Comparing Vessel Trajectories using Geographical Domain Knowledge and Alignments

Gerben K.D. de Vries
Informatics Institute
University of Amsterdam
Sciencepark 904, 1098 XH
Amsterdam, the Netherlands
Email: G.K.D.deVries@uva.nl

Willem Robert van Hage
Computer Science
VU University Amsterdam
de Boelelaan 1081a, 1081 HV
Amsterdam, the Netherlands
Email: wrvhage@few.vu.nl

Maarten van Someren
Informatics Institute
University of Amsterdam
Sciencepark 904, 1098 XH
Amsterdam, the Netherlands
Email: M.W.vanSomeren@uva.nl

***Abstract*—This paper presents a similarity measure that combines low-level trajectory information with geographical domain knowledge to compare vessel trajectories. The similarity measure is largely based on alignment techniques. In a clustering experiment we show how the measure can be used to discover behavior concepts in vessel trajectory data that are dependent both on the low-level trajectories and the domain knowledge. We also apply this measure in a classification task to predict the type of vessel. In this task the combined measure performs better than similarities based on domain knowledge or low-level information alone.**

***Index Terms*—vessel trajectories; trajectory alignments; geographical domain knowledge; trajectory clustering**

I. Introduction

To gain insight into the behavior of moving objects, such as vessels, it can be useful to cluster their trajectories into groups of similar movement patterns. Moving object trajectories are a kind of multivariate time-series. They have the property that they are usually different in temporal length, distance traveled and the number of data points. Alignment methods, such as dynamic time warping or edit distance, are designed to handle these kinds of variations. Thus, such methods can make suitable similarity measures for clustering moving object trajectories.

The trajectories of moving objects exist in spaces where places and regions have semantics of their own. In the case of vessels there are concepts such as anchoring areas, sea lanes and harbors. If we add this information to the trajectories and incorporate it in the similarity measure, then we can potentially discover more complex and interesting behavior patterns.

A simple example of such a pattern is the trajectories of large cargo ships and tankers that use a shipping lane to sail north. Using the shipping lane information the trajectories of these vessels can be discriminated from other vessels moving in that direction, such as the fishing vessels that do not use a shipping lane to go fishing up north. Also different types of ports, e.g. petrol docks and cargo terminals, can help us discriminate between the trajectories of tankers and those of cargo vessels, which look really similar in terms of movement.

In this paper we present an alignment based similarity measure that combines low-level vessel trajectories with geographical information, such as the name and type of the regions that vessels pass through and where they stop. We use this similarity measure in a clustering experiment and a classification task. The clustering experiment shows a number of interesting vessel behaviors that are discovered based on the combination of trajectories and geographical domain knowledge. In the classification task we try to predict the type of the vessel, e.g. tanker, tug or cargo vessel, that a trajectory belongs to. Again, we see the power of the combination of trajectories and domain knowledge, since this combination gives the best average classification accuracy. For both clustering and classification we use kernel based algorithms, so our measure is presented in the form of a kernel.

Our work has similarities with [2], where the authors use a series of clusterings with different similarity measures to discover interesting movement patterns. In this paper we incorporate more domain knowledge and also provide the combination of trajectory and domain knowledge as one measure where weights can be used to determine the influence of each component.

There are more papers researching the problem of clustering trajectories, mostly coming from the Moving Object Database community. The authors of [17] take an alignment based approach to clustering, as we do. In [11] the distance between trajectories is computed based on the area between them. And in [10] the authors use compression based on the minimum description length principle to simplify trajectories. Then all three above use a density based clustering algorithm. Using a density based clustering algorithm assumes that clusters are not (densely) connected. However, we do not know whether this is actually the case when using the complex alignment based similarity that we define below, therefore, we prefer a k-means based method. In [13] trajectories are first converted into a grid based representation and then clustered using fuzzy c-means. Using a c-means method is closer to our approach, however we do not use a grid representation.

The rest of this paper is organized as follows. In Section II we will describe the geographical domain knowledge that we have used. Then we will define trajectories and how we enriched them with the domain knowledge in Section III. Our similarity measure for low-level trajectories labeled with geographical information is described in Section IV. The

clustering and classification experiments that we on vessel trajectory data, using this measure, are presented in Section V. We end with some conclusions and suggestions for future work.

II. Geographical Domain Knowledge

Our geographical domain knowledge comes in the form of two simple ontologies. Both ontologies are stored as RDF. One ontology contains the definition of different anchorages, clearways and other areas at sea, which we call $\mathrm{AnchoragesAndClearways}$. All of these geographical features were converted to RDF from shape files from Rijkswaterstaat (RWS), part of the Netherlands Ministry of Transport, Public Works and Water Management. The other ontology has definitions for different types of harbors, such as liquid bulk and general cargo (containers), which we call $\mathrm{Harbors}$. All harbors were manually copied from the harbor branches map of the Port of Rotterdam Authority.[1] The concepts in these ontologies have a unique identifier, are assigned polygon regions, and have a type.

