VOL.39 Nº1 2022
https://doi.org/10.22491/IMECTA.11033.en

Creation of autonomous vessel trajectories in a virtual simulation environment based on historical AIS data

Antônio L. C. Doneda[a], Ricardo S. Bastos[a], Julio Cesar Duarte[b]
[a]Centro de Análise de Sistemas Navais
Ed. 23 do AMRJ - R. da Ponte, s/n, 20.091-000, Centro,
Rio de Janeiro - RJ – Brazil
[b]Instituto Militar de Engenharia, Seção de Engenharia da Computação – SE/9
Praça General Tibúrcio, 80, 22290-270, Praia Vermelha,
Rio de Janeiro, RJ, Brazil

ABSTRACT: Virtual simulations have grown in importance for manpower training but it is important to portray reality with a satisfactory degree of fidelity. In this study, we applied a modified DBSCAN clusterization method, along with data pre- and post-processing, to obtain, from AIS messages, routes representing the common behavior of vessels in a port area and use them to control non-playable characters in navigation simulation training for the Brazilian Navy. The algorithm was effective in generating 19 different routes depicting five selected vessel types from 76.179 AIS messages collected for 48 hours.

KEYWORDS: Virtual Simulations. Clusterization. DBSCAN. AIS. Routes.

RESUMO: Simulações virtuais têm crescido muito em importância para treinamento de recursos humanos, mas é importante que representem a realidade com um grau satisfatório de fidelidade. Neste trabalho, aplicamos o método de clusterização DBSCAN modificado, em conjunto com pré e pós-processamentos de dados, para obter derrotas representativas do comportamento comum de navios em uma zona portuária, a partir de mensagens AIS, para utilizá-las no controle de NPC em uma simulação de treinamento de navegação da Marinha do Brasil. O algoritmo mostrou-se efetivo, gerando 19 diferentes trajetórias representativas de 5 tipos de navios selecionados, a partir de 76.179 mensagens AIS coletadas durante 48 horas.

PALAVRAS-CHAVE: Simulações Virtuais. Clusterização. DBSCAN. AIS. Trajetórias.

1. Introduction

Simulators enable the Brazilian Navy to recreate vessel navigation, maneuver conditions, and maritime traffic interactions, reducing operational costs. It also preserves Navy assets and has a wide range of trainable procedures, benefiting the readiness of the Naval Force by greatly decreasing costs, risks, and equipment wear.

The Brazilian Navy Center for Analysis of Naval Systems (CASNAV) has developed a Bridge Simulator to meet the navigation training demands of its personnel [1]. In this simulator, the maritime traffic interacting with users is controlled by an instructor who must define trajectories for autonomous vessels (NPC - non-playable characters). Each NPC can be associated with a predetermined trajectory. This solution is interesting only for a specific simulation. For vessels entering and leaving ports, the simulated maritime traffic should ideally be as similar as possible to reality.

Thus, some vessels (freighters, tugboats, offshores, barges, etc.) have certain trajectory patterns. Manual trajectory creation for exercises requires instructors to know the common waterways of a port area (PA) and what vessel category they use. If instructors lack this knowledge, they may create exercises which disagree with reality.

A tool able to assist in the generation of specific vessel trajectories enables instructors, via basic commands, to populate simulations with NPCs portraying daily found trajectories as reliably as possible. Thus, instructors can focus on users' procedures without worrying about building a trajectory for each NPC.

This study aims to propose the use of an unsupervised machine learning technique, DBSCANSD (Density

Based Spatial Clustering Application with Noise by Varying Densities) [2], to generate trajectories representing common vessel movement based on the navigation history of a given port. These trajectories would be stored in the simulation system and instructors would only choose vessel types, entrusting the system to determine their trajectories.

Thus, it would be possible to simulate maritime traffic in any port in Brazil or abroad just by collecting historical navigation data, honing users' training. Moreover, it is of paramount importance that the generated trajectories have the characteristics of the observed trajectories to produce greater realism.

Section 2 describes related studies on trajectory prediction. Section 3 shows how we obtained and treated historical vessel movement data in Guanabara Bay. Section 4 describes DBSCANSD use for predicting trajectories. Section 5 provides the results obtained by our proposal and, finally, section 6 offers our final considerations and conclusions.

