Context-Matching in Mobility Trajectories Using Particle Filters

Mohammadreza Amini[1], Mahmoud Sakr[1,2]

[1]Université Libre de Bruxelles, Brussels, Belgium
[2]Ain Shams University, Cairo, Egypt

Abstract

In this paper we propose a novel kind of trajectory analysis, so called context matching. It matches a given trajectory against a movement model, which is given as a Markov chain, capturing what we call the movement context. Particle filters are then used to match a given trajectory against its movement context with the ultimate goal of inferring semantic properties of the movement trajectory. Firstly, we introduce the method in its generality by illustrating multiple showcases. Secondly, we develop the formal model of context-matching. Finally, we illustrate an example use-case for annotating trajectories of fishing vessels into fishing and sailing segments. By doing so, we show the effectiveness of context matching and its use in enriching real-world trajectory datasets.

Keywords

Trajectory, Context Matching, Particle Filters, Markov Chains, Mobility Data

1. Introduction

Location tracking of moving objects is becoming more and more feasible, thanks to the recent advancement of various technologies, including GPS, Wi-Fi-tracking, Bluetooth, etc. The data collected from these devices are typically sequences of location and time. The semantics of the movement, such as the purpose of the trip and the mode of transport, are not captured during the observation process. This gap has been identified and addressed by extensive research work under the title of *semantic trajectories*, where one major task is to annotate trajectory segments by some behavioral properties.

In the literature, many papers approach the matching of semantic representation to a given trajectory dataset by using segmentation algorithms [1, 2, 3]. Although these papers show promising results, the methodologies that they propose are specified uniquely to a given dataset or a given context and do not work in a generalized way. The objective of this paper is to generalize the problem and approach it in a unified way.

The abstract idea of this work is that the object is moving in two spaces: it is moving in the geospatial space, as observed by the location tracking device, and in parallel, it moves in a semantic space changing its semantic properties, which we call *context*. We assume that this context space can be modeled, using domain knowledge. The task of inferring the movement semantics is thus mapped into matching the observed spatiotemporal trajectory against the modeled context, i.e., *context-matching*.

This paper aims to propose a general methodology that solves the problem of context-matching. Particle filters [4] are generally used to solve estimation problems, with wide adoption in the fields of robotics and trajectory guessing. The proposed methodology uses particle filtering in its core alongside Hidden Markov Models and concepts seen in Map-Matching [5] in order to match a defined set of context to a given set of trajectory data.

To represent the context, we use Markov chain which is a statistical model consisting of a set of states and the probabilities of moving between them. The model defines the context that needs to be matched to the spatiotemporal trajectory. After defining the model, particle filtering algorithm is applied to the spatiotemporal trajectory in order to match a context to each trajectory point. The particle filtering process allows us to generate particles for each given observation and with the help of our model representation, we can choose the best particle between particles (filtering process) and keep it as the context result associated with that given observation. With the above considerations, our main contributions are:

- the introduction of context-matching, as a generic analysis for inferring trajectory semantics
- a generalized technique based on particle filtering in order to match context to the trajectory data
- illustration of the application of the proposed methodology on real-world data

The rest of the paper is organized as follows. In Section 2, we casually describe context-matching and the motivation behind it. Section 3 discusses related work.

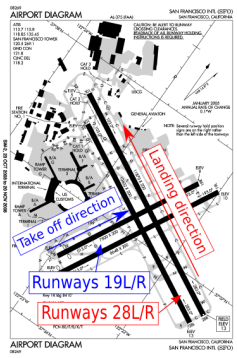
Proceedings of the Workshop on Big Mobility Data Analytics (BMDA) co-located with EDBT/ICDT 2023 Joint Conference (March 28-31, 2023), Ioannina, Greece

mohammadreza.amini@ulb.be (M. Amini);
mahmoud.sakr@ulb.be (M. Sakr)

Section 4 provides the formal development of context-matching. We then illustrate a showcase in Section 5. Lastly, Section 6 concludes this paper and discusses possible future directions on the topic.

