, the observed trajectory and noised future trajectory on top of the scene to produce an image which is subsequently denoised. Their results demonstrate trajectories which are natural, diverse and close to collision-free, but they too note the slow inference time.

3. Preliminaries

1. Denoising Diffusion Probabilistic Models

The intuition behind DDPMs can be described as two separate, stochastic processes; the forward diffusion process, which gradually destroys a sample through a sequence of steps, and the reverse diffusion process, which reconstructs or *denoises* in a similar, but reversed, sequence of steps. This idea can be seen illustrated in Fig. 3. The forward process produces a sequence of stochastic variables, $(x_1, x_2, \ldots, x_T)$ through a Markov process. This sequence is obtained by applying the transition kernel $q(x_t|x_{t-1})$ to a data sample, x_0, from the training data. For the forward process we leverage the original work by [21], where the transition kernel is a Gaussian perturbation of the previous state.

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t;\ \sqrt{1-\beta_t}\mathbf{x}_{t-1},\ \beta_t\mathbf{I}), \quad (1)$$

where β_t is defined according to some fixed variance schedule, $\{\beta_t \in (0,1)\}_{t=1}^{T}$, such that $\beta_1 < \beta_2 < \ldots < \beta_T$. Defining the transition kernel as Gaussian perturbation allows us to marginalise the joint conditional probability $q(x_1, x_2, \cdots x_T|x_0)$, as the product of Gaussian probability functions is also a Gaussian probability function

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t;\ \sqrt{\bar{\alpha}_t}\mathbf{x}_0,\ (1-\bar{\alpha}_t)\mathbf{I}), \quad (2)$$

where $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ and $\alpha_t = 1 - \beta_t$. This allows us to obtain x_t directly without applying $q(x_t|x_{t-1})$ repeatedly. Given a data sample x_0 from the training set, we can obtain the noised version of the sample at that timestep x_t, according to the fixed variance schedule as

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad (3)$$

where $\epsilon \sim \mathcal{N}(\mathbf{0},\ \mathbf{I})$.

The reverse process then involves iterating backwards through the sequence $(x_T, x_{T-1}, \ldots, x_1)$, using the transition kernel $q(x_{t-1}|x_t)$. The probability distribution $q(x_{t-1}|x_t)$ is an unknown, but can be approximated using a neural network p_θ as illustrated in Fig. 3. As above, using Gaussian perturbation for the transition kernel, we have

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1};\ \mu_\theta(\mathbf{x}_t,\ t),\ \Sigma_\theta(\mathbf{x}_t,\ t)), \quad (4)$$

where $\mu_\theta(\mathbf{x}_t,\ t)$ is the mean and $\Sigma_\theta(\mathbf{x}_t,\ t)$ the variance of the predicted Gaussian probability distribution. For the reverse process, we leverage the work by [22], predicting the added noise rather than the mean of the conditional probability distribution as originally shown by [21]. We thus use the reparameterisation of the mean seen in [22]

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_0(x_t, t)) \quad (5)$$

where $\epsilon_\theta(x_t, t)$ is an approximator intended to predict the added noise ϵ to a noised sample x_t. We can then model the approximator $\epsilon_\theta(x_t, t)$ using a neural network with the training objective

$$\begin{aligned} L_t &= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\epsilon_t}\left[\|\epsilon_t - \epsilon_\theta(\mathbf{x}_t,\ t))\|^2\right] \quad (6)\\ &= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\epsilon_t}\left[\|\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon,\ t)\|^2\right] \quad (7)\end{aligned}$$

where x_0 is an uncorrupted sample from the training data set and $\epsilon\ \mathcal{N}(\mathbf{0},\ \mathbf{I})$. It is important to mention that we, similar to [22], also maintain a constant variance due to their empirical findings indicating that it has minimal impact on the outcome.

2. Problem Definition

For the scene compliant trajectory prediction problem described in Sec. 1, we define the past positions of an agent at times $t_1, t_2, \ldots, t_n$ as $Z = z_1, z_2, \ldots, z_n$, where $z_t \in \mathbb{R}^2$ represents coordinates in a Cartesian plane. The future and predicted position at times $t_{n+1}, t_{n+2}, \ldots, t_{n+j}$ are then defined as $Y = y_{n+1}, y_{n+2}, \ldots, y_{n+j}$ for future positions and $\hat{Y} = \hat{y}_{n+1}, \hat{y}_{n+2}, \ldots, \hat{y}_{n+j}$ for predicted positions, where $\hat{y}_t \in \mathbb{R}^2$ and $y_t \in \mathbb{R}^2$ are coordinates in the same Cartesian plane as Z.

Given a semantic map M, the task is then to predict $\hat{Y}$ such that any point $\hat{y}_t$ aligns with the environmental constraints, i.e. $\forall \hat{y} \in \hat{Y} E(\hat{y}) = 1$, where E is a binary representation of M with 1 representing navigable, collision-free space and 0 representing non-navigable space. In adhering to these environmental constraints, we also want to maintain performance and produce natural trajectories.