2. Related studies

Several studies have aimed to predict vessel trajectory for various purposes based on past trajectories obtained by Automatic Identification System (AIS) equipment providing, among other information, the latitude and longitude of a vessel. Historical data can be divided into groups (clusters) to extract and classify trajectories into a model which can represent them.

Perera et al. [3] implemented an algorithm with artificial neural networks and the extended Kalman filter to predict trajectories, mainly focusing on vessel detection and monitoring.

Vries and Someren [4] defined a kernel-based machine learning method to group and classify maritime traffic and detect behavior anomalies. Initially, AIS data is compressed into linear trajectories via geometric operations. Next, similarities are compared by grouping them with kernel k-means and classifying them by support vector machines.

Duca et al. [5] used an algorithm based on a K-Nearest Neighbor classifier to predict vessel positions after 30, 45, and 60 minutes. This algorithm receives current data as input and returns an array of probabilities for a future position in a pre-established grid. This study reached a 79.4% precision, 78.5% coverage, and 93.1% accuracy.

Pallotta et al. [6] used the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm, first proposed by Ester et al. [7], to group position points based on AIS data attributes. A circular area is projected over a vessel current state. All routes passing through this area are considered compatible with the vessel. Once positions are grouped together, trajectory predictions can be made based on this history.

Liu et al. [2] applied the DBSCAN approach with non-spatial attributes such as vessel speed and direction to compare the most used trajectories with those predicted by the rules and regulations of each port.

These studies show an evident concern with predicting trajectories to monitor vessels, detect anomalies, and search and rescue.

Unlike [2], this study uses data pre- and post-processing, based on PA characteristics and vessel types making up maritime traffic, to obtain realistic trajectories to be inserted in a simulated training environment.

3. DATABASE USED

The data used by our algorithm is the dynamic and static information vessels emitted via an AIS equipment; in this case, a transponder transmitting information such as latitude, longitude, speed, direction, record, name, and data-time of information collection in VHF frequency.

This study approached the Guanabara Bay, in the state of Rio de Janeiro, housing the Rio de Janeiro port, oil and gas terminals, and passenger ferry stations composing the Rio de Janeiro-Niterói route. To acquire information, an area of interest was limited by the coordinates 22°45.0'S and 043°14.0'W, 22°45.0'S and 043°05.0'W, 23°0.0'S and 043°14.0'W, and 23°0.0'S and 043°05.0'W, equivalent to Directorate of Hydrography and Navigation nautical chart 1501.

VOL.39 Nº1 2022
https://doi.org/10.22491/IMECTA.11033.en

Then, we obtained the AIS data of all vessels for August 14 to 16, 2019, totaling 111,015 AIS messages. Positions in which vessels moved at a speed below 0.5 knots were removed so only vessels in real motion were used, maintaining, after removal, a total of 76,179 messages.

Data were original NMEA-183 message processes, from which we extracted the following information of interest: vessel identification (MMSI), latitude, longitude, speed, direction (destination), and date-time.

Figure 1 shows the Guanabara Bay data plotted in nautical chart 1501.

