Kernel Trajectory Maps for Multi-Modal Probabilistic Motion Prediction

Weiming Zhi [1] **and Lionel Ott** [1] **and Fabio Ramos** [1,2]
[1] Department of Computer Science, University of Sydney, Australia
[2] NVIDIA, USA
{firstname.lastname}@sydney.edu.au

Abstract: Understanding the dynamics of an environment, such as the movement of humans and vehicles, is crucial for agents to achieve long-term autonomy in urban environments. This requires the development of methods to capture the multi-modal and probabilistic nature of motion patterns. We present **k**ernel **t**rajectory **m**aps (KTM) to capture the trajectories of movement in an environment. KTMs leverage the expressiveness of kernels from non-parametric modelling by projecting input trajectories onto a set of representative trajectories, to condition on a sequence of observed waypoint coordinates, and predict a multi-modal distribution over possible future trajectories. The output is a mixture of continuous stochastic processes, where each realisation is a continuous functional trajectory, which can be queried at arbitrarily fine time steps.

Keywords: Trajectory Learning, Motion prediction, Kernel methods

1 Introduction

Autonomous agents may be required to operate in environments with moving objects, such as pedestrians and vehicles in urban areas, for extended periods of time. A probabilistic model that captures the movement of surrounding dynamic objects allows an agent to make more effective and robust plans. This work presents **k**ernel **t**rajectory **m**aps (KTM) [1], that capture the multi-modal, probabilistic, and continuous nature of future paths. Given a sequence of observed waypoints of a trajectory up to a given coordinate, a KTM is able to produce a multi-modal distribution over possible future trajectories, represented by a mixture of stochastic processes. Continuous functional trajectories, which are functions mapping queried times to trajectory coordinates, can then be sampled from the output stochastic process.

Early methods to predict future motion trajectories generally extrapolate based on physical laws of motion [1]. Although simple and often effective, these models have the drawback of being unable to make use of other observed trajectories, or account for environment topology. For example, physics-based methods fail to take into account that trajectories may follow a road that exists in a map. To address this shortcoming, methods have been developed that map the direction or flow of movements in an environment in a probabilistic manner [2, 3, 4, 5]. These methods are able to output distributions over future movement directions or velocities, conditioned on the current queried coordinate. Using these models, one can sequentially forward sample to obtain a trajectory. This forward sampling approach makes the Markov assumption, assuming that the object dynamics only depend on the current position of the object. These approaches discard useful information from the trajectory history, and can accumulate errors from the forward simulation.

Motivated to overcome these aforementioned limitations of past methods, we utilise *distance substitution kernels* [6, 7] with the Fréchet distance [8, 9, 10] to project trajectory data onto a representative set of trajectories to obtain high-dimensional projection features. Using a neural network with a single hidden layer with the projection features, we learn a multi-modal mixture of stochastic processes. The resulting mixture is also a stochastic process, and can be viewed as a distribution over functions, where each realisation is a continuous functional trajectory. Figure 1 shows observed trajectories and realised trajectory predictions, demonstrating the probabilistic and multi-modal na-

[1]Code available at https://github.com/wzhi/KernelTrajectoryMaps

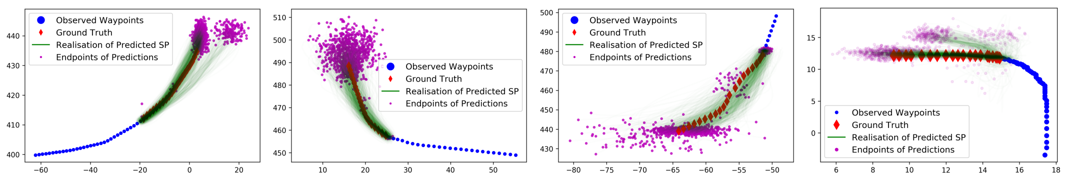

Figure 1: Observed waypoints (blue) and predicted trajectories (green with magenta end-points) sampled from KTM outputs. The ground truth is indicated in red. The probabilistic and multi-modal nature of KTMs is able to capture the complexity of the motion patterns. Units in meters.

ture of KTMs. The probabilistic nature of the output provides an estimate for uncertainty, which can be used for robust planning and decision making. We contribute the KTM, a method that:

1. is trajectory history aware and captures dependencies over the entire trajectory;
2. models the output as a mixture of stochastic process, providing a multi-modal distribution over possible trajectories;
3. represents realised trajectories as continuous functions, allowing them to be queried at arbitrary time resolution.

2 Related Work

Kernel Trajectory Maps (KTMs) learn motion patterns in an environment, and represent sampled outputs as continuous trajectories. i.e. trajectories that can be queried at arbitrarily fine time resolutions. Here we briefly revisit literature on modelling motion dynamics and continuous trajectories.

Motion Modelling. Some of the simplest approaches to model trajectory patterns are kinematic models that make extrapolations based on a sequence of observed coordinates. Popular examples include the constant velocity and constant acceleration models [11]. Some other attempts to understand dynamics take the approach of extending occupancy mapping beyond static environments by building occupancy representations along time [12, 13, 14]. This approach tends to be memory intensive, limiting scalability. Other recent approaches have incorporated global spatial [2, 3, 4, 5] and temporal information [2, 15, 16]. The authors of [3] propose *directional grid maps*, a model that learns the distribution of motion directions in each grid cell of a discretised environment. This is achieved by fitting a mixture of von-Mises distributions on the motion directions attributed to each cell. A similar method is also presented in [4], where a map of velocity distributions in the environment is modelled by semi-wrapped Gaussian mixture models. Continuous spatiotemporal extensions are provided in [2]. These methods are able to capture the uncertainty of motion at a given point coordinate, but require forward sampling to obtain trajectories.

