Neural Networks 71 (2015) 88–104

Contents lists available at ScienceDirect

Neural Networks

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

Incremental multi-class semi-supervised clustering regularized by Kalman filtering

Siamak Mehrkanoon [*], Oscar Mauricio Agudelo, Johan A.K. Suykens

KU Leuven, ESAT-STADIUS, Kasteelpark Arenberg 10, B-3001 Leuven (Heverlee), Belgium

ARTICLE INFO

Article history:
Received 25 September 2014
Received in revised form 9 July 2015
Accepted 2 August 2015
Available online 14 August 2015

Keywords:
Incremental semi-supervised clustering
Non-stationary data
Video segmentation
Low embedding dimension
Kernel spectral clustering
Kalman filtering

ABSTRACT

This paper introduces an on-line semi-supervised learning algorithm formulated as a regularized kernel spectral clustering (KSC) approach. We consider the case where new data arrive sequentially but only a small fraction of it is labeled. The available labeled data act as prototypes and help to improve the performance of the algorithm to estimate the labels of the unlabeled data points. We adopt a recently proposed multi-class semi-supervised KSC based algorithm (MSS-KSC) and make it applicable for on-line data clustering. Given a few user-labeled data points the initial model is learned and then the class membership of the remaining data points in the current and subsequent time instants are estimated and propagated in an on-line fashion. The update of the memberships is carried out mainly using the out-of-sample extension property of the model. Initially the algorithm is tested on computer-generated data sets, then we show that video segmentation can be cast as a semi-supervised learning problem. Furthermore we show how the tracking capabilities of the Kalman filter can be used to provide the labels of objects in motion and thus regularizing the solution obtained by the MSS-KSC algorithm. In the experiments, we demonstrate the performance of the proposed method on synthetic data sets and real-life videos where the clusters evolve in a smooth fashion over time.

1. Introduction

In many real-life applications, ranging from data mining to machine perception, obtaining the labels of input data is often cumbersome and expensive. Therefore in many cases one encounters a large amount of unlabeled data while the labeled data are rare. Semi-supervised learning (SSL) is a framework in machine learning that aims at learning from both labeled and unlabeled data points (Zhu, 2006). SSL algorithms received a lot of attention in the last years due to rapidly increasing amounts of unlabeled data. Several semi-supervised algorithms have been proposed in the literature (Belkin, Niyogi, & Sindhwani, 2006; Chang, Pao, & Lee, 2012; He, 2004; Mehrkanoon & Suykens, 2012; Wang, Chen, & Zhou, 2012; Xiang, Nie, & Zhang, 2010; Yang et al., 2012). However, most of the SSL algorithms, operate in batch mode, hence requiring a large amount of computation time and memory to handle data streams like the ones found in real-life applications such as voice and face recognition, community detection of evolving networks and object tracking in computer vision. Therefore designing SSL algorithms that can operate in an on-line fashion is necessary for dealing with such data streams.

In the context of on-line clustering, due to the complex underlying dynamics and non-stationary behavior of real-life data, attempts have been made to design adaptive clustering algorithms. For instance, evolutionary spectral clustering based algorithms (Chakrabarti, Kumar, & Tomkins, 2006; Chi, Song, Zhou, Hino, & Tseng, 2007; Ning, Xu, Chi, Gong, & Huang, 2010), incremental K-means (Chakraborty & Nagwani, 2011), self-organizing time map (Sarlin, 2013) and incremental kernel spectral clustering (Langone, Agudelo, De Moor, & Suykens, 2014). However, in all above-mentioned algorithms the side-information (labels) is not incorporated and therefore they might underperform in certain situations.

A semi-supervised incremental clustering algorithm that can exploit the user constraints on data streams is proposed in Halkidi, Spiliopoulou, and Pavlou (2012). The user's prior information are presented to the algorithm in the form of must-link and cannot-link constraints. The authors in Kamiya, Ishii, Furao, and Hasegawa (2007) introduced an on-line semi-supervised algorithm based on a self-organizing incremental neural network.

Here we adopt the recently proposed multi-class semi-supervised kernel spectral clustering (MSS-KSC) algorithm

[*] Corresponding author. Tel.: +32 16328653; fax: +32 16321970.
E-mail addresses: siamak.mehrkanoon@esat.kuleuven.be, mehrkanoon2011@gmail.com (S. Mehrkanoon), mauricio.agudelo@esat.kuleuven.be (O.M. Agudelo), johan.suykens@esat.kuleuven.be (J.A.K. Suykens).

http://dx.doi.org/10.1016/j.neunet.2015.08.001
0893-6080/© 2015 Elsevier Ltd. All rights reserved.

(Mehrkanoon, Alzate, Mall, Langone, & Suykens, 2015) and make it applicable for an on-line data clustering/classification. In MSS-KSC the core model is kernel spectral clustering (KSC) algorithm introduced in Alzate and Suykens (2010). MSS-KSC is a regularized version of KSC which aims at incorporating the information of the labeled data points in the learning process. It has a systematic model selection criterion and the out-of-sample extension property. Moreover, as it has been shown in Mehrkanoon and Suykens (2014), it can scale to large data.

In contrast to the methods described in Adankon, Cheriet, and Biem (2009), Belkin et al. (2006), Chang et al. (2012), Xiang et al. (2010), Yang et al. (2012), in the MSS-KSC approach a purely unsupervised algorithm acts as a core model and the available side information is incorporated via a regularization term. In addition, the method can be applied for both on-line semi-supervised classification and clustering and uses a low-dimensional embedding. In the MSS-KSC approach, one needs to solve a linear system of equations to obtain the model parameters. Therefore with n number of training points, the algorithm has $\mathcal{O}(n^3)$ training complexity with naive implementations. The MSS-KSC model can be trained on a subset of the data (training data points) and then applied to the rest of the data in a learning framework. Thanks to the previously learned model, the out-of-sample extension property of the MSS-KSC model allows the prediction of the membership of a new point. However, in order to cope with non-stationary data-stream one also needs to continuously adjust the initial MSS-KSC model.

To this end, this paper introduces the Incremental MSS-KSC (I-MSS-KSC) algorithm which takes advantage of the available side-information to continuously adapt the initial MSS-KSC model and learn the underlying complex dynamics of the data-stream. The proposed method can be applied in several application domains including video segmentation, complex networks and medical imaging. In particular, in this paper we focus on video segmentation.

There have been some reports in the literature on formulating the object tracking task as a binary classification problem. For instance in Teichman and Thrun (2012) a tracking-based semi-supervised learning algorithm is developed for the classification of objects that have been segmented. The authors in Badrinarayanan, Budvytis, and Cipolla (2013) introduced a tree structured graphical model for video segmentation.

Due to the increasing demands in robotic applications, Kalman filtering has received significant attention. In particular Kalman filter has been applied in wide applications areas such as robot localization, navigation, object tracking and motion control (see Chen, 2012 and references therein). The authors in Suliman, Cruceru, and Moldoveanu (2010) use the Kalman filter for monitoring a contact in a video surveillance sequence. In Zhong and Sclaroff (2003), a Kalman filter based algorithm is presented to segment the foreground objects in video sequences given non-stationary textured background. An adaptive Kalman filter algorithm has been used for video moving object tracking in Weng, Kuo, and Tu (2006).