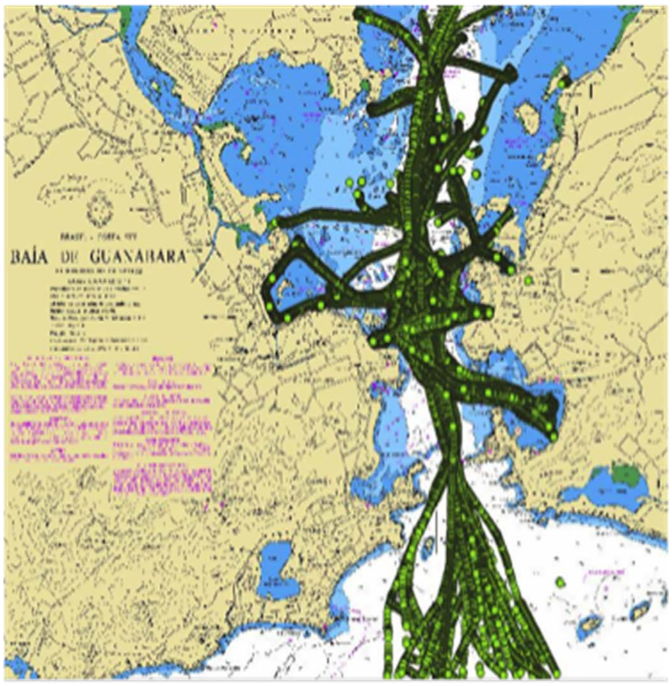

Fig. 1 - Positions of all vessels.

4. DBSCANSD trajectory prediction

Short for 'Density Based Spatial Clustering of Application with Noise,' DBSCAN consists of nonparametric density-based clustering [7]. This method is effective for detecting arbitrary clusters of different sizes, finding and separating data noise, and detecting "natural" clusters and their arrangement within the data space without any preliminary group information.

Clusters and the DBSCAN algorithm apply to two- and three-dimension Euclidean spaces and any characteristically high-dimensional space [7]. The idea of the DBSCAN method is that every point in a cluster has at least a certain number of points in its neighborhood of a given radius.

4.1 Definitions

ε-neighborhood of a point *p* (Nε (*p*)): The vicinity of an object *p* with radius ε is given by Nε (*p*) = {*q* in **D** | dist (*p*, *q*) < ε}. In **Figure 2**, circles represent the ε-neighborhood of point *q* and the ε-neighborhood of point *p* [8], respectively.

Central Point: If the ε-neighborhood of an object *p* contains at least a minimum number of objects (*MinPts*), then object *p* is called the central point.

Edge points: If the ε–neighborhood of an object *p* is less than MinPts but contains some center point, then object *p* is an edge point.

Direct density reach: An object *p* is direct density reachable by object *q*, with respect to ε and *MinPts*, if *p* is in the ε-neighborhood of *q* and *q* is a central point.

Density connection: An object *p* is density connected to object *q*, with respect to ε and *MinPts* in an object set **D**, if **D** contains an object *o* such that both *p* and *q* are direct density reachable with respect to ε and *MinPts*.

DBSCAN Cluster: Is the set of density connected points.

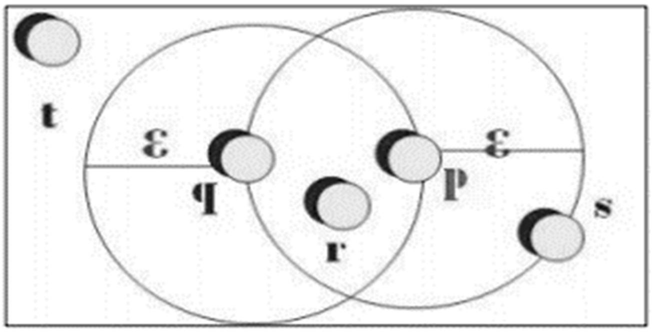

Fig. 2 - ε-neighborhood of a point p. Source: [8].

4.2 Method

According to [8], DBSCAN finds clusters by checking the neighborhood of each point in the database, starting with an arbitrary object *p*. If *p* is a central point, a new cluster with *p* as a center is created. If *p* is an edge point, no point is density reachable from *p* and the algorithm visits the next point on the database. DBSCAN then iteratively collects direct density reachable objects from center points, which may involve joining some density reachable clusters. The process ends when no new points can be added to any cluster.

In the DBSCAN algorithm, any two central points with a distance less than or equal to ε are placed in the same cluster. Any edge point near a central point is placed in the same cluster as the central point. Points not directly attainable by some central point are classified as noise.

In addition to the basic DBSCAN idea, Liu et al. [2] adopted two factors: maximum speed variation (*MaxSpd*) and maximum direction variation (*MaxDir*). Their aim was to cover not only proximal neighbors but also direction- (Course Over Ground, COG) and speed-similar ones (Speed Over Ground, SOG), thus creating DBSCAND.

