Constrained Gaussian Processes with Integrated Kernels for Long-Horizon Prediction of Dense Pedestrian Crowd Flows

Stefan H. Kiss, Kavindie Katuwandeniya, Alen Alempijevic, and Teresa Vidal-Calleja

***Abstract*—In this paper, we present a novel approach for predicting pedestrian crowd dynamics over longer time horizons (30s). In dense environments over long time horizons, the number of pedestrian interactions is high, leading to the degradation of traditional pedestrian trajectory estimation techniques. Alternatively, we consider the macroscopic properties of the crowd as a whole, focusing on the flow of density. This approach benefits from not considering pedestrians individually, and can probabilistically estimate the existence of previously unobserved individuals. Initially, a coarse resolution prediction is generated by a Convolutional Recurrent Neural Network (ConvRNN), and subsequently smoothly interpolated by a Gaussian Process (GP). Using the linearity properties of GPs, a continuous representation of the crowd is produced that complies with both the ConvRNN's prediction and a conservation of density constraint. The approach is trained and analysed on the dense ATC dataset, where we show the advantages of the approach and the improvements from our contributions.**

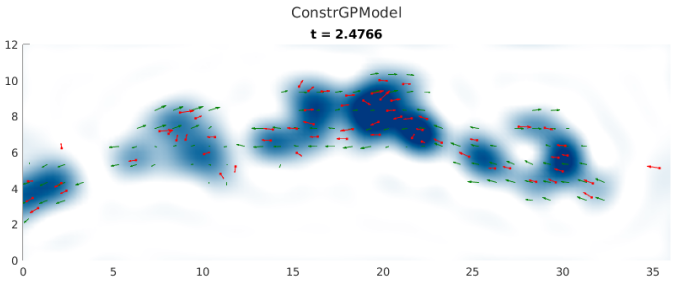


Fig. 1: The output of the presented method, overlaid with the ground-truth. As predicted almost 2.5 seconds into the future, the smooth density field ρ is shown in blue and the velocity field $\mathbf{v}$ is shown by green arrows. The red arrows indicate the true position and velocity of the pedestrians at this future time. Compare with Fig. 6.

I. Introduction

HUMANS tend to anticipate the motion of the people in their surroundings when navigating in crowds. This anticipation is their prediction capability, and is based on a life-long experience of walking in crowds where the physical and social attributes of crowd motions are implicitly learnt. The ability to foresee the future motion of the crowd allows humans to effectively plan paths less invasive of the crowd flow. This is the ideal behaviour we wish a robotic agent to execute when deployed in a dynamic crowded environment.

Robots deployed in crowded environments such as museums, airports, and shopping malls must be socially compliant. The research community has attempted to model socially compliant motion using potential fields [1], velocity obstacles [2], graph-based methods [3], and reinforcement learning methods [4], where social compliance is defined in terms of reaching a destination efficiently [5], [6], executing human-like maneuvers [4], or minimising invasiveness to the crowd's flow [7]. In non-myopic frameworks, the main idea is to predict in some way the motion of the crowd and effectively plan into the predicted future to reach a destination while satisfying the social compliance criteria. In this vein, we focus on developing an effective crowd prediction model, intended to be utilized for planning robot trajectories through the crowd with minimal social invasiveness. Predicting a pedestrian crowd's motion, also termed "crowd modelling", has been gaining the attention of the research community for planning robot motions [8], [9], and for various other purposes including efficient crowd evacuation [10], and traffic management and public safety [11].

Crowds can be described using their microscopic or macroscopic properties. Microscopic modelling considers each pedestrian individually, and typically handles interactions in a pairwise manner. Focusing on individual behaviours and interactions has been shown to work well locally, and is suitable in small, sparse crowds.

Alternatively, macroscopic modelling considers the entire crowd as a whole, focusing on the large-scale dynamics of denser crowds. At higher crowd densities, the microscopic properties of individuals have fewer degrees of freedom [12] and therefore have a small effect on the overall crowd behaviour. This emphasizes the importance of using macroscopic properties at larger scales. In our work, we are interested in modelling large and dense crowds and thus the focus is on macroscopic crowd properties, detailed in Section II.

Previous work on macroscopic crowd modelling [9] was shown to be effective for planning robot trajectories through dense crowds. A Convolutional Recurrent Neural Network (ConvRNN) framework was used to generate a discrete and fixed-resolution model. In this work, we improve upon the predictive model presented by using the coarsely predicted macroscopic crowd features to inform a constrained Gaussian Process (GP), giving a smooth and continuous representation of the crowd into the future. By filling-in the course resolution with a continuous probabilistic estimate,

All authors with the Robotics Institute at the University of Technology Sydney (UTS:RI), Australia.
Corresponding author stefan.h.kiss@student.uts.edu.au.

the crowd properties can be more accurately estimated at arbitrary query locations in space-time. With the inclusion of a known physical constraint, the conservation of density, the GP can intelligently interpolate to produce a physically plausible estimate. Moreover, the GP is able to correct unlikely estimates from the ConvRNN.

Additionally, as the framework produces a smooth and continuous representation of the crowd, gradients can be easily extracted from the GP to improve downstream applications such as planning and trajectory optimisation techniques [13].

An overview of our approach is as follows. The crowd is considered as a whole, and the macroscopic features of density and velocity are captured as averages over discretised space and time. A ConvRNN is used to encode the history of gridded values and predict future gridded values. Subsequently, as the main contribution of this paper, a constrained Gaussian process is used to reconstruct a smooth and continuous representation of the pedestrian crowd's macroscopic features, while adhering to physical constraints. This smooth representation of the macroscopic crowd properties produced by the prediction framework is shown in Fig. 1.

The macroscopic perspective of a pedestrian crowd is described in Section II, and an overview of the ConvRNN is presented in Section III. Gaussian Processes are explained in Section IV, with specific modifications made for the problem, such as interfacing with the ConvRNN and upholding physical constraints, detailed in Sections IV-A to IV-E. Finally, implementation details and results are given in Section V.

II. Macroscopic Crowd Features/Properties

Typical formulations of pedestrian prediction represent the crowd as a collection of 2D point masses moving in time. In contrast, we consider the crowd to be continuous in time and space, described by a set of *macroscopic properties*. Our model has no concept of individual pedestrians, but is defined over the ambient space. We denote a location in space-time as $\mathbf{x} = [t, x, y]^\mathsf{T} \in \mathbb{R}^3$, and describe the crowd by its density ρ and velocity $\mathbf{v} = [v_x, v_y]^\mathsf{T}$ at every location[1],

$$\begin{aligned} \rho &: \mathbb{R}^3 \to \mathbb{R}_{\geq 0}\,, \\ \mathbf{v} &: \mathbb{R}^3 \to \mathbb{R}^2\,. \end{aligned} \tag{1}$$

This formulation helps to capture the uncertainty of pedestrian positions over time in a probabilistic manner. We propose to model the crowd density and velocity fields as follows.