The modeling of the concepts follows the Geonames ontology, with the exception of the positioning properties ($\mathrm{wgs84} : \{\mathrm{lat}|\mathrm{long}\}$) and the type property ($\mathrm{geo} : \mathrm{featureCode}$). Geonames specifies feature types with the $\mathrm{geo} : \mathrm{featureClass}$ property for general classes, like $\mathrm{geo} : \mathrm{P}$ for populated feature types and geo:H for hydrographical feature types. Specific types, like $\mathrm{geo} : \mathrm{H.HBR}$ for harbors, are specified with the $\mathrm{geo} : \mathrm{featureCode}$ property. For our experiment we require more specific types than just harbor, e.g. dry bulk harbor. We assigned these specific types as extra types to the features. The specific types are modeled as subclasses of the original $\mathrm{geo} : \mathrm{featureCodes}$. To allow RDFS reasoning over the $\mathrm{featureCodes}$ and their new subclasses we temporarily asserted that $\mathrm{geo} : \mathrm{featureCode}$ is a subproperty of $\mathrm{rdf} : \mathrm{type}$, which makes each $\mathrm{featureCode}$ an $\mathrm{rdfs} : \mathrm{Class}$ containing all the features of that type as instances. Geonames uses WGS84 latitude longitude coordinates, while we use polygons of WGS84 coordinates which we express in the GeoRSS Simple vocabulary[2].

The polygons that define the different regions can be overlapping. For example, an anchorage area can overlap with a harbor approach. Each of the harbor regions is assigned a polygon demarcating the land area of the harbor (the port) and not the part of the water (the dock), because the same dock can be shared by two ports of different types. For instance, there can be container cranes on one side of the basin and oil valves on the other. This is not the case for harbors found in Geonames, because these are located by points in the middle of the dock. An example of the representation of a harbor with a specific type and polygon shape can be found in Figure 1.

We have created two web services to enrich trajectories with geographical features. One of these services, $\mathrm{NearestHarbor}$, matches a latitude, longitude point to the nearest harbor in $\mathrm{Harbors}$ within a predetermined range, the actual range used in this experiment will be discussed in Section V. The label and most specific type of this harbor is then returned, e.g. 'place_DryBulk4' and 'ptype_DryBulkHarbor'. Similarly, the other service, $\mathrm{Intersection}$, returns a set of label, most specific type pairs corresponding to the regions in $\mathrm{AnchoragesAndClearways}$ that intersect with a given point. Both web services were implemented in SWI-Prolog using the Space package [15].

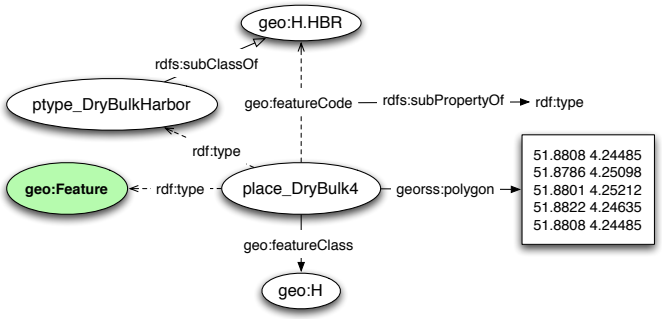

Fig. 1. Example of the RDF representation of the geographical domain knowledge using a combination of the Geonames ontology and GeoRSS. This figure shows a dry bulk cargo harbor.

We represent the geographical domain knowledge in RDF using the Geonames and GeoRSS ontologies. Besides these ontologies, the SWI-Prolog Space package also supports other geolocation representations in RDF, like those used by DBpedia,[3] Freebase[4] and LinkedGeoData (OpenStreetMap).[5] Therefore the services that match the trajectories to geographical features could just as well use RDF from these sources by directly loading them over the web. The main reason to use RWS sea maps and manually converted Port of Rotterdam harbor types is that currently there are hardly any maritime polygons to be found on the web. Moreover, the maritime features that do exist are not of the suitable level of abstraction. In the case of Geonames, the lowest level of abstraction is often not low enough, as discussed before, while in LinkedGeoData the existing levels of abstractions are too low or inappropriate. For example, harbors are categorized as leisure areas for angling, and each separate trash bin is listed as such.

III. Trajectories

Vessel trajectories are an example of moving object trajectories. A moving object trajectory in 2-dimensional space is represented by a sequence of vectors $\langle x_1, y_1, t_1\rangle, \ldots, \langle x_n, y_n, t_n\rangle$. Where x_i and y_i represent the position of the object at time t_i. However, in this paper we ignore the temporal dimension and consider a trajectory as $T = \langle x_1, y_1\rangle, \ldots, \langle x_n, y_n\rangle$. The length of a trajectory, i.e. the number of vectors, is denoted as: $|T|$. Furthermore, let $T(i) = \langle x_i, y_i\rangle$. The sample rate of trajectories is not fixed, thus the temporal difference between

[1] http://www.portofrotterdam.com/en/Port/port-maps/Pages/branches.aspx
[2] See http://www.georss.org/simple and http://www.georss.org/rdf_rss1
[3] http://dbpedia.org
[4] http://freebase.com
[5] http://linkedgeodata.org

