Engineering Applications of Artificial Intelligence 90 (2020) 103499

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Time series segmentation for state-model generation of autonomous aquatic drones: A systematic framework[☆]

Alberto Castellini [*], Manuele Bicego, Francesco Masillo, Maddalena Zuccotto, Alessandro Farinelli

Department of Computer Science, University of Verona, Verona, Italy

ARTICLE INFO

Keywords:
Time series segmentation
Situation assessment
State-model generation
Autonomous surface vessels
Activity recognition
Water monitoring
Model interpretation/explanation
Sensor data analysis

ABSTRACT

Autonomous surface vessels are becoming increasingly important for water monitoring. Their aim is to navigate rivers and lakes with limited intervention of human operators, to collect real-time data about water parameters. To reach this goal, these intelligent systems must interact with the environment and act according to the situations they face. In this work we propose a framework based on the integration of recent time-series clustering/segmentation methods and cluster validity indices, for detecting, modeling and evaluating aquatic drone states. The approach is completely data-driven and unsupervised. It takes unlabeled multivariate time series of sensor traces and returns both a set of statistically significant state-models (generated by different mathematical approaches) and a related segmentation of the dataset. We test the approach on a real dataset containing data of six campaigns, two in rivers and four in lakes, in different countries for about 5.6 h of navigation. Results show that the methodology is able to recognize known states and to discover unknown states, enabling novelty detection. The approach is therefore an easy-to-use tool for discovering and interpreting significant states in sensor data, that enables improved data analysis and drone autonomy.

1. Introduction

Autonomous robots have recently had a strong impact in the transition from manual (passive) to autonomous (active) water monitoring. These intelligent systems, used also in several other application domains, such as surveillance and monitoring (Farinelli et al., 2012), are able to autonomously collect large amounts of data, providing crucial support to human operations. Aquatic drones involved in autonomous monitoring of catchments navigate rivers and lakes acquiring real-time data about water parameters, such as pH and dissolved oxygen. While human operators are usually involved in such data collection activities, direct tele-operation of the drones is often not an option for an entire mission, hence autonomous navigation is required. Navigation strategies usually aim at maximizing the information content of acquired data (Bottarelli et al., 2016, 2019), while adapting to the conditions of the environment. Although data are very noisy in this context, applications require minimal number of sensors to reduce the costs.

A key factor for the success of autonomous data acquisition campaigns is *mission awareness* (Endsley, 1995), which is composed of three main elements: knowledge of mission objectives, internal self-situational awareness, and external self-situational awareness. In this work we specifically focus on the problem of detecting, modeling and interpreting aquatic drone states with data-driven methods, an aspect of self-situational awareness. By state we mean an abstract, compact and informative descriptor of key properties of the drone-environment system. In particular, we aim at developing *interpretable models of drone states* from traces of sensor data acquired during water-monitoring campaigns, by means of machine learning and artificial intelligence methods (Hastie et al., 2001; Bishop, 2006; Russell and Norvig, 2009). Generating such a set of drone state-models is important for two reasons, namely, it supports *offline data analysis* by improving the extraction of knowledge from large sensor traces, and it enhances the autonomy of the drone by providing key information for *online decision making* (Kaelbling and Lozano-Perez, 2013; Asperti et al., 2019).

Automatic detection of aquatic drone states from sensor data can be performed by supervised or unsupervised methods. Supervised methods are typically more accurate than unsupervised methods but they need labeled datasets, usually hard, expensive and sometimes impossible to collect in real monitoring campaigns. Ad-hoc experiments could be performed to generate labelings, but they usually consider only subsets of situations that the drone faces during real campaigns. On the other hand, many data are usually available from past campaigns that can be mined by unsupervised methods.

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.engappai.2020.103499.

* Corresponding author.
E-mail address: alberto.castellini@univr.it (A. Castellini).

https://doi.org/10.1016/j.engappai.2020.103499
Received 20 May 2019; Received in revised form 20 December 2019; Accepted 13 January 2020
Available online 21 January 2020

This work focuses on unsupervised approaches, namely *clustering* and *time series segmentation*, able to split multivariate time series into groups of observations corresponding to system states and having common properties that can be compactly represented by mathematical *models*. The goal is to discover these states (and models) using data-driven methods from sensor data of past campaigns. The literature (see Section 2) proposes several methods for this purpose, characterized by different assumptions and extracting different types of patterns. The main difference between the works in the literature and our work is that we propose a *systematic framework* for generating and evaluating statistically significant state-models for aquatic drones, while the literature mainly proposes novel clustering methods or it compares standard methods in different application domains.

We first investigated clustering and subspace clustering methods for detecting aquatic drone states in Castellini et al. (2018b, 2019c). Here, we extend those works using both classic (Bishop, 2006) and very recent methods, including SubCMedians (Peignier et al., 2018), Toeplitz Inverse Covariance-based Clustering (TICC) (Hallac et al., 2017) and Inertial Hidden Markov Models (IHMM) (Montanez et al., 2015). The proposed framework is tested on a large dataset with observations from many campaigns. State-models are analyzed and interpreted in terms of situations faced by the drones. The statistical significance of state-models is computed by comparing their properties with those of random clusters. Since different aspects of state-model performance must be evaluated, we select a set of validity indices (Arbelaitz et al., 2013) satisfying the requirements of our domain.

The main contributions of this paper are summarized in the following:

- we propose an easy-to-use framework for systematically generating and evaluating significant state-models in multivariate time series;
- we successfully apply the proposed framework to a real dataset of sensor data collected by aquatic drones involved in water monitoring;
- we present, analyze and interpret, with high level of detail, both the discovered state-models and the application procedures used to generate these models, which makes this manuscript a valuable reference also for practitioners interested in analyzing similar data and performing extensive cross-comparison of methodologies;
- we present and make available the dataset used in this analysis.[1]

The rest of the manuscript is organized as follows. Section 2 provides an overview of the state-of-the-art on this research topic. Section 3 introduces the aquatic drone architecture and the proposed framework for state-model generation. In Section 4 we describe the dataset and the labelings. Section 5 introduces clustering and segmentation methods, and the procedures for the generation of random clusterings and segmentations. Section 6 defines some clustering validity indices and performance measures. Section 7 illustrates the results and some state-models generated by the proposed framework. Conclusions and future directions are drawn in Section 8.

2. Related work

From the *application* point of view, strong similarities are present with sensor-based human activity recognition (Chen et al., 2012; Dhiman and Vishwakarma, 2019), where sensors are used to acquire data about human movements and machine learning methods are employed to generate activity models and to predict human activities in novel contexts. The main difference between our problem and human activity recognition is that data collected by aquatic drones are very noisy, since they come from several sources (not only accelerometers as in applications of human activity recognition) and are strongly influenced by unstructured and diversified environments (e.g., rivers and lakes in different parts of the world have disparate environmental properties). Moreover, aquatic drones collect two kinds of data, some relating to movement, others to water properties, and both sources of information can be used to assess the drone state.

From a *methodological* viewpoint, the main theoretical connections with our work concern clustering (Bishop, 2006) and time series segmentation (Fu, 2011; Castellini et al., 2015). K-means, Gaussian mixture models (GMM) and hierarchical clustering, have been recently used to identify activities of both humans (Abdallah et al., 2012; Trabelsi et al., 2013; Kwon et al., 2014; Barták and Vomlelová, 2017) and flying drones (Barták and Vomlelová, 2017) from sensor data. Hidden Markov models (HMMs) have been applied (Kim et al., 2010; Trabelsi et al., 2013; Barták and Vomlelová, 2017) and also extended (Fox et al., 2008; Montanez et al., 2015) in the same context. Time series segmentation (Hallac et al., 2016a, 2017; Chiu et al., 2003), change point detection (Barnett and Onnela, 2016) and motif discovery methods, have been employed to identify homogeneous intervals in sequential time-dependent data. The last techniques have been very recently applied also to problems related to driver identification (Hallac et al., 2016b) and state representation of modern automobiles (Hallac et al., 2018).

In previous works we tested standard clustering methods on single campaigns (Castellini et al., 2018a,b) and introduced the usage of subspace clustering for generating sparse state-models (Castellini et al., 2019c,a). What differentiates this paper from our previous work and the approaches in the literature mentioned above is that here we propose a systematic framework for generating statistically significant state-models using very recent techniques and, most important, for evaluating them by several internal and external validity indices. Moreover, we test the proposed framework on a large real dataset in the application domain of autonomous water monitoring and we analyze the statistical properties of detected states. Furthermore, we select some validity indices (Arbelaitz et al., 2013; Moshtaghi et al., 2019) and used them to evaluate and rank the state-models generated by five clustering techniques.

3. System overview

In this section we describe the two main elements of our system, namely the aquatic drone architecture and the framework for state-model generation.

3.1. Data acquisition system: autonomous aquatic drones

Data acquisition campaigns are performed by Lutra mono hull boats (see Fig. 1) produced by Platypus[2] and customized in the EU Horizon 2020 INTCATCH project[3] to accomplish water monitoring of catchments. Localization and orientation are provided by an on-board smartphone which gathers information from GPS, compass and gyroscope. Sensor management and sensor data transmission to the cloud is performed by a Go-Sys BlueBox[4] control unit connected to an Arduino e-board. Operators can define desired paths by setting waypoints in a map on a tablet, to perform autonomous navigation, or they can manually drive the drone using an RC controller. Drones are equipped with sensors for GPS position, water temperature, dissolved oxygen and electrical conductivity, commands to propellers and battery voltage. Sensor traces are stored in log files on the smartphone or transmitted to the cloud by a Go-Sys BlueBox. Log files are preprocessed using Platypus Python libraries to obtain a matrix of time series having one sensor signal in each row and time instants in columns. Since different sensors have different sampling frequencies the alignment of sensor traces was obtained via interpolation and re-sampling, with sampling frequency of 1 Hz.

[1] The dataset will be submitted to *Data in Brief* upon acceptance of this manuscript.

[2] http://senseplatypus.com.

[3] http://www.intcatch.eu.

[4] https://www.go-sys.de/en/bluebox/.