2. Context-Matching

This section casually describes the idea of context-matching, through examples in different mobility domains, and different application focus. The first example (Figure 1a, Figure 1b), presented in [6], tries to infer the flight operations in San Francisco airport. The ADS-B trajectories of aircraft are analyzed in order to identify the combination of landing, turnover, and take-off resources (e.g., runways) used by the aircraft. This analysis is important for addressing the challenges of airspace monitoring. The paper proposes a non-supervised waypoint-based trajectory clustering to identify and group the turning points into different configurations. Fig. 2 illustrates a simplistic Markov chain that represents the semantic sequence of events through which flight operations can be determined. We call this Markov chain, the context-diagram. The proposed context-matching will stochastically devise a correct assignment of the labels in this context diagram to the trajectory points. Such semantic enrichment of the trajectory points would thus enable the identification of flight operations.



(a) San Francisco airport diagram with take-off and landing direction in the west configuration [6]

(b) Northern California TRACON (NCT) standard traffic patterns, west configuration [6]

Figure 1: Example patterns of flight operations

A second work that can be achieved by context-matching is the segmentation of fishing vessel trajectories. For instance, [3] proposes window-based trajectory segmentation algorithm that aims to detect fishing activities as completely as possible. This example will be illustrated in the experiments in Section 5.

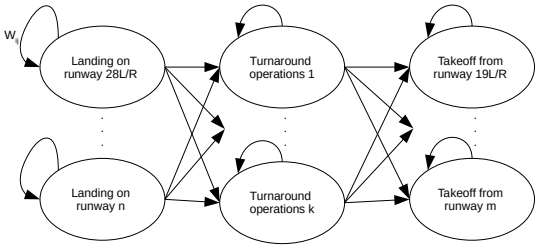


Figure 2: An example Markov Chain capturing the general model of flight ground operations. Every flight undergoes three main steps: (1) landing on a selected runway, (2) turnaround operations on the ground, i.e., operations for handling inbound and outbound exchanges of passengers, crew, catering, cargo, and baggage, (3) taking-off from a selected runway. For every step, there are multiple choices, i.e., landing/take-off runway, and turnaround operations. The transitions/edges are weighted. Weights might come from the operations manual of the airport, or observed in the data history. An operations-mode is abstracted in this diagram as the combination of landing, turnaround, and takeoff states. Notice that this is a simplified 'proof-of-concept' diagram, that might be further detailed, e.g., by further breaking down turnaround operations.

Context-matching is proposed as an abstraction of these analyses. Abstractly speaking, these applications try to infer semantic properties of the moving trajectory, which were lost during the observation process. Context-matching assumes that these analyses, and alike, can be carried out through matching the trajectories to models that capture their movement mode/behavior/semantics, here called *context*. While the movement model can vary across data and application domains, the matching algorithm remains the same. The analysis task then transforms into defining a movement model that best captures the context. As such, we are able to carry them out in a unified way using the same algorithm.

3. Related Work

Moving object databases are mainly focused on storing trajectories as location sequences [7, 8, 9]. Understanding the movement semantics is however essential for many analyses. In the literature, different methodologies are used for inferring the trajectory semantics. The common aspect of these methodologies is that they are specific to a certain data modality and certain semantic properties. In [10, 11], each trajectory point is assigned to a specific label describing the activity of the fishing vessel at that point. On another class of applications, [2, 12, 1, 3, 13, 14] deal with trajectory segmentation problem and partition the trajectory data into segments and label each segment with a given activity. These approaches are not general-

izable, as they are tightly fitted to specific domains.

The proposed context-matching methodology inspires by the methods introduced in the *Map-Matching* domain [5, 15]. The problem of map-matching is defined as the procedure that determines which road a vehicle is on utilizing data from sensors. Hence, in the context of map-matching, we are matching the trajectory data with the underlying road network. This is similar to what we desire to achieve in context-matching which is matching a given set of labels to trajectory data. In [15], a state of the art is given for different methods used to solve the problem of map-matching. These methods range from geometric approaches to topological approaches and probabilistic approaches.

In [15], a *Hidden Markov Model* is used as a probabilistic model to solve the problem of map-matching. We inspire from this approach in our methodology in order to define how context is defined and how to transition from one context to another by using the Markov chain. In HMM map-matching, the Viterbi algorithm [16] is used generally in order to procure the most probable lattice through the hidden state and output it as the result. This algorithm works fine in map-matching but in context-matching it is difficult to apply it because of added context to the state space. This is why we opted to use particle filter which is more flexible.

Particle filtering [17] is a sequential Monte Carlo algorithm that is a probabilistic model used in the methodology to assign context to a given point by using the defined model's transition probabilities. Particle filtering is a genetic algorithm. Its objective is to find the most probable sequence of hidden states in a given model.

