Gaussian Process Based Motion Pattern Recognition with Sequential Local Models

Mattias Tiger[1] and Fredrik Heintz[1]

***Abstract*— Conventional trajectory-based vehicular traffic analysis approaches work well in simple environments such as a single crossing but they do not scale to more structurally complex environments such as networks of interconnected crossings (e.g. urban road networks). Local trajectory models are necessary to cope with the multi-modality of such structures, which in turn introduces new challenges. These larger and more complex environments increase the occurrences of lack of motion and self-overlaps in observed trajectories which impose further challenges. In this paper we consider the problem of motion pattern recognition in the setting of sequential local motion pattern models. That is, classifying sub-trajectories from observed trajectories in accordance with which motion pattern that best explains it. We introduce a Gaussian process (GP) based modeling approach which outperforms the state-of-the-art GP based motion pattern approaches at this task. We investigate the impact of varying local model overlap and the length of the observed trajectory trace on the classification quality. We further show that introducing a pre-processing step filtering out stops from the training data significantly improves the classification performance. The approach is evaluated using real GPS position data from city buses driving in urban areas.**

I. INTRODUCTION

Typical trajectory analysis consists of activity recognition (classification), abnormality detection and prediction such as in vehicular trajectory analysis [1], pedestrian prediction [2] and maritime traffic classification [3]. Much of the focus in trajectory analysis of vehicular traffic has been on single intersections, using data acquired from stationary traffic monitoring cameras. Today sensor-rich vehicles and unmanned aerial vehicle surveillance produce longer trajectories.

GPs have been used extensively in computer vision and robotics, and also for trajectory analysis. Good results have been shown on recognition, prediction and abnormality detection of vehicle [4][5], pedestrian [2] and marine vessel [3] motion patterns. Several challenges arise from scaling up trajectory analysis from single intersections to multiple intersections and larger road networks, with relevance outside of the vehicular traffic behavior analysis domain.

The state-of-the-art GP-based methods learn individual motion patterns from each input lane to each output lane in a crossing. However, the potential number of paths though an urban road network grows rapidly with the size of the road network, and the number of commonly occurring motion patterns even more. There is also the problem of self-overlapping trajectories which arise from multiple intersections, roundabouts and turning spaces. Both issues necessitate the use of sequential local trajectory models. For example to model individual crossings and connect them in a serial way, or to make sure that the vector field of motion velocity is not averaged erroneously due to self-overlap. Fig. 1 illustrates these issues where the red sub-trajectories cannot belong to the same motion pattern (trajectory model).

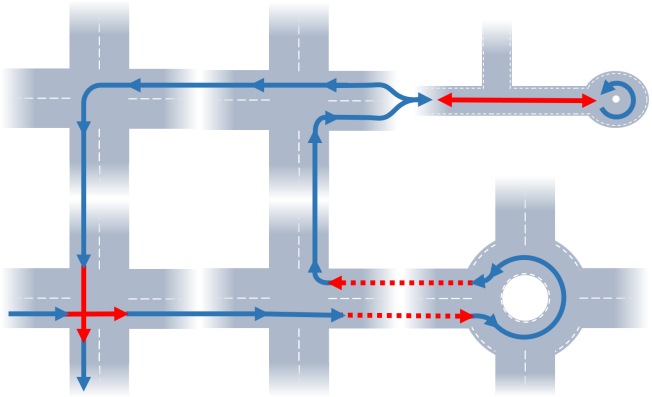

Fig. 1
Example: A long trajectory through a complex road structure. Problematic self-overlap is indicated in red. Self-overlap between lanes can occur when precision is low (dashed red).

In this paper we focus on the recognition (classification) task were observations are given and are to be associated to specific motion models (the classes). We introduce a novel motion pattern model which extends and outperform the state-of-the-art GP-based model for this task when using sequential local motion models (one class per local model). The data likelihood used for classification is commonly used in a similar manner for abnormality detection, and the result is therefore also related to the possibility of accurate abnormality detection. We further show that suppressing stops in the training data significantly improves the classification performance for both the conventional and proposed model.

Finally, the proposed model also allows us to explicitly model the probabilistic spatial extents of the roads or paths. Lane centerlines can be used for transferring motion pattern models to other intersections by spatial normalization [5].

II. RELATED WORK

The two most common approaches to modeling trajectories for trajectory analysis are using Hidden Markov models (HMM) with Gaussian Mixture Models (GMM) [1] and Gaussian process (GP) [2][3][4][5][6]. In the HMM-GMM approaches the HMM is used to model spatio-temporal

[1]Mattias Tiger and Fredrik Heintz are with the Department of Computer and Information Science, Linköping University, Sweden. mattias.tiger@liu.se, fredrik.heintz@liu.se

relations (the dynamics) and GMM is used for generalizing the observed motion pattern in the state space. HMM-GMM approaches are used for vehicular traffic based trajectory analysis such as in [7][1] but they are less suitable for modeling spatial extents (e.g. road or path) and they tend to have complex structures with many hard to tune parameters. The Gaussian process is a Bayesian non-parametric model which is very suitable for modeling trajectories and trajectory-based motion patterns. We focus on GPs since they provide a highly flexible nonlinear model with only a few hyper-parameters.