In case of the video segmentation, we show how Kalman filter can be integrated into the I-MSS-KSC algorithm as a regularizer by providing an estimation of the labels throughout the whole video sequences. This paper is organized as follows. In Section 2, the kernel spectral clustering (KSC) algorithm is briefly reviewed. In Section 3, an overview of the multi-class semi-supervised clustering (MSS-KSC) algorithm is given. The incremental multi-class semi-supervised clustering regularized by Kalman filtering approach is described in Section 4. In Section 5, experimental results are given in order to confirm the validity and applicability of the proposed method. The experimental findings and the demonstrative videos are provided in the supplementary material (see Appendix A) of this paper.[1]

2. Brief overview of KSC

The KSC method corresponds to a weighted kernel PCA formulation providing a natural extension to out-of-sample data i.e. the possibility to apply the trained clustering model to out-of-sample points. Given training data $\mathcal{D} = \{x_i\}_{i=1}^n$, $x_i \in \mathbb{R}^d$, the primal problem of kernel spectral clustering is formulated as follows (Alzate & Suykens, 2010):

$$\begin{aligned} \min_{w^{(\ell)}, b^{(\ell)}, e^{(\ell)}} \quad & \frac{1}{2}\sum_{\ell=1}^{N_c-1} w^{(\ell)^T} w^{(\ell)} - \frac{1}{2n}\sum_{\ell=1}^{N_c-1} \gamma_\ell e^{(\ell)^T} V e^{(\ell)} \\ \text{subject to} \quad & e^{(\ell)} = \Phi w^{(\ell)} + b^{(\ell)} 1_n, \ \ell = 1, \ldots, N_c - 1 \end{aligned} \tag{1}$$

where N_c is the number of desired clusters, $e^{(\ell)} = [e_1^{(\ell)}, \ldots, e_n^{(\ell)}]^T$ are the projected variables and $\ell = 1, \ldots, N_c - 1$ indicates the number of score variables required to encode the N_c clusters. $\gamma_\ell \in \mathbb{R}^+$ are the regularization constants. Here

$$\Phi = [\varphi(x_1), \ldots, \varphi(x_n)]^T \in \mathbb{R}^{n \times h}$$

where $\varphi(\cdot) : \mathbb{R}^d \to \mathbb{R}^h$ is the feature map and h is the dimension of the feature space which can be infinite dimensional. A vector of all ones with size n is denoted by 1_n. $w^{(\ell)}$ is the model parameters vector in the primal. $V = \text{diag}(v_1, \ldots, v_n)$ with $v_i \in \mathbb{R}^+$ is a user defined weighting matrix.

Applying the Karush–Kuhn–Tucker (KKT) optimality conditions one can show that the solution in the dual can be obtained by solving an eigenvalue problem of the following form:

$$V P_v \Omega \alpha^{(\ell)} = \lambda \alpha^{(\ell)}, \tag{2}$$

where $\lambda = n/\gamma_\ell$, $\alpha^{(\ell)}$ are the Lagrange multipliers and P_v is the weighted centering matrix:

$$P_v = I_n - \frac{1}{1_n^T V 1_n} 1_n 1_n^T V,$$

where I_n is the $n \times n$ identity matrix and Ω is the kernel matrix with ij-th entry $\Omega_{ij} = K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$. In the ideal case of N_c well separated clusters, for a properly chosen kernel parameter, the matrix $V P_v \Omega$ has $N_c - 1$ piecewise constant eigenvectors with eigenvalue 1.

The eigenvalue problem (2) is related to spectral clustering with random walk Laplacian. In this case, the clustering problem can be interpreted as finding a partition of the graph in such a way that the random walker remains most of the time in the same cluster with few jumps to other clusters, minimizing the probability of transitions between clusters. It is shown that if

$$V = D^{-1} = \text{diag}\left(\frac{1}{d_1}, \ldots, \frac{1}{d_n}\right),$$

where $d_i = \sum_{j=1}^n K(x_i, x_j)$ is the degree of the ith data point, the dual problem is related to the random walk algorithm for spectral clustering.

From the KKT optimality conditions one can show that the score variables can be written as follows:

$$\begin{aligned} e^{(\ell)} &= \Phi w^{(\ell)} + b^{(\ell)} 1_n = \Phi \Phi^T \alpha^{(\ell)} + b^{(\ell)} 1_n \\ &= \Omega \alpha^{(\ell)} + b^{(\ell)} 1_n, \quad \ell = 1, \ldots, N_c - 1. \end{aligned}$$

The out-of-sample extensions to test points $\{x_i\}_{i=1}^{n_{\text{test}}}$ is done by an Error-Correcting Output Coding (ECOC) decoding scheme. First

[1] ftp://ftp.esat.kuleuven.be/stadius/siamak/155-spt.zip.

the cluster indicators are obtained by binarizing the score variables for test data points as follows:

$$q^{(\ell)}_{\text{test}} = \text{sign}(e^{(\ell)}_{\text{test}}) = \text{sign}(\Phi_{\text{test}} w^{(\ell)} + b^{(\ell)} 1_{n_{\text{test}}})$$
$$= \text{sign}(\Omega_{\text{test}} \alpha^{(\ell)} + b^{(\ell)} 1_{n_{\text{test}}}),$$

where $\Phi_{\text{test}} = [\varphi(x_1), \ldots, \varphi(x_{n_{\text{test}}})]^T$ and $\Omega_{\text{test}} = \Phi_{\text{test}}\Phi^T$. The decoding scheme consists of comparing the cluster indicators obtained in the test stage with the codebook (which is obtained in the training stage) and selecting the nearest codeword in terms of Hamming distance.

3. Semi-supervised clustering using MSS-KSC

Consider training data points

$$\mathcal{D} = \{\underbrace{x_1, \ldots, x_{n_{UL}}}_{\substack{\text{Unlabeled} \\ (\mathcal{D}_U)}}, \underbrace{x_{n_{UL}+1}, \ldots, x_n}_{\substack{\text{Labeled} \\ (\mathcal{D}_L)}}\}, \tag{3}$$

where $\{x_i\}_{i=1}^n \in \mathbb{R}^d$. The first n_{UL} data points do not have labels whereas the last $n_L = n - n_{UL}$ points have been labeled. Assume that there are Q classes ($Q \le N_c$), then the label indicator matrix $Y \in \mathbb{R}^{n_L \times Q}$ is defined as follows:

$$Y_{ij} = \begin{cases} +1 & \text{if the } i\text{th point belongs to the } j\text{th class} \\ -1 & \text{otherwise.} \end{cases} \tag{4}$$

The information of the labeled data is incorporated to the kernel spectral clustering (1) by means of a regularization term. The aim of this term is to minimize the squared distance between the projections of the labeled data and their corresponding labels. The formulation of Multi-class semi-supervised KSC (MSS-KSC) described in Mehrkanoon et al. (2015) in primal is given as follows:

$$\begin{aligned} \min_{w^{(\ell)}, b^{(\ell)}, e^{(\ell)}} \quad & \frac{1}{2}\sum_{\ell=1}^{Q} w^{(\ell)^T} w^{(\ell)} - \frac{\gamma_1}{2}\sum_{\ell=1}^{Q} e^{(\ell)^T} V e^{(\ell)} \\ & + \frac{\gamma_2}{2}\sum_{\ell=1}^{Q} (e^{(\ell)} - c^{(\ell)})^T \tilde{A} (e^{(\ell)} - c^{(\ell)}) \\ \text{subject to} \quad & e^{(\ell)} = \Phi w^{(\ell)} + b^{(\ell)} 1_n, \quad \ell = 1, \ldots, Q, \end{aligned} \tag{5}$$

where c^{ℓ} is the ℓ-th column of the matrix C defined as

$$C = [c^{(1)}, \ldots, c^{(Q)}]_{n \times Q} = \left[\begin{array}{c} 0_{n_{UL} \times Q} \\ \hline Y \end{array}\right]_{n \times Q}, \tag{6}$$

where $0_{n_{UL} \times Q}$ is a zero matrix of size $n_{UL} \times Q$ and Y is defined as previously. The matrix $\tilde{A}$ is defined as follows:

$$\tilde{A} = \left[\begin{array}{c|c} 0_{n_{UL} \times n_{UL}} & 0_{n_{UL} \times n_L} \\ \hline 0_{n_L \times n_{UL}} & I_{n_L \times n_L} \end{array}\right],$$

where $I_{n_L \times n_L}$ is the identity matrix of size $n_L \times n_L$. V is the inverse of the degree matrix defined as previously.