4. Proposed Methodology

Context-Matching consists of matching a given set of semantic properties (context) to a given sequence of spatiotemporal points. We assign labels to each trajectory point which describes its movement semantics. Hence, the input data is a set of trajectory points that should at least provide spatiotemporal coordinates. In addition, each trajectory point could have the necessary information regarding the context such as speed, heading, time, distance from the last collected trajectory point, etc. Based on this input, the methodology decides how to assign the semantic representation to each trajectory point. The output of the methodology is the annotated trajectory, in which each trajectory point is augmented by labels that represent its semantic (context).

The methodology has two main parts. Firstly we need to define the model by specifying its Markov chain. Then, we need to apply particle filtering to this model in order to obtain the matching of trajectory data in which each trajectory instance has a label assigned to it that describes its movement semantic.

4.1. Input definition

We start by defining the input for the context-matching analysis. As mentioned earlier, this methodology uses a Markov chain as a representation of state-space, which describes the movement context. A Markov Chain is a stochastic process that undergoes transitions from one state to another. The Markov property restricts that each step in the process is *memory-less*. In other words, only the current state of the process is affecting its successor state, and not the long history of the state transitions.

This stochastic process may thus be described as a Bayesian systems where the probability of the current state depends only on the previous state. We describe this context, i.e., the Markov chain, by a weighted directed graph, where the edges indicate the transition probabilities of going from one state c to another. In the sequel, we call the Markov chain used for representing the movement context as the *context-diagram*.

As input, we also get a spatiotemporal trajectory in the form of a sequence of observations $< (loc_i, t_i) >$. Other spatiotemporal properties of the moving objects may be available or computed, such as the speed, heading, distance from certain geospatial objects, etc. In an abstract sense, a list of observations, either recorded by sensors or derived is available, denoted as $Z = \{z_1, ..., z_n\}$.

What makes context-matching complex in its modeling is the state space. In addition to geographic information such as the speed, direction, and coordinates, a label feature is added as well to represent the context. Hence we need a genetic algorithm that could consider this when finding an optimal path through the model lattice.

4.2. Particle filters

A particle filter, similar to the Kalman filter, is a technique for estimating the state of a dynamic system. It is a recursion-based filter which takes the current belief and updates it based on so-called motion information or control commands and based on observations. In contrast to the Kalman, it relaxes the assumption that we are in a Gaussian space. It actually allows you to describe arbitrary probability distributions and it uses a non-parametric form in order to do this.

What the particle filter does is that it just uses a number of so-called *particles*, which are hypotheses for the system being in one single state. Every particle refers to one guess that the system is in that state. For context-matching, the state represents some semantic properties of a moving object. Then a thousand particles will be a thousand guesses or a thousand hypotheses about what the object's state could be in the real-world.

The particle filtering process aims at estimating the most probable sequence of states from the given sequence of observations. To do so, it applies the weight-and-resampling method to perform a kind of survival-of-the-fittest principle by injecting the tendency to replicate particles which have a high importance weight and to forget those particles which have a low importance weight. In general, particle filtering process consists of three steps. For each observation z_t at time t, the particle filtering process deals with multiple particles $S_t = \{x_t^1, x_t^2, ..., x_t^l\}$, each of which is weighted by an importance weight $W_t = \{w_t^1, w_t^2, ..., w_t^l\}$. Assuming that the particles and weights are already initialized at time 0, here are the three steps of the particle filtering process:

1. Sampling: particles in S_t are sampled with replacement according to their weights in W_t, i.e., bootstrapping. This shall result, as illustrated in Fig.3, that the higher-weight particles get selected multiple times, while some lower-weight particles get never selected. The resampling step can be compared to "natural selection" where the best samples survive.

2. Update/Drift & Diffuse: for each particle $x_t^i \in S_t$ a new particle x_{t+1}^i is generated. This step is broken into:

a given the context-state of the selected particle x_t^i, let it be denoted c_t^i, draw a sample context-state c_{t+1}^i using the transition probabilities in the Markov chain

b based on x_t^i and c_{t+1}^i, generate a new particle x_{t+1}^i by applying the corresponding motion model F. The motion model will result in an update action u_{t+1}, e.g., a translation with a certain speed and heading. This is denoted *Drift* in Fig. 3. Here we are assuming the spatiotemporal properties of the movement are different per context-state. This is a realistic assumption since otherwise, we know that there is no sufficient information to infer the movement semantics. For instance, a fishing vessel will undergo different speeds, turns, etc, when fishing than when sailing. Next, a noise model (Diffuse) is applied in order to cover some unlikely hypotheses and to differentiate the particles that are copied from the same x_t^i. In practice, drift and diffuse can be combined in the motion model F. In such a case, F is represented as a set of probability distributions of the motion delta of every spatiotemporal property.