4. Methodology

1. Overview

For the trajectory prediction problem described in Sec. 3.2, we leverage DDPMs described in Sec. 3.1 as a stochastic generative framework, opting for a (i) closed-loop prediction approach, using each predicted timestep to predict the next in an autoregressive manner, and (ii) use polar coordinates to represent the trajectories as velocity changes relative to an agent's current heading. We use the same notation as Sec. 3.2 to describe these trajectories, denoting the past trajectory as pt, future trajectory ft, and predicted trajectories $\hat{ft}$. Given a semantic map representation M and the past trajectory pt as relative polar velocity changes, we predict $\hat{ft}_{n+1}$ using a transformer-based network modelling $p_\theta(\mathbf{x}_{t-1}|x_t)$ to diffuse a random Gaussian into x_0 and applying Eq. 5. The entire predicted trajectory $\hat{ft}$ can then be constructed by continually adding the newly predicted $\hat{ft}_{n+1}$, to the past trajectory pt, and diffusing a new random Gaussian to get $\hat{ft}_{n+2}$ in the same manner. We then obtain the future positions $\hat{Y}$ by converting the polar values of $\hat{ft}$ into Cartesian values and integrating to find the Cartesian coordinates.

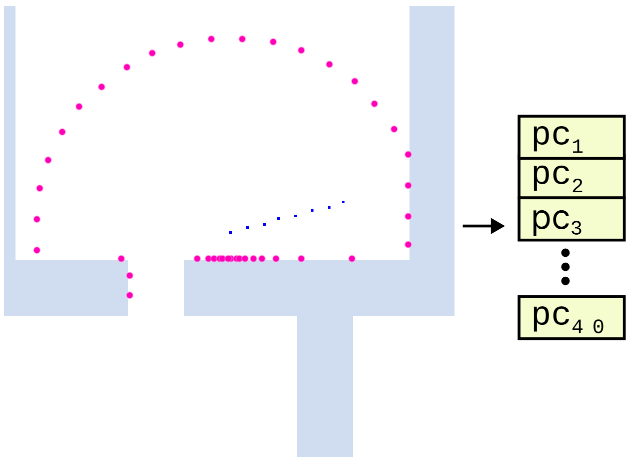


Figure 4: A depiction of the point cloud, used as input to the model, on the map showing how far each point reaches before it either hits max distance or a wall. The point cloud is represented in pink and the past trajectory is blue. The last point of the past trajectory, i.e. the point in the centre of the image, is the centre of the point cloud.

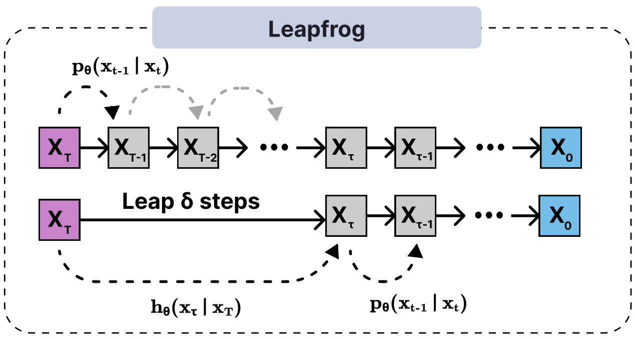


Figure 5: A regular diffusion reverse process at the top, with the corresponding diffusion process using a leap step to skip the first δ steps at the bottom.

We alleviate the DDPM inference burden highlighted in Sec. 1 through the application of strategies described in Sec. 2.2, by reducing the dimensionality of M, and, by employing a leapfrogging layer to skip large parts of the reverse diffusion process in Sec. 3.1. Under the closed-loop approach mentioned previously, we reduce the dimensionality of M by considering only the parts of the map relevant at each step. To accomplish this, we encode the relevant part as uniform, euclidean distance point cloud pc_u, capturing a limited number of points u in an arc centred around an agent's last known position, to reflect collision-free directions. This drastically reduces the dimensionality compared to a rasterised image representation of size $w \times h$, as in our case $w \times h \gg n$. We illustrate this point cloud representation on Fig. 4, where a rasterised representation of size 160×160 from the PFSD dataset is reduced to a point cloud of size 40. We then make further inference speed improvements by employing a teacher-student strategy, training a similar network to model a transition kernel $h_\theta(x_\tau | x_T)$ which skip δ steps of the reverse diffusion process, leaving p_θ to only diffuse the remaining τ steps. We illustrate this on Fig. 5.

2. Model Architecture

The model is based on the transformer-encoder architecture, which have previously been used as the foundations for DDPMs with SOTA performance. The transformer architecture affords us a couple of advantages. Firstly, Transformers are specifically designed for sequence modelling tasks, making them well-suited for trajectory prediction tasks where historical information plays a crucial role; and as noted and demonstrated by [14], they are well equipped to capture the temporal dependencies of trajectories. Secondly, a key component of the transformer architecture is the attention module, which is particularly beneficial for trajectory prediction as it allows the model to attend to the relevant spatial and temporal context while predicting future trajectories. In Fig. 6, we illustrate the model architecture from a training perspective. The model takes as input the point cloud representation pc_{40}, the past trajectory pt, and the next *noised* polar coordinate $\hat{fp}_{n+1}$ at some diffusion timestep t. Following [54], we embed t with a Sinusoidal Position Embedding, followed by two linear layers sandwiching a ReLU layer. This embedding is then concatenated with the contextual information pt and pc_n before passing it through a ConcatSquashLinear (CSL) upsampling layer to match the dimensions of the transformer-encoder. Before passing it to the transformer, we add a positional embedding to the input as is common in transformer-based networks. Lastly, using three CSL layers, we progressively downsample the output of the transformer to the original dimensions of the noised input. The loss between the estimated noise $\hat{\epsilon}$ and the actual noise ϵ is then computed using the Huber loss function seen in Eq. 8.