Continuous Trajectories. Continuous representations of trajectories, often modelled by a Gaussian processes [17] or a sparse low rank approximations of Gaussian processes [18], have arisen in previous works for trajectory estimation [19] and motion planning [20, 21]. In this work, we also formulate a method to produce continuous trajectories, and then leverage continuous trajectories for extrapolation, rather than the estimation and interpolation problems addressed in previous works.

3 Methodology

3.1 Problem Formulation and Overview

We work with continuous trajectory outputs, Ξ, and discrete trajectories inputs, $\boldsymbol{\xi}$. Discrete trajectories are an ordered set of waypoint coordinates indexed by time, $\boldsymbol{\xi} = \{(x_t, y_t)\}_{t=1}^{T}$. Continuous trajectories, $\Xi(\cdot)$, are functions that map time to coordinates. Continuous trajectories can be discretised by querying at time steps, $\boldsymbol{t} = 1, \ldots, T$. In this paper, continuous trajectories, $\Xi(\cdot)$, are defined by weighted combinations of features, $\boldsymbol{\phi}(\cdot)$, where $\boldsymbol{w}$ contains the weight parameters. $\boldsymbol{\phi}(\cdot)$ is dependent on the queried time. We discuss continuous trajectories in detail in subsection 3.3.

Given a dataset of N pairs of trajectories, $\mathcal{D} = \{\boldsymbol{\xi}_n^{\text{Obs}}, \boldsymbol{\xi}_n^{\text{Tar}}\}_{n=1}^{N}$, where $\boldsymbol{\xi}^{\text{Obs}}$ is an observed input trajectory, and $\boldsymbol{\xi}^{\text{Tar}}$ is a target trajectory. The input contains coordinates up to a given time, and the target is a continuation of the same trajectory thereafter. We seek to predict a probability distribution over possible future trajectories beyond the given observed waypoints, $p\big(\Xi^*(\cdot)|\boldsymbol{\xi}^*, \mathcal{D}, \boldsymbol{\phi}(\cdot)\big)$, where $\boldsymbol{\xi}^*$ is a queried discrete trajectory, $\Xi^*(\cdot)$ is a predicted continuous trajectory starting from the last time step of $\boldsymbol{\xi}^*$. To find the distribution over future trajectories, we write the marginal likelihood as,

$$p(\Xi^*(\cdot)|\boldsymbol{\xi}^*, \mathcal{D}, \boldsymbol{\phi}) = \int p(\Xi^*(\cdot)|\boldsymbol{\phi}, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D})d\boldsymbol{w}. \tag{1}$$

To evaluate the marginal likelihood, we learn $p\big(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D}\big)$ and sample realisations of weights to conduct inference (detailed in subsection 3.5). This learning can be summarised by the following steps:

1. Construct high-dimensional feature vectors of observed discrete trajectories, by projecting to a set of representative trajectories, using discrete Fréchet [9] kernels (DF-Kernels). (Subsection 3.2)
2. Concisely represent each trajectory as a continuous function, defined by a vector of weights and predetermined basis functions. (Subsection 3.3)
3. Train a single hidden layer mixture density network (MDN) model on the projection features, with weight vectors as targets, to obtain $p\big(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D}\big)$. (Subsection 3.4)

3.2 Generating Projection Features from Discrete Trajectories

In this subsection, we describe the conversion from discrete input trajectories to high-dimensional kernel projections. We make use of distance substitute kernels [22, 6, 7], which are defined as $k(x, x') = k(d(x, x'))$, for kernel function, $k(\cdot)$, and distance measure d that is symmetric, i.e. $d(x, x') = d(x', x)$, and has zero diagonal, i.e. $d(x, x) = 0$. In this work, we use the discrete Fréchet distance [9] substituted in a radial basis function (RBF) kernel. The Fréchet distance [8] between curves is defined as,

$$Fr(P, Q) = \inf_{\alpha,\beta} \max_{t\in[0,1]} ||P(\alpha(t)) - Q(\beta(t))||, \tag{2}$$

where P, Q are parameterisations of two curves, and α, β range over all continuous and monotone increasing functions. We use a discrete approximation of the Fréchet distance, which provides a distance metric between ordered sets of arbitrary length. The discrete Fréchet distance between two trajectories can be computed efficiently in $O(pq)$, where there are p and q waypoints in each of the trajectories. The discrete Fréchet distance takes into consideration the ordering of waypoints, and can in general distinguish a given trajectory with its reverse. An algorithm to compute the discrete Fréchet distance is outlined in [9]. We name this kernel the *discrete Fréchet (DF) kernel*, given by:

$$k_{DF}(\boldsymbol{\xi}, \boldsymbol{\xi'}) = \exp\bigg\{ -\frac{\big(d_{DF}(\boldsymbol{\xi}, \boldsymbol{\xi'})\big)^2}{2\ell_{DF}} \bigg\}, \tag{3}$$

where $\boldsymbol{\xi}$ and $\boldsymbol{\xi'}$ are discrete trajectories, which can be of different lengths; ℓ_{DF} is the length scale parameter of the RBF kernel; d_{DF} is the discrete Fréchet distance.

