2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

KoopCast: Trajectory Forecasting via Koopman Operators

Jungjin Lee Jaeuk Shin Gihwan Kim Joonho Han Insoon Yang

***Abstract*— We present KoopCast, a lightweight yet efficient model for trajectory forecasting in general dynamic environments. Our approach leverages Koopman operator theory, which enables a linear representation of nonlinear dynamics by lifting trajectories into a higher-dimensional space. The framework follows a two-stage design: first, a probabilistic neural goal estimator predicts plausible long-term targets, specifying *where to go*; second, a Koopman operator-based refinement module incorporates intention and history into a nonlinear feature space, enabling linear prediction that dictates *how to go*. This dual structure not only ensures strong predictive accuracy but also inherits the favorable properties of linear operators while faithfully capturing nonlinear dynamics. As a result, our model offers three key advantages: (i) competitive accuracy, (ii) interpretability grounded in Koopman spectral theory, and (iii) low-latency deployment. We validate these benefits on ETH/UCY, the Waymo Open Motion Dataset, and nuScenes, which feature rich multi-agent interactions and map-constrained nonlinear motion. Across benchmarks, KoopCast consistently delivers high predictive accuracy together with mode-level interpretability and practical efficiency.**

I. Introduction

Trajectory forecasting of dynamic agents, such as pedestrians and surrounding vehicles, is a cornerstone of reliable robotic navigation systems. Yet, despite extensive research, forecasting remains intrinsically difficult: the intentions of dynamic agents are unobservable, their motion patterns are shaped by complex environmental contexts, and the resulting trajectories exhibit strong nonlinearity. To cope with these challenges, many state-of-the-art models employ large neural networks [1], [2], [3], [4], which achieve strong benchmark performance. However, these approaches come with critical limitations: their black-box nature undermines interpretability and trust in safety-critical decision-making pipelines, and their substantial computational demands hinder real-time deployment in downstream planning and control. This gap underscores the need for forecasting frameworks that combine predictive accuracy with transparency and computational efficiency, ideally by exploiting structured mathematical representations, such as linear operator models, to faithfully capture nonlinear dynamics while remaining interpretable and lightweight.

In the pursuit of interpretable and computationally efficient data-driven models for nonlinear dynamics, the Koopman operator has recently gained prominence in the robotics community [5], [6], [7], [8]. This framework lifts nonlinear systems into high-dimensional linear representations, enabling data-driven methods to learn the parameters of the lifted dynamics. By virtue of its linear structure, the Koopman operator supports fast execution and consistent long-term behavior. Moreover, this linearity facilitates a principled analysis of the original nonlinear system, allowing it to be interpreted through standard linear algebraic tools [9]. Consequently, the Koopman operator framework offers a distinctive combination of rapid inference, strong interpretability, and competitive accuracy, making it particularly well suited for trajectory forecasting in robotic navigation.

Building on these advantages, we introduce KoopCast, a trajectory forecasting model that leverages Koopman operators in a data-driven setting. To the best of our knowledge, KoopCast is the first model to perform trajectory forecasting by explicitly learning Koopman operators. By representing nonlinear dynamics through lifted linear systems, KoopCast delivers strong performance across several dimensions: achieving high predictive accuracy, ensuring stable trajectory generation via spectral analysis, and enabling efficient training and inference through its lightweight structure.

Concretely, our approach leverages Koopman operator theory to represent dynamics as linear in a high-dimensional lifted space, incorporating both intent and history into nonlinear observables. This is particularly powerful because, once an agent's high-level intent (e.g., reaching a goal) is specified, its motion typically follows consistent nonlinear dynamics rather than being governed solely by uncertainty [10], [11], [12]. This structure enables accurate trajectory refinement via a linear representation. Accordingly, KoopCast performs prediction in two stages: (i) goal estimation that integrates history, structural context (e.g., map and lane information), and interactions with other agents; and (ii) trajectory refinement, where the Koopman operator captures nonlinear motion within a linear framework.

Through this Koopman operator-based trajectory reconstruction, our framework not only improves predictive accuracy but also enables interpretation and analysis of agent dynamics through the lens of linear algebra. Extending this perspective, KoopCast may further contribute to a deeper understanding of pedestrian and vehicle behaviors in future research. At the same time, it achieves low-latency inference via a simple operator-multiplication structure, making it well suited for autonomous driving applications. Together, these properties highlight the significance of KoopCast in ensuring reliable navigation performance when integrated with downstream planners or controllers.

This work was supported in part by the Information and Communications Technology Planning and Evaluation (IITP) grants funded by MSIT No. 2022-0-00124, No. 2022-0-00480 and No. RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University).

All authors are with the Department of Electrical and Computer Engineering and ASRI, Seoul National University, Seoul, 08826, Korea {jungbbal, sju5379, kgh3115, snowhan1021, insoonyang}@snu.ac.kr

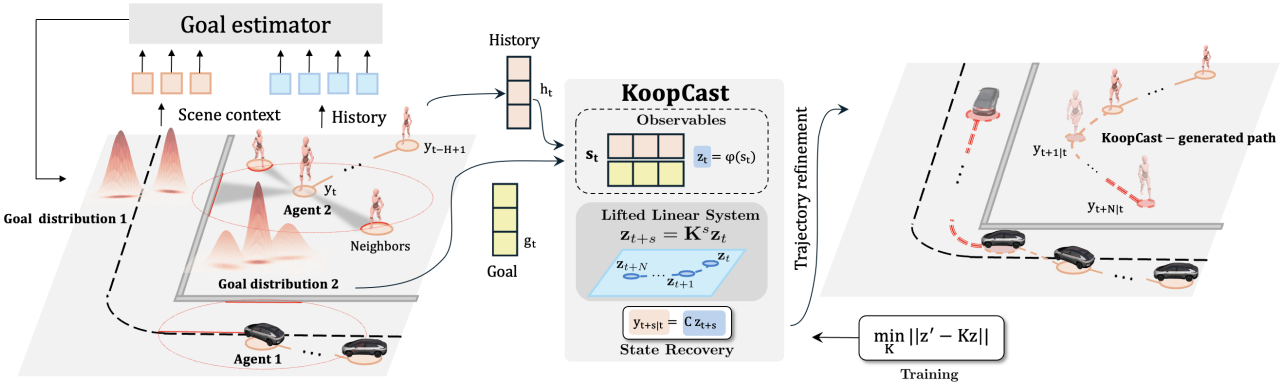


Fig. 1: Overview of KoopCast. The prediction process is divided into two phases: (1) estimating a temporal goal from history, map, and interactions; (2) refining the trajectory via Koopman operator theory, where nonlinear lifting functions enhance accuracy by enabling stable and interpretable linear evolution of motion.

