Linköping University | Department of Computer and Information Science
Master's thesis, 30 ECTS | Datateknik
202019 | LIU-IDA/LITH-EX-A--2019/001--SE

Trajectory-based Arrival Time Prediction using Gaussian Processes

-

Trajektoriebaserad ankomsttidsprediktion med Gaussiska Processer

Sebastian Callh

Supervisor : Mattias Tiger
Examiner : Fredrik Heintz

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

Abstract

`Abstract.tex`

Acknowledgments

`Acknowledgments.tex`

Contents

Abstract **iii**

Acknowledgments **iv**

Contents **v**

List of Figures **vi**

1 Introduction **1**
1.1 Problem Description 2
1.2 Aim 3
1.3 Research questions 3
1.4 Delimitations 3
1.5 Report Outline 3

2 Background **4**
2.1 Trajectory Data 4
2.2 Observed Machine Learning 4
2.3 Gaussian Processes 4
2.4 Kernel as Priors 5
2.5 Related Work 6
2.6 Arrival Time Prediction 6
2.7 Motion Pattern Learning 6
2.8 Event Detection 8

3 Data **9**

4 Method **10**
4.1 System Description 10
4.2 Trajectory Comparison 10

5 Results **14**

6 Discussion **15**
6.1 Results 15
6.2 Method 15
6.3 The work in a wider context 16

7 Conclusion **17**

Bibliography **18**

List of Figures

1.1 An illustration of the data flow in a system that can make arrival time predictions and detect events in motion patterns. 2

2.1 Synthetic data showing two motion patterns with two trajectories in each. Trajectory 2 and 3 belong to one motion pattern and Trajectory 1 and 4 belong to a second motion pattern. 6

4.1 An illustration of local distances from T_2 to T_1 as point-wise orthogonal projections. 10
4.2 Remaining distance for some route as a function of time for unsynchronised trajectories. The functions cannot be fully compared, since the they have different support $[0, a]$, $[0, b]$, $[0, c]$. 11
4.3 Remaining distance for some route as a function of time for synchronised trajectories. The motion patterns are comparable from start to end, since they all have support $[0, 1]$. 11
4.4 An illustration of the synchronisation model $\mathcal{M}^{(i)}_{synch}$, shown as the mappings of f_i and g_i. When synchronising a trajectory, it is mapped by $f_i \circ g_i$. 12
4.5 Spatial progression of a trajectory before data augmentation. 13
4.6 Spatial progression of a trajectory after data augmentation. 13
4.7 An illustration of the prediction model. When predicting the arrival time for a trajectory, it is mapped by $h_i \circ g_i$. 13

1 Introduction

As cities grow, efficient public transport systems are becoming increasingly important. To offer a more efficient service, public transport providers use systems that predict arrival times of buses, trains and similar vehicles, and present this information to the general public. The accuracy and reliability of these predictions are paramount, since many people depend on them, and erroneous predictions reflect badly on the public transport provider.

Machine learning algorithms have been applied with great promise to predict arrival times [5, 10, 9], and research is still ongoing. Modern approaches have seen heavy use of Recurrent Neural Networks (RNNs) to directly model arrival times from the current state of public transport vehicles, an approach which has shown very good prediction accuracy but has major drawbacks:

1. It does not quantify prediction uncertainties.
2. It is severely lacking in explainability.

To make good decisions based of model predictions, it is important to know the prediction uncertainty. Being able to explain how a model makes its predictions is also a extremely desirable, since it allows reasoning about why a model made a certain prediction, and about how a model would act in previously unseen scenarios. To address the first problem, a generative model can be used instead of RNNs. However, solving the second problem and achieving a satisfying level of explainability requires a much richer model of the data altogether. One promising approach to create such a model is *motion pattern learning* (or *trajectory learning*), where the goal is to learn a *motion pattern* (also known as *activity pattern* or *activity model*) which represents a cluster of individually observed trajectories. Motion pattern learning is a problem which spans several fields, such as computer vision [8, 23, 6, 1], pattern recognition [15], autonomous vehicles [3], and health informatics [11], and is like arrival time prediction an active area of research.

By combining motion pattern learning and generative modeling, it is possible to make sophisticated predictions, and gain a lot of insight on how the system performs. For instance, it would be possible to present the probability of arriving on time, in real-time, to passengers planning trips with several transits. The public transport provider could see routes where predictions are particularly inaccurate, and analyse the motion patterns on those routes, to help explain their poor performance. The motion patterns could be automatically analysed for detecting events of interest, such as “velocity change from 50 to 40 km/h”, “stop”, “queue”, “emergency break”, to help explain the bad predictions. They could also be processed for anomaly detection, to automatically detect if stations are missed or if public transport vehicles stray from the designated route, or stand still for a long time. Based on the information extracted, the public transport provider could make efforts to improve the routes by, for instance, moving routes, moving stops or changing time schedules. Motion patterns created after the changes have been made could then be compared to previous ones, to see what effects the changes had. By iterating this process, continuous improvements could be made.