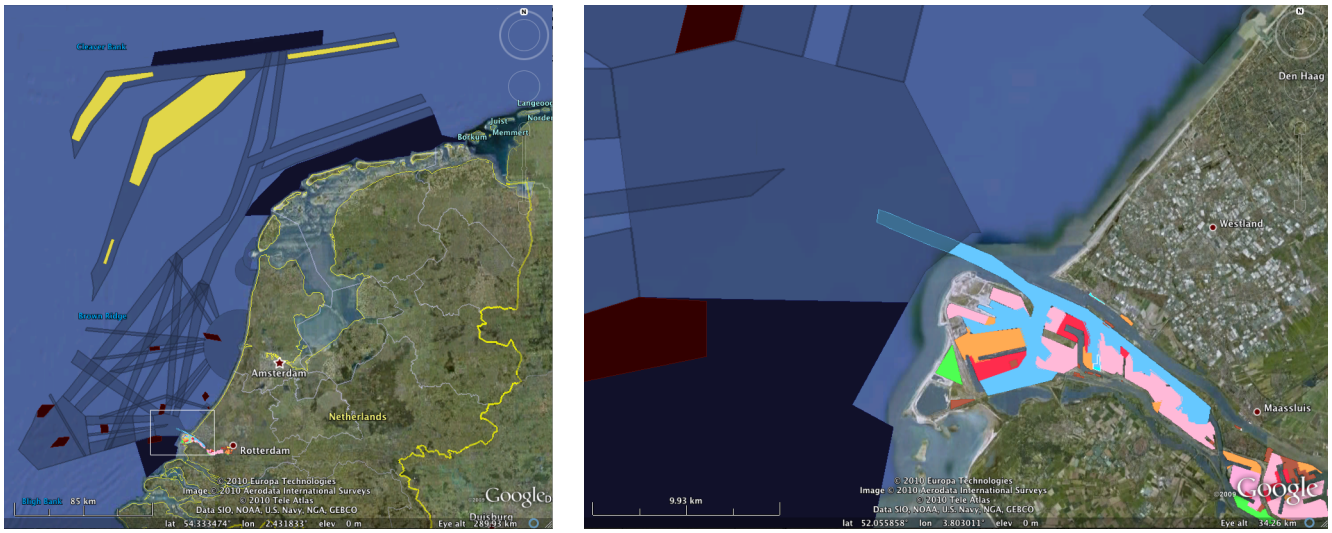

Fig. 2. Visualization of the geographical domain knowledge in KML. All the clearways and approach areas (translucent), anchorages (dark red), restriction zones (dark blue), and separation zones (yellow) are shown in the picture on the left. The various harbor types and the deep water lane (light blue) for large vessels is shown in the picture on the right, which corresponds to the rectangular area outlined with a white line in the picture on the left.

consecutive samples $\langle x_i, y_i \rangle$ and $\langle x_{i+1}, y_{i+1} \rangle$ is not the same. Also, there are more dimensions to trajectories that can be derived from the x, y, t information that we do not consider, such as speed and direction. In some tasks and applications these attributes might be more relevant than the absolute position x, y. In principle these dimensions could just be added. In the following we refer to a vector $\langle x_i, y_i \rangle$ as trajectory element or point.[6] In the *stop* and *move* model of [14], the trajectories in our experiment are moves. They are delimited by the vessel entering the area of observation or starting (from being stopped) and the vessel leaving the area of observation or stopping.

Using the $\mathrm{Intersection}$ service we create a sequence of sets of geo-labels $T^L = L_1, \ldots, L_{|T|}$ for a trajectory T, where $L_i = \{(\mathrm{name}_1, \mathrm{type}_1), \ldots, (\mathrm{name}_m, \mathrm{type}_m)\}$. Thus, each L_i is a set of pairs where name_j is the label of the region such that $T(i)$ is contained in the polygon that defines that region and type_j is the type of that region. Let $T^L(i) = L_i$. L_i is a set because a point can be in multiple regions. L_i can be empty since the defined regions do not cover everything.

We treat the start $T(1)$ and end $T(|T|)$ of a trajectory with special interest and define for a trajectory T the objects $T^{\mathrm{start}} = (\mathrm{stopped}, L_{\mathrm{start}})$ and $T^{\mathrm{end}} = (\mathrm{stopped}, L_{\mathrm{end}})$. $\mathrm{stopped}$ is a boolean value indicating whether the vessel is stopped. L_{start}, respectively L_{end}, is a set of pairs $(\mathrm{name}, \mathrm{type})$. To add domain knowledge about whether a vessel is docked at a port we use the $\mathrm{NearestHarbor}$ service to find the geographically closest harbor $(\mathrm{name}, \mathrm{type})$ to the point $T(1)$, respectively $T(|T|)$. If this service returns a harbor and the vessel is also stopped, then we put this pair in L_{start}, respectively L_{end}. If the vessel is stopped but there is no harbor close, then we use the $\mathrm{Intersection}$ service to find the regions that $T(1)$, respectively $T(|T|)$, is in, and add those to L_{start}, respectively L_{end}. We do the same if the vessel is not stopped. Thus, we are interested in harbors if a vessel is docked there, where docked is defined as being close to that harbor and stopped. We could have also added the start and stop harbors to the sequence of geo-labels T^L, however, by treating these separately we have more flexibility in weighing the importance of the start and stop harbors.

So, for each trajectory we have four objects, the trajectory itself, T, a sequence of sets of geo-labels, T^L, and start and end information, T^{start} and T^{end}.

The above labeling process has similarities with the work in [1]. However, we label not only the stops (or starts and ends in our terminology), but also the moves. Furthermore we use RDF based web services instead of a geographic database.

IV. Trajectory Similarity

Both the trajectory T and the sequence of sets of geo-labels T^L are sequences. Similarity between sequences can be computed using alignment methods, such as dynamic time warping and edit distance. In previous work [6] we studied the performance of different alignment measures on a vessel trajectory clustering task under the influence of trajectory compression. The best performing alignment in this work was a form of edit distance, similar to [4]. We define this alignment below, with notation taken from [16].