The contributions of this paper are summarized as follows:

- We propose *KoopCast*, a trajectory forecasting model grounded in Koopman operator theory to address the nonlinear nature of agent dynamics. By leveraging its linear structure, KoopCast offers a straightforward learning procedure, further enhanced by our design of goal-conditioned observable functions that improve learning efficiency. As a result, KoopCast achieves competitive prediction accuracy across ETH/UCY [13], [14], the Waymo Open Motion Dataset (WOMD) [15], and nuScenes [16].
- We conduct a spectral analysis on the learned Koopman matrices and confirm that their spectral radii remain bounded by 1, ensuring marginal stability of the forecasts. Moreover, we identify specific Koopman modes near the unit circle ($|\lambda| \approx 1$) that dominate agent behavior, illustrating KoopCast's potential for interpreting complex motion dynamics.
- We design KoopCast with a shallow observable function and Koopman operator-based matrix formulation, substantially reducing computational overhead. Empirically, we demonstrate inference times up to 93.5% lower than recent deep learning-based methods [1], making KoopCast well suited for downstream navigation tasks.

The implementation is publicly available at `https://github.com/KoopCast/KoopCast`.

II. Related Work

Trajectory prediction. Early efforts in dynamical agent trajectory prediction (e.g. pedestrians and vehicles) employed physically grounded and rule-based models [17], [18]. However, these approaches suffered from limited local accuracy and an inability to capture latent features, as they relied heavily on predefined rules and prior domain knowledge. With the rise of deep learning and large-scale datasets, data-driven methods have become dominant, spanning LSTM-based sequence models [19], [20], convolutional and graph architectures [21], [1], generative models such as GANs [22], [2] and diffusion models [3], as well as Transformers [4], [23]. A particularly promising direction is the intention–refinement paradigm [11], [10], where high-level intentions are first localized and then refined into detailed motions. While achieving strong performance, modern deep learning–based approaches suffer from long inference times and limited interpretability, motivating alternative designs. Yet although these improve efficiency via lighter architectures and offer partial structural interpretability [24], [25], they rarely provide a principled analytic treatment grounded in dynamical-systems theory. KoopCast addresses these limitations by providing a lightweight, supervised alternative that combines high interpretability with practical efficiency, without sacrificing predictive accuracy.

Koopman operator theory. The Koopman operator [26], a long-standing framework for representing nonlinear dynamical systems in linear form, has gained renewed interest with the rise of data-driven methods that can construct effective approximate representations [27]. Thanks to its linear structure, the Koopman operator framework enables a straightforward extension of control-theoretic tools developed for linear systems and has been applied to diverse robotics problems, including system identification, prediction, and model-based control [28], [29]. Whereas prior efforts have primarily focused on modeling *macroscopic* crowd behavior using Koopman operators (e.g., [30]), we instead apply the framework at the level of individual agents, directly learning and predicting their inherently nonlinear movements. This shift allows Koopman operator-based methods to capture fine-grained pedestrian, vehicle, and cyclist dynamics that cannot be explained by macroscopic approximations alone.

III. Background: Koopman Operators

We consider a discrete-time nonlinear dynamical system on a state space $\mathcal{S} \subset \mathbb{R}^n$ with dynamics $\mathbf{s}_{k+1} = F(\mathbf{s}_k)$, where $\mathbf{s}_k \in \mathcal{S}$. Let $\Phi \subset \{\varphi : \mathcal{S} \to \mathbb{R}\}$ denote a vector space of *observables*, which can be chosen, for example, as the space of bounded continuous functions $C_b(\mathcal{S})$ or the space of square-integrable functions $L^2(\mathcal{S})$. The Koopman operator $\mathcal{K} : \Phi \to \Phi$ is defined through composition as $(\mathcal{K}\varphi)(\mathbf{s}) = \varphi\big(F(\mathbf{s})\big)$, $\varphi \in \Phi$. The operator $\mathcal{K}$ is linear on Φ, even when the underlying map F is nonlinear. Thus,

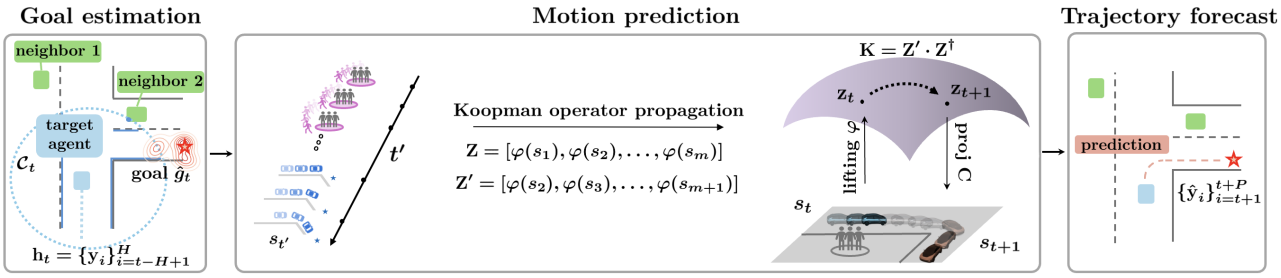


Fig. 2: Illustration of the KoopCast framework.

the Koopman operator provides a linear propagation of measurement functions of the system state along the flow of the dynamics.

Ideally, one would select observables that form an invariant subspace; however, since this is generally infeasible, practical approaches rely on effective observables with well-characterized approximation errors [31]. Common choices φ include polynomial basis functions [32], [33], approximate eigenfunctions [34], and data-driven lifting functions learned via neural networks [35], [36]. In this work, we adopt polynomial basis functions to construct the approximation.