Then, they modified the ε-neighborhood definition to: The vicinity of an object *p* with radius ε is given by $N\varepsilon(p) = \{q \in \mathbf{D} \mid \text{dist}(p, q) < \varepsilon, |p.\text{SOG} - q.\text{Sog}| < MaxSpd$ and $|p.\text{COG} - q.\text{Cog}| < MaxDir\}$. Dist (p, q) is given by the geographical distance between *p* and *q* – considering that they are in a maximum circle of radius approximate to that of Earth.

After DBSCANSD is applied to collected data, nearby geographic positions with similar speed and direction are grouped into a cluster. Then, arbitrary cluster formats can be formed and subdivided depending on speed and direction.

A Gravity Vector (GV) was also applied, dividing a cluster into multiple parts. Thus, each cluster can have multiple GVs. GV is a vector formed by average COG, SOG, latitude, longitude, and distance [2].

Thus, this method was chosen for its robustness, better results, and faster processing than some other clustering algorithms, such as the k-means method.

However, the method was insufficient to obtain the representative trajectories of a port area, producing unusual or impractical trajectories. It was essential to develop and use post-processing to optimize the final result, eliminate unfeasible trajectories, and optimize those obtained.

5. Application and results obtained

DBSCAN requires five input parameters: *DatasetM* (number of points in a trajectory), *Epsilon* (ε-neighborhood), *MinPts* (minimum number of objects), *MaxDir* (maximum direction variation), and *MaxSpd* (maximum speed variation). The complexity of this algorithm is $O(n^2)$, in which *n* is the size of *DatasetM* [2].

Parameter values will depend on the waterway characteristics of each PA [2]. In fact, we found that the Rio de Janeiro PA has particularities capable of influencing results, such as, for example, the existence of a north-south oriented main channel in which virtually all vessel types travel at some point in their demand, whereas vessels of different types and purposes use several other transverse routes on a larger or smaller scale.

Due to the algorithm low average processing time, we could make several attempts to determine the best combination of these parameters. From the experiment, we observed that, in addition to the characteristics inherent to the conformation of PA waterways, the best parameter combination also depends on the typical behavior of the target vessels, i.e., their direction variability, most common navigation routes, speed, and geographical dispersion.

Given the results obtained, we found that we could improve the discrimination of common trajectories by applying a prior filter for individual vessel types to the input data. We justify this choice by the fact that vessels of the same category tend to have similar origins, destinations, and circulation routes, falling within a similar direction and speed range.

Thus, we decided to generate models for each of the defined five vessel categories, enabling the algorithm to separately explore them: Barge (passengers), Freighter (container and bulk carrier), Offshore (platform support vessel), Tanker (oil and gas), and Tugboat.

Figure 3 shows the number of AIS messages per vessel type. Barges carrying passengers exchange a large number of messages due to their regular and constant intervals between trips. Offshore vessels also trade a considerable number of messages due to the proximity of the Rio de Janeiro port to the Campos Basin.

VOL.39 Nº1 2022
https://doi.org/10.22491/IMECTA.11033.en

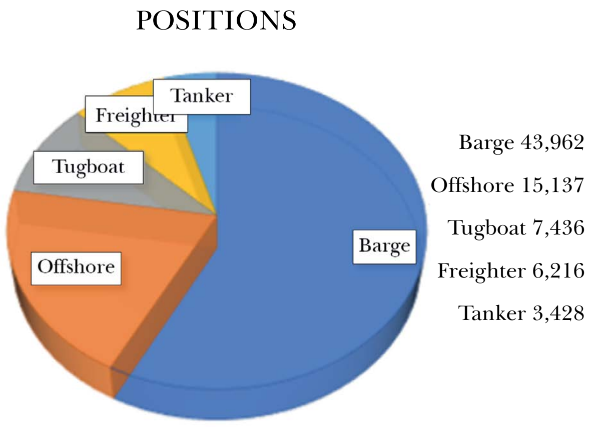

Fig. 3 - Division of entry data into the five vessel categories.