An edit distance alignment π, possibly with gaps, of $p \geq 0$ positions between two sequences S and T is a pair of p-tuples:

$$\pi = ((\pi_1(1), \ldots, \pi_1(p)), (\pi_2(1), \ldots, \pi_2(p))) \in \mathbb{N}^{2p} \quad , \quad (1)$$

which furthermore has the constraints that

$$1 \leq \pi_1(1) < \pi_1(2) < \ldots < \pi_1(p) \leq |S| \quad , \quad (2)$$

$$1 \leq \pi_2(1) < \pi_2(2) < \ldots < \pi_2(p) \leq |T| \quad . \quad (3)$$

[6] We use the generic symbols x and y, but in our experiment, depending on the context, these are longitude and latitude coordinates or projections of these coordinates to an x, y-plane.

This means that not all elements of the two sequences have to be aligned and there is no repetition of elements.

The edit distance score for an alignment π of two sequences S and T is equal to:

$$\begin{aligned} s(\pi) &= \sum_{i=1}^{|\pi|} \mathbf{sub}(S(\pi_1(i)), T(\pi_2(i))) \\ &+ g(|S| - |\pi|) + g(|T| - |\pi|) \ . \end{aligned} \quad (4)$$

To compute this score a substitution function $\mathbf{sub}$ is needed that gives the cost of substituting an element of S with an element of T. We also need a gap penalty g, which represents the cost of not aligning an element.

The substitution function for trajectories T, defined below, is simply the negative of the Euclidean distance between the two elements $\langle x_i, y_i \rangle$ and $\langle x_j, y_j \rangle$.

$$\mathbf{sub}_{\text{traj}}(\langle x_i, y_i \rangle, \langle x_j, y_j \rangle) = -\|\langle x_i - x_j, y_i - y_j \rangle\| \quad (5)$$

In previous work [6] on the same type of data and in the same domain we found a good value for g and we use this here.

In case of the similarity between sequences of sets of geo-labels T^L, the substitution function in Equation 6 expresses how many labels the sets of labels L_i and L_j have in common.

$$\mathbf{sub}_{\text{lab}}(L_i, L_j) = \frac{\sum_{l \in L_i, k \in L_j} ([l_1 = k_1] + [l_2 = k_2])}{2 \max(|L_i|, |L_j|)} - 1 \quad (6)$$

Note the use of the Iverson brackets ([]) which return 1 if the condition in between is satisfied and 0 otherwise. Furthermore, l_1 and k_1 indicate the first element of the $(\text{name}, \text{type})$ pair, i.e. the name, and l_2 and k_2 the second element, i.e. the type. Thus, we count the number of names and type labels that both sets have in common and divide this by the maximum that the sets can have in common. We subtract 1 to get the maximum penalty when there are no labels in common. If both L_i and L_j are empty, the score is -1. We set $g = -1$, the same as the maximum penalty for a substitution.

The similarity between two sequences S and T is the value of the function s (Equation 4) for the alignment that maximizes the function s for the set of all possible alignments between these sequences $\Pi_{S,T}$. Thus,

$$Sim(S, T) = \frac{\max_{\pi \in \Pi_{S,T}} s(\pi)}{|S| + |T|} \ . \quad (7)$$

We divide by the sum of the lengths of both sequences to get an average similarity per element. We do this because this allows for better comparison between sequences of very different length, which are quite common in our dataset.

In the experiments we will use kernel based algorithms, hence we turn the above similarity into a kernel. For all sequences T_i and T_j in a set of sequences $\mathcal{T}$, we compute a kernel K in the following way. We first take

$$K(i, j) = Sim(T_i, T_j) \ , \quad (8)$$

and then we normalize K and make a kernel out of it by:

$$K = 1 - \frac{K}{\min(K)} \ . \quad (9)$$

We do this for trajectories T to get a kernel K_{traj} and for sequences of sets of geo-labels T^L to get a kernel K_{lab}.

Computing the similarity between two start/end objects is straightforward; it can immediately be put into kernel form. For all $(\text{stopped}_i, L_i)$ and $(\text{stopped}_j, L_j)$ in a set of start/end objects, we compute a kernel as:

$$K(i, j) = 1 - \frac{[\text{stopped}_i \neq \text{stopped}_j] + \mathbf{sub}_{\text{lab}}(L_i, L_j)}{2} \ . \quad (10)$$

Thus the similarity between two start/end objects is determined by whether the vessel is stopped or not and how much labels there are in common. Using Equation 10 we get a kernel K_{start} for the start objects and a kernel K_{end} for the end objects.

The four kernels defined above are combined together into a new kernel by taking the weighted sum:

$$K_{\text{all}} = w_1 K_{\text{traj}} + w_2 K_{\text{lab}} + w_3 K_{\text{start}} + w_4 K_{\text{end}} \ . \quad (11)$$

To keep the kernel properly normalized the weights should sum to 1. Clearly, this kernel is symmetric, but it is not guaranteed to be positive semi-definite (PSD). However, this does not prevent us from using it in our clustering and classification tasks. The weighted kernel is inspired by work in computational biology on combined kernels for comparing protein sequences and DNA [5].

The K_{start} and K_{end} sub-kernels are cheap to compute. For K_{traj} and K_{lab} we use the dynamic programming approach that is common for edit distances.

V. Experiments

In this section we present the dataset and the clustering and classification experiments that we used this dataset in.

A. Dataset

Our dataset consists of 1917 vessel trajectories in a 50km radius around the Port of Rotterdam, collected using the Automatic Identification System (AIS). The position data of these vessels was originally recorded in a latitude, longitude format, but has been converted, using a suitable map projection, to allow for normal Euclidean geometry in compression and alignment. These ship trajectories are compressed with a trajectory compression algorithm [8]. We took the compression settings from earlier work [6], in which we showed that compression actually improves performance on a vessel trajectory clustering task. Under the used settings the amount of data is reduced by 95%, which reduces computation time for trajectory similarity drastically.