We project each observed trajectory with DF-kernel onto a set of representative trajectories. We obtain $\boldsymbol{\varphi}_n \in \mathcal{R}^{M_{\boldsymbol{\xi}}}$, a vector of projections from $\boldsymbol{\xi}_n$ onto $\{\hat{\boldsymbol{\xi}_1}, \ldots, \hat{\boldsymbol{\xi}_{M_\xi}}\}$. A set of M_ξ trajectories, $\{\hat{\boldsymbol{\xi}_1}, \ldots, \hat{\boldsymbol{\xi}_{M_\xi}}\}$, are selected from the set of all observed input trajectories. We refer to the selected trajectories as *representative trajectories*. An alternative view of this process is placing basis functions over representative trajectories. The corresponding high-dimensional features over all N observations are given by,

$$K_{N\times M_\xi} = \begin{bmatrix} \boldsymbol{\varphi}_1^T \\ \vdots \\ \boldsymbol{\varphi}_N^T \end{bmatrix} = \begin{bmatrix} k_{DF}(\boldsymbol{\xi}_1, \hat{\boldsymbol{\xi}_1}) & \ldots & k_{DF}(\boldsymbol{\xi}_1, \hat{\boldsymbol{\xi}_{M_\xi}}) \\ \vdots & \ddots & \vdots \\ k_{DF}(\boldsymbol{\xi}_N, \hat{\boldsymbol{\xi}_1}), & \ldots, & k_{DF}(\boldsymbol{\xi}_N, \hat{\boldsymbol{\xi}_{M_\xi}}). \end{bmatrix} \tag{4}$$

We later input the projection features to a simple neural network model, and do not operate directly on the Gram matrix. This can be viewed as learning combinations of fixed basis functions, similar to sparse Gaussian process (GP) regression [18]. Selecting good representative trajectories can be done in a manner similar to selecting inducing points for the Nyström Method [23, 24, 25, 26] in sparse GPs. Even though randomly selecting a subset of trajectories from the observed trajectories is sufficient, we outline a quick and simple sampling scheme, similar to the leverage score sampling method [24]. Provided a square matrix of the discrete Fréchet distances between all trajectories in a dataset of observation, $D_{N\times N}$, sort the columns of the matrix by its L_2 norm, and select every i^{th} column, with i being a fixed stepsize. The corresponding trajectory of each column selected is added to the representative set. The intuition is that almost identical trajectories would likely be sorted adjacent to one another. Hence, our heuristic discourages selecting multiple almost identical representative trajectories, and encourages selecting a more diverse set of representations.

Though not explored deeply in this work, projecting input trajectories to a fixed set of representative trajectories may also be exploited to efficiently condition on high-dimensional trajectories. Challenges can arise from the "vastness" of space trajectories can lie in. If trajectories in high-dimensional space belong to only a few groupings, and in practice only occupy a limited volume in high-dimensional space, inputs may be adequately represented by a not-too-large representative set.

The projected feature vectors generated are representations of our discrete input observations, whereas continuous output trajectories sampled from KTMs are in concise functional forms. Details for constructing functional trajectories are described in the next subsection.

3.3 Constructing Continuous Functional Trajectories

The conversion of target trajectories from ordered sets of coordinates to parameterised functions can be viewed as finding a concise low-dimensional representation of discrete trajectories. We assume that each output trajectory comprises independent functions, $x(t)$ and $y(t)$, that model the x, y coordinates of the trajectory over time t. $x(t)$ and $y(t)$ give coordinates relative to the last waypoint coordinate of the queried discrete trajectory. A target trajectory recorded from time T' to T, $\boldsymbol{\xi^{Tar}} = \{(x_t, y_t)\}_{t=T'}^{T}$, is represented as weighted sums of projections to square exponential basis functions placed at fixed times, $\Xi(\cdot) = (\boldsymbol{w_x}^T\boldsymbol{\phi}(\cdot), \boldsymbol{w_y}^T\boldsymbol{\phi}(\cdot))$, where $\boldsymbol{\phi}(\cdot)$ represents the features, and $\boldsymbol{w_x}$, $\boldsymbol{w_y}$ are weights. Squared exponential basis functions are smooth and often used as a least informative default, though we are not restricted to using squared exponential bases. The weights parameters are found by solving kernel ridge regression problems with constraint $t = 0$:

$$\min_{\boldsymbol{w_x}} \sum_{n=1}^{T-T'} \big(x_n - \boldsymbol{w_x}^T\boldsymbol{\phi}(t_n)\big)^2 + \lambda_1 ||\boldsymbol{w_x}||^2 \tag{5a}$$

$$\text{s.t.} \quad \boldsymbol{w_x}^T\boldsymbol{\phi}(0) = 0 \tag{5b}$$

$$\min_{\boldsymbol{w_y}} \sum_{n=1}^{T-T'} \big(x_n - \boldsymbol{w_y}^T\boldsymbol{\phi}(t_n)\big)^2 + \lambda_1 ||\boldsymbol{w_y}||^2 \tag{6a}$$

$$\text{s.t.} \quad \boldsymbol{w_y}^T\boldsymbol{\phi}(0) = 0 \tag{6b}$$

where λ_1 is a regularisation coefficient, and $\boldsymbol{\phi(\cdot)}$ is a feature map defined by,

$$\boldsymbol{\phi}(t) = [k(t, \hat{t}_1), \dots, k(t, \hat{t}_{M_t})] = \Bigg[\exp\Big(-\frac{||\hat{t}_1 - t||^2}{2\ell_t}\Big), \dots, \exp\Big(-\frac{||\hat{t}_{M_t} - t||^2}{2\ell_t}\Big)\Bigg] \tag{7}$$