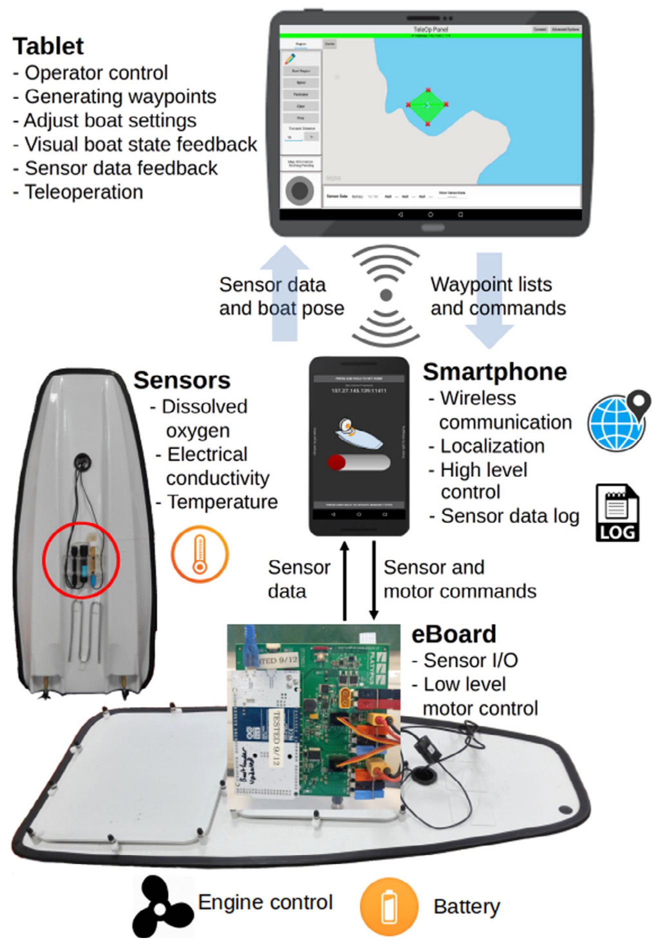

Fig. 1. Overview of the drone architecture.

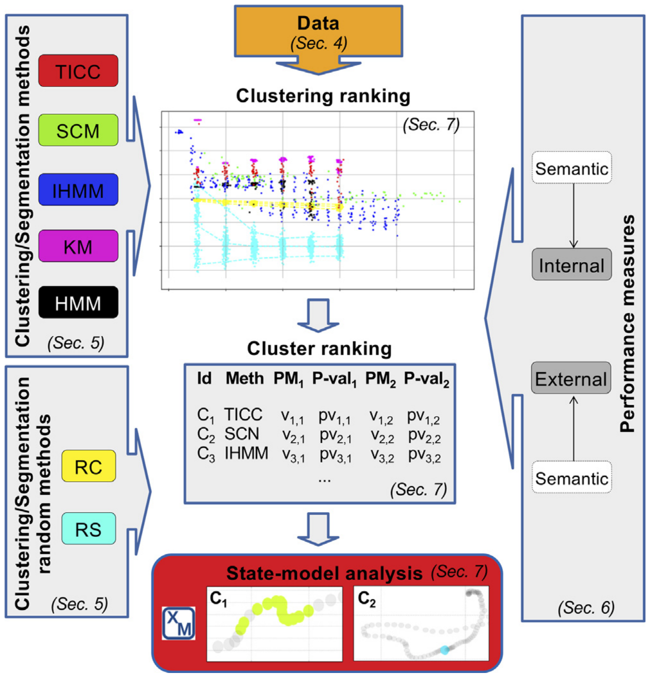
Data (Sec. 4)
Clustering/Segmentation methods
TICC
SCM
IHMM
KM
HMM
(Sec. 5)
Clustering ranking
(Sec. 7)
Cluster ranking
Id Meth PM1 P-val1 PM2 P-val2
C1 TICC v1,1 pv1,1 v1,2 pv1,2
C2 SCN v2,1 pv2,1 v2,2 pv2,2
C3 IHMM v3,1 pv3,1 v3,2 pv3,2
...
(Sec. 7)
Clustering/Segmentation random methods
RC
RS
(Sec. 5)
State-model analysis (Sec. 7)
C1
C2
Performance measures
Semantic
Internal
External
Semantic
(Sec. 6)

Fig. 2. Overview of the proposed framework for state-model generation and evaluation.

3.2. Framework for state-model generation and evaluation

The framework proposed in this work is outlined in Fig. 2. The input *dataset* is a matrix of multivariate time series with engineered features (see Section 4), which contains sensor readings from multiple campaigns. Data are processed by five *clustering and segmentation methods*, namely, k-means (KM), Toeplitz Inverse Covariance-based Clustering (TICC), Hidden Markov Models (HMM), Inertial Hidden Markov Models (IHMM), and SubCMedians (SCM). They generate clusterings depending on parameter settings. Multiple instances of random clustering (RC) and random segmentation (RS) are also generated. They are used as baselines to evaluate the significance of the state-models generated by real clustering algorithms (see Section 5).

Clusterings and related clusters are then evaluated by means of *performance measures* (see Section 6). They have different semantics and can favor different kinds of patterns (i.e., states) in the data (e.g., the *silhouette* is maximized if clusters are both compact and distant from each other, while *spread* considers only the cluster compactness). Performance measures enable to rank clusterings and clusters, and to identify the best state-models. After computing performance, we also determine cluster (clustering) p-values using random partitioning as baselines. Only clusters (clusterings) with low p-values are considered statistically significant. The last step of the proposed framework involves the *analysis and interpretation of significant state-models* (performed in Section 7). Since each state-model is generated by a clustering method, evaluated by some performance measures, and interpreted as a situation, the framework enables different kinds of analyses involving combinations of these properties. For instance, we analyze the statistical properties of significant state-models, compare the capability of different methods to discover specific situations, and compare the capability of different performance measures to rank situations. State-model analysis is supported by a Python tool called eXplainable Modeling[5] (Castellini et al., 2019d) that integrates several data visualization and statistical tools.

4. Dataset

We analyze sensor traces generated in six independent campaigns (also called experiments in the following). Table 1 shows the name, number of samples, duration and type of catchment (i.e., river or lake) of each campaign. Since our goal is to generate a unique set of state-models, we concatenated the traces of all the campaigns, obtaining a single dataset (called *CONCAT*) with 20 187 observations and about 5.6 h of navigation, since the sampling frequency is 1 Hz. Variables available in the raw dataset are time, latitude, longitude, altitude, speed, electrical conductivity, dissolved oxygen, temperature, battery voltage, heading, acceleration, command to propeller 0 and command to propeller 1 (the boat has two propellers). Using only these variables we obtain experiment-dependent state-models because of the strong differences in environmental parameters among different campaigns. To avoid this problem we generate new variables by feature extraction. In particular, we compute *moving means* and *standard deviations* over a sliding windows of 10 s, and *variations* between couples of consecutive observations. The list of 27 variables in the final dataset is reported in Table 2. Z-score standardization was performed on each variable to improve the performance of clustering and segmentation methods.

Mathematical notation. In the following, we use notation $X = \{x_1, x_2, \dots x_n\}$ to represent the dataset, where n is the number of observations (i.e., $n = 20\,187$ in our dataset), each observation $x_i \in X$ has D variables (i.e., $D = 27$ in our dataset). Each variable is represented by a number ranging from 1 to D, and the set of all variables is denoted $\mathcal{D} = \{1, \dots, D\}$.

[5] https://github.com/XModeling/XM.

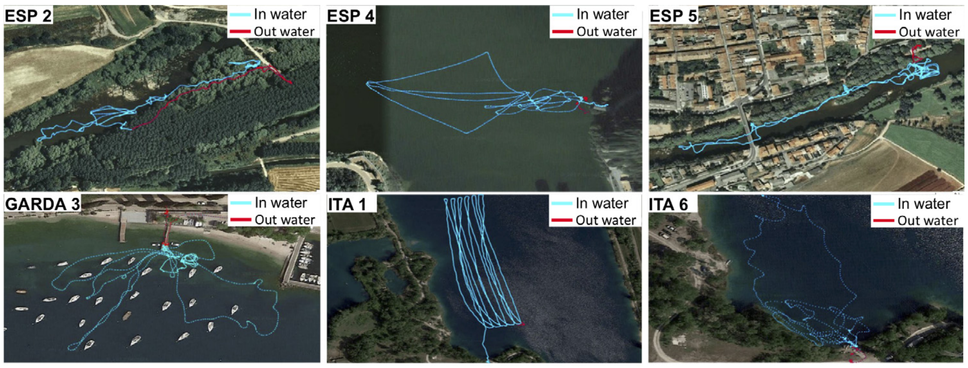

Fig. 3. Geo-localization of monitoring campaigns and manual labeling of situations "drone into the water" (blue) and "drone out of the water" (red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
List of data acquisition campaigns in the dataset.

Id	Campaign name	Samples	Duration	Lake/River
1	ESP2	2814	47′	R
2	ESP5	3601	60′	R
3	ESP4	2374	39′	L
4	GARDA3	2451	40′	L
5	ITA1	7243	121′	L
6	ITA6	1704	28′	L
–	CONCAT	20 187	335′	–

Table 2
List of variables extracted from the dataset and used for clustering/segmentation.

Symbol	Description
s, υ, a	Instantaneous speed, voltage, acceleration
m_0, m_1	Instantaneous signal to propeller 0 and 1
$\bar{s}, \bar{\upsilon}, \bar{a}$	Moving average mean of speed, voltage, acceleration
$\bar{m}_0, \bar{m}_1$	Moving average mean of signal to propeller 0 and 1
$\hat{s}, \hat{\upsilon}, \hat{a}$	Moving average std of speed, voltage, acceleration
$\hat{ec}, \hat{do}, \hat{T}$	Moving average std of electrical conductivity, dissolved oxygen, temperature
$\hat{m}_0, \hat{m}_1$	Moving average std of signal to propeller 0 and 1
$\hat{h}$	Moving average std of heading
$\widetilde{s}, \widetilde{a}, \widetilde{\upsilon}$	Variation of speed, voltage, acceleration
$\widetilde{m_0}, \widetilde{m_1}$	Variation of signal to propeller 0 and 1
$\widetilde{ec}, \widetilde{do}, \widetilde{h}$	Variation of electrical conductivity, dissolved oxygen, temperature

4.1. Known drone states

Some drone states are easy to identify by observing the drone paths in geographical maps but hard to detect from sensor traces, hence recognizing them is not a trivial task for clustering methods. We use these states to test the ability of different methods to detect real situations. The states that we manually label are: drone into the water (IW), drone out of the water (OW), upstream navigation (US), downstream navigation (DS), no water stream (NS), manual drive (MD), autonomous drive (AD), and turning (T). Fig. 3 shows the labeled paths of states IW (cyan) and OW (red).

4.2. Dimensionality reduction analysis

We use t-Distributed Stochastic Neighbor Embedding (t-SNE) (van der Maaten and Hinton, 2008) to see if known situations correspond to implicit structures in the data. t-SNE allows the implicit structure in the data to influence the way in which subset of data points are gathered, hence it reveals structures at different scales. In Fig. 4a, for instance, colors represent experiments (e.g., ESP2) and in Fig. 4c they represent situations in/out water. Projections are informative, they show grouping of observations and correspondence between groups and situations (colors). For instance, the coloring related to in/out water (Fig. 4c) identifies well separated clusters, as expected, although more than one dense region is present for each label.