Within the above mentioned 50km radius, the $\text{AnchoragesAndClearways}$ ontology contains the names and polygons for approximately 50 regions of 6 different types. There are around 90 different harbors in Harbors that are distinguished into 7 different types. For each of the 1917

trajectories we created a sequence of sets of geo-labels, a start object and an end object. The threshold used in the $\mathrm{NearestHarbor}$ service is set to 100m. This threshold range was determined manually and is suitable given the size of the vessels, docks and clearways, and is of a larger order of magnitude than the GPS and compression errors. Due to the use of a trajectory compression algorithm there is the potential risk that a trajectory is not labeled with a region that a vessel has actually passed through. However, under the used compression settings, the amount of points that are retained is high compared to the number of traversed regions. Moreover, visual inspection of some of the labeled trajectories suggests that no regions are missed.

B. Clustering

In the first experiment we use our similarity measure for clustering the dataset. We evaluate the clustering by giving a number of examples of clusters that are the result of combining trajectories with geographical domain knowledge.

1) Setup: One of the more recent, flexible and advanced clustering algorithms is spectral clustering [12]. This algorithm can deal well with arbitrarily shaped clusters and clusters with different densities. Clustering is done by treating the similarity matrix as graph adjacency matrix and computing a type of graph cut, for instance the normalized one. Just as with standard k-means, this algorithm requires a parameter k for the number of clusters. Ordinarily spectral clustering involves the computation of eigenvectors, which can sometimes be computationally problematic. In [7] the authors show how a weighted kernel k-means approach can be used to do spectral clustering without computing eigenvectors. In our experiments we will use weighted kernel k-means to compute the normalized graph cut. Therefore, we have defined our similarity measure as a kernel. It is remarked in [7] that, if a kernel is not PSD, then the weighted kernel k-means algorithm is not guaranteed to converge. However, clustering results are often better with non-PSD kernels. To overcome the potential non-convergence problem, and the fact that k-means can get stuck in local optima, we run the weighted kernel k-means algorithm a number of times (100), with different random initializations. As in regular k-means, we keep the result that has the lowest inter cluster spread.

In the following we will show three clustering examples for three different settings of the weights in equation 11, thus, for three different kernels. The first setting is $w_1 = \frac{1}{2}, w_2, w_3, w_4 = \frac{1}{6}$, the resulting kernel being K_{comb}. This setting weighs the trajectory information and the ontological information equally. There is also a setting for just the trajectory information, $w_1 = 1, w_2, w_3, w_4 = 0$, K_{traj}, and one for just the ontological information $w_1 = 0, w_2, w_3, w_4 = \frac{1}{3}$, K_{onto}. We define these three distinct settings to investigate the effect of using low-level trajectory information and domain knowledge in clustering.

We have no gold standard clustering that we wish to achieve or a specific criterion that we want to optimize, thus it is difficult to determine a good value for the number of clusters parameter k. Therefore, we manually experimented with different values and finally selected $k = 40$, which gave cluster examples that show the differences between the three kernels well. We could have also used a clustering algorithm that has no parameter k, e.g. density based ones are popular in combination with moving object trajectories. However, these algorithms have other parameters that need to be determined, which also is a manual process. Moreover, we are not sure that the combined kernel that we have defined induces a feature space in which density based algorithms work well.

2) Examples: The three examples we give below illustrate behavior clusters of vessels that arise in the combined setting, i.e. using kernel K_{comb}. For each example we will also show the clusters from the other two settings (K_{traj} and K_{onto}) that resemble the behavior the most. All figures show the trajectories in one cluster in black against a background of all trajectories in gray. For the trajectories in a cluster, the start of a trajectory is indicated by a dot and the end by an asterix.

Figure 3 illustrates the behavior of vessels anchoring in a specified anchoring area. In Figure 3A we show a cluster resulting from the combined information kernel K_{comb}. We see that all the tracks end up in one anchoring area. If we use only the trajectory information, i.e. K_{traj}, we get the result in Figure 3B. In this case there are a number of other trajectories that do not end in the anchoring area. For the clustering with only the ontological information, K_{onto}, we see something different (Figure 3C). Here there is another track of a vessel anchoring in another anchoring area. So, the combination of trajectory and ontological information results in the discovery of the behavior "anchoring in a specific anchoring area".

The cluster in 4A shows the docking behavior of vessels in a certain part of the harbor. There is some noise in the cluster, not all trajectories go to that part. This cluster is a result of clustering with the combined kernel K_{comb}. We see something similar in Figure 4B. However, here we have used the kernel K_{traj} and thus only the trajectory information. The result is that the cluster also contains trajectories starting from anchoring areas. This differs from the combined setting, where we only have trajectories coming from outside the observation area, i.e. the open sea. In Figure 4C, the ontology only setting, K_{onto}, we also have trajectories going to the harbor from outside the observation area, but, all trajectories stop in the deep water lane, not on a dock. This is somewhat odd and is the result of only considering ontological information. In the combined case we have stopped in the deep water lane, but also in the adjacent docks. Thus, the combined case shows the behavior of "docking in certain part of the harbor, coming directly from the open sea".