Table 1 shows the parameter combination adopted for each processed category. We began parameter selection by arbitrarily choosing an Epsilon value based on the approximate width of the assessed waterways. Note that, in the specific case of Barges, we needed to reduce this radius since the crossing of several trajectories favored the massification of different trajectories in the same cluster. In the chosen Port Area, there are large variations in the typical trajectories of vessels entering or departing from its various port terminals. Choosing low *MaxDir* values would divide the same trajectory into several clusters. Thus, we opted for a high value for this parameter, which we repeated for all vessel types. Possible *MaxSpd* were limited to a range of feasible values for each chosen vessel type, experimentally defined with *MinPts* values, reaching the parameter combination in **Table 1**.

Tab. 1 - Parameter Selection.

Category	DatasetM (messages)	Epsilon (Nautical Miles)	MinPts (positions)	MaxDir (degrees)	MaxSpd (knots)
Barge	43,962	0.001	2	20.0°	15.0
Freighter	6,216	0.003	6	20.0°	15.0
Offshore	15,137	0.003	6	20.0°	8.0
Tugboat	7,436	0.003	4	20.0°	10.0
Tanker	3,428	0.003	6	20.0°	15.0

5.1 Result post-processing

Category processing improved the discrimination of the most common trajectories, but we still found discrepancies that could be resolved to optimize results.

We first assessed the generation of clusters with few positions, automatically deleting single-point trajectories. Those consisting of two to four positions established the criterion for the distance traveled by the trajectory, automatically eliminating those moving less than 100 yards per position. **Table 2** describes the processed trajectories.

Tab. 2 - Generated, eliminated, and maintained trajectories.

Category	**(A)**	**(B)**	**(C)**	**(D)**	**(E)**
Barge	76	36	28	1	4
Freighter	23	2	4	0	3
Offshore	56	21	8	4	4
Tugboat	29	8	10	2	5
Tanker	30	3	7	0	3

Note: (A) - Total trajectories generated; (B) - trajectories with only one position; (C) - trajectories with less than four positions; (D) - land trajectories; (E) - maintained trajectories.

Trajectories over land or non-navigable areas were also a problem. To automatically detect and eliminate them, we created an image of the nautical chart, as **Figure 4** shows, in which land and non-navigable areas were colored in **RGB** = {255, 0, 0} red. We then mapped the latitudes and longitudes of trajectory points so they corresponded to the pixel positions in the generated image. For each trajectory point, the equivalent pixel is checked, discarding the entire trajectory if its **R value** equals 255.

For the CASNAV Bridge Simulator, it is not interesting to have very close points representing a trajectory since NPC are programmed to always seek the next point in the trajectory and, once reached, correct their course and speed toward the next one. Trajectories with very close points would cause erratic behavior in the NPC, decreasing the fidelity of their behavior to reality.

To solve this problem, we optimized trajectories by reducing collinear points (or those very close to the path) via the Douglas-Peucker algorithm [9].

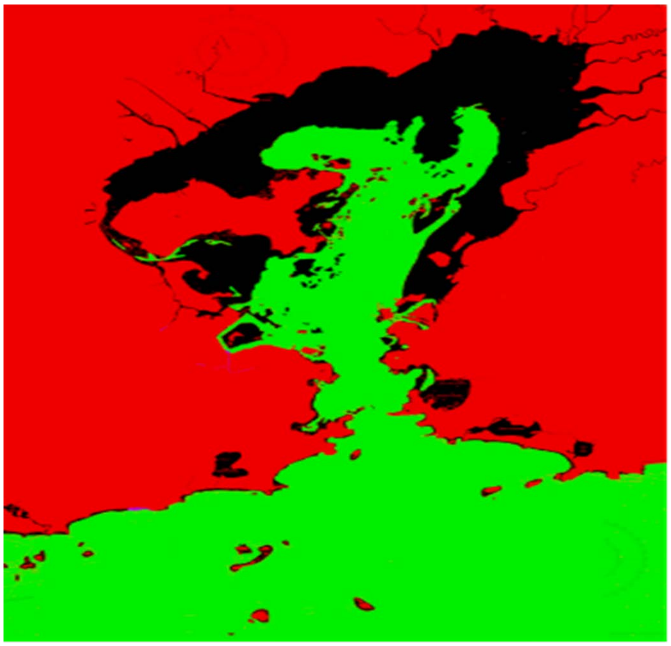
Fig. 4 - Image generated to detect trajectory points on land or non-navigable areas, represented in red, and navigable areas, in green.