The density of the crowd ρ can be considered as the intensity of a *point process*: the expected number N of people within an area A at time t is given by the associated integral over that area,

$$\mathbb{E}\left[N_A(t)\right] = \iint\limits_{(x,y)\in A} \rho(t,x,y)\,\mathrm{d}x\,\mathrm{d}y\ . \tag{2}$$

The intensity of a point process is similar in many ways to a probability distribution: it is constrained to be non-negative, however it does not necessarily integrate to unity.

[1]With a slight abuse of notation, we will refer to functions f and their evaluated values $f(\mathbf{x})$ interchangeably.

The uncertainty of the velocity field is handled through the explicit inclusion of velocity variance $\sigma^2_{\mathbf{v}}$. Through a probabilistic loss function (described in Section III-A), this property captures two distinct factors: the uncertainty of the model's prediction, and, due to the discretised nature of the model's output, the variability of the velocities of pedestrians found within each discretisation region. This property is important to understand the coherence (conversely: irregularity) of the crowd flow.

III. ConvRNN Model

The problem of dynamic crowd prediction is first tackled by forecasting the macroscopic features at a coarse resolution. The space and time dimensions are discretised into fixed-resolution cells, and a Convolutional Recurrent Neural Network (ConvRNN) is used to learn the average future crowd properties across those cells.

As described in Section II, the macroscopic crowd features of interest are the density ρ and velocity $\mathbf{v}$ of the pedestrian crowd. The neural network is trained to estimate the *average* density and velocity values $\bar{\rho}$ and $\bar{\mathbf{v}}$ across the discretised cells $\mathfrak{c}$,

$$\begin{aligned} \bar{\rho}(\mathfrak{c}) &= \frac{1}{|\mathfrak{c}|} \iiint\limits_{\mathbf{x}\in\mathfrak{c}} \rho(\mathbf{x})\,\mathrm{d}t\,\mathrm{d}x\,\mathrm{d}y\,, \\ \bar{\mathbf{v}}(\mathfrak{c}) &= \frac{1}{|\mathfrak{c}|} \iiint\limits_{\mathbf{x}\in\mathfrak{c}} \mathbf{v}(\mathbf{x})\,\mathrm{d}t\,\mathrm{d}x\,\mathrm{d}y\ . \end{aligned} \tag{3}$$

We note that while the *true* target density function is not easy to define smoothly (an intuitive definition includes a sum of Dirac δ-functions at each pedestrian), this complication is avoided by considering the spatio-temporal averages.

For our experiments, we consider the network input to be a time-shifted version of the target output; a sequence of past crowd-property images is used to predict a sequence of future crowd-property images. However, we note that the input information could easily be of another available type, such as top-down colour images of the area of interest. For this work, the input and output are rasterised top-down images with cells describing the average macroscopic crowd properties across

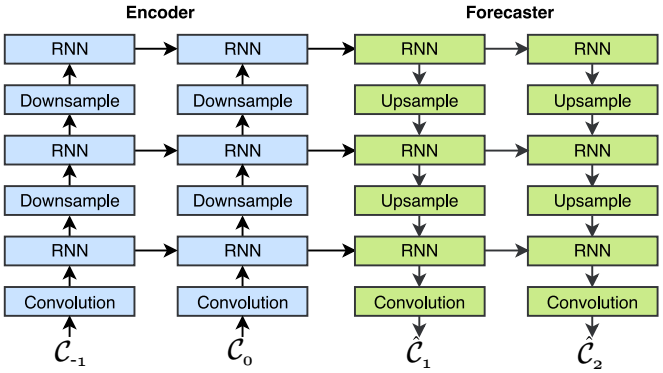


Fig. 2: The ConvRNN encoder-forecaster model. Past observations $\mathcal{C}_{-1}$ and $\mathcal{C}_0$ are used to generate future predictions $\hat{\mathcal{C}}_1$ and $\hat{\mathcal{C}}_2$. This structure can be repeated for arbitrary input and output sequence lengths. Figure adapted from [14].

them. In Fig. 2, these images are referred to as $\mathcal{C}_t$ at each time instance t.

The deep neural network model used was an encoder-forecaster structure consisting of ConvRNN blocks, where the Recurrent Neural Network (RNN) [15] responsible for maintaining the *temporal* relationships is selected as a Gated Recurrent Unit (GRU) [16] and the gating operations within the GRU are convolutions to maintain the *spatial* relationships. A graphical overview of the network structure is given in Fig. 2. The model was initially used for for weather prediction [14], [17], and we later applied it to macroscopic crowd prediction [9].

Convolutional Neural Networks (CNNs) [18] in the encoder-forecaster structure capture the *spatial* relationship of features by applying 2D convolutional kernels across the input images. The convolution blocks employ zero-padding such that their output is the same size as their input, while downsampling and upsampling blocks employ strided convolution and transposed-convolution respectively. The downsampling layers of the encoder allow it to learn coarse, high-level features at deeper layers while the upsampling layers of the forecaster allow it to build the predictive image back to its original resolution, assisted by the high-resolution hidden state passed from the previous timestep.

Encoder towers are iterated for each observation in the input resulting in hidden feature maps at differing resolutions, which are fed into forecaster towers to generate outputs of the same resolution to the input at future time steps of interest. The recurrent nature of the encoder-forecaster setup and the ability to store the hidden states in online usage allows the observations to be encoded and predictions to be made in constant time with respect to input length, and linearly with respect to prediction length.

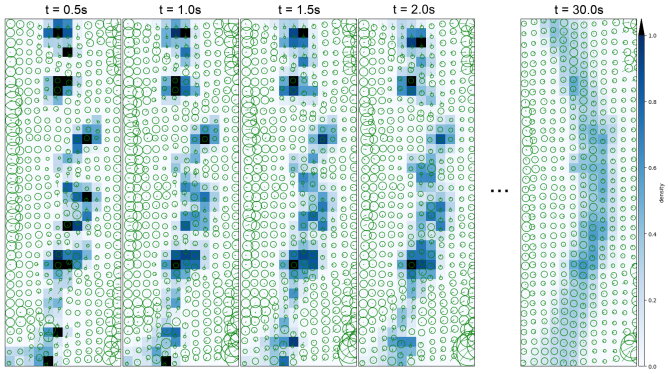


Fig. 3: The short and long time horizon behaviour of the ConvRNN prediction. Note that the final densities are not consistent for all inputs, the model seems to learn the overall density without being explicitly provided with the time of day information. Crowd density ρ is shown in blue, while the green arrows show the expected velocity with a circle at one standard deviation.

A. Probabilistic Loss Function

Through careful specification of the loss function, the learned model generates a *probabilistic* prediction of the future crowd state. The crowd is modelled as a *marked spatial point process*, with the spatial positions of an uncertain number of pedestrians each associated with an uncertain velocity. The density $\bar{\rho}(\mathfrak{c})$ is considered a Poisson random variable (as a Poisson point process) and the velocity $\bar{\mathbf{v}}(\mathfrak{c})$ a 2-dimensional isotropic Gaussian variable with a mean $\boldsymbol{\mu}_{\mathbf{v}}$ and a covariance $\sigma^2_{\mathbf{v}}\mathbb{I}_2$. Using the forward Kullback-Leibler (KL) divergence from the predictive distribution to the target ground truth as the model's loss function, the model is effectively optimised by Maximum Likelihood Estimation (MLE). See [9] for more details.