Since in Eq. (5) the feature map φ is not explicitly known, one uses the kernel trick and solves the problem in the dual. The Lagrangian of the constrained optimization problem (5) becomes

$$\begin{aligned} \mathcal{L}(w^{(\ell)}, b^{(\ell)}, e^{(\ell)}, \alpha^{(\ell)}) = {} & \frac{1}{2}\sum_{\ell=1}^{Q} w^{(\ell)^T} w^{(\ell)} - \frac{\gamma_1}{2}\sum_{\ell=1}^{Q} e^{(\ell)^T} V e^{(\ell)} \\ & + \frac{\gamma_2}{2}\sum_{\ell=1}^{Q} (e^{(\ell)} - c^{(\ell)})^T \tilde{A} (e^{(\ell)} - c^{(\ell)}) \\ & + \sum_{\ell=1}^{Q} \alpha^{(\ell)^T} \left(e^{(\ell)} - \Phi w^{(\ell)} - b^{(\ell)} 1_n\right), \end{aligned}$$

where $\alpha^{(\ell)}$ is the vector of Lagrange multipliers. Then the Karush–Kuhn–Tucker (KKT) optimality conditions are as follows,

$$\begin{cases} \dfrac{\partial \mathcal{L}}{\partial w^{(\ell)}} = 0 \to w^{(\ell)} = \Phi^T \alpha^{(\ell)}, \quad \ell = 1, \ldots, Q, \\ \dfrac{\partial \mathcal{L}}{\partial b^{(\ell)}} = 0 \to 1_n^T \alpha^{(\ell)} = 0, \quad \ell = 1, \ldots, Q, \\ \dfrac{\partial \mathcal{L}}{\partial e^{(\ell)}} = 0 \to \alpha^{(\ell)} = (\gamma_1 V - \gamma_2 \tilde{A}) e^{(\ell)} + \gamma_2 c^{(\ell)}, \\ \quad \ell = 1, \ldots, Q, \\ \dfrac{\partial \mathcal{L}}{\partial \alpha^{(\ell)}} = 0 \to e^{(\ell)} = \Phi w^{(\ell)} + b^{(\ell)} 1_n, \quad \ell = 1, \ldots, Q. \end{cases} \tag{7}$$

Elimination of the primal variables $w^{(\ell)}$, $e^{(\ell)}$ and making use of Mercer's Theorem (Vapnik, 1998), results in the following linear system in the dual (Mehrkanoon et al., 2015):

$$\gamma_2 \left(I_n - \frac{R 1_n 1_n^T}{1_n^T R 1_n}\right) c^{(\ell)} = \alpha^{(\ell)} - R\left(I_n - \frac{1_n 1_n^T R}{1_n^T R 1_n}\right) \Omega \alpha^{(\ell)}, \tag{8}$$

where $R = \gamma_1 V - \gamma_2 \tilde{A}$. As is shown in Mehrkanoon et al. (2015), given Q labels the approach is not restricted to finding just Q classes and instead is able to discover up to 2^Q hidden clusters. In addition, it uses low embedding dimension to reveal the existing number of clusters which is important when one deals with large number of clusters. In fact one maps the data points to a Q-dimensional space, which from now on will be referred to as α-space, and the solution vectors $\alpha^{(\ell)}$ ($\ell = 1, \ldots, Q$) represent the embedding of the input data in this space. Therefore every point x_i is associated with the point $[\alpha_i^{(1)}, \ldots, \alpha_i^{(Q)}]$ in the α-space. (space spanned by the solution vector $\alpha^{(\ell)}$).

Once the solution to (8) is found, the codebook $\mathcal{CB} \in \{-1, 1\}^{p \times Q}$ is formed by the unique rows of the binarized solution matrix (i.e. $[\text{sign}(\alpha^{(1)}), \ldots, \text{sign}(\alpha^{(Q)})]$). Here each code-word is a binary word of length Q and represents a cluster. The maximum number of clusters that can be decoded is 2^Q since the maximum value that p can take is 2^Q.

In the MSS-KSC formulation, the clusters in the projection space (e-space) obtained by $e^{(\ell)}$ form lines with well-tuned RBF kernel parameters. Whereas the projection of the points in the α-space obtained by $\alpha^{(\ell)}$ show a localized behavior. (See Mehrkanoon et al., 2015 for more details.) For the sake of clarity we illustrate the projected points in both α and e-spaces, in the case of a synthetic two moons data set in Fig. 1.

The MSS-KSC algorithm (Mehrkanoon et al., 2015) is summarized in Algorithm 1.

4. Incremental multi-class semi-supervised clustering

It has been shown in Mehrkanoon et al. (2015) that for the MSS-KSC approach, one has to solve a linear system of size n (number of training data points) in the dual to obtain the cluster membership of the data points. This is fine for batch mode but does not fit practical applications such as on-line semi-supervised clustering, in which the data are entered sequentially. If the distribution of the new arriving data points is not in line with the one of the training points, then the trained model cannot explain well the new distribution. Therefore in those cases an adaptive learning mechanism is required. In what follows we will show how one can use the out-of-sample extension property of the MSS-KSC model for dealing with data streams in an on-line fashion.

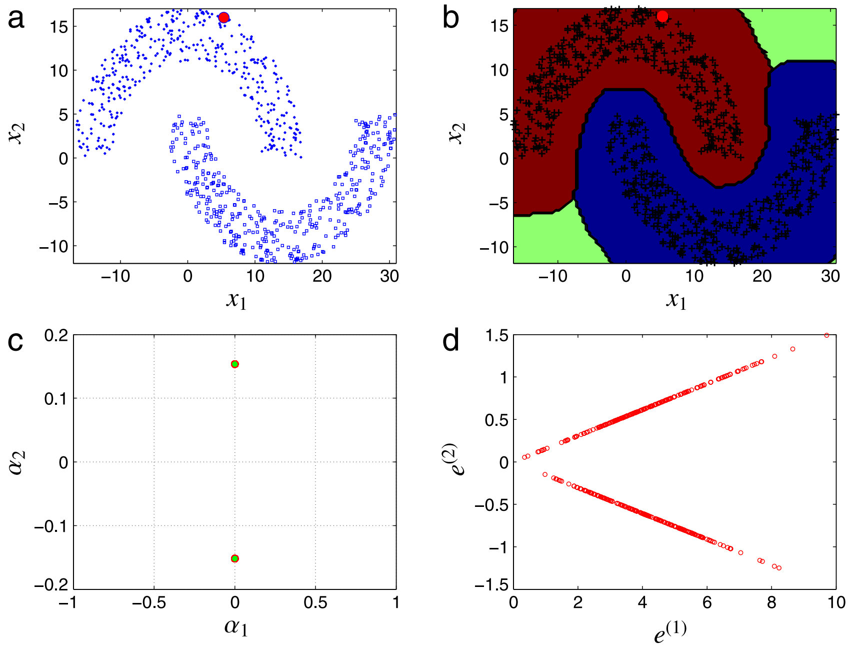

Fig. 1. Two moons data set: The labeled data point of only one class is available and is depicted by the red circle (●). (a): Data points in the original space. (b): The result of MSS-KSC algorithm with RBF kernel. (c): The mapped data points in the α space. (d): The mapped data points in the e space. (For the colored figure, the reader is referred to the web version of this article.)

Algorithm 1: MSS-KSC: Semi-supervised clustering (Mehrkanoon et al., 2015)

```
Input: Training data set $\mathcal{D}$, labels $Y$, the tuning parameters
       $\{\gamma_i\}_{i=1}^2$, the kernel parameter (if any), number of
       clusters $N_c$, the test set $\mathcal{D}^{test} = \{x_i^{test}\}_{i=1}^{n_{test}}$ and number
       of available class labels i.e. $Q$
Output: Cluster membership of test data points $\mathcal{D}^{test}$

1 Solve the dual linear system (8) to obtain $\{\alpha^\ell\}_{\ell=1}^Q$ and
  compute the bias term $\{b^\ell\}_{\ell=1}^Q$.
2 Binarize the solution matrix
  $S_\alpha = [\text{sign}(\alpha^{(1)}), \dots, \text{sign}(\alpha^{(Q)})]_{n\times Q}$, where
  $\alpha^\ell = [\alpha_1^\ell, \dots, \alpha_n^\ell]^T$.
3 Form the codebook $\mathcal{CB} = \{c_q\}_{q=1}^p$, where $c_q \in \{-1, 1\}^Q$,
  using the $N_c$ most frequently occurring encodings from
  unique rows of solution matrix $S_\alpha$.
4 Estimate the test data projections $\{e_{test}^{(\ell)}\}_{\ell=1}^Q$ using (9).
5 Binarize the test projections and form the encoding matrix
  $[\text{sign}(e_{test}^{(1)}), \dots, \text{sign}(e_{test}^{(Q)})]_{n_{test}\times Q}$ for the test points (Here
  $e_{test}^{(\ell)} = [e_{test,1}^{(\ell)}, \dots, e_{test,n_{test}}^{(\ell)}]^T$).
6 $\forall i$, assign $x_i^{test}$ to class/cluster $q^*$, where
  $q^* = \text{argmin}_q\, d_H(e_{test,i}^{(\ell)}, c_q)$ and $d_H(\cdot,\cdot)$ is the Hamming
  distance.
```

4.1. Out-of-sample solution vector

In the batch MSS-KSC algorithm (Mehrkanoon et al., 2015), the cluster membership of new and unseen test points $\mathcal{D}^{test} = \{x_i\}_{i=1}^{n_{test}}$ is done by an Error-Correcting Output Coding (ECOC) decoding scheme. First the cluster indicators are obtained by binarizing the score variables for test data points as follows:

$$\begin{aligned} q_{test}^{(\ell)} &= \text{sign}(e_{test}^{(\ell)}) = \text{sign}(\Phi_{test} w^{(\ell)} + b^{(\ell)} 1_{n_{test}}) \\ &= \text{sign}(\Omega_{test}\alpha^{(\ell)} + b^{(\ell)} 1_{n_{test}}), \quad \ell = 1, \dots, Q, \end{aligned}$$

where $\Phi_{test} = [\varphi(x_1), \dots, \varphi(x_{n_{test}})]^T$ and $\Omega_{test} = \Phi_{test}\Phi^T \in \mathbb{R}^{n_{test}\times n}$, ($n$ is the number of training points). The decoding scheme consists of comparing the cluster indicators obtained in the test stage with the codebook $\mathcal{CB}$ (which is obtained in the training stage) and selecting the nearest codeword in terms of Hamming distance.

