NTNU
Norwegian University of
Science and Technology

AIS-based Vessel Trajectory Prediction for ASV Collision Avoidance

Simen Hexeberg

Master of Science in Cybernetics and Robotics
Submission date: June 2017
Supervisor: Edmund Førland Brekke, ITK
Co-supervisor: Bjørn-Olav Holtung Eriksen, ITK
Andreas Lindahl Flåten, ITK
Øystein Engelhardtsen, DNV GL

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem formulation

There has been an increased focus on development and research of autonomous surface vessels (ASVs) in the recent years. A reliable ASV has several advantages over manually operated vessels. There exist, for instance, a big potential of reducing human caused accidents at the sea. Also, the reduction of crew and crew facilities free extra cargo space, saving money and reducing emissions. A robust collision avoidance (COLAV) system is imperative for the public success and acceptance of ASVs. The constant velocity model (CVM) is the prevailing method in today's COLAV systems for predicting nearby vessels' future trajectories. Reduced frequency of close to collision situations and more optimal path planning in COLAV algorithms can be achieved if better predictions are obtained.

The main purpose of this thesis is to investigate to what extent historical automatic identification system (AIS) data can be utilized to predict future vessel trajectories. Real-world AIS data from the Trondheimsfjord is provided by DNV GL. The AIS based predictions are intended to work proactively to avoid collision situations. In a preliminary specialization project [1], a novel trajectory prediction method named the single point neighbor search (SPNS) method has been developed and briefly tested. The following tasks are proposed for this thesis:

1. Investigate and resolve several unexpected biases in the SPNS algorithm as developed in the specialization project [1].
2. More extensive testing of the SPNS algorithm:
   (a) Test the algorithm on a wider spectrum of scenarios in order to reveal shortcomings.
   (b) Test the performance on curved trajectories and compare it with the CVM method.
   (c) Test the performance on straight line trajectories and compare it with the CVM method.
3. Evaluate the potential of better AIS based speed predictions.
4. Generalize the method to handle branching of sea lanes. Also introduce uncertainty measures of the predictions.
5. Do a brief computational time analysis.

Preface

This thesis defines the end of my master degree in Engineering Cybernetics at the Norwegian University of Science and Technology(NTNU).

Working with real-world AIS data have been truly exciting. Especially the freedom of the task, which let me explore new paths towards better predictions of vessel trajectories, have been very motivating, though also challenging at times.

I would like to thank my supervisor Edmund F. Brekke together with my two co-supervisors at NTNU, Bjørn-Olav H. Eriksen and Andreas L. Flåten, for their engagement in my work including all the hours spent discussing and guiding me. I will also thank my DNV GL co-supervisor Øystein Engelhardtsen and senior researcher at DNV GL, Hans Anton Tvete, for assisting me with real-world AIS data. Last, but not least, I will thank Natallia for keeping my motivation up throughout the semester and for all the delicious meals and support.

Abstract

The automotive industry has already taken a big step towards fully autonomous vehicles. This trend is now spreading to the maritime industry with development of autonomous surface vessels (ASVs). A reliable collision avoidance (COLAV) system is essential in this context. In order to avoid collision with other vessels, one must predict the future trajectories of nearby vessels. The constant velocity model (CVM) is the prevailing approach for trajectory prediction in today's COLAV systems. The main purpose of this thesis is to investigate to what extent it is possible to predict future vessel trajectories based on historical automatic identification system (AIS) data for prediction horizons up to about 15 minutes. A survey of other relevant prediction methods are also presented.

Two new, AIS based methods for vessel trajectory prediction are developed and tested: the single point neighbor search (SPNS) method and the neighbor course distribution method (NCDM). Three speed prediction methods are also tested: the straightforward constant speed method, a method using the median speed of the predicted state's close neighbors (CNs) and lastly a linear transition in predicted time between the two former methods.

The SPNS method is compared to a CVM approach and yields significantly better results on curved trajectories, in terms of lower average and median path and trajectory errors. However, a major part of vessels' transit time is spent on straight line trajectories. The SPNS algorithm shows also good path predicting capabilities on close to straight line trajectories, although the CVM method yields the lowest errors in such environments. The SPNS algorithm outputs a single predicted trajectory which tends to follow the most AIS-dense sea lane ahead. Hence, it does neither facilitate any uncertainty measure nor the possibility to suggest multiple possible route choices. The more computational demanding NCDM algorithm, which outputs multiple predicted trajectories, does better facilitate prediction uncertainty and it is also capable of dividing the predicted trajectories into multiple branching sea lanes. Its predicted positions at certain time instants are clustered with the density-based spatial clustering of applications with noise (DBSCAN) algorithm. The predictions are further statistically evaluated with respect to the distances to the nearest cluster centers.

Lastly, a computation time analysis is presented. The computational time is reduced from an earlier version of the algorithm by storing data in a k-d tree. However, the NCDM algorithm with the tested decision parameters is not practically feasible in real-time. Several suggestions to reduce the computation time are given.

Sammendrag

Bilindustrien har allerede kommet et godt stykke på vei mot fullt autonome kjøretøyer. Denne trenden spres seg nå over til den maritime industrien med utviklingen av autonome overflate fartøyer (ASVer). Et pålitelig kollisjonsungåelsessystem (COLAV) er essensielt i denne sammenheng. For å unngå kollisjon med andre fartøyer må man predikere de fremtidige trajektorene til fartøyer i nærheten. Den konstante hastighetsmodellen (CVM) er den rådende fremgangsmåten for trajektor-prediksjon i dagens COLAV systemer. Hovedhypotesen som er undesrsøkt i denne masteroppgaven, er i hvilken grad det er mulig å predikere fremtidige fartøy-trajektorer basert på historiske automatisk identifikasjonssystem (AIS) data for prediksjonshorisonter opp til omtrent 15 minutter. En undersøkelse av andre relevante prediksjonsmetoder er også presentert.

To nye, AIS-baserte metoder for prediksjon av fartøy-trajektorer har blitt utviklet og testet: en metode kalt "single point neighbor search" (SPNS) og en metode kalt "neighbor course distribution method" (NCDM). Tre hastighetsprediksjonsmetoder er også testet: den trivielle konstante hastighetsprediksjonen, en metode som benytter median hastigheten til den predikerte tilstanden's nærmeste naboer (CNs) og til slutt en linear transisjon i predikert tid melom de to førstnevnte metodene.

SPNS metoden er sammenlignet med en CVM metode og gir signifikant bedre resultater på krumme trajektorer i form av lavere gjennomsnittlig og median bane- og trajektorfeil. Når det er sagt blir en stor andel av fartøyer's reisetid brukt i rettlinjede trajektorer. SPNS algoritmen viser også gode bane-følgende egenskaper på rettlinjede trajektorer, selv om CVM metoden gir de laveste feilene i slike omgivelser. SPNS algoritmen produserer en enkelt predikert trajektor som har en tendens til å følge de mest AIS-tette sjøleiene. SPNS algoritmen legger derfor hverken til rette for estimering av usikkerhet i prediksjonene eller for muligheten til å foreslå flere ulike rutealternativer. Den mer regnekraftskrevende NCDM algoritmen, som produserer mange predikerte trajektorer, legger bedre til rette for prediksjonsusikkerhet i tillegg til at den har egenskapen til å fordele de predikerte trajektorene i flere forgreinede sjøleier. Dens predikerte posisjoner ved gitte tidspunkter blir automatisk gruppert ved hjelp av "density-based spatial clustering of applications with noise" (DBSCAN) algoritmen. Prediksjonene er videre statistisk evaluert med tanke på avstandene til de nærmeste kluster-sentrene.

En kjøretidsanalyse er presentert avslutnignsvis. Kjøretiden har blitt redusert med en faktor på 186 i forhold til en tidligere versjon av algoritmen ved å lagre dataene i et k-d tre. Dette til tross er NCDM algoritmen, med de testede beslutningsparameterne, ikke praktisk gjennomførbar i sanntid. Flere forslag til reduksjon av kjøretiden er gitt.

Contents

List of Figures

List of Tables

List of Terms

AIS automatic identification system

ASV autonomous surface vessel

ASVs autonomous surface vessels

CN close neighbor

CNs close neighbors

COG course over ground

COLAV collision avoidance

COLREGS International Regulations for Preventing Collisions at Sea

CS constant speed

CVM constant velocity model

DBSCAN density-based spatial clustering of applications with noise

IMO International Maritime Organization

KB-PF knowledge based particle filter

KB-VM knowledge based velocity model

kNN k-nearest neighbor

LMNN large margin nearest neighbor

MDL minimum description length

MMSI maritime mobile service identity

NCDM neighbor course distribution method

SOG speed over ground

SOLAS safety of life at sea

SPNS single point neighbor search

ST	speed transition
TRACLUS	trajectory clustering
TREAD	traffic route extraction for anomaly detection

Chapter 1

Introduction

1.1 Motivation

The automotive industry have already come a long way in autonomy, with companies such as Google, Uber and Tesla in the very front of the fast development. This trend is spreading to the maritime industry with the development of autonomous surface vessels (ASVs). Reliable ASVs have potential for large economical and safety related benefits by eliminating errors caused by human operators, increasing cargo space as a result of eliminating crew facilities and by facilitating more optimal navigation and cooperation between vessels.

Safety and reliability are very important aspects for the public's and governments' acceptance of fully or partly autonomous vessels. A robust collision avoidance (COLAV) system is essential in this context. In order to avoid collision with other vessels, one must predict the future trajectories of nearby vessels. A commonly used approach today is to assume constant speed and course (e.g. [2]). By utilizing historical positional, course and speed information from automatic identification system (AIS) data, it may be possible to obtain more refined predictions. However, since patterns from historical AIS data best reflect normal vessel behavior, as opposed to behavior in close-to-collision situations, AIS based predictions may be mostly used for proactive maneuvers in order to prevent potential collision situations. Faster and more reliable data sources, such as radars, may be better choices if the situation is already critical. Better estimates of surrounding vessels' future trajectories can also be used to improve situational awareness systems.

At sea, the autonomy challenges differ from the ones on the roads. As opposed to cars on the roads, vessels can move in all directions. Additionally, the International Regulations for Preventing Collisions at Sea (COLREGS) is less quantifiable and more dependent on common sense than standard car traffic rules, making it even harder to predict future vessel behavior. On the other hand, vessel movement patterns revealed by historical AIS data, as illustrated in Figure 2.3, are likely to contain implicit information about where it is possible, safe and smart to maneuver, as a majority of vessels navigate in map-advised sea lanes and areas.

The main purpose of this thesis is to investigate to what extent it is possible to predict future vessel trajectories based on historical automatic identification system

(AIS) data.

1.2 Outline

The outline of this thesis is as follows:

Chapter 2 introduces the AIS dataset, including notations and some key findings from the dataset. Chapter 3 contains a survey of related trajectory prediction techniques. Further, Chapter 4 and Chapter 5 present the single point neighbor search (SPNS) method and the neighbor course distribution method (NCDM), respectively, before the methods' prediction performances are tested and evaluated in Chapter 6. Some concluding remarks and suggestions for further work is given in Chapter 7 and Chapter 8.

Chapter 2

AIS data

This chapter gives a brief introduction to automatic identification system (AIS) data in general in addition to an overview of the real-world AIS dataset given by DNV GL. Most of this chapter can also be found in [1], but is included here as well for completeness.

2.1 Automatic identification system data (AIS)

The automatic identification system (AIS) is an automatic tracking system used on ships and by vessel traffic services for identifying and locating vessels by electronically exchanging data with other nearby ships, AIS base stations and satellites. An AIS system is required installed on international voyaging ships with gross tonnage (GT) of 300 or more and on all passenger ships regardless of size according to the safety of life at sea (SOLAS) convention given by the International Maritime Organization (IMO) [3].

Vessels equipped with AIS transceivers (a device capable of both transmitting and receiving AIS signals) can exchange AIS data with other nearby vessels. The range is about 10-20 nautical miles which is limited by the range of VHF (very high frequency) signals at sea. However, the later years AIS transceivers have been installed on several satellites, yielding global AIS coverage. There are two classes of AIS messages: Class A and class B. Class A transceivers usually transmit at a higher rate than class B transceivers, ranging from 30 times per minute for high velocity vessels to every third minute for anchored or moored vessels.

A typical AIS message includes information such as vessel identification (MMSI/IMO number), positional coordinates, course over ground (COG) and speed over ground (SOG). Depending on the on-board systems and what is manually typed in by the crew, other information such as heading, ship type, ship status and ship dimensions may be available.

2.2 AIS dataset

DNV GL have provided real AIS data from the Trondheimsfjord. The dataset contain a total of 3 million AIS messages, gathered during the one year period from January

1st 2015 to December 31st 2015. A brief explanation of the parameters provided in the AIS message is shown in Table 2.1.

AIS parameter	**Explanation**
IMO	7 digit vessel identification number that remains unchanged upon transfer of the vessel's registration to another country [3].
MMSI	The maritime mobile service identity is a 9 digit, unique vessel identification number.
Long	Degrees longitude. Range: $[-180°W, 180°E]$
Lat	Degrees latitude. Range: $[90°S, 90°N]$
COG	The course over ground is the clockwise course of the vessel's velocity vector relative to true north.
SOG	The speed over ground is the absolute value of the vessel's velocity vector.
Heading	The direction of the vessel's nose or bow relative to true north. It is not necessarily equal to COG.
Timestamp	A number representing the number of days elapsed since 1.jan 1900, 00:00.[1]
Shiptype	Chosen from a set of predefined ship types from level 5 in Lloyds database. 70 different ship types are detected in the dataset.
Status	Navigational status message manually set by the crew. A total of 16 different statuses can be chosen, e.g. ”under way using engine”.

Table 2.1: Explanation of the AIS parameters in the dataset.

Although the IMO number is described in Table 2.1 as unique, it is not reliable to use as vessel identification. After removing NaN values from the dataset, only 1323 “unique” IMO values are left compared to 1555 unique MMSI values. Therefore, the 9 digit MMSI number will be used to identify vessels, although some literature (e.g. [4]) state that the same MMSI number may be incorrectly assigned to two or more vessels. Despite that, the MMSI numbers are assumed unique in this thesis.

The elapsed time between subsequent received AIS messages for each unique vessel is displayed in Figure 2.1 in order to show the data resolution. Subsequent messages with more than 15 minutes gap are not displayed as these messages are likely the result of vessels docking at a port and continuing at a later time. The time between messages is clearly varying but with a large peak around 6 minutes. AIS datasets with higher message update frequency exist, and it would have been preferable, but is not considered in this thesis.

The unsorted dataset in its original format is shown in Figure 2.2 while Figure 2.3 show 8% of the dataset's positions on top of a map of the Trondheimsfjord.

It appears from testing that a small amount of the AIS messages have an incorrect timestamp. This is discovered when subsequent AIS messages with respect to the

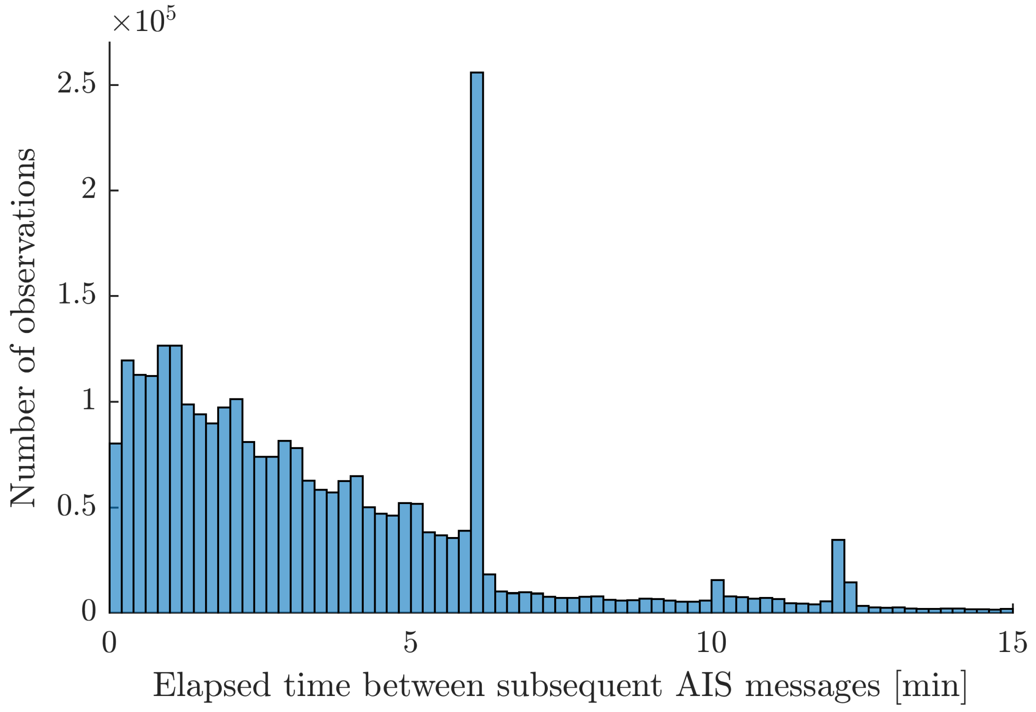
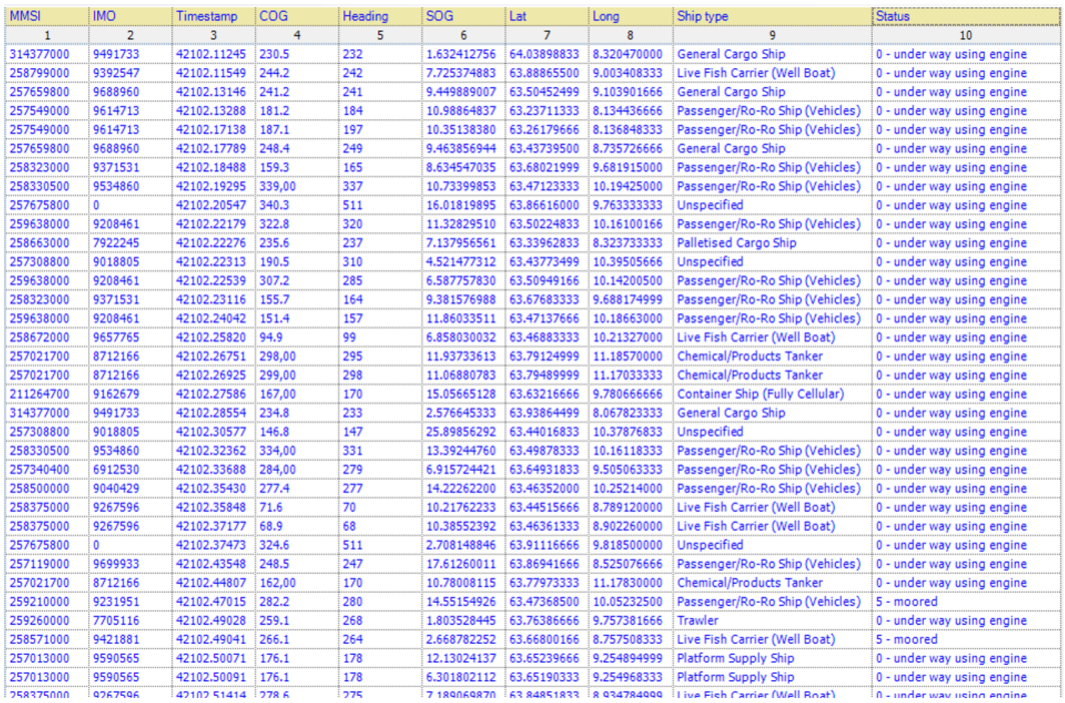


Figure 2.1: Elapsed time between subsequent AIS messages of same MMSI.

timestamp do not appear in a subsequent positional order although their courses are consistently indicating a movement in a given direction. Such incorrect AIS messages are removed from the dataset.



MMSI	IMO	Timestamp	COG	Heading	SOG	Lat	Long	Ship type	Status
1	2	3	4	5	6	7	8	9	10
314377000	9491733	42102.11245	230.5	232	1.632412756	64.03898833	8.320470000	General Cargo Ship	0 - under way using engine
258799000	9392547	42102.11549	244.2	242	7.725374883	63.88865500	9.003408333	Live Fish Carrier (Well Boat)	0 - under way using engine
257659800	9688960	42102.13146	241.2	241	9.449889007	63.50452499	9.103901666	General Cargo Ship	0 - under way using engine
257549000	9614713	42102.13288	181.2	184	10.98864837	63.23711333	8.134436666	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
257549000	9614713	42102.17138	187.1	197	10.35138380	63.26179666	8.136848333	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
257659800	9688960	42102.17789	248.4	249	9.463856944	63.43739500	8.735726666	General Cargo Ship	0 - under way using engine
258323000	9371531	42102.18488	159.3	165	8.634547035	63.68021999	9.681915000	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
258330500	9534860	42102.19295	339,00	337	10.73399853	63.47123333	10.19425000	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
257675800	0	42102.20547	340.3	511	16.01819895	63.86616000	9.763333333	Unspecified	0 - under way using engine
259638000	9208461	42102.22179	322.8	320	11.32829510	63.50224833	10.16100166	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
258663000	7922245	42102.22276	235.6	237	7.137956561	63.33962833	8.323733333	Palletised Cargo Ship	0 - under way using engine
257308800	9018805	42102.22313	190.5	310	4.521477312	63.43773499	10.39505666	Unspecified	0 - under way using engine
259638000	9208461	42102.22539	307.2	285	6.587757830	63.50949166	10.14200500	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
258323000	9371531	42102.23116	155.7	164	9.381576988	63.67683333	9.688174999	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
259638000	9208461	42102.24042	151.4	157	11.86033511	63.47137666	10.18663000	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
258672000	9657765	42102.25820	94.9	99	6.858030032	63.46883333	10.21327000	Live Fish Carrier (Well Boat)	0 - under way using engine
257021700	8712166	42102.26751	298,00	295	11.93733613	63.79124999	11.18570000	Chemical/Products Tanker	0 - under way using engine
257021700	8712166	42102.26925	299,00	298	11.06880783	63.79489999	11.17033333	Chemical/Products Tanker	0 - under way using engine
211264700	9162679	42102.27586	167,00	170	15.05665128	63.63216666	9.780666666	Container Ship (Fully Cellular)	0 - under way using engine
314377000	9491733	42102.28554	234.8	233	2.576645333	63.93864499	8.067823333	General Cargo Ship	0 - under way using engine
257308800	9018805	42102.30577	146.8	147	25.89856292	63.44016833	10.37876833	Unspecified	0 - under way using engine
258330500	9534860	42102.32362	334,00	331	13.39244760	63.49878333	10.16118333	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
257340400	6912530	42102.33688	284,00	279	6.915724421	63.64931833	9.505063333	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
258500000	9040429	42102.35430	277.4	277	14.22262200	63.46352000	10.25214000	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
258375000	9267596	42102.35848	71.6	70	10.21762233	63.44515666	8.789120000	Live Fish Carrier (Well Boat)	0 - under way using engine
258375000	9267596	42102.37177	68.9	68	10.38552392	63.46361333	8.902260000	Live Fish Carrier (Well Boat)	0 - under way using engine
257675800	0	42102.37473	324.6	511	2.708148846	63.91116666	9.818500000	Unspecified	0 - under way using engine
257119000	9699933	42102.43548	248.5	247	17.61260011	63.86941666	8.525076666	Passenger/Ro-Ro Ship (Vehicles)	0 - under way using engine
257021700	8712166	42102.44807	162,00	170	10.78008115	63.77973333	11.17830000	Chemical/Products Tanker	0 - under way using engine
259210000	9231951	42102.47015	282.2	280	14.55154926	63.47368500	10.05232500	Passenger/Ro-Ro Ship (Vehicles)	5 - moored
259260000	7705116	42102.49028	259.1	268	1.803528445	63.76386666	9.757381666	Trawler	0 - under way using engine
258571000	9421881	42102.49041	266.1	264	2.668782252	63.66800166	8.757508333	Live Fish Carrier (Well Boat)	5 - moored
257013000	9590565	42102.50071	176.1	178	12.13024137	63.65239666	9.254894999	Platform Supply Ship	0 - under way using engine
257013000	9590565	42102.50091	176.1	178	6.301802112	63.65190333	9.254968333	Platform Supply Ship	0 - under way using engine
258375000	9267596	42102.51414	278.6	275	7.189069870	63.84851833	8.934784999	Live Fish Carrier (Well Boat)	0 - under way using engine

Figure 2.2: The AIS dataset in its original format

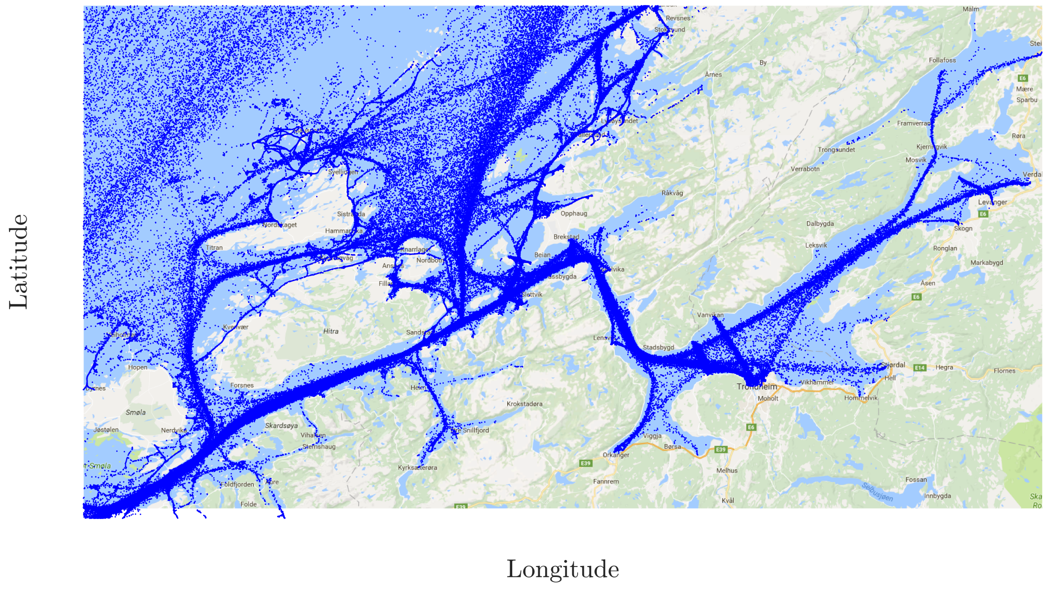


Figure 2.3: A total of 8% of the AIS data positions plotted on top of a map of the Trondheimsfjord. Note that the plotted AIS coordinates do not fully coincide with the map coordinates.

Chapter 3

A survey of vessel movement prediction techniques

This chapter summarizes a survey on maritime vessel prediction techniques given in [1]. See [1] for more detailed descriptions of the methods.

Little research have been done in terms of utilizing AIS data with the aim of vessel movement predictions for time horizons in the level of a few minutes. However, there exist AIS-based methods aiming for relatively long prediction horizons and some methods developed for animal and weather movement prediction which may facilitate shorter prediction horizons. This section summarizes a few of the most prevailing approaches which may be adapted to COLAV applications.

3.1 Typical prediction approach

Prediction methods for movement of objects are often based on path clustering and can typically be divided in four main steps in an AIS data context:

1. Cluster paths in the historical data to yield sets of similar route patterns.
2. Classify sequences of new, incoming AIS data points to assign vessels to the route patterns found in step 1.
3. Create representative paths of the common behavior in every route pattern.
4. Predict the vessels' future trajectories, for instance along their representative paths or by using a particle filter.

Various methods are used in all four steps across the literature. The path clustering in step 1 can be further divided into clustering of whole paths and sub-paths. Clustering of whole paths can be useful to predict destination and estimate time of arrival, while clustering of shorter sub-paths has better potential in the context of COLAV.

3.2 Clustering-based methods

The trajectory clustering (TRACLUS)[1] algorithm [5] is a sub-path clustering algorithm tested on hurricane data and animal movement data. The algorithm can be separated in two main parts; path partitioning and sub-path clustering. Paths, which originally consist of straight lines connecting subsequent positional points, are divided into even coarser line segments. The line segments are connected between so called *characteristic points* at which the directional change is to a certain extent. Hence, the length of line segments dynamically adapt with respect to the curvature of the path. The threshold of directional change before a characteristic point occur, is a trade-off between minimizing *preciseness* and *conciseness*. The former is the subsequent sub-paths' deviation from the original path and the latter is the number of partitions, i.e., the number of line segments between characteristic points. An optimal trade-off can be found e.g. by using the minimum description length (MDL) principle [6]. Similar line segments are clustered with an adapted version of the density-based spatial clustering of applications with noise (DBSCAN) algorithm [7]. A distance metric accounts for both the distance and the angular deviation between line segments, facilitating differentiation between perfectly aligned, but opposite moving objects, which is a necessary property in vessel movement prediction. Representative line segments are calculated after clusters are obtained.

Some studies (e.g. [8] and [9]) raise concern about TRACLUS' high sensitivity to its two clustering decision parameters which relates to the area density: A neighborhood radius and a lower limit for the number of line segments needed to form a cluster. In [10], the algorithm is improved with respect to this specific issue, yielding an algorithm that outputs much of the same clusters, but for a wider range of decision parameters which reduces the need of domain knowledge when applying the algorithm on a given dataset. Further, the method may experience problems in datasets with circular motion and frequently crossing paths. This is addressed in [11], which adjust the algorithm to better handle such behavior. Another drawback, in vessel prediction context, is that spatial connection between subsequent representative line segments cannot be guaranteed. In other words, a predicted path consisting of subsequent representative line segments is not necessarily continuous.