The trajectories in Figure 5A, on which we zoom in in Figure 5B, are a result of clustering with the combined kernel. The figures show trajectories that do not stop and continue on the river to the land behind. These vessels are smaller, and in Figure 5B we see that all of them do not pass through the deep water lane. Figure 5C is a cluster from clustering with the trajectory only kernel K_{traj}. The trajectories in this

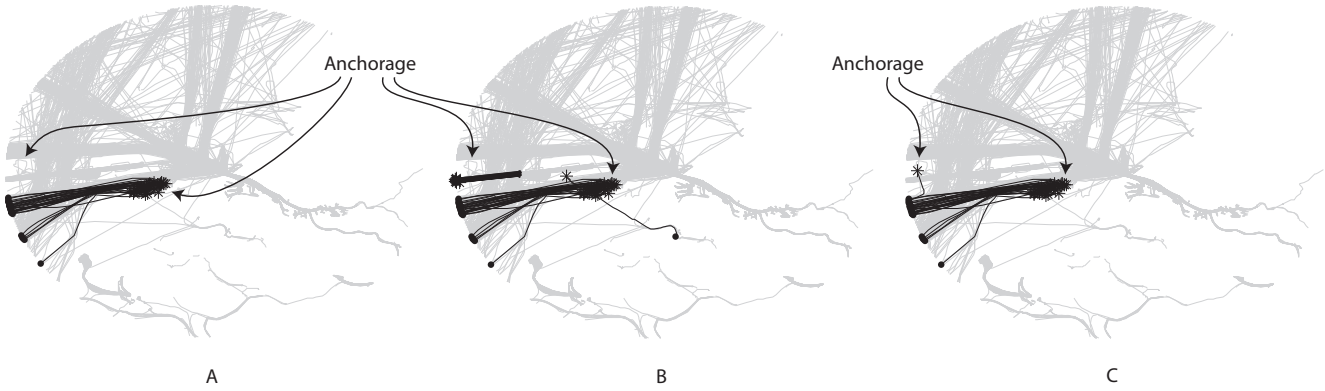

Fig. 3. Example of a cluster of trajectories showing anchoring behavior. The start of trajectory is indicated by a dot, the end by an asterix.

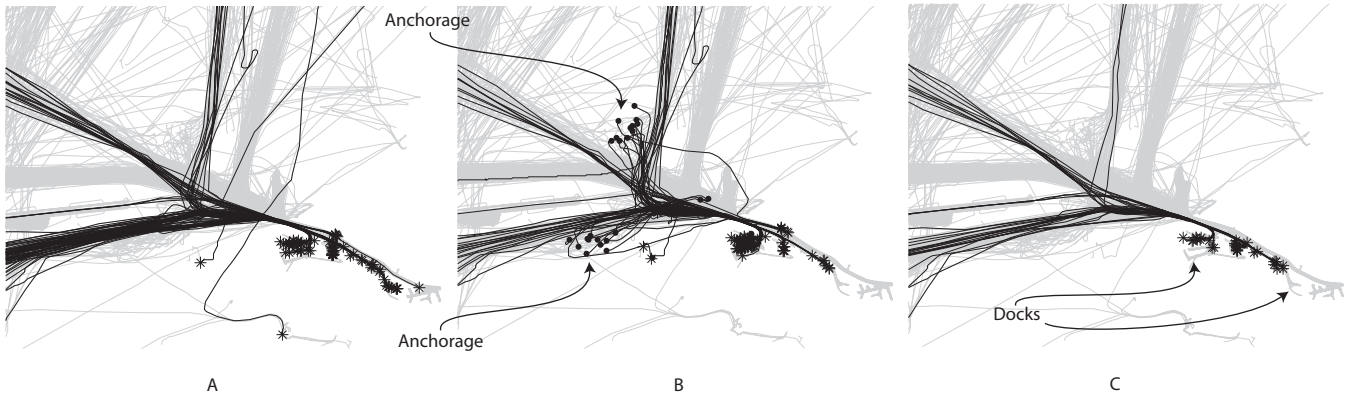

Fig. 4. Example of a cluster of trajectories showing docking behavior. The start of trajectory is indicated by a dot, the end by an asterix.

cluster both stop and go on and some of them go through the deep water lane and some of them do not. Because no domain knowledge is used the deep water lane and non-deep water lane trajectories are difficult to keep apart, since they do not differ much in shape. The comparable cluster for the K_{onto} kernel, Figure 5D, shows trajectories that go in different directions and some noise. Using only domain knowledge does not guarantee that trajectories that go in different directions are not put together. The combined kernel discovers the behavior of "smaller ships coming from sea and continuing directly to the land behind".

The above examples show that a combination of low-level trajectory information and geographical domain knowledge in one similarity measure can lead to the discovery of interesting vessel behavior patterns that are indeed due to a combination of these two information sources.

C. Classification

Because quantitatively evaluating the quality of clustering, and hence the utility of our similarity measure, is difficult without a gold standard, we also defined a classification task.

In our similarity measure and thus the clustering task, we did not use any information about the type of the vessel. But, because AIS also contains ship type information, all the 1917 trajectories are in fact labeled with one of 18 different vessel types. Inspection of the clustering results suggests that our similarity measure can also be used in a classification setting to identify these vessel types. The 18 types include tanker, cargo ship, pilot vessel and tug. The distribution of the types is skewed, with the largest class containing 34% of the dataset and the smallest class containing just 1 example.

With the similarity already defined as a kernel, we used a support vector machine (SVM) as our classifier. Even though the kernels are non-PSD, this is not problematic for using them in an SVM context [9], though convergence is not guaranteed. For a number of weight settings we created kernels using equation 11. There are 5 settings for kernels that only use domain knowledge, 1 kernel that uses only low-level trajectory information and 2 kernels that combine domain knowledge and trajectory information. Among these kernels there are also the three variants that we used in the clustering experiment.