For an on-line fashion, once the model is built using the training data points, one can use the above procedure to estimate the cluster membership of the new test points. But in order for the model to be able to track the non-stationary changes in the data stream, the initial codebook $\mathcal{CB}$ should be adapted on-line so that it has the information of the more recent data points.

In addition one has to incrementally update the solution vectors α. Since in the MSS-KSC approach one needs to solve a linear system of equations, it is possible to use for instance the Sherman–Morrison–Woodbury formula (Golub & Van Loan, 2012) to efficiently update the inverse of the coefficient matrix whenever a new data point is arrived without explicitly computing the matrix inverse. In this case, also one should use some decremental algorithm to cope with non-stationary data stream.

Here we aim at using the out-of-sample extension capability of the MSS-KSC model. Consider n_{test} new data points, $\mathcal{D}^{test} = \{x_i\}_{i=1}^{n_{test}}$. The score variables are:

$$\begin{aligned} &e_{test}^{(\ell)} = \Phi_{test} w^{(\ell)} + b^{(\ell)} 1_{n_{test}} = \Omega_{test}\alpha^{(\ell)} + b^{(\ell)} 1_{n_{test}}, \\ &\quad \ell = 1, \dots, Q, \end{aligned} \tag{9}$$

where Φ_{test} and Ω_{test} are defined as previously. The third KKT condition in (7),

$$\alpha^{(\ell)} = (\gamma_1 V - \gamma_2 A)e^{(\ell)} + \gamma_2 c^{(\ell)}, \quad \ell = 1, \dots, Q,$$

links the score variables for training, i.e. e, to the solution vector α. The idea now is to extend this link to out-of-sample projections, such that we obtain an out-of-sample solution with localized properties. The out-of-sample solution vector $\alpha^{(\ell)}$, $\ell = 1, \dots, Q$ for the new test data points are then defined as follows:

$$\hat{\alpha}_{test}^{(\ell)} \triangleq (\gamma_1 V_{test} - \gamma_2 A_{test})e_{test}^{(\ell)} + \gamma_2 c_{test}^{(\ell)}, \quad \ell = 1, \dots, Q, \tag{10}$$

where c_{test} consists of label information of some data points. $V_{test} = D_{test}^{-1} = \text{diag}(\frac{1}{d_1}, \dots, \frac{1}{d_{n_{test}}})$ is the inverse degree matrix for the test data points. If there is no label available, one can simply estimate the solution vector by setting c_{test} and A_{test} equal zero. In case that the test data set is sampled from the same distribution as the training data points, then the approximated out-of-sample solution vector $\hat{\alpha}_{\text{test}}$, from Eq. (10), will display localized cluster structures. Thus we have embed the data points $x_i \in \mathbb{R}^d$ into the Q-dimensional Euclidean space called α-space, i.e.

$$x_i \to \alpha_i := (\alpha_i^{(1)}, \ldots, \alpha_i^{(Q)}), \quad \forall i = 1, \ldots, n_{\text{test}}.$$

In the case of well separated clusters, the data points that lie in the same cluster in the original space, are all mapped to one point in α-space. But in practical applications where clusters are not well separated, the data points in the same cluster in the input space will be close to each other in the α-space with respect to the other points in different clusters. Using this localized representation for out-of-sample solutions in α-space it is possible to introduce the representative or conceptual centroid of a cluster in this space.

From now on, we use two spaces: the original space $\mathcal{X}$ where the data point x_i lies and the α-space where the embedded solution vector α_i lies. Before starting to introduce the on-line semi-supervised clustering algorithm, let us introduce some definitions that will be used in the remaining of the papers.

Definition 1. The representative or conceptual centroid of the ith cluster $\mathcal{A}_i$ in the $\mathcal{X}$-space, is defined as the mean value of the data points in $\mathcal{A}_i$. We denote the cluster representative in the $\mathcal{X}$-space by $rep_{\mathcal{X}}(\mathcal{A}_i)$.

Definition 2. The representative or conceptual centroid of the ith cluster $\mathcal{A}_i$ in the α-space, is defined as the mean value of the embedded solution vector $\alpha_{k\in\mathcal{J}}$, ($\mathcal{J} = \{j \mid x_j \in \mathcal{A}_i\}$), across all dimensions of the features. We denote the cluster representative in the α-space by $rep_{\alpha}(\mathcal{A}_i)$.

Definition 3. A prototype is defined as a point in the $\mathcal{X}$-space or α-space that has been labeled. The jth prototype is denoted by $prot_{\mathcal{X},j}$ and $prot_{\alpha,j}$ in $\mathcal{X}$-space and α-space respectively.

Definition 4. Assume that the cluster representatives $rep_{\mathcal{X}}(\mathcal{A}_i(k))$ at time step k are obtained. A new set of data points $\mathcal{D}^{(k+1)}$ at time step $k+1$ are defined as outliers or in other words they form a new cluster if their kernel evaluations with respect to all training data points are very close to zero. Therefore $x_* \in \mathcal{D}^{(k+1)}$ is considered as outlier if $\sum_{i=1}^{n_{tr}} K(x_*, x_i)^2 < \theta_0$ where θ_0 is a user defined threshold. Furthermore if there is no single data point and prototype assigned to the ith cluster $\mathcal{A}_i$ then this cluster is eliminated.

In what follows, the on-line semi-supervised algorithm will be described. The proposed on-line multi-class semi-supervised clustering consists of two stages. In the first stage, one trains the MSS-KSC algorithm (Mehrkanoon et al., 2015) using n training data points $\mathcal{D}$ (that contains both label and unlabeled data points) to obtain the initial solution vectors α_i and the cluster memberships. Assuming that N_c clusters are detected, the initial cluster representative $rep_{\mathcal{X}}(\mathcal{A}_i)$ and $rep_{\alpha}(\mathcal{A}_i)$ are then obtained using Definitions 1 and 2. The aim of the second stage is to predict the membership of the new arriving data points using the updated solution vectors α_i. When batch of new data points are arrived the out-of-sample extension properties of the MSS-KSC algorithm is used to approximate the score variables associated with the new points. Next steps composed of the estimation of the projection of the points in the α-space using (10) and calculating the membership of the points. Finally the cluster representatives in both α and $\mathcal{X}$-spaces are updated. (step 13 in Algorithm 2).

Remark 1. If the algorithm is initialized poorly (the first stage), then one cannot expect to have a good clustering performance for the on-line stage (the second stage). The good initialization can be achieved by the aid of user labels and well tuned model parameters. The performance of the initialization can be monitored by checking the value of an internal quality index such as Silhouette, Fisher and Davies–Bouldin indices.

Remark 2. The data points that are to be operated can arrive either one-by-one or as a batch of new points. In the proposed I-MSS-KSC algorithm when a batch of new data points arrives at time step k, more than one cluster can be detected without the need of using any extra step (such as applying K-means in the projection space). Given Q cluster representatives at time instant $k-1$, the total number of new clusters that can be created at time step k is Q. The binarized projections of the outlier points in the α-space is used as an indicator for the number of new clusters at time instance k. In the case of sequential one-by-one case since at time instance k, only one sample is fed to the algorithm, there will be a possibility of creation of at most one cluster.

The proposed on-line semi-supervised clustering algorithm is summarized in Algorithm 2.[2]The general stages of the I-MSS-KSC approach are described by the flow-chart in Fig. 2.