Such a probabilistic model is very beneficial for large-scale, long-horizon prediction of the macroscopic features. When predicting an intrinsically unknowable future, it is crucially important to capture the uncertainty in the modelling process. This allows the learned model to focus on regions it can be more certain about, and convey its uncertainty in regions it is not. The ConvRNN model shows its uncertainty at long time horizons by estimating large velocity variances, and blurring its predicted density image. In our crowd prediction problem, this is particularly useful as the model can seamlessly transition between collision-avoidance-like local behaviours at short time horizons, to a global environment-steady-state-like average behaviour at long time horizons. See Fig. 3 for a pictorial demonstration of this effect.

IV. Gaussian Processes

From the course resolution predicted grid of macroscopic crowd features, we wish to obtain a smooth, continuous representation of these values across space and time. We employ a Gaussian Process (GP) to smoothly interpolate these values, while constraining the output using known physical constraints.

A Gaussian process is a generalisation of a Gaussian random variable to a random function. A Gaussian process essentially describes an infinite number of jointly-Gaussian random variables $f(\mathbf{x}_i)$ at locations $\mathbf{x}_i$. It is described by a mean function $\mu(\mathbf{x})$ (commonly zero or constant), and a covariance or kernel function $K(\mathbf{x}, \mathbf{x}')$,

$$\begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_n) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{x}_1) \\ \vdots \\ \mu(\mathbf{x}_n) \end{bmatrix}, \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \cdots & K(\mathbf{x}_1, \mathbf{x}_n) \\ \vdots & \ddots & \vdots \\ K(\mathbf{x}_n, \mathbf{x}_1) & \cdots & K(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix} \right)$$

$$\begin{aligned} [f(\mathbf{x}_i)]_i &\sim \mathcal{N}([\mu(\mathbf{x}_i)]_i, [K(\mathbf{x}_i, \mathbf{x}_j)]_{i,j}) \ , \\ f(\mathbf{x}) &\sim \mathcal{GP}(\mu(\mathbf{x}), K(\mathbf{x}, \mathbf{x}')) \ . \end{aligned} \tag{4}$$

We introduce $[\bullet]_i$ and $[\bullet]_{i,j}$ notation to indicate vector and matrix construction (respectively) from repeated function evaluation.

Crucially, as all $f(\mathbf{x})$ variables are jointly-Gaussian, conditioning admits a closed-form solution. Given a set of observed data $\mathbf{y}_d = [f(\mathbf{x}_{d_i})]_i$ and a set of query locations $\mathbf{x}_q$, the

queried values $\mathbf{y}_q = [f(\mathbf{x}_{q_i})]_i$ are jointly-Gaussian, distributed as

$$\begin{aligned}\mathbf{y}_q \mid \mathbf{y}_d \sim \mathcal{N}\Big(\boldsymbol{\mu}_q &+ \boldsymbol{\Sigma}_{q,d}\,\boldsymbol{\Sigma}_{d,d}^{-1}\,(\mathbf{y}_d - \boldsymbol{\mu}_d),\\ &\boldsymbol{\Sigma}_{q,q} - \boldsymbol{\Sigma}_{q,d}\,\boldsymbol{\Sigma}_{d,d}^{-1}\,\boldsymbol{\Sigma}_{d,q}\Big)\,,\end{aligned} \tag{5}$$

where

$$\begin{aligned}\boldsymbol{\mu}_d &= [\mu(\mathbf{x}_{d_i})]_i\,, & \boldsymbol{\mu}_q &= [\mu(\mathbf{x}_{q_i})]_i\,,\\ \boldsymbol{\Sigma}_{d,d} &= [K(\mathbf{x}_{d_i},\mathbf{x}_{d_j})]_{i,j} + \boldsymbol{\Sigma}_{n,n}\,, & \boldsymbol{\Sigma}_{d,q} &= [K(\mathbf{x}_{d_i},\mathbf{x}_{q_j})]_{i,j}\,,\\ \boldsymbol{\Sigma}_{q,d} &= [K(\mathbf{x}_{q_i},\mathbf{x}_{d_j})]_{i,j}\,, \quad \text{and} & \boldsymbol{\Sigma}_{q,q} &= [K(\mathbf{x}_{q_i},\mathbf{x}_{q_j})]_{i,j}\,;\end{aligned}$$

assuming the observed data is additionally corrupted by some (also jointly Gaussian) noise with covariance $\boldsymbol{\Sigma}_{n,n}$.

Note that the definitions given for a scalar function $f(\mathbf{x})$ work equally well for a vector-valued function $\mathbf{f}(\mathbf{x})$. This however requires the definition of a vector-valued mean function $\boldsymbol{\mu}(\mathbf{x})$ and a (square) matrix-valued covariance function $\mathbf{K}(\mathbf{x},\mathbf{x}')$. The resulting data-data and query-data covariance matrices are similarly constructed as block matrices.

A. Linear Transformations of Gaussian Processes

One interesting and useful property of Gaussian processes is their linear transformation rules. Consider $\mathbf{f}_1(\mathbf{x})$, a GP defined by $\boldsymbol{\mu}_1(\mathbf{x})$ and $\mathbf{K}_{1,1}(\mathbf{x},\mathbf{x}')$. Given a linear operator $\mathcal{L}_2$ mapping $\mathbf{f}_1$ to $\mathbf{f}_2 : \mathbf{f}_2(\mathbf{x}) = \mathcal{L}_2\,\mathbf{f}_1(\mathbf{x})$, $\mathbf{f}_2(\mathbf{x})$ is also a Gaussian process, with associated mean and covariance functions (with itself, and with the un-transformed $\mathbf{f}_1$),

$$\begin{aligned}\boldsymbol{\mu}_2(\mathbf{x}) &= \mathcal{L}_2\,\boldsymbol{\mu}_1(\mathbf{x})\,,\\ \mathbf{K}_{2,2}(\mathbf{x},\mathbf{x}') &= \mathcal{L}_2\,\mathbf{K}_{1,1}(\mathbf{x},\mathbf{x}')\,\mathcal{L}_2^{\mathsf{T}}\,,\\ \mathbf{K}_{1,2}(\mathbf{x},\mathbf{x}') &= \quad\ \mathbf{K}_{1,1}(\mathbf{x},\mathbf{x}')\,\mathcal{L}_2^{\mathsf{T}}\,,\\ \mathbf{K}_{2,1}(\mathbf{x},\mathbf{x}') &= \mathcal{L}_2\,\mathbf{K}_{1,1}(\mathbf{x},\mathbf{x}')\,.\end{aligned} \tag{6}$$

The notation of these expressions can be confusing: the base kernel $\mathbf{K}_{1,1}$ is a matrix-valued function of 2 locations; the linear operator $\mathcal{L}_2$ operates from the left on the left location $\mathbf{x}$ and, with a slight abuse of notation, the transposed linear operator $\mathcal{L}_2^{\mathsf{T}}$ is understood to operate on the kernel from the right with respect to the right location $\mathbf{x}'$.