Table 3
Learning parameters of all clustering methods tested.

Method	Parameter	Values
KM	k	$\{5, 10, 15, 20, 25, 30\}$
	# repeats	50
SCM	NbExtClust	$\{2, 3, 4, 5, 6, 10, 15, 20, 25, 30\}$
	# repeats	10
TICC	k	$\{5, 10, 15, 20, 25, 30\}$
	λ	$\{0.1, 0.5, 0.7, 1.0\}$
	β	$\{0, 50, 100, 150, 200\}$
	w	$\{1, 3\}$
	# repeats	1
HMM	k	$\{5, 10, 15, 20, 25\}$
	# repeats	50
IHMM	k	$\{2, 4, 6, \dots, 38, 40\}$
	ζ	$\{0, 5, 10, \dots, 65, 70\}$
	# repeats	1
RC	k	$\{5, 10, 15, 20, 25, 30\}$
	# repeats	200
RS	k	$\{5, 10, 15, 20, 25, 30\}$
	# repeats	200

5. Clustering and time series segmentation methods

We generate our state-models by five clustering or time series segmentation methods, namely, k-means, SubCMedians, TICC, HMMs and IHMMs. The main difference between clustering and time series segmentation is that clustering does not consider time proximity between observations, while time series segmentation considers it, generating groups of *adjacent* observations (called segments) having common properties. Here we briefly introduce the methodologies and their peculiarities. The sets of parameters used in the training phase, for each method, are also described (see Table 3). Since all methods are unsupervised, the real number of clusters is unknown, hence we test several combinations of methods and parameters and leave the selection of the best state-models to subsequent statistical analysis. Finally, we describe the procedures for generating random clusterings and segmentations.

5.1. K-means (KM)

K-means[6] is an iterative descent clustering method (Bishop, 2006) which aims at minimizing the objective function $J = \sum_{i=1}^{n} \sum_{c=1}^{k} r_{ic} \|x_i -$

[6] https://scikit-learn.org/.

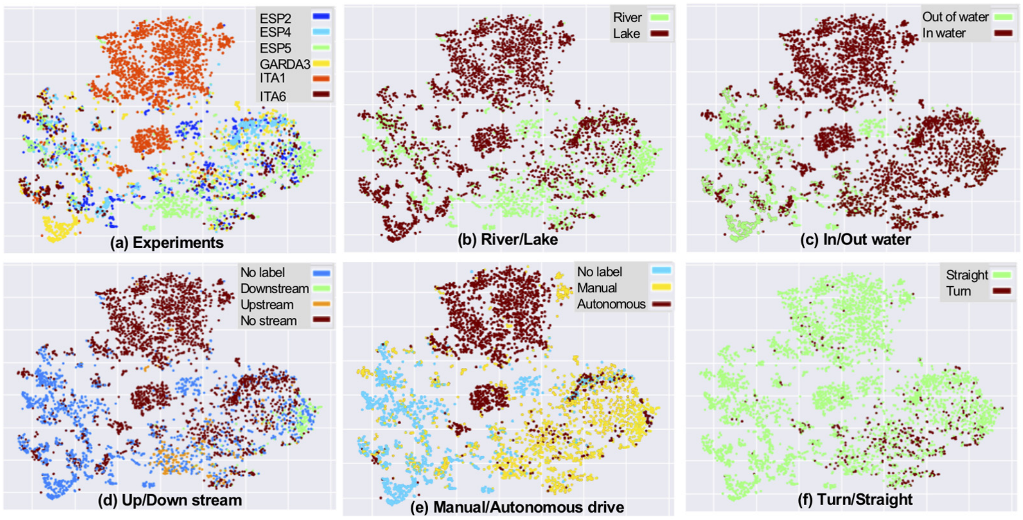

Fig. 4. t-SNE projections. Points represent data observations and colors correspond to known situations. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$\mu_c\|^2$, where $r_{ic} \in \{0,1\}$ is a binary indicator of point-cluster membership, x_i is a data point, μ_c is the centroid of cluster c, n is the number of data points and k the number of clusters. Each clustering is a set of centroids that minimizes J. We use Euclidean distance $\|\cdot\|^2$, number of clusters k listed in Table 3, and for each clustering, we re-initialized the algorithm 100 times and selected the best clustering, since initial conditions influence the solution. We compute 50 clusterings (# repeats in Table 3) for each k.

5.2. SubCMedians (SCM)

SubCMedians is a recent center-based subspace clustering technique (Peignier et al., 2018). This algorithm is based on a K-medians paradigm and it aims at clustering data points around suitable candidate centers $m_i \in \mathcal{M}$, where centers are defined in different subspaces (i.e., subsets of variables) $\mathcal{D}_i \subseteq \mathcal{D}$. In our work, each subspace cluster represents a putative state of the aquatic drone. Formally, the goal of SCM is to build a set of centers $\mathcal{M}$, so as to minimize the Sum of Absolute Errors between the dataset and the centers $SAE(X,\mathcal{M}) = \sum_{x\in X} AE(x,\mathcal{M})$, and such that $Size(\mathcal{M}) \leq SD_{max}$, where $Size(\mathcal{M}) = \sum_i |\mathcal{D}_i|$, and SD_{max} is a parameter denoting the maximum Sum of Dimensions used in $\mathcal{M}$ to describe all its centers. The Absolute Error $AE(x,\mathcal{M})$ represents the distance between each point $x \in X$ and its closest center $m_i \in \mathcal{M}$, and it is computed as $AE(x,\mathcal{M}) = min_{m_i \in \mathcal{M}} dist(x, m_i)$, where $dist(x,m_i) = \sum_{d\in\mathcal{D}_i} |x_d - m_{i,d}| + \sum_{d\in\mathcal{D}\setminus\mathcal{D}_i} |x_d - \mu_d|$ is an extension of the Manhattan distance, with $m_{i,d}$ the coordinate of m_i along variable d, and μ_d the mean of the coordinates of all points in X along d.

The algorithm[7] has three main parameters, namely SD_{max} (described above), the sample size N (the algorithm considers only N randomly chosen observations at each iteration) and the number of iterations $NbIter$ of the training process. The number of centers is not fixed in advance. In Peignier et al. (2018), guidelines are provided to compute all parameters from a single meta-parameter called $NbExpClust$ and representing the expected number of clusters. The actual number of clusters is then computed during training. Table 3 shows the values of $NbExpClust$ that we test and the number of repetitions of each test. The algorithm needs less than one minute to compute a clustering on an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60 GHz with 8 GB of RAM.

[7] https://sergiopeignier.github.io/.

5.3. Toeplitz Inverse Covariance-Based Clustering (TICC)

TICC clusters are modeled as sparse Gaussian inverse covariance (Toeplitz) matrices representing dependencies between variables. In particular, off-diagonal elements represent partial correlations and on-diagonal elements the inverse of variable variances (i.e., variable compactness) inside the cluster. Formally, TICC computes a set of k Toeplitz matrices $\Theta = \{\Theta_1,\ldots,\Theta_k\}$ and a clustering (i.e., assignment of observations to clusters) $P = \{P_1,\ldots,P_k\}$ that solve the following optimization problem (Hallac et al., 2017):

$$\underset{\Theta\in\mathcal{T},P}{\text{argmin}} \sum_{j=1}^{k} \left[\overbrace{\left\|\lambda\circ\Theta_j\right\|_1}^{\text{sparsity}} + \sum_{Y_i\in P_j} \left(\overbrace{-\ell\ell(Y_i,\Theta_j)}^{\text{log likelihood}} + \overbrace{\beta\mathbb{1}\{Y_{i-1}\notin P_j\}}^{\text{temporal consistency}} \right) \right]$$

where $\mathcal{T}$ is the set of symmetric block Toeplitz matrices, $\left\|\lambda\circ\Theta_j\right\|_1$ is an ℓ_1-norm penalty of the Hadamard product aiming to sparsify the inverse covariance matrices, λ is a matrix of regularization parameters that we set to a single value $\lambda \in \mathbb{R}$ to simplify parameter setting, Y_i is a concatenation of observations $x_{i-w+1},\ldots,x_i$, $w \in \mathbb{R}$, $\ell\ell(Y_i,\Theta_j)$ is the log-likelihood that observation Y_i belongs to cluster Θ_j, β is a regularization parameter for temporal consistency, and $\mathbb{1}\{Y_{i-1}\notin P_j\}$ is an indicator function checking if neighboring observations are assigned to same cluster.

The algorithm[8] uses four parameters, namely, λ that controls Toeplitz matrix sparsity, β that controls temporal consistency in clusters, the windows size w used to generate matrix Y from the dataset X, and the number of clusters k. The parameter values and the number of repetitions we test are displayed in Table 3. We set the maximum number of iterations to 100. For time reasons, tests using $w = 3$ are performed only with $\lambda = 1.0$ and $\beta = 0.0$. On an Intel(R) Core(TM) i7-6700HQ CPU @ 2.60 GHz with 8 GB of RAM the algorithm takes from 1 to 30 min to compute a clustering with $w = 1$ (longer time is taken with smaller λs and βs) and between 40 min and 1.5 h with $w = 3$.

5.4. Hidden Markov Models (HMM)

Hidden Markov models (Rabiner, 1989; Bishop, 2006) are probabilistic models which describe Markovian stochastic processes. Observation models are set to single component multivariate Gaussian distributions (with one dimension for each observed variable). The

[8] https://github.com/davidhallac/TICC.

initial state distribution is set to uniform over the set of hidden states, the initial transition matrix is set to a random stochastic matrix, initial means are computed by k-means and initial covariance matrices are set according to the obtained k-means clusters. The maximum number of iterations for the EM algorithm[9] is set to 100. The Viterbi algorithm (Bishop, 2006) is used to generate the most likely sequence of hidden states (i.e., drone states) given the observed sequence of sensor readings. We generated models having number of hidden states (i.e., clusters) listed in Table 3. The learning algorithm was not able to generate clusterings with 30 or more clusters which are instead available for all other methods.

5.5. *Inertial Hidden Markov Models (IHMM)*

IHMMs (Montanez et al., 2015) are a regularization-based extension of HMMs in which the transition matrix is biased towards the inertial property, namely, it has increased self-transition (i.e., on-diagonal) values to better adapt to naturally “long lasting” activities observed in several contexts, such as human activity recognition. The basic idea is to introduce prior knowledge, in the form of a supplementary learning parameter ζ, related to the expected duration of activities, so that the HMM tends to reduce state transitions and, consequently, to generate long segments along the time axis instead of fragmenting adjacent observations in several states. The observation model of each state is represented by the parameters of a multivariate Gaussian distribution. IHMMs are trained by standard EM algorithm, where the transition matrix update is modified to consider parameter ζ. In our tests we set parameters k and ζ as shown in Table 3. The algorithm[10] needs between 30 s and 100 min (longer time is needed when more hidden states are used) to compute a single clustering on an Intel(R) Core(TM) i7-6700 CPU @ 3.40 GHz with 16 GB of RAM.