In Algorithm 2, the data-stream might already have some labeled samples which then can be considered as prototypes. Otherwise, depending on the application, the prototypes can be provided by the user or for instance, for a video segmentation task the prototypes of the objects in motion can be estimated by means of a Kalman filter.

4.2. Computational complexity

The computational complexity of the proposed I-MSS-KSC (Algorithm 2) consists of two parts. In the first stage of the algorithm the MSS-KSC is employed to obtain the initial clusters representatives. As in MSS-KSC one needs to solve a linear system of size $n \times n$, therefore the algorithm has $\mathcal{O}(n^3)$ training complexity with naive implementations.

In the second stage which corresponds to updating the clusters representatives for the arriving data-stream, mainly computing the kernel matrix, score variables and out-of-sample solutions vectors contribute to the complexity of the algorithm. As in the second stage, the number of training points is $n_{\text{tr}} = N_c$ (see step 6 of Algorithm 2), the overall complexity of the second stage of Algorithm 2, neglecting lower order terms, is $\mathcal{O}(n_{\text{points}} \times d \times n_{\text{tr}})$ with $n_{\text{tr}} \ll n_{\text{points}}$ and $d \ll n_{\text{points}}$. Therefore the complexity of the on-line algorithm is linear with respect to the number of data-points (n_{points}) at each time instant.

4.3. Regularizing I-MSS-KSC via Kalman filtering

The Kalman filter, also known as Linear Quadratic Estimator (LQE), is an algorithm that provides an efficient computational (recursive) means to estimate the state of a linear dynamical system from noisy measurements, in a way that the variance of the estimation error is minimized.

The Kalman filter was introduced in the sixties by Kalman (1960), and it has been successfully applied to the guidance, navigation and control of vehicles, particularly aircraft and spacecraft. In computer vision, the Kalman filter has been extensively used for tracking objects, and it is precisely in this context that we apply this tool in order to generate the labels for

[2] $\mathcal{A}_i(k)$ is the ith cluster at time k.

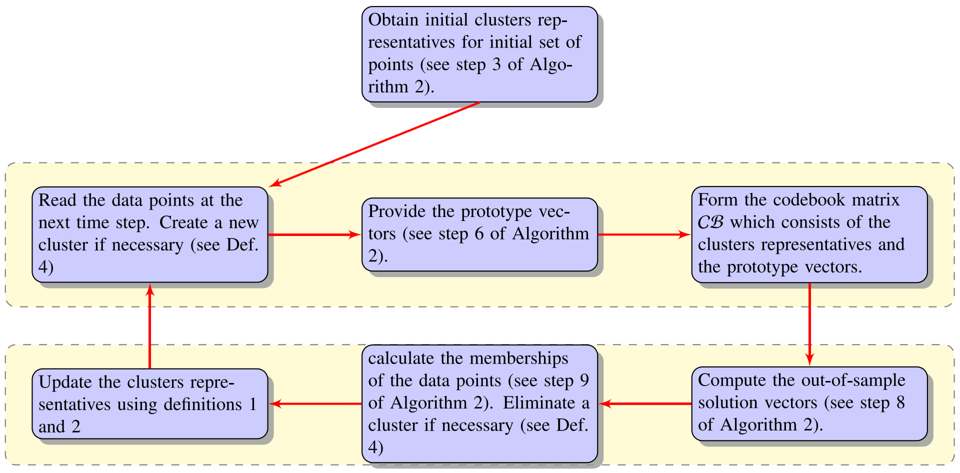
Obtain initial clusters representatives for initial set of points (see step 3 of Algorithm 2).
Read the data points at the next time step. Create a new cluster if necessary (see Def. 4)
Provide the prototype vectors (see step 6 of Algorithm 2).
Form the codebook matrix $\mathcal{CB}$ which consists of the clusters representatives and the prototype vectors.
Compute the out-of-sample solution vectors (see step 8 of Algorithm 2).
calculate the memberships of the data points (see step 9 of Algorithm 2). Eliminate a cluster if necessary (see Def. 4)
Update the clusters representatives using definitions 1 and 2

Fig. 2. Flow-chart of the incremental multi-class semi-supervised kernel spectral clustering (I-MSS-KSC) algorithm.

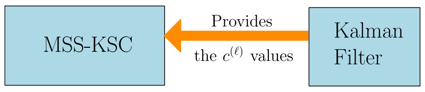

Fig. 3. Kalman filter acts as a regularizer for the MSS-KSC algorithm.

objects in motion. As it can be seen in Eq. (5) the third term of the cost function is influenced by the labels $c^{(\ell)}$ which are provided by either the user or a Kalman filter. Therefore the Kalman filter is reqularizing the solution of the MSS-KSC through $c^{(\ell)}$ values associated with the pixels of the objects in motion in a given video sequence. (See the conceptual diagram in Fig. 3.)

Consider the following discrete-time linear state-space model of a given dynamical system,

$$x(k+1) = Ax(k) + Bu(k) + Gw(k) \qquad (11)$$
$$y(k) = Cx(k) + v(k)$$

where $x(k) \in \mathbb{R}^{n_x}$, $u(k) \in \mathbb{R}^{n_u}$ and $y(k) \in \mathbb{R}^{n_y}$ are the state, input and output vectors respectively, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$ and $C \in \mathbb{R}^{n_y \times n_x}$ are the matrices defining the system dynamics, $G \in \mathbb{R}^{n_x \times n_w}$ is a weighting matrix and $w(k) \in \mathbb{R}^{n_w}$ and $v(k) \in \mathbb{R}^{n_y}$ are random variables that represent the process (model uncertainties) and measurement (measurement uncertainties) noises respectively. The process noise $w(k)$ is modeled as a Gaussian white noise with zero mean and covariance matrix $Q \in \mathbb{R}^{n_w \times n_w}$ and the measurement noise $v(k)$ is modeled as a Gaussian white noise with zero mean and covariance matrix $R \in \mathbb{R}^{n_y \times n_y}$.

Notice that for control and object tracking purposes, it is necessary to know the state vector $x(k)$. However, in general, this vector is not always available. Therefore the use of an estimator such as the Kalman filter becomes necessary in order to provide an estimate of $x(k)$ from the inputs and outputs of the system, on the basis of a mathematical model. The estimate of the state vector $x(k)$ will be denoted by $\hat{x}(k)$. For the derivation of the Kalman filter equations, readers are referred to Barrero (2005) and Franklin, Powell, and Workman (1990). The Kalman filter is summarized in Algorithm 3,[3] where $P^{\mathrm{f}}(k)$ is the prior error covariance matrix, $\hat{x}(k)$ is the estimate of $x(k)$, $P(k)$ is the estimation error covariance matrix and $y(k)$ is a vector comprising the measurements.

In this work, we use some image processing techniques to roughly determine the position (measurement) of a moving object for which we would like to provide a label, and afterwards we further improve this position estimate by using a Kalman filter. We use the following kinetic model to describe the object motion:

$$s_x(k) = s_x(k-1) + Tv_x(k-1) + \frac{T^2}{2}a_x(k-1) \qquad (12)$$
$$v_x(k) = v_x(k-1) + Ta_x(k-1)$$
$$s_y(k) = s_y(k-1) + Tv_y(k-1) + \frac{T^2}{2}a_y(k-1)$$
$$v_y(k) = v_y(k-1) + Ta_y(k-1)$$

where T is the sampling time, $s_x(k)$, $v_x(k)$ and $a_x(k)$ are the position, velocity and acceleration of the object in the x-coordinate, and $s_y(k)$, $v_y(k)$ and $a_y(k)$ are the position, velocity and acceleration of the object in the y-coordinate. If we define the state vector as $x(k) = [s_x(k), s_y(k), v_x(k), v_y(k)]^T$, we can write down the kinematic model in a state-space form as follows:

$$x(k+1) = Ax(k) + Ga(k) \qquad (13)$$
$$y(k) = Cx(k) + v(k)$$

where

$$A = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad G = \begin{bmatrix} T^2/2 & 0 \\ 0 & T^2/2 \\ T & 0 \\ 0 & T \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

and $a = [a_x(k), a_y(k)]^T$. Here it is assumed that $a_x(k)$ and $a_y(k)$ are normally distributed, with zero mean and standard deviations σ_{a_x} and σ_{a_y} respectively. Observe that there is no $Bu(k)$ term in the previous equations given that there are no control inputs. Finally, the covariance matrices of the process and measurement noise are defined as follows:

$$Q = \begin{bmatrix} \sigma_{a_x}^2 & 0 \\ 0 & \sigma_{a_y}^2 \end{bmatrix}, \qquad R = \begin{bmatrix} \sigma_{m_x}^2 & 0 \\ 0 & \sigma_{m_y}^2 \end{bmatrix},$$

where σ_{m_x} and σ_{m_y} are the standard deviations of the measured position of the object in the x and y coordinates respectively. These measurements are generated by using some basic image processing techniques (object detection based on color, binarization, computation of centroids, etc.). The interaction between Kalman filter and I-MSS-KSC algorithm is shown in Fig. 4.