1.1 Problem Description

This section addresses the problems that need to be solved to create a system that both learns motion patterns and makes arrival time predictions that quantifies uncertainty. It also proposes solutions to these problems. An illustration of the data flow of the system can be seen in Figure 1.1.

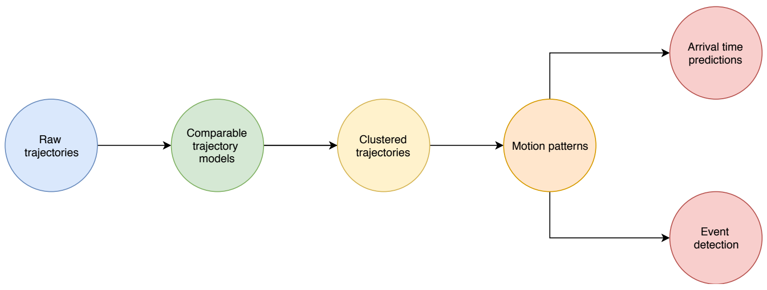


Figure 1.1: An illustration of the data flow in a system that can make arrival time predictions and detect events in motion patterns.

Trajectory Comparison

The first step in the system requires a way of comparing trajectories, which is not straightforward. The reason for this is that trajectories can have different lengths and unevenly spaced observations. The proposed way of doing this is by *synchronising* them, by modeling the trajectories spatial position as two independent GPs $f_x(\tau) \mapsto x$, $f_y(\tau) \mapsto y$ for $\tau = [0, 1]$, and learning an inverse GP $f(x, y) \mapsto \tau$. A similarity metric for model M_k is then given by the likelihood $P(u, v|M_k)$, where u and v are given by the function compositions $u = (f_y \circ g)(x, y)$, $v = (f_x \circ g)(x, y)$.

Trajectory Clustering

Given a similarity metric, trajectories can be clustered. Agglomerative clustering is proposed for this task. One difficult question with clustering is what value of similarity to consider as “close”. A lot of manual experiments would be needed to answer this, which could be very time consuming. Fortunately, the thesis project can proceed without properly solving this task, as clusters with a single trajectory can be considered.

Motion Pattern Learning

The challenge after clustering is to find a motion pattern model M_k that represents trajectories of cluster k well. This can be done by fitting a GP that maximises the likelihood of the cluster data, but also in several other more sophisticated ways, should time permit. For the case of a single trajectory in a cluster, this problem is reduced to ordinary Gaussian process regression (GPR).

Arrival Time Prediction

Once motion patterns have been learned, arrival time t can be predicted by modeling $P(t|x, y, M_k)$ for observed position (x, y). The proposed solution to this is one GPR model per motion pattern, combined in a mixture to form the final predictive distribution

$$P(t|x, y) = \sum_k P(t|x, y, M_k) P(M_k|x, y).$$

Event Detection

The final problem is automatically extracting events from motion patterns. This can be done by asserting a GP for the motion pattern, and searching over different kernel structures to maximise marginal likelihood. By searching over kernels which carry a specific meaning (in this case kernels would correspond to specific events), it is possible to detect events based on the kernel composition.

1.2 Aim

The aim of this thesis project is to model motion patterns of public transport vehicles using Gaussian Processes (GPs), and use these models to make arrival time predictions with an accuracy that is competitive to current state-of-the-art models [13, 4, 10]. Furthermore, the project aims to detect specific characteristics from motion patterns, such as “the driver had to emergency break”, or “the vehicles speed was very slow”, to see if certain characteristics are more common in motion patterns where the vehicle is too early or too late, compared to publicly available time schedules. Finally, this project aims to investigate how outlier motion patterns can be detected.

1.3 Research questions

1. How can Gaussian processes be used to capture trajectory motion patterns, such that arrival time predictions based on these patterns minimise MAE?
2. How can similar trajectories be clustered, such that ...?
3. How can specific events be automatically detected in a motion pattern?
4. How can outlier motion patterns be detected?

1.4 Delimitations

The project limits itself to arrival time predictions on consecutive but stops. Predictions for bus stops several steps ahead will consequently not be considered.

1.5 Report Outline

2 Background

This chapter describes theoretic results that this thesis is based on. Should the reader already be familiar with machine learning, and in particular Gaussian Processes, this chapter can be safely skipped.

2.1 Trajectory Data

Explain the difficulties of comparing them

2.2 Observed Machine Learning

Explain core concept Explain MAP

2.3 Gaussian Processes

A GP generalises a multivariate normal distribution, and can be seen as a distribution over functions, completely defined by its mean function $m(x)$ and covariance function $k(x, x', \theta)$ [12]. Here, x and x' are elements in the domain of modeled functions, and θ a vector of hyper-parameters for the covariance function. For any input vector x, the output y is assumed jointly normally distributed according to

$$y = f(x) \sim \mathcal{N}(\mu(x), \Sigma(x)) \tag{2.1}$$

where

$$\mu(x) = m(x) + K(x, \mathbf{x})\mathbf{V}^{-1}(y - m(x))^T, \tag{2.2}$$

$$\Sigma(x) = K(x, x) + \sigma_n^2\mathbf{I} - \mathbf{K}(x, \mathbf{x})\mathbf{V}^{-1}\mathbf{K}(x, \mathbf{x})^T, \tag{2.3}$$