A velocity field, or *flow field*, approach where GPs are used to map directly from position to velocity ($(p_x, p_y) \rightarrow v_x, v_y$) is a common GP based trajectory analysis modeling approach and used in the work of [4][5][2][6]. In [4][5][2] they learn a mixture of GPs (MoGP) where each GP represent a motion pattern. Recognition is finding which GP that best explains the observed data. Abnormalities are found by a lack of significant data likelihood of the observations and step-ahead predictions are made using for example a particle filter with the velocity field as a state transition model [5]. All these approaches have severe problems with self-overlapping trajectories when learning. The overlapping area and its neighborhood will have a vector field that is averaged over the opposing directions resulting in nonsense. They do however work well in confined regions without self-overlaps.

In [4] they additionally map from normalized time to time velocity ($(p_x, p_y, t) \rightarrow v_x, v_y, v_t$) and can thereby handle stops in the middle of their models, such as stopping at a red light before continuing. The cost of time as explicit input is however that they can only model trajectories from entry to exit point with a single GP, making the approach incompatible with sequential local trajectory models.

In [2] they use change point detection to do abnormality detection in order to update their MoGP online with new observed motion patterns. In this paper we focus on cases where trajectories have to be segmented into local GP models, e.g. due to self-overlaps, since the models used would break otherwise. Change point detection is one suitable way to segment trajectories into sequential local GP models.

Many different kinds of predictions are in addition considered in motion pattern based trajectory analysis such as step-ahead trajectory prediction, prediction of the class of motion pattern given a non-complete trajectory [4] and prediction of future sequences of motion patterns [8]. The scope of this paper is limited to classification problem of an observed trajectory in the context of sequential motion pattern models.

Past work on sequential local models have considered some of the benefits in the vehicular traffic behavior [9] and maritime [10] domains. They do however primarily focus on the trajectory clustering problem and do not use GPs.

III. Gaussian Process Modeling

To model trajectories we consider the regression model

$$y = f(x) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_n^2) \tag{1}$$

where y and x are vectors, ϵ is Normal distributed noise with zero mean and variance σ_n^2 and f is a continuous function. One usually seeks a good estimate of f in order to make accurate predictions of the value y of the function for other arguments x than those already observed. However, in this work we want a distribution over functions representing the possible trajectories belonging to a certain motion pattern. If a velocity field is learned from many observed trajectories we want to represent not just the average velocity at each position but also the variance in velocity at each position.

The Gaussian process is a distribution over functions [11] and is completely described by its mean function $m(x)$ and covariance function $k(x_1, x_2)$ such that for a vector of inputs $\bar{x}$ the outputs $\bar{y}$ of the function are jointly Normal distributed

$$\bar{y} = f(\bar{x}) \sim \mathcal{N}(\mu(\bar{x}), \Sigma(\bar{x})) \tag{2}$$

with $\mathbb{E}(f(\bar{x})) = \mu(\bar{x}), \quad \mathbb{C}ov(f(\bar{x})) = \Sigma(\bar{x})$ and

$$\mu(\bar{x}) = m(\bar{x}) + \mathbf{K}(\bar{x}, \mathbf{x})\mathbf{V}^{-1}(\mathbf{y} - m(\mathbf{x}))^T, \tag{3}$$

$$\Sigma(\bar{x}) = \mathbf{K}(\bar{x}, \bar{x}) + \sigma_n^2\mathbf{I}_{\bar{x}} - \mathbf{K}(\bar{x}, \mathbf{x})\mathbf{V}^{-1}\mathbf{K}(\bar{x}, \mathbf{x})^T \tag{4}$$

where $\mathbf{K}$ is the gram matrix with entries $(\mathbf{K})_{ij} = k(x_i, x_j)$, $\mathbf{V} = [\mathbf{K}(\mathbf{x}, \mathbf{x}) + \sigma_n^2\mathbf{I}_x]$, and $\mathbf{x}$,$\mathbf{y}$ are vectors of input and output training data. $\mathbf{I}_{\bar{x}}$ and $\mathbf{I}_x$ are identity matrices of appropriate size. We make the common assumption that each output dimension is independent and we model each output dimension as a separate function $f : \mathbb{R}^D \rightarrow \mathbb{R}$ with a Gaussian process model $f \sim \mathcal{GP}(m(x), k(x_1, x_2))$.

For our applications we consider the zero mean function $m(x) = 0$ and the Squared Exponential covariance function

$$k(x_1, x_2) = \sigma_f^2 e^{(-\frac{1}{2}(x_1 - x_2)\Lambda(x_1 - x_2)^T)}, \tag{5}$$

where Λ is a diagonal matrix with length scales for each input dimension and σ_f^2 is the signal variance. Together with the noise variance σ_n^2 these are the hyper parameters $\theta = \{\sigma_n^2, \sigma_f^2, \Lambda\}$. We estimate the hyper-parameters from the data by maximizing the marginal log likelihood,

$$\log p(\mathbf{y}|\mathbf{x}, \theta) = -\frac{1}{2}\mathbf{y}\mathbf{V}^{-1}\mathbf{y}^T - \frac{1}{2}\log|\mathbf{V}| + C, \tag{6}$$

using Conjugate Gradient, where C is a constant. The problem is non-convex and we consequently do random restarts.