where $\hat{t_1}, \dots, \hat{t_{M_t}}$ is a set of M_t fixed points in time. We refer to these points as *inducing points*, and center the basis functions on them. ℓ_t is a length scale of the square exponential bases. Note that $\boldsymbol{\phi}$, projects to inducing points in time, and $\boldsymbol{\varphi}$, projects to representative trajectories. By including equations 5b and 6b as squared penalty terms, with penalty coefficient λ_2, to equations 5a and 6a, and equating derivatives to zero gives the solution to the minimisation problems,

$$\begin{aligned}\boldsymbol{w_x} &= \Big(\lambda_1\boldsymbol{I} + \lambda_2\boldsymbol{\phi}(0)^T\boldsymbol{\phi}(0) + \sum_{n=1}^{N}\boldsymbol{\phi}(t_n)^T\boldsymbol{\phi}(t_n)\Big)^{-1}\Big(\sum_{n=1}^{N} x_n\boldsymbol{\phi}(t_n)\Big),\\ \boldsymbol{w_y} &= \Big(\lambda_1\boldsymbol{I} + \lambda_2\boldsymbol{\phi}(0)^T\boldsymbol{\phi}(0) + \sum_{n=1}^{N}\boldsymbol{\phi}(t_n)^T\boldsymbol{\phi}(t_n)\Big)^{-1}\Big(\sum_{n=1}^{N} y_n\boldsymbol{\phi}(t_n)\Big).\end{aligned} \tag{8}$$

We can solve the minimisation problem to obtain vector of weights, $\boldsymbol{w_x}$ and $\boldsymbol{w_y}$, that parameterise the function $x(t)$ and $y(t)$ respectively. In this work, we define the same set of inducing points for $x(t)$ and $y(t)$, so both $\boldsymbol{w_x}$ and $\boldsymbol{w_y}$ are of dimensionality M_t, as there is a weight for each basis.

3.4 Learning a Mixture of Stochastic Processes

We extend our functional representation of trajectories to stochastic processes, akin to distributions over functions. To model stochastic processes $\{x_t\}_t$ and $\{y_t\}_t$, we fit distributions over the weight parameters of $x(t)$ and $y(t)$. Namely, we wish to find the probability distribution, $p(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D})$, where $\boldsymbol{w}$ is a vector containing both $\boldsymbol{w_x}$ and $\boldsymbol{w_y}$, and $\boldsymbol{\xi}^*$ is a queried trajectory. We consider the concatenation of vectors $\boldsymbol{w_x}$ and $\boldsymbol{w_y}$, $\boldsymbol{w}$ which has $2M_t$ elements. To permit multiple modes over the mean function, assume $\{x_t\}_t$ and $\{y_t\}_t$ can be expressed as a linear sum of R individual stochastic processes, which we shall call components. We can express $p(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D})$ as a linear sum with mixture coefficients $\alpha_r[\boldsymbol{\varphi}]$, where $\sum_{r=1}^{R} \alpha_r[\boldsymbol{\varphi}] = 1$. Each $\alpha_r[\boldsymbol{\varphi}^*]$ is a function on $\boldsymbol{\varphi}^*$, the projections of $\boldsymbol{\xi}^*$ via the DF-kernel, detailed in subsection 3.2. Defining the shorthand $\alpha_r := \alpha_r[\boldsymbol{\varphi}^*]$, we have,

$$p\big(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D}\big) = p(\boldsymbol{w}|\boldsymbol{\varphi}^*) = \sum_{r=1}^{R} \alpha_r p_r(\boldsymbol{w}|\boldsymbol{\varphi}^*). \tag{9}$$

In this work, we approximate the probability distribution of each element of $\boldsymbol{w}$ in each component, given a queried trajectory $\boldsymbol{\xi}^*$, to be independent Gaussian distributions. The mean, $\mu_{r,m}[\boldsymbol{\varphi^*}]$, and standard deviations, $\sigma_{r,m}[\boldsymbol{\varphi^*}]$, of the m^{th} weight of the r^{th} component are functions of $\boldsymbol{\varphi}^*$. For brevity, we use the shorthand $\mu_{r,m} := \mu_{r,m}[\boldsymbol{\varphi^*}]$ and $\sigma_{r,m} := \sigma_{r,m}[\boldsymbol{\varphi^*}]$. For the m^{th} weight of the r^{th} component, we have $p_r(w_m|\boldsymbol{\varphi}^*) = \mathcal{N}(\mu_{r,m}, \sigma^2_{r,m})$. Assuming weights are independent, the conditional probability over the vector of weights, of each component r, is $p_r(\boldsymbol{w}|\boldsymbol{\varphi}^*) = \prod_{m=1}^{2M} \mathcal{N}(\mu_{r,m}, \sigma^2_{r,m})$. We subsequently derive a loss function to learn $\mu_{r,m}$, $\sigma_{r,m}$, and α_r, for all r and m.