and $\mathbf{K}$ is the gram matrix with elements $K_{ij} = k(x_i, x_j)$ and $\mathbf{V} = K(x, x) + \sigma_n^2 I$. The mean function $m(x)$ can be assumed to be $m(x) = 0$ without loss of generality, making the covariance function $k(x, x', \theta)$ the only free parameter. Picking a covariance function represents a prior belief on how values close in y are related, expressed in x. Training a GP is typically done using maximum likelihood estimation. That is, the GP parameters θ are optimised to maximise the data likelihood. Unfortunately, there is no expression for the data likelihood in closed form, so iterative methods have to be used. Because the likelihood function is non-convex, there is a high risk of finding local minimas, so random restarts are typically required to find a good θ.

2.4 Kernel as Priors

Talk about different kernels and what prior beliefs they represent

Gaussian Processes Regression

Structure Discovery Using Gaussian Processes

2.5 Related Work

This section covers related work on arrival time prediction and motion pattern modeling using spatio-temporal data. It also covers related work relevant to clustering of trajectory data and event detection in motion patterns.

2.6 Arrival Time Prediction

Arrival time prediction is, in a nutshell, the problem of answering the question "When does the bus arrive?". Formally, the goal is to learn a function $t = f(x)$ for arrival time t and some vector x, which contains information on the current state of the world. For instance, it could contain the position of the closest vehicle and the time of day. In recent years, machine learning techniques have been very successful at predicting the arrival time with small errors, and Long Short-Term Memory Networks (LSTMs) in particular have proven extremely effective. J. Pang et al. predicted bus arrival times to the next station in a continuous setting using LSTMs given the current position of a bus and static domain knowledge about its last stop [10]. D. Nguyen et al. also used LSTMs, but with entire trajectories [9]. They predicted arrival times of ships by training a model to generate continuations of trajectories, and when the model was presented with a new unfinished trajectory, it generated a probable continuation until it arrived at a port. This was then used to prediction both the destination and the arrival time.

While these approaches do perform admirably with respect to accuracy, they lack explainability and a way to quantify prediction uncertainty.

2.7 Motion Pattern Learning

Motion pattern learning is the problem of learning motion patterns from a set of trajectories, such that each pattern captures a different characteristic of the trajectories. An example with synthetic data can be seen in Figure 2.1. Note that the term *trajectory learning* is often used in the literature for the same problem, however, the term "trajectory" is ambiguous so in the context of this thesis the name motion pattern learning will be used.

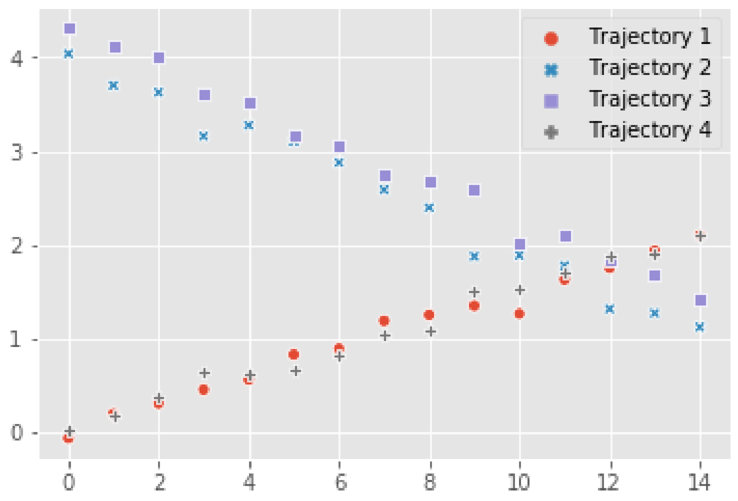


Figure 2.1: Synthetic data showing two motion patterns with two trajectories in each. Trajectory 2 and 3 belong to one motion pattern and Trajectory 1 and 4 belong to a second motion pattern.

Motion pattern learning has a natural interpretation as a clustering problem, but clustering trajectories is difficult, since different trajectories can have different lengths. This means that they do not exist in the same vector space and are not naturally comparable using similarity metrics based on Euclidan distance. Because of this, more advanced similarity metrics often need to be used.

Trajectory Similarity Metrics

There are several ways to construct a similarity metric for trajectories. Some widely-used metrics are Euclidan distance, Dynamic Time Warping (DTW), Longest Common Sub-Sequence (LCSS), and Edit Distance (ED) [21]. DTW and LCSS have seen use in motion pattern learning [15, 20], but suffer from high time-complexity, making them unsuitable for real time processing [23] An alternative to previously mentioned similarity metrics are probabilistic approaches, where models are fit to trajectories and similarity is measured as data likelihood [6, 19, 18]. One advantage of a probabilistic approach are that it is not limited by purely spatial information, since any information about the trajectories can be modeled. For instance, spatial position could be combined with velocity and acceleration.