B. Conservation Constraint

Of the possible $\rho(\mathbf{x})$ and $\mathbf{v}(\mathbf{x})$ functions, defined over space and time, not all are physically feasible. The movement of density ρ through time is specified exactly by the flow velocity $\mathbf{v}$; as such these functions must obey the following differential equation to satisfy the conservation of density:

$$\frac{\partial\rho}{\partial t} + \frac{\partial\rho v_x}{\partial x} + \frac{\partial\rho v_y}{\partial y} = 0\,. \tag{7}$$

This equation describes the crowd's momentum vector $\rho\mathbf{v}$. The change in density over time is exactly the negative divergence of the momentum vector field: the density at a position decreases with the net flow of pedestrians away from that position. We can model this property of the flow as a constrained Gaussian process.

By the careful choice of kernel, the GP can be crafted to satisfy this constraint at all locations. Firstly, the constraint is described as a linear operation $\mathcal{L}_c$ (termed the *constraint operator*) on a *state* $\mathbf{s}$,

$$\begin{aligned}\mathbf{s} &= \begin{bmatrix}\rho\\ \rho v_x\\ \rho v_y\end{bmatrix}\,, \qquad \mathcal{L}_c = \begin{bmatrix}\partial_t & \partial_x & \partial_y\end{bmatrix}\,,\\ \mathcal{L}_c\,\mathbf{s} &= \begin{bmatrix}\partial_t & \partial_x & \partial_y\end{bmatrix}\begin{bmatrix}\rho\\ \rho v_x\\ \rho v_y\end{bmatrix} = 0\,,\end{aligned} \tag{8}$$

where $\partial_\bullet$ is the derivative operator $\frac{\partial}{\partial\bullet}$.

Subsequently, our aim is to find a linear operator $\mathcal{L}_s$, we term the *producing operator*, which satisfies

$$\begin{aligned}\mathbf{s} &= \quad \mathcal{L}_s\,\mathbf{g}\,,\\ 0 &= \mathcal{L}_c\,\mathcal{L}_s\,\mathbf{g}\,, \quad \forall\mathbf{g}\,.\end{aligned} \tag{9}$$

For any (vector-valued) continuous function $\mathbf{g}(\mathbf{x})$ defined over space and time, $\mathcal{L}_s\,\mathbf{g}$ should satisfy the constraint. We term $\mathbf{g}$ the *latent state*, and aim to find a valid solution for $\mathcal{L}_s$,

$$0 \underbrace{\xleftarrow{\mathcal{L}_c}}_{\substack{\text{constraint}\\ \text{operator}}} \underbrace{\mathbf{s}(\mathbf{x})}_{\text{state}} \underbrace{\xleftarrow{\mathcal{L}_s}}_{\substack{\text{producing}\\ \text{operator}}} \underbrace{\mathbf{g}(\mathbf{x})}_{\substack{\text{latent}\\ \text{state}}}\,. \tag{10}$$

A 1-dimensional solution for $\mathcal{L}_s$ can be written as a linear combination of scalar basis operators. Higher-dimensional solutions can be found spanning the null space of this linear combination, leading to a more a flexible model still satisfying the constraint. For a complete description of this process, readers are referred to [19]. For our specific problem, we find a 3-dimensional solution of the form:

$$\begin{aligned}\mathcal{L}_s &= \begin{bmatrix}0 & -\partial_y & \partial_x\\ \partial_y & 0 & -\partial_t\\ -\partial_x & \partial_t & 0\end{bmatrix}\,,\\ \mathcal{L}_c\,\mathcal{L}_s &= \begin{bmatrix}\partial_t & \partial_x & \partial_y\end{bmatrix}\begin{bmatrix}0 & -\partial_y & \partial_x\\ \partial_y & 0 & -\partial_t\\ -\partial_x & \partial_t & 0\end{bmatrix}\\ &= \begin{bmatrix}0 & 0 & 0\end{bmatrix}\,.\end{aligned} \tag{11}$$

Note that the composition of linear operators remains a linear operator, namely the *zero operator*, taking any 3-dimensional latent state $\mathbf{g}$ and reducing it to (scalar) zero everywhere.

This solution relies on the commutative property of differential operators, $\partial_x\partial_y = \partial_y\partial_x$. The solution can also be seen as the skew-symmetric matrix representation of the *curl* operator, and it is a well known result that the divergence of the curl of a vector field is always zero.

The properties of linear operators on GPs (as seen in Section IV-A) allow us to specify a covariance kernel $\mathbf{K}_{g,g}(\mathbf{x},\mathbf{x}')$ defining a GP for the latent state $\mathbf{g}$, and apply the operator $\mathcal{L}_s$. The resulting continuous function $\mathbf{s}$ must, by definition, satisfy the constraint $\mathcal{L}_c$. As the operator $\mathcal{L}_s$ is linear, the function of interest $\mathbf{s}$ remains a GP, with mean and kernel functions defined in terms of $\boldsymbol{\mu}_g$, $\mathbf{K}_{g,g}$, and $\mathcal{L}_s$,

$$\begin{aligned}\mathbf{g}(\mathbf{x}) &\sim \mathcal{GP}\Big(\quad \boldsymbol{\mu}_g(\mathbf{x}),\quad \mathbf{K}_{g,g}(\mathbf{x},\mathbf{x}')\quad\Big)\,,\\ \mathbf{s}(\mathbf{x}) &\sim \mathcal{GP}\Big(\mathcal{L}_s\,\boldsymbol{\mu}_g(\mathbf{x}),\,\mathcal{L}_s\,\mathbf{K}_{g,g}(\mathbf{x},\mathbf{x}')\,\mathcal{L}_s^{\mathsf{T}}\Big)\,.\end{aligned} \tag{12}$$

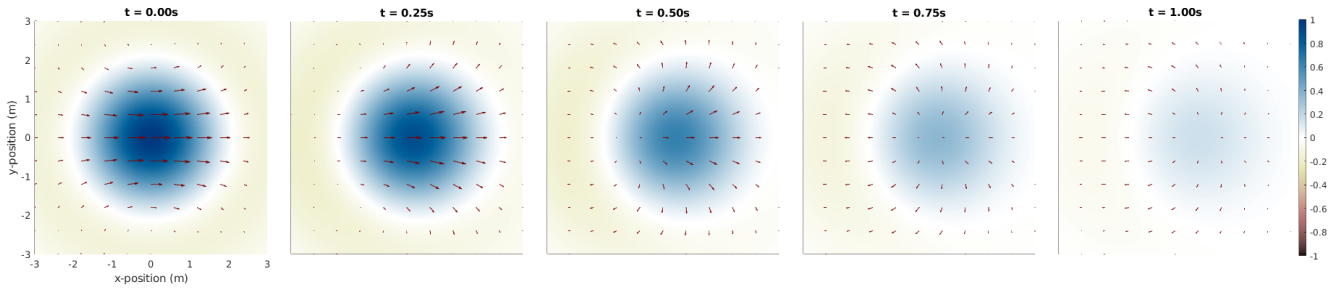


Fig. 4: A visualisation of the state's kernel $K_{s,s}$, showing how it satisfies the conservation of density constraint. The density field ρ is shown in orange-blue, while the maroon arrows show the flow of momentum $\rho\mathbf{v}$.