$$\mathcal{L}_{Huber} = \begin{cases} \frac{1}{2}(\epsilon - \hat{\epsilon})^2, & if\ |\epsilon - \hat{\epsilon}| < 1 \\ |\epsilon - \hat{\epsilon}| - \frac{1}{2}, & otherwise \end{cases} \quad (8)$$

In Fig. 5, we illustrate the leapfrogging technique, skipping δ diffusion steps, where $\delta = (T - \tau)$. The approach is similar to the above, but rather than modelling the transition kernel $p_\theta(x_{t-1}|x_t)$ for all timesteps, we introduce a transition kernel $h_\theta(x_\tau | x_T)$

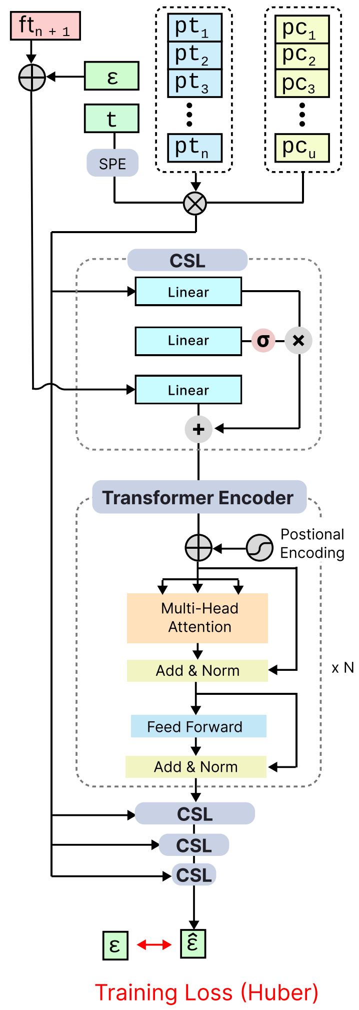


Figure 6: Diagram of our model architecture, showing how the different inputs are combined before being passed through the model. $\times N$ denotes the number of transformer layers.

skipping δ steps in a single step, where x_T is pure Gaussian noise and x_τ is the noised sample at δ skipped steps. Unlike the leapfrog technique described in [42] where they construct separate modules to predict the mean and variance, we take inspiration from the distilling approach described in Sec. 2, and use the previously described model as a teacher to train the leapfrog model as a student. The leapfrog model is then tasked with diffusing x_T to x_τ in one step. During training, the teacher model is sampled δ steps, starting from the same pure Gaussian noise as given to the student model, to get the noise at x_τ, the training target of the student model. Like the teacher, the student also uses the Huber loss function (Eq. 8).

At inference, the process is twofold; the leapfrog model is given the pure Gaussian noise at x_T and

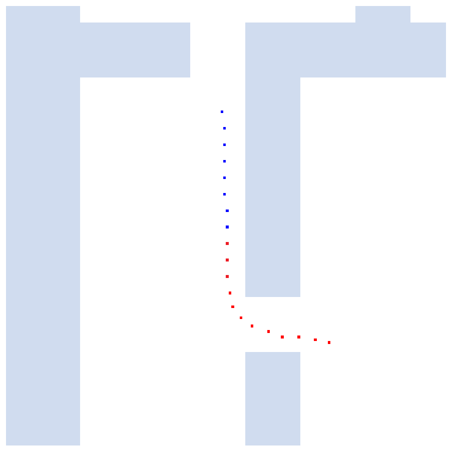


Figure 7: A sample from the PFSD dataset showing a scene with the past trajectory shown in blue and future trajectory shown in red.

outputs the intermediate noise at x_τ. This is subsequently passed to the original model to further denoise in the remaining τ steps, to arrive at x_0. Utilising the leapfrog model during inference, enables the whole diffusion process to be reduced from T diffusion steps to $(\tau + 1)$ steps.

5. Experiments

1. Experimental Setup

We conduct our experiments on the PFSD dataset which is a synthetic dataset comprising a total of 50,000 scenes. Leveraging the Social Force Model, this dataset simulates the navigation of agents within scenes that are more complex to navigate than that of traditional datasets like [43, 44, 47, 48]. The PFSD dataset provides a native binary environment consisting only of navigable and non-navigable space. This makes the dataset ideal for our purposes, as it allows us to demonstrate the model's ability to navigate between obstacles, whereas the aforementioned datasets have different classes of navigable space, i.e. pavement, roads, terrains and medians that divide them. This proves a significant issue for the approach mentioned in Sec. 4.1, as trajectories usually follow the navigable space they are on, but may cross into other classes. The point cloud representation mentioned in Sec. 4.1 is not sufficient to capture this, as it only relies on the distance to obstacles, leaving the model to consider all navigable spaces as equal. The dataset provides trajectories as sequences of 20 points, split into 8 observed points and 12 future points. These points include the actual position, the velocity, and the acceleration. The scene provided is a localised view of the environment, centred around the agent's last observed position. In Fig. 7, we have included a sample from the dataset, plotting out the observed and future trajectories on the corresponding scene. As mentioned in Sec. 4.1, we predict the velocities as relative polar coordinates rather than Cartesian coordinates. PFSD provides its values as Cartesian, so we convert them to relative polar for training. The point cloud pc is constructed by taking the agent's last observed position and determining the distance to nearby obstacles in an arc around the position. For illustration, we refer to the previously mentioned Fig. 4, where we use a radius r of 70 pixels and a number n of 40 points to represent the environment, as is done in the experiment. The dataset is split into training, validation, and testing sets, following common conventions and best practices.