Probabilistic Approaches

The probabilistic approach to trajectory learning assumes a model for each motion pattern and infers its parameters from data. This is incredibly similar to how probabilistic models are used to create similarity metrics, but instead this is done for a cluster of trajectories. Two popular approaches for this is GPs and hierarchical Bayesian models.

Gaussian Processes

GPs have been used in several domains for modeling trajectories. M. Pimentel et al. used GPs to model the vital signs of patients [11], and were successfully able to cluster the data using hierarchical clustering and the GP model likelihoods. They then modeled the motion pattern for a cluster as a GP with the average mean and variance for all GPs in it. K. Kim. et al. used GPs to model the motion patterns of cars from surveillance camera [6]. They introduce the concept of a *frame*, in which the trajectories are discretisised before fitting GPs to them. Having discrete trajectories meant that the local likelihood for observed data x_t, y_t in time step t could be computed as $P(y_t|x_t, M_k)$ for model M_k, which were aggregated to compute a global similarity metric L_k, which in turn was used to cluster the trajectories. To compute the motion patterns of clusters, a sampling scheme was used. In each time point, three GPs were drawn uniformly without replacement from the cluster. The mean value of all drawn GPs were then used as data points to fit a GP for the clusters motion pattern.

Q. Tran and J. Firl used data with both spatial position and velocity, and used GPs to model a vector field for each observed trajectory [19]. Before GPs were trained, the trajectories were normalised using spline interpolation and a sampling scheme. They did not perform any clustering. Instead, they constructed their motion patterns by driving their own test vehicle. When presented with a new trajectory, the model could compute the most likely motion pattern, and use a particle filter for the vector field to predict continuations of motion patterns.

The work of M. Tiger and F. Heintz aimed to improve upon the approach of K. Kim et al. who had implicitly assumed that all trajectories were generated from one underlying processes [18]. Doing so causes the model to underestimate its variance, which in turn causes the models likelihood to be too small for data that should be considered likely. To avoid this, sets of trajectories were modeled as a mixture-of-Gaussian-processes (MoGP). They then considered "slices" of trajectories, orthogonal to progression, which were approximated as Gaussian distributions. The posterior of these approximations was then assumed to come from a single GP, which was approximated on synthetic data, a process they call *inverse Gaussian process regression*. M. Tiger and F. Heintz also suggested an improvement on the vector field approach proposed by Q. Tran and J. Firl [17], in which they use GPs for the functions $(x, y) \mapsto \tau \mapsto x, y, x', y'$ for $\tau = [0, 1]$, compared to $(x, y) \mapsto x', y'$ used by Q. Tran and J. Firl. Their approach was shown to perform better in benchmarks.

C. Leysen et al. also aim to improve on the work of K. Kim et al [7]. Instead of fitting a GP to re-sampled trajectories, they simply select the trajectory in a cluster which maximises the overall likelihood of the data points in the cluster. Their approach still assumed a single underlying function for all trajectory data, and consequently still underestimated model variance.

Hierarchical Bayesian Models

The idea behind hierarchical Bayesian models used for trajectory learning is borrowed from the natural language processing field, where they are known as topic modeling. Topic modeling are unsupervised techniques for clustering documents into different topics, and by considering spatio-temporal trajectories as documents, their observations as words, and the motion patterns they belong to as topics, the same techniques can be used to model motion pattern. These models are usually based on Latent Dirichlet Allocation (LDA) or a generalisation thereof known as Hierarchical Dirichlet Process (HDP) introduced by Teh et al. [16], which are both so called *bag- of-words*-models. A bag-of-words-model assumes independence between words in a document, which in the domain of trajectories translates to the assumption that observed data points are independently drawn.

Both LDA and HDP require a set amount of clusters, which is a great weakness. Wang et al. proposed a model called *Dual Hierarchical Dirichlet Process* (Dual-HDP) [22], which improves upon the HDP model by allowing the model to learn the number of topics and documents from data. Zhou et al. further improved upon HDP and LDA by using Markov random fields to encode prior beliefs in a model called *Random Field Topic* (RFT) [24].

2.8 Event Detection

In the context of motion patterns and trajectory data, event detection can be seen as the problem of finding patterns in time series. For this, convolution can be used with specific convolution kernels corresponding to the event of interest [14].

Duvenaud et al. explored a different approach, where they modeled time series as GPs with a composite kernel [2]. To construct the kernel they greedily searched over different kernel structures, and composed kernel functions to maximise the marginal likelihood. The structure of the resulting kernel could then be analysed to reveal characteristics of the time series.

3 Data

Explain one observation per second Explain the clusters at stops

4 Method

This chapter describes the proposed system for arrival time prediction and event detection, and its implementation.

4.1 System Description

This section formally describes the system piece-wise. On a conceptual level, the system assumes that similar trajectories, with respect to their motion patterns, should arrive at approximately the same time. When presented with a new trajectory, the system finds previously observed trajectories with similar motion patterns, and predict arrival times based on the most similar ones.