Local Gaussian processes [12][13][14] are GPs that cover different areas in the state space. Overlap at the borders between local GPs is often important for the GPs to retain the desired shape close to the border and it is achieved by letting neighboring local GPs share data points. We use local GPs in this work to model each sequential local trajectory models.

IV. The Flow Field Motion Pattern Model

The flow field approach of [4][5][2] consists of a mapping from 2D position to 2D velocity ($(p_x^*, p_y^*) \rightarrow v_x, v_y$) where v_x and v_y are assumed to be independent. The flow field model $\mathcal{M}^A = \langle f_{v_x}, f_{v_y} \rangle$ consists of two GP function models

$$\begin{bmatrix} v_x & v_y \end{bmatrix} = \begin{bmatrix} f_{v_x}(p_x^*, p_y^*) & f_{v_y}(p_x^*, p_y^*) \end{bmatrix} = flow(p_x^*, p_y^*) \tag{7}$$

where $p_x^*, p_y^* \in \mathbb{R}$ and from equation (3-4) we have

$$v_x \sim \mathcal{N}\big(\mu_{v_x}([p_x^* \;\; p_y^*]),\ \Sigma_{v_x}([p_x^* \;\; p_y^*])\big), \tag{8}$$

$$v_y \sim \mathcal{N}\big(\mu_{v_y}([p_x^* \;\; p_y^*]),\ \Sigma_{v_y}([p_x^* \;\; p_y^*])\big). \tag{9}$$

This approach to modeling motion patterns works well in localized contexts such as an intersection [4][5][2]. The motion patterns as a group have clear starts and stops, e.g. where the camera view of the road (or lane) begins and ends. The motion patterns are to a large extent overlapping and the velocity component vary significantly between different motion patterns which is how they are efficiently discriminated.

V. The Proposed Motion Pattern Model

A typical crossing can be represented by a directed acyclic graph (DAG), where all edges are from a source node to a sink node. The motion under this topology can be represented by flow fields [4][5][2] or as a DAG of serially connected motion models [9][10][8]. The latter is useful if a segmentation of serially connected distinct motion patterns is desired. We are interested in modeling more complex traffic structures than individual crossings, in which serially connected motion pattern models are required.

Urban roads may consist of a large number of connected crossings and roundabouts, and cyclic directed graph topologies form a core part of our road networks. Vehicles may drive in circles, perform U-turns on multi-lane roads with too low position accuracy to tell different lanes apart (e.g. due to radio shadow) or drive back on the same single lane road. These cases cause trajectories to self-overlap.

Local serial models are necessary to capture all possible paths because of the possibility to drive in circles. Local serial models are also important for reducing the total number of necessary motion pattern models, because the number of paths between each road network start point (source) and end point (sink) blows up quickly with additional crossings and roundabouts. We seek a motion pattern model suitable for serially connectivity and not just for parallel overlap.

Serially connected motion patterns require management of where one model stops and another starts. The flow field model has no clear start nor stop since it is a fully 2D vector field. Its omni-directional influence away from the data points interferes with adjacent serially connected motion patterns.

Our proposed approach is a mapping from 2D position to 1D normalized temporal parametrization to 4D state $((p_x^*, p_y^*) \rightarrow \tau \rightarrow p_x, p_y, v_x, v_y)$. The proposed model enables us to do three critical things. Firstly it allows us to describe the progression of the state (p_x, p_y, v_x, v_y) from the beginning of the motion pattern ($\tau = 0.0$) to the its end ($\tau = 1.0$) using the mapping $\tau \rightarrow p_x, p_y, v_x, v_y$. It represents the spatial extent of the motion pattern, i.e. the road/path and where we spatially expect new observations to be.

Secondly it allows us to retrieve the progression (τ) of the motion pattern from a spatial position (p_x^*, p_y^*) using the mapping $(p_x^*, p_y^*) \rightarrow \tau$. It allows us to calculate the progress of an object along the completion of a motion pattern and to use (10-11) to get a prior on the future state of the object.

Thirdly it allows us to detect if an observation is within the range of the motion pattern model ($\tau \in [0.0\ 1.0]$), before it ($\tau < 0.0$) or after it ($\tau > 1.0$). This enables us to modify τ in a suitable manner when it falls outside of this range

We will refer to our approach as the *inverse mapping* approach and its model $\mathcal{M}^B = \langle f_{p_x}, f_{p_y}, f_{v_x}, f_{v_y}, f_\tau \rangle$. consists of five GP function models

$$\begin{bmatrix} p_x & p_y \end{bmatrix} = \begin{bmatrix} f_{p_x}(\tau^*) & f_{p_y}(\tau^*) \end{bmatrix}, \tag{10}$$

$$\begin{bmatrix} v_x & v_y \end{bmatrix} = \begin{bmatrix} f_{v_x}(\tau^*) & f_{v_y}(\tau^*) \end{bmatrix}, \tag{11}$$

$$\tau = f_\tau(p_x^*, p_y^*), \tag{12}$$

where $p_x^*, p_y^* \in \mathbb{R}$ and from equation (3-4) we have

$$\tau \sim \mathcal{N}\big(\mu_\tau([p_x^*\ \ p_y^*]),\ \Sigma_\tau([p_x^*\ \ p_y^*])\big), \tag{13}$$

$$p_x \sim \mathcal{N}\big(\mu_{p_x}(\tau^*),\ \Sigma_{p_x}(\tau^*)\big), \tag{14}$$

$$p_y \sim \mathcal{N}\big(\mu_{p_y}(\tau^*),\ \Sigma_{p_y}(\tau^*)\big), \tag{15}$$

$$v_x \sim \mathcal{N}\big(\mu_{v_x}(\tau^*),\ \Sigma_{v_x}(\tau^*)\big), \tag{16}$$

$$v_y \sim \mathcal{N}\big(\mu_{v_y}(\tau^*),\ \Sigma_{v_y}(\tau^*)\big). \tag{17}$$