We train both the teacher and student for 5000 epochs, using the Adam optimiser [55] with a learning rate of 0.001. We use a linear scheduler to obtain beta values, with values starting at 0.0001 and progressing to 0.02. Both models are then trained on a batch size of 128. The transformer-encoder is of size 64, with 4 attention heads, 3 layers, and a feedforward layer of size 256. The first CSL layer upsamplse to size 64, while the subsequent ones progressively downsample the output of the transformer from 64 to 32, then further to 16 dimensions, and finally to 2 dimensions. These parameters were found through empirical experiments, leaving tuning as future work.

2. Evaluation Metrics

For evaluating the performance of our approach, we employed the standard evaluation metrics minimum Average Displacement Error ($minADE$), minimum Final Displacement Error ($minFDE$) as is common in most SOTA approaches [12, 16, 17, 18, 40], and crucially, Environment Collision-Free Likelihood ($ECFL$) [15, 16, 40], to showcase the models ability to generate natural, collision-free trajectories. These metrics enable a quantitative assessment of the accuracy and usability of our model. Prior to evaluation, the predicted relative polar velocities are converted back to Cartesian positions to reflect the expected units of the formulas.

The average displacement error denotes the average Euclidean distance between each pair of the predicted positions and the actual future positions. The minimum ADE is then the best of these values across k predictions. The formula for $minADE$ can be seen in Eq. 9

$$minADE(\tilde{Y}, Y) = \min_{i=1}^{k} ADE(\hat{Y}_i, Y) \tag{9}$$

where $\tilde{Y} = [\hat{Y}_1, \hat{Y}_2 \ldots, \hat{Y}_k]$, the set of k predicted future trajectories. The final displacement error denotes the Euclidean distance between the last predicted point of the predicted position and the actual future position. The minimum FDE is then the best of k predictions. The formula for $minFDE$ can be seen in Eq. 10

$$minFDE(\tilde{Y}, Y) = \min_{i=1}^{k} FDE(\hat{Y}_i, Y), \tag{10}$$

where $\tilde{Y}$ again is the set of predicted future trajectories. $ECFL$ reports the probability that the model predicts a path that is free of collision with the environment. For all k predictions, the percentage of collision-free predicted trajectories are computed using the formula seen in Eq. 11

$$ECFL(\tilde{Y}, E) = \frac{1}{k} \sum_{i=1}^{k} \prod_{t=n+1}^{t_{n+j}} E[\tilde{Y}_{i,t,0}, \tilde{Y}_{i,t,1}]. \tag{11}$$

where E is the binary matrix representation of the environment, with values of 1 representing navigable space and values of 0 representing non-navigable space. The metric classifies all predictions into either 1 or 0, depending on whether any point on the predicted trajectory is positioned inside the environment. The classifications are then averaged across all predictions to find the percentage of colliding trajectories.

3. Quantitative Results

The evaluation of our model on the PFSD dataset shows our diffusion-based model with leapfrog skipping $\delta = 50$ of the first $\Gamma = 100$, is able to outperform existing SOTA methods in most metrics, both on $k = 20$ trajectories and $k = 50$ trajectories. The results can be seen listed in Tab. 1. While a closer look reveals that our model on $k = 20$ outperforms all other methods on $minADE$ and $ECFL$, the $minFDE$ metrics still remain competitive in its results, by only differing 0.02 from the best model AgentFormer. Our model with $k = 50$ shows results that either surpass or match current SOTA models. Compared to $k = 20$ it now matches the results of AgentFormer on $minFDE$, with a result of 0.09. More noticeable are the other metrics $minADE$ and $ECFL$, which outperform SOTA; $minADE$ improves compared to $k = 20$, while the $ECFL$ metric takes a small hit of 0.01%.

Table 1: Results on the PFSD dataset for $k = 20$ and $k = 50$ predictions, using the leapfrogging method to skip 50% of the diffusion steps. The predictions use 3.2 seconds (8 frames) for observation and 4.8 seconds (12 frames) for the prediction. The errors are measured in meters and the best results are shown in **bold** with the second best represented by underline.

PFSD Results				
k	Model	minADE ↓	minFDE ↓	ECFL ↑
20	Trajectron++ [12]	0.17	0.37	83.22
	Y-Net [18]	0.13	0.20	91.52
	AgentFormer [17]	<u>0.08</u>	**0.11**	94.54
	MUSE-VAE [16]	0.09	0.19	97.40
	MEAT-DDPM [40]	0.34	0.36	<u>99.08</u>
	Ours	**0.07**	<u>0.13</u>	**99.86**
50	Trajectron++ [12]	0.14	0.25	83.39
	Y-Net [18]	0.09	<u>0.12</u>	91.74
	AgentFormer [17]	0.08	**0.09**	95.37
	MUSE-VAE [16]	<u>0.07</u>	0.13	97.50
	MEAT-DDPM [40]	0.31	0.28	<u>99.11</u>
	Ours	**0.06**	**0.09**	**99.85**