4.2 Trajectory Comparison

The first step is establishing a similarity metric for trajectories which have different temporal lengths, and unevenly distributed observations. Consider the local distance from an observation on T_2 to T_1 as the orthogonal projection onto T_2, as illustrated in Figure 4.1. This approach would eliminate the need to align observations before comparing, however, it requires a continuous trajectory representation $h_i(t)$, while observed trajectories are discrete samples. Assuming the samples are jointly Normally distributed, and come from a smoothly-varying

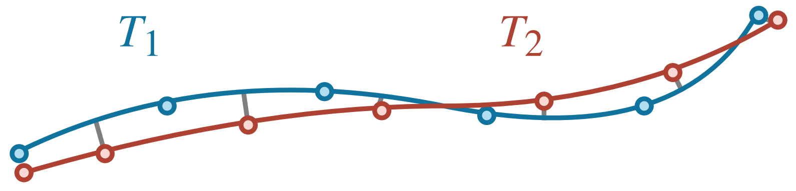


Figure 4.1: An illustration of local distances from T_2 to T_1 as point-wise orthogonal projections.

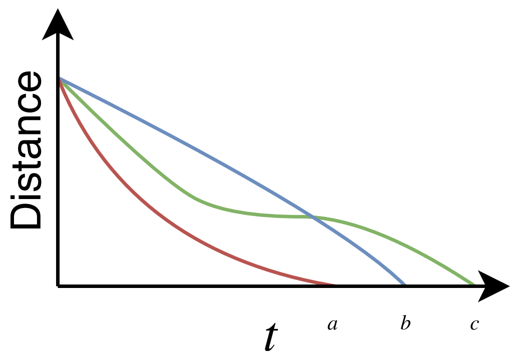


Figure 4.2: Remaining distance for some route as a function of time for unsynchronised trajectories. The functions cannot be fully compared, since the they have different support $[0, a]$, $[0, b]$, $[0, c]$.

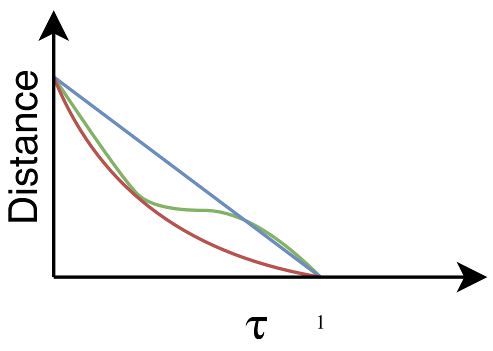


Figure 4.3: Remaining distance for some route as a function of time for synchronised trajectories. The motion patterns are comparable from start to end, since they all have support $[0, 1]$.

function, a GP can be fit to the observations to approximate $x_i^{(n)} = h_i(t)$. This GP will model observations based on points in time, but the *exact points in time* are not relevant, since trajectories for a single segment are generated at many different occasions. Instead, consider modeling trajectory observations as $x_i^n = f_i(\tau)$, where $\tau = [0, 1]$ is the *progress* from start ($\tau = 0.0$) to finish ($\tau = 1.0$) of a trajectory. This GP models observations based on progression for any trajectory on a segment in a unified way. Finding a projection for point x onto trajectory T_i can now be seen as finding how far along the trajectory x would have traveled. More precisely, finding the τ that x corresponds to. This can be done by fitting another GP to $\tau = g_i(x)$, which maps an observation onto the progress relative to T_i. The projection onto T_i is then given by the composition $X = (f \cdot g)(x)$.

as discrete samples. The samples can be seen as samples from a smoothly-varying function of time $x_i^{(n)} = h_i(t)$, which is naturally modeled by a GP.

$T_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(N)})$. Let $T_i = (x_i^{(1)}, x_i^{(2)}, \dots, x_i^{(N)})$ denote trajectory i of length N, where each observation $x_i^{(n)} = (p_x, p_y, v_x, v_y)$ contains position and velocity, and live in state space S. A trajectory can be seen as a sequence of samples from a time dependent, smoothly-varying function, which is naturally modeled by a GP. In light of that observation, it is - reasonable to represent every observed trajectory as a GP, and use the GP log likelihood to represent similarity. However trajectories typically have different temporal lengths, which is referred to as being *unsynchronised*. To make comparisons in a meaningful way, they need to have the *same* temporal length, that is, they need to be *synchronised*. Figure 4.2, and Figure 4.3 illustrate the concept of synchronised trajectories.

Synchronisation Model

Consider modeling observations from a trajectory as $x_i^n = f_i(\tau)$ with an independent multi-output GP for each dimension of S, where $\tau = [0, 1]$ is the progress from start ($\tau = 0.0$) to finish ($\tau = 1.0$) of a trajectory. This model gives a way of to compare the progression of T_i with any other trajectory via the models likelihood, which for a new observation $\bar{x}$ is

$$\begin{aligned} \log P(\tau | \bar{x}, f_i) &= -\frac{1}{2}(\tau - \mu(\bar{x}))[\Sigma]^{-1}(\bar{x} - \mu(\tau)) \\ &= -\frac{1}{2}\log|\Sigma| + C, \end{aligned} \tag{4.1}$$