The resulting finite-dimensional model takes the following form. Let $U = \mathrm{span}\{\varphi_1, \ldots, \varphi_p\} \subset \Phi(\mathcal{S})$ denote an approximate $\mathcal{K}$-invariant subspace, which specifies the chosen dictionary of lifting functions. Each basis element $\varphi_j : \mathcal{S} \to \mathbb{R}$ is observable, and the collection $\boldsymbol{\varphi} = (\varphi_1, \ldots, \varphi_p)$ defines the lifting map. The lifted state

$$\mathbf{z}_k = \boldsymbol{\varphi}(\mathbf{s}_k) \in \mathbb{R}^p, \tag{1}$$

resides in the finite-dimensional feature space $\mathbb{R}^p$, where the Koopman operator admits a matrix representation. This yields the following finite-dimensional *lifted linear* model

$$\mathbf{z}_{k+1} \;=\; K\,\mathbf{z}_k, \tag{2}$$

where $K \in \mathbb{R}^{p\times p}$ approximates the action of $\mathcal{K}$ on U. The original state $\mathbf{s}_k$ can be recovered from the lifted coordinates via a map $C : \mathbb{R}^p \to \mathbb{R}^n$. Depending on the observables, the map may be as simple as coordinate extraction—as in the polynomial case, where the state itself is included—or as complex as a neural network decoder.

A practical data-driven construction of K is provided by extended dynamic mode decomposition (eDMD) [37]. Given snapshot pairs $\{(\mathbf{s}_k, \mathbf{s}_{k+1})\}_{k=1}^{m}$, we lift them to $\mathbf{z}_k = \boldsymbol{\varphi}(\mathbf{s}_k)$ and form

$$Z \;=\; \big[\,\mathbf{z}_1, \ldots, \mathbf{z}_m\,\big], \qquad Z' \;=\; \big[\,\mathbf{z}_2, \ldots, \mathbf{z}_{m+1}\,\big], \tag{3}$$

after which we solve the least-squares problem

$$K \;=\; Z'\,Z^{\dagger}, \tag{4}$$

where $Z^{\dagger}$ denotes the Moore–Penrose pseudoinverse.

At prediction time, a state $\mathbf{s}_k$ is first *lifted* to $\mathbf{z}_k = \boldsymbol{\varphi}(\mathbf{s}_k)$, then *propagated* linearly as $\mathbf{z}_{k+\ell} = K^{\ell}\mathbf{z}_k$, and finally *projected back* via $\hat{\mathbf{s}}_{k+\ell} = C\,\mathbf{z}_{k+\ell}$. This framework recasts nonlinear dynamics as linear evolution in a lifted space, enabling the use of linear-system tools such as control-oriented formulations, lightweight computation, and spectral analysis [9].

IV. Trajectory Forecasting via Koopman Operators

A. Problem formulation

An overview of the *KoopCast* framework is shown in Figure 2. We consider a point agent with physical position $y_\tau \in \mathbb{R}^d$ ($d = 2$ in our setting), sampled at discrete time τ. At time t, we observe an H-step history $h_t := (y_{t-H+1}, \ldots, y_t) \in (\mathbb{R}^d)^H$, together with the scene context $\mathcal{C}_t = \{\, q \in \mathbb{R}^2 \mid \|q - y_t\| \le r \,\}$. We then define the constrained context $c_t = \{\, q_{(j)} \in \mathcal{C}_t \mid j = 1, \ldots, \min(|\mathcal{C}_t|, N) \,\}$, where $q_{(j)}$ denotes the j-th nearest point to y_t. That is, we retain at most N nearest points to y_t in order to reduce computational overhead while preserving locality. For pedestrian-only scenarios, the context corresponds to the positions of other agents, whereas for vehicles, c_t additionally incorporates lane-boundary and centerline points. For compactness, we denote the observed input by $x_t := (h_t, c_t)$. The forecasting task is therefore to model the conditional distribution of the P-step future $y_{t+1:t+P} := (y_{t+1}, \ldots, y_{t+P})$ given x_t.

A central challenge in future prediction lies in the high degree of uncertainty, primarily due to the unknown intentions and latent characteristics of surrounding agents. We adopt an "intention/target-first" framework, in which a likely destination is first localized and then a feasible motion is refined around it—a paradigm followed by several recent high-performance models [11], [12], [10]. Accordingly, KoopCast employs a two-stage scheme: (i) estimating a distribution of temporal goals using map and agent context, and (ii) refining the trajectory conditioned on the selected target.

Note that KoopCast constitutes a general framework, both theoretically and practically, as it is largely independent of specific agent type. Once a goal is specified, the Koopman operator-based refinement leverages motion history—capturing velocity, acceleration, and related dynamics—thereby ensuring consistent prediction quality across pedestrians, vehicles, and cyclists alike.

We introduce a *goal* random variable at horizon P, $g_t \in \mathcal{G} \subseteq \mathbb{R}^d$, associated with observation time t, where $\mathcal{G}$ denotes the set of map-feasible endpoints. Using g_t as a latent

variable, the future distribution is factorized as

$$p(y_{t+1:t+P}|x_t) = \int_{\mathcal{G}} p(g_t|x_t)\, p(y_{t+1:t+P}|g_t, x_t)\, dg_t, \quad (5)$$

which naturally decomposes prediction into two complementary components. Although (5) introduces a latent goal variable, we adopt the modeling assumption that the latent goal corresponds to the future endpoint y_{t+P} during training. The *goal predictor* estimates a compact distribution $p(g_t \mid x_t)$ over map-consistent temporal targets from history and scene context, capturing multimodal intent at inference time. Conditioned on the predicted goal, the *trajectory refinement module* models $p(y_{t+1:t+P} \mid g_t, x_t)$ using Koopman operator theory.

B. Temporal goal estimator

Following the above modeling assumption, we define the temporal goal at horizon P as $g_t := y_{t+P} \in \mathbb{R}^d$. To capture the multimodal nature of future intentions shaped by map structure and interactions with other agents, we employ a Gaussian mixture model (GMM) as the goal predictor.

Given the input $x_t = (h_t, c_t)$, consisting of the agent's motion history and local scene context, a mixture density network (MDN) outputs the parameters of a Gaussian mixture with M components over g_t:

$$p(g_t \mid x_t) = \sum_{j=1}^{M} \pi_j(x_t) \mathcal{N}\big(g_t; \mu_j(x_t), \mathrm{diag}(\sigma_j^2(x_t))\big), \quad (6)$$

where $\pi_j(x_t)$ are mixture weights with $\pi_j(x_t) \geq 0$ and $\sum_{j=1}^{M} \pi_j(x_t) = 1$, and $\mu_j(x_t) \in \mathbb{R}^d$, $\sigma_j^2(x_t) \in \mathbb{R}_+^d$.