4. Qualitative Results

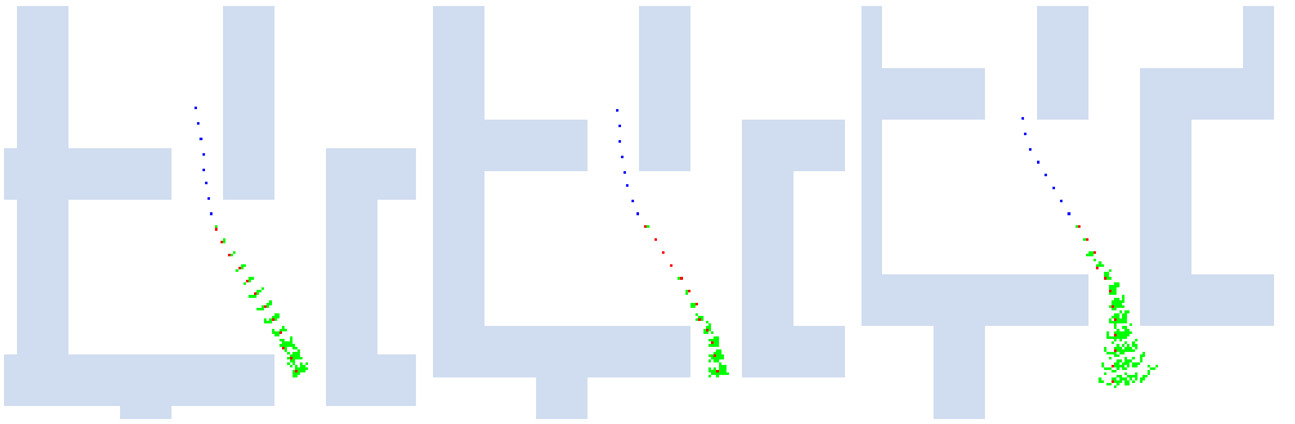
We illustrate the performance of our model by showcasing the model's performance as it moves through the environment, and a few cases demonstrating the predictions on various environments of the PFSD dataset. In Fig. 8 we see the trajectories trend towards the closest opening from its current heading, demonstrating that our model does indeed interpret the environment in a sufficient manner. As it moves through the environment, the trajectories then fan out as the included environmental details pose no further restrictions on the possible trajectories - the space in front of it is more or less open. In Fig. 9 we see 5 cases from the predictions on the PFSD dataset. Case 1 shows the model's ability to capture the temporality of the past trajectory in relation to the map, discontinuing the turn and proceeding straight ahead in accordance with the environment. Cases 2 and 3 show the model is capable of generating multiple different trajectories based on the environment. Both cases contain adjacent rooms which are accessible, showcasing the model's ability to capture the stochastic nature of trajectory prediction. Cases 4 and 5 show scenarios where there is only one possible direction, based on the past trajectory, and highlight the model's ability to understand and follow the environment towards the only opening ahead, while still fanning out in multiple possible directions as it moves through the corridor and into open space.

Table 2: Results of the ablation study on the PFSD dataset using $k = 20$ predictions and $k = 50$, 3.2 seconds (8 frames) for observation and 4.8 seconds (12 frames) for prediction. The errors are measured in meters and the best results are shown in **bold** with the second best represented by underline.

Ablation Study				
k	Leapfrog steps	minADE ↓	minFDE ↓	ECFL ↑
20	0% (baseline)	**0.06**	**0.12**	99.83
	25%	<u>0.07</u>	0.15	**99.89**
	50%	<u>0.07</u>	<u>0.13</u>	<u>99.86</u>
	75%	0.12	0.29	95.02
	90%	1.41	2.91	78.82
50	0% (baseline)	**0.05**	**0.08**	99.83
	25%	<u>0.06</u>	0.10	**99.86**
	50%	<u>0.06</u>	<u>0.09</u>	<u>99.85</u>
	75%	0.08	0.15	94.90
	90%	1.19	2.45	78.99

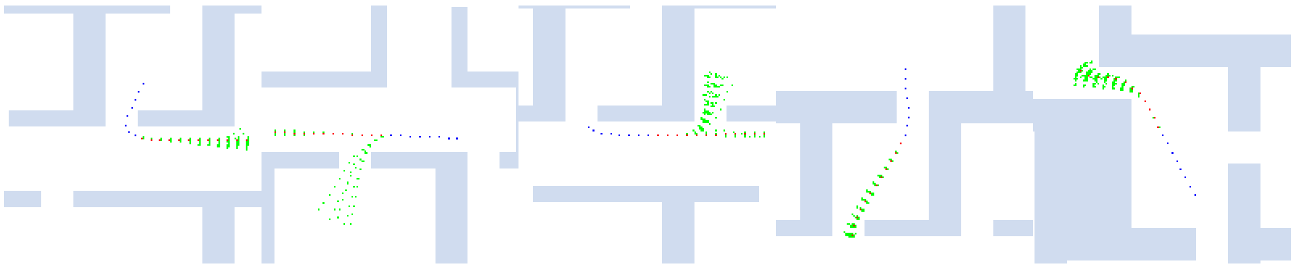
5. Ablation Study

In Tab. 2, we present an ablation study that aims to investigate the influence of skipping different proportions of diffusion steps using the leapfrog method discussed in Sec. 4.2. The purpose of this study is to understand the impact of reducing the computational cost of the diffusion process on the overall performance of our proposed model. We establish a baseline by considering the results obtained when none of the diffusion steps are skipped. The baseline model is competitive with AgentFormer, closing the gap on $minFDE_{k=20}$ by 0.01, and outperforms remaining models on all metrics, similar to the proposed solution as seen in Sec. 5.3. Next, we explore the effects of skipping 25% and 50% of the diffusion steps. These reductions have a minimal impact on the performance, and the model remains competitive with other SOTA models, as described in Sec. 5.3. This suggests that certain portions of the diffusion process can be safely omitted using the leapfrogging technique, leading to a significant reduction in computational requirements while maintaining comparable performance. However, as we proceed to skip 75% and 90% of the diffused steps, the negative impact on the model becomes evident. Across all evaluation metrics - $minFDE$, $minADE$, and $ECFL$ - the performance degrades significantly with the increasing number of skipped steps. These results highlight the importance of a sufficient number of diffusion steps for capturing intricate details and predicting accurate future trajectories. Nonetheless, it is worth noting that the ablation study reveals a noteworthy improvement: by reducing the inference time of the diffusion process by 50%