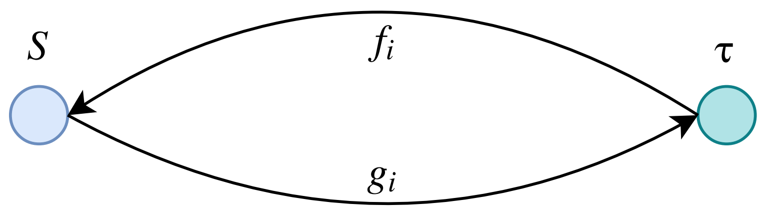


Figure 4.4: An illustration of the synchronisation model $\mathcal{M}^{(i)}_{synch}$, shown as the mappings of f_i and g_i. When synchronising a trajectory, it is mapped by $f_i \circ g_i$.

where $\mu(\bar{x})$ and $\Sigma(\bar{x})$ are given by equations 2.2, and 2.3 for some f_i. However, a way to compute τ for $\bar{x}$ is needed. This is solved by using another GP to learn the inverse model $g_i : pS \mapsto \tau$ for each trajectory T_i. Any observation $\bar{x}$ can consequently be given a similarity metric relative to T_i by $\log P(\tau|X, f_i)$, where $\tau = g(\bar{x})$, and $X = (f \circ g)(\bar{x})$. Together, f_i and g_i make up the *synchronisation model* $\mathcal{M}^{(i)}_{synch} = (f_i, g_i)$ of T_i, which is illustrated in Figure 4.7.

Learning the Synchronisation Model

Learning a synchronisation model $\mathcal{M}^{(i)}_{synch}$ means learning f_i and g_i by MAP estimation. However, there are more constraints on the functions than can be formulated in priors. In particular, a critical property of g_i is that it should be monotonically increasing in the direction of progression, and stationary orthogonal to it. This is explained by the fact that a vehicle is no closer to its destination should it drive more to the left or right on a road; only the progression *along* the road matters. To enforce this property, data augmentation is used.

When training GPs using a stationary kernel function, it is assumed that the data is uniformly distributed, which is not the case for the data available data set. In particular, during stand-stills, many observations are generated in close proximity, which skews the learning of the kernel lengthscale parameters to small values. To avoid this problem, a technique called stop-compression is used.

Data Augmentation

For reasons previously described, g_i should be monotonically increasing in the direction of progression and stationary orthogonal to it. To enforce learning such a function, each observation $T_i^{(i)}$ is duplicated by placing a normal distribution over it, orthogonal to the spatial progression vector $(T_x^{(i+1)} - T_x^{(i)}, T_y^{(i+1)} - T_y^{(i)})^T$, and drawing several samples with the same progression as $T^{(i)}$. This process is illustrated in Figure 4.5 and Figure 4.6.

Stop Compression

Stop compression aggregates observations in close proximity into a single observation through averaging. Finding observations that should be aggregated is done using... Kalman smoother talk to Tiger

Similarity Metric

A similarity metric can be defined by

$$sim(T_i, T_j) = \frac{1}{M} \sum_{i=m}^{M} P(T_i^m | T_i), \tag{4.2}$$

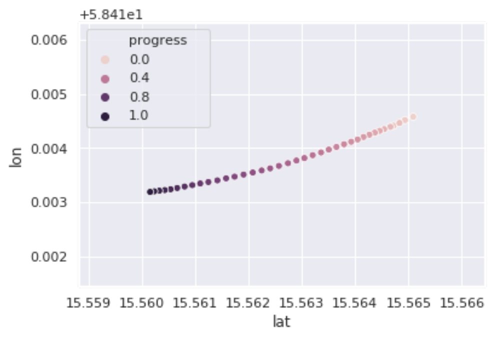


Figure 4.5: Spatial progression of a trajectory before data augmentation.

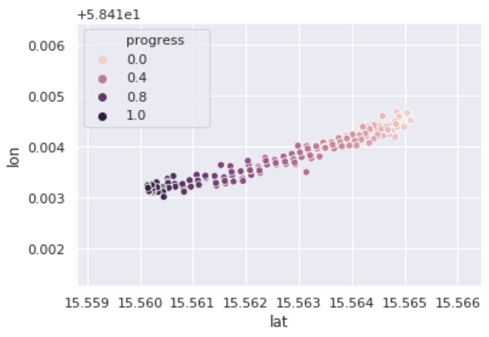
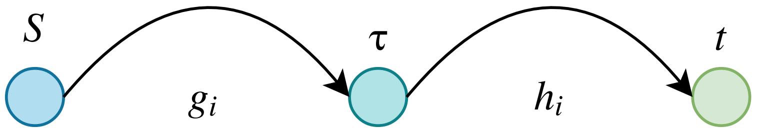


Figure 4.6: Spatial progression of a trajectory after data augmentation.



$$\begin{tikzcd} S \arrow[r, bend left, "g_i"'] & \tau \arrow[r, bend left, "h_i"'] & t \end{tikzcd}$$

Figure 4.7: An illustration of the prediction model. When predicting the arrival time for a trajectory, it is mapped by $h_i \circ g_i$.

where trajectory T_j has length M. The likelihood is computed by synchronising T_j using $\mathcal{M}^{(i)}_{synch}$. Observe that this metric is not symmetric, nor does it satisfy the triangle inequality. Consequently it is not a proper distance.