The traffic route extraction for anomaly detection (TREAD) ([12] and [13]) is a route pattern clustering method used in several papers dealing with AIS data. The area of interest is limited by a rectangular bounding box. TREAD clusters neither whole paths nor sub-paths, but certain *waypoints*: (1) *entry points*, (2) *exit points* and (3) *stationary points*. An entry point is marked when a vessel enters the bounding box and an exit point is marked when a vessel leaves the boundary box. The stationary points are detected when a vessel's speed is kept under a certain limit for a certain period of time and are included to account for areas where the vessels are docking or anchoring up. As waypoints are discovered from new incoming AIS data, they are clustered with the DBSCAN algorithm. The clusters are then used to classify routes inside the bounding box as paths are assigned to a start and an end cluster. A start cluster can either be an entry point or a stationary point while an end cluster can

[1]The use of *trajectory* can be misleading as temporal information is not considered in the clustering.

either be an exit point or a stationary point. All paths that belong to the same set of start and end clusters are grouped into a unique route pattern. Hence, the algorithm is able to separate between opposite moving vessels. After creating route patterns, representative paths are calculated and an entropy function, measuring the level of disorder in the pattern, serves as a quality measure.

As opposed to the sub-path clustering method in TRACLUS, TREAD's waypoint approach can theoretically allow large deviations in the distances between paths inside the same pattern since the only requirement to belong to a specific pattern is to belong to the same set of start and end clusters. A possible adaptation of the TREAD methodology to COLAV applications is to divide the area of interest into grid cells where each cell play the role as the bounding box. Hence, clustering of waypoints is performed in each cell and paths between these waypoints are then grouped into shorter patterns, better facilitating short time horizon predictions.

Mazzarella et al. propose a knowledge based velocity model (KB-VM) [14] and a knowledge based particle filter (KB-PF) [4] for vessel movement prediction based on AIS data. In both approaches, the TREAD method with a few adjustments is used to cluster AIS generated paths into route patterns. Then, based on position, course and speed, new paths are classified to one of the clustered route patterns using the k-nearest neighbor (kNN) algorithm [15] with the Mahalanobis distance metric. In order to fit the Mahalanobis distance metric to the specific dataset, the metric is *learned* with the supervised large margin nearest neighbor (LMNN) algorithm. If a path is assigned to a route pattern, the KB-PF method predicts future course and speed using a modified particle filter [4] while the KB-VM method predicts course and speed based on the current state's nearest neighbors in the classified pattern, as follows: The course is set to be the median course of all identified neighbors while the speed is set to be the last received speed from the AIS message. Both methods are compared, and the more complex and computational demanding particle filter yields more accurate predictions, but mainly for time horizons exceeding 4-5 hours.

Another approach based on particle filter is suggested in [16]. Implicitly, the particles yield a probability measure of future positions and are distributed over branching sea lanes, an important property for COLAV applications.

Chapter 4

Single Point Neighbor Search Method

The single point neighbor search (SPNS) method for AIS-based vessel trajectory predictions was first proposed in the 5th year specialization project [1] which lead to a paper accepted to the 20th International Conference on Information Fusion [17]. Instead of relying on path clustering methods, this approach estimates future course and speed at every prediction time based directly on historical AIS data.

The following updates of the algorithm have been made during this thesis:

1. Several confusing prediction error biases have been investigated and resolved.
2. Some minor notational changes have been made to improve readability.
3. An extra decision parameter *minCn* is added in Algorithm 1 to require a minimum number of CNs when calculating the CNs median course or speed as the median is not very suitable for extremely small sets of data. This is accomplished by expanding the search radius if the number of CNs is less than *minCn*.
4. The computational time of the CN search is reduced by utilizing a k-d tree for data storage.

4.1 Notation and definitions

Let us define $\boldsymbol{X}$ to be a matrix of historical AIS data according to

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_2 & ... & \boldsymbol{X}_n \end{bmatrix}^T \tag{4.1}$$

where n is the total number of AIS messages and

$$\boldsymbol{X}_i = \begin{bmatrix} \text{MMSI}_i & t_i & \boldsymbol{p}_i^T & \chi_i & v_i \end{bmatrix}, \tag{4.2}$$

for $i \in \{1, 2, ..., M\}$ is a vector where MMSI_i, t_i, $\boldsymbol{p}_i$, χ_i and v_i are a vessel's maritime mobile service identity (MMSI) number, timestamp, position vector, course over ground (COG) and speed over ground (SOG), respectively. The position vector can further be written as $\boldsymbol{p}_i = \begin{bmatrix} \lambda_i & \phi_i \end{bmatrix}^T$, where λ_i and ϕ_i is the WGS84 longitude and latitude, respectively. The matrix $\boldsymbol{X}$ is sorted such that AIS messages with equal

MMSI number are grouped together and messages within every such group are sorted with respect to ascending timestamp.

A predicted trajectory consists of K predicted positions at certain instants of time, including the initial true state. At every iteration $k \in \{1, ..., K-1\}$, the predicted state is separated into one *a priori* state, denoted $\hat{\boldsymbol{X}}_k^-$, and one *a posteriori* state, denoted $\hat{\boldsymbol{X}}_k^+$. These two states are written

$$\hat{\boldsymbol{X}}_k^- = \begin{bmatrix} \text{MMSI}_i & \hat{t}_k & \hat{\boldsymbol{p}}_k^T & \hat{\chi}_k^- & \hat{v}_k^- \end{bmatrix} \tag{4.3}$$

and

$$\hat{\boldsymbol{X}}_k^+ = \begin{bmatrix} \text{MMSI}_i & \hat{t}_k & \hat{\boldsymbol{p}}_k^T & \hat{\chi}_k^+ & \hat{v}_k^+ \end{bmatrix}. \tag{4.4}$$

Notice that the MMSI number, the position vector and the timestamp are the same for both the *a priori* and the *a posteriori* state. The only difference is the predicted course and predicted speed. The *a priori* predicted course and speed at iteration k, $\hat{\chi}_k^-$ and $\hat{v}_k^-$, represent the predicted course and speed between the previous position $\hat{\boldsymbol{p}}_{k-1}$ and the current position $\hat{\boldsymbol{p}}_k$. Similarly, the *a posteriori* predicted course and speed at iteration k, $\hat{\chi}_k^+$ and $\hat{v}_k^+$, represent the predicted course and speed between the current position $\hat{\boldsymbol{p}}^k$ and the next position $\hat{\boldsymbol{p}}^{k+1}$. Also notice that the subscript $\cdot_i$ is only present on the MMSI number because the MMSI number is the only value that exists in the dataset $\boldsymbol{X}$. The remaining values are predicted values and can not be found in the dataset.

All states $\boldsymbol{X}_i$ that are *close* to an *a priori* predicted state $\hat{\boldsymbol{X}}_k^-$ with respect to a distance metric, are defined as close neighbors (CNs) of that predicted state. The set of CNs at a given prediction step k is defined as

$$\boldsymbol{C}_k = \{\boldsymbol{X}_i \mid d(\hat{\boldsymbol{p}}_k, \boldsymbol{p}_i) \leq r_c, \chi_i \in S, \boldsymbol{X}_i \in \boldsymbol{X}\} \tag{4.5}$$

where

$$d(\hat{\boldsymbol{p}}_k, \boldsymbol{p}_i) = 2R\sin^{-1}\left(\left(\sin^2\left(\frac{\hat{\phi}_k - \phi_i}{2}\right) + \cos(\phi_i)\cos(\hat{\phi}_k)\sin^2\left(\frac{\hat{\lambda}_k - \lambda_i}{2}\right)\right)^{\frac{1}{2}}\right) \tag{4.6}$$

is defined as the distance between the current predicted position and any position in the AIS dataset given by the Haversine rule[1][18], R is earth's radius, r_c is the search radius and S is an interval of course angles defined by

$$S = \left[\hat{\chi}_k^- - \Delta\chi, \hat{\chi}_k^- + \Delta\chi\right], \tag{4.7}$$

where $\Delta\chi > 0$ is the maximum course angle deviation. Hence, the distance metric restricts all CNs to be within a given radius of the predicted position, as well as having a course within a maximum deviation from the *a priori* predicted course. After the set of CNs is found around $\hat{\boldsymbol{X}}_k^-$, the CNs' mean or median course and speed are used to predict the vessel's course and speed between the current and next position, i.e.,

[1]The Haversine formula assumes the earth to be spherical, which differs from the more precise WGS84 model in which the AIS coordinates are represented in. At the latitude of our AIS data, the spherical model introduces an innaccuracy of about 0.5% for distances up to a few hundred meters.

they are used to calculate the *a posteriori* state $\hat{\boldsymbol{X}}_k^+$. To simplify the notation in the following, we denote every state that belongs to the set of CNs at prediction step k, i.e., all $\boldsymbol{X}_i \in \boldsymbol{C}_k$, by $\boldsymbol{X}_k^c = \begin{bmatrix}\text{MMSI}_i^c & t_k^c & \boldsymbol{p}_k^c & \chi_k^c & v_k^c\end{bmatrix}$ for $c \in \{1, ..., C_k\}$, where C_k is the number of CNs at the given k.

Lastly, a predicted trajectory starting from a given state $\boldsymbol{X}_i$ is defined as:

$$\hat{\boldsymbol{T}}_i = \left\{\begin{bmatrix}\hat{\boldsymbol{p}}_1 & \hat{t}_1\end{bmatrix}, ..., \begin{bmatrix}\hat{\boldsymbol{p}}_K & \hat{t}_K\end{bmatrix}\right\}. \tag{4.8}$$

Similarly, a vessel's true trajectory, $\boldsymbol{T}_i$, starting from a given state $\boldsymbol{X}_i$, is defined as

$$\boldsymbol{T}_i = \left\{\begin{bmatrix}\boldsymbol{p}_i & t_i\end{bmatrix}, ..., \begin{bmatrix}\boldsymbol{p}_{i+L} & t_{i+L}\end{bmatrix}\right\}. \tag{4.9}$$

where L is the number of measured AIS states in the trajectory. Notice that K and L is not necessarily equal, as several prediction step can occur between two subsequent AIS messages. Also notice that the first element in $\hat{\boldsymbol{T}}_i$ and $\boldsymbol{T}_i$ are always equal, as the starting point for both trajectories are given by the same state, $\boldsymbol{X}_i$. Be aware that trajectories and paths are not equally defined. A trajectory consists of time-parameterized subsequent positional points while a path consists of subsequent positional points independent of the elapsed time since the starting position.

The key notation introduced is summarized in Table 4.1.

Symbol	**Explanation**
$\boldsymbol{X}_i$	AIS message / state vector
$\boldsymbol{X}$	Set of all AIS messages $\boldsymbol{X}_i$
$\boldsymbol{C}_k$	Set of CNs at prediction step k
$\boldsymbol{X}_k^c$	States belonging to the set of CNs at prediction step k
$\hat{\boldsymbol{X}}_k^-$	*A priori* predicted state at prediction step k
$\hat{\boldsymbol{X}}_k^+$	*A posteriori* predicted state at prediction step k
$\Delta\chi$	Maximum course angle deviation for CNs
r_c	Search radius [m]
Δl	Step length [m]
K	Number of predicted states, including the initial state
C_k	Number of CNs at iteration k

Table 4.1: List of symbols for SPNS

4.2 Method

The prediction method is described step by step in Algorithm 1. The steps are described in more detail in the continuation of this section.

Algorithm 1 Single Point Neighbor Search prediction

```
 1: $\boldsymbol{X}_i$ given                    ▷ The known state we want to predict from
 2: **Set decision parameters**
        (a) $\Delta l$                          ▷ Step length [m]
        (b) $r_c$                               ▷ Search radius [m]
        (c) $\Delta\chi$                        ▷ Maximum course angle deviation [deg]
        (d) $minCn$                             ▷ Minimum number of CNs
        (e) $K$                                 ▷ Number of states to predict, including the initial state
 3: Set $\hat{\boldsymbol{X}}_1^- = \boldsymbol{X}_i$
 4: **for** $k = 1$ to $K - 1$ **do**
 5:     Find all CNs $\boldsymbol{X}_k^c$ around $\hat{\boldsymbol{X}}_k^-$
 6:     **if** Number of CNs $< minCn$ **then**
 7:         Expand search radius until $minCn$ is reached or until search radius $\geq 4r_c$
 8:     **end if**
 9:     Calculate $\hat{\boldsymbol{X}}_k^+$ by:
            (a) Calculating $\hat{\chi}_k^+$ based on $\boldsymbol{X}_k^c$
            (b) Calculating $\hat{v}_k^+$ based on $\boldsymbol{X}_k^c$
10:     Calculate the next predicted position at its predicted point in time by:
            (a) Calculating $\hat{\boldsymbol{p}}_{k+1}$ according to (4.10)
            (b) Set $\hat{t}_{k+1} = \hat{t}_k + \frac{\Delta l}{\hat{v}_k^+}$
11:     Set $\hat{\boldsymbol{X}}_{k+1}^- = \begin{bmatrix} \text{MMSI}_i & \hat{t}_{k+1} & \hat{\boldsymbol{p}}_{k+1} & \hat{\chi}_k^+ & \hat{v}_k^+ \end{bmatrix}$
12: **end for**
```

4.2.1 Decision parameters

Step length

The step length Δl decides how far the next position should be propagated, according to

$$\hat{\boldsymbol{p}}_{k+1} = \hat{\boldsymbol{p}}_k + \Delta l \left[\sin(\hat{\chi}_k^+) f(\hat{\phi}_k), \cos(\hat{\chi}_k^+) g(\hat{\phi}_k)\right], \tag{4.10}$$

where $f(\hat{\phi}_k)$ and $g(\hat{\phi}_k)$ are functions of the current latitude $\hat{\phi}_k$, which transform from meters to degrees longitude and degrees latitude, respectively. The step length Δl should reflect the curvature of the sea lanes ahead. Short step lengths facilitate tighter turns and smoother trajectories, but increases the computational requirements. If the step length is set too long, some turns may not be possible to follow and a predicted state might fall into a region with no CNs. Ideally, the algorithm would adapt the step length to the curvature ahead of the vessel, but this is not implemented in this thesis.

Search radius

The search radius r_c restricts CNs to be within a maximum Euclidean distance from the current predicted position. In order for subsequent search areas not to overlap, meaning that the same neighbor is never evaluated more than once, the relationship between the search radius and the step length may be set according to $r_c = \frac{\Delta l}{2}$.

Maximum course deviation

The maximum course deviation parameter $\Delta\chi$ defines the distance metric together with the search radius r_c, as defined in (4.5). This parameter features two main objectives:

1. Excluding neighbor vessels within the search radius which move in the opposite direction of the vessel from which trajectory we want to predict.

2. Avoiding neighbor vessels within the search radius which come from *crossing* sea lanes[2].

Figure 4.1 illustrates the concept of the distance metric with $r_c = 400$ m and $\Delta\chi = 35°$. Figure 4.1a shows two sets of AIS states: the ones CNs that satisfy both the r_c and the $\Delta\chi$ requirement are plotted in green and the ones that only are within the search radius are plotted in red. The course distributions of the two sets of states are shown in Figure 4.1b and Figure 4.1c. The two distributions show an average course separation of 180.7°, meaning that the two groups' average direction of travel is opposite of eachother.

[2]A *crossing* sea lane is, as the name suggest, a sea lane that crosses the sea lane of interest, coming from another area. This differ from a *branching* sea lane which is a sea lane that branch out from the sea lane of interest, coming from the same area.

Minimum number of CNs

The minimum number of CNs *minCn* is included to better avoid predictions based on a very sparse sets of CNs. This can typically be an issue when the step length, and hence also the search radius, is set short in order to capture steep turns. A maximum expansion of $4r_c$ is chosen and if *minCn* is not reached with this search radius the algorithm continues with the current amount of CNs it has, if any.

Number of prediction steps

The number of predicted states K must be calculated from the desired prediction horizon t_h, which is typically up to 15 minutes for COLAV applications. That is, K must be chosen to satisfy

$$t_h \leq \sum_{k=1}^{K-1} \frac{\Delta l}{\hat{v}_k^+} \tag{4.11}$$

K can be calculated prior to prediction if all the predicted speeds are assumed constant. If not, the prediction algorithm must run until (4.11) is satisfied.

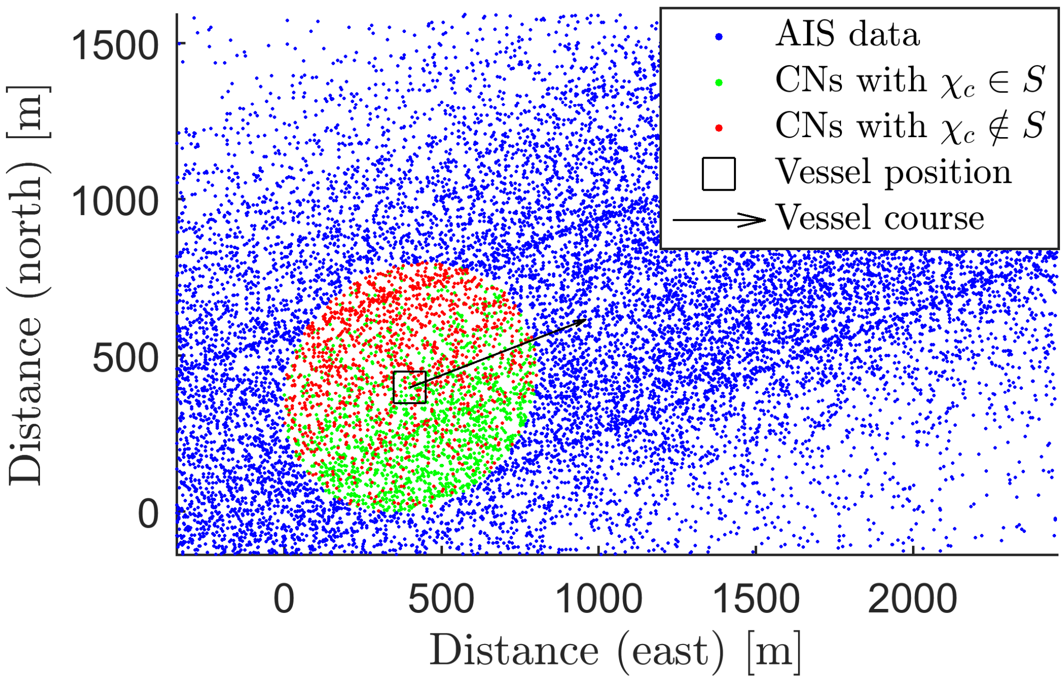
4.2.2 Course prediction

A constant velocity model is used whenever $\boldsymbol{C}_k = \emptyset$. When $\boldsymbol{C}_k \neq \emptyset$, the *a posteriori* course prediction at a given iteration, $\hat{\chi}_k^+$, is based on the behavior of the CNs where the CNs are defined in (4.5). Two course predictions were tested and compared in [1]: $\hat{\chi}_k^+ = \bar{\chi}_k^c$ and $\hat{\chi}_k^+ = \tilde{\chi}_k^c$ where the former is the mean course of all CNs and the latter is the median course of all CNs. The mean course was shown to be more influenced by course outliers than the median course. The median course yielded lower 10th and 50th percentiles[3] of absolute prediction errors in a single timestep test evaluated on 2000 randomly chosen initital states [1]. By this reason, the median course of CNs is chosen as the course prediction in this thesis. Since the course is periodic in $[0°, 360°]$, caution must be taken when calculating the median $\tilde{\chi}_k^c$.

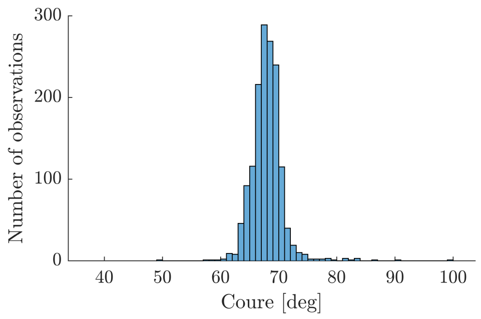
4.2.3 Speed prediction

Instead of assuming constant speed throughout the whole prediction, the mean or median speed of the CNs at iteration k, denoted $\bar{v}_k^c$ and $\tilde{v}_k^c$ respectively, may be used. It may also be a good idea to implement a transition from using constant speed (and perhaps course) at $k = 1$ towards predictions fully based on the CNs as k increases and the validity of the last received AIS message decreases.

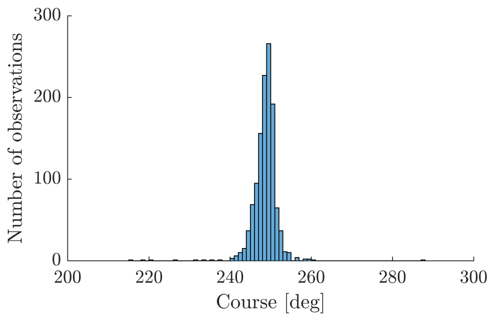
[3] A percentile is a measure indicating the value which a given percentage of observations in a group of observations fall below [19].



(a)



(b) Course distribution of the CNs with $\chi_c \in S$



(c) Course distribution of the CNs with $\chi_c \notin S$

Figure 4.1: Illustration of the distance metric for CNs with $\Delta\chi = 35°$ and $r_c = 400$ m.

Chapter 5

Neighbor Course Distribution Method

It will be illustrated in Chapter 6 that due to the SPNS method's single predicted trajectory, it does neither handle branching of sea lanes nor does it facilitates a probability distribution estimate of future route choices or any statistical quantification of uncertainty. However, any information about where surrounding vessels are likely to be in the nearest future is of interest in a COLAV context in order to proactively avoid collision situations. As an example, assume that an ASV has three different route options to its destination where all routes are more or less equally optimal with respect to a given set of optimality conditions. Further assume that the SPNS algorithm predicts a nearby vessel to be close to collision with the ASV if the ASV continues on route number 1. As a result, the ASV's COLAV algorithm will suggest to travel on either route 2 or route 3, but none of them are preferred above the other since they apparently are equally optimal. Based on the historical AIS data it might be that 50% of all historical trajectories, which started in a similar state as the current state of the nearby vessel, traveled route 1 while the remaining 50% traveled route 2. From this new information, route 3 seems now the safest route choice for the ASV. Estimates of the probability of route choices would clearly be valuable in such situations.

This section present an algorithm which is able to handle branching and which better facilitates statistical quantification of uncertainty and probability estimates of future route choices. The algorithm is based on many of the same principles as the SPNS method. At each and every prediction step, the method randomly draws a number of courses from the current state's CNs and then uses these courses to predict new positions. No weighting are used between predictions or between CN states in order to limit the number of user specified decision parameters and to keep it as simple as possible. The method is called the neighbor course distribution method (NCDM) and an illustration of the method is shown in Figure 5.1.

5.1 Notation and definitions

The NCDM follows the same notation and definitions as in Section 4.1 except that all subscripts $\cdot_{(k)}$ now are replaced with the subscripts $\cdot_{(k,j)}$, where k still refers to the iteration number (which also is the level of the prediction tree) and j refers to the

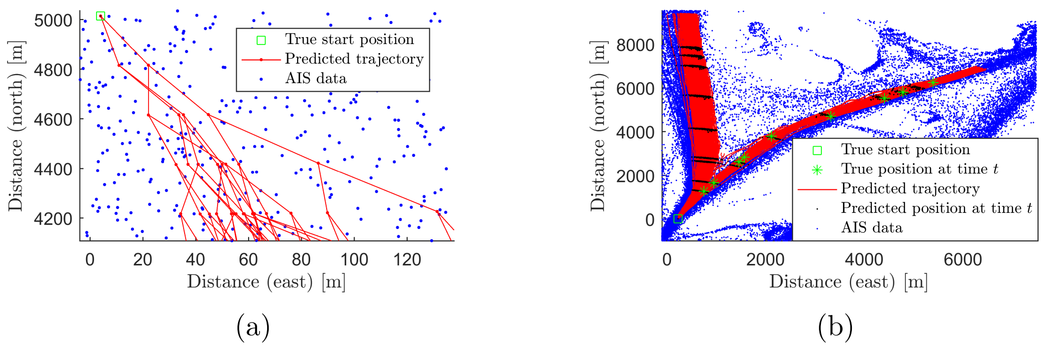
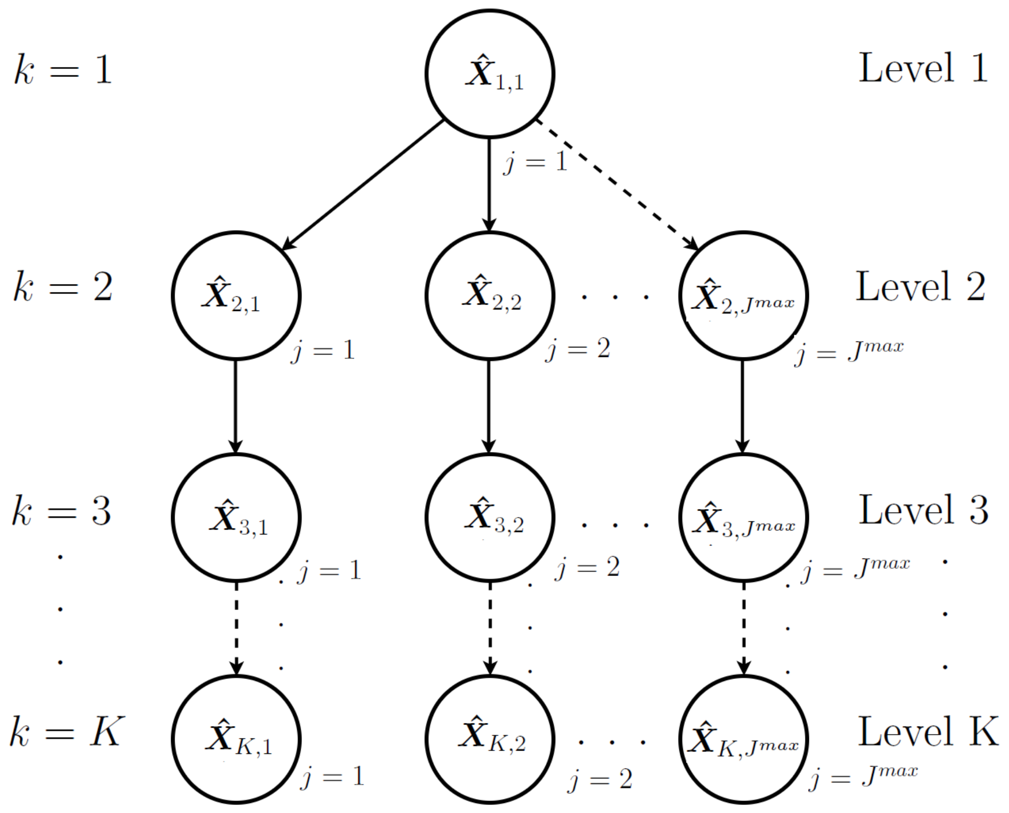


Figure 5.1: An illustrating example of the prediction expansion of the NCDM algorithm is shown in Figure 5.1a and an example of a full prediction is shown in Figure 5.1b. In Figure 5.1a, the algorithm draws 2 courses randomly from each predicted state's CNs at each prediction step. The prediction step length Δl is constant, but it might appear to vary due to the range differences between the x-axis and the y-axis.

horizontal width index of the tree at a given level k (see Figure 5.2 for an illustration). For instance, the *a priori* state is now denoted $\hat{\boldsymbol{X}}^-_{k,j}$, the *a posteriori* state is denoted $\hat{\boldsymbol{X}}^+_{k,j}$ and the set of CNs is written $\boldsymbol{C}_{k,j}$. A list of notation very similar to Table 4.1 is summarized in Table 5.1. A couple of the listed symbols will be defined in the next section.

Symbol	**Explanation**
$\boldsymbol{X}_i$	AIS message / state vector
$\boldsymbol{X}$	Set of all AIS messages $\boldsymbol{X}_i$
$\boldsymbol{C}_{k,j}$	Set of all CNs at tree index (k,j)
$C_{k,j}$	Number of CNs at tree index (k,j)
$\boldsymbol{X}^c_{k,j}$	States belonging to the set of CNs at tree index (k,j). $c \in \{1,...,C_{k,j}\}$.
$\hat{\boldsymbol{X}}^-_{k,j}$	*A priori* predicted state at tree index (k,j)
$\hat{\boldsymbol{X}}^+_{k,j}$	*A posteriori* predicted state at tree index (k,j)
$\Delta\chi$	Maximum course angle deviation for CNs
r_c	Search radius [m]
Δl	Step length [m]
K	Total number of tree levels
$N_{k,j}$	Number of child nodes at tree index (k,j)
J_k	Number of nodes at tree level k

Table 5.1: List of symbols for NCDM



$k = 1$
$\hat{\boldsymbol{X}}_{1,1}$
Level 1
$j = 1$
$k = 2$
$\hat{\boldsymbol{X}}_{2,1}$
$\hat{\boldsymbol{X}}_{2,2}$
$\hat{\boldsymbol{X}}_{2,J^{max}}$
Level 2
$j = 1$
$j = 2$
$j = J^{max}$
$k = 3$
$\hat{\boldsymbol{X}}_{3,1}$
$\hat{\boldsymbol{X}}_{3,2}$
$\hat{\boldsymbol{X}}_{3,J^{max}}$
Level 3
$j = 1$
$j = 2$
$j = J^{max}$
$k = K$
$\hat{\boldsymbol{X}}_{K,1}$
$\hat{\boldsymbol{X}}_{K,2}$
$\hat{\boldsymbol{X}}_{K,J^{max}}$
Level K
$j = 1$
$j = 2$
$j = J^{max}$

Figure 5.2: Illustration of the prediction tree structure. The root node (top) is at level 1 ($k = 1$) and it has $N_{k,j} = N_{1,1} = J^{max}$ child nodes (branches). From level $k = 2, ..., K - 1$, the number of child nodes are $N_{k,j} = 1$ unless the number of CNs is zero, then the current path terminates. The number of nodes at a given level k is $J_k = J_1 = 1$ at level 1 and $J_k = J^{max}$ for level $k \geq 2$ unless some of the paths are terminated. Note that each node in practice consists of both the *a priori* and the *a posteriori* state ($\hat{\boldsymbol{X}}^-_{k,j}$ and $\hat{\boldsymbol{X}}^+_{k,j}$) which is only represented with $\hat{\boldsymbol{X}}_{k,j}$ in the figure.