3. Measure: This is where the observation z_t comes into the play. A weight is assigned for each generated particle $x_{t+1}^i \in S_{t+1}$ proportional to the emission probabilities of the model $P_e(z_{t+1}, x_{t+1}^i)$. Weights can be as simple as the inverse of the euclidean distance between

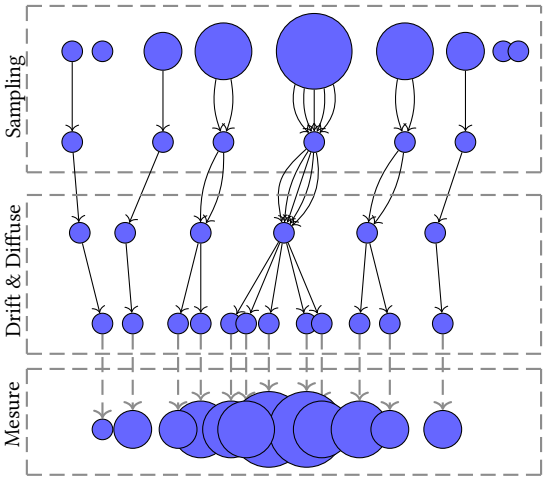


Figure 3: The particle filtering process

the particle and the observation, or involve other complex similarity measures, depending on the application.

In other words, particle filters allow us to estimate the internal states in dynamical systems when partial observations are made and there are perturbations in the data. The principle is to randomly generate particles following the transition probability laws, and then select the most likely ones according to the observations and the emission probability laws defined in the model. The objective is to calculate the posterior process of a Markov model given the observation state. This posterior state space is represented by with temporal set S_t with n weighted particles. The sum of all importance weights at time t for a sample set x is 1. Mathematically particle filters are solving the following:

- given a set of particles $S_t = \{x_t^1, x_t^2, ..., x_t^l\}$, representing the posterior $Bel(x_t) = P(x_t|z_1, ..., z_t, u_1, ..., u_t)$
- generate a set of particles $S_{t+1} = \{x_{t+1}^1, x_{t+1}^2, ..., x_{t+1}^l\}$ of $Bel(x_t) = P(x_{t+1}|z_1, ..., z_t, \mathbf{z_{t+1}}, u_1, ..., u_t, \mathbf{u_{t+1}})$

Instead of attempting to solve the exact Bayes system to find the posterior, particle filters represents the distribution of the posterior by a set of particles drawn from this distribution. Such a representation is approximate, but it is non-parametric, and therefore can represent a much broader space of distributions than, for example, Gaussians, which is needed to enable a wide range of applications for context-matching.

As can be seen, in *Drift & Diffuse* step in Fig. 3, no observation is taken into consideration and the particle generation is based on the Markov chain context and

the motion model to generate particles. It is at this step that we take into account the transition probabilities of the model. The *Measure* step is in charge of assigning a weight to each generated particle. This step uses the emission probabilities of the model in order to give each sample a well-defined weight. Hence, during the first step, new samples are generated from the previous ones, regardless of the observation z_{t+1}. In context-matching application, the particle filtering process samples generate many pseudo-random trajectory points associated with context-states and then select the ones that stick the most to the observation state.

It should also be noted that for each time instant t, there are multiple sample candidates and even if they are classified by the *importance sampling* step, there is not only one retained. This dynamic setup makes the system resilient to noise and observation errors.