5.6. *Random clustering (RC)*

Random clusterings are generated by assigning to each observation in the dataset a uniformly random number from 1 to k (the number of clusters). The obtained vector of labels (i.e., numbers from 1 to k) is used as a clustering, hence observations assigned to the same label are put together in the same group. We generate 200 random clusterings for each $k \in \{5, 10, 15, 20, 25, 30\}$ (see Table 3) and use them to compute the statistical significance of clusterings and clusters generated by standard methods.

5.7. *Random segmentation (RS)*

Random segmentations are generated by selecting $k-1$ different random splitting points between 2 and $n-1$, and then assigning label 1 to the observations before the first splitting point, label 2 to observations between the first and the second splitting point, and so on, until the last interval of observations (between the last splitting point and the last observation) which was assigned to label k. In this way we generate k segments of random length, in which each segment is related to a single cluster. As for RC we generate 200 random segmentations for each $k \in \{5, 10, 15, 20, 25, 30\}$ (see Table 3).

6. Performance measures

A key element for evaluating state-models generated by different clustering methods are performance measures. Since different aspects of the performance must be evaluated, here we propose an ensemble of indices that satisfy the requirements of our and possibly other application domains. Selected indices can be split into three categories, namely, measures for evaluating *clusterings*, measures for evaluating single *clusters* (i.e., state-models in our context), and measures for evaluating state-model *variables*. The first and second categories can be further divided into *external* and *internal*. The former uses a ground truth to evaluate the clustering/cluster, while the latter does not require any labeling. Since the goal of the proposed framework is to provide quality state-models from unlabeled data, we focus our analysis on internal performance measures, however, some external measures are presented to assess the capability of clustering methods to detect known situations. For each internal and external measure we specify if it can be applied at clustering level, at cluster level or both. The measures are then used in Section 7 to evaluate, rank, select and interpret state-models generated by different methods. Symbol ↑ (↓) is used to identify measures that must be maximized (minimized). In all indices below the notation $d_e(x_i, x_j)$ is used to represent the Euclidean distance between observations x_i and x_j. We notice that the performance indices here used focus on cluster and clustering goodness, not on their prediction capabilities. We do not split our dataset in training and test set, compute models on training set and evaluate them on test set (a way to evaluate prediction capabilities of state-models). The problem we tackle here comes before the prediction problem, in fact we generate state-models that could be eventually processed to learn prediction models. An advantage of this approach is a lower time complexity (computing prediction performance on test sets needs time consuming cross-validation) which allows us to select optimal state-model among a large set of clusters generated by several combinations of clustering methods and parameter settings.

6.1. *Internal measures*

Silhouette ($\mathcal{S}, \uparrow$). The *silhouette* (Rousseeuw, 1987; Arbelaitz et al., 2013) is an internal measure that contrasts the average distance to elements in the same cluster with the average distance to elements in other clusters. Cluster cohesion is measured based on the distance between all the points in the same cluster, the separation between clusters is based on the nearest neighbor distance. The silhouette of a single observation x_i assigned to a cluster z_c is defined as:

$$\mathcal{S}(x_i^c) = \frac{b(x_i, z_c) - a(x_i, z_c)}{\max\{a(x_i, z_c), b(x_i, z_c)\}}$$

where $a(x_i, z_c)$ is the average distance of x_i from the other observations in cluster z_c and $b(x_i, z_c)$ is the minimum average distance between x_i and the observations in clusters $z_l \neq z_c$. Silhouette can be computed for a specific cluster z_c, as $\mathcal{S}(z_c) = 1/|z_c| \sum_{x_i \in z_c} \mathcal{S}(x_i)$, or for an entire clustering Z, as $\mathcal{S}(Z) = 1/n \sum_{z_c \in Z} \sum_{x_i \in z_c} \mathcal{S}(x_i)$. Its values range from -1 to 1 where high values indicate points belonging to perfectly compact and separated clusters and low values indicate clustering with mixed clusters.

Davies–Bouldin index ($\mathcal{DB}, \downarrow$). Davies–Bouldin index (Davies and Bouldin, 1979; Arbelaitz et al., 2013) estimates the cohesion as the distance from the observations in a cluster to its centroid (computationally faster than computing distances between all pairs of observations in the cluster, as in silhouette) and the separation based on the distance between centroids (also faster than silhouette). The cohesion is divided by the separation, hence the index must be minimized. The index formula is

$$\mathcal{DB}(Z) = 1/k \sum_{z_c \in Z} max_{z_l \neq z_c} \left\{ \frac{C(z_c) + C(z_l)}{d_e(\bar{z}_c, \bar{z}_l)} \right\},$$

where $\bar{z}_c$ is the centroid of cluster z_c and $C(z_c)$ is the estimated cohesion of cluster z_c, $C(z_c) = 1/|z_c| \cdot \sum_{x_i \in z_c} d_e(x_i, \bar{z}_c)$.

Calinski–Harabasz index ($\mathcal{CH}, \uparrow$). Calinski–Harabasz index (Caliński and Harabasz, 1974; Arbelaitz et al., 2013) estimates cluster cohesion from the distances between cluster points and related cluster centroids. The separation is estimated from the distance between the centroids and the global centroid of the dataset $\bar{X}$. The separation term

[9] https://scikit-learn.org/.

[10] https://github.com/george-montanez/InertialRegularizedHMM.

is finally divided by the cohesion term, hence this index is ratio-based and must be maximized. Formally,

$$\mathcal{CH}(Z) = \frac{n-k}{k-1} \frac{\sum_{z_c \in Z} |z_c| d_e(\bar{z_c}, \bar{X})}{\sum_{z_c \in Z} \sum_{x_i \in z_c} d_e(x_i, \bar{z_c})}$$

where $\bar{z_c}$ is the number of observations in cluster z_c, $\bar{z_c}$ is the centroid of z_c.

Spread $(\mathcal{Q}, \downarrow)$. The spread of a cluster is a measure of cluster cohesion (Kelley et al., 1996). Given a cluster z_c containing $|z_c|$ observations the spread is given by

$$\mathcal{Q}(z_c) = \frac{(\sum_{x_i \in z_c} \sum_{x_j \in z_c, j>i} d_e(x_i, x_j))}{|z_c|(|z_c|-1)/2}.$$

The measure can be extended to clusterings by averaging cluster spreads as $\mathcal{Q}(Z) = \frac{\sum_{c=1}^{k} \mathcal{Q}(z_c)}{k}$.

Weighted spread $(\mathcal{R}, \downarrow)$. Since clusters with small number of observations are more likely to be more compact, and consequently to have smaller spread than large clusters, we computed a weighted version of the cluster spread, in which the spread is divided by the percentage of observations in the cluster, namely,

$$\mathcal{R}(z_c) = (\mathcal{Q}(z_c)/|z_c|) \cdot n.$$

The extension to clusterings is obtained as a sum of weighted cluster spread, that is $\mathcal{R}(Z) = \sum_{z_c \in Z} \mathcal{R}(z_c)$.

NMRCLUST penalty $(\mathcal{P}, \downarrow)$. In Kelley et al. (1996) an internal measure is proposed to compare clusterings having different number of clusters and possibly being generated by different methods. The index is computed for a clustering Z as $\mathcal{P}(Z) = \mathcal{NQ}(Z) + k$, where the first term is the sum of the normalized average spread of the clustering

$$\mathcal{NQ}(Z) = \left(\frac{n-2}{max_i(\mathcal{Q}(Z_i)) - min_i(\mathcal{Q}(Z_i))} \right) (\mathcal{Q}(Z) - min_i(\mathcal{Q}(Z_i))) + 1,$$

where $max_i(\mathcal{Q}(Z_i))$ and $min_i(\mathcal{Q}(Z_i))$ are the maximum and minimum values of the average spread of all available clusterings, and the second term is the number of clusters in Z, which is used to compensate the change of normalized average spread among clusterings having different number of clusters.

6.2. External measures

Purity $(\mathcal{U}, \uparrow)$. The purity of a clustering Z with respect to a labeling L is a measure of the extent to which clusters contain a single class. It is computed by the formula $\mathcal{U}(Z) = \frac{1}{n} \sum_{c=1}^{k} \max_{l \in L} |z_c \cap l|$, where Z is a clustering, n is the total number of observations, k is the number of clusters, z_c is the cth cluster, L is the set of classes (i.e., observations with specific labels). Purity close to $1/|L|$ represents fragmented clusterings, while purities close to 1 identify clusterings with almost only one label for each cluster.

Precision $(\mathcal{P}, \uparrow)$. The precision of a cluster z_c with respect to a label class l is a measure of the extent to which the cluster contains the label class. It is computed as $\mathcal{P}_l(z_c) = \frac{|z_c \cap l|}{|z_c|}$, where $|z_c \cap l|$ is the number of observations in the intersection between cluster z_c and label class l, and $|z_c|$ is the number of observations in the cluster z_c. Values close to 1 are obtained when all the observations in the cluster correspond to label class l, values close to 0 are obtained when no observation in z_c corresponds to class label l. We use this measure to find clusters having good match with known states. For instance, to find clusters corresponding to drone turning we search clusters z_c having $\mathcal{P}_T(z_c) \geq 0.5$, where $\mathcal{P}_T$ is the precision for drone turning.

6.3. Measures for model variables

Symmetrical uncertainty $(\mathcal{SU}, \uparrow)$. Symmetrical uncertainty (Hong et al., 2008) is a measure of relevance of a variable $v_d, d \in \{1, \dots, D\}$ with respect to a clustering Z and can be computed as

$$\mathcal{SU}(v_d, Z) = 2\left(\frac{IG(v_d \mid Z)}{H(v_d) + H(Z)} \right)$$

where $H(Z)$ is the entropy of the clustering labels and $IG(v_d \mid Z)$ is the information gain that is computed as $IG(v_d \mid Z) = H(v_d) - H(v_d \mid Z)$, and $H(v_d)$ is the entropy of variable v_d and $H(v_d \mid Z)$ is the conditional entropy of v_d given Z. A value 1 of $\mathcal{SU}$ indicates that the variable v_d is completely related to clustering Z while a value 0 means that the variable v_d is absolutely irrelevant since it does not share any information with clustering Z. It happens for instance, if v_d is a uniformly distributed random variable.

6.4. Statistical significance of clusterings and clusters

For each internal and external measure defined above it is possible to compute the statistical significance, based on p-value, of a clustering Z with respect to the random clustering RC and the random segmentation RS described in Sections 5.6 and 5.7, respectively. The p-value of a clustering Z with respect to a performance measure I is computed as the percentage of random clusterings (random segmentations) that outperform clustering Z in terms performance measure I. The same approach can be used to compute the statistical significance of single clusters. Only clusters/clusterings with percentage less than 0.05 are considered statistically significant.