The network is trained end-to-end by minimizing the negative log-likelihood of the observed ground-truth goals, enabling it to capture diverse, map-consistent endpoints. During inference, we sample multiple goals from the learned distribution, each giving rise to a distinct predicted trajectory. This formulation allows the model to represent the stochastic nature of the prediction task. Additional architectural details of the goal predictor are provided in the Appendix.

C. Koopman operator-based trajectory refinement

We now refine motion behavior by generating a plausible trajectory conditioned on the estimated goal. Conventional trajectory refinement methods [10], [11] typically operate by progressively pruning trajectories through offset regression, iterative query updates, and candidate scoring using large-scale multilayer perceptron (MLP) or transformer-based architectures. Although these mechanisms can improve accuracy, they depend on dense candidate generation and multilayer attention modules, which incur heavy computational overhead and limit interpretability.

In contrast, KoopCast avoids candidate-intensive refinement by employing a lightweight Koopman operator formulation. By lifting motion history into a higher-dimensional observable space, future trajectories evolve linearly under Koopman dynamics, with the goal providing contextual guidance. This linear propagation enables efficient closed-form rollout without requiring dense proposal sets or multilayer attention, thereby substantially reducing computational cost. Beyond efficiency, the spectral structure of the Koopman operator offers a natural avenue for interpretability: eigenvalues characterize the persistence or decay of dynamic modes, while eigenfunctions capture direction-specific behaviors such as turning or acceleration. As a result, KoopCast combines fast inference with transparent dynamics analysis, making it well suited for downstream planning tasks where both latency and reliability are critical.

Since the dominant uncertainty in trajectory forecasting arises from high-level target ambiguity [38], [39], the refinement dynamics themselves can be reasonably modeled as *deterministic* when training the Koopman operator. Extending the framework to stochastic refinement remains an important direction for future work.

Building on this perspective, we reformulate trajectory prediction and demonstrate its effectiveness through empirical results. Recall $y_t \in \mathbb{R}^d$ (agent position) and $h_t = (y_{t-H+1}, \ldots, y_t)$. Our aim is to obtain a *deterministic* formulation of trajectory refinement. In this setting, we use (h_t, g_t) instead of (h_t, c_t), since the context c_t is already encoded in the goal g_t. Accordingly, we construct an *autonomous* augmented state by appending a temporal goal $g_t \in \mathbb{R}^d$: $\mathbf{s}_t = (h_t,\, g_t) \in \mathcal{S} \subset \mathbb{R}^{d(H+1)}$ and posit the nonlinear but closed evolution

$$\mathbf{s}_{t+1} = F(\mathbf{s}_t), \quad (7)$$

which shifts the history and updates the goal component. During training, we set $g_t := y_{t+P}$ and $g_{t+1} := y_{t+1+P}$ from data; during testing, we obtain $\hat{g}_t$ from a pretrained goal estimator and hold it fixed during rollout.

Let $\mathcal{F}(\mathcal{S})$ be a linear space of observables and $\boldsymbol{\varphi} = (\varphi_1, \ldots, \varphi_p) : \mathcal{S} \to \mathbb{R}^p$ the dictionary that defines the lifting. Ideally, the Koopman operator is defined by $(\mathcal{K}\boldsymbol{\varphi})(\mathbf{s}) = \boldsymbol{\varphi}(F(\mathbf{s}))$, which acts on functions in an infinite-dimensional space. To obtain a computable representation, however, we restrict our attention to the finite-dimensional subspace $U = \mathrm{span}\{\varphi_1, \ldots, \varphi_p\}$ and approximate $\mathcal{K}$ by its projection onto U. With this approximation, the lifted state $\mathbf{z}_t = \boldsymbol{\varphi}(\mathbf{s}_t) \in \mathbb{R}^p$ evolves linearly according to

$$\mathbf{z}_{t+1} = K\mathbf{z}_t. \quad (8)$$

Therefore, the nonlinear dynamics of (7) can be represented as a linear dynamical system (8) within the Koopman operator framework.

During training, we compute the Koopman matrix via eDMD using a row-stacked formulation equivalent to the column-stacked form in Sec. III. Snapshot pairs $\{(\mathbf{s}_t, \mathbf{s}_{t+1})\}_{t=1}^{m}$ are lifted via $\boldsymbol{\varphi}$ to $\mathbf{z}_t, \mathbf{z}_{t+1}$ and stacked as

$$Z = \begin{bmatrix} \mathbf{z}_1^\top \\ \vdots \\ \mathbf{z}_m^\top \end{bmatrix}, \qquad Z' = \begin{bmatrix} \mathbf{z}_2^\top \\ \vdots \\ \mathbf{z}_{m+1}^\top \end{bmatrix}.$$

We then solve the ridge-regularized least-squares problem

$$K = \arg\min_{W} \|ZW - Z'\|_F^2 + \gamma\|W\|_F^2 \qquad (9)$$
$$= (Z^\top Z + \gamma I)^{-1} Z^\top Z'.$$

The ridge-regularized estimator in (9), with regularization parameter $\gamma > 0$, ensures numerical stability. In our experiments, the learned Koopman operators exhibit bounded spectral radii (see Sec. V-D), and no ill-conditioning or instability is observed in practice.

The KoopCast trajectory prediction process at test time is illustrated in Figure 2: (i) The temporal goal vector $\hat{g}_t$ is computed from the history (h_t) and surrounding context ($\mathcal{C}_t$) by the goal predictor; (ii) the estimated goal $\hat{g}_t$ is incorporated into the lifted state, which is concatenated with the trajectory history h_t to form the full observable $\mathbf{z}_t = \boldsymbol{\varphi}(\mathbf{s}_t)$; (iii) the Koopman operator is applied to propagate the lifted state linearly, $\mathbf{z}_{t+1} = K\mathbf{z}_t$; (iv) the predicted physical state $\hat{y}_{t+1|t}$ is extracted from the lifted state $\mathbf{z}_{t+1}$; (v) this process is repeated autoregressively for P steps using the same Koopman operator at each step:

$$\mathbf{z}_{t+\ell} = K^\ell \mathbf{z}_t, \quad \hat{y}_{t+\ell\,|\,t} = C\,\mathbf{z}_{t+\ell}, \quad 1 \le \ell \le P.$$

To capture nonlinearity, we employ a second-order polynomial observable dictionary for $\boldsymbol{\varphi}$. This choice reflects a trade-off between expressive power and computational efficiency: first-order lifting underfits nonlinear motion (e.g., turning and acceleration), whereas higher-order polynomials substantially increase the lifted dimension and computational cost with only marginal accuracy gains. The projection map $C \in \mathbb{R}^{d\times p}$ extracts the current position y_t from the lifted state. Since y_t corresponds to the last d entries of the history block in $\mathbf{z}_t$, we set $C = [\,0 \;\; I_d \;\; 0\,]$, yielding $\hat{y}_{t+\ell|t} = C\,\mathbf{z}_{t+\ell}$ for $\ell = 1, \ldots, P$.