Trajectory Clustering

Clusters of one trajectory are currently considered.

Arrival Time Prediction

Why is this done from tau and not state?

5 Results

This chapter presents the results. Note that the results are presented factually, striving for objectivity as far as possible. The results shall not be analyzed, discussed or evaluated. This is left for the discussion chapter.

In case the method chapter has been divided into subheadings such as pre-study, implementation and evaluation, the result chapter should have the same sub-headings. This gives a clear structure and makes the chapter easier to write.

In case results are presented from a process (e.g. an implementation process), the main decisions made during the process must be clearly presented and justified. Normally, alternative attempts, etc, have already been described in the theory chapter, making it possible to refer to it as part of the justification.

6 Discussion

This chapter contains the following sub-headings.

6.1 Results

Are there anything in the results that stand out and need be analyzed and commented on? How do the results relate to the material covered in the theory chapter? What does the theory imply about the meaning of the results? For example, what does it mean that a certain system got a certain numeric value in a usability evaluation; how good or bad is it? Is there something in the results that is unexpected based on the literature review, or is everything as one would theoretically expect?

6.2 Method

This is where the applied method is discussed and criticized. Taking a self-critical stance to the method used is an important part of the scientific approach.

A study is rarely perfect. There are almost always things one could have done differently if the study could be repeated or with extra resources. Go through the most important limitations with your method and discuss potential consequences for the results. Connect back to the method theory presented in the theory chapter. Refer explicitly to relevant sources.

The discussion shall also demonstrate an awareness of methodological concepts such as replicability, reliability, and validity. The concept of replicability has already been discussed in the Method chapter (4). Reliability is a term for whether one can expect to get the same results if a study is repeated with the same method. A study with a high degree of reliability has a large probability of leading to similar results if repeated. The concept of validity is, somewhat simplified, concerned with whether a performed measurement actually measures what one thinks is being measured. A study with a high degree of validity thus has a high level of credibility. A discussion of these concepts must be transferred to the actual context of the study.

The method discussion shall also contain a paragraph of source criticism. This is where the authors' point of view on the use and selection of sources is described.

In certain contexts it may be the case that the most relevant information for the study is not to be found in scientific literature but rather with individual software developers and open source projects. It must then be clearly stated that efforts have been made to gain access to this information, e.g. by direct communication with developers and/or through discussion forums, etc. Efforts must also be made to indicate the lack of relevant research literature. The precise manner of such investigations must be clearly specified in a method section. The paragraph on source criticism must critically discuss these approaches.

Usually however, there are always relevant related research. If not about the actual research questions, there is certainly important information about the domain under study.

6.3 The work in a wider context

There must be a section discussing ethical and societal aspects related to the work. This is important for the authors to demonstrate a professional maturity and also for achieving the education goals. If the work, for some reason, completely lacks a connection to ethical or societal aspects this must be explicitly stated and justified in the section Delimitations in the introduction chapter.

In the discussion chapter, one must explicitly refer to sources relevant to the discussion.

7 Conclusion

This chapter contains a summarization of the purpose and the research questions. To what extent has the aim been achieved, and what are the answers to the research questions?

The consequences for the target audience (and possibly for researchers and practitioners) must also be described. There should be a section on future work where ideas for continued work are described. If the conclusion chapter contains such a section, the ideas described therein must be concrete and well thought through.

Bibliography

[1] Damian Campo, Mohamad Baydoun, Lucio Marcenaro, Andrea Cavallaro, and Carlo S. Regazzoni. “Modeling and classification of trajectories based on a Gaussian process decomposition into discrete components”. In: *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)* (Aug. 2017), pp. 1–6. DOI: 10.1109/AVSS.2017.8078495.

[2] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B Tenenbaum, and Zoubin Ghahramani. “Structure discovery in nonparametric regression through compositional kernel search”. In: *arXiv preprint arXiv:1302.4922* (2013).

[3] Sepideh Afkhami Goli, Behrouz H. Far, and Abraham O. Fapojuwo. “Vehicle Trajectory Prediction with Gaussian Process Regression in Connected Vehicle Environment⋆”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)* (June 2018), pp. 550–555. ISSN: 1931-0587. DOI: 10.1109/IVS.2018.8500614.

[4] Zegeye Kebede Gurmu and Wei (David) Fan. “Artificial Neural Network Travel Time Prediction Model for Buses Using Only GPS Data”. In: *Scholar Commons* 17.2 (2014), p. 3. DOI: 10.5038/2375-0901.17.2.3.

[5] ByeoungDo Kim, Chang Mook Kang, Seung Hi Lee, Hyunmin Chae, Jaekyum Kim, Chung Choo Chung, and Jun Won Choi. “Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network”. In: *arXiv preprint arXiv:1704.07049* (2017).