A video sequence consists of several frames see Fig. 5 and each frame will be treated as batch of new data points for the algorithm.

[3] Here index k denotes the kth frame.

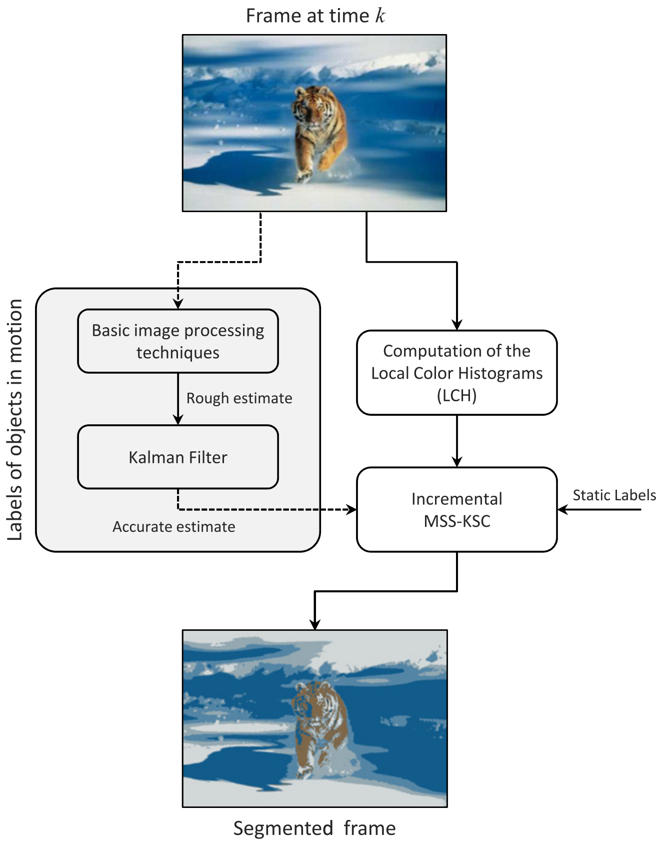

Fig. 4. Diagram showing the interaction between Kalman filter and the I-MSS-KSC algorithm for video segmentation purposes. (For the colored figure, the reader is referred to the web version of this article.)

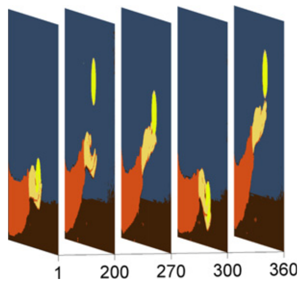

Fig. 5. Some of the frames of the second video sequence. Each slice is treated as a batch of new data points that are fed to the algorithm.

Algorithm 2: I-MSS-KSC: On-line Semi-supervised clustering

```
Input: Training data set $\mathcal{D}$, labels $Y$, the tuning parameters $\{\gamma_i\}_{i=1}^2$, the kernel parameter (if any), number of clusters $N_c$, number of prototypes $p$ and number of available class labels i.e. $Q$
Output: Cluster membership of test data points

First stage: INITIALIZATION OF CLUSTERS REPRESENTATIVES.
1  Read the training data points (initial set of points, $k$=1).
2  Train the MSS-KSC model using Algorithm 1 and obtain the cluster membership of the training data points.
3  Calculate the initial cluster representative $rep_{\mathcal{X}}(\mathcal{A}_i)$ and $rep_{\alpha}(\mathcal{A}_i)$ for $i = 1, \cdots, N_c$ using Definition 1 and 2.

Second stage: UPDATING THE CLUSTERS REPRESENTATIVES
for k=2 to the end of the data-stream do
4      Read the set of data points ($n_{\text{points}}$) at time $k$, $x_i(k), i = 1, \ldots, n_{\text{points}}$.
5      Detect the indices of the outlier points according to Def. 4.
6      Provide the prototypes ($prot_{\alpha,j}(k), j = 1, \ldots, p$) and form the codebook matrix $\mathcal{CB}$ for the current time instant $k$:
       $\mathcal{CB} = \left[ rep_{\alpha}(\mathcal{A}_i(k))\Big|_{i=1,\ldots,N_c}, prot_{\alpha,j}(k)\Big|_{j=1,\ldots,p} \right]^T \in \mathbb{R}^{(N_c+p)\times Q}$.
7      Employ the $\left[ rep_{\mathcal{X}}(\mathcal{A}_i(k))\Big|_{i=1,\ldots,N_c} \right]$ as training points and calculate the score variables $e_i^{\ell}(k), i = 1, \ldots, n_{\text{points}}$ for $\ell = 1, \ldots, Q$ using (9).
8      Compute the out-of-sample solution vectors $\alpha_i(k), i = 1, \ldots, n_{\text{points}}$ using (10).
9      Form the encoding matrix for the outlier points by binarizing the obtained $\alpha_i(k)$, for all $i$ belonging to the set of outlier indices. .
10     The unique rows of the encoding matrix obtained in step 9, indicates the number of new clusters at time step $k$.
11     For non-outlier points, assign $x_i(k)$ to cluster $q^*$, where $q^* = \text{argmin}_j\, d_{Euc}(\alpha_i(k), \mathcal{CB}(j,:))$. Here $d_{Euc}(\cdot,\cdot)$ is the Euclidean distance and the $j$th row of the matrix $\mathcal{CB}$ is denoted by $\mathcal{CB}(j,:)$.
12     Eliminate a cluster if necessary according to Def. 4.
13     Update the cluster representative $rep_{\mathcal{X}}(\mathcal{A}_i(k))$ and $rep_{\alpha}(\mathcal{A}_i(k))$ according to the Definition 1 and 2.
```

5. Experimental results

In this section, some experimental results are presented to illustrate the applicability of the proposed I-MSS-KSC algorithm. In the implementation of Algorithm 2, there are two possibilities:

- I-MSS-KSC $(-)$: the labels (prototypes) are only provided in the first stage, i.e. just for obtaining the initial cluster representatives and the subsequent set of data points do not have any label information.
- I-MSS-KSC $(+)$: the user can also provide the labels (prototypes) for some of the subsequent set of data points.

In order to illustrate the effect of prototypes (labels), we start with synthetic problems and show the differences between the obtained results when I-MSS-KSC$(+)$ and I-MSS-KSC$(-)$ are applied (see Figs. 6 and 8). Next we show the application of I-MSS-KSC reqularized by a Kalman filter to video segmentation. We used RBF kernels for all experiments unless otherwise noted.

It should be noted that at new time step k, the algorithm can receive either batch of data points or one data point. We first analyze the case that batch of new data points are fed to the algorithm at each time instant.

5.1. Synthetic data sets

In Fig. 6, there is a cloud of points which can be clustered in three groups (red, blue and green). The red and green clusters are static over time, whereas the blue cluster is moving toward the other two clusters and then it returns to its initial position.