5.2 Method

The set of predictions for a single vessel forms a tree structure where the starting state $\hat{\boldsymbol{X}}_{1,1} = \boldsymbol{X}_i$ is the root node and each node at every level k of the tree with index (k, j) has $N_{k,j}$ number of child nodes. A node corresponds to a predicted state and the number of child nodes to a given node is the number of predictions branching out from that node. The tree level equals the number of subsequent predictions[1] meaning that with a constant step length Δl, the accumulated predicted distance at tree level k is $(k - 1)\Delta l$. The number of nodes at a given level k is denoted J_k. The structure and indexing of the prediction tree is illustrated in Figure 5.2 and the algorithm is presented in Algorithm 2.

[1]The initial state is counted as first prediction, although known exact.

[2](4.10) is used to calculate the next position, but the use of subscripts in the eqution does not match the use of subscripts in this chapter.

Algorithm 2 Neighbor Course Distribution Method

```
 1: $\boldsymbol{X}_i$ given                    ▷ The known state we want to predict from
 2: Set decision parameters
        (a) $\Delta l$                          ▷ Step length [m]
        (b) $r_c$                               ▷ Search radius [m]
        (c) $\Delta\chi$                        ▷ Maximum course angle deviation [deg]
        (d) $N_{k,j}$                           ▷ Number of child nodes to be predicted from a given state
        (e) $K$                                 ▷ Total number of tree levels
 3: Set $\hat{\boldsymbol{X}}^{-}_{1,1} = \boldsymbol{X}_i$
 4: for $k = 1$ to $K - 1$ do                   ▷ Runs through the levels of the tree
 5:     $q = 0$                                 ▷ Indexing variable at level $k$
 6:     for $j = 1$ to $J_k$ do                 ▷ Runs through all nodes at given level $k$
 7:         Find all CNs $\boldsymbol{X}^c_{k,j}$, where $c \in \{1, ..., C_{k,j}\}$, around $\hat{\boldsymbol{X}}^{-}_{k,j}$
 8:         for $N_{k,j}$ iterations do
 9:             Calculate $\hat{\boldsymbol{X}}^{+}_{k,j}$:
                    (a) Set $\hat{\chi}^{+}_{k,j}$ to a random course from the CNs courses $\chi^c_{k,j}$
                    (b) Set $\hat{v}^{+}_{k,j} = v_i$
10:             Calculate the child node's next predicted position and time:
                    (a) $q = q + 1$
                    (b) Calculate $\hat{\boldsymbol{p}}_{k+1,q}$ according to (4.10)$^2$
                    (c) Set $\hat{t}_{k+1,q} = \hat{t}_{k,j} + \frac{\Delta l}{\hat{v}^{+}_{k,j}}$
11:             Set $\hat{\boldsymbol{X}}^{-}_{k+1,q} = \begin{bmatrix} \text{MMSI}_i & \hat{t}_{k+1,q} & \hat{\boldsymbol{p}}_{k+1,q} & \hat{\chi}^{+}_{k,j} & \hat{v}^{+}_{k,j} \end{bmatrix}$
12:         end for
13:     end for
14: end for
```

5.2.1 Decision parameters

The same decision parameters as in Algorithm 1 have to be chosen except for the minimum number of CNs ($minCn$) which is omitted[3]. Additionally, we have to decide the number of child nodes $N_{k,j}$ at each predicted state, or in other words how the prediction tree shall grow. The prediction tree, and hence the computational time, grows exponentially as long as $N_{k,j} > 1$. Therefore, the size of the tree must be limited for practical purposes. Depending on how we choose the tree to grow, the resulting predictions will have slightly different properties. For instance, if we set $N_{k,j} = N_{1,1} >> 1$, i.e., we set the initial number of child nodes high, chances are better that the algorithm will seed predicted trajectories in both sea lanes. In the opposite scenario, where $N_{1,1}$ is very small and the sea lane is branching early, the algorithm may not seed enough predicted trajectories in both sea lanes, causing one sea lane to be partially or fully excluded.

As mentioned, there are several possible ways to grow the prediction tree. In this thesis we will aim for a constant tree width J^{max} for three main reasons:

(a) To have the same number of predictions to base the statistical evaluation on throughout the prediction horizon.

(b) Increase the chance to seed predicted trajectories in multiple sea lanes when branching occur close to the initial state.

(c) To get a more predictable computational time as opposed to any non-constant tree structure.

When aiming for a constant tree width throughout the prediction horizon, the tree structure will grow in the following way: the maximum width of the tree is set to J^{max} and the number of child nodes from the initial state is set to $N_{1,1} = J^{max}$. From level $k = 2$ and beyond, the number of child nodes from a given node is $N_{k,j} = min(1, C_{k,j})$. This means that the tree expands to maximum width at the first prediction step and aims to keep this width constant throughout the prediction horizon. In the scenarios where a node does not have any CNs, i.e., $C_{k,j} = 0$, its predicted path is terminated[4] and the tree width is hence reduced by one. It may often occur that $C_{1,1} < N_{1,1} = J^{max}$ or in other words that the number of CNs in the initial state is less than the number of wanted child nodes. In order to make sure that the initial spread of courses are as wide as possible in these situations, the CNs' courses are all selected equally many times until there are left less courses to be picked than there are number of CNs. At this point, the remaining number of child nodes are randomly selected, without replacement, from the CNs' course distribution. More precisely, the initial state's child nodes are calculated from its CNs courses according to:

1. All its CNs courses are chosen ($N_{1,1}$ div $C_{1,1}$) number of times where div denotes integer division.

[3]The $minCn$ was introduced in the SPNS algorithm to avoid calculating the median course and speed from a very sparse set of data. However, we do not rely on calculations of medians in the NCDM algorithm and therefore $minCn$ is of less importance.

[4]An alternative could be to continue the prediction with constant course and speed. Termination is chosen because the NCDM algorithm in principle is data driven and since we can afford to terminate some predicted trajectories and still represent several future route choices, in most cases.

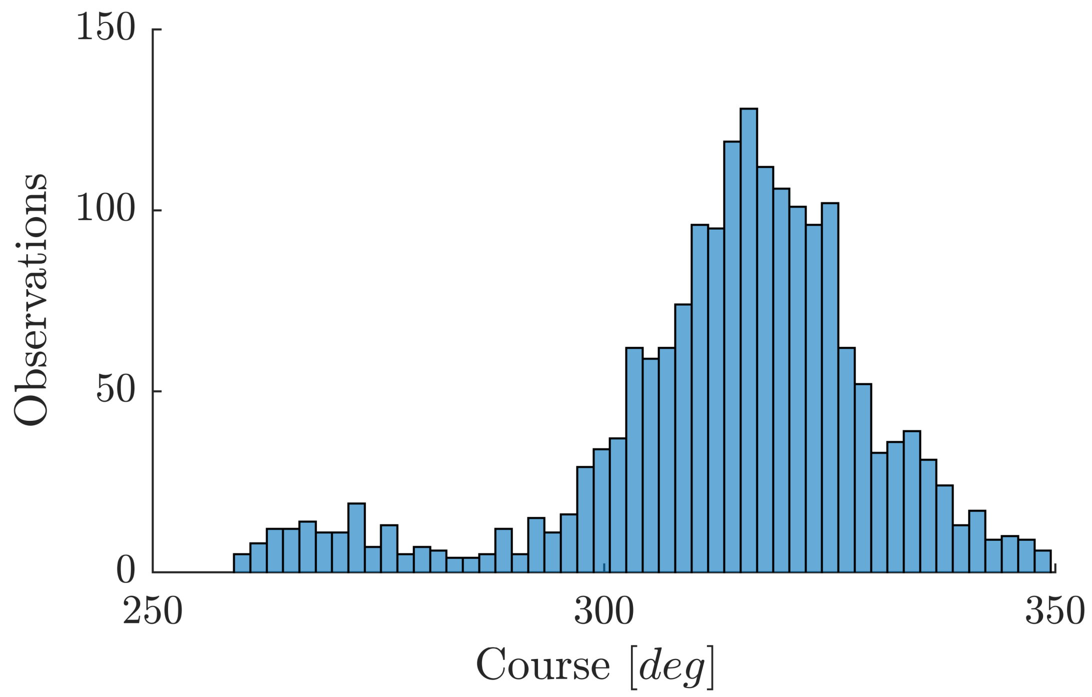


Figure 5.3: An example of a CNs course distribution. There is one major direction of travel, centered around 315°, and a smaller group of courses centered around 270°.

2. The remaining ($N_{1,1}$ mod $C_{1,1}$) number of child nodes are calculated based on the courses randomly drawn, without replacement, from the CNs.

After the first prediction step, i.e., for $k \geq 2$, each single child node is calculated from a single course randomly drawn from the CNs course distribution.

An example of a CNs course distribution is shown in Figure 5.3. The more courses that are randomly drawn from the distribution, the better the predicted courses will represent the distribution. It might seem that letting $N_{k,j} = 1$ for $k \geq 2$ is not enough to represent the course distribution in a given state. However, there are typically many predicted states in the same area with similar or equal distribution of CN courses. If for instance there are 10 predicted states at the same position with identical CN course distributions and we are randomly drawing 1 CN course from each of these equal states, it is equivalent to having a single state with $N_{k,j} = 10$.

5.2.2 Course and speed prediction

The course and speed predictions are described in step 9(a) and 9(b) in Algorithm 2. The course predictions at a given predicted state are simply $N_{k,j}$ random courses pulled from the CNs course distribution. The speed predictions are all set constantly equal to the initial, last known true speed v_i. Alternatively, the predicted speed can be set to the median or mean of all the CNs speed values, as suggested in Algorithm 1, or a linear combination of the constant speed and a CN based speed. A fourth option is to assign to the predicted speed a randomly drawn speed from the CNs speed distribution, in the same manner as with the course prediction.

5.3 Suggestion for probability measure

The NCDM algorithm is intended to better support a measure of uncertainty as well as to better reflect the probability of future route choices for a vessel in a given state. Using Figure 5.1b as an illustrative example, the relative number of predicted trajectories in each sea lane is intended to represent the probability of sea lane choice for the initial state marked in the figure. Further, the spread of the predicted positions at given time instants (black dots) intend to represent uncertainty of the prediction. Further, the relative number of predicted positions in two or more clusters at a given prediction time is intended to represent the probability distribution of the vessel's position at that given time. Whether or not the predictions actually reflect these suggested probability measures is yet to be investigated and is not tested in this thesis.

Chapter 6

Tests and results

6.1 SPNS' potential and shortcomings

The SPNS algorithm is first run on 10 manually chosen scenarios to highlight the algorithm's potential and to reveal shortcomings. The decision parameters[1] are given in Table 6.1 and the predictions are illustrated in Figure 6.1[2]. To recap some notation, the *a posteriori* course and speed predictions $\hat{\chi}_k^{k+}$ and $\hat{v}_k^{k+}$ are equally set to the median of all the CNs' course and speed values, denoted $\tilde{\chi}_k^c$ and $\tilde{v}_k^c$, respectively.

Decision parameter	**Value**	**Explanation**
r_c	50 m	Search radius for CNs
Δl	$2r_c$	Prediction step length
$\Delta\chi$	25°	Maximum course deviation for CNs
$\hat{\chi}_k^{k+}$	$\tilde{\chi}_k^c$	Course prediction method used at every iteration k
$\hat{v}_k^{k+}$	$\tilde{v}_k^c$	Speed prediction method used at every iteration k

Table 6.1: Decision parameters for the SPNS algorithm

Figure 6.1a - Figure 6.1e yield relatively accurate trajectory predictions (in both course and speed). Figure 6.1f shows accurate speed prediction, but the predicted path deviates from the true path inside the wide sea lane, illustrating that although the algorithm can potentially follow sea lanes, it can not guarantee to take the correct path within it.

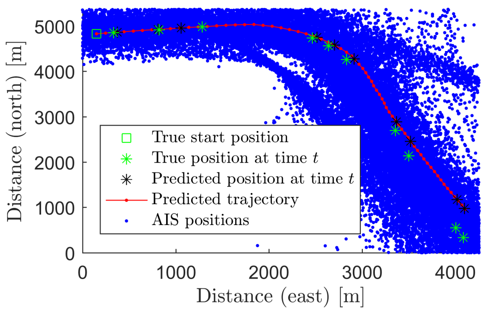
Figure 6.1g illustrates a situation where the turn was too tight for the given decision parameters. Therefore, by reducing the step length from 50 m to 25 m, the algorithm is able to follow the turn, as shown in Figure 6.1h. The relationship between $\Delta\chi$ and Δl is important. For a given step length, a too low value of $\Delta\chi$ will limit the radius of curvature that the algorithm can follow. On the other hand, a too large value of

[1]Due to the tight turns in Figure 6.1c, the plot is generated with half the step length (and hence half search radius) and a larger accepted maximum course deviation for neighbors, meaning $r_c = \frac{\Delta l}{2} = 25\text{m}$ and $\Delta\chi = 35°$.

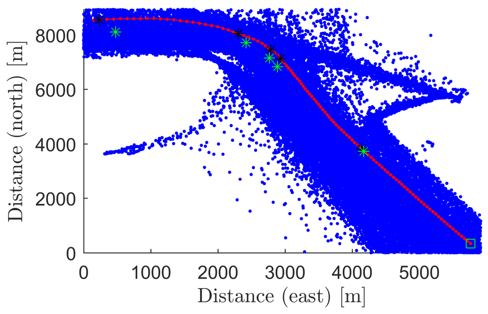
[2]Be aware that the ratio between the x-axis and the y-axis are not 1:1 and that it vary among the subfigures.

$\Delta\chi$ may for instance cause a prediction to suddenly be pulled into a crossing sea lane. However, by reducing the step length, the maximum course deviation can be reduced without compromising the range of radii of curvatures to follow, at the same time as crossing sea lanes are less likely to be an issue.

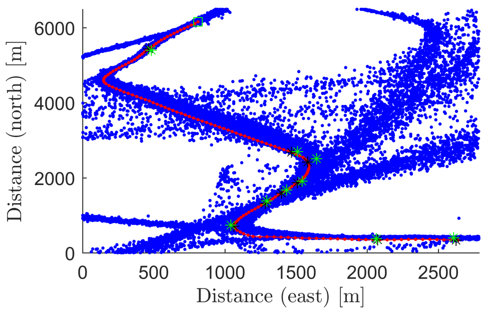
Figure 6.1i and Figure 6.1j demonstrate that the algorithm does not handle branching of sea lanes. It typically follows the most dense path ahead.



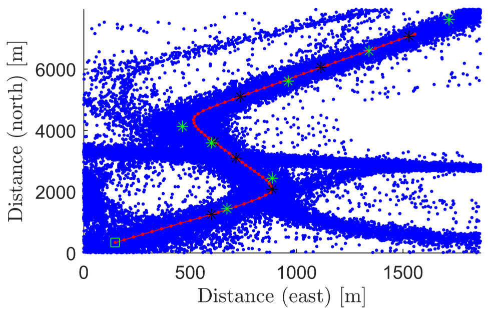
(a) Time horizon: 18.87 min



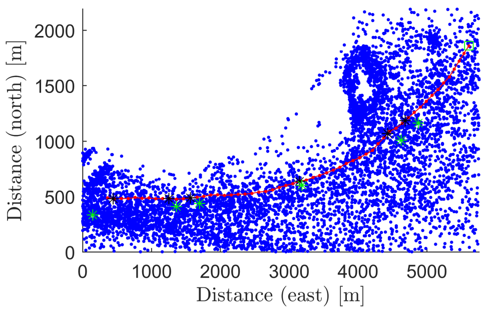
(b) Time horizon: 30.38 min



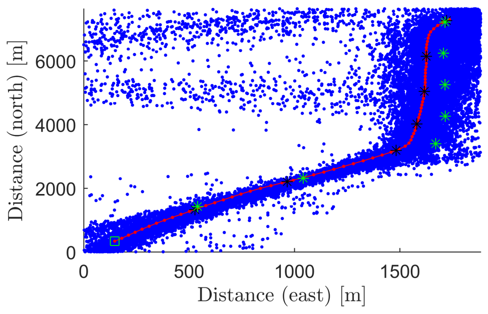
(c) Time horizon: 26.35 min



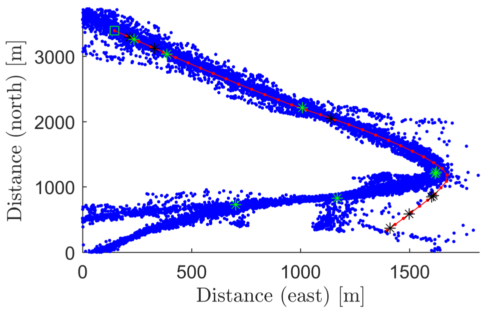
(d) Time horizon: 21.82 min



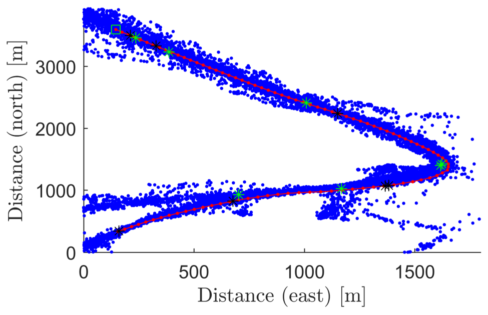
(e) Time horizon: 23.22 min



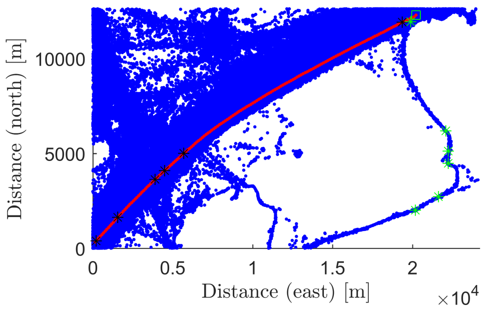
(f) Time horizon: 21.32 min



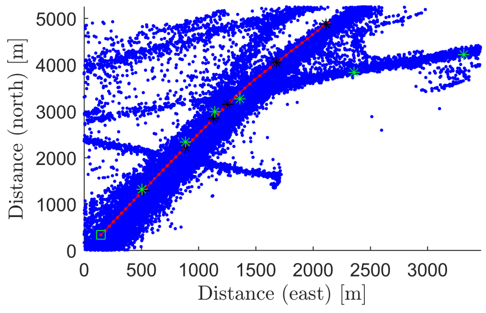
(g) Time horizon: 18.33 min



(h) Time horizon: 18.33 min



(i) Time horizon: 49.17 min



(j) Time horizon: 15.32 min

Figure 6.1: The SPNS algorithm tested on a set of manually chosen curved trajectories. Explanation of the sub figures' markers are shown in Figure 6.1a.

6.2 SPNS on curved trajectories

The scenarios in Figure 6.1 are both few and manually chosen. Hence, they are not representable for the algorithm's overall performance. Therefore, the algorithm is tested in more depth.

6.2.1 Test setup

The SPNS algorithm is tested and compared with the constant velocity model on the first 350 randomly drawn curved trajectories that fulfill the following requirements:

1. The elapsed time from the beginning to the end of the trajectory[3] must be between 15 and 25 minutes.

2. To get a minimum of true evaluation points, the trajectory must consist of at least 4 AIS messages including the starting point.

3. To test trajectories with a minimum of curvature is the accumulated course change between every subsequent pair of AIS messages in the trajectory required to be at least 40°, i.e., $\sum_{i=2}^{L} |\chi_i - \chi_{i-1}| \geq 40°$ where L is the number of AIS messages in the trajectory.

4. In order not to start in an empty area (where the SPNS approach makes little sense), a minimum of 5 CNs around the starting state is required.

5. To restrict the test to moving vessels the starting speed must be minimum 3 knots.

The 350 trajectories are predicted with the SPNS algorithm. The decision parameters in Table 6.1 are used, but two variants of the speed prediction $\hat{v}_k^+$ are tested: the first variant uses the median speed of the CNs ($\tilde{v}_k^c$) and the second uses the last received speed from the AIS message (v_i) throughout the whole prediction. These two variations are compared to a constant velocity model (CVM) approach, i.e. a prediction model that keeps the course and speed constantly equal the initial known course

[3]Note that the tested trajectories can be sub-trajectories of longer trajectories.

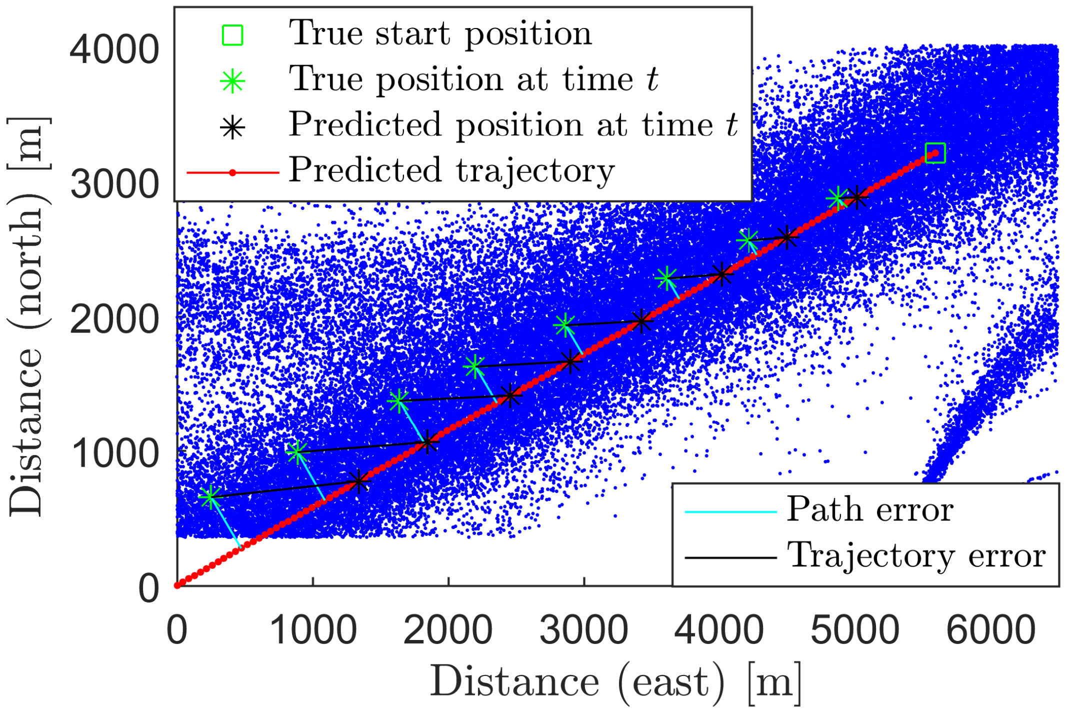


Figure 6.2: Illustration of path errors and trajectory errors.

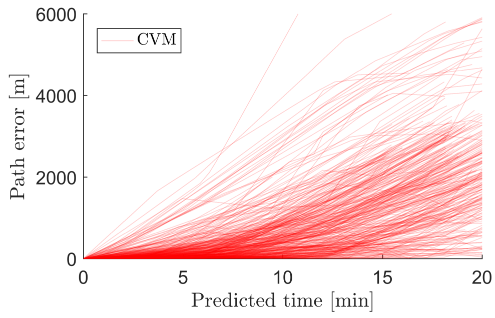
and speed throughout the whole prediction.

The upcoming sections will test trajectory errors and path errors. The path errors for a given true path is calculated by subsequently finding the shortest distance from the current true position to any position on the predicted path, as long as the predicted path's position is ahead (in the direction of travel) of the previous closest position. The trajectory error at a given time t is the distance from the true position at time t to the predicted position at time t. The difference between path and trajectory error is illustrated in Figure 6.2.

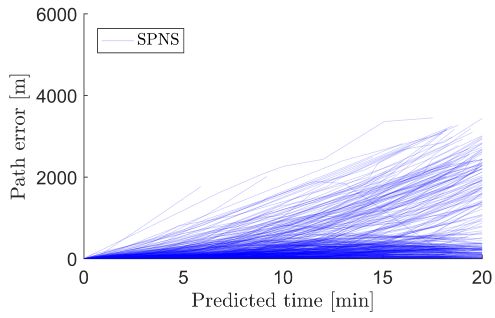
In the plots of the average and median error values in the next subsections, the values are calculated from the 350 trajectories at every minute starting from time zero, i.e., it is calculated at minute 0,1,...,15. Since the AIS messages are received at various instants of time the exact errors at the chosen time instants are typically not available. Therefore, the average and median values at each minute are calculated from the trajectories' linearly interpolated error values. Note that the number of trajectories used in the calculations decreases from 350 to 0 between 15 and 25 minutes. The x-axis in the average and median plots are therefore limited to 15 minutes in order to have an equal amount of error values in the calculation at each evaluation point.

6.2.2 Path errors

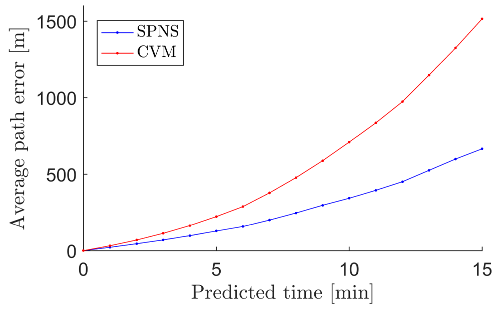
Graphical results of the path error tests are shown in Figure 6.3. Figure 6.3a and Figure 6.3b show the path errors for all the test trajectories as a function of predicted time, i.e., how well the CVM and SPNS methods perform based on predicted course



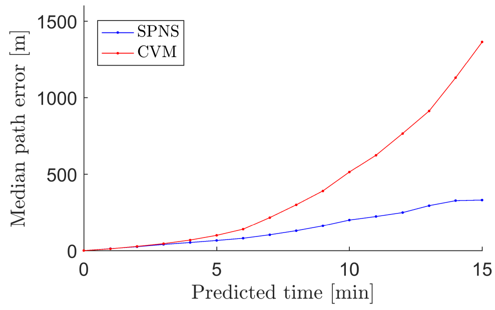
(a) Path error CVM



(b) Path error SPNS



(c) Average path errors



(d) Median path errors

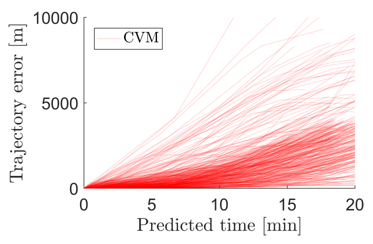
Figure 6.3: Path error comparison between SPNS and CVM for curved trajectories.

along the path but regardless of the predicted speed[4]. The test trajectories' average and median path errors as a function of predicted time are shown in Figure 6.3c and Figure 6.3d. As expected, the CVM path error is heavily increasing as the predicted time goes by. The SPNS predictions yield significantly better results than the CVM method. At prediction time 15 [min], the SPNS' average and median path errors are 665 m and 331 m, respectively, while the CVM's average and median path errors are 1545 m and 1387 m, respectively.

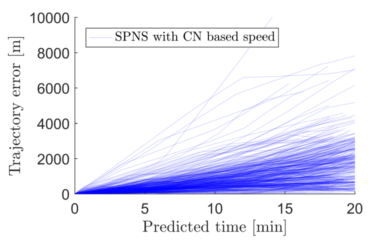
6.2.3 Trajectory errors

Figure 6.4 shows the trajectory error for the SPNS and the CVM method. The trajectory error accounts for predicted course but also for predicted speed along the path. Hence, the trajectory error will always be larger or equal to the path error. Figure 6.4a, Figure 6.4b and Figure 6.4c show the trajectory error for the test trajectories when CVM, SPNS with CN based speed and SPNS with constant speed is used, respectively. The CVM method still yields the largest errors. When comparing the two SPNS variants, the SPNS with constant speed seems to have a larger density of errors close to zero compared to the SPNS method with CN based speed. These observations are also supported by Figure 6.4d and Figure 6.4e where the SPNS method with constant speed method yields both the lowest median and the lowest average error values at all

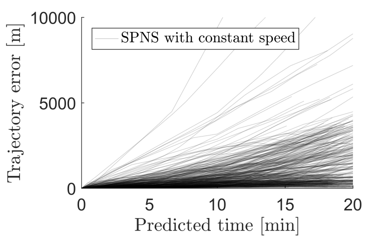
[4]Since a constant step length is used, the shape of any path is independent of the speed along it. However, since the prediction horizon is constant, the *length* of the path is speed dependent: low speed predictions yield short paths and vice versa. Therefore, in scenarios where the predicted path is shorter than the true path the predicted path is prolonged in order to evaluate the path error for the whole true path.