7. Results and discussion

We generate 1076 clusterings of our dataset using the five clustering methods described in Section 5 with different parameter settings for each method (see Table 3): 126 clusterings are generated by TICC, 300 by IHMM, 100 by SCM, 300 by KM and 250 by HMM. The total number of clusters generated in this way is 19 320 (i.e., 2205 clusters produced by TICC, 5739 by IHMM, 2376 by SCM, 5250 by KM and 3750 by HMM). To evaluate the statistical significance of clusterings and clusters we compute 200 random clusterings (RC) and 200 random segmentations (RS) for each $k \in \{10, 15, 20, 25, 30\}$, a total of 1200 random segmentations (21 000 random segments) and 1200 random clusterings (21 000 random clusters), and we use them to compute clustering and cluster p-values with respect to different performance measures. We rank both single clusters and entire clusterings according to their performance, and compute their statistical significance with respect to the random clusterings/segmentations. In this way, we select a subset of clusterings and clusters having clear evidence of being non-random and to represent drone states. In the following, we first perform an analysis of single cluster and then of entire clusterings. We always compare clusters (clusterings) having the same parameter k since all performance measures considered are influenced by this parameter. Specific focus is put on $k = 10$ and $k = 20$, two levels of granularity (i.e., abstraction) of interest to discover macroscopic states (e.g., in water) and microscopic states (e.g., turning). We notice that the extraction of statistically significant state-models is often better achieved using cluster validity indices than clustering performance indices, because good (e.g., compact and separated) clusters are sometimes present also in clusterings having average/low performance, which would not be selected using only clustering performance indices. This happens, for instance, when a high number of clusters is used, which favors the identification of small patterns but also generates non-significant clusters that reduce the overall performance of the clustering, even in the presence of good clusters. This motivates our choice to analyze deeper single clusters than complete clusterings, although the analysis of clusterings is an important tool for identifying, for instance, the number of clusters in the dataset.

7.1. Analysis of single clusters

Clusters are first ranked according to performance measures of Section 6. We consider only statistically significant clusters, having p-value less than 0.05 for at least one performance measure. A summary of properties and performance of investigated clusters is reported in Table 4. Fig. 5 shows the results for two internal measures, i.e., silhouette

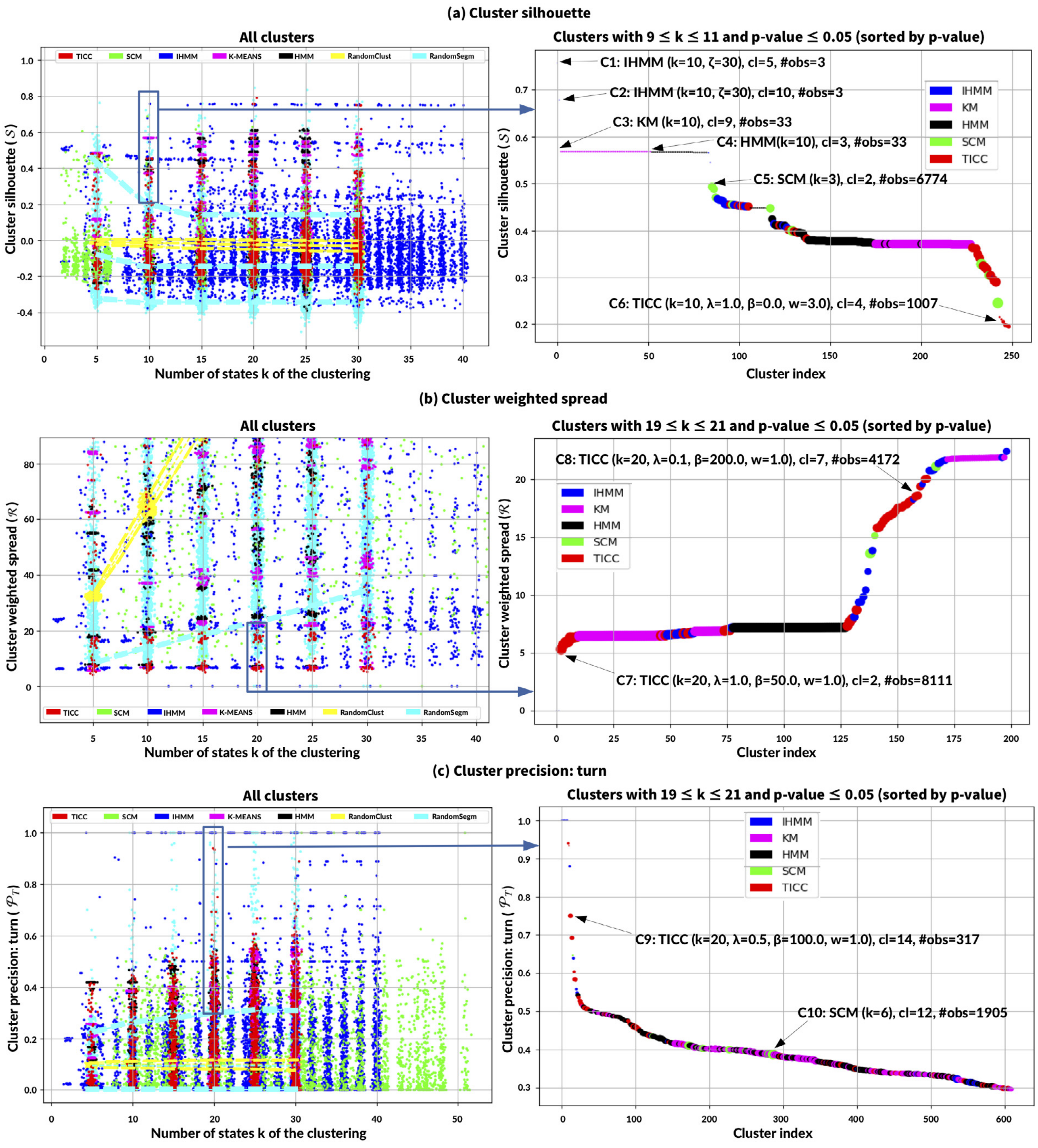

Fig. 5. Performance of single clusters. Left: X-axes are number of states k in the clustering, y-axes are values of cluster performance, colors are clustering methods, light blue dashed lines represent 5th and 95th percentiles for RS, yellow dashed lines 5th and 95th percentiles for RC. Right: statistically significant clusters sorted by performance. *(a)* Cluster silhouette: significant if above the upper dashed lines. *(b)* Cluster weighted spread: significant if below the lower dashed lines; only the 5th percentile line is visible for RS because the figure is zoomed on the lower part of the y-axis. *(c)* Cluster precision for drone turns: significant if above the upper dashed lines. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

($\mathcal{S}$) and weighted spread ($\mathcal{R}$), and one external measure, i.e., precision in detecting drone turns ($\mathcal{P}_T$). For each performance measure, we show on the left a scatter plot displaying all the 61 320 clusters (19 320 generated by clustering methods, 21 000 by RC and 21 000 by RS) where each point is a cluster, the x-axis is the number of states k in the clustering, and the y-axis is the performance of the cluster. On the right, we display clusters having a specific range of k and p-value less than 0.05 for RS. Below, we propose an analysis of few of these clusters, showing that they have a clear interpretation in terms of drone states. Further analysis is reported in supplementary material.

Ranking by cluster silhouette. Fig. 5a shows cluster silhouette and the ranking by silhouette of clusters with k between 9 and 11. The cyan and yellow dashed lines, on the left, characterize the 5th and the 95th percentile with respect to RS and RC, respectively. Clusters located above these lines are statistically significant. Focusing on k between 9 and 11 (see the blue box on the left of Fig. 5a) we find 249 clusters, of which 27 generated by TICC, 21 by IHMM, 9 by SCM, 100 by KM and 92 by HMM. These clusters are ranked by silhouette on the right of Fig. 5a where the point color depends on clustering techniques and point size on cluster size.

Clusters C_1 and C_2 have the highest silhouette, respectively 0.76 and 0.68, and are generated by IHMM. As displayed in Table 4, they have a very small number of observations, namely three per cluster (see column $\mathcal{O}$), they do not correspond to a turn ($\mathcal{P}_T$ = 0.00), but

Table 4
Performance measures and main properties of ten selected clusters. *Id* is the cluster identifier, *clustering method* the technique by which the cluster was generated, *selection method* the performance measure by which it was selected (only clusters having p-value less than 0.05 for that measure were considered), *parameters* are the clustering parameters used to generate the cluster, $\mathcal{O}$ is the number of observations in the cluster, $\mathcal{S}$ is the cluster silhouette, $\mathcal{R}$ its weighted spread, $\mathcal{P}_T$ the cluster precision for drone turns, $\mathcal{P}_{IW}$ the precision for state "in water" (notice that the precision for the state "out of water" can be calculable as $1-\mathcal{P}_{IW}$), $\mathcal{P}_{MD}$ the precision for state "manual drive" (precision of autonomous drive is $1-\mathcal{P}_{MD}$), $\mathcal{P}_{US}$ is the precision for state "upstream navigation", $\mathcal{P}_{DS}$ is the precision for state "downstream navigation", $\mathcal{P}_{NS}$ is the precision for state "no-stream", and *p-val* is the p-value for RS related to the index in the selection method.