4.3. Context-Matching

To sum-up, the proposed context-matching, as illustrated in Algorithm 1, model consists of:
1. A movement context, given as a Markov chain $G = (C = \{c_1, ..., c_k\}, P_t = \{p_{ij} | 1 \leq i, j \leq k, 0 \leq p_{ij} \leq 1\})$
2. A trajectory, given as a sequence of observations $Z = \{z_1, ..., z_n\}$, where every observation contains the spatiotemporal coordinates and possibly other movement attributes.
3. A movement model F, computed from ground truth. This model is used in the drift & diffuse step in order to change the sampled particles.
4. The particle filter process, as follows:

a A set of particles $S_t = \{x_t^1, x_t^2, ..., x_t^l\}$, which are initially randomized uniformally over the state-space. Alongside, there is also the set of weights $W_t = \{w_t^1, w_t^2, ..., w_t^l\}$. W_0 is initialized with equal weights $1/l$ for all particles. For both sets t iterates in the range $[1..n[$, where n is the number of observations in Z, (lines 2,3 in Alg. 1).

b A bootstrapping routine for the weighted sampling of particles$Z$, (line 5 in Alg. 1).

c A drift & diffuse routine for applying the movement model F to the sampled particles, and producing $S_{t+1} = \{x_{t+1}^1, x_{t+1}^2, ..., x_{t+1}^l\}$, (line 6).

d A *measure* routine to assign weights $W_{t+1} = \{w_{t+1}^1, w_{t+1}^2, ..., w_{t+1}^l\}$, (line 9).

e Steps b – d are repeated until all the n trajectory observations are consumed.

5. The particle filter is an online process that keeps updating the hypotheses, without retaining the history of updates. For context matching, we would like to track the history for performing the semantic annotation. For this, we upgrade the particles with additional memory each $M_i = \{m_i^1, m_i^2, ..., m_i^n\}$. At every drift & diffuse step, we append the sampled context-state in the memory of the particle in hand, (line 7 in Alg. 1). The mechanism of the survival-of-the-fittest shall result that the surviving particles at the end of the particle filter process being the ones that mostly performed the correct state updates during the filter iterations. We then perform an average, by means of majority voting, over the memory of all surviving particles $\{M_1, M_2, ..., M_l\}$ to predict the state annotations of the trajectory observations, (lines 16-18 in Alg. 1). That is, the state $c \in C$ that will be assigned as annotation for the observation z_i, will be the state which repeats the most in $\{m_j^i\}, 1 \leq j \leq l$.

The practical design decisions, in terms of parameter settings, that we choose for this model are as follows:

- A trade-off has to be made between the number of particles l, and the efficiency of the particle filtering process. A higher value for l would potentially improve the approximation of the density distribution of the state-space, but it would increase the computational cost. In context-matching, we expect that the state-space will consist of 10s of context-states. Therefore a good number of particles would be in the order of 100s.

- Instead of asking the movement model F as input, we compute it by means of data analysis of a ground-truth. The ground-truth provided by the user, needs to be labeled by the context-state. The analysis will then learn statistical summaries about the spatiotemporal proprieties of the motion refined at context-state level. Many models exist for this analysis, in the fields of robotics and location prediction e.g., [18, 19, 20].

- In the *measure* step, we use emission probabilities in order to give a weight to each sample, (line 9 in Alg. 1). This emission probability can be found in item 4.3

$$p(z_t, x_{t,i}) = \frac{1}{\sqrt{2\pi}\sigma_z} e^{0.5(\frac{||z_t - x_{t,i}||_{\text{great circle}}}{\sigma_z})^2}$$

  where σ_z is the standard deviation of location error, e.g., a widely accepted value for GPS error is 5 meters.

5. Experiments

In this section, we showcase the context-matching method in an application for segmenting the trajectories of fishing vessels. The goal is to annotate the points in the AIS trajectory of a vessel by either fishing or sailing. We illustrate the use of context-matching: data preparation, parameter setting, etc. We also compare the results

Algorithm 1 Algorithm context-matching

Input list of observations $Z = [z_1 ... z_n]$, Markov chain G, motion model F
Output annotation list $[a_1, ..., a_n]$

Initialize lists S_1, W_1
for $t = 1$ **to** n **do**
  $S_{t+1} = \phi$, $\eta = 0$, $A = \phi$, $l =$ num particles
  for $i = 1$ **to** l **do**
    x= sample one particle in S_t using the weights W_t
    c= sample one context-state in G setting the current state equal to the state of x_i
    Append c to the memory of x
    $\bar{x}$= sample a new particle from $P(\bar{x}|x, F_c)$, where F_c is the motion model of state context c
    $\bar{w}$= $P(z_{t+1}|\bar{x})$ {measure}
    $\eta = \eta + \bar{w}$ {sum weights for normalization}
    $S_{t+1} =$ Append $\bar{x}$ to S_{t+1}
    $W_{t+1} =$ Append $\bar{w}$ to W_{t+1}
  Normalize W_{t+1} by dividing all weights by η {generate the annotation a_t}
for $t = 1$ **to** n **do**
  $a_t =$ find the most recurring c in the t^{th}-slot of memory of the particles $S_n = \{x_n^1, x_n^2, ..., x_n^l\}$
return $[a_1, ..., a_n]$

with [3], which proposed a dedicated method for this segmentation.