V. Experiments

A. The setup

Datasets. To evaluate the efficiency of our approach, we conduct experiments on established trajectory forecasting benchmarks that include trajectories of pedestrians, vehicles, and cyclists. The benchmark datasets consist of the pedestrian-centric ETH/UCY datasets [13], [14], as well as two large-scale autonomous driving datasets: the Waymo Open Motion Dataset (WOMD) [15] and nuScenes [16].

Metrics. We follow the common practice of reporting *minADE* and *minFDE* [14], [19], i.e., best-of-K displacement errors, where in our case the stochasticity arises from goal prediction. For dataset-specific settings, we use 8 history and 12 prediction steps on ETH/UCY; a 2 sec history with a 6 sec prediction horizon at 2 Hz on nuScenes; and a 1 sec history with a 3 sec prediction horizon at 10 Hz on Waymo.

Baseline comparisons. For prediction accuracy, we construct the comparison table using results reported in studies with comparable settings [40], [41]. For inference time, we evaluate Trajectron++ [1] and EigenTrajectory [42] using the publicly available code released by the respective authors.

B. Implementation details

Observable function. To capture hidden linearity, KoopCast augments the observable dictionary with motion histories, their quadratic terms, and temporal goals, i.e., $\boldsymbol{\varphi}(\mathbf{s}_t) = [\,h_t,\; h_t^{\odot 2},\; g_t\,]^\top$. We define a projection map $C : \mathbb{R}^p \to \mathbb{R}^d$ that extracts y_t, i.e., the last d entries of h_t.

Normalization. For large-scale datasets (e.g., Waymo, nuScenes), we stabilize training by shifting the map to the origin and standardizing coordinates using the training-set mean and variance.

C. Main results: Prediction performance

Tables I, II, and III report a quantitative comparison of KoopCast against baseline methods. Our approach achieves the largest gains on the large-scale Waymo and nuScenes benchmarks, consistently outperforming several competitive baselines. On the ETH/UCY dataset, KoopCast also delivers competitive results, particularly in complex and dynamic scenes (Univ, Zara01, Zara02). These improvements stem from combining efficient closed-form rollout with interpretable spectral dynamics, enabling both fast inference and reliable performance on large-scale benchmarks. ETH and Univ involve abrupt, interaction-driven motions that are less compatible with our goal-conditioned assumption, yielding smaller gains. Future work will study unified Koopman models without explicit goal conditioning to improve robustness.

TABLE I: Performance on ETH/UCY dataset (ADE/FDE in meters). The top three results are highlighted in bold.

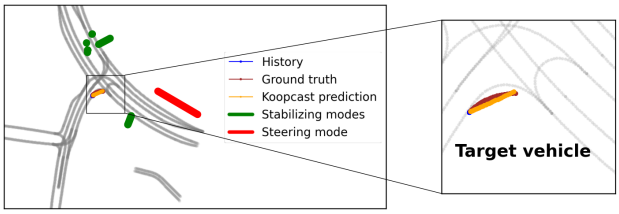
	ETH	Univ	Zara01	Zara02
Linear	1.07/2.28	0.52/1.16	0.42/0.95	0.32/0.72
SocialGAN [22]	0.64/1.09	0.56/1.18	0.33/0.67	0.31/0.64
SoPhie [43]	0.70/1.43	0.54/1.24	0.30/0.63	0.38/0.78
SocialWays [44]	**0.39/0.64**	0.55/1.31	0.44/0.64	0.51/0.92
STAR [23]	**0.36/0.65**	0.31/0.62	0.29/0.52	0.22/0.46
TransformerTF [45]	0.61/1.12	0.35/0.65	**0.22/0.38**	**0.17/0.32**
MANTRA [46]	0.48/0.88	0.37/0.81	**0.22/0.38**	**0.17/0.32**
MemoNet [47]	**0.40/0.61**	**0.24/0.43**	**0.18/0.32**	**0.14/0.24**
PECNet [48]	0.54/0.87	0.35/0.60	0.22/0.39	0.17/0.30
Trajectron++ [1]	0.54/0.94	**0.28/0.55**	0.21/0.42	0.16/0.32
AgentFormer [49]	**0.45/0.75**	**0.25/0.45**	**0.18/0.30**	**0.14/0.24**
BiTraP [50]	0.56/0.98	**0.25/0.47**	0.23/0.45	0.16/0.33
SGNet-ED [51]	0.47/0.77	0.33/0.62	**0.18/0.32**	**0.15/0.28**
KoopCast	0.66/1.22	0.35/0.72	**0.21/0.40**	**0.17/0.32**

TABLE II: Performance on Waymo Open Motion Dataset. The top two results are highlighted in bold.