Fig. 6, shows the snapshots of the evolution at specific time instants where one can see the impact of having prototypes in the incremental semi-supervised clustering. At time instants $k = 11$ and 12, where the blue cluster is close to the other two clusters, there are some points that are not correctly clustered using the I-MSS-KSC$(-)$ algorithm. On the other hand I-MSS-KSC$(+)$ that uses the prototypes (shown by small-squares in Fig. 6) is able

Algorithm 3: Kalman filter

```
    INITIALIZATION.
1   Provide the initial guess for state vector $\hat{x}(0)$ and the
    estimation error covariance matrix $P(0)$.
    for k=1 to end do
        TIME UPDATE (PREDICTION)
2       Propagate the state vector $\hat{x}(k-1)$ one-step ahead,
            $\hat{x}^{\mathrm{f}}(k) = A\hat{x}(k-1) + Bu(k-1)$
3       Propagate the covariance matrix $P(k-1)$ one-step ahead,
            $P^{\mathrm{f}}(k) = AP(k-1)A^T + GQG^T$
        MEASUREMENT UPDATE (CORRECTION)
4       Compute the Kalman gain,
            $L(k) = P^{\mathrm{f}}(k)C^T \left(CP^{\mathrm{f}}(k)C^T + R\right)^{-1}$
5       Update $\hat{x}^{\mathrm{f}}(k)$ to $\hat{x}(k)$ by using the measurements $y(k)$,
            $\hat{x}(k) = \hat{x}^{\mathrm{f}}(k) + L(k)\left(y(k) - C\hat{x}^{\mathrm{f}}(k)\right)$
6       Update $P^{\mathrm{f}}(k)$ to $P(k)$,
            $P(k) = (I - L(k)C)\,P^{\mathrm{f}}(k)$
```

to cluster all the data points correctly. Hence incorporating the prototypes helps to improve the performance. In order to evaluate the performance of the two I-MSS-KSC(−) and I-MSS-KSC(+) algorithms quantitatively, the adjusted rand index (ARI) (Halkidi, Batistakis, & Vazirgiannis, 2001) is used and the obtained results are tabulated in Table 1. ARI is an external evaluation criterion which measures the agreement between two partitions and takes values between zero and one. The higher the value of the ARI the better the clustering result is. In this example, at new time step k, the algorithm receives batch of data where the number of data points is the same as that of time step $k-1$. Initially at time step $k = 1$, there are 1191 data points forming three clusters. The total number of labeled data points is 21 and is fixed along all the time steps. The regularization parameters and the kernel bandwidth are $\gamma_1 = 1, \gamma_2 = 10^{-3}$ and $\sigma = 0.7$ respectively.

The proposed I-MSS-KSC algorithm is able to detect the creation of more than one new cluster at the given time step k, when batch of new data are fed to the algorithm. In the next example[4], we consider the case that three new clusters are created and eliminated at different time steps. At time step 1, the data set consists of three clusters as in the previous example (see Fig. 7). Three other new clusters (clusters 4, 5 and 6) are created at time step 2. The cluster 4 and 5 are eliminated at time step 10 whereas cluster 6 disappears at time step 12. Definition 4 is used along with the Algorithm 2 and all the above mentioned events are correctly detected. Fig. 7, shows the snapshots of the evolution at specific time instants where clusters are detected and eliminated. A video of this simulation is provided in the supplementary material (see Appendix A) of the paper. The number of data points at time step $k = 1$ is 1171. In the next step 1371 new data points that form six clusters are fed to the algorithm. This number of data points is fixed until time step $k = 10$ where two clusters are eliminated and therefore the total number of points is 1241 and finally at time step $k = 12$ another cluster disappears from this step onward the number of data points fed into the algorithm at each step is 1171. The model parameters are $\gamma_1 = 1$, $\gamma_2 = 1$ and $\sigma = 1$ respectively.

Table 1
Averaged ARI index over time for the synthetic data points and time-series.

Experiment	I-MSS-KSC(−)	I-MSS-KSC(+)
Synthetic data points	0.992	**0.999**
Synthetic time-series	0.624	**0.998**

5.2. Synthetic time-series

We show the applicability of the proposed I-MSS-KSC algorithm for on-line time-series clustering. The idea is to cluster signals with similar *fundamental frequencies* using a sliding window approach. Therefore we have generated two groups of signals with length 600 (each group contains 18 signals) with fundamental frequencies 0.1 rad/s and 0.3 rad/s respectively. Then from time instant $k = 200$ till $k = 400$, some of the pure signals of the first group are contaminated with noise which has the same fundamental frequency as the other group. The ground-truth of the time-series are shown in Fig. 8. For I-MSS-KSC, we have labeled one of the pure signals and a contaminated one from the first group. The proposed I-MSS-KSC with and without labels has been applied to cluster the given time-series using a moving window approach. In this experiment the window size was set to 150. To evaluate the outcomes of the model, the average adjusted rand index (ARI) (Halkidi et al., 2001) is used and the results are reported in Table 1.

Here the similarity between the time-series is computed using the RBF kernel with the correlation distance (Warren Liao, 2005). The obtained clustering results are compared with the known ground-truth. The snapshots of the obtained results at certain time instants, where the signals from the first group have noise, are depicted in Fig. 9, which shows the advantage of having labels. From Fig. 9, one can observe that when the labels are not provided to the algorithm, it mixes things up, some of the signals from the first group are assigned to the second group and vice versa. However when the prototypes are used by the algorithm, this pattern is not observed.

5.3. Real-life video segmentation

In this section the proposed I-MSS-KSC algorithm is tested on real-life videos[5]. We compare the performance of the proposed method with a semi-supervised incremental clustering algorithm (SemiStream) (Halkidi et al., 2012), incremental K-means (IKM) (Chakraborty & Nagwani, 2011) and the Efficient Hierarchical Graph-Based Video Segmentation (EHGB) algorithm proposed in Grundmann, Kwatra, Han, and Essa (2010).

The approach described in Halkidi et al. (2012) is an incremental clustering method that exploits the user constraints on data streams in form of must-link and cannot-link constraints. In our experiments, this algorithm is initialized by the MSS-KSC approach (Mehrkanoon et al., 2015). Given the number of constraints we worked with (around 700), it is difficult to evaluate qualitatively the segmentation results when the constraints are displayed. Therefore they are omitted in Figs. 10–13.

[4] The data set can be found in https://sites.google.com/site/smkmhr/Publications.

[5] The data set can be found in https://sites.google.com/site/smkmhr/Publications.