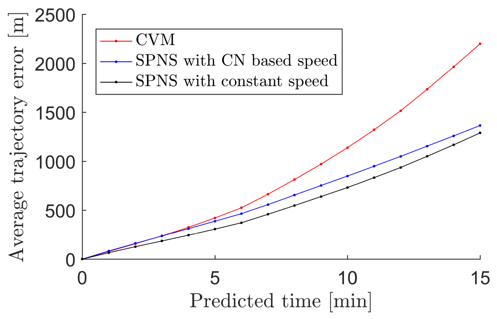
(a) CVM



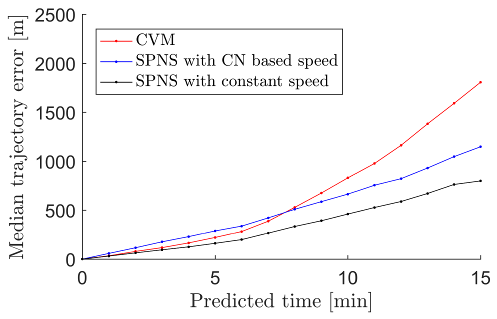
(b) SPNS, CN speed



(c) SPNS, const speed



(d) Average trajectory errors



(e) Median trajectory errors

Figure 6.4: Trajectory error comparison between SPNS and CVM for curved trajectories.

prediction times.

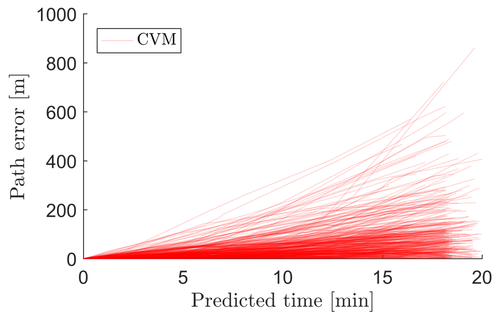
6.3 SPNS on straight line trajectories

The SPNS approach yields clearly lower path and trajectory errors than the CVM for curved trajectories. However, vessels follow relatively straight line paths most of their travel time. It is therefore important to evaluate how well the SPNS algorithm perform on straight line trajectories.

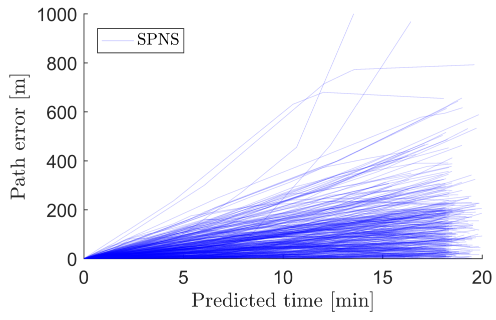
6.3.1 Test setup

A similar test as in Section 6.2 is performed, but now for straight line trajectories. That is, the same decision parameters are used in the SPNS algorithm (Table 6.1) and the 350 predicted trajectories are randomly drawn from those fulfilling the same requirements as in Section 6.2 except for a difference in requirement number 3:

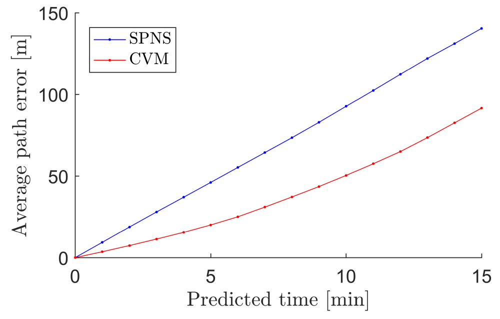
1. The elapsed time from the beginning to the end of the trajectory must be between 15 and 25 minutes.
2. The trajectory must consist of at least 4 AIS messages including the starting point.
3. The accumulated course change between every subsequent pair of AIS messages in the trajectory cannot exceed 10°, i.e., $\sum_{i=2}^{L} |\chi_i - \chi_{i-1}| \leq 10°$ where L is the number of AIS messages in the trajectory.
4. A minimum of 5 CNs around the starting position is required.



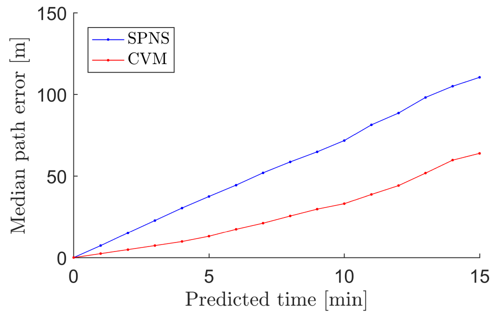
(a) Path error CVM



(b) Path error SPNS



(c) Average path errors



(d) Median path errors

Figure 6.5: Path error comparison between SPNS and CVM for straight line trajectories.

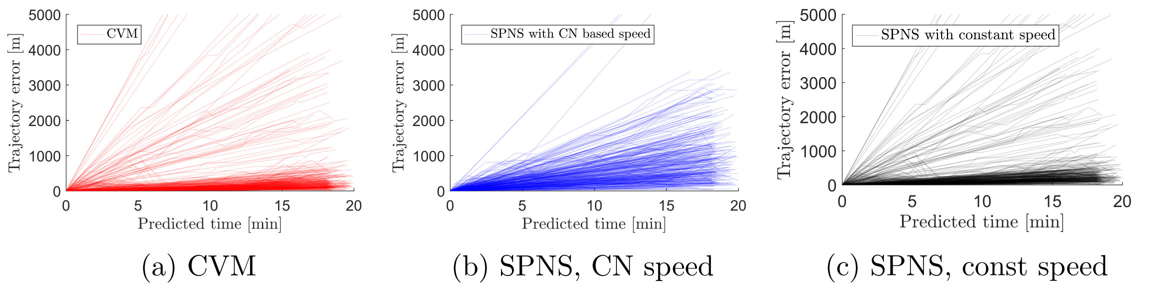
5. The starting speed must be at least 3 knots.

6.3.2 Path errors

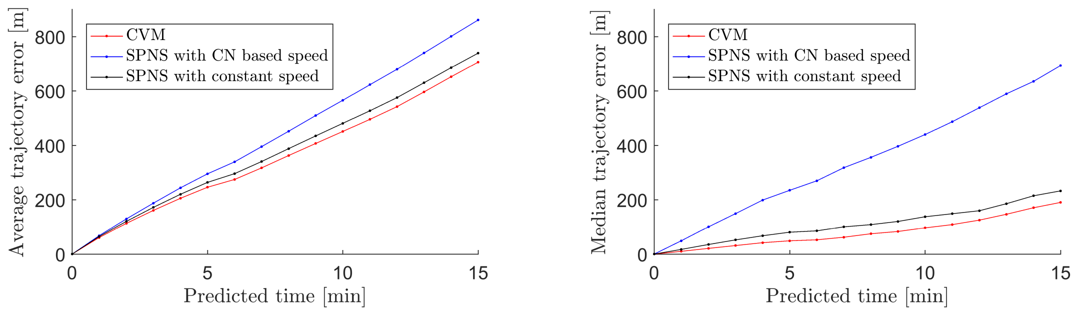
Path errors for all the 350 test trajectories are shown in Figure 6.5a and Figure 6.5b while the average and median path errors are shown in Figure 6.5c and Figure 6.5d. Not surprisingly, the CVM has relatively small path errors since we are limiting the accumulated course change of the test trajectories to only 10°. The SPNS method has larger path errors overall. Its average and median path errors are still relatively low with an average and median path error after 15 minutes of 141 m and 110 m, respectively, compared to the CVM's 92 m and 64 m.

6.3.3 Trajectory errors

The trajectory errors are shown in Figure 6.6. When comparing the CVM trajectory errors in Figure 6.6a with the trajectory errors from the SPNS with constant speed in Figure 6.6c, they seem to be almost identical. This is also supported by their very similar average and median trajectory errors, as seen in Figure 6.6d and Figure 6.6e. The speed prediction is the same in both methods and the two methods yield almost identical results because errors caused by wrong speed predictions are the main contribution to the trajectory error on close to straight line trajectories. It is clear from Figure 6.6d and Figure 6.6e that the CN based speed (median speed of all CNs) is overall less accurate than the constant speed.

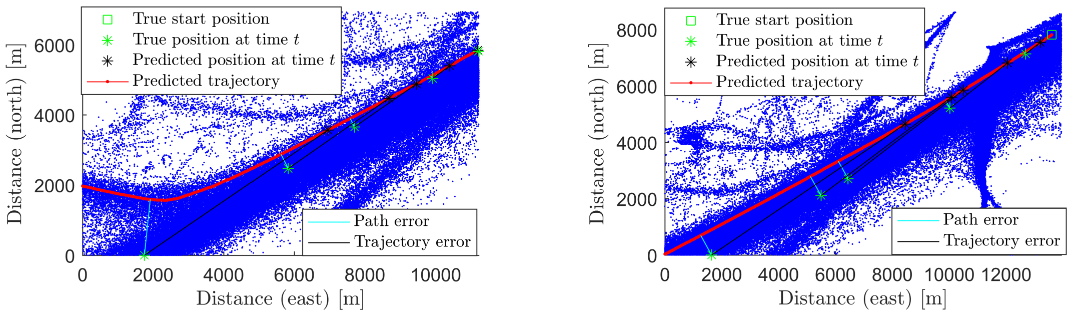


(a) CVM (b) SPNS, CN speed (c) SPNS, const speed



(d) Average trajectory errors (e) Median trajectory errors

Figure 6.6: Trajectory error comparison between SPNS and CVM for straight line trajectories.



(a) The largest path error (b) The largest trajectory error

Figure 6.7: The largest SPNS path error from Figure 6.5b after 15 minutes and the largest SPNS trajectory error from Figure 6.6b after 15 minutes.

The size of the path and trajectory errors in Figure 6.5b and Figure 6.6b may be misleading. The predicted trajectories in these two figures that yield the largest SPNS path error and the largest SPNS trajectory error at 15 minutes are illustrated in Figure 6.7. Despite Figure 6.7a being the largest path error from Figure 6.5b, the predicted path is fairly close to the true path. As a result, we can conclude that the SPNS algorithm have good path following capabilities for straight line trajectories. The predicted trajectory in Figure 6.7b, which has the largest trajectory error, is far off the true trajectory because the predicted speed is much lower than the actual speed. It can also be mentioned that the path error in Figure 6.7b is the second largest observed.

6.4 Evaluation of the speed prediction potential

The results in the two previous sections (Section 6.3 and Section 6.2) suggested that assuming constant speed throughout the prediction, i.e. $\hat{v}_k^+ = v_i$ for $k = 1, ..., K-1$, is preferred over the CN-based methods $\hat{v}_k^+ = \tilde{v}^c$ or $\hat{v}_k^+ = \bar{v}^c$. It is reasonable to think that the constant speed method is more accurate closer to the beginning of the prediction. This section evaluates the potential for improvement for the speed prediction by comparing the three methods v_i, $\tilde{v}^c$ and $\bar{v}^c$. The notation from the SPNS algorithm in Chapter 4 is used.

6.4.1 Test setup

The test is performed on 5000 trajectories in order to evaluate how the constant speed method typically deviates from the initial speed as the time elapses. The trajectories are randomly drawn from those trajectories fulfilling the following two requirements:

1. Maximum 10 minutes between any two subsequent messages.
2. The trajectory must be between 20 and 25 minutes.

No overlapping parts of any trajectory are included twice. The total number of tested AIS messages are 39641 as each trajectory consists of several messages. A summary of the test requirements with additional information is presented in Table 6.2. Since all tested trajectories end between 20 and 25 minutes[5] the tests in the next section are limited to 20 minutes such that an equal amount of trajectories (5000) is used in the evaluation at each evaluation point.

Explanation	**Value**
Number of test trajectories	5000
Total number of predictions	39641
Maximum time between subsequent messages	10 minutes
Minimum trajectory length	20 min
Maximum trajectory length	25 min

Table 6.2: Test requirements and information for the speed prediction evaluation.

6.4.2 Test results

Comparisons as functions of elapsed trajectory time

The average speed deviation and the 50th percentile speed deviation for the three tested methods as a function of elapsed trajectory time are shown in Figure 6.8. The

[5]The trajectory does not have to physically end in this time interval, but it must have at least one AIS message in this interval so the trajectory can be cut at this point.

percentiles are calculated from absolute deviation values. We can only evaluate the true speed deviation at the time instants of the AIS messages in each trajectory and since these time instants varies among the 5000 tested trajectories, linear interpolations are used to obtain 5000 speed deviation values at each minute starting from $t = 0$ to $t = 20$ minutes, where t is the elapsed time of a trajectory.

As expected, it does not seem to be any correlation between the CN-based speed deviations and the elapsed trajectory time t. Further, the speed deviation based on the median of all the CNs' speed values ($\tilde{v}^c$) yields both lower average absolute deviation and lower 50th percentile deviation than the method using the mean of all the CNs' speed values ($\bar{v}^c$). Naturally, the deviation of the constant speed method v_i is zero initially. The constant speed method has the lowest average values, but only up to 5 minutes. If we want to minimize the average absolute deviation, this result suggests to use v_i the first 5 minutes and then switch to $\tilde{v}^c$. However, the constant speed method's 50th percentile after 5 minutes is quite much lower than the same measure for $\tilde{v}^c$, and these two curves do not intersect before 13 minutes. These results suggest that the constant speed method is best between 0 and 5 minutes, worse than $\tilde{v}^c$ from 13 minutes and beyond and between 5 and 13 minutes the constant speed method has the lowest 50% deviations, but also a larger amount of very large deviations, relative to $\tilde{v}^c$, which has caused its high average values. Which method to use in the interval 5 - 13 minutes depends on whether we want to have as large amount of the lowest deviations as possible, at the cost of having a smaller part of the predictions very far off, or whether we want the majority of predictions to have a more stable deviation with less of the best predictions and less of the worst predictions.

The average absolute value and the 50th percentiles give us some overall indication of how well the various methods performs. What is also of interest is to know how much the deviations of the various methods vary. The speed deviation variances as a function of elapsed trajectory time for the three methods are illustrated in Figure 6.9a. Once again, as expected, the variances of the CN-based methods seems to be independent of elapsed time with close to constant values in the interval $26 - 28\,[kn^2]$. The variance of the constant speed deviation is steadily increasing and intersects with the $\tilde{v}^c$ variance after only 3.5 minutes. Both the CN-based methods have significantly lower variances from 3.5 minute and beyond, where the $\bar{v}^c$ has slightly lower variance than $\tilde{v}^c$. Figure 6.9b shows the mean speed deviation values (not absolute deviation) as a function of elapsed time. This plot is included to make sure that non of the methods are highly biased. A method with very low variances can be misleading if the mean deviation values are far from zero. However, all three methods have mean values relatively close to zero. Since the deviations are calculated as $deviation = true\,speed - predicted\,speed$, the CN-based methods' positive mean deviation values imply that the predicted speeds on average are lower than true speed, while the constant speed's negative values (up to 17 minutes) imply speed predictions larger than true speed, on average.

The variance results better confirms the suggestion of replacing the constant speed method by a CN-based method somewhere between 5 and 13 minutes. A change of method does not have to occur instantly. A weighted combination, for instance $\hat{v}_k^+ = (1 - a)v_i + a\tilde{v}^c$ where $a \in \Big[0, ..., 1\Big]$ can be applied. The relative weighting parameter a can increase from 0 to 1 in the interval we want the transition to occur.

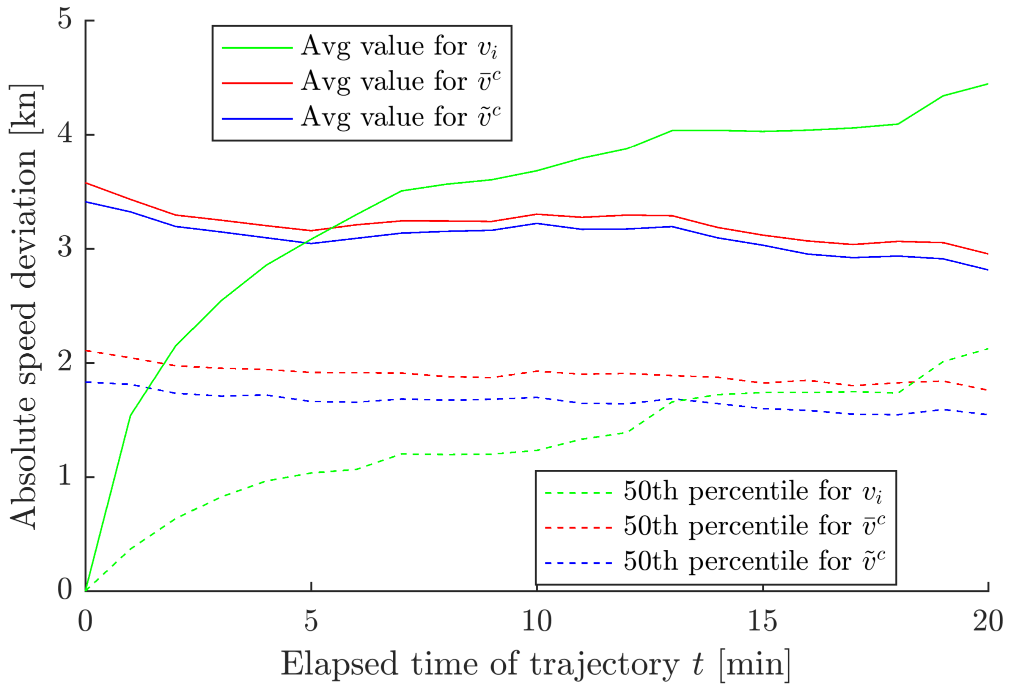
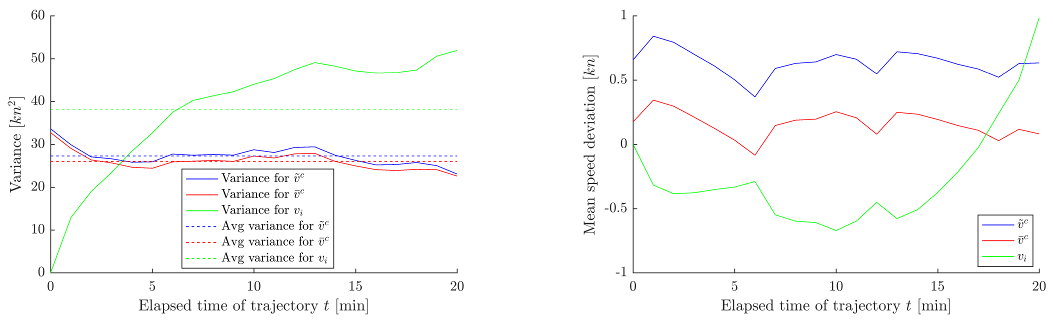


Figure 6.8: The average speed deviation and the 50th percentile speed deviation for the three tested methods as a function of elapsed trajectory time for 5000 trajectories. The three different methods are: The constant speed from the initial known speed at $t = 0$ (v_i), the mean of all the CNs' speed values ($\bar{v}^c$) and the median of all the CNs' speed values ($\tilde{v}^c$).



(a) (b)

Figure 6.9: Figure 6.9a show speed deviation variance for the three tested methods as a function of elapsed trajectory time for 5000 trajectories. The average variance values are also included, illustrated with dashed lines. Figure 6.9b show the mean values of all the 5000 interpolated speed deviations as a function of elapsed trajectory time.

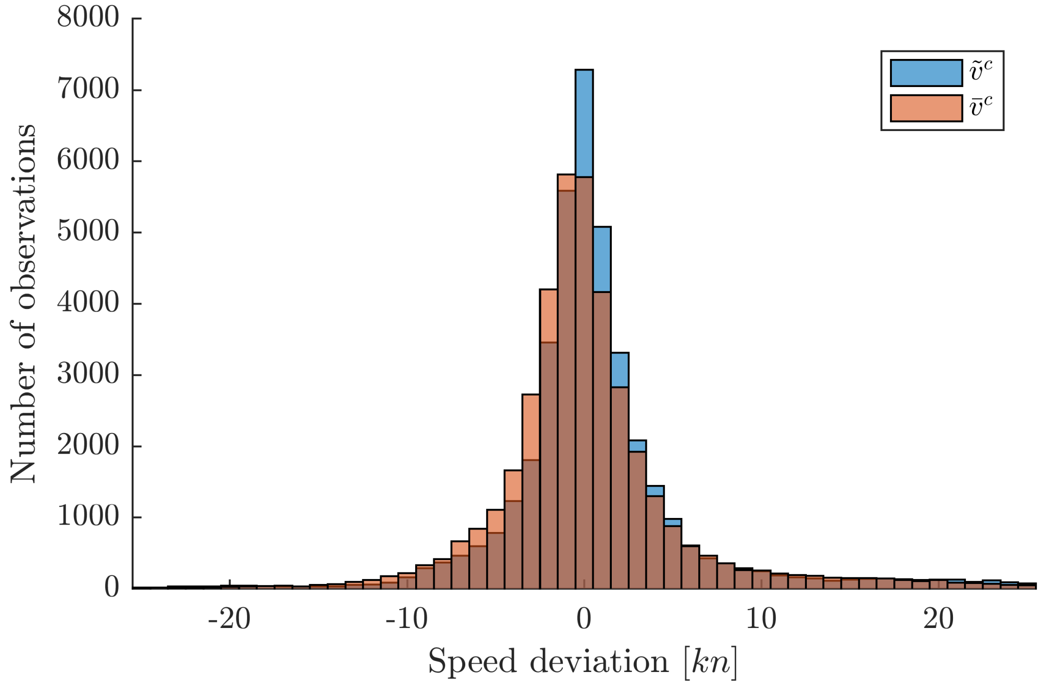


Figure 6.10: Speed deviation distribution for the CN-based methods for the 5000 trajectories independent of elapsed trajectory time (a total of 39641 observations for each method).

Comparison of the CN-based methods independent of time

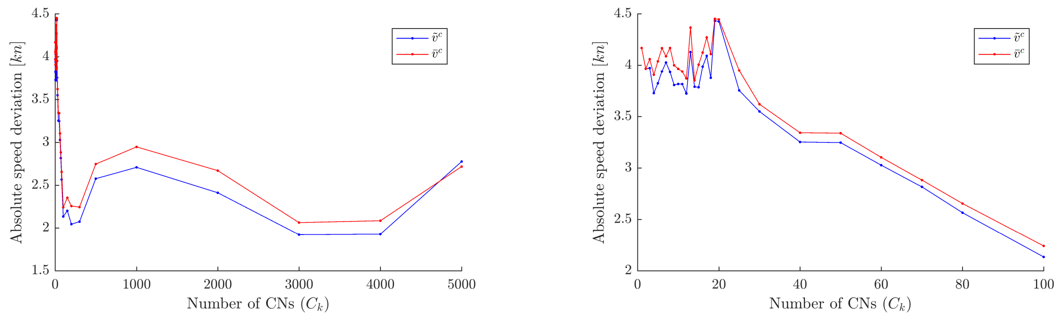
Since the speed deviation from the two CN-based methods are independent of elapsed trajectory time it is more interesting to compare their overall performance independent of time (interpolated values are not longer necessary). A distribution of speed deviation is shown in Figure 6.10. Clearly, the $\tilde{v}^c$ method has a larger amount of deviations closer to zero. Relative to each other, the $\tilde{v}^c$ method tend to yield a too low speed while the $\bar{v}^c$ method tend to yield a too high speed. Although the $\tilde{v}^c$ method had a slightly higher variance in Figure 6.9a, the $\tilde{v}^c$ method will be preferred over the $\bar{v}^c$ method due to its significantly higher number of small deviations.

The CN-based methods as a function of the number of CNs

Lastly, Figure 6.11 shows how the absolute speed deviation for the two CN-based methods change with the number of CNs (C_k). Figure 6.11a shows the results for the whole range of C_k while Figure 6.11b shows the same results but is limited to $C_k = 100$. Not surprisingly, the speed deviation is largest when C_k is very low, especially when it is less than 20[6]. This information can be incorporated into the speed prediction suggested above, for instance by using the constant speed whenever the number of CNs is below a certain limit. The speed prediction may then be:

$$\hat{v}_k^+ = (1-a)v_i + a\tilde{v}^c \tag{6.1}$$

[6]It is possible that the number of CNs is not the direct cause for these results. It may be, for instance, that vessels in rarely navigated areas tend to have a less determined planned trajectory and hence larger variability in speeds. Rarely navigated areas implies less AIS-dense areas and hence a lower number of CNs.



(a) Full C_k range

(b) Limited C_k range from Figure 6.11a

Figure 6.11: The average speed deviation for the two CN-based methods as a function of the number of CNs (C_k). The average values are calculated from all predictions within a given C_k interval in order to have enough observations at each evaluation point. Each interval equals the range of CNs between two subsequent dots, as seen in the figures. For instance, the average absolute speed deviation for states with $C_k = 5000$ is calculated as the average deviation from all states with $C_k \in \langle 4000, 5000\big]$.

with

$$a = \begin{cases} 0 & if \quad t < t_a \quad or \quad C_k < minCn \\ \frac{t-t_a}{t_b-t_a} & if \quad t_a \leq t \leq t_b \\ 1 & if \quad t > t_b \end{cases} \tag{6.2}$$

where t, t_a and t_b are the predicted time (running variable), start time of transition and end time of transition, respectively. The two latter parameters were suggested above to be set to $t_a = 5$ minutes and $t_b = 13$ minutes. For simplicity, the transition between the constant speed and the CN-based method in (6.2) is linear. Based on Figure 6.11, the number of CNs $minCn$ is suggested to be set to approximately 20.

6.5 NCDM test results

6.5.1 Test setup

Algorithm 2 is tested with the decision parameters in Table 6.3 from a set of random initial states. Ten of these predictions are illustrated in Figure 6.12. The predicted trajectories are plotted with a very weak, red color in order to better see the prediction density: a stronger red color indicates a higher density. Be aware that in Table 6.3 the search radius and the step length are set equally to $\Delta l = r_c = 100$ m, as opposed to in in the SPNS test in Table 6.1 where $\Delta l = 2r_c = 100$ m. A short step length is necessary to be able to follow tight turns, but a short search radius can often result in the number of CNs ($C_{k,j}$) being very low or even zero. By reducing the $\frac{\Delta l}{r_c}$ ratio from 2 to 1 we double the search radius while keeping the same step length.

Decision parameter	**Value**	**Explanation**
r_c	100 m	Search radius for CNs
Δl	100 m	Prediction step length
$\Delta\chi$	35°	Maximum course deviation for CNs
$N_{1,1}$	500	Number of child nodes from start node
$N_{k,j}$, $k \geq 2, \forall j$	$min(1, C_{k,j})$	Number of child nodes from any node except for the start node
J^{max}	500	Maximum width of the prediction tree
t_h	20 min	Prediction horizon

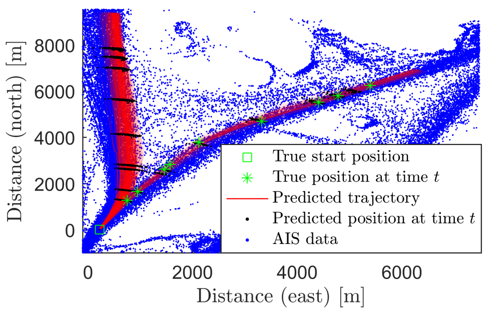
Table 6.3: Decision parameters for the NCDM test

6.5.2 NCDM's potential and shortcomings

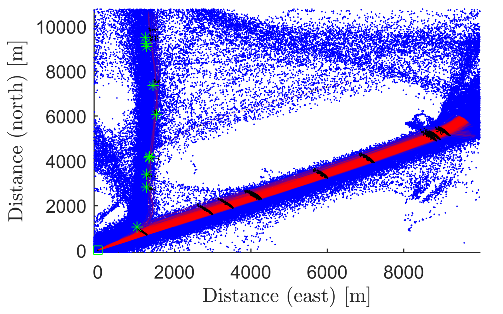
Although the 10 selected plots in Figure 6.12 do not represent the overall prediction results, they show that Algorithm 2 is capable to divide into multiple branching sea lanes, a feature which Algorithm 1 does not possess. As the prediction time increases the predicted areas (black dots) are typically expanding. This is reasonable since the uncertainty of the predicted state increases with time. Figure 6.12a - Figure 6.12f show predictions where branching occurs. In the first two cases, the true paths fall inside the least dense sea lanes. More precisely, they fall into the sea lanes which only hold 7% and 2% of all predicted paths (J^{max}), respectively. However, these two single results alone can neither verify nor disprove whether the distributions of predictions are statistical reasonable. Further, Figure 6.12e and Figure 6.12f show that with a small enough maximum course deviation parameter ($\Delta\chi = 35°$ in this test), the predictions do not expand to crossing sea lanes. The crossing sea lanes in the two previously mentioned examples are caused by passenger ferries traveling back and forth the same route, causing very dense sea lanes where most other vessel types do not travel. Therefore is the preferred behavior to avoid such crossing sea lanes[7].