Id	Clustering method	Selection method	Parameters	$\mathcal{O}$	$\mathcal{S}$	$\mathcal{R}$	$\mathcal{P}_T$	$\mathcal{P}_{IW}$	$\mathcal{P}_{MD}$	$\mathcal{P}_{US}$	$\mathcal{P}_{DS}$	$\mathcal{P}_{NS}$	p-val
C_1	IHMM	$\mathcal{S}$ (1st)	$k=10, \zeta=30$	3	**0.76**	21 816.9	0.00	**1.00**	**1.00**	0.00	0.00	**1.00**	0.00
C_2	IHMM	$\mathcal{S}$ (2nd)	$k=10, \zeta=30$	3	**0.68**	29 516.8	0.00	**1.00**	**1.00**	0.00	0.00	**1.00**	0.001
C_3	KM	$\mathcal{S}$ (3rd)	$k=10$	33	**0.57**	5143.7	0.00	**0.64**	**1.00**	0.00	0.00	**1.00**	0.0025
C_4	HMM	$\mathcal{S}$ (53th)	$k=10$	33	**0.57**	5143.7	0.00	**0.64**	**1.00**	0.00	0.00	**1.00**	0.0025
C_5	SCM	$\mathcal{S}$ (86th)	$NbExpClust=3$	6774	**0.49**	17.8	0.02	**0.99**	0.08	0.00	0.01	**0.99**	0.005
C_6	TICC	$\mathcal{S}$ (246th)	$k=10, \lambda=1.0$, $\beta=0.0, w=3.0$	1007	**0.21**	132.9	0.12	**1.00**	**0.86**	**0.77**	0.00	0.23	0.047
C_7	TICC	$\mathcal{R}$ (3th)	$k=20, \lambda=1.0$, $\beta=50.0, w=1.0$	8111	0.35	**5.32**	0.02	**0.98**	0.16	0.00	0.03	**0.97**	0.0005
C_8	TICC	$\mathcal{R}$ (160th)	$k=20, \lambda=0.1$, $\beta=200.0, w=1.0$	4172	−0.024	**18.59**	0.01	0.22	**0.89**	0.00	0.00	**1.00**	0.0305
C_9	TICC	$\mathcal{P}_T$ (13th)	$k=20, \lambda=0.5$, $\beta=100.0, w=1.0$	317	−0.174	766.20	**0.75**	**1.00**	**1.00**	0.41	0.00	**0.59**	0.0045
C_{10}	SCM	$\mathcal{P}_T$ (287th)	$NbExpClust=6$	1905	−0.19	68.87	**0.39**	**1.00**	**0.92**	0.10	0.17	**0.73**	0.027

they correspond to locations in which the drone was into the water ($\mathcal{P}_{IW}=1.00$), manually driven ($\mathcal{P}_{MD}=1.00$) and navigating outside strong streams ($\mathcal{P}_{NS}=0.00$). Note that information about precision comes from manual labeling. It is used for result validation and not provided to the (unsupervised) clustering learning process.

We discovered that these clusters identify a real pattern in experiment ESP4 which can be traced back to a specific (possibly *anomalous*) situation. The boxplot of variable $\hat{ec}$ in Fig. 6a shows that clusters C_1 and C_2 have much higher standard deviation of electrical conductivity than other clusters. Then, the boxplot of variable $\tilde{ec}$, in the same figure, points out that in C_1 the variation of ec is positive (increment) and in C_2 it is negative (decrement). The third and fourth boxplots instead say the two clusters have also high standard deviation of temperature and voltage. The geolocalization in Fig. 6b shows that cluster C_2 precedes cluster C_1. All these information, together, suggest that this pair of clusters could be associated to a location where the drone was suddenly extracted from and put back into the water. The location of the clusters is in the middle of a lake, hence the situation could be due to manual intervention of an operator from a boat, anomalous conditions (e.g., obstacles or waves), or sensor faults. It is important to detect such situations to improve data analysis and avoid misinterpretations of sensor readings.

Other key information about this state is provided by the parameters of the IHMM representing the state-models. Fig. 6c shows the heatmaps of variable means for each cluster (on the left) and the transition matrix (on the right). Cluster C_1 has strongly positive means for $\hat{ec}$ and $\tilde{ec}$ (see dark green cells in the first column of the means matrix) and cluster C_2 has strongly positive mean for $\hat{ec}$ and strongly negative mean for $\tilde{ec}$ (second column of the means matrix). Moreover, the switch between cluster C_2 and cluster C_1 is represented by the high parameter in the highlighted cell of the transition matrix (on the right). We reported other analysis on clusters C_3 to C_6 in the supplementary material.

Ranking by cluster weighted spread. This ranking of clusters is displayed in Fig. 5b. On the right we show the significant clusters with k between 19 and 21. We found 199 significant clusters, of which 42 generated by TICC, 29 by IHMM, 3 by SCM, 75 by KM and 50 by HMM. Cluster C_7 has almost the best performance in the ranking (two other clusters perform better but they contain only one observation). It was generated by TICC, contains 8111 observations, has weighted spread 5.32 and silhouette 0.35. This cluster corresponds to observations in which the drone was into the water (i.e., $\mathcal{P}_{IW}=0.98$), autonomously driven (i.e., $\mathcal{P}_{MD}=0.16$), not in strong streams (i.e., $\mathcal{P}_{NS}=0.97$) and not turning (i.e., $\mathcal{P}_T=0.02$). Interestingly enough, this cluster contains almost the same points of cluster C_5, which was generated by SubCMedians and selected from the silhouette ranking. This shows that different clustering methods (i.e., SubCMedians and TICC in this case) were able to discover the same state of the drone although using different state representations (i.e., centroids and Toeplitz matrices). Cluster C_8 is analyzed in the supplementary material.

Ranking by cluster precision for drone turning. The third ranking we analyze is based on the precision to detect drone turns. A scatter plot of clusters arranged by k (x-axis) and precision to detect drone turns $\mathcal{P}_T$ (y-axis) is displayed on the left of Fig. 5c. We focus, in particular, on k between 19 and 21. These clusters are 609 in total, of which 101 generated by TICC, 36 by IHMM, 17 by SCM, 212 by KM and 243 by HMM. The best 15 clusters, having $\mathcal{P}_T \geq 0.69$, are all generated by TICC or IHMM that seem to have the best capability to detect drone turns.

Cluster C_9 is the first "large" cluster in the ranking (317 observations) and it is generated by TICC. Its precision on drone turns $\mathcal{P}_T$ is 0.75, meaning that the 75% of its observations in the cluster correspond to real turn, according to our manual labeling. According to Table 4 this cluster corresponds to observations taken into the water (i.e., $\mathcal{P}_{IW}=1.00$) during manual drive (i.e., $\mathcal{P}_{MD}=1.00$), partially in upstream navigation and partially with no stream (i.e., $\mathcal{P}_{US}=0.41$ and $\mathcal{P}_{NS}=0.59$). Among the main statistical properties of variables characterizing this clusters there are high standard deviation of signal to propellers $\hat{m}_0$ (and $\hat{m}_1$), and high standard deviation of voltage $\hat{v}$, as shown in the two boxplots of Fig. 7a. The geolocalization of this cluster confirms its correspondence to curves in the drone path, as shown in Fig. 7b that displays five locations belonging to three campaigns (i.e., ESP2, ESP5 and ITA6). We observe that the cluster really characterizes the turning pattern in the data. Fig. 7c shows the on-diagonal elements (on the left) and the off-diagonal elements (on the right) of the Toeplitz matrix representing this state. Cluster C_{10} is analyzed in the supplementary material.

7.2. Analysis of clusterings

Here we perform a second kind of analysis based on clustering significance (the previous one was on cluster significance). We evaluate our clusterings, computed by different methods and different parameter settings, according to four internal measures, namely silhouette ($\mathcal{S}$), Davis–Bouldin index ($\mathcal{DB}$), weighted spread ($\mathcal{R}$), and Calinski–Harabaz index ($\mathcal{CH}$). Results are summarized in Fig. 8, which has a similar structure to Fig. 5. Scatter plots, on the left, contain one point for

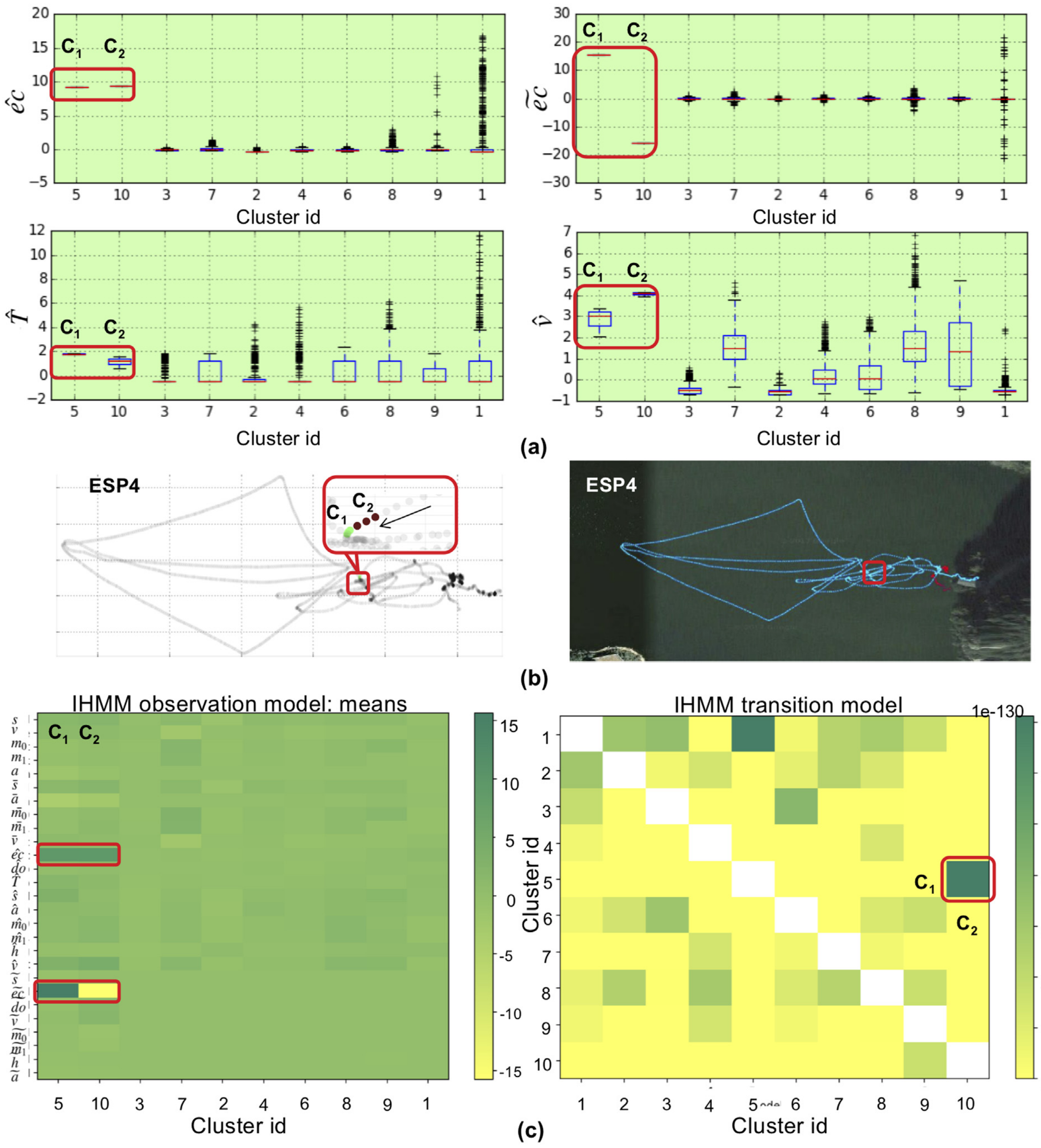

Fig. 6. Clusters C_1 and C_2. (a) Box plots of variables $\hat{ec}$, $\widetilde{ec}$, $\hat{T}$, $\hat{v}$. (b) Maps of cluster locations. (c) State-model parameters (variable means and transition matrix).

each clustering. The x-axis represents the number of clusters k in the clustering and the y-axis the performance measure of interest. Point colors correspond to different clustering methods. On the right hand side some selections of significant clusterings, with specific k and p-value less than or equal to 0.05, are displayed by ascending/descending performance.