This algorithm traverses the trajectory component points, evaluating them three by three and eliminating intermediate points within a predetermined distance of the straight segment formed by its most distal points.

This algorithm can create trajectory segments which cross land or non-navigable points. The maximum distance for eliminating a point must be adjusted to prevent this. The reported experiments adopted values smaller than half the width of the main Guanabara Bay channel, i.e., approximately 100 yards.

This automatic post-processing, as the analysis of **Table 2** shows, still left many trajectories. Determining objective criteria to evaluate the obtained results is difficult. The best analysis seems to be the subjective evaluation of specialists who know the characteristics of maritime traffic in the discussed region. This was the last criterion used to reach the final number of trajectories in **Table 2**.

Values above Epsilon and below MinPts decrease trajectory discrimination, joining different trajectories. However, adjusting parameters to prevent different trajectories from joining creates the tendency of separating stretches of the same trajectory into different clusters.

Another strategy to obtain objective criteria was estimating the standard deviation and coefficient of variation of the directions and velocities of each trajectory, seeking to obtain trends which would help us to choose the best trajectories. We noted that, for some cases, the best trajectories were those with lower variances, whereas, for others, they were those with the highest values. Thus, this is a poor choice criterion.

Figures 5 to **9** show the trajectories obtained before and after post-processing.

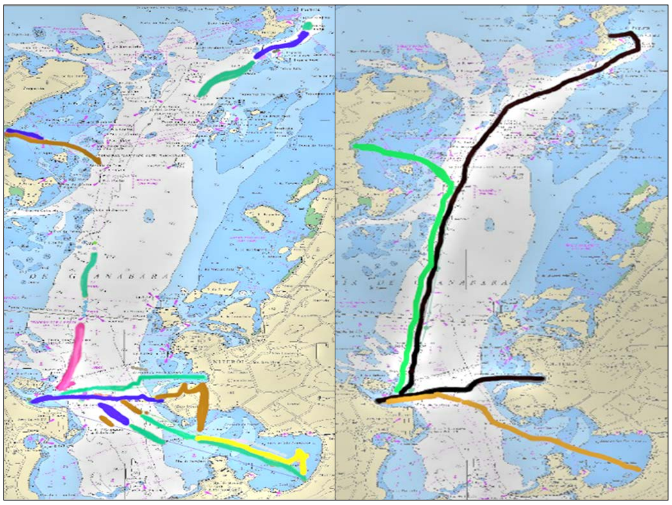
Fig. 5 - Trajectories generated for Barges. Before post-processing (left) and after post-processing (right).

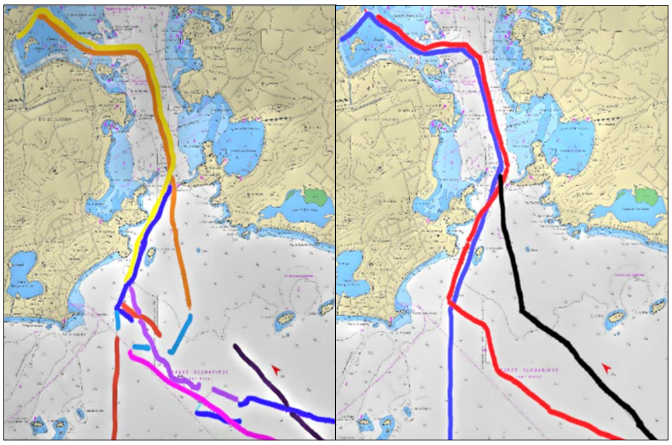
Fig. 6 - Trajectories generated for Freighters. Before post-processing (left) and after post-processing (right).