We note that for our work we define $\boldsymbol{\mu}_s = \mathcal{L}_s\,\boldsymbol{\mu}_g$ directly, as a constant positive density with zero velocity, with the understanding that the associated $\boldsymbol{\mu}_g$ exists and the constraint $\mathcal{L}_c$ is satisfied.

The base kernel $\mathbf{K}_{g,g}$ is defined with the typical squared-exponential,

$$\mathbf{K}_{g,g} = \begin{bmatrix} K_{g_1} & 0 & 0 \\ 0 & K_{g_2} & 0 \\ 0 & 0 & K_{g_3} \end{bmatrix}, \tag{13}$$

$$K_{g_i} = a_i^2 \exp\left(-\sum_{j\in(t,x,y)} \frac{[\mathbf{x}-\mathbf{x}']_j^2}{2\,l_{i,j}^2} \right), \tag{14}$$

where $[\bullet]_j$ denotes extracting the scalar value from the applicable dimension of the associated vector, such that different length scales $l_{i,j}$ can be used for each dimension.

The diagonal nature of $\mathbf{K}_{g,g}$ establishes the 3 dimensions of the latent state $\mathbf{g}$ as uncorrelated, however we note that the 3 dimensions of the produced state $\mathbf{s}$ *are* correlated as $\mathcal{L}_s$ contains off-diagonal elements.

Three variances a_i^2 and nine length-scales $l_{i,j}$ define hyper-parameters of this kernel. To retain symmetry in the xy-plane, 6 equality constraints are introduced leaving 6 tuneable hyper-parameters, which are selected empirically.

The constrained kernel $\mathbf{K}_{s,s}$ is visualised in Fig. 4, as the prediction generated under a single observation of a density moving towards the right. It can be see through the multiple time slices that the density complies with the flow of momentum; in this example the density flows towards the right and dissipates into the surroundings. Note that while negative densities are shown, multiple kernels will be added together with the mean function, resulting in positive densities.

C. Integral Kernel

The output of the ConvRNN outlined in Section III is a 3-dimensional array of predicted crowd properties, interpreted as the average values across cells in space-time. This naturally gives rise to a piecewise-constant model. A smoother model is produced by using these average values to inform a GP model of the ConvRNN's predictions. We introduced a similar approach for correlating rectangular spatial regions of differing sizes in [20].

The average value over a cellular region is calculated by the triple integral over space and time. This transformation can be described as a linear operator $\mathcal{L}_m$ which we use to define a *measurement state* $\mathbf{m}$,

$$\begin{bmatrix} \bar{\rho} \\ \overline{\rho v_x} \\ \overline{\rho v_y} \end{bmatrix} = \mathbf{m}(\mathfrak{c}) = \mathcal{L}_m\,\mathbf{s}(\mathbf{x}) = \frac{1}{|\mathfrak{c}|} \iiint_{\mathbf{x}\in\mathfrak{c}} \mathbf{s}(\mathbf{x})\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z \tag{15}$$

Note the similarities to the average values $\bar{\rho}(\mathfrak{c})$ and $\bar{\mathbf{v}}(\mathfrak{c})$ introduced in Section III.

As the *measurement operator* $\mathcal{L}_m$ is linear, if the state $\mathbf{s}$ is a GP then the measurement state $\mathbf{m}$ is also a GP with the covariance kernel modified by $\mathcal{L}_m$,

$$\underbrace{\mathbf{m}(\mathfrak{c})}_{\substack{\text{measurement}\\ \text{state}}} \underbrace{\longleftarrow \mathcal{L}_m \longrightarrow}_{\substack{\text{measurement}\\ \text{operator}}} \underbrace{\mathbf{s}(\mathbf{x})}_{\text{state}}. \tag{16}$$

We note that this averaging operation is equivalent to convolving the kernel with an appropriately sized rectangular window filter. This approach can therefore be considered more generally, by convolving the state's kernel with any number of measurement filters, as all convolutions are linear integration operators. Analytic evaluation of the convolved kernel, however, may prove more difficult.

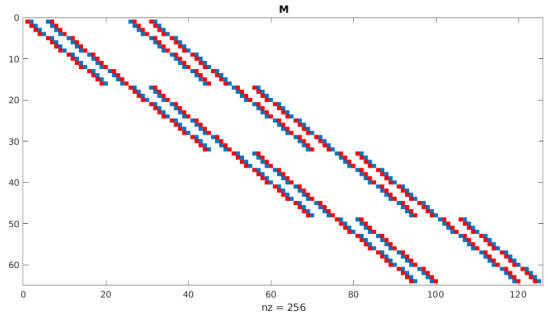
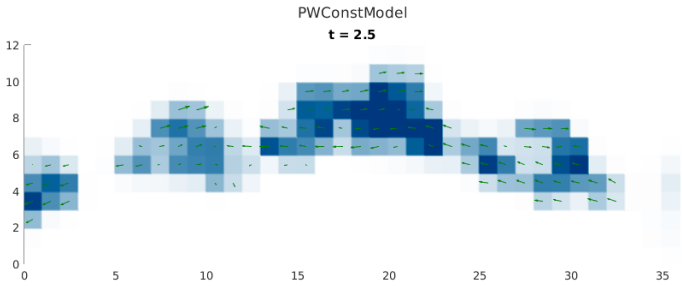
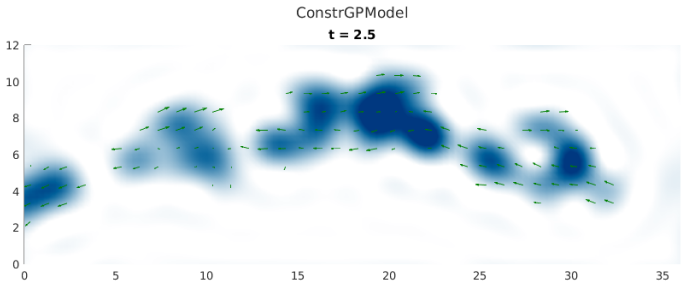


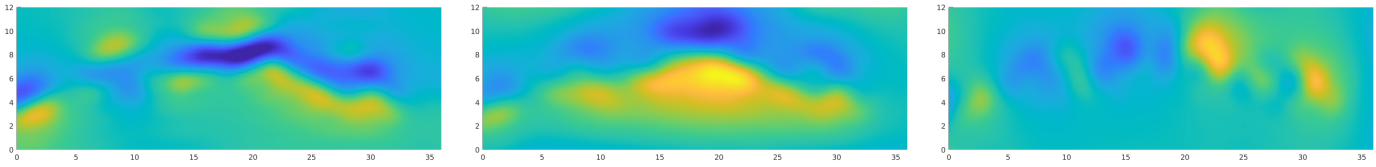
Fig. 5: Demonstrating the sparse nature, the $\mathbf{M}_{\mathfrak{c},\mathbf{x}}$ matrix is shown for a $4\times4\times4 = 64$ cell volume with $(4{+}1){\times}(4{+}1){\times}(4{+}1) = 125$ corners. Due to the surface-volume ratio, as the number of cells increases the aspect of this matrix becomes more square, increasing the efficiency. Blue cells indicate positive entries whereas red cells indicate negative entries.