Let us consider the set of N observations of input and target trajectories, $\mathcal{D} = \{(\boldsymbol{\xi}^{\text{Obs}}, \boldsymbol{\xi}^{\text{Tar}})_n\}_{n=1}^{N}$. At the n^{th} observation, $\boldsymbol{\xi_n}^{\text{Obs}}$ is projected using the DF-kernel to obtain high-dimensional projections, $\boldsymbol{\varphi_n}$. Weights, $\boldsymbol{w_n}$, that parameterise $\boldsymbol{\Xi}^{\text{Tar}_n}$, continuous representations of discrete target trajectories, are then found by evaluating equation 8. Assuming that observations are independent and identically distributed, we can write the conditional density as,

$$p(\{\boldsymbol{w_n}\}_{n=1}^{N}|\{\boldsymbol{\varphi_n}\}_{n=1}^{N}) = \prod_{n=1}^{N} p(\boldsymbol{w_n}|\boldsymbol{\varphi_n}) = \prod_{n=1}^{N}\sum_{r=1}^{R} \alpha_r[\boldsymbol{\varphi_n}] \prod_{m=1}^{2M_t} \mathcal{N}(\mu_{r,m}, \sigma^2_{r,m}) \tag{10}$$

Fitting the conditional probabilities over weight parameters can be done by maximising 10. We define the loss function as,

$$\mathcal{L} = -\log\big\{p(\{\boldsymbol{w_n}\}_{n=1}^{N}|\{\boldsymbol{\varphi_n}\}_{n=1}^{N})\big\} \tag{11}$$

$$= -\sum_{n=1}^{N}\log\Big\{\sum_{r=1}^{R}\exp\Big[\log(\alpha_r) - 2M\log(2\pi) + \sum_{m=1}^{2M}\log(\sigma_{r,m}) - \sum_{m=1}^{2M}\frac{(w_{n,m}-\mu_{r,m})^2}{2\sigma^2_{r,m}}\Big]\Big\} \tag{12}$$

Constraints $\sum_{r=1}^{R}\alpha_r = 1$ can be enforced by applying a softmax activation function, $\alpha_r = \frac{\exp(z_r^a)}{\sum_{r=1}^{R}\exp(z_r^a)}$, where z_r^a denotes the network outputs of α_r. To enforce $\sigma_{r,m} \geq 0$, an exponential activation function, $\sigma_{r,m} = \exp(z^{\sigma}_{r,m})$, is applied to the network outputs corresponding to standard deviation. By utilising the expressiveness of our projection features, a simple mixture density network [27, 28], with a single hidden layer can then be used to learn the functions of parameters $\alpha_r[\boldsymbol{\varphi}]$, $\mu_{r,m}[\boldsymbol{\varphi}]$, $\sigma_{r,m}[\boldsymbol{\varphi}]$, by minimising our loss function via Stochastic Gradient Descent (SGD).

3.5 Conducting Inference and Obtaining Trajectory Realisations

After learning the functions $\alpha_r[\boldsymbol{\varphi}]$, $\mu_{r,m}[\boldsymbol{\varphi}]$, and $\sigma_{r,m}[\boldsymbol{\varphi}]$ as described in subsection 3.4, we have $p\big(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D}\big)$ via equation 9, and the assumption of independent Gaussian distributed weights. Given a vector of feature maps, $\boldsymbol{\phi}(t)$, to evaluate $p(\Xi^*(t)|\boldsymbol{\xi}^*, \mathcal{D}, \boldsymbol{\phi}(t)) = \int p(\Xi^*(t)|\boldsymbol{\phi}(t), \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D})d\boldsymbol{w}$, we have $p(\Xi^*(t)|\boldsymbol{\phi}(t), \boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}^T\boldsymbol{\phi}(t), s^2\mathcal{I})$ [29], where s denotes the standard deviation of the observation error. It is possible to estimate s^2 via $p(s^2|\mathcal{D}) \propto p(\mathcal{D}, s^2) = \int p(D|s^2, \boldsymbol{w})p(\boldsymbol{w})p(s^2)d\boldsymbol{w}$. Like [30] and [31], in this work, we focus on the deterministic observation case, where $s = 0$.

The inference process to sample continuous trajectories Ξ^{out} is outlined in algorithm 1. Under the assumption of deterministic observations, we evaluate $p(\Xi^*|\boldsymbol{\xi}^*, \mathcal{D}, \boldsymbol{\phi}) = \int p(\Xi^*|\boldsymbol{\phi}, \boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D})d\boldsymbol{w}$, by randomly sampling $p\big(\boldsymbol{w}|\boldsymbol{\xi}^*, \mathcal{D}\big)$, and obtaining realisations of continuous trajectories $\Xi^{out}(\cdot) \sim p(\Xi^*(\cdot)|\boldsymbol{\phi}(\cdot), \boldsymbol{w})$ by evaluating $(\boldsymbol{w}_x^T\boldsymbol{\phi}(\cdot), \boldsymbol{w}_y^T\boldsymbol{\phi}(\cdot))$. We can obtain a discrete trajectory $\boldsymbol{\xi}^{out}$ by querying $\Xi^{out}(\cdot)$ at times, $\boldsymbol{t} = [t_1, \ldots, t_n]$, i.e. $\boldsymbol{\xi}^{out} \leftarrow \Xi^{out}(\boldsymbol{t})$.