In this work we make use of an approximation of the above model where we use the maximum a posteriori (MAP) of τ, $\tau^* = \mu_\tau$, discarding the covariance. A more accurate approximation is possible [15] but we get a sufficiently good estimation of τ from 2D positions close to the model.

The purpose of the inverse mapping (12) is to approximate a projection from any 2D position in the world onto the closest point on the mean function (3) of the trajectory model in (10-11). This is done in time linear to the number of data points of the model and estimates the non-linear projection.

The inverse mapping requires the trajectory to not be self-overlapping (self-intersecting) which means that the mapping (10) has to be bijective. We have used an inverse mapping in previous preliminary work to classify tracked object's motion patterns as well as to compare the similarity and overlap between different learned motion patterns [8].

The Gaussian process models are learned by finding the hyper parameters θ which are the maximizers of the data likelihood (18). The observations are 2D positions and 2D velocities together with a time stamp. In this work we estimate the velocities from position observations using a time varying Kalman filter, since velocities are not directly observed in the general case. The function models $f_\square$ in respectively $\mathcal{M}^A$ and $\mathcal{M}^B$ are optimized independently.

An example of a modeled motion pattern using artificial data is shown in Fig. 2 with both the conventional approach, $\mathcal{M}^A$, (top) and the proposed approach, $\mathcal{M}^B$, (bottom).

VI. Classification of Observed Motion Pattern

We assume that we have K motion pattern models, $\{\mathcal{M}_1, \ldots, \mathcal{M}_K\}$, of the same class and consisting of the same number of data points. Given a matrix of observations, $\bar{z}$, (from a trajectory or sub-trajectory) where each row is the observed state at that time point and each column is one state dimension, $\bar{z} = [\bar{p}_x^*\ \ \bar{p}_y^*\ \ \bar{v}_x^*\ \ \bar{v}_y^*]$, we want to classify $\bar{z}$. The classification of an observation or sequence of observations is calculated by comparing the relative likelihood of each model and selecting the model with the highest data likelihood as the class $c = \arg\max_k P(\bar{z}|\mathcal{M}_k)$ of observation $\bar{z}$. This is the core of what is done in the related work [4][5][2].

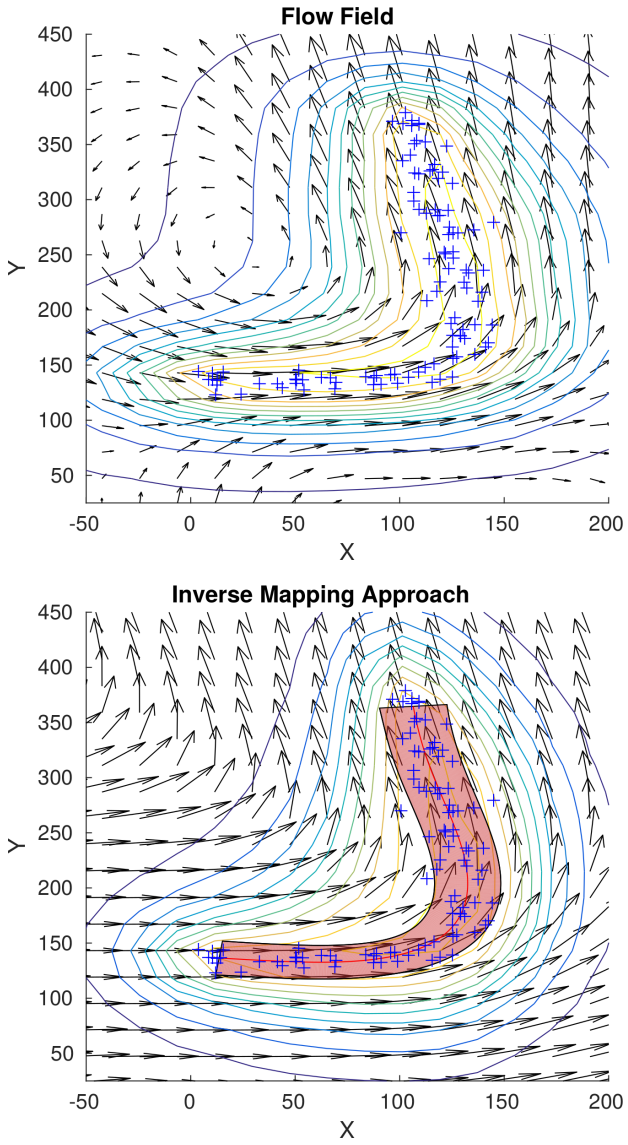

Fig. 2
Velocity field for the Flow field approach $\mathcal{M}^A$ (**TOP**) and for the proposed Inverse mapping approach $\mathcal{M}^B$ (**BOTTOM**). Data points (position and velocity in 2D) as +, mean predictive function (3) as a red line and 95% probability interval of the mean predictive variance (4) as a red envelope. A contour (yellow to blue) is also shown of the predictive variance.