VOL.39 Nº1 2022
https://doi.org/10.22491/IMECTA.11033.en

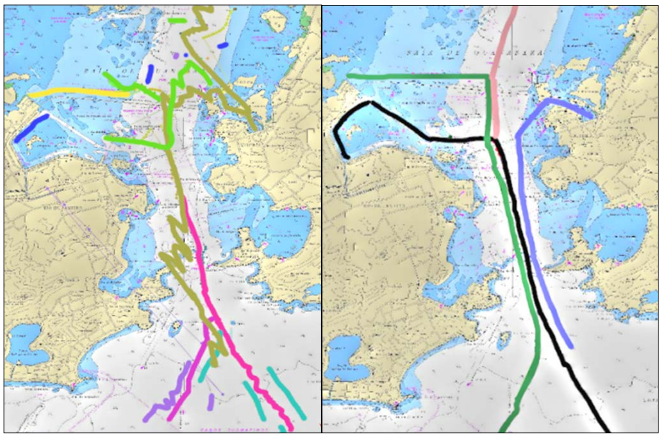
Fig. 7 - Trajectories generated for Offshore. Before post-processing (left) and after post-processing (right).

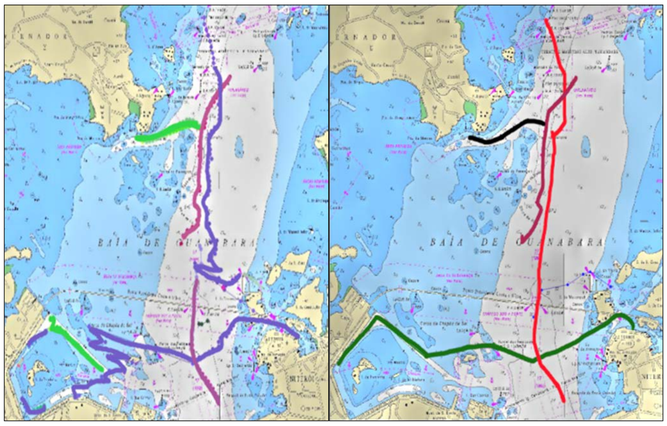
Fig. 8 - Trajectories generated for Tugboats. Before post-processing (left) and after post-processing (right).

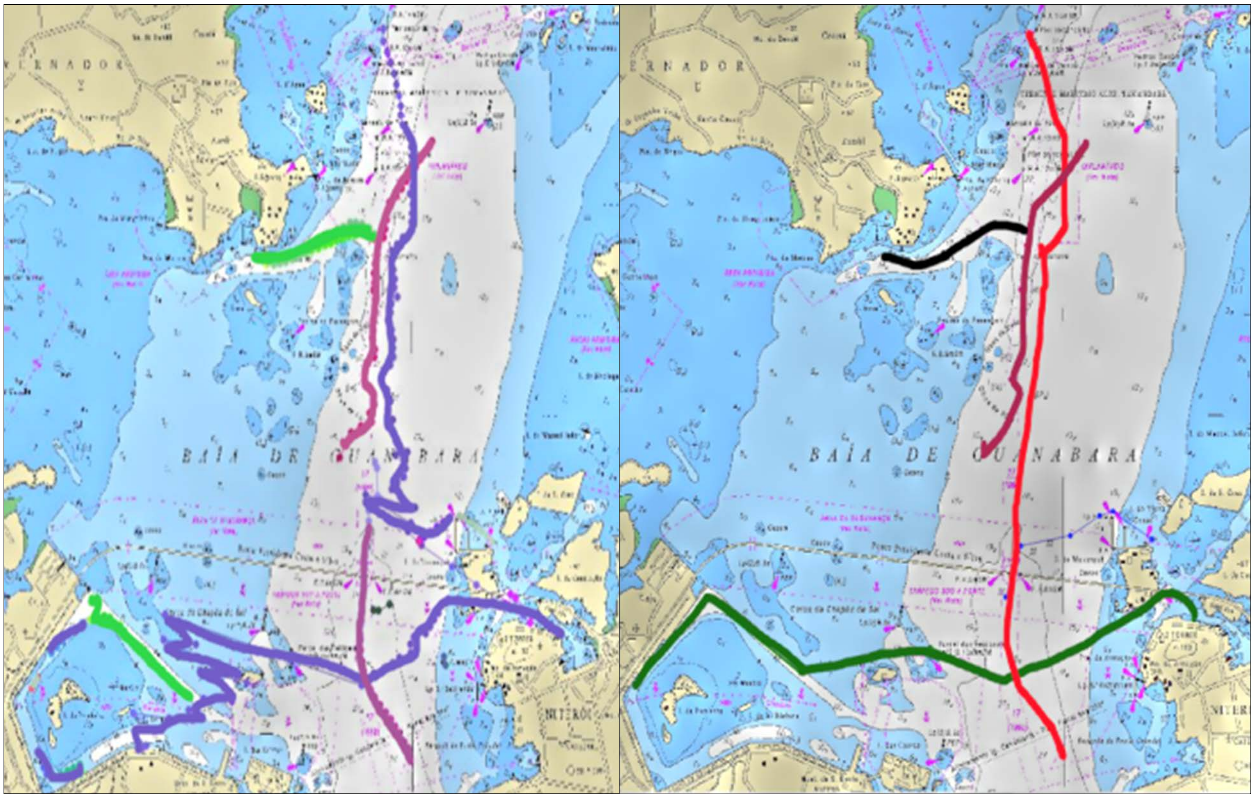
Fig. 9 - Trajectories generated for Tankers. Before post-processing (left) and after post-processing (right).