Algorithm 1: KTM Inference

```
input  : $\boldsymbol{\xi}^*$,$\alpha_r[\boldsymbol{\varphi}]$, $\mu_{r,m}[\boldsymbol{\varphi}]$, $\sigma_{r,m}[\boldsymbol{\varphi}]$, $\boldsymbol{\varphi}(\cdot)$, $\boldsymbol{\phi}(\cdot)$
output: Realised Continuous Trajectory, $\Xi^{\text{out}}(\cdot)$
1 begin
2     $\boldsymbol{\varphi}^* \leftarrow \boldsymbol{\varphi}(\boldsymbol{\xi}^*)$ // generate projections with DF-kernel
3     Evaluate $\alpha_r[\boldsymbol{\varphi}^*]$, $\mu_{r,m}[\boldsymbol{\varphi}^*]$, $\sigma_{r,m}[\boldsymbol{\varphi}^*]$ // Find parameters of mixture of SP
4     $p(\boldsymbol{w}|\boldsymbol{\varphi}^*) \leftarrow \sum_{r=1}^{R} \alpha_r p_r(\boldsymbol{w}|\boldsymbol{\varphi}^*)$
5     $\boldsymbol{w} \sim p(\boldsymbol{w}|\boldsymbol{\varphi}^*)$ // Sample $p(\boldsymbol{w}|\boldsymbol{\varphi}^*)$
6     $\Xi^{out}(\cdot) \leftarrow (\boldsymbol{w}_x^T\boldsymbol{\phi}(\cdot), \boldsymbol{w}_y^T\boldsymbol{\phi}(\cdot))$ // retrieve continuous trajectory
7 end
```

4 Experiments and Discussions

We wish to highlight the benefits KTMs bring. In particular: (1) map-awareness; (2) trajectory history awareness; (3) multi-modal probabilistic predictions, with continuous trajectory realisations.

4.1 Experimental Setup

We run experiments on both simulated and real-life trajectory datasets, including:

1. Simulated dataset (S): Simulated trajectories of pedestrians crossing a road, similar to the simulated datasets used in [3]
2. Edinburgh dataset [32] (E): Pedestrian trajectories in the real-world on September 24th
3. Lankershim dataset [33] (L): Subset of valid vehicle trajectories in the region between x-coordinates $-100\text{m}\sim 100\text{m}$ and y-coordinates $250\text{m}\sim 500\text{m}$

In experiments, each whole trajectory is segmented with an input-target ratio of 1:3, 1:1, or 3:1. The subset of the Lankershim dataset contains 6580 pairs of trajectories; Edinburgh 5972; simulated 600. We use $R = 4$ mixture components, and length scale $\ell_{DF} = 100$, for the DF-kernel, and $\ell_t = 10$ for the square exponential bases. Bases are centered evenly at 2.5 time step intervals for the Edinburgh dataset and 5 for the simulated and Lankershim datasets. Half the trajectories are used as representative trajectories. To adequately evaluate the ability of KTMs, we ensure representative trajectories are not included in testing. We randomly select 20% of trajectories outside of the representative set as test examples, or 10% of the total. To account for stationary vehicles, for the Lankershim dataset, we only evaluate trajectories that move more than 20m in 20 time steps. All values reported in metres. We train for 80 epochs, then evaluate on the test set. Inference can be conducted efficiently, with an average time below 0.2 sec for predicting mixture of processes, on all of our experiments with a standard desktop. Experiments are repeated 5 times, each with randomly selected test examples. We give quantitative results on the following realised trajectories from the output:

1. KTM-Weighted Average (KTM-W): A linear combination of the mean of each mixture component, weighted by the mixture coefficient;
2. KTM-Closest (KTM-C): The mean trajectory of the mixture component that is the closest to ground truth. Selecting the trajectory in this manner assumes the decision of which option, out of the four possible trajectories to take, is made correctly. This allows us to evaluate the quality of the predicted trajectory, without taking into account of the quality of decision-making;

3. Constant Velocity (CV): The trajectory is generated by a model that the velocity remains constant beyond the observations;
4. Directional Grid Maps (DGM): Directional grid map [3] is a recent method capable of producing directional predictions. We conduct forward sampling on a DGM, with a step size equal to that of the last observed step.

The metrics used to evaluate our trajectories are: (1) Euclidean distance (ED) between the end points of predicted and ground truth trajectories; (2) Discrete Fréchet distance (DF) [9, 10] between predicted and ground truth trajectories. Continuous trajectories are discretised for comparison.

4.2 Map-Awareness

Kernel Trajectory Maps learn to predict trajectories from a dataset of observed trajectories, which contain rich information about the structure of the environment, such as obstacles and designated paths. Methods that learn from a set of observed trajectories are intrinsically *map-aware* [11], and can account for environment geometry. Dynamics based models are often map-unaware, and are not able to anticipate a future changes in direction due to environmental factors, such as a turning road.

An example of map-awareness is demonstrated in figure 2. We sample realisations from the predicted mixture of a KTM, and compare it against the constant velocity (CV) model and ground truth trajectory. The sharp turn the ground truth trajectory takes is due to the road structure in the dataset, and there is little indication from the behaviour of the observed trajectory. The turn is not captured by the CV model, but is captured by the KTM.

Table 1 contains the quantitative results of methods described in subsection 4.1. We predict future trajectories over a horizon of 20 time steps. We see that map-aware methods, such as KTM and DGM, tend to outperform the CV model. Notably the CV model performs strongly for the Lankershim dataset, outperforming all but the KTM-C method, due to vehicle trajectories in that dataset being approximately constant velocity over small distances. The Edinburgh dataset contains pedestrian motion trajectories which are much more unstructured and unrestricted. Thus, KTM-C and KTM-W perform significantly stronger than the CV model.

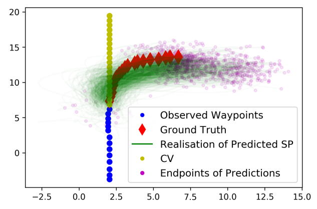

Figure 2: 1000 sampled trajectories from output mixture (green with magenta endpoints) anticipate the turn, as shown by the ground truth (red). There is little indication of the turn from observed waypoints (blue). The CV (yellow) model does not.