[6] Kihwan Kim, Dongryeol Lee, and Irfan Essa. “Gaussian process regression flow for analysis of motion trajectories”. In: *2011 International Conference on Computer Vision* (Nov. 2011), pp. 1164–1171. ISSN: 2380-7504. DOI: 10.1109/ICCV.2011.6126365.

[7] Christiaan Leysen, Mathias Verbeke, Pierre Dagnely, and Wannes Meert. “Energy consumption profiling using Gaussian processes”. In: *2016 IEEE 8th International Conference on Intelligent Systems (IS)* (Sept. 2016), pp. 470–477. DOI: 10.1109/IS.2016.7737463.

[8] Brendan T. Morris and Mohan M. Trivedi. “Learning and Classification of Trajectories in Dynamic Scenes: A General Framework for Live Video Analysis”. In: *2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance* (Sept. 2008), pp. 154–161. DOI: 10.1109/AVSS.2008.65.

[9] Duc-Duy Nguyen, Chan Le Van, and Muhammad Intizar Ali. *Vessel Destination and Arrival Time Prediction with Sequence-to-Sequence Models over Spatial Grid*. ACM, June 2018. ISBN: 978-1-4503-5782-1. DOI: `10.1145/3210284.3220507`.

[10] Junbiao Pang, Jing Huang, Yong Du, Haitao Yu, Qingming Huang, and Baocai Yin. "Learning to Predict Bus Arrival Time From Heterogeneous Measurements via Recurrent Neural Network". In: *IEEE Transactions on Intelligent Transportation Systems* (2018).

[11] Marco A. F. Pimentel, David A. Clifton, and Lionel Tarassenko. *Gaussian process clustering for the functional characterisation of vital-sign trajectories*. IEEE, Sept. 2013. DOI: `10.1109/MLSP.2013.6661947`.

[12] Carl Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[13] Mathieu Sinn, Ji Won Yoon, Francesco Calabrese, and Eric Bouillet. "Predicting arrival times of buses using real-time GPS measurements". In: *2012 15th International IEEE Conference on Intelligent Transportation Systems* (Sept. 2012), pp. 1227–1232. ISSN: 2153-0017. DOI: `10.1109/ITSC.2012.6338767`.

[14] Steven W Smith et al. "The scientist and engineer's guide to digital signal processing". In: (1997).

[15] Jingren Tang, Hong Cheng, Yang Zhao, and Hongliang Guo. "Structured dynamic time warping for continuous hand trajectory gesture recognition". In: *Pattern Recognit.* 80 (Aug. 2018), pp. 21–31. ISSN: 0031-3203. DOI: `10.1016/j.patcog.2018.02.011`.

[16] Yee W Teh, Michael I Jordan, Matthew J Beal, and David M Blei. "Sharing clusters among related groups: Hierarchical Dirichlet processes". In: *Advances in neural information processing systems*. 2005, pp. 1385–1392.

[17] Mattias Tiger and Fredrik Heintz. "Gaussian Process Based Motion Pattern Recognition with Sequential Local Models". In: *2018 IEEE Intelligent Vehicles Symposium (IV)* (June 2018), pp. 1143–1149. ISSN: 1931-0587. DOI: `10.1109/IVS.2018.8500676`.

[18] Mattias Tiger and Fredrik Heintz. "Online sparse Gaussian process regression for trajectory modeling". In: *2015 18th International Conference on Information Fusion (Fusion)* (July 2015), pp. 782–791. URL: `https://ieeexplore.ieee.org/abstract/document/7266640/citations?tabFilter=papers#citations`.

[19] Quan Tran and Jonas Firl. *Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression*. IEEE, June 2014. DOI: `10.1109/IVS.2014.6856480`.

[20] M. Vlachos, G. Kollios, and D. Gunopulos. *Discovering similar multidimensional trajectories*. IEEE, Feb. 2002. DOI: `10.1109/ICDE.2002.994784`.

[21] Haozhou Wang, Han Su, Kai Zheng, Shazia Sadiq, and Xiaofang Zhou. *An effectiveness study on trajectory similarity measures*. Australian Computer Society, Inc., Jan. 2013. ISBN: 978-1-921770-22-7. URL: `http://dl.acm.org/citation.cfm?id=2525416.2525418`.

[22] Xiaogang Wang, Keng Teck Ma, Gee-Wah Ng, and W. Eric L. Grimson. "Trajectory analysis and semantic region modeling using a nonparametric Bayesian model". In: *2008 IEEE Conference on Computer Vision and Pattern Recognition* (June 2008), pp. 1–8. ISSN: 1063-6919. DOI: `10.1109/CVPR.2008.4587718`.

[23] Zhang Zhang, Kaiqi Huang, and Tieniu Tan. *Comparison of Similarity Measures for Trajectory Clustering in Outdoor Surveillance Scenes*. Vol. 3. Aug. 2006. DOI: `10.1109/ICPR.2006.392`.

[24] Bolei Zhou, Xiaogang Wang, and Xiaoou Tang. "Random field topic model for semantic region analysis in crowded scenes from tracklets". In: *CVPR 2011* (June 2011), pp. 3441–3448. ISSN: 1063-6919. DOI: `10.1109/CVPR.2011.5995459`.