(a) $Y = y_{n+1} \cdots y_{n+12}$ **(b)** $Y = y_{n+3} \cdots y_{n+15}$ **(c)** $Y = y_{n+6} \cdots y_{n+18}$

Figure 8: The figure shows k = 50 prediction at three different points in time, to illustrate the models ability to evaluate the environment as the agent progresses through it. They are spaced 1.2 seconds apart (3 frames). The blue points represents past trajectory, red points are ground truth future trajectory and green is the predicted future trajectories.



(a) Case 1 **(b)** Case 2 **(c)** Case 3 **(d)** Case 4 **(e)** Case 5

Figure 9: The figure shows k = 50 prediction at 5 differing environments from the dataset PFSD, demonstrating its ability to interpret the map and capture the stochastic element of trajectory predictions.

with negligible performance impact, we achieve a drastic increment in computational efficiency. Overall, the ablation study provides insights into the trade-off between computational efficiency and prediction performance in our proposed model, shedding light on the significance of the diffusion steps and offering a balance between accuracy and efficiency in trajectory prediction tasks.

6. Conclusion

In this paper, we make further advancements in the field of trajectory prediction, demonstrating DDPMs as a viable option for generating scene-compliant trajectories. The proposed model, SCTP, predicts trajectories in a closed-loop, using previous predictions to predict subsequent ones and a point cloud to represent the environment. The point cloud is a collection of Euclidean distance values uniformly distributed in an arc around the agent's last position, reflecting collision-free directions as perceived by the agent. This proves an effective way of representing the map, as the model scores best in class on the intended PFSD dataset, outperforming other SOTA models on the collision evaluation metric ECFL and average displacement metric minADE, and remains competitive on the final displacement metric minFDE, as can be seen in Sec. 5.3. We make further demonstrations in Sec. 5.4, showcasing the model's ability to generate natural, scene-compliant trajectories which align with the environment. Following recent advancements in the field, we built our model on the transformer architecture, and address the notoriously slow inference time of DDPMs through the reduced semantic map representation of the point cloud, and by employing a leapfrogging stage during inference, skipping 50% of the diffusion steps with a negligible performance impact. The leapfrogging stage alone thus nets us a 100% increase in inference performance, bringing DDPM-based trajectory prediction models much closer to viable, real-time predictions.

1. Future Work and Limitations

The PFSD dataset, on which we conduct our experiments, does not contain interactions with other agents, as has been done in some prior work (see Sec. 1). Research into this aspect of trajectory prediction is thus warranted to further the advancement of DDPMs as viable candidates for trajectory prediction. Furthermore, the model operates under continuous movement in a binary environment, which leaves several points for future work. Firstly, the model is incapable of handling datasets such as those provided by [43, 44] in a sufficient manner, due to the n-ary environment detailed in Sec. 5.1. Secondly, while the model is able to aptly interpret the environmental constraints, the point cloud representation is not adequate to represent neither the complex intents of pedestrians or points of interest, as it is composed of Euclidean collision distances. Lastly, further experiments are required to cement our approach as a general solution. Evaluating the model under different circumstances than continuous movements, such as an agent coming to a stop, slowing down significantly, or when making more erratic movements would strengthen that position.

On the subject of inference time, and despite our efforts in reducing the computational burden of the forward diffusion process, there is still room for improvement, and augmenting our approach with another technique, or replacing it with a better alternative is also a possible point for future work.

7. Acknowledgements

In our efforts to develop a practical and real-time capable model that remains compliant with the environment, we would like to express our appreciation to some individuals and institutions. First and foremost, we would like to show gratitude to our supervisor, Jilin Hu, and co-supervisor, Haomin Yu, from Aalborg University, for their invaluable advice and guidance throughout the research process. Additionally, we would like to express our gratitude to the University of Aalborg for generously providing the necessary computational resources to carry out our research. The support from AAU Claudia has been of tremendous significance, enabling us to conduct our work effectively.

References

[1] Mengmeng Liu et al. *LAformer: Trajectory Prediction for Autonomous Driving with Lane-Aware Scene Constraints.* 2023. arXiv: 2302 . 13933 [cs.CV].

[2] Scott Ettinger et al. *Large Scale Interactive Motion Forecasting for Autonomous Driving : The Waymo Open Motion Dataset.* DOI: 10 . 48550 / ARXIV.2104.10133. URL: https://arxiv.org/abs/2104.10133.

[3] Edward Schmerling et al. *Multimodal Probabilistic Model-Based Planning for Human-Robot Interaction.* 2017. DOI: 10 . 48550 / ARXIV . 1710 . 09483. URL: https://arxiv.org/abs/1710.09483.

[4] Christoph Rösmann et al. "Online trajectory prediction and planning for social robot navigation". In: *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM).* 2017, pp. 1255–1260. DOI: 10.1109/AIM.2017.8014190.

[5] François de La Bourdonnaye, Rossitza Setchi, and Cecilia Zanni-Merk. "Gaze Trajectory Prediction in the Context of Social Robotics". In: *IFAC-PapersOnLine* 49.19 (2016), pp. 126–131. ISSN: 2405-8963. DOI: https : / / doi . org / 10 . 1016 / j . ifacol . 2016 . 10 . 473. URL: https : / / www . sciencedirect . com / science / article / pii / S2405896316320638.