The data log likelihood of a set of observed data points $\bar{x}, \bar{y}$ given a GP model is

$$\log p(\bar{y}|\bar{x}, \mathbf{y}, \mathbf{x}, \theta) = -\frac{1}{2}\big(\bar{y} - \mu(\bar{x})\big)\big[\Sigma(\bar{x})\big]^{-1}\big(\bar{y} - \mu(\bar{x})\big)^T - \frac{1}{2}\log|\Sigma(\bar{x})| + C \quad (18)$$

where $\mu(\bar{x})$ and $\Sigma(\bar{x})$ are the predictive mean function (3) and predictive covariance function (4) of the GP.

The Flow field data log likelihood for some observed data point or sequence of observed data points $\bar{z}$ is

$$P_A(\bar{z}|\mathcal{M}_k^A) = \log p(\bar{v}_x^*|[\bar{p}_x^*, \bar{p}_y^*], \mathcal{M}_k^A) + \log p(\bar{v}_y^*|[\bar{p}_x^*, \bar{p}_y^*], \mathcal{M}_k^A) \quad (19)$$

due to the independence assumption between f_{v_x} and f_{v_y}. With the same available input we can derive three variations of the data log likelihood for the inverse mapping approach. For $(p_x^*, p_y^*) \rightarrow \tau \rightarrow v_x, v_y$ we have P_{B_v}, for $(p_x^*, p_y^*) \rightarrow \tau \rightarrow p_x, p_y$ we have P_{B_p} and the combination of the two P_B with $(p_x^*, p_y^*) \rightarrow \tau \rightarrow p_x, p_y, v_x, v_y$, as

$$P_{B_v}(\bar{z}|\mathcal{M}_k^B) = \log p(\bar{v}_x^*|g([\bar{p}_x^*\ \bar{p}_y^*]), \mathcal{M}_k^B) + \log p(\bar{v}_y^*|g([\bar{p}_x^*\ \bar{p}_y^*]), \mathcal{M}_k^B), \quad (20)$$

$$P_{B_p}(\bar{z}|\mathcal{M}_k^B) = \log p(\bar{p}_x^*|g([\bar{p}_x^*\ \bar{p}_y^*]), \mathcal{M}_k^B) + \log p(\bar{p}_y^*|g([\bar{p}_x^*\ \bar{p}_y^*]), \mathcal{M}_k^B), \quad (21)$$

$$P_B(\bar{z}|\mathcal{M}_k^B) = P_{B_v}(\bar{z}|\mathcal{M}_k^B) + P_{B_p}(\bar{z}|\mathcal{M}_k^B), \quad (22)$$

where the function g is used to calculate the deterministic τ. Three variations of g has been investigated and these are

$$g_1([\bar{p}_x^*\ \bar{p}_y^*]) = \mu_\tau([\bar{p}_x^*\ \bar{p}_y^*]) \quad (23)$$

$$g_2([\bar{p}_x^*\ \bar{p}_y^*]) = clamp(\mu_\tau([\bar{p}_x^*\ \bar{p}_y^*]),\ 0.0,\ 1.0) \quad (24)$$

$$g_3([\bar{p}_x^*\ \bar{p}_y^*]) = mirror(\mu_\tau([\bar{p}_x^*\ \bar{p}_y^*]),\ 0.0,\ 1.0) \quad (25)$$

The second, g_2, forces τ to be between $[0.0\ 1.0]$ and thereby keeping τ within the restricted range of the motion pattern model. The third, g_3, transforms τ whenever it is outside of $[0.0\ 1.0]$ by mirroring the value through the boundary 0.0 or 1.0 depending if τ is smaller or larger than 0.0 back towards the center 0.5. As a consequence, reducing the border similarity further than g_2 between motion pattern models that are sequential neighbors. This approach works best for the recognition task and is used in the experiments.

VII. Handling Stop by Stop Compression

To detect stops we estimate the 2D velocity at each state observation via Bayesian filtering. We do this by applying a time-varying Kalman filter with a constant-velocity motion model to the sequence of observations and their timestamps. The filter needs to be time-varying to handle the variation in duration between observed positions. The constant-velocity motion model assumes that the observed positions are noisy with normal-distributed noise. Data points corresponding to stops are found by calculating if the 95% probability density of the velocity in both x and y dimensions are simultaneously containing velocity zero. A compact sequence of data points that are stops, $\mathbf{y}_{i:i+k}$, are summarized by a single data point, $\mathbf{y}_n$, such that the full new sequence $\mathbf{y} = [\mathbf{y}_{1:i-1}, \mathbf{y}_n, \mathbf{y}_{i+k+1:\ldots}]$ with timestamps $\mathbf{x}$ are defined by

$$\mathbf{y}_n = \text{average}(\mathbf{y}_{i:i+k}), \quad \mathbf{x}_n = \mathbf{x}_i, \quad \mathbf{x}_{i+k+1:\ldots} = \mathbf{x}_{i+k+1:\ldots} - (\mathbf{x}_{i+k} - \mathbf{x}_i) \quad (26)$$

This is illustrated in Fig. 3. Reducing the stop data points to a single data point removes the effects of noise the GP has trouble suppressing. The reason for this is that the stop data points are very much aligned in a straight line, while

surrounded by steep curvatures, which cause the SE kernel (5) to produce ripple waves when trying to fit. It is to a large extent reduced with the collapsing of adjacent stop data points but still present. The distinct plateau with its long and mostly flat shape surrounding the stop data point still allow for a soft curve on either side of it (such as a sinusoidal in the example case in Fig. 3). Ripple in form of such curves is penalized by the data log likelihood (6) if the distance of the stop data point to its adjacent data points is reduced.