We train classifiers with the different kernels using the LibSVM [3] package for MatLab, version 2.91. Apart from the fact that we use precomputed kernels, all settings are on default. For each kernel we do a 10 fold cross validation

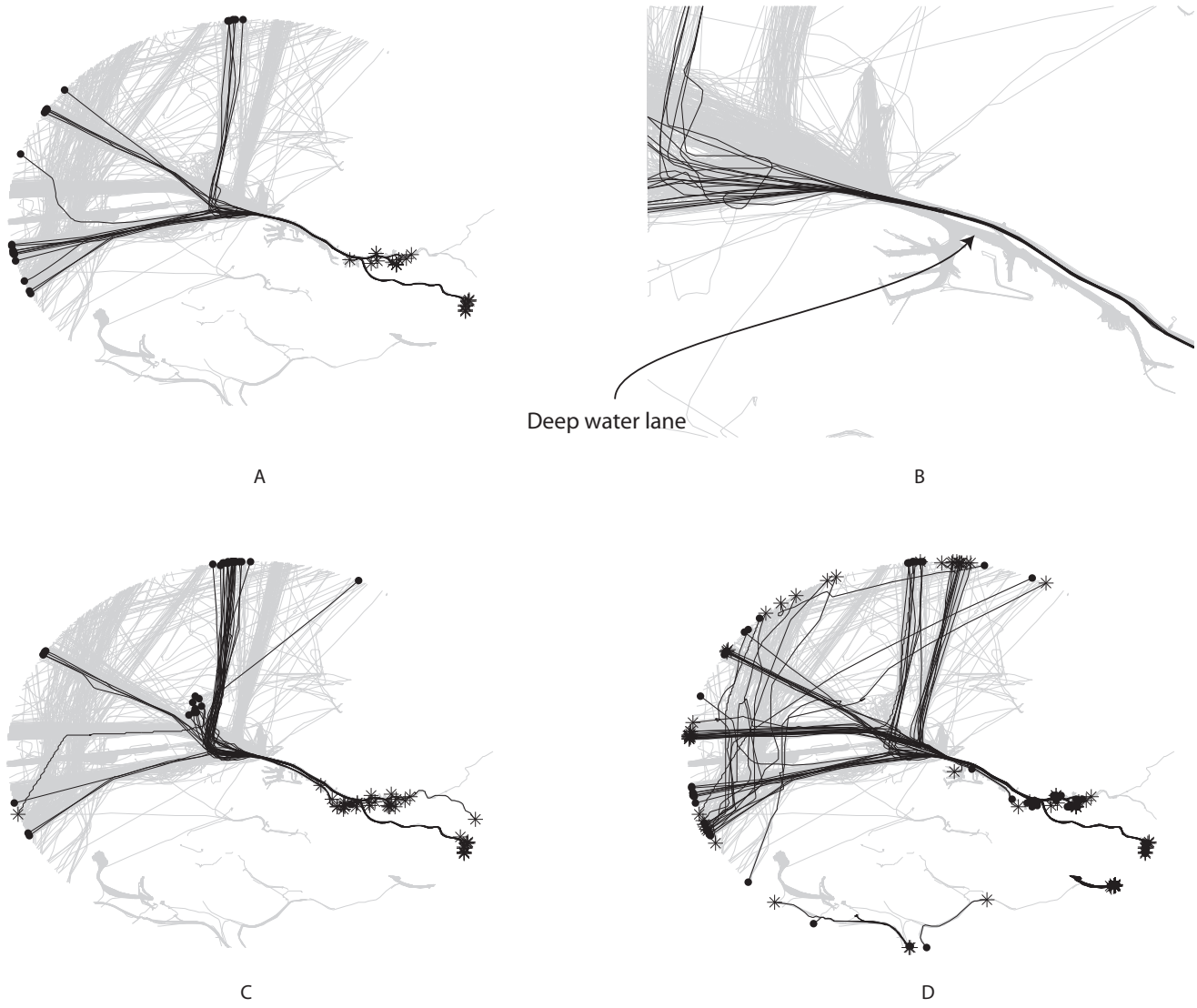

Fig. 5. Example of a cluster of trajectories continuing through the harbor, not going through the deep water lane. The start of trajectory is indicated by a dot, the end by an asterix.

experiment. We compare the results for each kernel with each other kernel using a standard paired t-test with $p < 0.05$. In Table I we show the average classification accuracy results for the different weight settings. The results indicated with an asterix (*) do not differ significantly from each other under the described t-test. All other results do differ significantly.

From Table I we can see that the kernels that combine domain knowledge and trajectory information achieve the best performance in terms of classification accuracy, significantly outperforming the other kernels. From these two kernels, the equal weight, $w_1 = \frac{1}{2}, w_2, w_3, w_4 = \frac{1}{6}$, kernel has the highest accuracy. Using only trajectory information ($w_1 = 1$) gives better classification accuracy results than using only geographical domain knowledge ($w_1 = 0$). Among the kernels that use only domain knowledge, the kernel that combines the different types of knowledge ($w_2, w_3, w_4 = \frac{1}{3}$) significantly outperforms the other four domain knowledge-only kernels.