5.1. Dataset

We use the same labeled dataset in [3]. It consists of AIS trajectories obtained from the Website of the Danish Maritime Authority [1], then annotated into sailing, and fishing segments. The data used is the AIS files of the week between Nov 14, 2021, and Nov 20, 2021. The dataset is restricted to the fishing vessels only, thus other types of vessels are not included in the analysis. The dataset contains $1,080,220$ points and has an average sampling interval of 10.63 seconds. It is made of 128 trajectories that were manually labeled in [3].

5.2. Context-matching of fishing vessels activities

The goal of this analysis is to assign a label to each trajectory point describing the activity of the fishing vessel at that point. As in [3], we consider the activities of sailing and fishing. From a sequence of AIS positions, a context-matching algorithm should be able to distinguish the fishing segments and the sailing segments. This use case is in fact a true issue because the detection of vessel activities allows for combating illegal fishing activities. In addition, depending on the model, boats must be equipped with an automatic identification system (AIS), a system that contains a GPS, so it is possible to track them. Therefore this segmentation has been a point of attention in research, e.g., in [21] a method based on Hidden Markov Model is proposed for detecting ship activities. Note that the goal of this experiment is not to propose a better trajectory segmentation method. Rather, our goal is to illustrate how the proposed context-matching method can be specialized for a semantic annotation application.

As discussed during section 4, we need to first design a Markov Chain G, that captures the context of motion. In this segmentation application, the context consists of two states and the transition probabilities of these two states. This model is presented in Figure 4. We can see that the probability of staying in the same state for both states is 90% while the probability of transitioning from one state to another for both states is 10%. They are obtained by simple statistical analysis of the ground truth.

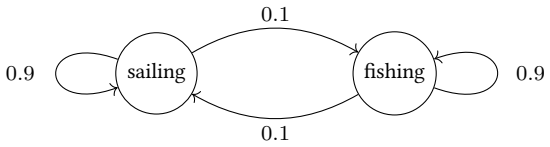


Figure 4: Context diagram for detecting fishing vessels activities with two states

Before applying particle filtering to the data, we need to define some parameters. These parameters are:

- **Number of particles** l : as discussed in section 4, for each observation, a fixed number of particles need to be generated. In this experiment, we fix this value to 100.
- **Motion Model** F: this parameter is used in the drift & diffuse step in order to update the particles. Here we use a simple model based on Gaussian and uniform distributions of speed and heading changes respectively, as detailed in the following points.

Let us now describe the drift&diffuse process and how it is applied in this experiment. This process is the core of the particle filters and this methodology as discussed in section 4. The principle of this process is to randomly update the sample x_t^i to the sample x_{t+1}^i. This is where the transition probability $p_t(x_t^i, x_{t+1}^i)$ and the Markov chain defined in Figure 4 are used. In this experiment, the process is split into four parts, in order to update all the attributes of the particle. Here are these four parts:

[1] https://web.ais.dk/aisdata/

- **Update heading:** The heading change is randomly drawn from -22.91 to $+22.91$ degree from the previous one. The distribution is uniform. This range is meant to reflect the degree of the maneuverability of a fishing vessels.

$$p_t(x^i_{t+1}.(\text{dir}) = y, x^i_t.(\text{dir}) = \hat{y}) \sim \mathcal{U}(\hat{y} - +22.91)$$

- **Update context:** This is where we use the Markov chain model (Figure 4) in order to update the context.

$$p_t(x^i_{t+1}.(\text{context}) = \text{sail}, x^i_t.(\text{context}) = \text{fish}) =$$

$$p_t(x^i_{t+1}.(\text{context}) = \text{fish}, x^i_t.(\text{context}) = \text{sail}) = 0.1$$

$$p_t(x^i_{t+1}.(\text{context}) = \text{sail}, x^i_t.(\text{context}) = \text{sail}) =$$

$$p_t(x^i_{t+1}.(\text{context}) = \text{fish}, x^i_t.(\text{context}) = \text{fish}) = 0.9$$

- **Update speed:** The speed is set according to the context. The value is drawn following the Gaussian distribution whose parameters are computed using the ground-truth.