[7]More testing must be done investigate to what extent vessels move from one sea lane to a crossing sea lane.

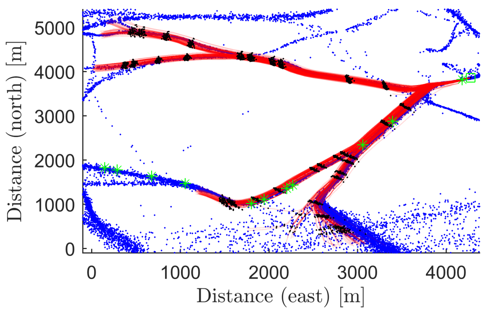
As seen from Figure 6.12i and Figure 6.12j, Algorithm 2 is also able to follow straight line trajectories in areas with various AIS data density.



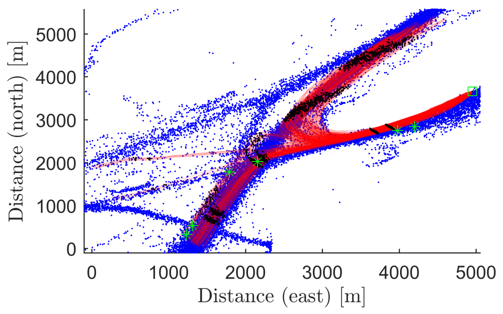
(a)



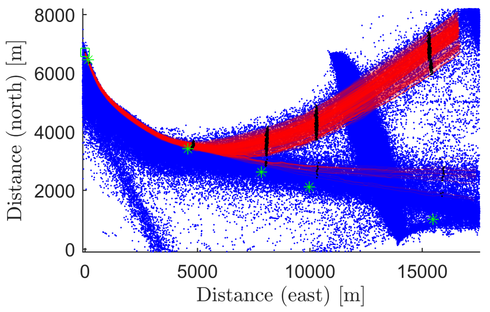
(b)



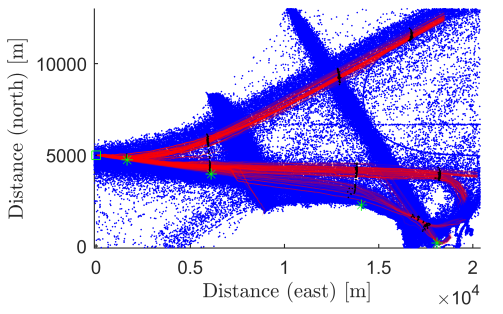
(c)



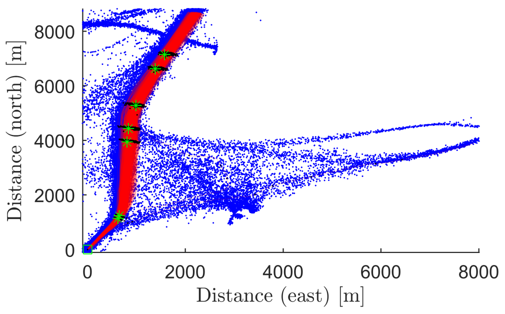
(d)



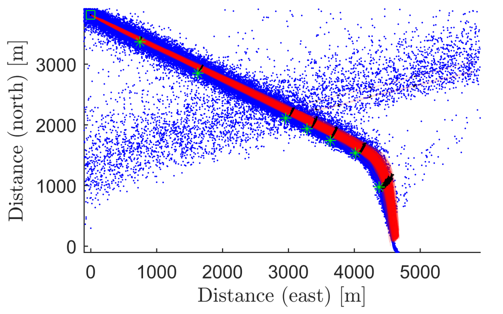
(e)



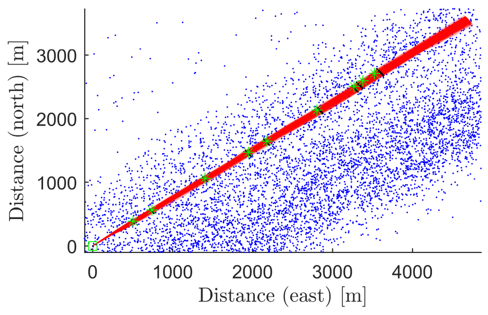
(f)



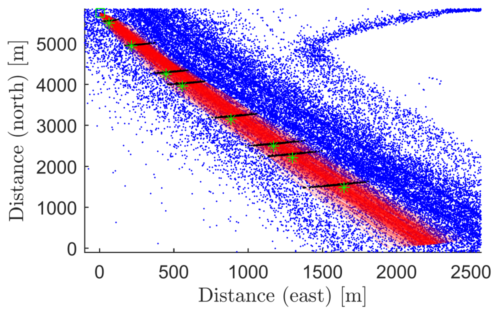
(g)



(h)



(i)



(j)

Figure 6.12: The NCDM algorithm tested on a set of manually chosen trajectories. Explanation of the sub figures' markers are only shown in Figure 6.12a and notice that the last three markers are either stronger or larger than in the actual plot.

6.5.3 Statistical testing

There are several performance measures of interest for the NCDM predicted trajectories, such as:

1. How well are the true paths captured by the predicted paths (independent of time)? For instance, what is the lowest path errors?

2. How close are the predicted trajectories to the true trajectories?

3. How well does the density of predicted trajectory states (positions at given timestamps) reflect the probability of the vessels being there at the same instant of time?

4. How well does the distribution of predicted trajectories in branched sea lanes reflect vessels probability of sea lane choices?

We will in the following run statistical tests on the dataset to evaluate the performance of the trajectory predictions (point 2 in the list above). An evaluation of whether the density of predicted trajectories reflect the probability of vessel movement or not will not be covered but is important to test since such properties in a COLAV system would be an advantage for an ASV's path planning algorithm.

A necessity for the upcoming tests is to automatically cluster the predicted positions without knowing the number of clusters. The widely used density-based spatial clustering of applications with noise (DBSCAN) algorithm is implemented for this purpose. One or more clusters are obtained at every known timestamp of the tested true trajectory. An example of clusters is shown in Figure 6.13. The DBSCAN is, as the name suggest, a density based clustering algorithm. Given a set of points in a space to be clustered, the algorithm differ between three classes of points: *Core points*, *density-reachable points* or *outliers*. More specifically [20]:

1. A point p is a *core point* if at least *minPts* number of points (including p itself) are within a distance ϵ of it. Core points are said to be *directly reachable* from p.

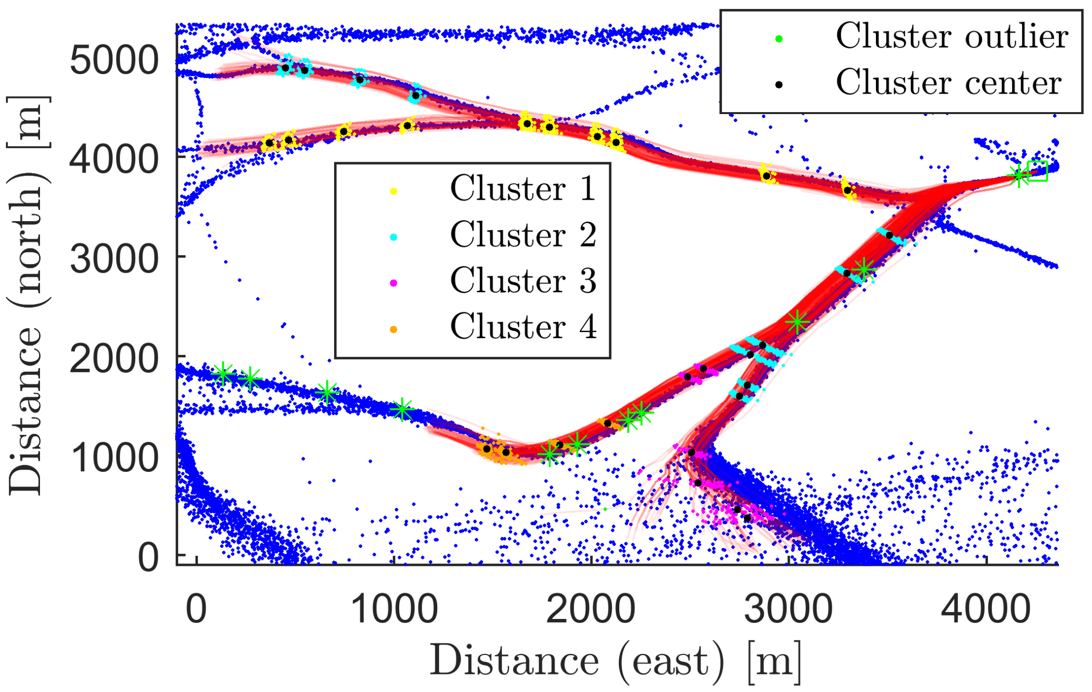


Figure 6.13: An example of the DBSCAN clustering algorithm on a NCDM predicted trajectory.

2. A point q is *density-reachable* from p if there is a sequence of points $p_1, ..., p_n$ with $p_1 = p$ and $p_n = q$, where each p_{i+1} is directly reachable from p_i. That is, all the points in the sequence, with the exception of p_n, must be core points.

3. All points that are neither core points nor density-reachable points are outliers.

If p is a core point, it forms a cluster with all other points that is either directly reachable from it (other core points) or density-reachable points. Outliers are omitted and do not belong to any cluster. A distance metric and the two decision parameters ϵ and $minPts$ have to be chosen.

Test setup

The DBSCAN algorithm is run with an Euclidean distance metric and with ϵ = 100 m and $minPts$ = 10 positions. The NCDM algorithm is used with decision parameters listed in Table 6.3. The test is performed on the first 400 randomly drawn trajectories that satisfy the requirements given in Table 6.4. The future true trajectory is always removed from the dataset prior to prediction and no initial state is used more than once. Approximately 57% of the randomly checked states did not meet the requirements, where 35% of these were due to too few CNs initially, 33% were due to exceeding the maximum time between subsequent AIS messages, 28% were due to too low initial speed and the remaining 4% were due to the last trajectory message not falling inside the required time interval. The DBSCAN decision parameters are given in Table 6.5. Of all the predicted positions fed to the NCDM algorithm are only 0.3% classified as cluster outliers and the rest are assigned to various clusters.

Explanation	**Value**
Number of test trajectories	400
Maximum time between subsequent messages	10 min
Minimum number of initial CNs (C_1)	10
Minimum initial speed (v_i)	5 kn
Minimum trajectory length	15 min
Maximum trajectory length	20 min

Table 6.4: Test requirements for the NCDM trajectory evaluation

Explanation	**Value**
DBSCAN distance metric	Euclidean
ϵ	100 m
$minPts$	10 positions

Table 6.5: Decision parameters for the DBSCAN clustering algorithm.

Distances to the nearest clustered positions

Figure 6.14 show the average value together with three different percentiles of the distances to the nearest predicted positions as a function of elapsed trajectory time. There are one or more clusters for each time instant in the true trajectory. The distance from the true position to the nearest position in any of the clusters is used as a performance measure. Predicted positions classified as outliers by the DBSCAN algorithm is not regarded. As before, in order to obtain estimated values from all test trajectories at each evaluation point (each minute), all values are linearly interpolated before the average and percentile values are calculated at each evaluation point.

As seen from Figure 6.14, all measures are increasing with elapsed trajectory time. As before, this is expected since the deviation is zero initially and as time goes by the range of possible vessel positions increases. The 25th and 50th percentiles stay low for the whole prediction horizon with maximum values of 77 m and 236 m after 15 minutes. The 75th percentile reaches a maximum value of 864 m after 15 minutes. The 25% gap from the 50th percentile to the 75th percentile is very large relative the 25%gap between the former two percentiles. Furthermore, the average distances are typically above the 75th percentile. In other words are more than 75% of the lowest distances less than the average distance. This is caused by the upper 25th percentile (the 25% largest values) being very large and pulling the average values up.

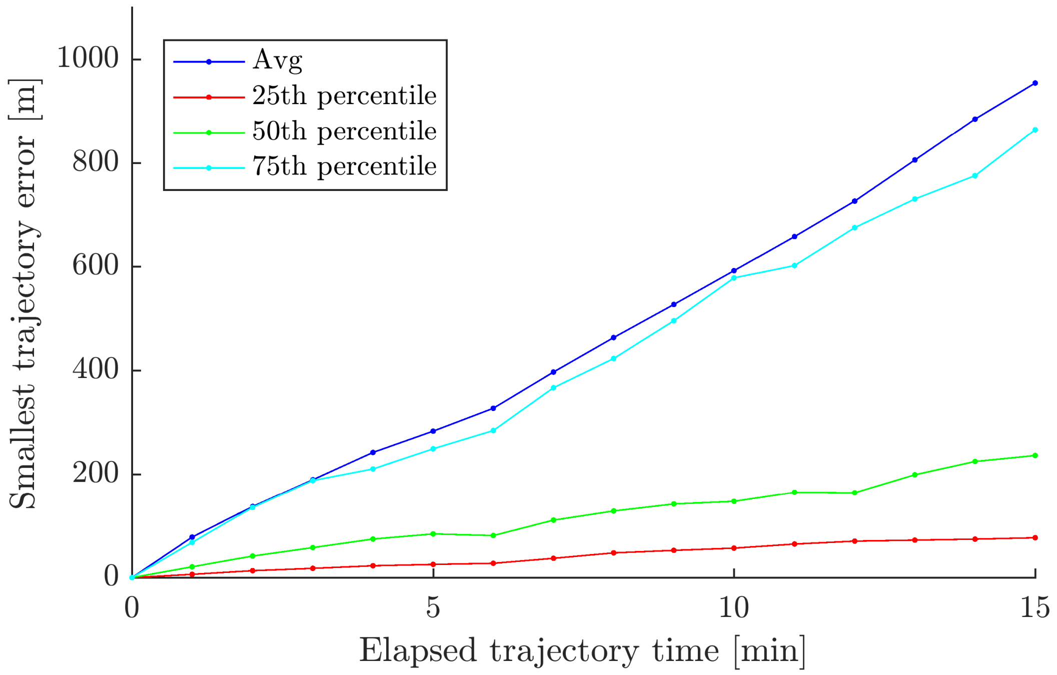


Figure 6.14: Three different percentiles and average values of the distances between the true positions and the nearest clustered positions, plotted as a function of elapsed trajectory time.

Distances to the nearest cluster centers

Figure 6.16 show the average value and three different percentiles of the distances to the nearest cluster centers as a function of elapsed trajectory time. The cluster center is calculated as the center of mass with all cluster positions regarded as point masses with masses equal to 1. Hence, the center of any given cluster is calculated as

$$\boldsymbol{p_{clus}} = \begin{bmatrix} \lambda_{clus} & \phi_{clus} \end{bmatrix} = \begin{bmatrix} \frac{\sum_{j=1}^{N_{clus}} \lambda_j}{N_{clus}} & \frac{\sum_{j=1}^{N_{clus}} \phi_j}{N_{clus}} \end{bmatrix} \tag{6.3}$$

where N_{clus} is the number of positions in a cluster, ϕ_j is latitude and λ_j is longitude. See Figure 6.15 for an illustrative example on how the distances are calculated for a single trajectory prediction. All distance values are linearly interpolated as before.

As seen in Figure 6.16, the various measures looks very similar in both shapes and values to those in Figure 6.14. Naturally, since the distance to the nearest cluster center always is larger or equal to the distance to the nearest clustered position, the values in Figure 6.16 are slightly larger for all elapsed trajectory times. The 25th, 50th and 75th percentiles are increasing up to a maximum of 162 m, 368 m and 924 m, respectively. Once again, the upper 25th percentile must consist of relatively large values since the average and the 75th percentile almost coincide.

Further analysis

To get an idea about why some predictions yield very poor results, the predictions yielding the 20 largest distances to cluster centers at the end of the 15 minutes tra-

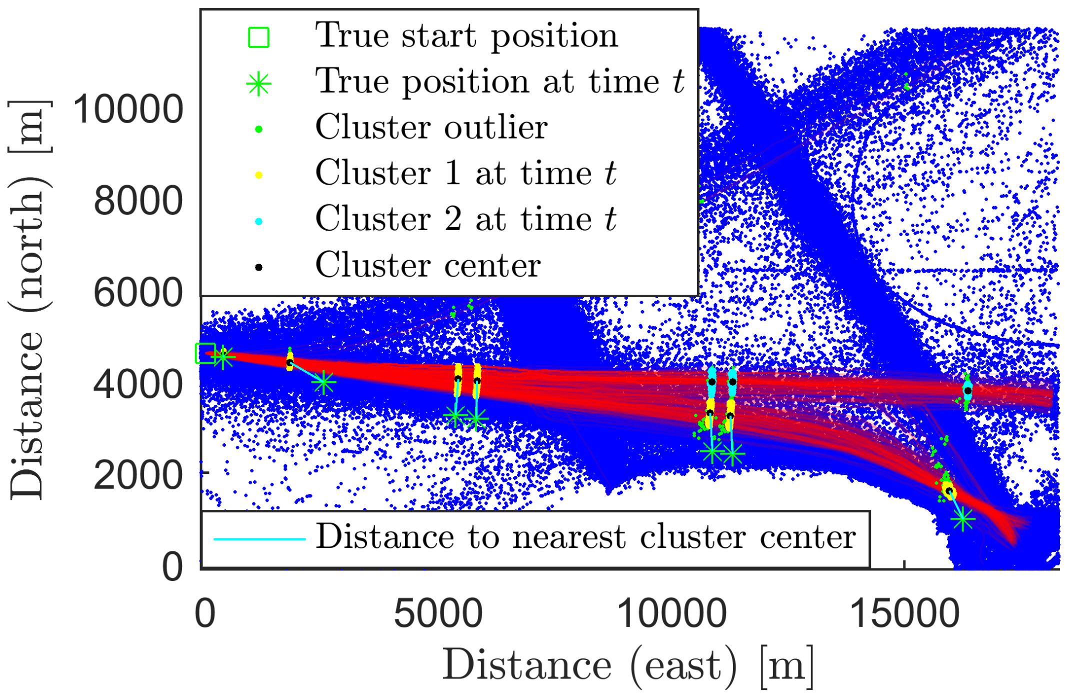


Figure 6.15: An illustration of distances from true positions to their nearest cluster centers.

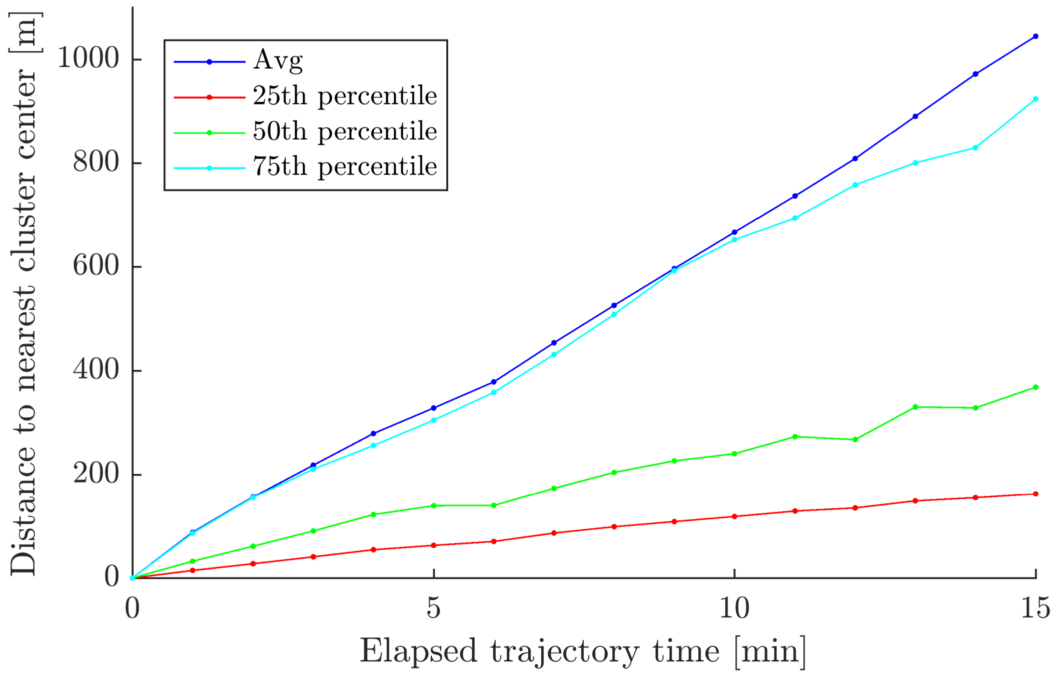


Figure 6.16: Three different percentiles and average values of the distances between true positions and their nearest cluster density centers, plotted as a function of elapsed trajectory time.

jectory are plotted in Appendix A. By observing these 20 predictions we can conclude that:

1. 10/20 of the large end point deviations are due to poor speed prediction as the vessel has changed its speed significantly since the initial state.
2. 6/20 of the large end point deviations are a result of the predicted trajectories not covering the true path fully (Figure A.1m, Figure A.1j, Figure A.1f, Figure A.1e, Figure A.1b, Figure A.1a)
3. 2/20 of the large end point deviations are caused by the vessels making a too tight turn for the chosen decision parameters (Figure A.1s, Figure A.1l).
4. 1/20 of the large end point deviations is caused by the vessel making a U-turn which the algorithm will rarely follow (Figure A.1o).
5. 1/20 of the large end point deviations is a result of the predicted branch terminating before it reaches the final prediction time. This is caused by the vessel reaching land/port, meaning that no CNs exist and further prediction is impossible (Figure A.1g).

Although not a statistical proof, these results indicate that about half of the largest deviations are a result of incorrect speed prediction. This seems reasonable as inaccurate speed prediction was discovered in Section 6.3 to be the main contribution to the trajectory errors. Therefore, there is a potential to significantly reduce the average distances to the nearest cluster centers in Figure 6.16 by implementing a smarter speed prediction, for instance the one suggested in (6.1).

The second largest portion of the deviations (6/20) is a result of the predictions not covering the true path to the end. In two of these scenarios (Figure A.1m and Figure A.1e), the vessels choose to branch out from a very dense sea lane onto a much less dense lane. If the NCDM algorithm's distribution of predictions in branched sea lanes reflect the statistical probability of vessels' sea lane choices (which is not yet verified), then such scenarios should not occur more often than what the historical AIS data reflects, which is very rarely. Such rare scenarios, however, are difficult to avoid with the highly data-driven algorithm. In the remaining 4 of these 6 scenarios, there are not enough predictions close enough to form clusters with the chosen NCDM decision parameters.

The last observed case, where a vessel reaches a port or land, will typically result in poor predictions for the NCDM algorithm since the closer a vessel get to land, the lower the speed, while the predicted speed is held constant. However, the algorithm can be expanded to detect when land is approaching or reached. Land is typically reached if there is a consistent, very low speed among the CNs and if also there is a sudden change of AIS density from a typically very high number at the port/land to zero. Zero CNs will cause the predicted trajectories to terminate. The AIS dataset in this thesis was prior to reception already filtered to exclude AIS messages with speed values below 0.5 kn. If these values are included in the dataset will detection of ports and land likely be easier as the density of low speed AIS messages should be high in such areas.

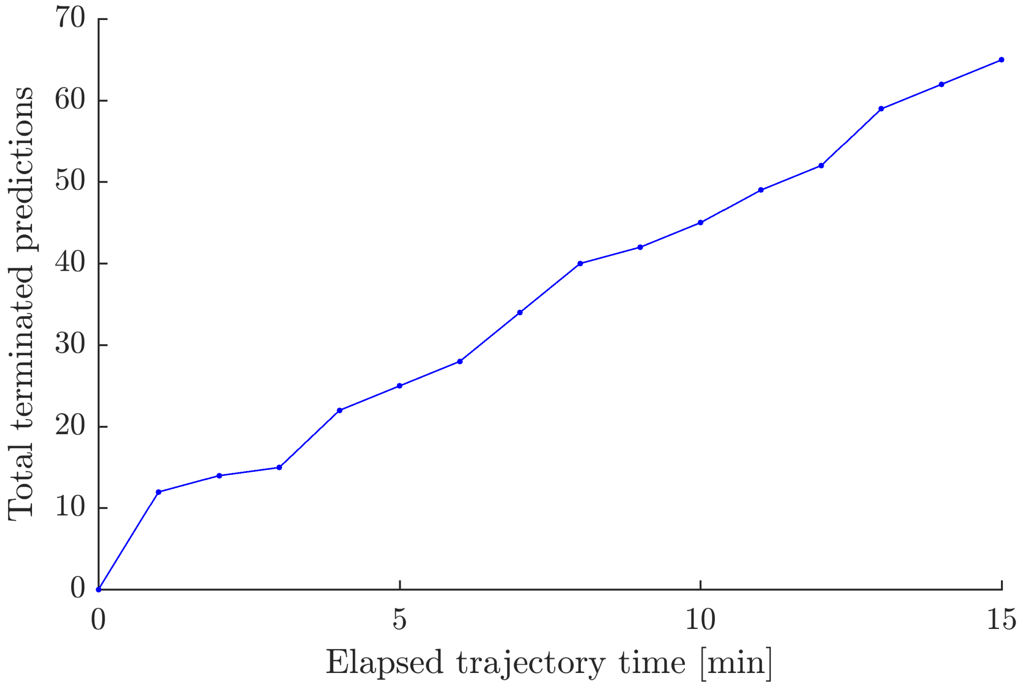


Figure 6.17: The total number of terminated predictions at each time interpolation.

Each interpolated value in Figure 6.16 is calculated from the 400 tested predictions. However, some predictions terminate[8] before they reach the end of the prediction horizon. This results in a lower number of evaluation points as the elapsed trajectory time increases. The total number of terminated predictions as a function of interpolated time is shown in Figure 6.17. More and more predictions terminate as the prediction time elapse, until a total of 65 out of 400 predictions have fully terminated after 15 minutes. Clearly, we want as few predictions as possible to terminate before the end of the prediction horizon.

Appendix B shows 20 of the 65 terminated predictions. The 20 predictions are drawn randomly. By observing these plots it can be concluded that 20/20 of the predictions terminate because the vessels reach land. In most of the predictions, the vessels either turn back where they came from or move on towards other destinations after a short stop at land. Note that the maximum possible time a vessel can be moored or at rest is 10 minutes since the maximum time between subsequent AIS messages is set to 10 minutes in Table 6.4 and because AIS messages with speed over ground less than 0.5 kn are not present in the dataset. These behaviors are explained by the fact that these vessels are passenger ferries traveling back and forth the same route.

Comparison between the constant speed and the speed transition method

This subsection compares the NCDM algorithm with two variants of the speed prediction: the constant speed (CS) method and the speed transition (ST) method. The

[8]A prediction terminates if there are no clusters formed at a given time instant, although cluster outliers may exist.

former is already tested in the previous subsections, while the latter method is the speed prediction method given by (6.1) and (6.2) as suggested in Section 6.4.2. It is repeated here for convenience:

$$\hat{v}_k^+ = (1-a)v_i + a\tilde{v}^c$$

with

$$a = \begin{cases} 0 & \mathit{if} \quad t < t_a \quad \mathit{or} \quad C_k < minCn \\ \frac{t-t_a}{t_b-t_a} & \mathit{if} \quad t_a \leq t \leq t_b \\ 1 & \mathit{if} \quad t > t_b \end{cases}$$

where t, t_a and t_b are predicted time (running variable), start time of transition and end time of transition, respectively, v_i is the constant speed and $\tilde{v}^c$ is the median speed of all CNs. Recap that the decision parameters $t_a = 5$ min, $t_b = 13$ min and $minCn = 20$ are chosen based on the statistical testing in Section 6.4.2. The NCDM algorithm is run with the same decision parameters and on the same 400 trajectories as in Section 6.5.3, but now with the suggested speed transition method. The performance is compared with the constant speed method and shown in Figure 6.18.

Both Figure 6.18a and Figure 6.18b show the same trend: by replacing the CS method with the ST method, the average error values and the 75th percentile error values are reduced for $t > t_a = 5$ min. At $t = 15$ min, the average error in Figure 6.18b is reduced by 11.6% down from 1045 m to 924 m. At the same time, the 25th and 50th percentiles are all larger with the ST method. This means that the ST method reduces the error in situations where the constant speed gives very poor predictions, but it increases the errors in trajectories with close to constant speed. It is not necessarily preferred to try to minimize the average prediction error in COLAV. If the prediction error is reduced from 5000 meter to 4000 meter it can have a significant influence on the average error, but it is not of much added value in terms of avoiding collision. The constant speed method is therefore preferred over the speed transition method.