		KTM-C	KTM-W	CV	DGM
(S)	ED	1.3±0.1	1.8±0.2	6.5±0.3	4.4±0.1
	DF	1.4±0.1	1.9±0.1	6.3±0.3	4.4±0.1
(E)	ED	0.7±0.1	0.9±0.1	1.4±0.1	1.1±0.1
	DF	0.8±0.1	0.9±0.1	1.4±0.1	1.1±0.1
(L)	ED	5.8±0.3	11.5±0.2	11.3±0.2	11.4±0.2
	DF	6.3±0.3	11.5±0.5	10.7±0.2	11.4±0.2

Table 1: The performance of KTMs compared to a baseline CV model and an map-aware DGM model [3], on the Simulated dataset (S), the Edinburgh dataset (E), and the Lankershim dataset (L). We see that KTM-C outperforms the other methods, while KTM-W also gives a strong performance. KTMs benefit from map and trajectory history awareness. Note that the CV model performs well on the Lankershim dataset (L) due to the vehicle trajectories being approximately constant velocity of relatively short time horizons. Results given in meters.

4.3 Trajectory History Awareness

Recent attempts to encode multi-modal directional distributions in a map [3, 4, 5] largely condition only on the most recent coordinate, and are unable to utilise the full trajectory history of the object. KTMs are trajectory history aware, as demonstrated by trajectories sampled from a KTM trained on the simulated dataset, shown in figure 4. The predicted trajectories sampled vary significantly, though the positions of the last observed location are similar. Methods that condition solely on the most recent coordinate, can not differentiate between the two observed trajectories. The latter portion of the observed trajectories are similar, but with dissimilar early portions. By exploiting DF-kernels, KTMs give predictions conditioned on the entire trajectory. Although directional flow methods, such as DGM [3], are able to capture the general movement directions of dynamic objects,

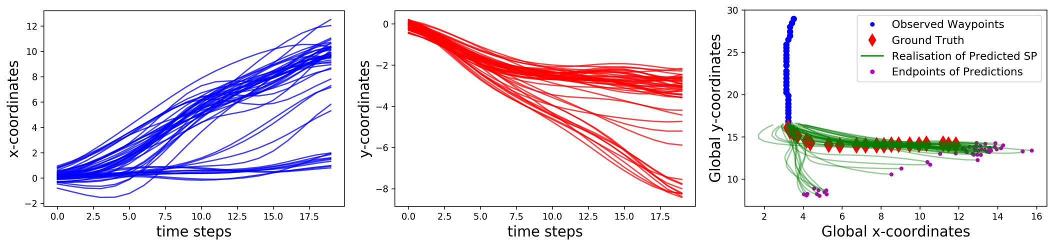

Figure 3: 50 realisations of $x(t)$ and $y(t)$ (left and center respectively), and the corresponding predicted trajectories (right). $x(t)$ and $y(t)$ give coordinates relative to the last observed coordinate.

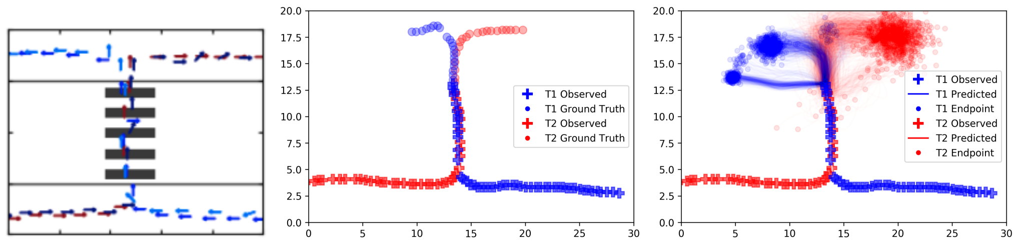

Figure 4: Examples of simulated trajectories (left). All trajectories starting at the lower left terminate at the upper right, and those starting at the lower right terminate at the upper left. The ground truth of the two trajectories, one starting at the lower left, the other at the lower right, are shown (center). Though the latest waypoints of both are similar, the KTM predictions are visibly different.

trajectories can only be obtained by making the Markov assumption and forward sampling. This process is sensitive to errors, and recursive behaviour can also arise. For example, a prediction at A points to B, which in turn may give a prediction pointing back to A. KTMs allow for realisations of entire trajectories, without forward sampling or making Markovian assumptions. The trajectory history awareness of KTMs explain the strong performance of KTMs relative to the DGM method, specifically on the simulated dataset, as shown in table 1.

4.4 Multi-modal Probabilistic Continuous Outputs

KTMs output mixtures of stochastic processes, corresponding to multi-modal distributions over functions. This provides us with information about groups of possible future trajectories with associated uncertainty. Figure 3 illustrates sampling functions from the outputted mixtures. The left and center plots show realisations of the functions $x(t)$ and $y(t)$, and the right plot shows the corresponding trajectory. There is clear multi-modality in the distribution over future trajectories.

A major benefit of KTMs is that realisations of the output are continuous functional trajectories. These are smooth and continuous, and do not commit to an *a priori* resolution. We can query any time value to retrieve predicted coordinates at the given time point. The functional representation with square exponential bases is inherently smooth, allowing us to operate on the derivatives of displacement. This property permits us to constrain certain velocity, acceleration, or jerk values.