(a) Measurement state $\mathbf{m}$.



(b) Continuous state $\mathbf{s}$.



(c) The three components of latent state $\mathbf{g}$: g_1, g_2, and g_3 from left to right.

Fig. 6: The coarse $\mathbf{m}$ grid, the continuous $\mathbf{s}$, and the components of $\mathbf{g}$. In Figs. 6a and 6b, the density field is shown in blue and the flow velocity is shown with green arrows. In Fig. 6c, the latent values are shown in blue-yellow with teal zero. All views are taken at a single slice in time, and Fig. 1 additionally shows the true pedestrian locations at this time.

To simplify evaluation, the measurement operator $\mathcal{L}_m$ is split into two parts. We use an approach similar to that of a continuous summed-area table. First, the quantity of interest is integrated in space and time with $\int_t \int_x \int_y$, where $\int_\bullet$ is the integration operator

$$\int_\bullet f(\bullet) = \int_{-\infty}^{\bullet} f(\tau)\,\mathrm{d}\tau \ . \tag{17}$$

Secondly, the function values at the corners are added and subtracted as appropriate to form the definite integral of Eq. (15). The data cells $\mathfrak{c}_d$ are specified by the eight (2^3) corners of each observation volume $\mathbf{x}_d$. The average value within each cell can be expressed as a sparse linear combination of the integrated value at the eight corners of the cell. For convenience, this transformation is given by the matrix $\mathbf{M}_{\mathfrak{c},\mathbf{x}}$. The sparse nature of $\mathbf{M}_{\mathfrak{c},\mathbf{x}}$ is shown in Fig. 5 for a $4 \times 4 \times 4 = 64$ cell volume.

Thus, the measurement operator $\mathcal{L}_m$ can be re-written as

$$\mathcal{L}_m = \mathbf{M}_{\mathfrak{c},\mathbf{x}} \int_t \int_x \int_y . \tag{18}$$

The integrations are applied directly to the kernel where appropriate (see Section IV-D) and the sparse matrix $\mathbf{M}$ is multiplied after calculating the kernel values at the data locations $\mathbf{x}_d$. This efficiently allows the kernel to be evaluated at each corner only once, and used in the calculation of each of the neighboring cell values. As most corners are shared by eight neighboring cells, this reduces the complexity of computing the kernels approximately eight-fold.

D. Kernels

The continuous latent state $\mathbf{g}(\mathbf{x})$ is defined as a GP, and transformed to the discretised cell measurements $\mathbf{m}(\mathfrak{c})$ observed and predicted by the ConvRNN with the linear operators $\mathcal{L}_s$ and $\mathcal{L}_m$ specified in Sections IV-B and IV-C respectively,

$$\mathbf{m}(\mathfrak{c}) \xleftarrow{\mathcal{L}_m} \mathbf{s}(\mathbf{x}) \xleftarrow{\mathcal{L}_s} \mathbf{g}(\mathbf{x}) \,. \tag{19}$$

Visualisations of these different states are shown in Fig. 6.

The kernel functions of interest can be specified following the linear transformation properties of GPs described in Section IV-A,

$$\begin{aligned} \mathbf{K}_{m,m}(\mathfrak{c},\mathfrak{c}') &= \mathcal{L}_m\, \mathcal{L}_s\, \mathbf{K}_{g,g}(\mathbf{x},\mathbf{x}')\, \mathcal{L}_s^\mathsf{T}\, \mathcal{L}_m^\mathsf{T} \,, \\ \mathbf{K}_{s,m}(\mathbf{x},\mathfrak{c}') &= \quad\ \mathcal{L}_s\, \mathbf{K}_{g,g}(\mathbf{x},\mathbf{x}')\, \mathcal{L}_s^\mathsf{T}\, \mathcal{L}_m^\mathsf{T} \,. \end{aligned} \tag{20}$$

Finally, the full covariance matrices between the observed data d and itself, and between the observed data and the queried values q, respectively, can be calculated as

$$\begin{aligned} \mathbf{\Sigma}_{d,d} &= \left[\mathbf{K}_{m,m}(\mathfrak{c}_{d_i},\mathfrak{c}_{d_j})\right]_{i,j} + \mathbf{\Sigma}_{n,n} \,, \\ \mathbf{\Sigma}_{q,d} &= \left[\mathbf{K}_{s,m}(\mathbf{x}_{q_i},\mathfrak{c}_{d_j})\right]_{i,j} \,. \end{aligned} \tag{21}$$

The observed data is assumed to be additionally corrupted by measurement noise with covariance $\mathbf{\Sigma}_{n,n}$, the form of which is non-trivial and is described in Section IV-E below. The Gaussian conditioning equations can then be used to calculate the posterior distribution as per Eq. (5). However are primarily focused on the mean of the prediction,

$$\mathbb{E}\left[\mathbf{s}_q \mid \mathbf{m}_d\right] = \boldsymbol{\mu}_q + \mathbf{\Sigma}_{q,d}\mathbf{\Sigma}_{d,d}^{-1}\left(\mathbf{m}_d - \boldsymbol{\mu}_d\right) \,. \tag{22}$$

E. Observation Measurement Noise

We consider the predictions from the ConvRNN as measurements to inform the Gaussian process. The estimated values of density $\bar{\rho}$ and velocity $\bar{\mathbf{v}}$ are assumed to be corrupted by noise, which allows the GP to correct unlikely (or impossible, due to the constraint) predictions from the neural network. The noise is assumed to be independent between density $\bar{\rho}$ and velocity $\bar{\mathbf{v}}$ measurements, therefore the noise on the

measurement state $\mathbf{m}$, defined in terms of momentum $\overline{\rho\mathbf{v}}$, is in fact correlated.

The neural network is considered to output parameters $(\mu_\rho, \mu_{v_x}, \mu_{v_y}, \sigma^2_{\mathbf{v}})$, describing the variables with uncertainty (from their Poisson and isotropic Gaussian definitions, as described in Sections II and III-A),

$$\begin{aligned} \mathbb{E}\left[\bar{\rho}\right] &= \mu_\rho\,, & \mathbb{V}[\bar{\rho}] &= \sigma^2_\rho\,, \\ \mathbb{E}\left[\bar{\mathbf{v}}\right] &= \begin{bmatrix}\mu_{v_x}\\ \mu_{v_y}\end{bmatrix}\,, & \mathbb{V}[\bar{\mathbf{v}}] &= \begin{bmatrix}\sigma^2_{\mathbf{v}} & 0\\ 0 & \sigma^2_{\mathbf{v}}\end{bmatrix}\,. \end{aligned} \tag{23}$$

The density variance σ^2_ρ is assigned a constant, empirically selected value of 0.01.