A deeper look into the prediction errors suggest the following hypothesis:

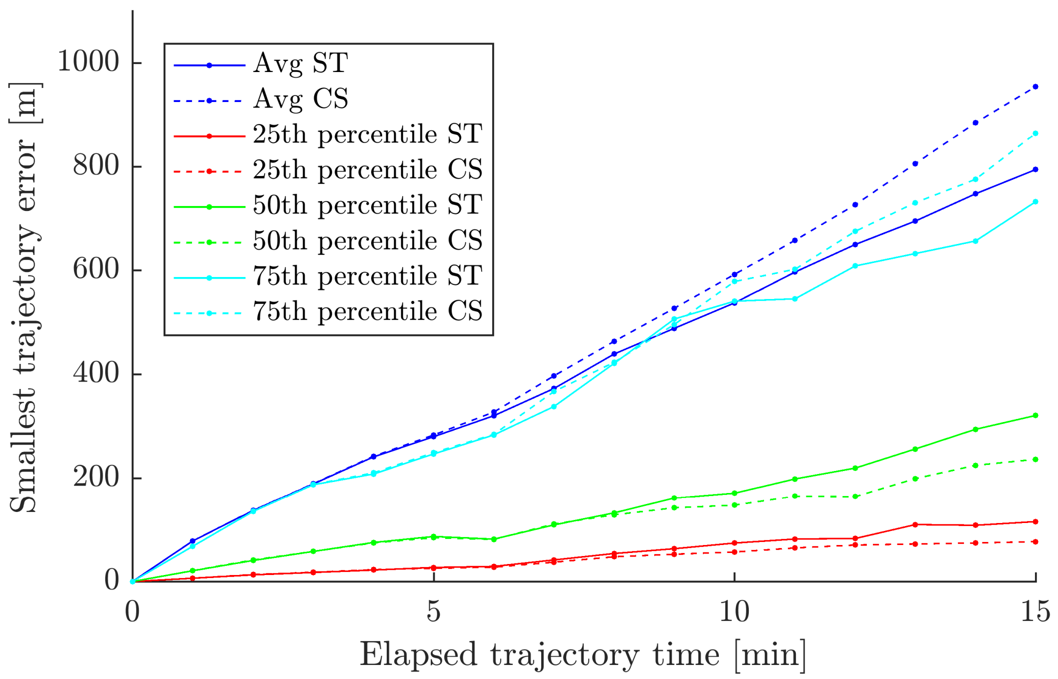
1. Vessels in transit, i.e., a few minutes after departure and a few minutes prior to arrival at land, tend to keep a close to constant speed. Hence, predicting with constant speed is reasonable.

2. In the first few minutes after departure, and the last few minutes prior to arriving at land, the vessels' speed is better predicted from the historical speeds in the area. Hence, a CN based speed prediction method is preferred.

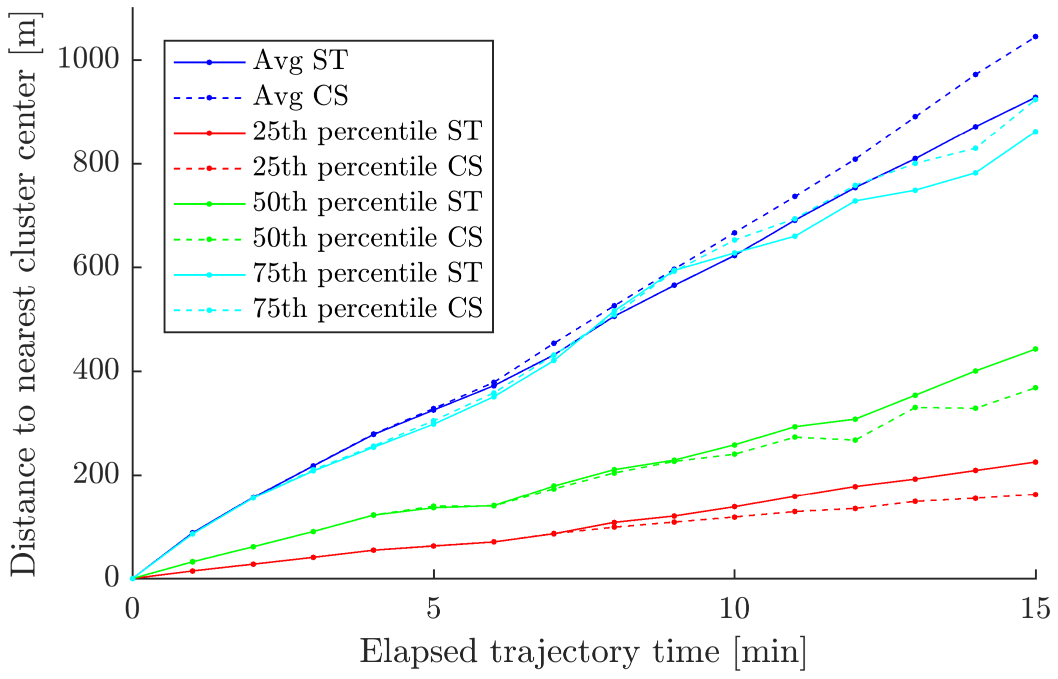
A more accurate speed prediction can be achieved if the above hypothesis is reasonable by:

1. First detecting if the vessel is approaching land or if it has just departed.

2. Decide a point on the trajectory where a switch of speed prediction will occur. The trajectory is predicted with the constant speed method when the vessel is in transit, while just after departure and prior to arrival at land, a CN based speed method is applied.



(a) Distance to the nearest clustered position



(b) Distance to the nearest cluster density center.

Figure 6.18: Comparison of the constant speed (CS) method and the speed transition (ST) method. Three different percentiles and average values for both methods are presented. The distance to the nearest clustered positions are shown in fig. 6.18a while the distance to the nearest cluster density centers are shown in fig. 6.18b.

As already mentioned above, detection of land may be discovered whenever there is a very dense area with a consistent low speed and where the density suddenly reduces to zero if the predictions are continued in the direction of travel onto the mainland. As seen from the figures in Appendix B, the termination of predictions (reduction of density to zero) seem to be a good indication of land being reached even without checking for speed in the area. Areas of departure can be discovered in a similar way.

6.6 Computational time analysis

The computational time tests are run on an Intel(R) Core(TM) i5-3470 CPU 3.20 GHz processor with 16 GB RAM and Windows 7 64-bit operating system.

6.6.1 The bottleneck

The NCDM algorithm spends on average 99.8% of its computational time while searching for CNs. In the original SPNS implementation [1], the CNs are searched for in a n x 4 matrix where n is the number of AIS messages, the first column is sorted longitude values and the remaining three columns are the corresponding latitude, course and speed values, respectively. Given the AIS dataset is already sorted with respect to longitude values, this approach finds all the CNs around a predicted state $\hat{\boldsymbol{X}}_k$ with position $\hat{\boldsymbol{p}}_k = \begin{bmatrix}\hat{\lambda}_k & \hat{\phi}_k\end{bmatrix}^T$ and course $\hat{\chi}_k$ by:

1. First finding the state $\boldsymbol{X}_i$ in the AIS matrix with longitude value λ_i closest to the predicted state's longitude value, using a binary search in $\mathcal{O}(\log n)$ time.

2. Secondly, running a sequentially search in $\mathcal{O}(n)$ time through all the sorted longitude values within the range $\hat{\lambda}_k - (r_c + d(\hat{\boldsymbol{p}}_k, \boldsymbol{p}_i)$[9] and storing the states that are within the Euclidean search radius r_c of the predicted state $\hat{\boldsymbol{X}}_k$. Notice that the distance $d(\hat{\boldsymbol{p}}_k, \boldsymbol{p}_i)$ between the predicted state $\hat{\boldsymbol{X}}_k$ and the state with the closest longitude value $\boldsymbol{X}_i$ is included to make sure that all states within the radius r_c of $\hat{\boldsymbol{X}}_k$ are covered.

3. Lastly, running a linear search in $\mathcal{O}(n)$ time through the states discovered in step 1 an 2 and eliminate the states which does not satisfy the maximum course deviation requirement $\Delta\chi$, i.e., the states with $\chi_i \notin S = \left[\hat{\chi}_k^- - \Delta\chi, \hat{\chi}_k^- + \Delta\chi\right]$ as defined in (4.5).

6.6.2 Computational time comparison between k-d tree storage and longitude sorted matrix storage

In order to reduce the computational time are the longitude and latitude values stored in k-dimensional tree structure (k-d tree for short) with $k = 2$ dimensions. The k-d tree is a binary tree in which every node is a k-dimensional point and it is a useful tree structure for range searches [21]. A range search in a k-d tree with N nodes are shown

[9]Assuming that $\hat{\lambda}_k$, r_c and $d(\hat{\boldsymbol{p}}_k, \boldsymbol{p}_i)$ all are represented in the same unit, i.e., either in meters or degrees.

to have a worst case time of $\mathcal{O}(kN^{1-\frac{1}{k}})$ [22] which with $N = 2n$ and $k = 2$ becomes $\mathcal{O}(2\sqrt{2n})$.

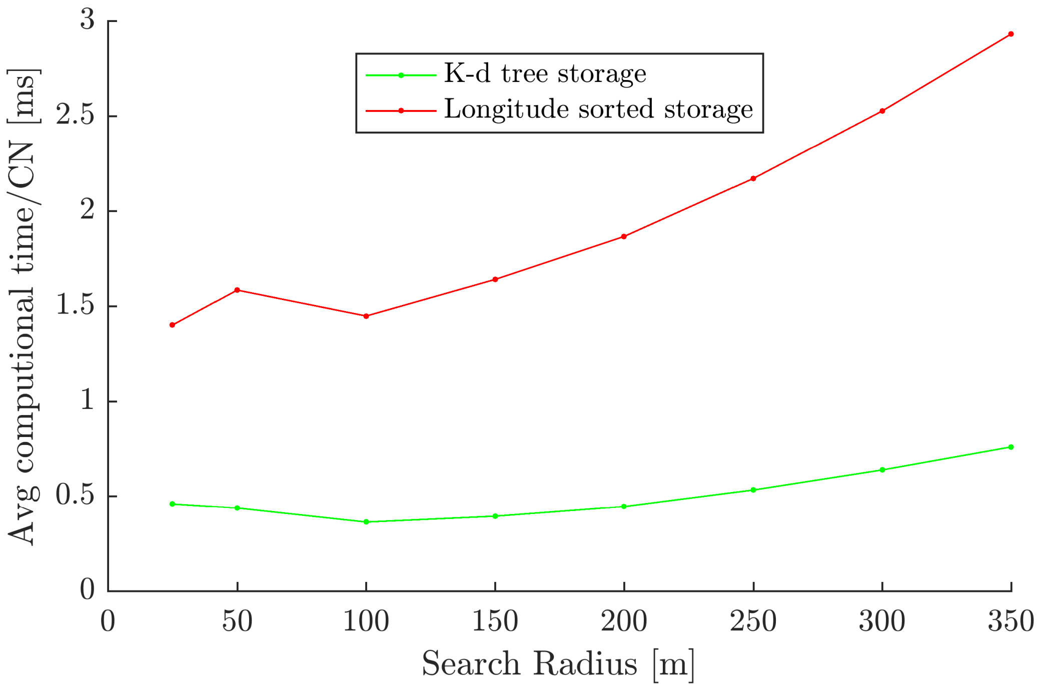
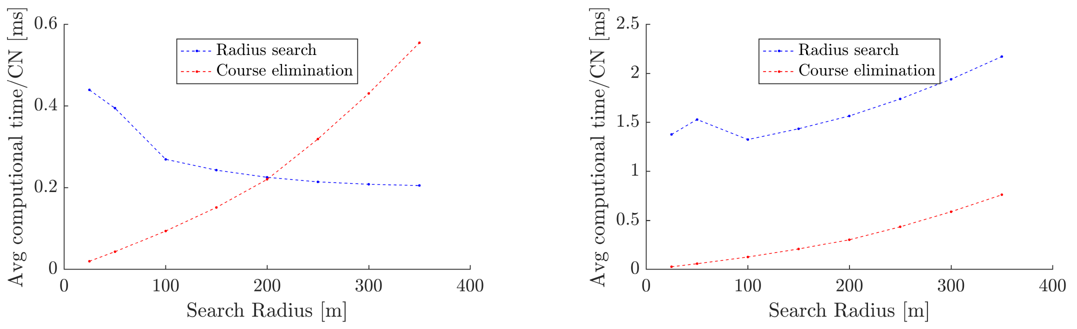


Figure 6.19: Comparison of the total computational time/ CN as function of search radius for the k-d tree storage method and the longitude sorted storage method.



(a) K-d tree storage method. (b) Longitude sorted storage method.

Figure 6.20: The average computational time for the two main parts of the CN search: the radius search and the course elimination. Figure 6.20a show the result for the K-d tree storage method and Figure 6.20b show the results for the longitude sorted storage method.

Figure 6.19 compares the average computational time per CN between the k-d tree storage method and the longitude sorted storage method. Figure 6.20 show the individual time consumptions for the two main parts of the CN search, i.e., the radius search and the course elimination part, for both storage methods. Hence, the sum of the two curves in Figure 6.20a equals the total computational time for the k-d tree storage method in Figure 6.19 (green curve) and similar for the longitude sorted storage method. All computational times are represented in milliseconds (ms).

The average computational times are based on 500 randomly drawn states from the AIS dataset[10] for each of the 8 tested search radii (25 m, 50 m, 100 m, 150 m, 200 m, 250 m, 300 m and 350 m). The two methods perform the CN searches from the same random states.

The computational times for both methods are consistently increasing for $r_c > 100$ m and beyond, as seen from Figure 6.19. For the k-d tree storage method, this increase is caused solely by the linear course elimination part of the CN search, as seen in Figure 6.20a. The same figure reveals that radius searches in k-d trees are more efficient, in terms of lower average computational time per CN, for growing search radius (and hence growing amount of CNs). This may indicate that the k-d tree search burns some processing time initially. The computational time of the radius search for the longitude sorted storage method, on the other hand, is steadily increasing as a function of the search radius. This occur because the method first performs a binary search, followed by a linear search which is a direct function of the search radius, as described in the list above. Note that the course elimination part of the CN search is the same for both methods.

At $r_c = 100$ m, and with $\delta\chi = 35°$, the average computational time for the k-d tree storage method is approximately 0.36 ms/CN while the longitude sorted matrix method has a computational time of 1.45 ms/CN at $r_c = 100$ m, which is a factor of 4 times higher.

6.6.3 Estimated computational time for SPNS and NCDM

The search radius and the maximum course deviation used in the NCDM algorithm were set to $r_c = 100$ m and $\Delta\chi = 35°$ in Table 6.3. With these decision parameters, the average computational time per discovered CN with the k-d tree search is $\bar{t}_{kd} = 0.36$ ms. The maximum[11] total computational time for a predicted trajectory with the NCDM algorithm is then approximately:

$$\begin{aligned} t_{NCDM} &= (1 + J^{max}(K-2))\bar{t}_{kd}\bar{\rho}\pi r_c^2 \\ &\approx J^{max}(K-2)\bar{t}_{kd}\bar{\rho}\pi r_c^2 \qquad if \quad 1 << J^{max}(K-2) \end{aligned} \tag{6.4}$$

where J^{max} is the maximum width of the prediction tree, K is the number of levels in the prediction tree and $\bar{\rho}$ is the average CN density per area. Usually, $1 << J^{max}(K-2)$ so the CN search from the initial state can be neglected. The value of K is a function of the step length Δl, the prediction horizon t_h and the predicted speed $\hat{v}_k^+$ according to (4.11). If a constant speed prediction is used, the value of K can be explicitly given as

$$K = \left\lceil t_h \frac{\hat{v}_k^+}{\Delta l} + 1 \right\rceil \tag{6.5}$$

[10]A small, random offset is added to the states in order to simulate the typical situations where the predicted states do not exist in the dataset. This increases the computational time, but only for the longitude sorted storage method.

[11]Assuming that no predictions are terminated.

where the value is rounded to the nearest integer to make sure that K iterations result in predictions of minimum t_h minutes.

By inserting (6.5) into (6.4) we get:

$$\begin{aligned} t_{NCDM} &= J^{max}(K-1)\bar{t}_{kd}\bar{\rho}\pi r_c^2 \\ &= J^{max}\left\lceil t_h \frac{\hat{v}_k^+}{\Delta l} - 1 \right\rceil \bar{t}_{kd}\bar{\rho}\pi r_c^2 \end{aligned} \tag{6.6}$$

If considering the average speed in the dataset $\bar{v} = 10.5$ kn $= 5.4$ m/s, the average CN density[12] $\bar{\rho} = 0.0048$ CNs/m^2 and with the decision parameters from Table 6.3, that is with $J^{max} = 500$, $t_h = 20$ min $= 1200$ s and $r_c = \Delta l = 100$ m, the computational time of predicting a trajectory with the NCDM algorithm becomes $t_{NCDM} = 28.6$ minutes. This is slightly larger than the observed average prediction time from the 400 test trajectories in Section 6.5 which was 25.1 minutes. This deviation occur because the tree width is not always constantly equal to J^{max} at all levels k due to termination of some predictions.

Similarly, the estimated computational time for the SPNS algorithm can be calculated as

$$\begin{aligned} t_{SPNS} &= (K-1)\bar{t}_{kd}\bar{\rho}\pi r_c^2 \\ &= \left\lceil t_h \frac{\hat{v}_k^+}{\Delta l} - 1 \right\rceil \bar{t}_{kd}\bar{\rho}\pi r_c^2 \end{aligned} \tag{6.7}$$

which with the same parameter values as above gives an estimated computation time per predicted trajectory of $t_{SPNS} = 3.52$ s.

6.6.4 Suggestions to reduce the computational time

The above computational time for the NCDM algorithm is clearly not practically feasible in real-time. Possible ways of reducing the computational time include:

1. Reduce the number of iterations $K - 1$ by:
   (a) Reducing J^{max}. The test value of 500 may be unnecessary high.
   (b) Implementing a step length which adapts to the curvature of the sea lane ahead. Straight line sea lanes can allow larger step lengths than curved sea lanes.
   (c) Limit the prediction horizon t_h. However, it should not be limited to more than approximately 15 minutes in a COLAV system.
2. Reduce the average computational time per discovered CN $\bar{t}_{kd}$ by:
   (a) Optimize the implementation of the linear search which eliminates the neighbors with $\chi_c \notin S$.
   (b) Partitioning the dataset into grid cells to create several, but smaller k-d trees and only search for CNs in the current or neighboring grid cell.
   (c) Storing the data in a more search efficient tree structure, if possible.

[12]The average density is calculated around the predicted trajectories in Section 6.5 with $r_c = 100$m and $\Delta\chi = 35°$.

3. Reduce the search radius after the initial prediction, i.e., for $k \geq 2$, since only one course is drawn anyway. Alternatively implement a kNN search with a low k value to avoid searching for unnecessary many neighbors in high density areas.

4. Only search for CNs once for identical predicted states. It will exist multiple identical states whenever the number of CNs are less than J^{max} initially. However, the likeliness of identical states decreases with predicted time.

Chapter 7

Concluding remarks

Two highly data-driven vessel trajectory prediction methods have been developed and tested. Little research have previously been done in terms of utilizing AIS data to predict vessel trajectories for use in COLAV systems. The proposed methods differ from relevant previous work in two major ways. First, the SPNS and the NCDM algorithms aim for shorter prediction horizons than most of other AIS based methods. The prediction horizons of interest for COLAV applications is up to about 15 minutes, which is beyond the reliable range of the CVM method. Secondly, the proposed methods use the historical AIS data more directly in the predictions, compared to the extensive use of route clustering techniques present in the literature. The proposed algorithms are only intended to proactively assist COLAV systems to avoid possible collision situations. If a potential collision situation is present, the ASV's maneuvers should not rely on the AIS based predictions as these predictions do not represent vessel behavior in such situations.

The SPNS algorithm yields better predictions than the CVM method on curved trajectories, in terms of both lower average and median path and trajectory error. At the same time, it shows good path following capabilities in environments with straight line trajectories. Hence, the SPNS algorithm provides a wider range of environments where predictions are reasonable, as compared with the CVM. Despite that, the SPNS algorithm lack some important features for use in COLAV systems. Each vessel is only assigned a single predicted trajectory, making it impossible to detect multiple branching sea lanes and difficult to provide measures of prediction uncertainty.

The NCDM algorithm is developed to cope with the SPNS' drawbacks. The algorithm shows the capability of dividing its predicted trajectories into multiple branching sea lanes. It is not yet tested to what extent the density of predicted trajectories represents the statistical movement of vessels. It further features prediction uncertainty in terms of the spread of the predicted positions at any given instant of time. The method shows reasonably low 25th and 50th trajectory error percentiles in terms of the distance to the nearest cluster density center and the distance to the nearest clustered position, but relatively much larger 75th percentile and average values as a result of a few, but large errors mostly caused by poor speed predictions.

Wrong predictions of speed are the main contributors to the trajectory errors for both the SPNS and the NCDM algorithm. Improving the speed predictions should therefore be in focus in the further development of the algorithms. Assuming constant speed is overall better than using a pure CN based speed prediction. However, the constant speed method seems only to be the most accurate choice when the vessel is in transit, while a CN based speed prediction approach seems to be the most suitable choice in the minutes after a vessel departures and in the minutes prior to reaching its destination at the shore.

An evaluation of the potential for better speed prediction suggests that at least two factors influence the prediction accuracy of the constant speed method and the CN based speed methods: elapsed prediction time and the number of CNs $C_{k,j}$. As expected, the constant speed prediction accuracy drops as a function of elapsed prediction time, while the CN based methods are independent of the elapsed time. On the other hand, the constant speed approach is independent of $C_{k,j}$ while the CN based methods reveal larger prediction errors for $C_{k,j}$ values less than around 30.

Lastly, the computational time for the NCDM is not practically feasible in real-time with the tested decision parameters and must be reduced significantly.

Chapter 8

Suggestions for further work

The following further work is suggested for the NCDM algorithm:

1. Does the density of predicted trajectories represent the probability of vessels' route choices given the initial state? If not, an alternative algorithm may be tested: instead of randomly pulling courses from the CNs at the initial state, the full trajectories of all CNs may be pulled out initially and predictions can be based on these trajectories. Such a method would have a significantly lower computational time, but it either requires higher resolution AIS data or a good interpolation method in order to obtain predictions at any instant of time.

2. Does the spread of predicted positions at a given predicted time instant reflect the uncertainty of vessel positions?

3. Implement detection of land/ports and test a speed prediction method which applies the initial true speed when the vessel is in transit and a CN based speed during the first minutes after departure and during the last minutes before arrival at land/port. The historical AIS density is very high in areas where vessels are mooring and the speed in these areas are low. Hence, a consistent decrease in speed together with a consistent increase in AIS density among the CNs, followed by zero density (all predictions terminate) is a reasonable indicator of reaching land/port.

4. Develop a measure to indicate how reliable a prediction is. The density of historical AIS data may be one such indicator.

5. It can be interesting to test the NCDM algorithm by randomly pulling speed values from the CNs, in the same manner as already done with the courses, in order to also include uncertainty in the speed prediction.

6. It can also be of interest to test the algorithm on trajectories where absolute all MMSI values of the predicted vessel are removed from the dataset, with the purpose of evaluating how well the algorithm generalizes to new vessels.

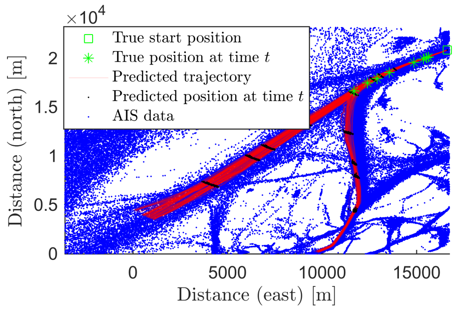
7. Significantly reduce the computational time of the NCDM algorithm, for instance by means of some of the proposed actions at the end of Section 6.6.

8. Implement the NCDM algorithm and a CVM method in a COLAV system and compare the COLAV performance.

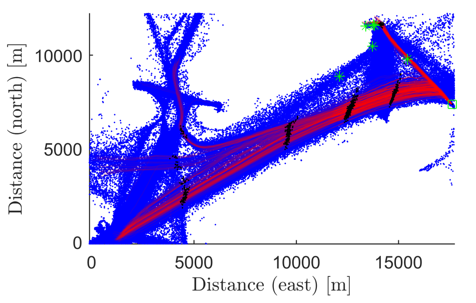
Beside further work on the NCDM algorithm it can be valuable to investigate prediction opportunities based on electronic nautical charts, in order to predict trajectories in situations where AIS data is not available.

Appendix A

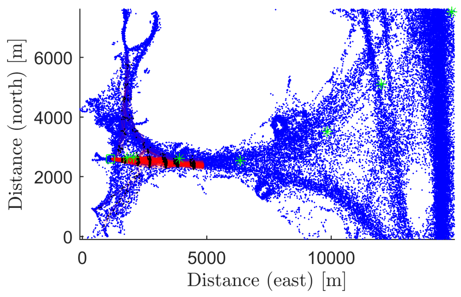
The 20 largest end point deviations from the test in Section 6.5.3



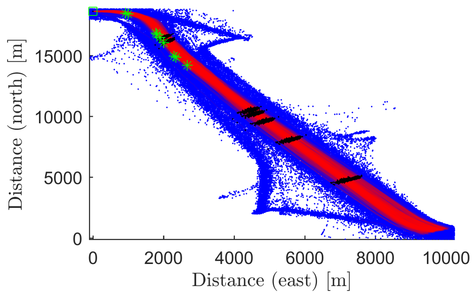
(a)



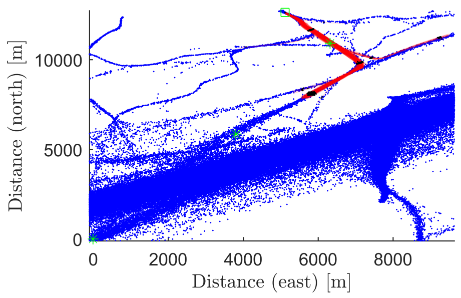
(b)



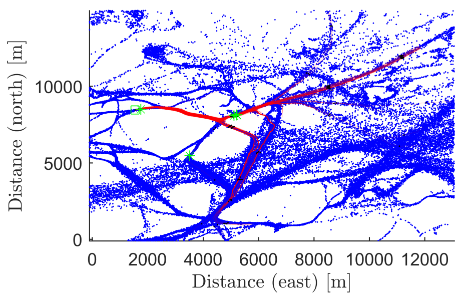
(c)



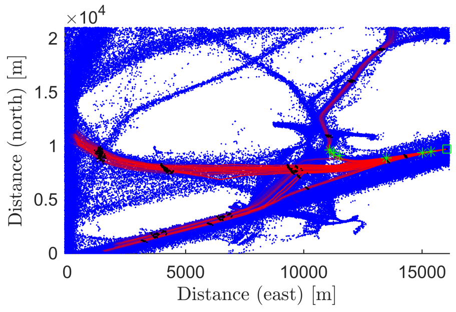
(d)

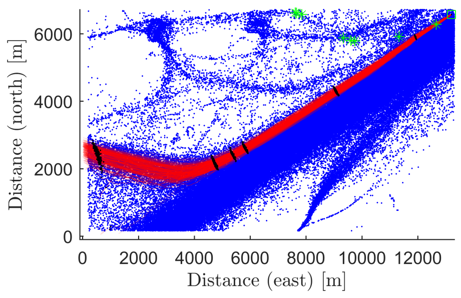


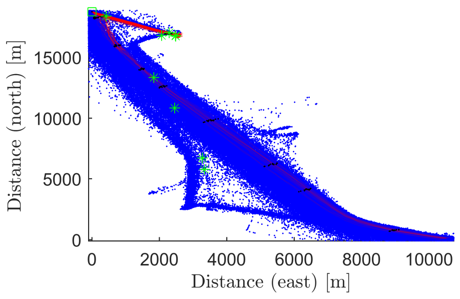
(e)



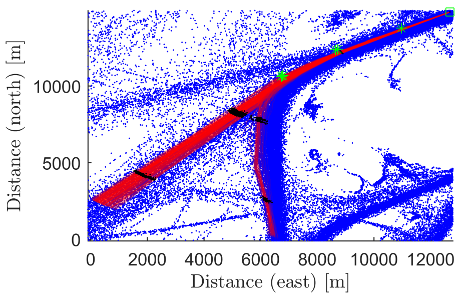
(f)



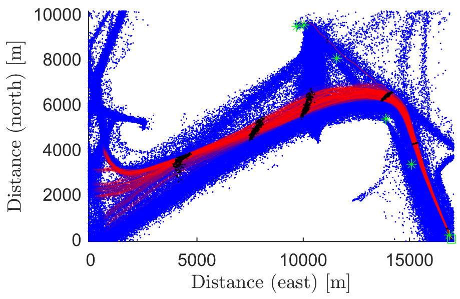



Distance (north) [m]
0
5000
10000
15000
0
2000
4000
6000
8000
10000
Distance (east) [m]

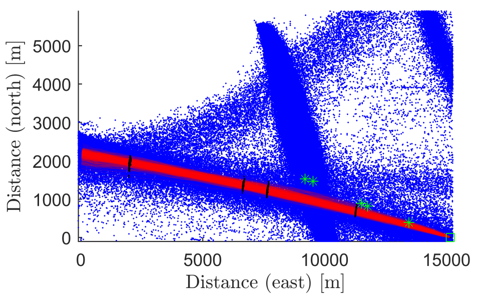
(i)


Distance (north) [m]
0
5000
10000
0
2000
4000
6000
8000
10000
12000
Distance (east) [m]

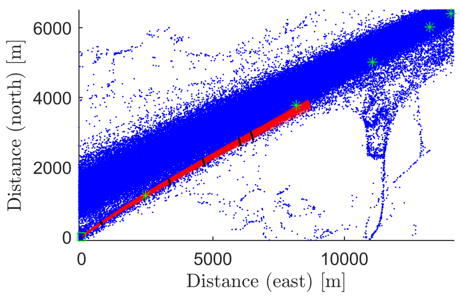
(j)


Distance (north) [m]
0
2000
4000
6000
8000
10000
0
5000
10000
15000
Distance (east) [m]