[6] Vahid Bastani, Lucio Marcenaro, and Carlo S. Regazzoni. "Online Nonparametric Bayesian Activity Mining and Analysis From Surveillance Video". In: *IEEE Transactions on Image Processing* 25.5 (2016), pp. 2089–2102. DOI: 10 . 1109 / TIP . 2016.2540813.

[7] Liang Lin et al. "Integrating Graph Partitioning and Matching for Trajectory Analysis in Video Surveillance". In: *IEEE Transactions on Image Processing* 21.12 (Dec. 2012), pp. 4844–4857. DOI: 10.1109/tip.2012.2211373. URL: https://doi.org/10.1109%2Ftip.2012.2211373.

[8] Peter Xenopoulos and Claudio Silva. "Graph Neural Networks to Predict Sports Outcomes". In: *2021 IEEE International Conference on Big Data (Big Data).* 2021, pp. 1757–1763. DOI: 10. 1109/BigData52589.2021.9671833.

[9] Ding Ding and H. Howie Huang. "A Graph Attention Based Approach for Trajectory Prediction in Multi-agent Sports Games". In: *CoRR* abs/2012.10531 (2020). arXiv: 2012 . 10531. URL: https://arxiv.org/abs/2012.10531.

[10] Junfeng Zhang et al. "Online four dimensional trajectory prediction method based on aircraft intent updating". In: *Aerospace Science and Technology* 77 (2018), pp. 774–787. ISSN: 1270-9638. DOI: https://doi.org/10.1016/j.ast.2018.03.037. URL: https://www.sciencedirect.com/science/article/pii/S1270963816313414.

[11] Alexandre Alahi et al. "Social LSTM: Human Trajectory Prediction in Crowded Spaces". In:

2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016, pp. 961–971. DOI: 10.1109/CVPR.2016.110.

[12] Tim Salzmann et al. "Trajectron++: Multi-Agent Generative Trajectory Forecasting With Heterogeneous Data for Control". In: *CoRR* abs/2001.03093 (2020). arXiv: 2001.03093. URL: http://arxiv.org/abs/2001.03093.

[13] Agrim Gupta et al. "Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks". In: *CoRR* abs/1803.10892 (2018). arXiv: 1803.10892. URL: http://arxiv.org/abs/1803.10892.

[14] Tianpei Gu et al. *Stochastic Trajectory Prediction via Motion Indeterminacy Diffusion*. 2022. arXiv: 2203.13777 [cs.CV].

[15] Samuel S. Sohn et al. "A2X: An Agent and Environment Interaction Benchmark for Multimodal Human Trajectory Prediction". In: *Proceedings of the 14th ACM SIGGRAPH Conference on Motion, Interaction and Games*. MIG '21. Virtual Event, Switzerland: Association for Computing Machinery, 2021. ISBN: 9781450391313. DOI: 10.1145/3487983.3488302. URL: https://doi.org/10.1145/3487983.3488302.

[16] Mihee Lee et al. "MUSE-VAE: Multi-Scale VAE for Environment-Aware Long Term Trajectory Prediction". In: *CoRR* abs/2201.07189 (2022). arXiv: 2201.07189. URL: https://arxiv.org/abs/2201.07189.

[17] Ye Yuan et al. "AgentFormer: Agent-Aware Transformers for Socio-Temporal Multi-Agent Forecasting". In: *CoRR* abs/2103.14023 (2021). arXiv: 2103.14023. URL: https://arxiv.org/abs/2103.14023.

[18] Karttikeya Mangalam et al. *From Goals, Waypoints & Paths To Long Term Human Trajectory Forecasting*. Dec. 2020.

[19] Hang Zhao et al. "TNT: Target-driveN Trajectory Prediction". In: *CoRR* abs/2008.08294 (2020). arXiv: 2008.08294. URL: https://arxiv.org/abs/2008.08294.

[20] Amir Sadeghian et al. "SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints". In: *CoRR* abs/1806.01482 (2018). arXiv: 1806.01482. URL: http://arxiv.org/abs/1806.01482.

[21] Jascha Sohl-Dickstein et al. "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In: *CoRR* abs/1503.03585 (2015). arXiv: 1503.03585. URL: http://arxiv.org/abs/1503.03585.

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *CoRR* abs/2006.11239 (2020). arXiv: 2006.11239. URL: https://arxiv.org/abs/2006.11239.

[23] Shelly Sheynin et al. *KNN-Diffusion: Image Generation via Large-Scale Retrieval*. 2022. DOI: 10.48550/ARXIV.2204.02849. URL: https://arxiv.org/abs/2204.02849.

[24] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2021. DOI: 10.48550/ARXIV.2112.10752. URL: https://arxiv.org/abs/2112.10752.

[25] Alex Nichol et al. *GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models*. 2021. DOI: 10.48550/ARXIV.2112.10741. URL: https://arxiv.org/abs/2112.10741.

[26] Prafulla Dhariwal and Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*. 2021. DOI: 10.48550/ARXIV.2105.05233. URL: https://arxiv.org/abs/2105.05233.

[27] Emiel Hoogeboom et al. *Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions*. 2021. DOI: 10.48550/ARXIV.2102.05379. URL: https://arxiv.org/abs/2102.05379.

[28] Tomer Amit et al. *SegDiff: Image Segmentation with Diffusion Probabilistic Models*. 2021. DOI: 10.48550/ARXIV.2112.00390. URL: https://arxiv.org/abs/2112.00390.