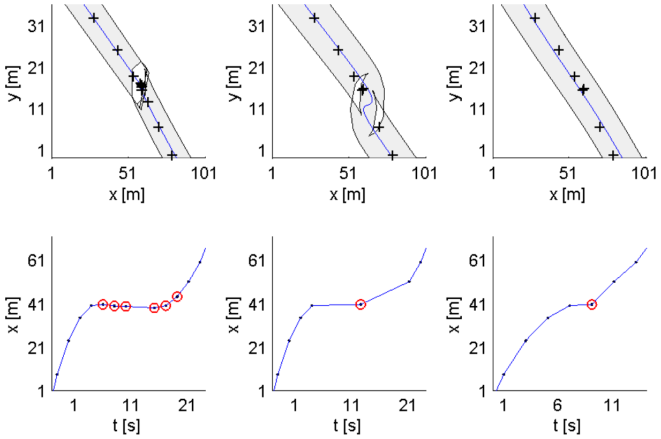

Fig. 3
Example of stop compression. **Top row:** Learned GP of 2D-position (10) over black data (+). **Bottom row:** x-position over time. Red circles are stops. Left, middle and right column shows respectively *no stop handling*, *stop removal but no time adjustment* and *stop removal with time adjustment*.

VIII. Handling Self-overlaps

Assume that we know the noise-free latent function f of the long trajectory that we want to model. A self-overlap occurs where f intersects or overlaps itself. If we have a noise-free vector representing f with resolution $\theta_{distance}$ between subsequent positions on f, then it is possible to find self-overlaps of f down to the resolution limit by pairwise comparing the distance between vector elements. All self-overlaps in f can be found if this limit is set sufficiently low given the application. In terms of vehicular traffic, the distance between lanes or between roads might be sufficient depending on the accuracy of the positioning of the vehicle. We do not know f but we can estimate a smooth approximation f^{est} and calculate a noise-free vector approximating f^{est} with resolution $\theta_{distance}$.

f^{est} is estimated by the mean predictive function (3) of a GP fitted on the entire long trajectory. The length of f^{est} is estimated by first evaluating (3) on a high resolution equidistant time point vector, with beginning and ending inputs that correspond to the long trajectory. The length is then sufficiently approximated by the Riemann sum of the resulting high-resolution vector. The GP length allow us to calculate the necessary number of evaluations, $M = GP_{length}/(0.5 \times \theta_{distance})$, of f^{est} in order to get a vector with resolution $\theta_{distance}$. The vector $[p_1, \ldots, p_M]$ is

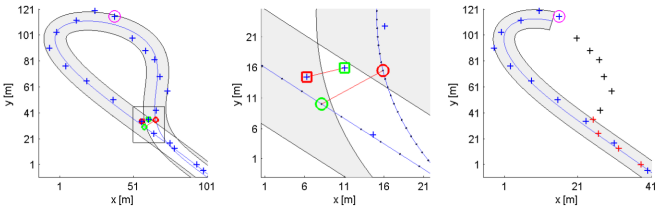

Fig. 4
Example of detecting and handling self-overlap. The red and green circles are the self-overlap points of the interpolated trajectory (black dots). The red and green squares are the closest points (in time) of self-overlap in the observed data (blue +). The Magenta circle highlights the split-point. The maximum overlap on either side of the split-point is 6 data points. The right figure shows the learned GP of the first half of the self-overlap region with blue pluses as own data points, black pluses as local model overlaping with the next model and red pluses as the self-overlap point and data points after it. Without the overlap the GP would not capture the full curve of the road. **Middle:** magnification of the left figure.

consequently calculated by (3) of the GP at M equidistant time points and the vector allow us to calculate self-overlaps.

The purpose of detecting any self-overlaps is to make sure that a model is not learned with the data of a sub-trajectory with self-overlap. The average of an overlapping velocity field is likely to be a very bad representation of the motion pattern. Since this constraint concerns all the data used to learn the model it also affects the overlap between the subsequent models. However, if the number of overlapping data points are too few there is a risk that the local models will be incapable of capturing the shape (curvature) of the trajectory near the borders. To minimize this problem we split the trajectory into two local models in a way that maximize the number of sharable data points. This is done by finding a split-point in the middle (in terms of number of data points) between the end and start of the self-overlap. See Fig. 4.

IX. Results and Evaluation

We have evaluated the presented approach on a data set with noisy GPS positions of city buses driving through urban areas. The data has a variable rate of 2-4 hz with sporadic gaps. The data is heavily anonymized to protect the drivers so we have no road maps. The data set consists of 17124 data points (2D position and time) of active bus driving of a total duration of 11.3 hours and a distance of 181 km.