Clustering silhouette is displayed in Fig. 8a. As expected the best silhouette is achieved by clustering with small number of clusters (e.g., $k = 2$ for IHMM, $k = 5$ for k-means and TICC, $k = 6$ for SCM). The average clustering silhouette however increases from $k = 10$ to $k = 25$ and then it decreases for $k > 25$, showing a peak around $k = 25$ for all methodologies. This is interesting because it suggests a best number of clusters (around 25) for this dataset. Moreover, silhouette of SCM and IHMM with $k > 30$ sharply degrades to zero or less than zero. Surprisingly, the best silhouette is achieved by k-means for all k (see pink points in the chart). Then TICC reaches the second best silhouette performance, followed by SubCMedians and IHMM that has similar average performance to HMM but better performance considering the best parameter settings. The silhouette of non-random clusterings is almost always higher than silhouette of random segmentations. This behavior is very different from that observed for clusters, wherein several superpositions were present. Ranking by silhouette of clusterings with k between 9 and 11 (on the right of Fig. 8a) shows that the best clustering was generated by SCM and has a silhouette of 0.17. It is followed by k-means (about 0.15) and TICC (about 0.14), then there is a big jump to reach the best IHMM clustering, having silhouette 0.08, and HMM with silhouette 0.07.

The Davis–Bouldin index, in Fig. 8b, is again dominated by k-means (see the pink points in the chart) that shows, as for silhouette, an optimum (i.e., a minimum for Davis–Bouldin index) in k between 20 and 25. The performance of the other methods (considering the best models for each technique while k varies between 5 and 30) are quite

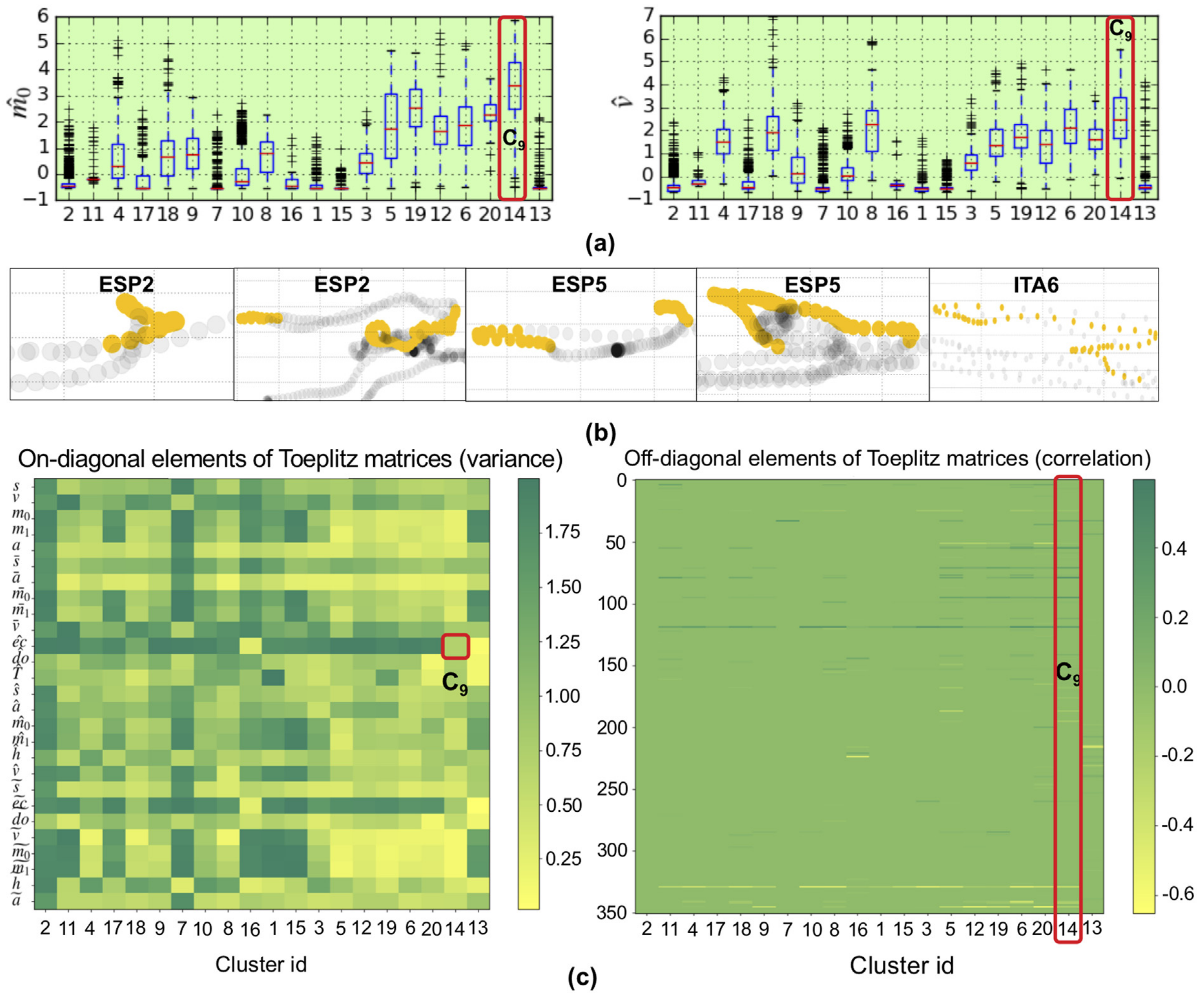

Fig. 7. Clusters C_9. (a) Box plot of variables $\hat{m}_0$, $\hat{v}$. (b) Maps showing cluster locations. (c) State-model parameters (on-diagonal and off-diagonal elements of Toeplitz matrices).

constants over k, with best performance achieved mainly by TICC, SCM and IHMM depending on k. Not considering small k, TICC has its best performance in $k = 25$, IHMM and HMM in $k = 20$, SCM in $k = 39$ (with small differences with other k). All points are below the cyan and yellow points of RS and RC (yellow points are not displayed because of too high values). Weighted spread and Calinski–Harabaz indices are analyzed in supplementary material.

A final comment is focused on clustering p-values. Differently from clusters, clusterings are almost all statistically significant with respect to RC and RS. This holds for all the four internal performance measures analyzed in this section, as displayed in Fig. 8, where the points related to non-random clusterings are almost always out of the areas delimited by the 5th and 95th percentile lines (yellow and cyan dashed lines). This is possibly due to the fact that randomly generate clusterings with performance similar to that of state-of-the-art clustering algorithms is more difficult than randomly generate single clusters with performance similar to that generated by state-of-the-art methods.

8. Conclusions and future work

The framework proposed in this work allows to identify significant states of aquatic drones involved in water monitoring by means of diverse unsupervised clustering and segmentation methodologies. The analysis of the models of these states, namely, centroids, Toeplitz matrices, and multivariate Gaussian distributions (depending on the methodology that generated them), allows us to discover the statistical properties that characterize some of these states and, consequently, to provide interpretations for the related models. This result has direct consequences on the analysis of the data acquired by the drones since we can now label the dataset by discovered states, obtaining a compact semantic-based way to represent each campaign. This could have strong impact on water monitoring projects involving the citizenship in collecting evidence about water healthiness (following the citizen science approach), since unskilled people need support in data interpretation.

From a more general point of view, the proposed framework represents an easy-to-use tool for discovering significant states in multivariate time series datasets and for comparing the capabilities of different clustering techniques. It only needs a dataset and a set of parameter settings for each methodology, and produces several rankings of clusterings/clusters with associated significance levels, allowing to compare the performance of different methods to identify states in specific application domains (and related datasets). The choice of a clustering/segmentation method for real datasets is a challenging activity and our approach could provide valuable support in this direction.

Future activities will aim to release an easy-to-use software for supporting the proposed framework. Then we want to merge the clusters discovered by different methods using different levels of granularity (i.e., parameter k) into a hierarchical (voting) structure, so that each observation could be part of several clusters of different abstraction levels (e.g., drone into the water, turning and moving upstream). Another goal is to focus on specific situations of interest, such as anomalies and dangerous states (e.g., high waves). We are planning specific field tests to this purpose. Finally, we want to integrate our state recognition method into online sequential decision making algorithms, such as

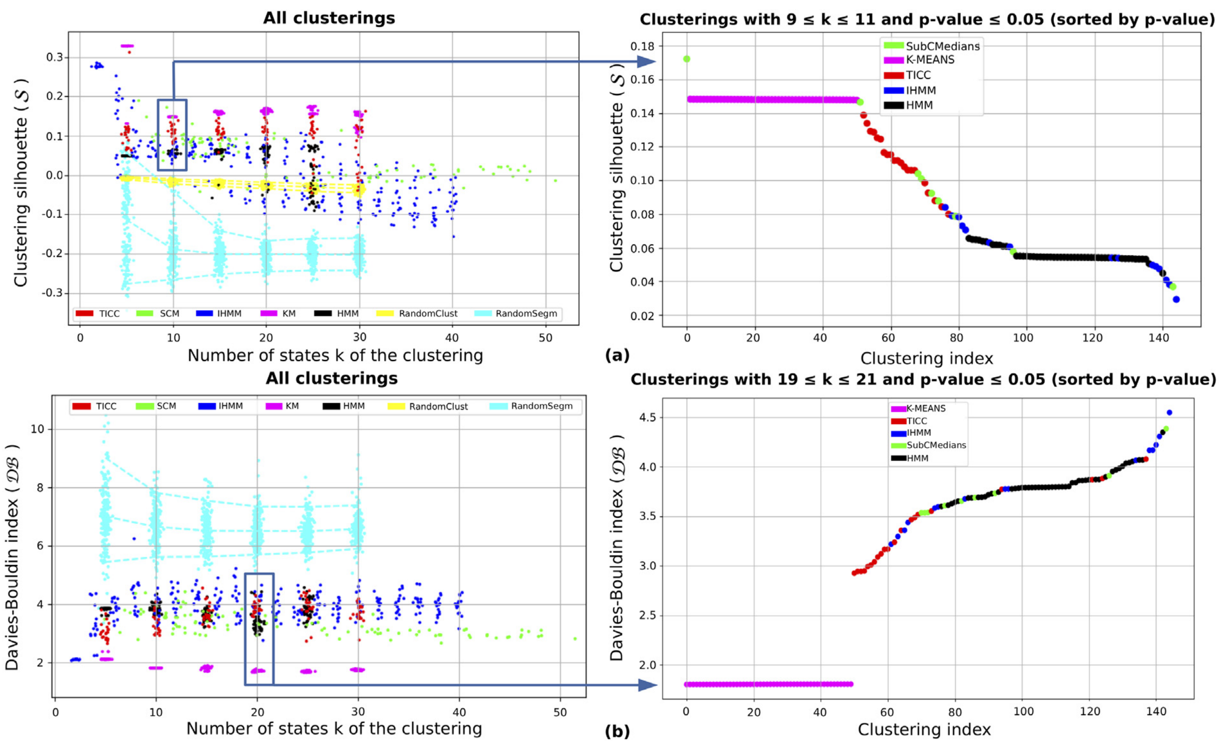

Fig. 8. Performance of clusterings. Left: x-axis is the number of states k, y-axis is the performance value, colors are clustering methods. Each point is a clustering. Right: significant clusterings sorted by performance. (a) silhouette, (b) Davis–Bouldin index. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

those based on Partially Observed Markov Decision Processes (known as POMDPs) that we started to develop in Castellini et al. (2019b). This direction could improve drone autonomy by supporting the generation of policies based on improved system states.