Based on the definition of covariance of a random variable, and using the independence assumption between ρ and $\mathbf{v}$, the covariance of the observation noise $\mathbf{\Sigma}_{n,n}$ can be written as

$$\begin{aligned} \mathbf{\Sigma}_{n,n} &= \mathbb{V}[\mathbf{m}] = \mathbb{E}\left[\mathbf{m}\mathbf{m}^\mathsf{T}\right] - \mathbb{E}\left[\mathbf{m}\right]\mathbb{E}\left[\mathbf{m}\right]^\mathsf{T} \\ &= \begin{bmatrix} \sigma^2_\rho & & \text{(sym.)} \\ \mu_{v_x}\sigma^2_\rho & \mu^2_\rho\sigma^2_{\mathbf{v}}+\mu^2_{v_x}\sigma^2_\rho+\sigma^2_\rho\sigma^2_{\mathbf{v}} & \\ \mu_{v_y}\sigma^2_\rho & \mu_{v_x}\mu_{v_y}\sigma^2_\rho & \mu^2_\rho\sigma^2_{\mathbf{v}}+\mu^2_{v_y}\sigma^2_\rho+\sigma^2_\rho\sigma^2_{\mathbf{v}} \end{bmatrix}. \end{aligned} \tag{24}$$

V. Implementation and Evaluation

A. Dataset

The ATC shopping centre dataset [21] was used for both training and evaluation of the proposed framework. The dataset captures pedestrians navigating the ATC shopping centre, two days a week for over two years. We used the weekend (Sunday) subset of the data as they typically show peak densities over three times greater than the weekday (Wednesday) data. We also focused our attention to the eastern corridor as it is an especially dense region. Note that this subset still contains a wide variety of densities, as the crowd levels fluctuate dramatically according to the time of day. With a total of 41 days, 30 were used for training, 10 for validation, and 1 for evaluation.

Fig. 7 shows an overview of the ATC dataset, showing the discretisation grid used over the eastern corridor.

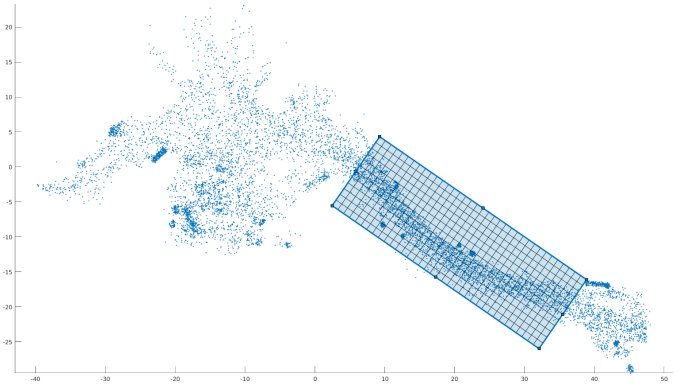


Fig. 7: A visualisation of the dense east hallway section of the ATC dataset used for evaluation. The 12×36 grid used for the discretisation process is shown. For context, $10\,000$ pedestrian locations are randomly selected from a typical Sunday.

B. Implementation Details

The ConvRNN model described in Section III was implemented in PyTorch [22]. The input observations and output predictions spatially are 12×36 raster image sequences where each cell is $1m \times 1m$. Temporally, each cell represents a duration of $0.5s$, and the network observes 10 frames for a total of $5s$. The model can then be used to predict any number of frames, however it was trained using gradient descent to predict 60 frames ($30s$).

The output of the ConvRNN is subsequently smoothed to any resolution using the proposed constrained GP method, implemented in MATLAB [23]. The kernel hyper-parameters (described in Section IV-B and Eq. (14)) were selected empirically as the following:

$$\begin{aligned} a_1 &= 0.5\,, & a_2 &= a_3 = 0.5\,, \\ l_{1,t} &= 2.0\,, & l_{2,x} &= l_{2,y} = l_{3,x} = l_{3,y} = 1.5\,, \\ l_{1,x} &= l_{1,y} = 1.5\,, & l_{2,t} &= l_{3,t} = 1.0\,. \end{aligned} \tag{25}$$

We note that these hyper-parameters as well as the neural network parameters could be trained using an objective function based on the continuous representation, back-propagated through the GP, however this is left for future work.

After interpolation, generated values are clamped within physically-plausible bounds, to prevent extreme outliers corrupting the results.

C. Evaluation Metrics

Traditional metrics for pedestrian trajectory prediction using microscopic features includes the Average Displacement Error (ADE) and Final Displacement Error (FDE). These are calculated as the difference in position per person between the prediction and the ground truth, averaged along the trajectory or at the final endpoint of the trajectory respectively. These metrics are typically extended to $\min_k$ADE and $\min_k$FDE respectively when the predictive model produces a distribution: k trajectory samples (often 20) are generated and evaluated, with the minimum error selected.

As our work focuses on the macroscopic crowd features, our approach has no concept of individual pedestrians; therefore we need to extend the tradition definitions of these metrics to models in the un-associated, macroscopic feature domain. We achieve this by sampling from the density field, considered as the intensity of a Poisson point process at every slice in time. A Poisson point process can be sampled by first uniformly sampling positions in the region of interest at a rate of the maximum intensity, and then accepting samples proportional to the underlying density field at that those locations. Positions are sampled from the density field k times over and the ground-truth pedestrian positions are associated with the closest sample (without overlap, using optimal assignment), achieving a generalisation of $\min_k$ADE and $\min_k$FDE.

Additionally note that our macroscopic approach allows us to jointly estimate location *and number* of pedestrians, and extends the $\min_k$ADE and $\min_k$FDE metrics to this modality, as opposed to traditional microscopic methods which are unable to estimate the existence of pedestrians which have not yet been observed. This is especially important in the

modelling of dense areas over longer time horizons, where the majority of pedestrians exit the scene and new individuals enter.

We also analyse the velocity predicted by the model, considering the magnitude of the velocity error at the true locations of the pedestrians. Similarly to displacement error, two metrics are produced: Average Velocity Error (AVE) and Final Velocity Error (FVE).

D. Results

Fig. 3 shows the prediction of the ConvRNN model, showing the short and long time horizon prediction behaviour. Qualitatively, we observe that the initial frames are high-contrast and quite certain, whereas the last frame is blurred uncertain. This demonstrates the power of the macroscopic feature approach: while individuals and small groups can be tracked initially, the model can smoothly blend towards a holistic, probabilistic crowd prediction.

At longer time horizons, the prediction converges towards a steady-state average behaviour, capturing the usual densities and traffic patterns of the space. We note that the densities in the corridor fluctuate dramatically throughout the day. Interestingly, we find the ConvRNN manages to learn this property; while the network is not directly informed of the time of day, it is able to learn average density patterns applicable to the time of day from only a few (10) observations of the current densities.

The model was evaluated on a previously unseen day of the ATC dataset. As we focus our attention to particularly dense scenarios, only sequences with an average density of 30 pedestrians or above are selected. A total of 1323 sequences are analysed; for each sequence, 5 seconds are observed by the framework and used to predict the following 30 seconds.

The $\min_k$ADE, $\min_k$FDE, AVE, and FVE metrics are calculated as described above, and displayed in Table I. The errors from the Constrained Gaussian Process (CGP) model are compared to a baseline Piece-Wise Constant (PWC) model, directly using the rasterised predictions from the ConvRNN. We note that due to the geometry of the spatial discretisation, the PWC model cannot reduce the $\min_k$ distance error metrics below a theoretical limit greater than 0.1.