The data is divided into 25 long trajectories, ranging from 5 to 60 minutes and 2 to 18 km. Each trajectory is pre-processed by a Kalman filter to estimate 2D velocities. Each long trajectory is treated separately. First all stops are suppressed (VII) with $\theta_{distance} = 10m$, then self-overlaps are detected (VIII) and the first necessary split-points of the long trajectory are found. A total of 204 stops and 14 self-overlaps were found in among the long trajectories.

The trajectories are further segmented to divide the data as equally as possible between local models. The data log

likelihood (18) penalizes a GP with fewer data points less than with more. There are various ways to achieve this in the conventional setup [2][4][5]. We also want to have many local models in order to evaluate how well the two modeling approaches work given a large number of sequential local motion patterns. The data set is very sparse (long spatial distances between samples) so we segment the remaining sub-trajectories in a greedy way with about 19 data points per local model (capturing 1-1.5 km per local model which is quite far in an urban environment). We use a local model overlap of 5, which we found by empirical experiments described further down. The two modeling approaches ($\mathcal{M}^A$ and $\mathcal{M}^B$) are applied to the data of each sub-trajectory to learn two different kinds of local models for comparison.

Since we do not do any clustering nor aggregation of similar motion patterns in this paper, there is a large risk of over-fitting the models as we only use a single observation (a single trajectory) as training data per trajectory model. This result in too low variance (4) of the model, making it very unlikely for the model to best explain (among the available models) a new observed trajectory that is close to the model but not exactly overlapping and aligned. We therefore add additional prior knowledge by modifying their covariance so that all models have at least a minimum variance (σ_n^2) of 2^2 meters (position mappings) and 2^2 km/h (velocity mappings). We do not expect the GPS to be more accurate than so. If the data uncertainty is larger we do not do any adjustments.

To evaluate the learned models we randomize new observations by first fitting a GP over the entire long trajectory. 5 trajectories are drawn from the GP and to each of these trajectories we add iid Gaussian noise with standard deviation 2 meters for the x and y component for each data point. The final evaluation score is averaged over the 5 draws.

Each long trajectory gives rise to a different number of local models, since they have very different length. The classification problem is that of a multi-class classifier, one for each of the four compared approaches (P_A, P_{B_v}, P_{B_p}, P_B) and with a different number of classes per evaluated data set (i.e. per long trajectory). To be able to compare and aggregate the accuracy of these it is necessary to adjust the accuracy first in a manner that reduces the impact of the different number of classes. One such statistic is Cohen's Kappa [16]. The Cohen's Kappa statistic is defined as

$$\text{Kappa} = \frac{Accuracy - \mathbb{E}[Accuracy]}{1 - \mathbb{E}[Accuracy]}, \tag{27}$$

$$Accuracy = \frac{1}{M}\sum_{m=1}^{M} \text{TP}(m), \quad \mathbb{E}[Accuracy] = \sum_{c=1}^{C} \frac{M_c}{M}\frac{\text{TP}_c}{M}$$

where M is the number of test data points and $\text{TP}(m) = 1$ if the classification of test point m was correct and 0 otherwise. TP_c is the number of correct classifications of class c (true positives). M_c is the frequency of class c in the test data set.

The test data is generated from the same data as the learned models so we know which model should be responsible for which part of the training data. To investigate the effect of the local model overlap (adjacent local models share some data points) we also used a ground truth which allowed two classes to be correct in the intervals of local model overlap. The strict ground truth and the less strict ground truth is called **Exact** and **Overlap** respectively in Fig. 5, and shown as $(b), (d)$ and $(a), (c)$ respectively in Fig. 6.

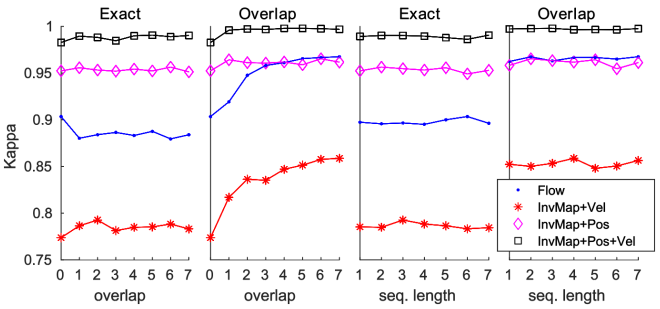

Fig. 5
Evaluation of changing no. of data points in overlap and changing sequence length used ($\bar{z}$). **Exact** require correct model, **Overlap** accept either of two overlapping models. Using the first 9 of the 25 trajectories.

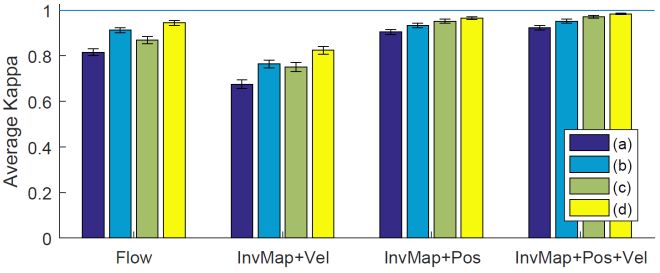

Fig. 6
(a,b) Without handling stops and self-overlaps. (b,d) Accept either of 2 overlapping models as correct classification. (a,c) Require correct model.