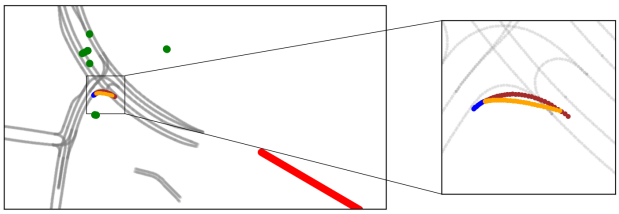
Waymo (Validation Set)	minADE_k (m)		minFDE_k (m)	
	$k=1$	$k=5$	$k=1$	$k=5$
Constant Velocity	2.04	-	5.25	-
Constant Lane	2.54	-	5.85	-
Kalman Filter	1.99	-	4.07	-
SAMPP [52]	1.26	-	2.80	-
IMAP [52]	0.97	-	2.03	-
Trajectron++	0.88	0.56	2.37	1.41
MotionCNN [53]	0.83	**0.40**	1.99	**0.81**
M2I [54]	0.67	0.42	1.60	0.85
ContextVAE [40]	**0.59**	**0.30**	**1.49**	**0.68**
KoopCast	**0.60**	**0.40**	**1.54**	0.98

TABLE III: Performance on nuScenes Dataset. The top two results are highlighted in bold.

nuScenes (Validation Set)	minADE_k (m)		minFDE_k (m)	
	$k=1$	$k=5$	$k=1$	$k=5$
Constant Velocity	4.61	-	11.21	-
Constant Lane	5.45	-	12.73	-
Kalman Filter	4.17	-	10.99	-
Trajectron++	4.08	2.41	9.67	5.63
P2T [55]	3.82	1.86	8.95	4.08
AutoBots-Ego [56]	3.86	**1.70**	8.89	3.40
ContextVAE	**3.54**	**1.59**	**8.24**	**3.28**
KoopCast	**2.50**	1.73	**4.74**	**3.31**



(a) Straight motion (t = 0.1s)



(b) Steering motion (t = 1.7s)

Fig. 3: Koopman modal decomposition of a predicted vehicle trajectory in a real road scene. (a), (b) Eigendecomposition of motion modes (left: full decomposition; right: zoom near prediction). The predicted trajectory (orange) is reconstructed by summing the green and red modes. (a) corresponds to straight motion before a right turn, while (b) shows turn initiation. During turning, the steering-dominant mode (red) aligns with the curvature and exhibits increased magnitude, indicating its dominant role in motion propagation.

D. Key properties of KoopCast

1) Stability guarantee: A key requirement for trajectory refinement is stability: without it, even plausible dynamics may yield exploding or oscillatory trajectories, which is unacceptable in safety-critical forecasting. Most existing approaches address this challenge by designing elaborate refinement modules, such as dense candidate selection [10] or query-based iterative adjustments [11]. In contrast, KoopCast inherently ensures stability through the spectral properties of the Koopman operator. The lifted state $\mathbf{z}_t$, which contains the physical state $\mathbf{s}_t$ together with history and goal features, remains bounded for arbitrarily long horizons, thereby ensuring stable trajectory refinement. Specifically, the propagation $\mathbf{z}_{t+\ell} = K^\ell \mathbf{z}_t$, $\ell = 1, \dots, P$, remains stable when the spectral radius is no greater than 1, i.e., $\rho(K) = \max_i |\lambda_i(K)| \leq 1$, where $\{\lambda_i(K)\}_{i=1}^p$ are the eigenvalues of K.

Because the Koopman matrix K is estimated via ridge regression (9), its spectrum can be explicitly controlled. Figure 4 illustrates that, in practice, the eigenvalues cluster near or within the unit circle; moreover, increasing the ridge coefficient further contracts the spectrum, ensuring $|\lambda_i(K)| < 1$ and thereby guaranteeing long-horizon stability.

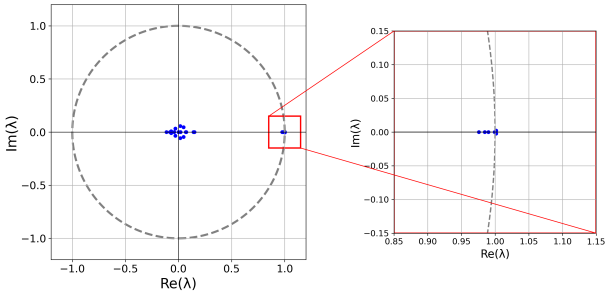


Fig. 4: The spectrum of K scattered on the complex plane.

2) Interpretability: Given the eigendecomposition $K = V\Lambda V^{-1}$ with eigenvalues $\{\lambda_i\}_{i=1}^p$ and right eigenvectors v_i, the ℓ-step Koopman propagation is $\hat{y}_{t+\ell|t} = CK^\ell \mathbf{z}_t = C\sum_{i=1}^p v_i \lambda_i^\ell (w_i^\top \mathbf{z}_t)$, where $w_i^\top$ is the i-th row of V^{-1}, $\mathbf{z}_t$ is the lifted state (including history and goal), and C projects the observable back to physical coordinates. Each term $C(v_i \lambda_i^\ell (w_i^\top \mathbf{z}_t))$ corresponds to the isolated trajectory contribution of a single mode. In Figure 3 the green and red mode curves represent the sequences $\{C(v_i \lambda_i^\ell (w_i^\top \mathbf{z}_t))\}_{\ell=1}^P$, while the full Koopman rollout (yellow) is obtained as their direct sum. The modes can be broadly divided into two categories: those with $|\lambda_i| \geq 0.8$, which exert a dominant influence, and those with $|\lambda_i| \leq 0.3$, which quickly decay. Among the persistent modes, steering maneuvers consistently activate a distinct *steering-dominant* mode (red), whose direction aligns with the turn curvature. Figure 3 illustrates this phenomenon: while all modal contributions (green/red) superimpose to reconstruct the trajectory (orange), the steering-related mode (red) is strongly amplified during turning.

Beyond interpretability, the modal decomposition also provides practical benefits. By identifying typical mode combinations, anomalies can be flagged when unusual activations occur (e.g., sudden U-turns). Moreover, detecting strong steering-related modes enables autonomous systems to anticipate turns earlier and adjust planning in advance, improving both prediction accuracy and safety. A more detailed investigation is left for future work.

TABLE IV: Average inference time per sample (ETH/UCY datasets). Values are averaged across all scenarios.

	Linear	**KoopCast**	**EigenTrajectory**	**Trajectron++**
Time [ms/sample]	0.0019	0.116	0.665	1.778

3) Training and inference time: We compare inference times on ETH/UCY in Table IV. The Linear baseline is the fastest owing to direct extrapolation. Excluding this trivial case, KoopCast achieves substantially lower latency than deep models such as EigenTrajectory and Trajectron++, with inference approximately 93.5% faster than the latter. This efficiency arises from its shallow goal estimator and linear

eDMD refinement, which avoid costly recurrent or attention mechanisms, making KoopCast well suited for real-time navigation and control.