CRediT authorship contribution statement

Alberto Castellini: Conceptualization, Methodology, Writing - review & editing, Investigation, Validation, Software. **Manuele Bicego:** Writing - review & editing, Formal analysis. **Francesco Masillo:** Software, Visualization. **Maddalena Zuccotto:** Software, Visualization. **Alessandro Farinelli:** Supervision, Project administration, Funding acquisition, Conceptualization, Writing - review & editing.

Acknowledgments

This work is partially funded by the European Union's Horizon 2020 research and innovation program under grant agreement No 689341 and partially supported by the project "Dipartimenti di Eccellenza 2018-2022", funded by the Italian Ministry of Education, Universities and Research (MIUR). This work reflects only the authors' view and the EASME is not responsible for any use that may be made of the information it contains.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.engappai.2020.103499. The document contains results and discussions not displayed in this manuscript for space reasons. In particular, supplementary material contains analysis about:

- ranking by cluster silhouette (clusters C_3 to C_6 in Fig. 5)
- ranking by cluster weighted spread (cluster C_8 in Fig. 5)
- ranking by cluster precision for drone turning (cluster C_{10} in Fig. 5)
- ranking by clustering weighted spread and Calinski-Harabaz index (extending ranking by clustering in Fig. 8)

References

Abdallah, Z.S., Gaber, M.M., Srinivasan, B., Krishnaswamy, S., 2012. CBARS: Cluster based classification for activity recognition systems. In: Hassanien, A.E., Salem, A.B.M., Ramadan, R., Kim, T. (Eds.), Advanced Machine Learning Technologies and Applications: First International Conference, AMLTA 2012, Cairo, Egypt, December 8-10, 2012. Proceedings. Springer Berlin Heidelberg, pp. 82–91.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J.M., Perona, I., 2013. An extensive comparative study of cluster validity indices. Pattern Recognit. 46 (1), 243–256.

Asperti, A., Cortesi, D., Sovrano, F., 2019. Crawling in Rogue's dungeons with (partitioned) A3C. In: The 4th Int. Conf. Machine Learning, Optimization and Data Science (LOD 2018), Volterra, Italy. Springer.

Barnett, I., Onnela, J.-P., 2016. Change point detection in correlation networks. Sci. Rep. 6, 18893.

Barták, R., Vomlelová, M., 2017. Using machine learning to identify activities of a flying drone from sensor readings. In: Proceedings of Florida Artificial Intelligence Research Society Conference, FLAIRS 2017, pp. 436–441.

Bishop, C.M., 2006. Pattern Recognition and Machine Learning. (Information Science and Statistics), Springer-Verlag New York, USA.

Bottarelli, L., Bicego, M., Blum, J., Farinelli, A., 2016. Skeleton-based orienteering for level set estimation. In: ECAI 2016 - 22nd European Conference on Artificial Intelligence. pp. 1256–1264.

Bottarelli, L., Bicego, M., Blum, J., Farinelli, A., 2019. Orienteering-based informative path planning for environmental monitoring. Eng. Appl. Artif. Intell. 77, 46–58.

Caliński, T., Harabasz, J., 1974. A dendrite method for cluster analysis. Comm. Statist. Simulation Comput. 3 (1), 1–27.

Castellini, A., Beltrame, G., Bicego, M., Bloisi, D., Blum, J., Denitto, M., Farinelli, A., 2018a. Activity recognition for autonomous water drones based on unsupervised learning methods. In: Proc. 4th Italian Workshop on Artificial Intelligence and Robotics (AI*IA 2017), Vol. 2054, pp. 16–21.

Castellini, A., Beltrame, G., Bicego, M., Blum, J., Denitto, M., Farinelli, A., 2018b. Unsupervised activity recognition for autonomous water drones. In: Proc. Symposium on Applied Computing, SAC 2018. ACM, pp. 840–842.

Castellini, A., Bicego, M., Bloisi, D., Blum, J., Masillo, F., Peignier, S., Farinelli, A., 2019a. Subspace clustering for situation assessment in aquatic drones: A sensitivity analysis for state-model improvement. Cybern. Syst. 50 (8), 658–671.

Castellini, A., Chalkiadakis, G., Farinelli, A., 2019b. Influence of state-variable constraints on partially observable monte carlo planning. In: Proc. 28th International Joint Conference on Artificial Intelligence (IJCAI 2019), pp. 5540–5546.

Castellini, A., Masillo, F., Bicego, M., Bloisi, D., Blum, J., Farinelli, A., Peigner, S., 2019c. Subspace clustering for situation assessment in aquatic drones. In: Proc. Symposium on Applied Computing, SAC 2019. ACM, pp. 930–937.

Castellini, A., Masillo, F., Sartea, R., Farinelli, A., 2019d. Explainable modeling (XM): Data analysis for intelligent agents. In: Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019). IFAAMAS, pp. 2342–2344.

Castellini, A., Paltrinieri, D., Manca, V., 2015. MP-GeneticSynth: inferring biological network regulations from time series. Bioinformatics 31, 785–787.

Chen, L., Hoey, J., Nugent, C.D., Cook, D.J., Yu, Z., 2012. Sensor-based activity recognition. IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) 42 (6), 790–808.

Chiu, B., Keogh, E., Lonardi, S., 2003. Probabilistic discovery of time series motifs. In: Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining. In: KDD '03, ACM, New York, USA, pp. 493–498.

Davies, D.L., Bouldin, D.W., 1979. A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. 1 (2), 224–227.

Dhiman, C., Vishwakarma, D.K., 2019. A review of state-of-the-art techniques for abnormal human activity recognition. Eng. Appl. Artif. Intell. 77, 21–45.

Endsley, M.R., 1995. Toward a theory of situation awareness in dynamic systems. Hum. Factors 37 (1), 32–64.

Farinelli, A., Nardi, D., Pigliacampo, R., Rossi, M., Settembre, G.P., 2012. Cooperative situation assessment in a maritime scenario. Int. J. Intell. Syst. 27 (5), 477–501.

Fox, E.B., Sudderth, E.B., Jordan, M.I., Willsky, A.S., 2008. An HDP-HMM for systems with state persistence. In: Proceedings of the 25th International Conference on Machine Learning. In: ICML '08, ACM, pp. 312–319.

Fu, T.-c., 2011. A review on time series data mining. Eng. Appl. Artif. Intell. 24 (1), 164–181.

Hallac, D., Bhooshan, S., Chen, M., Abida, K., Sosic, R., Leskovec, J., 2018. Drive2Vec: Multiscale state-space embedding of vehicular sensor data. In: Int. Conf. Intelligent Transportation Systems. IEEE, pp. 3233–3238.

Hallac, D., Nystrup, P., Boyd, S., 2016a. Greedy Gaussian segmentation of multivariate time series. Adv. Data Anal. Classif. 13 (3), 727–751.

Hallac, D., Sharang, A., Stahlmann, R., Lamprecht, A., Huber, M., Roehder, M., Sosic, R., Leskovec, J., 2016b. Driver identification using automobile sensor data from a single turn. In: 19th Int. Conf. Intelligent Transportation Systems. IEEE, pp. 953–958.

Hallac, D., Vare, S., Boyd, S., Leskovec, J., 2017. Toeplitz inverse covariance-based clustering of multivariate time series data. In: Proc. 23rd ACM SIGKDD. In: KDD '17, ACM, pp. 215–223.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning. In: Springer Series in Statistics, Springer, New York, USA.

Hong, Y., Kwong, S., Chang, Y., Ren, Q., 2008. Consensus unsupervised feature ranking from multiple views. Pattern Rec. Lett. 29 (5), 595–602.

Kaelbling, L.P., Lozano-Perez, T., 2013. Integrated task and motion planning in belief space. Int. J. Robot. Res. 32 (9–10).

Kelley, L.A., Gardner, S.P., Sutcliffe, M.J., 1996. An automated approach for clustering an ensemble of NMR-derived protein structures into conformationally related subfamilies. Protein Eng., Des. Sel. 9 (11), 1063–1065.

Kim, E., Helal, S., Cook, D., 2010. Human activity recognition and pattern discovery. IEEE Pervasive Comput. 9 (1), 48–53.

Kwon, Y., Kang, K., Bae, C., 2014. Unsupervised learning for human activity recognition using smartphone sensors. Expert Syst. Appl. 41 (14), 6067–6074.

van der Maaten, L., Hinton, G., 2008. Visualizing data using t-SNE. J. Mach. Learn. Res. 9, 2579–2605.

Montanez, G., Amizadeh, S., Laptev, N., 2015. Inertial hidden Markov models: Modeling change in multivariate time series. In: Proc. AAAI Conf. Artificial Intelligence. In: AAAI '15, pp. 911–916.

Moshtaghi, M., Bezdek, J.C., Erfani, S.M., Leckie, C., Bailey, J., 2019. Online cluster validity indices for performance monitoring of streaming data clustering. Int. J. Intell. Syst. 34 (4), 541–563.

Peignier, S., Rigotti, C., Rossi, A., Beslon, G., 2018. Weight-based search to find clusters around medians in subspaces. In: Proceedings of the Symposium on Applied Computing, SAC 2018. ACM, pp. 471–480.

Rabiner, L.R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE 77 (2), 257–286.

Rousseeuw, P.J., 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 20, 53–65.

Russell, S., Norvig, P., 2009. Artificial Intelligence: A Modern Approach, third ed. Prentice Hall Press, Upper Saddle River, NJ, USA.

Trabelsi, D., Mohammed, S., Chamroukhi, F., Oukhellou, L., Amirat, Y., 2013. An unsupervised approach for automatic activity recognition based on hidden Markov model regression. IEEE Trans. Autom. Sci. Eng. 10 (3), 829–835.