We evaluated the difference in classification performance when varying the length of observation sequences (*seq. length* in Fig. 5) at a time, from one data point to a vector of 7 data points ($\bar{z}$ in VI). We also varied the no. of data points for local model overlap with each sequence length. All four model types are practically invariant to the sequence length. This is very different from the setting in [4][5][2] where high classification accuracy usually requires a long trace. Here the motion pattern models are not largely overlapping (apart from at the edge of each model). Also, the use of sequential local models makes a longer sequence more likely to overlap two neighboring sequential local models. The sequence length 1 is used in the larger experiment.

The overlap number do affect the the classification accuracy significantly. Only the flow field (P_A) have higher accuracy with zero overlap, with a clear maximum together with sequence length but all model variations have higher accuracy for **Overlap**. The flow field variation (P_A) and the velocity based inverse mapping variation (P_{B_v}) increase significantly in accuracy with an increase in local model overlap. For the flow field variation there is a compromise

between having a high accuracy for **Exact** and for **Overlap**. We selected local model overlap 5 for the larger experiment.

Using the full data set we evaluated the classification performance of the flow field approach and the inverse mapping approach, and its three variations. The results are shown in Fig. 6. P_B has the highest accuracy and improves the accuracy over the baseline P_A by $\sim$10% for **Exact** and $\sim$5% for **Overlap** when handling stops and self-overlaps. Handling self-overlaps and suppressing stops in the training data has a large impact on all four approaches, increasing the accuracy by $\sim$5% for **Exact** and $\sim$3% for **Overlap**.

X. CONCLUSIONS

There exist many challenges for trajectory analysis on larger road networks with complex structures. We have in this paper considered some of these challenges and provided partial solutions to the motion pattern recognition problem for Gaussian process based sequential local models. We have proposed a novel motion pattern model which goes beyond the traditional flow field approach and incorporates spatial locality together with the discriminatory power of a velocity field. It is shown empirically to out-perform the flow field approach in a setting of sequential local motion pattern models. This is due to more discriminative power and specifically higher accuracy close to the borders between models. Varying the local model overlap significantly affects the velocity field approaches but not so much those using spatial locality. Handling self-overlaps and suppressing stops significantly increases the accuracy.

REFERENCES

[1] B. T. Morris and M. M. Trivedi, "Understanding vehicular traffic behavior from video: a survey of unsupervised approaches," *Journal of Electronic Imaging*, vol. 22, no. 4, pp. 041–113, 2013.

[2] S. Ferguson, B. Luders, R. C. Grande, and J. P. How, *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*. Springer International Publishing, 2015, ch. Real-Time Predictive Modeling and Robust Avoidance of Pedestrians with Uncertain, Changing Intentions.

[3] M. Smith, S. Reece, I. Rezek, I. Psorakis, and S. Roberts, "Maritime abnormality detection using gaussian processes," *Knowledge and Information Systems*, pp. 1–26, 2013.

[4] K. Kim, D. Lee, and I. Essa, "Gaussian process regression flow for analysis of motion trajectories," in *Proc. ICCV*, 2011.

[5] Q. Tran and J. Firl, "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression," in *IEEE Intelligent Vehicles Symposium*, June 2014.

[6] S. T. O'Callaghan, S. P. N. Singh, A. Alempijevic, and F. T. Ramos, "Learning navigational maps by observing human motion patterns," in *2011 IEEE International Conference on Robotics and Automation*.

[7] B. Morris and M. Trivedi, "Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[8] M. Tiger and F. Heintz, "Towards unsupervised learning, classification and prediction of activities in a stream-based framework," in *Proc. Scandinavian Conference on Artificial Intelligence (SCAI)*, 2015.

[9] L. Snidaro, C. Piciarelli, and G. L. Foresti, "Fusion of trajectory clusters for situation assessment," in *Proc. FUSION*, July 2006.

[10] N. Guillarme and X. Lerouvreur, "Unsupervised extraction of knowledge from S-AIS data for maritime situational awareness." in *Proc. FUSION*, 2013.

[11] C. E. Rasmussen, "Gaussian processes for machine learning." MIT Press, 2006.

[12] F. Meier, P. Hennig, and S. Schaal, "Incremental local gaussian regression," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 972–980.

[13] D. Nguyen-Tuong and J. Peters, "Local gaussian process regression for real-time model-based robot control," in *Proc. IROS*, 2008.

[14] M. Schneider and W. Ertel, "Robot learning by demonstration with local gaussian process regression." in *Proc. IROS*, 2010.

[15] A. Mchutchon and C. E. Rasmussen, "Gaussian process training with input noise," in *Advances in Neural Information Processing Systems 24*. Curran Associates, Inc., 2011, pp. 1341–1349.

[16] J. Cohen, "A Coefficient of Agreement for Nominal Scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46.

ACKNOWLEDGMENT

This work is partially supported by grants from the National Graduate School in Computer Science, Sweden (CUGS), the Swedish Research Council (VR) Linnaeus Center CADICS, ELLIIT Excellence Center at Linkping-Lund for Information Technology, and partially supported by Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. We would also like to thank Halmstad University for providing the bus data set.