VI. Conclusions

We have presented a motion prediction framework for dynamic agents grounded in Koopman operator theory, which lifts nonlinear dynamics into a higher-dimensional linear representation. Our predictor demonstrates strong performance across diverse environments with heterogeneous agents (e.g., pedestrians, vehicles, cyclists), while ensuring stable trajectory evolution through both theoretical guarantees and empirical validation based on Koopman spectral analysis. Beyond accuracy, the Koopman-based formulation provides interpretability for analyzing motion dynamics, together with efficient training and low-latency inference.

Appendix

In this appendix, we provide implementation details of the goal estimator.

A. Architecture of the goal estimator

At time t, we observe the input $x_t = (h_t, c_t)$ consisting of the H-step history $h_t = (y_{t-H+1}, \dots, y_t)$ and the local scene context c_t. Before predicting the temporal goal g_t, we normalize x_t by projecting geometric data into the ego frame of the target agent. Given the pose $(p_t, R_t) \in \mathrm{SE}(2)$, each q_t (representing other agents or lane points) is transformed as $\bar{q}_t := R_t^{-1}(q_t - p_t)$, which maps the agent to $(0, 0)$ with its heading aligned to the x-axis.

Our goal predictor outputs a distribution over 2D goals $\mathcal{G}$ given x_t, implemented as a 2-layer MLP. The final layer produces $\{\pi_j, \mu_j, \sigma_j\}_{j=1}^{M}$, parameterizing the GMM: $p_\theta(g|x) = \sum_{j=1}^{M} \pi_j(x;\theta)\mathcal{N}(g|\mu_j(x;\theta), \mathrm{diag}(\sigma_j^2(x;\theta)))$, thereby capturing the multi-modality of goals. The input x_t consists of the ego-aligned trajectory history together with local scene context. In particular, $x_t = (h_t, c_t)$, where c_t contains the N nearest points within radius r from y_t, representing nearby agents or lane geometry depending on the scene.

B. Training and inference

Both predictors are trained by minimizing the negative log-likelihood. Given data $\{x^{(k)}, g^{(k)}\}_{k=1}^{N}$, the objective is $\mathcal{L}_{\mathrm{NLL}}(\theta) = -\frac{1}{N}\sum_{k=1}^{N} \log p_\theta(g^{(k)}|x^{(k)})$, which is optimized using Adam [57]. During inference, the predictor outputs $p_\theta(g|x)$, from which we draw K samples $\{g^{(k)}\}_{k=1}^{K} \sim p_\theta(\cdot|x)$ to capture multi-modality. These candidates are used for best-of-K comparison (Sec. V-C). For navigation, one may instead use $\bar{g} = \mathbb{E}_{p_\theta}[g|x]$.

C. Implementation details

Table V lists the hyperparameters used to train the goal predictor of KoopCast across all datasets. For ETH/UCY, we use an 80/10/10% train/validation/test split, while for WOMD and nuScenes we use 90% for training and the official validation sets. The MDN output dimension equals $M \times (2d + 1)$, corresponding to mixture weights, means, and diagonal variances. The implementation is based on PyTorch [58] and experiments are run on an Intel i9-13900K CPU with an NVIDIA RTX 4090 GPU.

TABLE V: Goal estimator hyperparameters by dataset.

Hyperparameter	Waymo	nuScenes	ETH/UCY
Optimizer	Adam [57]	Adam	Adam
Learning rate	1e−3	1e−3	1e−3
Input dimension	98	98	86
MLP layers	2	2	2
Hidden units	128	128	128
Activation	ReLU	ReLU	ReLU
Mixtures (M)	5	5	6
Goal dim (d)	2	2	2
Batch size	128	128	1
Epochs	100	100	30
Ridge parameter (γ)	1	1	1e−3

Acknowledgment

The authors would like to thank Prof. Ram Vasudevan at the University of Michigan for his insightful discussion. ChatGPT (OpenAI) was used for language editing and assistance with code design; all content was reviewed and verified by the authors.

References

[1] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 683–700.

[2] Y. Hu, S. Chen, Y. Zhang, and X. Gu, "Collaborative motion prediction via neural motion message passing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6319–6328.

[3] W. Mao, C. Xu, Q. Zhu, S. Chen, and Y. Wang, "Leapfrog diffusion model for stochastic trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 5517–5526.

[4] P. S. Chib and P. Singh, "CCF: Cross correcting framework for pedestrian trajectory prediction," *arXiv preprint arXiv:2406.00749*, 2024.

[5] D. Bruder, X. Fu, R. B. Gillespie, C. D. Remy, and R. Vasudevan, "Data-driven control of soft robots using Koopman operator theory," *IEEE Trans. Robot.*, vol. 37, no. 3, pp. 948–961, 2020.

[6] G. Mamakoukas, M. L. Castano, X. Tan, and T. D. Murphey, "Derivative-based Koopman operators for real-time control of robotic systems," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 2173–2192, 2021.

[7] D. A. Haggerty, M. J. Banks, P. C. Curtis, I. Mezić, and E. W. Hawkes, "Modeling, reduction, and control of a helically actuated inertial soft robotic arm via the Koopman operator," *arXiv preprint arXiv:2011.07939*, 2020.

[8] W. A. Manzoor, S. Rawashdeh, and A. Mohammadi, "Vehicular applications of Koopman operator theory—a survey," *IEEE Access*, vol. 11, pp. 25 917–25 931, 2023.

[9] I. Mezić, "Spectrum of the Koopman operator, spectral expansions in functional spaces, and state-space geometry," *J. Nonlinear Sci.*, vol. 30, no. 5, pp. 2091–2145, 2020.

[10] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid *et al.*, "TNT: Target-driveN Trajectory prediction," in *Proc. Conf. Robot Learn.*, 2021, pp. 895–904.

[11] S. Shi, L. Jiang, D. Dai, and B. Schiele, "MTR-A: 1st place solution for 2022 waymo open dataset challenge–motion prediction," *arXiv preprint arXiv:2209.10033*, 2022.

[12] S. Shi, L. Jiang, D. Dai, and B. Schiele, "MTR++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 5, pp. 3955–3971, 2024.

[13] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in *Comput. Graph. Forum*, vol. 26, no. 3, 2007, pp. 655–664.

[14] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 261–268.

[15] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou *et al.*, "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 9710–9719.

[16] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11 621–11 631.

[17] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Phys. Rev. E*, vol. 51, no. 5, p. 4282, 1995.

[18] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH J.*, vol. 1, no. 1, p. 1, 2014.

[19] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 961–971.

[20] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 1468–1476.