5 Conclusion

In this paper, we introduce Kernel Trajectory Maps (KTM), a novel multi-modal probabilistic motion prediction method. KTMs are map-aware and condition on the whole observed trajectory. By projecting on a set of representative trajectories using expressive DF-kernels, we can use a simple single hidden layer mixture density network to arrive at a mixture of stochastic processes, equivalent to a multi-modal distribution over future trajectories. Each realisation of the mixture is a continuous trajectory, and can be queried at any time resolution. We recover whole trajectories without resorting to forward sampling coordinates. Empirical results show the awareness of the map and trajectory history improves performance when compared to a CV and map-aware, but not trajectory history aware, DGM model. The multi-modal and probabilistic properties of KTMs are also apparent from the experimental results. Future work will look into embedding social dynamics, and interaction between multiple predicted trajectories, into the KTM framework.

Acknowledgements

The authors thank Rafael Oliveira and Philippe Morere for the fruitful discussions. W. Zhi is supported by a Commonwealth of Australia Research Training Program Scholarship.

References

[1] X. Rong Li and V. P. Jilkov. Survey of maneuvering target tracking. part i. dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 2003.

[2] W. Zhi, R. Senanayake, L. Ott, and F. Ramos. Spatiotemporal learning of directional uncertainty in urban environments with kernel recurrent mixture density networks. *IEEE Robotics and Automation Letters*, 2019.

[3] R. Senanayake and F. Ramos. Directional grid maps: modeling multimodal angular uncertainty in dynamic environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

[4] T. P. Kucner, M. Magnusson, E. Schaffernicht, V. H. Bennetts, and A. J. Lilienthal. Enabling flow awareness for mobile robots in partially observable environments. *IEEE Robotics and Automation Letters*, 2017.

[5] L. McCalman, S. O'Callaghan, and F. Ramos. Multi-modal estimation with kernel embeddings for learning motion models. In *IEEE International Conference on Robotics and Automation*, 2013.

[6] B. Haasdonk and C. Bahlmann. Learning with distance substitution kernels. In *DAGM-Symposium, Lecture Notes in Computer Science*, 2004.

[7] B. Schölkopf. The kernel trick for distances. In *Advances in Neural Information Processing Systems*, 2000.

[8] M. M. Fréchet. Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, 1906.

[9] T. Eiter and H. Mannila. Computing discrete fréchet distance. Technical report, 1994.

[10] P. C. Besse, B. Guillouet, J. Loubes, and F. Royer. Review and perspective for distance-based clustering of vehicle trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 2016.

[11] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras. Human motion trajectory prediction: A survey. *CoRR*, 2019.

[12] D. Arbuckle, A. Howard, and M. Mataric. Temporal occupancy grids: a method for classifying the spatio-temporal properties of the environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[13] T. Kucner, J. Saarinen, M. Magnusson, and A. J. Lilienthal. Conditional transition maps: Learning motion patterns in dynamic environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[14] Z. Wang, R. Ambrus, P. Jensfelt, and J. Folkesson. Modeling motion patterns of dynamic objects by iohmm. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

[15] T. Krajnk, J. P. Fentanes, J. M. Santos, and T. Duckett. Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments. *IEEE Transactions on Robotics*, 2017.

[16] S. Molina, G. Cielniak, T. Krajník, and T. Duckett. Modelling and predicting rhythmic flow patterns in dynamic environments. In *Towards Autonomous Robotic Systems*, 2018.

[17] C. E. Rasmussen. *Gaussian Processes in Machine Learning*. 2004.

[18] E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, 2006.

[19] T. Barfoot, C. Tong, and S. Särkkä. Batch continuous-time trajectory estimation as exactly sparse gaussian process regression. In *Robotics: Science and Systems Conference*, 2014.

[20] Z. Marinho, B. Boots, A. Dragan, A. Byravan, G. J. Gordon, and S. Srinivasa. Functional gradient motion planning in reproducing kernel hilbert spaces. In *Robotics: Science and Systems*, 2016.

[21] G. Francis, L. Ott, and F. Ramos. Stochastic functional gradient for motion planning in continuous occupancy maps. In *IEEE International Conference on Robotics and Automation*, 2017.

[22] A. Woznica, A. Kalousis, and M. Hilario. Distances and (indefinite) kernels for sets of objects. In *International Conference on Data Mining (ICDM)*, 2006.

[23] C. K. I. Williams and M. Seeger. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, 2001.

[24] P. Drineas and M. W. Mahoney. On the nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 2005.

[25] A. E. Alaoui and M. W. Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems*, 2015.

[26] S. Kumar, M. Mohri, and A. Talwalkar. Sampling methods for the nyström method. *Journal of Machine Learning Research*, 2012.

[27] C. M. Bishop. Mixture density networks. Technical report, Dept. of Computer Science and Applied Mathematics, Aston University, 1994.

[28] A. Brando. Mixture density networks (mdn) for distribution and uncertainty estimation. Technical report, Universitat de Barcelona, 2017.

[29] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. 2006.

[30] N. De Freitas, A. J. Smola, and M. Zoghi. Exponential regret bounds for gaussian process bandits with deterministic observations. In *International Coference on International Conference on Machine Learning*, 2012.

[31] Z. Wang, B. Shakibi, L. Jin, and N. de Freitas. Bayesian multi-scale optimistic optimization. In *AISTATS*, 2014.

[32] B. Majecka. Statistical models of pedestrian behaviour in the forum. Technical report, School of Informatics, University of Edinburgh, 2009.

[33] Lankershim boulevard dataset. Technical report, Federal Highway Administration, 2007.