TABLE I: Experimental Results

Metric	PWC	CGP	Change	
$\min_k$ADE (m)	0.2387	0.2372	-0.001477	-0.62%
$\min_k$FDE (m)	0.2611	0.2586	-0.002504	-0.96%
AVE (m/s)	0.5780	0.5791	$+0.000541$	$+0.19\%$
FVE (m/s)	0.7158	0.7068	-0.004476	-1.25%

Overall, both models perform well under the distance error metrics. While the authors were unable to find publicly available results for comparable methods in a similarly dense environment or over such a long time horizon, they note that these results seem to outperform state-of-the-art Social GAN [24]. For example, when evaluated on the easier (as implied by [25]) ETH dataset [26], Social GAN's $\min_k$FDE scores drop from $1.22m$ to $1.52m$ between 8 and 12 seconds, whereas our approach maintains an accuracy of $0.26m$ at the considerably longer prediction horizon of 30 seconds.

The table shows that the CGP model improves upon the PWC model by a small percentage in the $\min_k$ADE and $\min_k$FDE metrics. Notably, the *final* displacement metric makes a greater improvement than the *average* displacement metric, implying that this method is more suitable at longer time horizon predictions. As the ConvRNN predictions become less certain, they become smoother, which agrees with the Gaussian process's assumption of smoothness.

The velocity error metrics demonstrates this phenomenon more strongly, which Fig. 8 shows in detail. Application of the Gaussian process is actually detrimental for velocity prediction for horizons less than 8 seconds, however it provides a consistent improvement from 10 seconds onwards. Again, we attribute this to the smoothness assumption of the Gaussian process kernel. Additional careful tuning of the kernel hyperparameters or modification of the base kernel $\mathbf{K}_{g,g}$ may render the Gaussian process equally applicable across all prediction horizons.

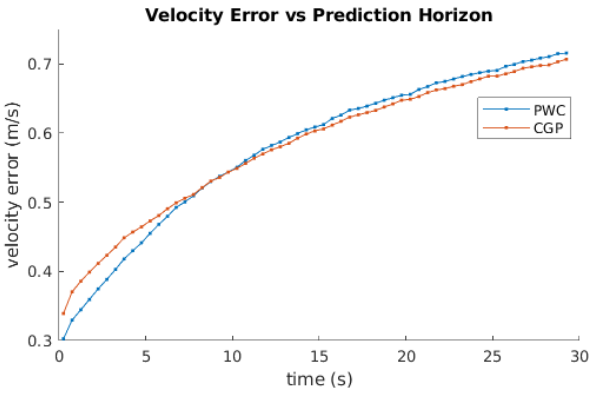


Fig. 8: While the CGP model does not improve upon the PWC prediction at short time horizons, it is more accurate at longer time horizons when the ConvRNN's predictions are smoother.

VI. Conclusion

In this work, we have proposed a method of crowd prediction using a continuous representation of the macroscopic features over space and time. The framework builds upon a ConvRNN model that that produces a coarsely discretised array of values into the future, representing the average expected value in each space-time cell. Using the properties of linear operators, a GP is used to interpolate the mean values into smooth continuous values while adhering to the conservation of density.

The dense ATC dataset is used for testing, and the framework is validated. The macroscopic approach is shown to be effective, and the constrained Gaussian process model provides a slight improvement over the baseline PWC model. In future work, we intend to make use of the probabilistic forecast to plan non-invasive robot trajectories through dense pedestrian crowds.

REFERENCES

[1] S. S. Ge and Y. J. Cui, "Dynamic Motion Planning for Mobile Robots Using Potential Field Method," *Autonomous robots*, vol. 13, no. 3, pp. 207–222, 2002.

[2] P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Environments Using Velocity Obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[3] J. J. H. Lee, C. Yoo, S. Anstee, and R. Fitch, "Efficient optimal planning in non-FIFO time-dependent flow fields," in *arXiv:1909.02198 [cs.RO]*, 2019, pp. 1–10.

[4] P. Henry, C. Vollmer, B. Ferris, and D. Fox, "Learning to navigate through crowded environments," *Proc. of IEEE ICRA*, pp. 981–986, 2010.

[5] W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "The interactive museum tour-guide robot," in *Aaai/iaai*, 1998, pp. 11–18.

[6] E. Prassler, J. Scholz, and P. Fiorini, "A robotics wheelchair for crowded public environment," *IEEE Robotics Automation Magazine*, vol. 8, no. 1, pp. 38–45, 2001.

[7] S. H. Kiss, K. To, C. Yoo, R. Fitch, and A. Alempijevic, "Minimally invasive social navigation," in *ACRA*, 2019, pp. 1–8.

[8] A. Rudenko, L. Palmieri, and K. O. Arras, "Joint Long-Term Prediction of Human Motion Using a Planning-Based Social Force Approach," pp. 4571–4577, 2018.

[9] S. H. Kiss, K. Katuwandeniya, A. Alempijevic, and T. Vidal-Calleja, "Probabilistic dynamic crowd prediction for social navigation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9269–9275.

[10] B. G. Silverman, M. Johns, J. Cornwell, and K. O'Brien, "Human behavior models for agents in simulators and games: part i: enabling science with pmfserv," *Presence: Teleoperators & Virtual Environments*, vol. 15, no. 2, pp. 139–162, 2006.

[11] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," *arXiv preprint arXiv:1610.00081*, 2016.

[12] R. Narain, A. Golas, S. Curtis, and M. C. Lin, "Aggregate Dynamics for Dense Crowd Simulation," in *ACM Transactinos on Graphics, SIGGRAPH*, vol. 28, no. 5, 2009, pp. 122:1—-122:8.

[13] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[14] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Y. Yeung, W.-k. K. Wong, and W.-c. C. Woo, "Deep learning for precipitation nowcasting: A benchmark and a new model," in *Advances in neural information processing systems*, vol. 2017-Decem, no. Nips, 2017, pp. 5617–5627.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

[16] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[17] X. Shi, Z. Chen, and H. Wang, "Convolutional LSTM Network," *Nips*, pp. 2–3, 2015.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] C. Jidling, N. Wahlström, A. Wills, and T. B. Schön, "Linearly constrained Gaussian processes," *arXiv*, no. Nips, 2017.

[20] L. Jin, J. Rückin, S. H. Kiss, T. Vidal-Calleja, and M. Popović, "Adaptive-Resolution Gaussian Process Mapping for Efficient UAV-based Terrain Monitoring," pp. 1–7, 2021. [Online]. Available: http://arxiv.org/abs/2109.14257

[21] D. Brščić, T. Kanda, T. Ikeda, and T. Miyashita, "Person tracking in large public spaces using 3-D range sensors," *IEEE Transactions on Human-Machine Systems*, vol. 43, no. 6, pp. 522–534, 2013.

[22] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[23] MATLAB, *version 7.10.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc., 2010.

[24] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks," pp. 1–10, 2018. [Online]. Available: http://arxiv.org/abs/1803.10892

[25] A. Rudenko, W. Huang, L. Palmieri, K. O. Arras, and A. J. Lilienthal, "Atlas: a Benchmarking Tool for Human Motion Prediction Algorithms," in *RSS, Workshop on Social Robot Navigation*, 2021.

[26] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 261–268.