6. Final Considerations

This study aimed to obtain, from a large mass of AIS message data, trajectories representing the common vessel movement around a PA and to use them to control NPC vessels in the CASNAV Bridge Simulator.

Using the algorithm proposed by Liu et al. [2], together with pre-processing and post-data processing, we could obtain 19 representative trajectories of the chosen five vessel categories, and we considered our results very satisfactory and applicable to simulation exercises conducted within the Brazilian Navy.

Future studies could explore the conjugation of this method with TraClus [10] and evaluate the results. There is ample room for improvements to data pre- and post-processing to optimize the obtained results.

References

[1] LAGE, M., CLUA, E., BARBOZA, D., et al. Simulador de Passadiço.**XI Simpósio Brasileiro de Jogos e Entretenimento Digital – SBGames**,5, 2012.

[2] LIU, Bo; DE SOUZA, ERICO N.; MATWIN, Stan, et al. Knowledge-based clustering of ship trajectories using density-based approach.**Proceedings - IEEE International Conference on Big Data**, 603,2014.

[3] PERERA, L. P., OLIVEIRA, P.; GUEDES SOARES, C.Maritime Traffic Monitoring Based on Vessel Detection, Tracking, State Estimation, and Trajectory Prediction**. IEEE Transactions on Intelligent Transportation Systems**, 13, 1188, 2012.

[4] VRIES, G. K. D. De and Someren, M.; Machine learning for vessel trajectories using compression, alignments and domain knowledge. **Expert Systems with Applications**. **2012**, 13.426, 2012.

[5] DUCA, A. Lo, BACCIU, C. and MARCHETTI, A.; A K-nearest neighbor classifier for ship route prediction. **OCEANS – Aberdeen**, October, 1, 2017.

[6] PALLOTTA, G., VESPE, M.; BRYAN, K.; Vessel Pattern Knowledge Discovery from AIS Data: A Framework for Anomaly Detection and Route Prediction. **Entropy**, 15, 2218, 2013.

[7] ESTER, M., KRIEGEL, H.-P., SANDER, J.; XU, X. Density-based spatial clustering of applications with noise. **Int. Conf. Knowledge Discovery and Data Mining**, 240, 6, 1996.

[8] MARA CASSIANO, K**.; Análise De Séries Temporais Usando Análise Espectral Singular (SSA) E Clusterização De Suas Componentes Baseada Em Densidade**, Tese (Doutorado em Engenharia Elétrica), Pontifícia Universidade Católica do Rio de Janeiro, Brasil, 2014.

[9] PEUCKER, T.; DOUGLAS, D. H. Reflection Essay: Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature Classics in Cartography: **Reflections on Influential Articles from Cartographica**, 29, 2011.

[10] LEE, J., HAN, J. and WHANG, K.-Y. Trajectory Clustering: A Partition-and-Group Framework. **Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data – ACM**, 593, 2007.