  The speed update is then applied using the following formula:

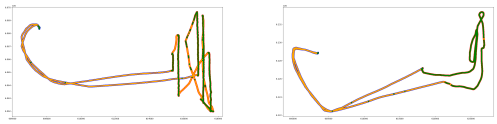
$$p_t(x^i_{t+1}.(\text{speed}) = v, x^i_t.(\text{speed}) = \hat{v}) =$$

$$\begin{cases} \mathcal{N}(\mu_{\text{sailing}}, \sigma_{\text{sailing}}), & \text{if } x^i_{t+1}.(\text{context}) = \text{sailing} \\ \mathcal{N}(\mu_{\text{fishing}}, \sigma_{\text{fishing}}), & \text{if } x^i_{t+1}.(\text{context}) = \text{fishing} \end{cases}$$

- **Update position:** The position is calculated using the heading and the speed.

$$\vec{x^i_{t+1}}.(\text{pos}) = \vec{z^i_{t+1}}.(\text{pos}) + \Delta t \times x^i_{t+1}.(\text{speed}) \times \vec{x^i_{t+1}}.(\text{dir})$$

  Here $\vec{x^i_{t+1}}.(\text{dir})$ denotes the unit vector whereas $x^i_{t+1}.(\text{dir})$ is its argument.



(a) MMSI #219002136 (b) MMSI #219011321

Figure 5: An example of two context-matched trajectories of a fishing vessel. Green points represent fishing activity while orange points represent sailing activity. Blue outlines are the actual sailing activity as per the ground-truth while red outlines are actual fishing activities. Hence, we should mostly have orange points with a blue outline and green points with red outlines.

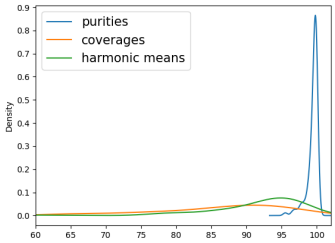


Figure 6: Density distribution of experiment result

5.3. Results

Running the context-matching as described in the previous two sections, we observed promising results that confirm our expectations. Table 1 reports the purity, coverage (described in [22]), and harmonic mean of our results. In comparison to the WBS-RLE method in [3], we consider that the achieved accuracy is good for a generic method, in comparison to a dedicated method. The density results of these three metrics can be found in Fig. 6. We can observe that the majority of trajectories have a really high purity and high coverage which results in a high harmonic mean.

method	purity	coverage	harmonic mean
Proposed methodology	0.990	0.847	0.908
WBS-RLE	0.890	0.974	0.927

Table 1
Results of context matching for fishing vessels

We observe however that the resulting number of segments is very high on average. The results include many short segments of a few points, e.g., short sailing segments inside a longer fishing segment and vice versa, as illustrated in the example in Fig. 5. Our best guess is that this is an issue of parameter setting, such as the transition probabilities in the Markov chain. This is an important point of further research, that we note for future work.

In Fig. 5, two examples of context-matched trajectories for trajectories $\#219011321 - 3$ and $\#219002136 - 3$ can be found.

6. Conclusions and Future Work

The paper proposed a new tool for the analysis of mobility trajectories called context-matching. The theoretical foundation is based on Markov models and particle filters. The Markov model enables users to describe a semantic context of the motion. A memory-enriched particle filter

process is then applied to annotate the trajectory point with the semantic states of this context.

Building on the flexibility of Markov models, it is possible to describe arbitrarily complex models. Also building on the dynamics of particle filters, no assumptions are made about the probability distribution of the resulting annotation. As a result, the proposed context-matching is capable of accurately annotating trajectory points in a variety of application scenarios.

While this work is still in its beginning, the preliminary experiments in this paper show promising results, confirming that the theory is correct. The experiments illustrate context-matching in application to trajectory segmentation. The results are almost as accurate as an algorithm which was developed in previous work specifically for this task and for this data modality.

The preliminary results in this paper encourage multiple directions for continuing this work. One challenge that a lot of context authoring is required. Our impression is that it is possible to automate most of it by incorporating other exploratory analysis methods. Secondly, some debugging tool is needed to help analysts unpack the stochastic process of particle filters, and fine-tune it. A third direction would be to experiment with other data and use cases, including the ones mentioned in this paper.