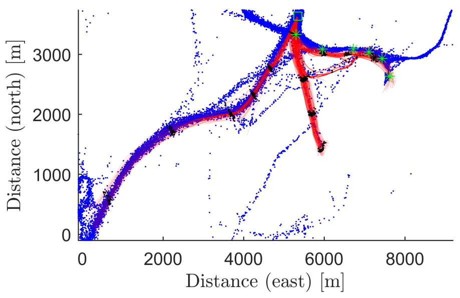
(k)


Distance (north) [m]
0
1000
2000
3000
4000
5000
0
5000
10000
15000
Distance (east) [m]

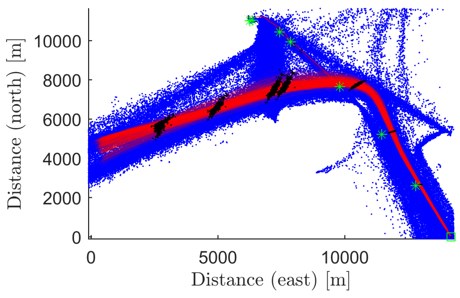
(l)


Distance (north) [m]
0
2000
4000
6000
0
5000
10000
Distance (east) [m]

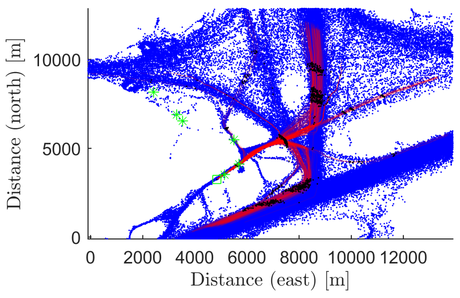
(m)


Distance (north) [m]
0
1000
2000
3000
0
2000
4000
6000
8000
Distance (east) [m]

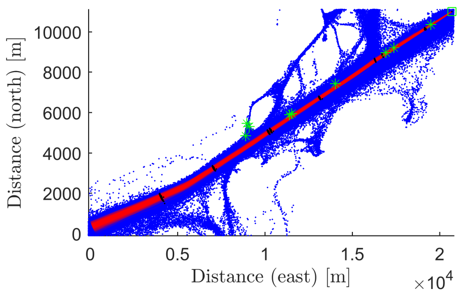
(n)


Distance (north) [m]
0
2000
4000
6000
8000
10000
0
5000
10000
Distance (east) [m]

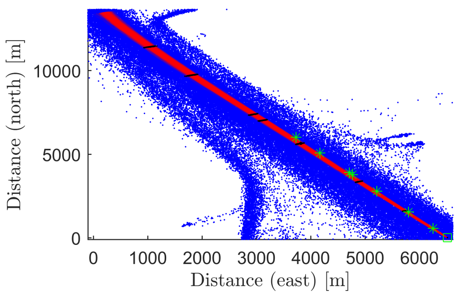
(o)


Distance (north) [m]
0
5000
10000
0
2000
4000
6000
8000
10000
12000
Distance (east) [m]

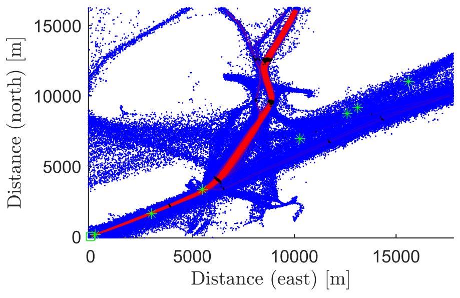
(p)


Distance (north) [m]
0
2000
4000
6000
8000
10000
0
0.5
1
1.5
2
Distance (east) [m]
$\times 10^4$

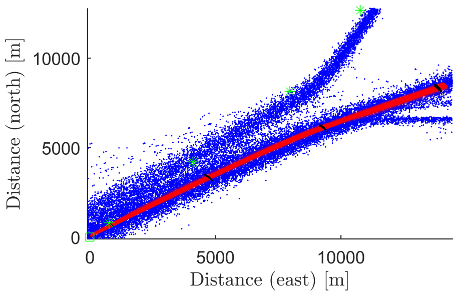
(q)


Distance (north) [m]
0
5000
10000
0
1000
2000
3000
4000
5000
6000
Distance (east) [m]

(r)



(s)

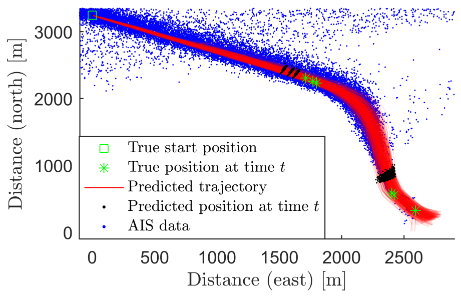


(t)

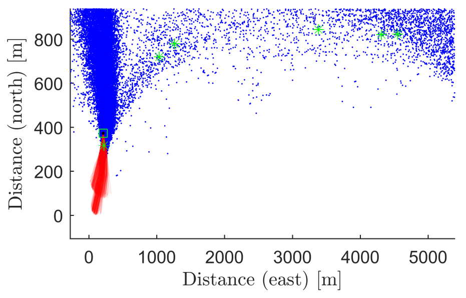
Figure A.1: The 20 largest cluster center devations from the test in Section 6.5.3. The plots are sorted on their end point deviations where Figure A.1a have the largest deviation. Plot markers are only explained in Figure A.1a.

Appendix B

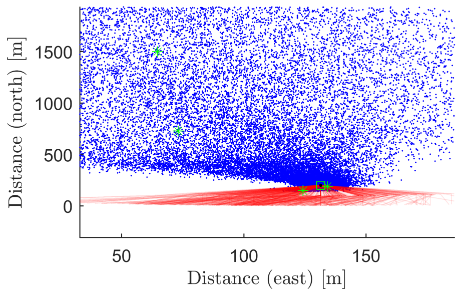
20 of the terminated predictions from the test in Section 6.5.3



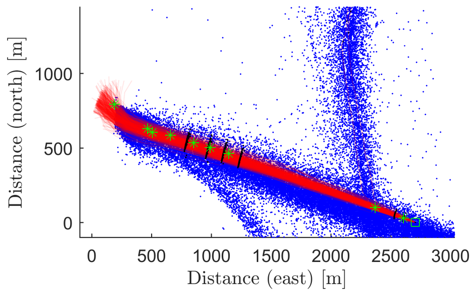
(a)



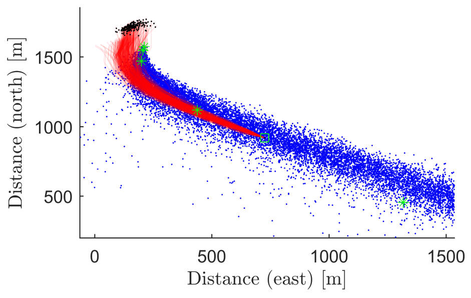
(b)



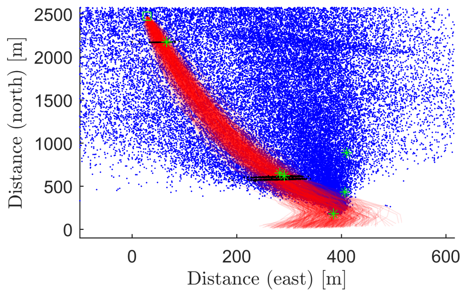
(c)



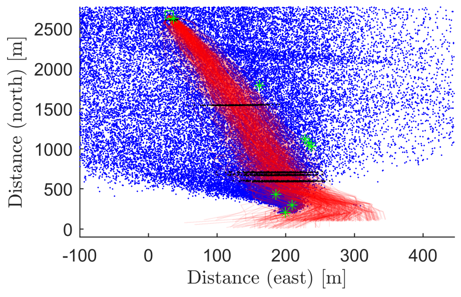
(d)

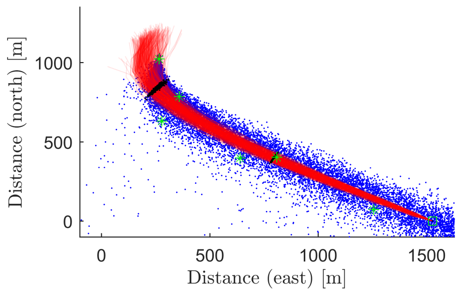


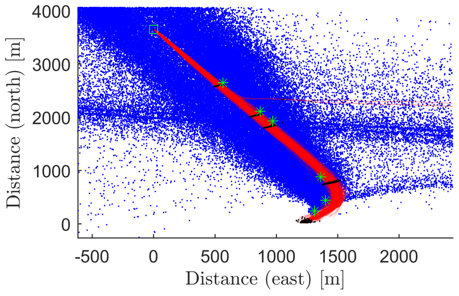
(e)



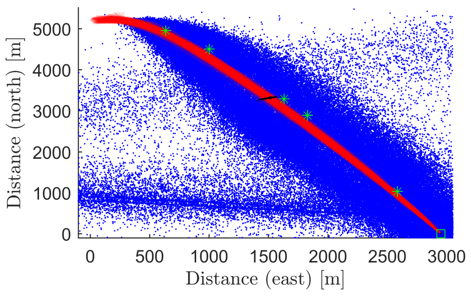
(f)



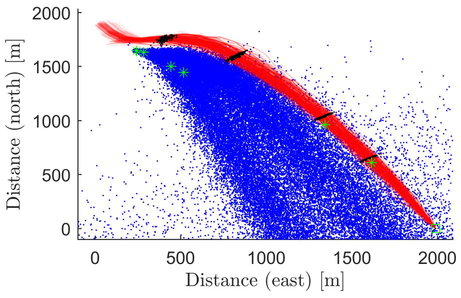



Distance (north) [m]
4000
3000
2000
1000
0
-500
0
500
1000
1500
2000
Distance (east) [m]

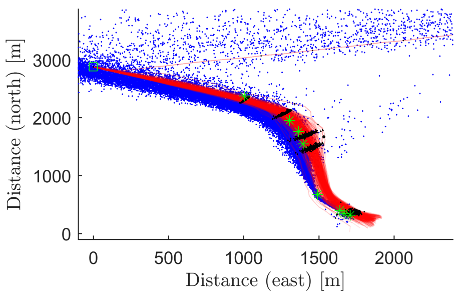
(a)


Distance (north) [m]
5000
4000
3000
2000
1000
0
0
500
1000
1500
2000
2500
3000
Distance (east) [m]

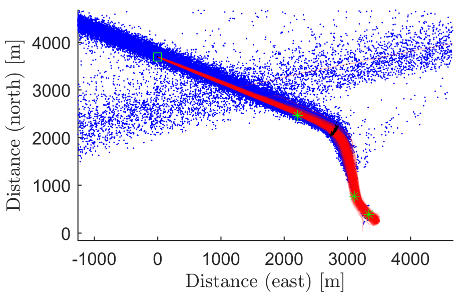
(b)


Distance (north) [m]
2000
1500
1000
500
0
0
500
1000
1500
2000
Distance (east) [m]

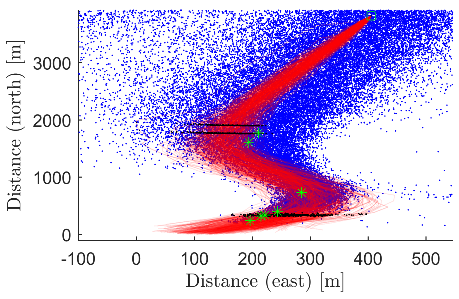
(c)


Distance (north) [m]
3000
2000
1000
0
0
500
1000
1500
2000
Distance (east) [m]

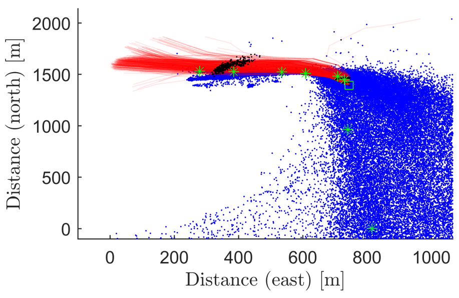
(d)


Distance (north) [m]
4000
3000
2000
1000
0
-1000
0
1000
2000
3000
4000
Distance (east) [m]

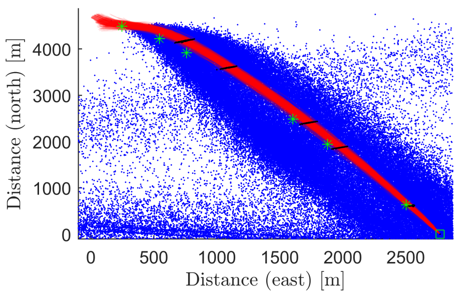
(e)


Distance (north) [m]
3000
2000
1000
0
-100
0
100
200
300
400
500
Distance (east) [m]

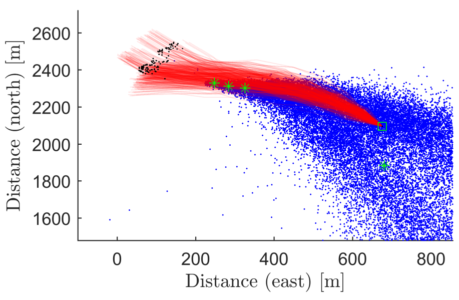
(f)


Distance (north) [m]
2000
1500
1000
500
0
0
200
400
600
800
1000
Distance (east) [m]

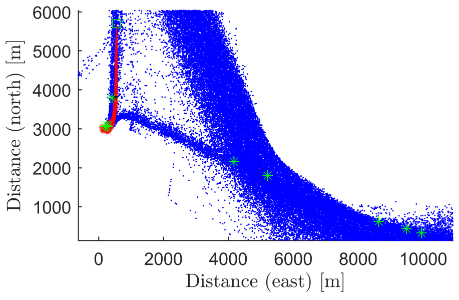
(g)


Distance (north) [m]
4000
3000
2000
1000
0
0
500
1000
1500
2000
2500
Distance (east) [m]

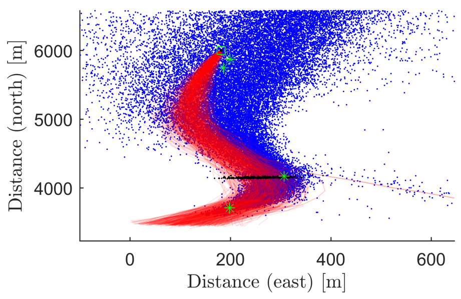
(h)


Distance (north) [m]
2600
2400
2200
2000
1800
1600
0
200
400
600
800
Distance (east) [m]

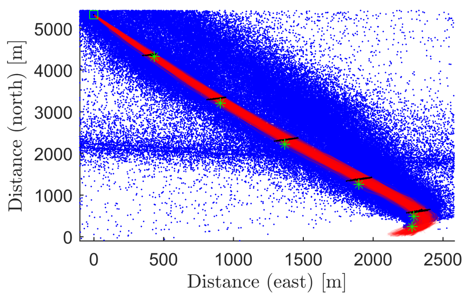
(i)


Distance (north) [m]
6000
5000
4000
3000
2000
1000
0
2000
4000
6000
8000
10000
Distance (east) [m]

(j)



(k)



(l)

Figure B.2: 20 predictions randomly chosen from the terminated predictions in Section 6.5.3. Plot markers are only explained in Figure B.1a.

Appendix C

Accepted paper for the 20th International Conference on Information Fusion 2017, Xi'an China

AIS-based Vessel Trajectory Prediction

Simen Hexeberg*, Andreas L. Flåten†, Bjørn-Olav H. Eriksen‡ and Edmund F. Brekke§
Department of Engineering Cybernetics
Norwegian University of Science and Technology (NTNU)
* simenhex@stud.ntnu.no
† andreas.flaten@ntnu.no
‡ bjorn-olav.h.eriksen@ieee.org
§ edmund.brekke@itk.ntnu.no

***Abstract*—In order for autonomous surface vessels (ASVs) to avoid collisions at sea it is necessary to predict the future trajectories of surrounding vessels. This paper investigate the use of historical automatic identification system (AIS) data to predict such trajectories. The availability of AIS data have steadily increased in the last years as a result of more regulations, together with wider coverage through AIS integration on satellites and more land based receivers. Several AIS-based methods for predicting vessel trajectories already exist. However, these prediction techniques tend to focus on time horizons in the level of hours. The prediction time of our interest typically ranges from a few minutes up to about 15 minutes, depending on the maneuverability of the ASV. This paper presents a novel data-driven approach which recursively use historical AIS data in the neighborhood of a predicted position to predict next position and time. Three course and speed prediction methods are compared for one time step predictions. Lastly, the algorithm is briefly tested for multiple time steps in curved environments and shows good potential.**

I. Introduction

The automotive industry have already come a long way in autonomy, with companies such as Google, Uber and Tesla in the very front of the fast development. This trend is spreading to the maritime industry with the development of autonomous surface vessels (ASVs). Reliable ASVs have potential for large economical and safety related benefits by eliminating errors caused by human operators, increasing cargo space due to elimination of crew facilities and perhaps by more optimal navigation and cooperation between vessels.

Safety and reliability are very important aspects for the public's and governments' acceptance of fully or partly autonomous vessels. A robust collision avoidance (COLAV) system is essential in this context. In order to avoid collision, one must predict the future trajectory of nearby vessels. A commonly used approach today is to assume constant speed and course (e.g. [1]). By utilizing positional, course and speed data from vessel transmitted automatic identification system (AIS) data, it may be possible to obtain more refined predictions. However, since patterns from historical AIS data best reflect normal vessel behavior, as opposed to behavior in close-to-collision situations, AIS based predictions are mainly useful for proactive maneuvers in order to prevent potential collision situations. Faster and more reliable data sources, such as radars, must be used if the situation is already critical.

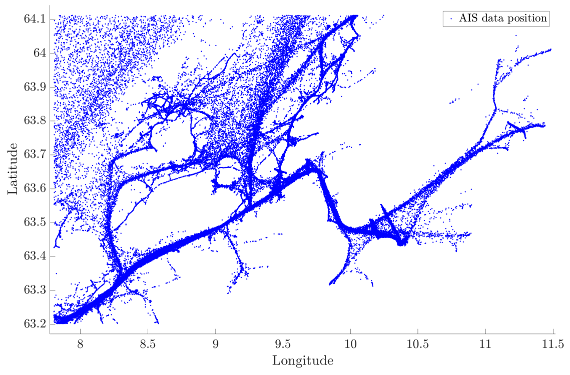


Fig. 1. 3.3% of the available AIS dataset from the Trondheimsfjord in Norway. Beside vessels' positions, which are displayed, the dataset includes information such as course and speed.

Better estimates of surrounding vessels' future trajectories can also be used to improve situational awareness systems.

At sea, the autonomy challenges differ from the ones on the roads. As apposed to cars on the roads, vessels can move in all directions. Additionally, the International Regulations for Preventing Collisions at Sea (COLREGS) is less quantifiable and more dependent on common sense than standard car traffic rules, making it even more difficult to predict future vessel behavior. On the other hand, vessel movement patterns revealed by historical AIS data, as illustrated in Figure 1, are likely to contain implicit information like where it is possible, safe and smart to maneuver, as a majority of vessels navigate in map-advised sea lanes and areas.

The outline of this paper is as follows: Section II covers four possible approaches from previous work. Further, a new method is described in Section III followed by an evaluation of the new method in Section IV. Lastly, conclusions and suggestions for further work are given in Section V and Section VI, respectively.

II. Related work

Little research have been done in terms of utilizing AIS data with the aim of vessel movement predictions for time horizons in the level of a few minutes. However, there exist AIS-based methods aiming for relatively long prediction horizons and some methods developed for animal and weather movement prediction which may facilitate shorter prediction horizons. This section summarizes a few of the most prevailing approaches which may be adapted to COLAV applications.

A. Typical prediction approach

Prediction methods for movement of objects are often based on path clustering and can typically be divided in four main steps in an AIS data context:

1) Cluster paths in the historical data to yield sets of similar route patterns.
2) Classify sequences of new, incoming AIS data points to assign vessels to the route patterns found in step 1.
3) Create representative paths of the common behavior in every route pattern.
4) Predict the vessels' future trajectories, for instance along their representative paths or by using a particle filter.

Various methods are used in all four steps across the literature. The path clustering in step 1 can be further divided into clustering of whole paths and sub-paths. Clustering of whole paths can be useful to predict destination and estimate time of arrival, while clustering of shorter sub-paths has better potential in the context of COLAV systems.

B. A survey on clustering-based methods

The trajectory clustering (TRACLUS)[1] algorithm [2] is a sub-path clustering algorithm tested on hurricane data and animal movement data. The algorithm can be separated in two main parts; path partitioning and sub-path clustering. Paths, which originally consist of straight lines connecting subsequent positional points, are divided into even coarser line segments. The line segments are connected between so called *characteristic points* at which the directional change is to a certain extent. Hence, the length of line segments dynamically adapt with respect to the curvature of the path. The threshold of directional change before a characteristic point occur, is a trade-off between minimizing *preciseness* and *conciseness*. The former is the subsequent sub-paths' deviation from the original path and the latter is the number of partitions, i.e., the number of line segments between characteristic points. An optimal trade-off can be found e.g. by using the minimum description length (MDL) principle [3]. Similar line segments are clustered with an adapted version of the density-based spatial clustering of applications with noise (DBSCAN) algorithm [4]. A distance metric accounts for both the distance and the angular deviation between line segments, facilitating differentiation between perfectly aligned, but opposite moving objects, which is a necessary property in vessel movement prediction. Representative line segments are calculated after clusters are obtained.

[1]The use of *trajectory* can be misleading as temporal information is not considered in the clustering.

Some studies (e.g. [5],[6]) raise concern about TRACLUS' high sensitivity to its two clustering decision parameters which relates to the area density: A neighborhood radius and a lower limit for the number of line segments needed to form a cluster. In [7], the algorithm is improved with respect to this specific issue, yielding an algorithm that outputs much of the same clusters, but for a wider range of decision parameters which reduces the need of domain knowledge when applying the algorithm on a given dataset. Further, the method may experience problems in datasets with circular motion and frequently crossing paths. This is addressed in [8], which adjust the algorithm to better handle such behavior. Another drawback, in vessel prediction context, is that spatial connection between subsequent representative line segments cannot be guaranteed. In other words, a predicted path consisting of subsequent representative line segments is not necessarily continuous.

The traffic route extraction for anomaly detection (TREAD) ([9],[10]) is a route pattern clustering method used in several papers dealing with AIS data. The area of interest is limited by a rectangular bounding box. TREAD clusters neither whole paths nor sub-paths, but certain *waypoints*: (1) *entry points*, (2) *exit points* and (3) *stationary points*. An entry point is marked when a vessel enters the bounding box and an exit point is marked when a vessel leaves the boundary box. The stationary points are detected when a vessel's speed is kept under a certain limit for a certain period of time and are included to account for areas where the vessels are docking or anchoring up. As waypoints are discovered from new incoming AIS data, they are clustered with the DBSCAN algorithm. The clusters are then used to classify routes inside the bounding box as paths are assigned to a start and an end cluster. A start cluster can either be an entry point or a stationary point while an end cluster can either be an exit point or a stationary point. All paths that belong to the same set of start and end clusters are grouped into a unique route pattern. Hence, the algorithm is able to separate between opposite moving vessels. After creating route patterns, representative paths are calculated and an entropy function, measuring the level of disorder in the pattern, serves as a quality measure.

As opposed to the sub-path clustering method in TRACLUS, TREAD's waypoint approach can theoretically allow large deviations in the distances between paths inside the same pattern since the only requirement to belong to a specific pattern is to belong to the same set of start and end clusters. A possible adaptation of the TREAD methodology to COLAV applications is to divide the area of interest into grid cells where each cell play the role as the bounding box. Hence, clustering of waypoints is performed in each cell and paths between these waypoints are then grouped into shorter patterns, better facilitating short time horizon predictions.

Mazzarella et al. propose a knowledge based velocity model (KB-VM) [11] and a knowledge based particle filter (KB-PF) [12] for vessel movement prediction based on AIS data. In both approaches, the TREAD method with a few adjustments is used to cluster AIS generated paths into route patterns. Then, based on position, course and speed, new paths are classified to one of the clustered route patterns using the k-nearest neighbor (kNN) algorithm [13] with the Mahalanobis distance metric. In order to fit the Mahalanobis distance metric to the specific dataset, the metric is *learned* with the supervised large margin nearest neighbor (LMNN) algorithm. If a path is assigned to a route pattern, the KB-PF method predicts future course and speed using a modified particle filter [12] while the KB-VM method predicts course and speed based on the current state's nearest neighbors in the classified pattern, as follows: The course is set to be the median course of all identified neighbors while the speed is set to be the last received speed from the AIS message. Both methods are compared, and the more complex and computational demanding particle filter yields more accurate predictions, but mainly for time horizons exceeding 4-5 hours.

Another approach based on particle filter is suggested in [14]. Implicitly, the particles yield a probability measure of future positions and are distributed over branching sea lanes, an important property for COLAV applications.

III. Single Point Neighbor Search Method

In this paper we propose a single point neighbor search (SPNS) method for AIS-based vessel trajectory prediction. The approach does not rely on path clustering methods, but estimates future course and speed at every prediction time based on historical AIS data.

A. Notation and definitions

Let us define $\boldsymbol{X}$ to be a matrix of historic AIS data according to

$$\boldsymbol{X} = \begin{bmatrix}\boldsymbol{X}_1 & \boldsymbol{X}_2 & \ldots & \boldsymbol{X}_M\end{bmatrix}^T \tag{1}$$

where M is the total number of AIS messages and

$$\boldsymbol{X}_i = \begin{bmatrix}\text{MMSI}_i & t_i & \boldsymbol{p}_i^T & \chi_i & v_i\end{bmatrix}, \tag{2}$$

for $i \in \{1, 2, ..., M\}$ is a vector where MMSI_i, t_i, $\boldsymbol{p}_i$, χ_i and v_i are the Maritime Mobile Service Identity (MMSI) number, timestamp, position vector, course over ground (COG) and speed over ground (SOG), respectively. The position vector can further be written as $\boldsymbol{p}_i = \begin{bmatrix}\lambda_i & \phi_i\end{bmatrix}^T$, where λ_i and ϕ_i is the WGS84 longitude and latitude, respectively. The MMSI is a unique[2] nine digit number that identifies every vessel. The matrix $\boldsymbol{X}$ is sorted such that AIS messages with equal MMSI number are grouped together and messages within every such group are sorted with respect to ascending timestamp.

A predicted trajectory consists of K predicted positions at corresponding instants of time. At every iteration $k \in \{1, ..., K\}$, the predicted state is separated into one *a priori* state, denoted $\hat{\boldsymbol{X}}_i^{k-}$, and one *a posteriori* state, denoted $\hat{\boldsymbol{X}}_i^{k+}$. These two states are written

$$\hat{\boldsymbol{X}}_i^{k-} = \begin{bmatrix}\text{MMSI}_i & \hat{t}^k & \hat{\boldsymbol{p}}^k & \hat{\chi}^{k-} & \hat{v}^{k-}\end{bmatrix} \tag{3}$$

and

$$\hat{\boldsymbol{X}}_i^{k+} = \begin{bmatrix}\text{MMSI}_i & \hat{t}^k & \hat{\boldsymbol{p}}^k & \hat{\chi}^{k+} & \hat{v}^{k+}\end{bmatrix}. \tag{4}$$

Notice that the MMSI, the position vector and the timestamp are the same for the *a priori* and the *a posteriori* state. The only difference is the predicted course and predicted speed. The *a priori* predicted course and speed at iteration k, $\hat{\chi}^{k-}$ and $\hat{v}^{k-}$, represent the predicted course and speed between the previous position $\hat{\boldsymbol{p}}^{k-1}$ and the current position $\hat{\boldsymbol{p}}^k$. Similarly, the *a posteriori* predicted course and speed at iteration k, $\hat{\chi}^{k+}$ and $\hat{v}^{k+}$, represent the predicted course and speed between the current position $\hat{\boldsymbol{p}}^k$ and the next position $\hat{\boldsymbol{p}}^{k+1}$.

All states $\boldsymbol{X}_i$ that are *close* to an *a priori* predicted state $\hat{\boldsymbol{X}}_i^{k-}$ with respect to a distance metric, are defined as close neighbors (CNs) of that predicted state. The set of CNs at a given prediction step k is defined as

$$\boldsymbol{C}^k = \left\{\boldsymbol{X}_i \mid d(\hat{\boldsymbol{p}}^k, \boldsymbol{p}_i) \le r_c, \chi_i \in S, \boldsymbol{X}_i \in \boldsymbol{X}\right\} \tag{5}$$

where

$$d(\hat{\boldsymbol{p}}^k, \boldsymbol{p}_i) = 2R\sin^{-1}\left(\left(\sin^2\left(\frac{\hat{\phi}^k - \phi_i}{2}\right) + \cos(\phi_i)\cos(\hat{\phi}^k)\sin^2\left(\frac{\hat{\lambda}^k - \lambda_i}{2}\right)\right)^{\frac{1}{2}}\right) \tag{6}$$

is defined as the distance between the current predicted position and any position in the AIS dataset given by the Haversine rule[3][15], R is earth's radius, r_c is the search radius and S is an interval of course angles defined by

$$S = \left[\hat{\chi}^{k-} - \Delta\chi, \hat{\chi}^{k-} + \Delta\chi\right], \tag{7}$$

where $\Delta\chi > 0$ is the maximum course angle deviation. Hence, the distance metric restricts all CNs to be within a given radius of the predicted position, as well as having

[2]Although supposedly unique, the same MMSI number may be incorrectly assigned to two or more vessels as mentioned e.g. in [12]. However, this problem is not considered in this paper.