TABLE I
AVERAGE CLASSIFICATION ACCURACY USING AN SVM FOR DIFFERENT KERNEL WEIGHT SETTINGS.

w_1	w_2	w_3	w_4	Average Accuracy
0	1	0	0	52.8%*
0	0	1	0	54.8%*
0	0	0	1	55.1%*
0	0	$\frac{1}{2}$	$\frac{1}{2}$	62.1%
0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	66.1%
1	0	0	0	72.2%
$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	74.4%
$\frac{1}{2}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	75.4%

Thus, combining low trajectory information and domain

knowledge into one kernel gives significantly better accuracy results on the vessel type classification task than using either the trajectory information or the geographical domain knowledge on its own.

VI. Conclusion & Future Work

In this paper we have defined a similarity measure between vessel trajectories enriched with geographical domain knowledge. This similarity measure consists of an edit distance alignment for trajectories and one for sequences of sets of geo-labels, combined with information about the start and end of a trajectory. We applied this similarity in a clustering task and gave examples of discovered interesting vessel behavior that is a combination of trajectory information and domain knowledge. The similarity was also used in a classification setting to predict vessel types where the combined similarity showed the best performance in terms of classification accuracy.

Evaluating the performance of clustering is difficult. In future work we would like to let domain experts label interesting vessel behavior in order to be able to better evaluate the quality of the clustering. We also plan to apply the measure in the task of outlier detection to discover strange vessel behavior. However, like clustering, this task is difficult to evaluate without labeled data. Furthermore, we are interested in applying this kind of similarity in other domains where comparable domain knowledge exists. The domain knowledge is a parameter of the similarity measure that can be varied, so the quality and content of this information is of direct influence on the similarity. So, next to applying this similarity in other domains we should also consider other ontologies in the same domain.

Currently, low-level trajectories and the sequences of sets of geo-labels are both separately compared using an edit distance. They are, however, two sequences with the same number of elements. So, we also want to investigate the result of integrating these two into one sequence of position and set of geo-label pairs and apply only one edit distance.

Both in the clustering and classification tasks, more complicated weighting schemes are possible for creating kernels with Equation 11. In the classification setting finding the optimal weights can be done automatically. However, for clustering this is more difficult, and playing around with weight settings is something left to a domain expert or end-user.

Acknowledgment

The authors wish to thank Véronique Malaisé for help with constructing the ontologies and the harbor polygon definitions. We are also grateful for the comments of the anonymous reviewers that have greatly helped to improve this paper.

This work has been carried out as a part of the Poseidon project in collaboration with Thales Nederland, under the responsibilities of the Embedded Systems Institute (ESI). This project is partially supported by the Dutch Ministry of Economic Affairs under the BSIK program.

References

[1] L. O. Alvares, V. Bogorny, B. Kuijpers, J. A. F. de Macedo, B. Moelans, and A. Vaisman, "A model for enriching trajectories with semantic geographical information," in *GIS '07: Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*. New York, NY, USA: ACM, 2007, pp. 1–8.

[2] G. Andrienko, N. Andrienko, and S. Wrobel, "Visual analytics tools for analysis of movement data," *SIGKDD Explor. Newsl.*, vol. 9, no. 2, pp. 38–46, 2007.

[3] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[4] L. Chen and R. Ng, "On the marriage of lp-norms and edit distance," in *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*. VLDB Endowment, 2004, pp. 792–803.

[5] M. Cuturi, *Positive Definite Kernels in Machine Learning*, 2010, available at http://www.princeton.edu/~mcuturi/pub.html.

[6] G. de Vries and M. van Someren, "Clustering vessel trajectories with alignment kernels under trajectory compression," in *ECML/PKDD (1)*, ser. Lecture Notes in Computer Science, J. L. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, Eds., vol. 6321. Springer, 2010, pp. 296–311.

[7] I. S. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors – a multilevel approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 1944–1957, 2007.

[8] J. Gudmundsson, J. Katajainen, D. Merrick, C. Ong, and T. Wolle, "Compressing spatio-temporal trajectories," *Computational geometry*, vol. 42, no. 9, pp. 825–841, 2009.

[9] B. Haasdonk, "Feature space interpretation of svms with indefinite kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 4, pp. 482–492, 2005.

[10] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," in *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2007, pp. 593–604.

[11] M. Nanni and D. Pedreschi, "Time-focused clustering of trajectories of moving objects," *J. Intell. Inf. Syst.*, vol. 27, no. 3, pp. 267–289, 2006.

[12] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NIPS*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 2001, pp. 849–856.

[13] N. Pelekis, I. Kopanakis, E. Kotsifakos, E. Frentzos, and Y. Theodoridis, "Clustering trajectories of moving objects in an uncertain world," in *ICDM '09: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 417–427.

[14] S. Spaccapietra, C. Parent, M. L. Damiani, J. A. F. de Macêdo, F. Porto, and C. Vangenot, "A conceptual view on trajectories," *Data & knowledge engineering*, vol. 65, no. 1, pp. 126–146, 2008.

[15] W. R. van Hage, J. Wielemaker, and G. Schreiber, "The space package: Tight integration between space and semantics," *T. GIS*, vol. 14, no. 2, pp. 131–146, 2010.

[16] J.-P. Vert, H. Saigo, and T. Akutsu, "Local alignment kernels for biological sequences," in *Kernel Methods in Computational Biology*, B. Scholkopf, K. Tsuda, and J.-P. Vert, Eds. The MIT Press, 2004, pp. 131–154.

[17] M. Vlachos, D. Gunopoulos, and G. Kollios, "Discovering similar multidimensional trajectories," in *ICDE '02: Proceedings of the 18th International Conference on Data Engineering*. Washington, DC, USA: IEEE Computer Society, 2002, p. 673.