Acknowledgement

The authors acknowledge the contribution of Marceau Caron, who has developed a first version of this work.

References

[1] J. A. M. R. Rocha, V. C. Times, G. Oliveira, L. O. Alvares, V. Bogorny, DB-SMoT: A direction-based spatio-temporal clustering method, in: 5th IEEE International Conference Intelligent Systems, 2010.

[2] L. A. Leiva, E. Vidal, Warped k-means: An algorithm to cluster sequentially-distributed data, Information Sciences 237 (2013) 196–210.

[3] S. Wu, E. Zimányi, M. Sakr, K. Torp, Semantic segmentation of ais trajectories for detecting complete fishing activities, in: 23rd IEEE MDM, 2022.

[4] H. R. Künsch, Particle filters, Bernoulli 19 (2013).

[5] P. Newson, J. Krumm, Hidden markov map matching through noise and sparseness, in: 17th ACM SIGSPATIAL, November 4-6, Seattle, WA, 2009.

[6] M. Gariel, A. Srivastava, E. Feron, Trajectory clustering and an application to airspace monitoring, IEEE Transactions on Intelligent Transportation Systems - TITS 12 (2010).

[7] M. S. Bakli, M. A. Sakr, T. H. A. Soliman, A spatiotemporal algebra in hadoop for moving objects, Geo-spatial information science 21 (2018) 102–114.

[8] E. Zimányi, M. Sakr, A. Lesuisse, M. Bakli, Mobilitydb: A mainstream moving object database system, in: Proceedings of the 16th International Symposium on Spatial and Temporal Databases, 2019.

[9] M. Bakli, M. Sakr, E. Zimanyi, Distributed moving object data management in mobilitydb, in: Proceedings of the 8th ACM SIGSPATIAL, 2019.

[10] F. Natale, M. Gibin, A. Alessandrini, M. Vespe, A. Paulrud, Mapping fishing effort through ais data, PloS 10 (2015).

[11] G. Rovinelli, S. Matwin, F. Pranovi, E. Russo, C. Silvestri, M. Simeoni, A. Raffaetà, Multiple aspect trajectories: a case study on fishing vessels in the northern adriatic sea., in: EDBT/ICDT Workshops, 2021.

[12] M. Etemad, A. Soares, E. Etemad, J. Rose, L. Torgo, S. Matwin, Sws: an unsupervised trajectory segmentation algorithm based on change detection with interpolation kernels, GeoInformatica 25 (2021) 269–289.

[13] N. Magdy, M. A. Sakr, K. El-Bahnasy, A generic trajectory similarity operator in moving object databases, Egyptian Informatics Journal 18 (2017).

[14] M. Attia Sakr, R. H. Güting, Spatiotemporal pattern queries in secondo, in: Advances in Spatial and Temporal Databases: 11th SSTD, Springer Berlin Heidelberg, 2009, pp. 422–426.

[15] M. A. Quddus, W. Y. Ochieng, R. B. Noland, Current map-matching algorithms for transport applications: State-of-the art and future research directions, Transportation Research Part C: Emerging Technologies 15 (2007) 312–328.

[16] C. Sammut, G. I. Webb (Eds.), Viterbi Algorithm, Springer US, Boston, MA, 2010, pp. 1025–1025.

[17] K. Kempinska, T. Davies, J. Shawe-Taylor, Probabilistic map-matching using particle filters, 2016.

[18] P. P. Choi, M. Hebert, Learning and predicting moving object trajectory: a piecewise trajectory segment approach, Robotics Institute 337 (2006).

[19] B. Krekelberg, M. Lappe, Temporal recruitment along the trajectory of moving objects and the perception of position, Vision Research 39 (1999).

[20] A. Elnagar, K. Gupta, Motion prediction of moving objects based on autoregressive model, IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 28 (1998) 803–810.

[21] E. N. de Souza, K. Boerder, S. Matwin, B. Worm, Improving fishing pattern detection from satellite ais using data mining and machine learning, PLOS ONE 11 (2016) 1–20.

[22] A. Soares Júnior, B. N. Moreno, V. C. Times, S. Matwin, L. d. A. F. Cabral, Grasp-uts: an algorithm for unsupervised trajectory segmentation, International Journal of Geographical Information Science 29 (2015) 46–68.