[3]The Haversine formula assumes the earth to be spherical, which differs from the more precise WGS84 model in which the AIS coordinates are represented in. At the latitude of our AIS data, the spherical model introduces an innaccuracy of about 0.5% for distances up to a few hundred meters.

a course within a maximum deviation from the *a priori* predicted course. After the set of CNs is found around $\hat{\boldsymbol{X}}_i^{k-}$, the CNs' mean or median course and speed are used to predict the vessel's course and speed between the current and next position, i.e., they are used to calculate the *a posteriori* state $\hat{\boldsymbol{X}}_i^{k+}$. To simplify the notation in the following, we denote every state that belongs to the set of CNs at prediction step k, i.e., all $\boldsymbol{X}_i \in \boldsymbol{C}^k$, by $\boldsymbol{X}_c^k = \begin{bmatrix}\mathrm{MMSI}_c^k & t_c^k & \boldsymbol{p}_c^k & \chi_c^k & v_c^k\end{bmatrix}$ for $c \in \{1, ..., C\}$, where C is the number of CNs at the given k.

Lastly, a predicted trajectory starting from a given state $\boldsymbol{X}_i$ is defined as:

$$\hat{\boldsymbol{T}}_i = \left\{\begin{bmatrix}\hat{\boldsymbol{p}}^1 & \hat{t}^1\end{bmatrix}, ..., \begin{bmatrix}\hat{\boldsymbol{p}}^K & \hat{t}^K\end{bmatrix}\right\}. \tag{8}$$

Similarly, a vessel's true trajectory, $\boldsymbol{T}_i$, starting from a given state $\boldsymbol{X}_i$, is defined as

$$\boldsymbol{T}_i = \left\{\begin{bmatrix}\boldsymbol{p}^1 & t^1\end{bmatrix}, ..., \begin{bmatrix}\boldsymbol{p}^L & t^L\end{bmatrix}\right\}. \tag{9}$$

where L is the number of measured AIS states in the trajectory. Notice that K and L is not necessarily equal, as several prediction step can be made between two subsequent AIS messages. Also notice that the first element in $\hat{\boldsymbol{T}}_i$ and $\boldsymbol{T}_i$ are equal, as the starting point for both trajectories are given by the same state, $\boldsymbol{X}_i$.

The key notation introduced is summarized in Table I.

TABLE I
LIST OF SYMBOLS

Symbol	Explanation
$\boldsymbol{X}_i$	AIS message / state vector
$\boldsymbol{X}$	Set of all AIS messages $\boldsymbol{X}_i$
$\boldsymbol{C}^k$	Set of CNs at prediction step k
$\boldsymbol{X}_c^k$	States belonging to the set of CNs at prediction step k
$\hat{\boldsymbol{X}}_i^{k-}$	*A priori* predicted state at prediction step k
$\hat{\boldsymbol{X}}_i^{k+}$	*A posteriori* predicted state at prediction step k
$\Delta\chi$	Maximum course angle deviation for CNs
r_c	Search radius [m]
Δl	Step length [m]
K	Number of prediction steps

B. Method

The prediction method is described step by step in Algorithm 1. The steps are described in more detail in the continuation of this section.

1) Decision parameters: The step length Δl decides how far the next position should be propagated, according to

$$\hat{\boldsymbol{p}}^{k+1} = \hat{\boldsymbol{p}}^k + \Delta l \left[\sin(\hat{\chi}^{k+}) f(\hat{\phi}^k), \cos(\hat{\chi}^{k+}) g(\hat{\phi}^k)\right], \tag{10}$$

where $f(\hat{\phi}^k)$ and $g(\hat{\phi}^k)$ are functions of the current latitude $\hat{\phi}^k$, which transform from meters to degrees longitude and degrees latitude, respectively. The step length Δl should reflect the curvature of the sea lanes ahead. Short step lengths facilitate tighter turns and smoother trajectories, but increases the computational requirements. If the step length is set too long, some turns may not be possible to follow and a predicted state might fall into a region with no CNs. Ideally, the algorithm would adapt the step length to the curvature ahead of the vessel, but this is not implemented in this paper.

The search radius r_c restricts CNs to be within a maximum Euclidean distance from the current predicted position. In order for subsequent search areas not to overlap, meaning that the same neighbor is never evaluated more than once, the relationship between the search radius and the step length may be set according to $r_c = \frac{\Delta l}{2}$.

The maximum course deviation parameter $\Delta\chi$ defines the distance metric together with the search radius, as defined in (5). This parameter is necessary in order to exclude neighbor vessels which move in the opposite direction of the vessel we want to predict as well as to avoid influence from crossing sea lanes. The number of prediction steps K must be adapted to the desired prediction horizon.

Algorithm 1 Single Point Neighbor Search prediction

```
1: $\boldsymbol{X}_i$ given                    ▷ The state we want to predict from
2: **Set decision parameters**
   (a) $\Delta l$                              ▷ Step length [m]
   (b) $r_c$                                   ▷ Search radius [m]
   (c) $\Delta\chi$            ▷ Maximum course angle deviation [deg]
   (d) $K$                               ▷ Number of prediction steps
3: Set $\hat{\boldsymbol{X}}_i^{1-} = \boldsymbol{X}_i$
4: **for** $k = 1$ to $K$ **do**
5:     Find all CNs $\boldsymbol{X}_c^k$ around $\hat{\boldsymbol{X}}_i^{k-}$
6:     Calculate $\hat{\boldsymbol{X}}_i^{k+}$ by:
          (a) Calculating $\hat{\chi}^{k+}$ based on $\boldsymbol{X}_c^k$
          (b) Calculating $\hat{v}^{k+}$ based on $\boldsymbol{X}_c^k$
7:     Calculate the next predicted position at its predicted
       point in time:
          (a) Calculate $\hat{\boldsymbol{p}}^{k+1}$ according to (10)
          (b) Calculate $\hat{t}^{k+1} = \hat{t}^k + \frac{\Delta l}{\hat{v}^{k+}}$
8:     Set $\hat{\boldsymbol{X}}_i^{(k+1)-} = \begin{bmatrix}\mathrm{MMSI}_i & \hat{t}^{k+1} & \hat{\boldsymbol{p}}^{k+1} & \hat{\chi}^{k+} & \hat{v}^{k+}\end{bmatrix}$
9: **end for**
```

2) Course prediction: A constant velocity model is used whenever $\boldsymbol{C}^k = \emptyset$. When $\boldsymbol{C}^k \neq \emptyset$, the *a posteriori* course prediction at a given iteration, $\hat{\chi}^{k+}$, is based on the behavior of the CNs, as defined in (5). Two course predictions, based on the mean and median of neighbors, are implemented and compared.

Since the course is periodic in $[0°, 360°]$, caution must be taken when calculating the mean course of all CNs. The mean course can be calculated as [16]:

$$\bar{\chi}_c = \begin{cases} \tan^{-1}\left(\frac{\bar{s}}{\bar{c}}\right) & \text{if} \quad \bar{s} > 0, \bar{c} > 0 \\ \tan^{-1}\left(\frac{\bar{s}}{\bar{c}}\right) + 180° & \text{if} \quad \bar{c} < 0 \\ \tan^{-1}\left(\frac{\bar{s}}{\bar{c}}\right) + 360° & \text{if} \quad \bar{s} < 0, \bar{c} > 0 \end{cases} \tag{11}$$

where

$$\begin{aligned} \bar{s} &= \tfrac{1}{C}\textstyle\sum_{c=1}^{C} \sin\left(\chi_c\right) \\ \bar{c} &= \tfrac{1}{C}\textstyle\sum_{c=1}^{C} \cos\left(\chi_c\right) \end{aligned} \tag{12}$$

and where C is the number of CNs.

The mean value have the drawback of being influenced by outliers. The median course of the CNs, denoted $\tilde{\chi}_c$, can be applied in order to reduce this impact. The same care, regarding the course as a periodic variable, must be considered in the calculation of $\tilde{\chi}_c$.

3) Speed prediction: Instead of assuming constant speed throughout the whole prediction, the mean or median speed of the CNs, denoted $\bar{v}_c$ and $\tilde{v}_c$ respectively, may be used. Although not tested in this paper, it may be a good idea to implement a transition from using constant speed (and perhaps course) at $k = 1$ towards predictions fully based on the CNs as k increases and the validity of the last received AIS message decreases.

IV. Tests and results

The SPNS algorithm is tested with one year of historical AIS data from the Trondheimsfjord in Norway, gathered in 2015. In order to better decide which course and speed prediction methods to use in Algorithm 1, three course and speed prediction methods are statistically tested for a single time step in Section IV-A[4]. Moving on with the most promising of these single step prediction methods, Algorithm 1 is tested on a few scenarios for multiple time steps in situations with curved vessel trajectories in order to get an idea about the method's potential and to reveal shortcomings.

A. Single time step

The decision parameters for the test are listed in Table II. The test is performed on 2500 randomly drawn messages from the dataset.

[4]A variation of these methods have also been tested by replacing the CNs' speed and course values from the AIS messages with their average speed and course between their current and next AIS position. Although those methods yielded better results for speed prediction in a single time step test, their results are not included here as the methods are less suited for multiple time step predictions on curved paths.

TABLE II
DECISION PARAMETERS FOR THE SINGLE TIME STEP TEST OF VARIOUS COURSE AND SPEED PREDICTION METHODS

Decision parameter	**Value**	**Description**
N	2500	Number of course and speed prediction tests
$\Delta\chi$	45°	Maximum course deviation for CNs
r_c	400 m	Search radius for CNs
t_{max}	8 min	Maximum time between two subsequent positions
Δt_i	$t_{i+1} - t_i$	State dependent prediction time step
k	1	Number of prediction steps

The time step, Δt_i, is set equal to the time between the drawn AIS message and the vessel's next received AIS message. Only subsequent messages with time difference less than t_{max} minutes are evaluated.

For the single time step test, predictions are compared to the "true" course and speed which are calculated as the straight-line values between the vessel's state, $\boldsymbol{X}_i$, and its next state, $\boldsymbol{X}_{i+1}$, according to:

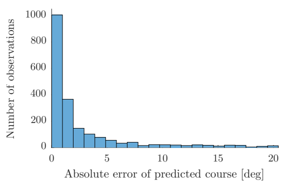
$$\chi_{true} = \operatorname{atan2}((\lambda_{i+1} - \lambda_i)\tfrac{1}{f(\phi_i)}, (\phi_{i+1} - \phi_i)\tfrac{1}{g(\phi_i)}) \tag{13}$$

$$v_{true} = \frac{d(\boldsymbol{p}_{i+1}, \boldsymbol{p}_i)}{t_{i+1} - t_i} \tag{14}$$

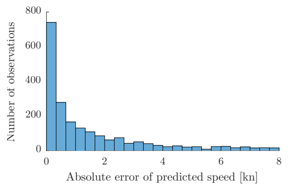
where $\frac{1}{f(\phi_i)}$ and $\frac{1}{g(\phi_i)}$ are functions of the current latitude, ϕ_i, which transform from degrees longitude and degrees latitude into meters and $d(\boldsymbol{p}_i, \boldsymbol{p}_{i+1})$ is given in (6).

The starting position of a prediction equals the true starting position in a single time step prediction. The prediction error in meters is therefore a result of the course error $\hat{\chi}^{1+} - \chi_{true}$ and the difference in length between the true path (a straight line) and the prediction step length. The latter difference is further a function of the speed error $\hat{v}^{1+} - v_{true}$ and the prediction time Δt_i, where Δt_i is equal for the predicted vector and the true vector. Hence, the prediction error in the single step test is a function of the course error and the speed error and these errors are treated separately in order to evaluate their individual impact. Figure 2 shows the absolute error for all predictions with explanations of the corresponding notation in Table III. For visualization purposes, the figure axis are limited to capture 90% of the error mass. Lastly, Table IV shows root mean square error (RMSE) values and three percentiles for the prediction methods. The percentiles are calculated from absolute errors.

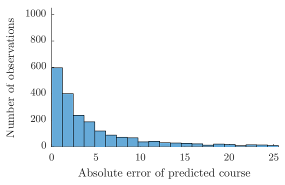
The RMSE and the presented percentiles are all measures of prediction accuracy. As opposed to the lower percentiles, the RMSE is sensitive to large errors. The course RMSE values are particularly large and are caused by a significant



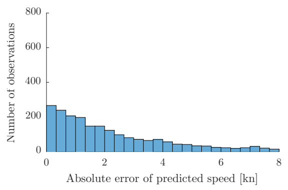
(a) Prediction method: χ_i



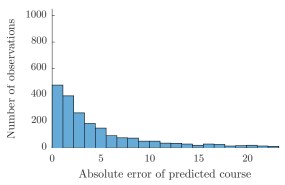
(b) Prediction method: v_i



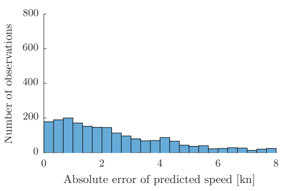
(c) Prediction method: $\tilde{\chi}_c$



(d) Prediction method: $\tilde{v}_c$



(e) Prediction method: $\bar{\chi}_c$



(f) Prediction method: $\bar{v}_c$

Fig. 2. Absolute course and speed prediction errors when applying Algorithm 1 with $k = 1$ on 2500 random states with decision parameters given in Table II. Figure 2a and Figure 2b use the last received course and speed from the AIS message, respectively. Figure 2c and Figure 2d use the median course of the CNs and the median speed of the CNs, respectively. Lastly, Figure 2e and Figure 2f use the mean course of the CNs and the mean speed of the CNs, respectively.

TABLE III
EXPLANATION OF COURSE AND SPEED PREDICTION METHODS

Prediction method	**Explanation**
χ_i	Last received vessel course (COG) from the AIS message
$\tilde{\chi}_c$	Median course of the CNs
$\bar{\chi}_c$	Mean course of the CNs
v_i	Last received vessel speed (SOG) from the AIS message
$\tilde{v}_c$	Median speed of the CNs
$\bar{v}_c$	Mean speed of the CNs

TABLE IV
COMPARISON OF COURSE AND SPEED PREDICTION METHODS FOR THE SINGLE TIME STEP TEST

Course/speed prediction method	**RMSE**	**Percentile (10th)**	**Percentile (50th)**	**Percentile (90th)**
χ_i	33.57°	0.142°	1.217°	20.44°
$\tilde{\chi}_c$	34.19°	0.435°	2.982°	25.52°
$\bar{\chi}_c$	33.83°	0.503°	3.290°	23.19°
v_i	5.801 kn	0.071 kn	0.865 kn	7.786 kn
$\tilde{v}_c$	5.497 kn	0.273 kn	1.842 kn	7.303 kn
$\bar{v}_c$	5.377 kn	0.405 kn	2.199 kn	7.870 kn

amount of situations where a vessel has made a U-turn between $\boldsymbol{X}_i$ and $\boldsymbol{X}_{i+1}$ resulting in a prediction error close to 180°. Further examination of these errors reveal that they are mainly from passenger ferries which travel back and forth the same route. Clearly, predictions made just before a ferry turns will cause large prediction errors and heavily impact the RMSE values. The algorithm can fairly easily be adjusted to cope with these passenger ferry situations, for instance by using the vessel type information provided by the AIS message together with detection of when the vessel is approaching an area where other ferries historically tended to make a U-turn. Based on this, the percentiles that are not affected by the prediction anomalies, such as the 50th percentiles, provide a more representative picture of the prediction accuracy.

Not surprisingly, as seen from Figure 2a and Figure 2b, using the course and speed from the most recent AIS message yield the most accurate course and speed predictions for one time step. However, these predictions assume the vessel to continue with constant course and speed without relying on the CNs historical behavior. The constant velocity model does not utilize the information in the historical AIS data and is therefore not suitable for curved paths. The CNs approaches, on the other hand, can adapt to the surroundings at every prediction step and have the potential to handle curved paths, as will be tested next.

B. Multiple time steps

The SPNS algorithm is tested on 10 manually chosen scenarios to highlight the algorithm's potential and to reveal shortcomings. The decision parameters[5] are given in Table V and the predictions are illustrated in Figure 3[6].

TABLE V
CURVED TRAJECTORY TEST: DECISION PARAMETERS

Decision parameter	**Value**	**Explanation**
r_c	50 m	Search radius for CNs
Δl	$2r_c$	Prediction step length
$\Delta\chi$	25°	Maximum course deviation for CNs
$\hat{\chi}_i^{k+}$	$\tilde{\chi}_c$	Course prediction method used at every iteration k
$\hat{v}_i^{k+}$	$\tilde{v}_c$	Speed prediction method used at every iteration k

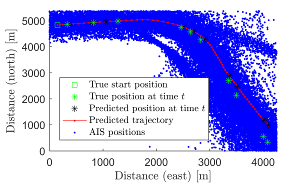
[5]Due to the tight turns in Figure 3c, the plot is generated with half the step length (and hence half search radius) and a larger accepted maximum course deviation for neighbors, meaning $r_c = \frac{\Delta l}{2} = 25\text{m}$ and $\Delta\chi = 35°$.

[6]Be aware that the ratio between the x-axis and the y-axis are not 1:1 and that it vary among the subfigures.

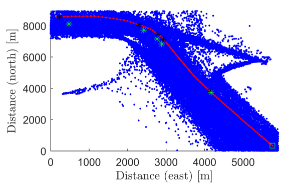
Figure 3a - Figure 3e all yield relatively accurate trajectory predictions (in both course and speed). Figure 3f shows accurate speed prediction, but the predicted path deviates from the true path inside the wide sea lane, illustrating that although the algorithm can potentially follow sea lanes, it can not guarantee to take the correct path within it.

Figure 3g illustrates a situation where the turn was too tight for the given decision parameters. However, by reducing the step length from 50 m to 25 m, the algorithm is able to follow the turn, as shown in Figure 3h. The relationship between $\Delta\chi$ and Δl is important. For a given step length, a too low value of $\Delta\chi$ will limit the radius of curvature that the algorithm can follow. On the other hand, a too large value may for instance cause a prediction to suddenly be pulled into a crossing sea lane. However, by lowering the step length, the maximum course deviation can be lowered without compromising the range of radii of curvatures to follow, at the same time as crossing sea lanes are less likely to be an issue.

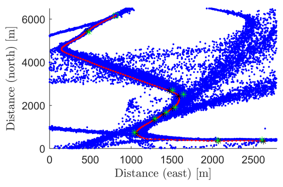
Figure 3i and Figure 3j demonstrate that the algorithm does not handle branching of sea lanes as it typically follows the most dense path ahead.



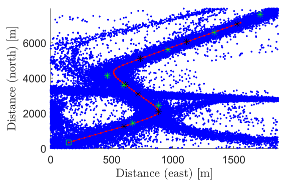
(a) Time horizon: 18.87 min



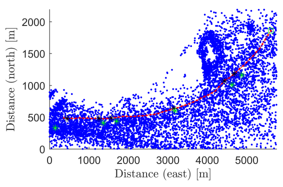
(b) Time horizon: 30.38 min



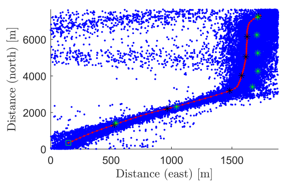
(c) Time horizon: 26.35 min



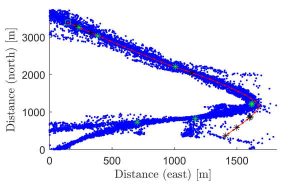
(d) Time horizon: 21.82 min



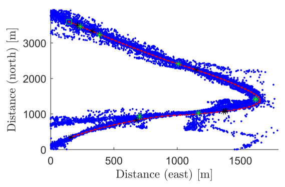
(e) Time horizon: 23.22 min



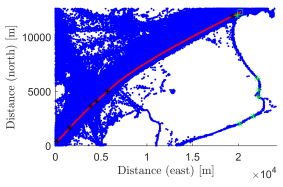
(f) Time horizon: 21.32 min



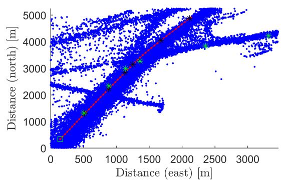
(g) Time horizon: 18.33 min



(h) Time horizon: 18.33 min



(i) Time horizon: 49.17 min



(j) Time horizon: 15.32 min

Fig. 3. The SPNS algorithm tested on various curved trajectories. Explanation of the sub figures' markers are shown in Figure 3a.

V. Conclusion

As opposed to the large amount of AIS-based prediction techniques that rely on clustering methods, the SPNS algorithm bases the prediction directly on historical AIS data. The algorithm shows good potential for vessel trajectory prediction for medium time horizons ranging up to about 30 minutes. Further, it is able to follow paths of various curvatures. However, the algorithm is sensitive to the choice of certain decision parameters. Also, the algorithm can not handle branching of sea lanes and it does not yield any uncertainty measure of the predictions. Despite these shortcomings, the algorithm may still be well suited to proactively assist collision avoidance systems.

VI. Suggestions for future work

- Automatically adapt the decision parameters Δl, r_c and $\Delta\chi$ to the curvature ahead of the vessel to avoid unnecessary small step lengths and to better fit the parameters to a given density and radius of curvature.
- Develop a measure of prediction uncertainty.
- Detect branching sea lanes. For instance by letting $\Delta\chi = 90°$ while searching for more than one peak in a course histogram of all CNs. Every peak will indicate a certain course where a significant amount of vessels tend to travel.
- Develop probability estimates for predicted trajectories, allowing several possible future trajectories for a single vessel, e.g. by sampling from the CNs instead of using the median or mean values.
- Use AIS data with higher time resolution to better validate a predicted trajectory against the true trajectory.

Acknowledgment

This work was supported by the Research Council of Norway through projects 223254 (Centre for Autonomous Marine Operations and Systems at NTNU) and 244116/O70 (Sensor Fusion and Collision Avoidance for Autonomous Marine Vehicles). The authors would like to express great gratitude to DNV GL, especially Øystein Engelhardtsen and Hans Anton Tvete, for contributing to initial ideas and providing real-world AIS data.

References

[1] Yoshiaki Kuwata et al. “Safe Maritime Autonomous Navigation With COLREGS, Using Velocity Obstacles”. In: *IEEE Journal of Oceanic Engineering* vol. 39, no. 1 (Jan. 2014), pp. 110–119.

[2] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. “Trajectory clustering”. In: *Proceedings of the 2007 ACM SIGMOD international conference on Management of data - SIGMOD ’07*. Association for Computing Machinery (ACM), 2007.

[3] Peter D. Grunwald, In Jae Myung, and Mark A. Pitt. *Advances in Minimum Description Length: Theory and Applications*. MIT Press Ltd, 2005.

[4] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: Association for the Advancement of Artificial Intelligence (AAAI) Press, 1996, pp. 226–231.

[5] Mahdi M. Kalayeh et al. “Understanding Trajectory Behavior: A Motion Pattern Approach”. In: 2015.

[6] Yu Zhang and Dechang Pi. “A Trajectory Clustering Algorithm Based on Symmetric Neighborhood”. In: *2009 WRI World Congress on Computer Science and Information Engineering*. IEEE, 2009.

[7] Chen Jiashun. “A new trajectory clustering algorithm based on TRACLUS”. In: *Proceedings of 2012 2nd International Conference on Computer Science and Network Technology*. IEEE, Dec. 2012.

[8] Shrikant Kashyap et al. “FARM : Feature-Assisted Aggregate Route Mining in Trajectory Data”. In: *2009 IEEE International Conference on Data Mining Workshops*. IEEE, Dec. 2009.

[9] Giuliana Pallotta, Michele Vespe, and Karna Bryan. “Vessel Pattern Knowledge Discovery from AIS Data: A Framework for Anomaly Detection and Route Prediction”. In: *Entropy* vol. 15, no. 6 (June 2013), pp. 2218–2245.

[10] Giuliana Pallotta, Michele Vespe, and Karna Bryan. “Traffic Knowledge Discovery from AIS Data”. In: *The 16th international conference on information fusion*. 2013.

[11] Fabio Mazzarella, Michele Vespe, and Carlos Santamaria. “SAR Ship Detection and Self-Reporting Data Fusion Based on Traffic Knowledge”. In: *IEEE Geoscience and Remote Sensing Letters* vol. 12, no. 8 (Aug. 2015), pp. 1685–1689.

[12] Fabio Mazzarella, Virginia Fernandez Arguedas, and Michele Vespe. “Knowledge-based vessel position prediction using historical AIS data”. In: *2015 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, Oct. 2015.

[13] N. S. Altman. “An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression”. In: *The American Statistician* vol. 46, no. 3 (Aug. 1992), pp. 175–185.

[14] Branko Ristic et al. “Statistical Analysis of Motion Patterns in AIS Data: Anomaly Detection and Motion Prediction”. In: *11th International Conference on Information Fusion*. 2008.

[15] *Haversine formula*. URL: https://en.wikipedia.org/wiki/Haversine_formula (visited on 02/21/2017).

[16] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York Inc., 2006, pp. 105–109.

Bibliography

[1] Simen Hexeberg, Andreas L. Flåten, Bjørn-Olav H. Eriksen, and Edmund F. Brekke. “AIS-based Vessel Movement Prediction for ASV Collision Avoidance”. In: *5th year specialization project at Engineering Cybernetics, NTNU*. 2016.

[2] Yoshiaki Kuwata, Michael T. Wolf, Dimitri Zarzhitsky, and Terrance L. Huntsberger. “Safe Maritime Autonomous Navigation With COLREGS, Using Velocity Obstacles”. In: *IEEE Journal of Oceanic Engineering* vol. 39, no. 1 (Jan. 2014), pp. 110–119.

[3] *Automatic identification system.* `https://en.wikipedia.org/wiki/Automatic_identification_system`. Accessed: 2016-09-03.

[4] Fabio Mazzarella, Virginia Fernandez Arguedas, and Michele Vespe. “Knowledge-based vessel position prediction using historical AIS data”. In: *2015 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, Oct. 2015.

[5] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. “Trajectory clustering”. In: *Proceedings of the 2007 ACM SIGMOD international conference on Management of data - SIGMOD ’07*. Association for Computing Machinery (ACM), 2007.

[6] Peter D. Grunwald, In Jae Myung, and Mark A. Pitt. *Advances in Minimum Description Length: Theory and Applications*. MIT Press Ltd, 2005.

[7] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: Association for the Advancement of Artificial Intelligence (AAAI) Press, 1996, pp. 226–231.

[8] Mahdi M. Kalayeh, Stephen Mussmann, Alla Petrakova, Niels da Vitoria Lobo, and Mubarak Shah. “Understanding Trajectory Behavior: A Motion Pattern Approach”. In: 2015.

[9] Yu Zhang and Dechang Pi. “A Trajectory Clustering Algorithm Based on Symmetric Neighborhood”. In: *2009 WRI World Congress on Computer Science and Information Engineering*. IEEE, 2009.

[10] Chen Jiashun. “A new trajectory clustering algorithm based on TRACLUS”. In: *Proceedings of 2012 2nd International Conference on Computer Science and Network Technology*. IEEE, Dec. 2012.

[11] Shrikant Kashyap, Sujoy Roy, Mong Li lee, and Wynne Hsu. “FARM : Feature-Assisted Aggregate Route Mining in Trajectory Data”. In: *2009 IEEE International Conference on Data Mining Workshops*. IEEE, Dec. 2009.

[12] Giuliana Pallotta, Michele Vespe, and Karna Bryan. “Vessel Pattern Knowledge Discovery from AIS Data: A Framework for Anomaly Detection and Route Prediction”. In: *Entropy* vol. 15, no. 6 (June 2013), pp. 2218–2245.

[13] Giuliana Pallotta, Michele Vespe, and Karna Bryan. “Traffic Knowledge Discovery from AIS Data”. In: *The 16th international conference on information fusion.* 2013.

[14] Fabio Mazzarella, Michele Vespe, and Carlos Santamaria. “SAR Ship Detection and Self-Reporting Data Fusion Based on Traffic Knowledge”. In: *IEEE Geoscience and Remote Sensing Letters* vol. 12, no. 8 (Aug. 2015), pp. 1685–1689.

[15] N. S. Altman. “An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression”. In: *The American Statistician* vol. 46, no. 3 (Aug. 1992), pp. 175–185.

[16] Branko Ristic, B. La Scala, M.R Moreland, and N. Gordon. “Statistical Analysis of Motion Patterns in AIS Data: Anomaly Detection and Motion Prediction”. In: *11th International Conference on Information Fusion.* 2008.

[17] Simen Hexeberg, Andreas L. Flåten, Bjørn-Olav H. Eriksen, and Edmund F. Brekke. “AIS-based Vessel Trajectory Prediction”. In: *Proceedings of the 20th International Conference on Information Fusion.* IEEE, 2017.

[18] *Haversine formula.* URL: `https://en.wikipedia.org/wiki/Haversine_formula` (visited on 02/21/2017).

[19] *Percentile.* `https://en.wikipedia.org/wiki/Percentile`. Accessed: 2017-05-29.

[20] *DBSCAN.* `https://en.wikipedia.org/wiki/DBSCAN`. Accessed: 2016-12-11.

[21] *k-d tree.* `https://en.wikipedia.org/wiki/K-d_tree`. Accessed: 2017-06-03.

[22] D.T. Lee and C.K. Wong. “Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees”. In: *Acta Informatica* 9.1 (1977).