[21] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-STGCNN: A social spatio-temporal graph convolutional neural network for human trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14 424–14 432.

[22] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2255–2264.

[23] C. Yu, X. Ma, J. Ren, H. Zhao, and S. Yi, "Spatio-temporal graph transformer networks for pedestrian trajectory prediction," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 507–523.

[24] S. Huang, L. Ye, M. Chen, W. Luo, D. Wang, C. Xu, and D. Liang, "Interpretable interaction modeling for trajectory prediction via agent selection and physical coefficient," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2025, pp. 15 422–15 429.

[25] F. G. Landry and M. A. Akhloufi, "TrajFusionNet: Pedestrian crossing intention prediction via fusion of sequential and visual trajectory representations," *arXiv preprint arXiv:2508.19866*, 2025.

[26] B. O. Koopman and J. v. Neumann, "Dynamical systems of continuous spectra," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 18, no. 3, pp. 255–263, 1932.

[27] I. Mezić and A. Banaszuk, "Comparison of systems with complex behavior," *Physica D*, vol. 197, no. 1-2, pp. 101–133, 2004.

[28] L. Shi, M. Haseli, G. Mamakoukas, D. Bruder, I. Abraham, T. Murphey, J. Cortés, and K. Karydis, "Koopman operators in robot learning," *IEEE Trans. Robot.*, 2026.

[29] V. Zinage and E. Bakolas, "Neural Koopman Lyapunov control," *Neurocomputing*, vol. 527, pp. 174–183, 2023.

[30] D. Lehmberg, F. Dietrich, and G. Köster, "Modeling Melburnians—using the Koopman operator to gain insight into crowd dynamics," *Transp. Res. Part C Emerg. Technol.*, vol. 133, p. 103437, 2021.

[31] M. Korda and I. Mezić, "On convergence of extended dynamic mode decomposition to the Koopman operator," *J. Nonlinear Sci.*, vol. 28, no. 2, pp. 687–710, 2018.

[32] M. Korda and I. Mezić, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149–160, 2018.

[33] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz, "Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control," *PLoS One*, vol. 11, no. 2, p. e0150171, 2016.

[34] M. Korda and I. Mezić, "Optimal construction of Koopman eigenfunctions for prediction and control," *IEEE Trans. Autom. Control*, vol. 65, no. 12, pp. 5114–5129, 2020.

[35] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature Communications*, vol. 9, no. 1, p. 4950, 2018.

[36] Q. Li, F. Dietrich, E. M. Bollt, and I. G. Kevrekidis, "Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator," *Chaos*, vol. 27, no. 10, 2017.

[37] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition," *J. Nonlinear Sci.*, vol. 25, pp. 1307–1346, 2015.

[38] J. Wu, J. Ruenz, H. Berkemeyer, L. Dixon, and M. Althoff, "Goal-oriented pedestrian motion prediction," *IEEE Trans. Intel. Trans. Sys.*, vol. 25, no. 6, pp. 5282–5298, 2023.

[39] Y. Liu, Z. Ye, R. Wang, B. Li, Q. Z. Sheng, and L. Yao, "Uncertainty-aware pedestrian trajectory prediction via distributional diffusion," *Knowledge-Based Syst.*, vol. 296, p. 111862, 2024.

[40] P. Xu, J.-B. Hayet, and I. Karamouzas, "Context-aware timewise VAEs for real-time vehicle trajectory prediction," *IEEE Robot. Autom. Letters*, vol. 8, no. 9, pp. 5440–5447, 2023.

[41] P. Xu, J.-B. Hayet, and I. Karamouzas, "SocialVAE: Human trajectory prediction using timewise latents," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 511–528.

[42] I. Bae, J. Oh, and H.-G. Jeon, "EigenTrajectory: Low-rank descriptors for multi-modal trajectory forecasting," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 10 017–10 029.

[43] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "SoPHie: An attentive gan for predicting paths compliant to social and physical constraints," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1349–1358.

[44] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with gans," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019, pp. 2964–2972.

[45] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso, "Transformer networks for trajectory forecasting," in *Proc. Int. Conf. Pattern Recognit.* IEEE, 2021, pp. 10 335–10 342.

[46] F. Marchetti, F. Becattini, L. Seidenari, and A. D. Bimbo, "MANTRA: Memory augmented networks for multiple trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7143–7152.

[47] C. Xu, W. Mao, W. Zhang, and S. Chen, "Remember intentions: Retrospective-memory-based trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6488–6497.

[48] K. Mangalam, H. Girase, S. Agarwal, K.-H. Lee, E. Adeli, J. Malik, and A. Gaidon, "It is not the journey but the destination: Endpoint conditioned trajectory prediction," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 759–776.

[49] Y. Yuan, X. Weng, Y. Ou, and K. M. Kitani, "AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 9813–9823.

[50] Y. Yao, E. Atkins, M. Johnson-Roberson, R. Vasudevan, and X. Du, "BitraP: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1463–1470, 2021.

[51] C. Wang, Y. Wang, M. Xu, and D. J. Crandall, "Stepwise goal-driven networks for trajectory prediction," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2716–2723, 2022.

[52] J. L. V. Espinoza, A. Liniger, W. Schwarting, D. Rus, and L. Van Gool, "Deep interactive motion prediction and planning: Playing games with motion prediction models," in *Proc. Learn. Dyn. Control Conf.*, 2022, pp. 1006–1019.

[53] S. Konev, K. Brodt, and A. Sanakoyeu, "MotionCNN: A strong baseline for motion prediction in autonomous driving," *arXiv preprint arXiv:2206.02163*, 2022.

[54] Q. Sun, X. Huang, J. Gu, B. C. Williams, and H. Zhao, "M2I: From factored marginal trajectory prediction to interactive prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit*, 2022, pp. 6543–6552.

[55] N. Deo and M. M. Trivedi, "Trajectory forecasts in unknown environments conditioned on grid-based plans," *arXiv preprint arXiv:2001.00735*, 2020.

[56] R. Girgis, F. Golemo, F. Codevilla, M. Weiss, J. A. D'Souza, S. E. Kahou, F. Heide, and C. Pal, "Latent variable sequential set transformers for joint multi-agent motion prediction," *arXiv preprint arXiv:2104.00563*, 2021.

[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." in *Proc. Int. Conf. Learn. Represent.*, 2015.

[58] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.