[29] Haoying Li et al. *SRDiff: Single Image Super-Resolution with Diffusion Probabilistic Models*. 2021. DOI: 10.48550/ARXIV.2104.14951. URL: https://arxiv.org/abs/2104.14951.

[30] Dmitry Baranchuk et al. *Label-Efficient Semantic Segmentation with Diffusion Models*. 2021. DOI: 10.48550/ARXIV.2112.03126. URL: https://arxiv.org/abs/2112.03126.

[31] Lilac Atassi. *Generating symbolic music using diffusion models*. 2023. arXiv: 2303.08385 [cs.SD].

[32] Hongfei Wang. *DiffuseRoll: Multi-track multi-category music generation based on diffusion model*. 2023. arXiv: 2303.07794 [cs.SD].

[33] Giorgio Mariani et al. *Multi-Source Diffusion Models for Simultaneous Music Generation and Separation*. 2023. arXiv: 2302.02257 [cs.SD].

[34] Shuhan Zheng and Nontawat Charoenphakdee. *Diffusion models for missing value imputation in tabular data*. 2023. arXiv: 2210.17128 [cs.LG].

[35] Juan Miguel Lopez Alcaraz and Nils Strodthoff. *Diffusion-based Conditional ECG Generation with Structured State Space Models*. 2023. arXiv: 2301.08227 [eess.SP].

[36] Ping Chang et al. *TDSTF: Transformer-based Diffusion probabilistic model for Sparse Time series Forecasting*. 2023. arXiv: 2301.06625 [cs.LG].

[37] Bhagyashree, Vandana Kushwaha, and G. C. Nandi. "Study of Prevention of Mode Collapse in Generative Adversarial Network (GAN)". In: *2020 IEEE 4th Conference on Information & Communication Technology (CICT)*. 2020, pp. 1–6. DOI: 10.1109/CICT51604.2020.9312049.

[38] Google Developer. *[GAN] Common Problems*. Accessed 16-12-2022. URL: https://developers.google.com/machine-learning/gan/problems.

[39] Florinel-Alin Croitoru et al. "Diffusion Models in Vision: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023), pp. 1–20. DOI: 10.1109/tpami.2023.3261988. URL: https://doi.org/10.1109%2Ftpami.2023.3261988.

[40] Mads Bach Andersen et al. *MEAT-DDPM: Multi-future Environment Aligned Trajectories using a Denoising Diffusion Probabilistic Model*. 2023. URL: https://projekter.aau.dk/projekter/da/studentthesis/meatddpm-multifuture-environment-aligned-trajectories-using-a-denoising-diffusion-probabilistic-model(a9532a52-d3ea-41f4-959c-226015d89a16).html.

[41] Rongqin Liang et al. *STGlow: A Flow-based Generative Framework with Dual Graphormer for Pedestrian Trajectory Prediction*. 2022. DOI: 10.48550/ARXIV.2211.11220. URL: https://arxiv.org/abs/2211.11220.

[42] Weibo Mao et al. *Leapfrog Diffusion Model for Stochastic Trajectory Prediction*. 2023. arXiv: 2303.10895 [cs.CV].

[43] Alexandre Robicquet et al. "Learning Social Etiquette: Human Trajectory Understanding In Crowded Scenes". In: *European Conference on Computer Vision*. 2016.

[44] Holger Caesar et al. *nuScenes: A multimodal dataset for autonomous driving*. 2019. DOI: 10.48550/ARXIV.1903.11027. URL: https://arxiv.org/abs/1903.11027.

[45] openai. *DALL·E 2 — openai.com*. https://openai.com/dall-e-2. [Accessed 09-Jun-2023].

[46] Sriram Narayanan et al. "Divide-and-Conquer for Lane-Aware Diverse Trajectory Prediction". In: *CoRR* abs/2104.08277 (2021). arXiv: 2104.08277. URL: https://arxiv.org/abs/2104.08277.

[47] Stefano Pellegrini, Andreas Ess, and Luc Van Gool. "Improving Data Association by Joint Modeling of Pedestrian Trajectories and Groupings". In: Sept. 2010, pp. 452–465. ISBN: 978-3-642-15548-2. DOI: 10.1007/978-3-642-15549-9_33.

[48] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. "Crowds by Example". In: *Computer Graphics Forum* (2007). ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2007.01089.x.

[49] Eike Rehder and Horst Kloeden. "Goal-Directed Pedestrian Prediction". In: *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*. 2015, pp. 139–147. DOI: 10.1109/ICCVW.2015.28.

[50] Jiyang Gao et al. "VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation". In: *CoRR* abs/2005.04259 (2020). arXiv: 2005.04259. URL: https://arxiv.org/abs/2005.04259.

[51] Tim Salimans and Jonathan Ho. "Progressive Distillation for Fast Sampling of Diffusion Models". In: *CoRR* abs/2202.00512 (2022). arXiv: 2202.00512. URL: https://arxiv.org/abs/2202.00512.

[52] Chenlin Meng et al. *On Distillation of Guided Diffusion Models*. 2023. arXiv: 2210.03142 [cs.CV].

[53] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising Diffusion Implicit Models". In: *CoRR* abs/2010.02502 (2020). arXiv: 2010.02502. URL: https://arxiv.org/abs/2010.02502.

[54] Niels Rogge and Kashif Rasul. *The Annotated Diffusion Model*. June 2022. URL: https://huggingface.co/blog/annotated-diffusion.

[55] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].