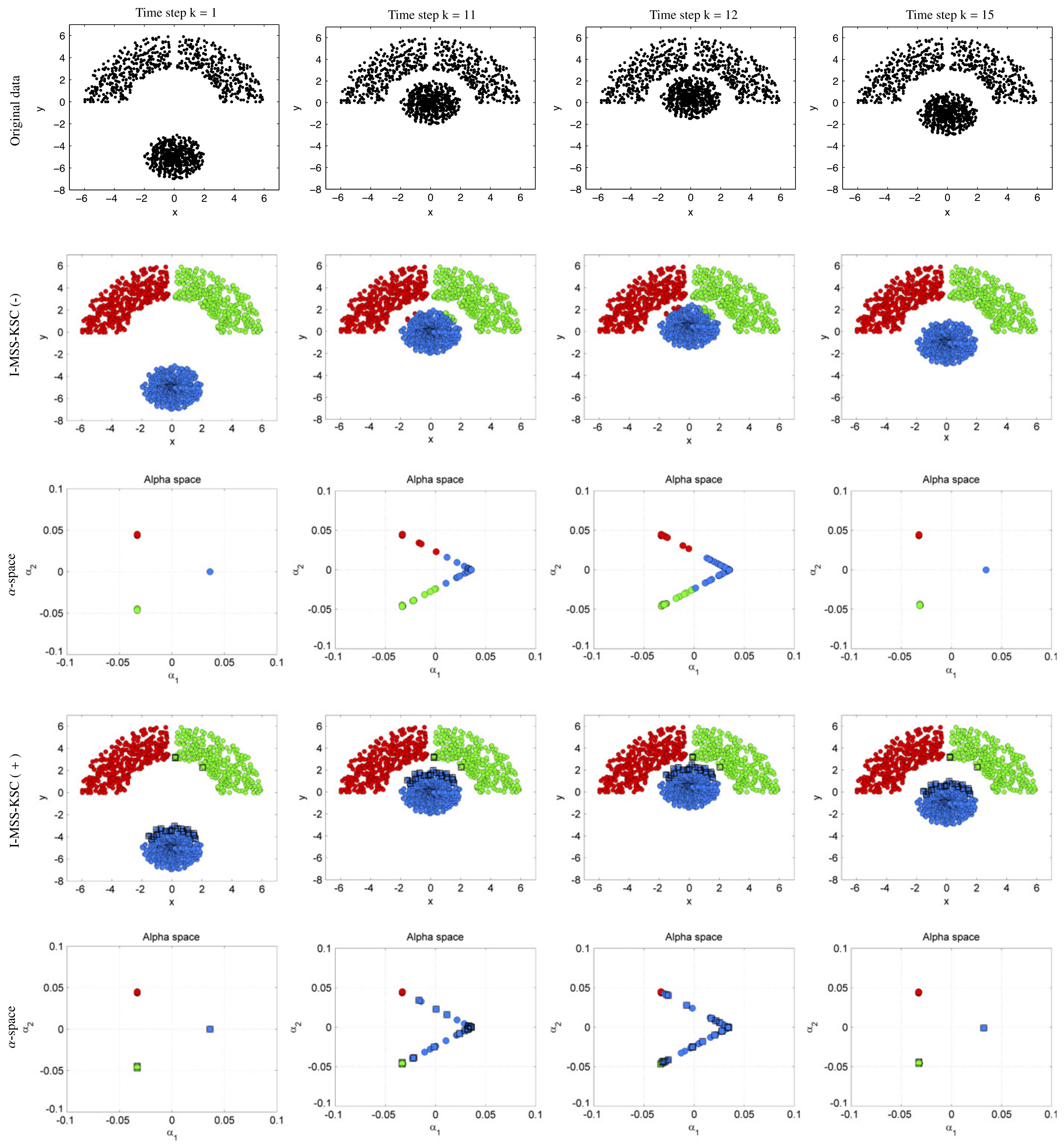

Fig. 6. Synthetic data sets. On-line semi-supervised clustering using the proposed I-MSS-KSC approach implemented in two modes with and without prototypes (i.e. I-MSS-KSC(−) and I-MSS-KSC(+)). *First row*: The original data points at different time steps. *Second row*: I-MSS-KSC(−): The results obtained by I-MSS-KSC algorithm without the help of any prototypes after the initialization. *Third row*: The embedded solution vector α when I-MSS-KSC(−) is applied. *Fourth row*: I-MSS-KSC(+): The results obtained by the proposed I-MSS-KSC algorithm with the help of prototypes. *Fifth row*: The embedded solution vector α when I-MSS-KSC(+) is applied. (For the colored figure, the reader is referred to the web version of this article.)

K-means is one of the most popular data clustering methods due to its simplicity and computational efficiency. It works by selecting some random initial centers and then iteratively adjusting the centers such that the total within cluster variance is minimized. In its incremental variant (Incremental K-means), at each time-step it uses the previous centroids to find the new cluster centers, thus avoiding to rerun the K-means algorithm from scratch (Chakraborty & Nagwani, 2011).

The EHGB algorithm is an efficient and scalable technique for spatio-temporal segmentation of long video sequences using a hierarchical graph-based algorithm. The algorithm begins with oversegmenting a volumetric video graph into space–time regions grouped by appearance. Then a "region graph" over the obtained segmentation is constructed and this process is repeated over multiple levels to create a tree of spatio-temporal segmentations (Grundmann et al., 2010). This algorithm comes with some

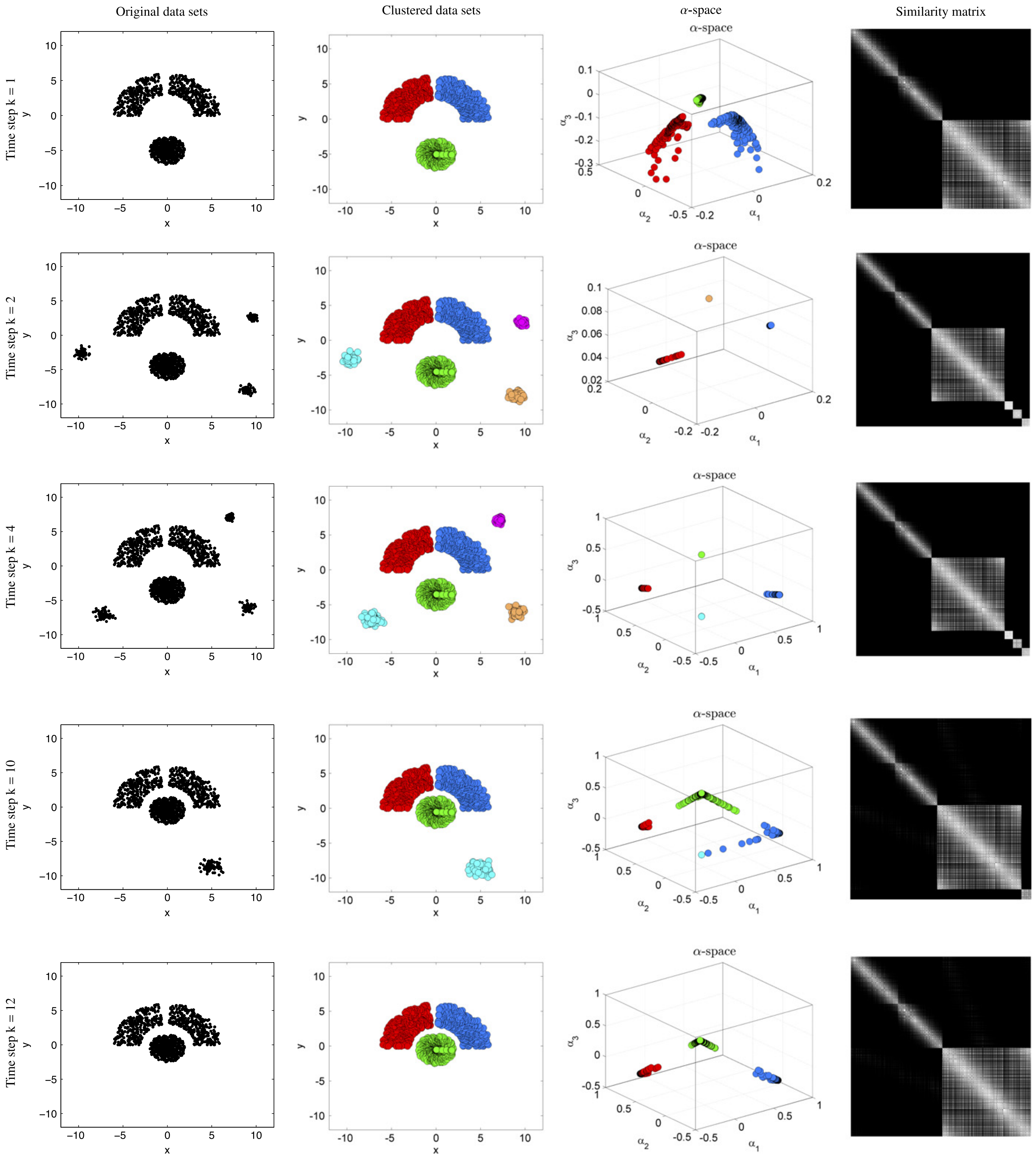

Fig. 7. Synthetic data sets. On-line detection of the creation of more than one cluster at time step k using the proposed I-MSS-KSC(+) approach. At time step $k = 2$, three new clusters appear and evolve. Two of them disappear at time step $k = 10$ and the third one dies out at $k = 12$. The labels are just provided for the consistent clusters i.e. the ones that are always present at all the time steps and can possibly evolve over time. The video of this simulation can be found in the supplementary material (see Appendix A) of the paper.

parameters. In all the experiments, we have selected a minimum and maximum number of regions which are stated in the corresponding caption of each of the tested video sequence. Although the EHGB algorithm does not employ labels, it is one of the state-of-the-art algorithms for video segmentation that uses past and future information (in offline mode) in order to segment the current frame. Also this algorithm uses advanced features, such as color and flow histograms. It should be noted that our algorithm uses the previous segmentation results to perform the segmentation of the current frame. And the algorithm uses only the color feature as discriminator (local color histograms).

Four real examples are used to test the validity of the proposed method. The first example shows two bouncing balls and the second example presents a human's hand throwing a ball upwards.

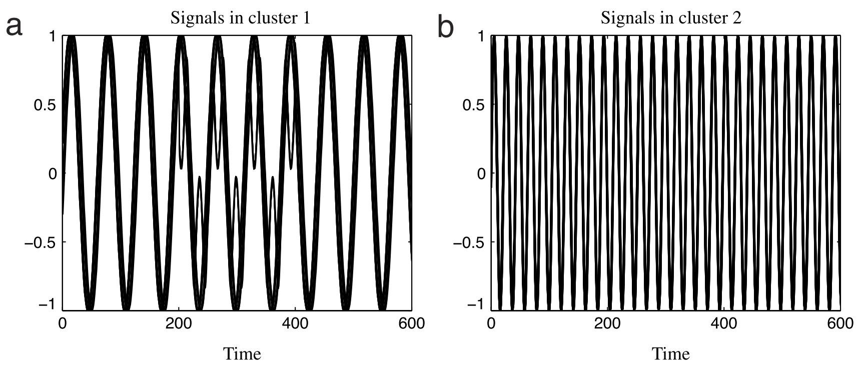

Fig. 8. Ground-truth of the time-series. (a) Signals that are in cluster 1, (a) signals that are in cluster 2.

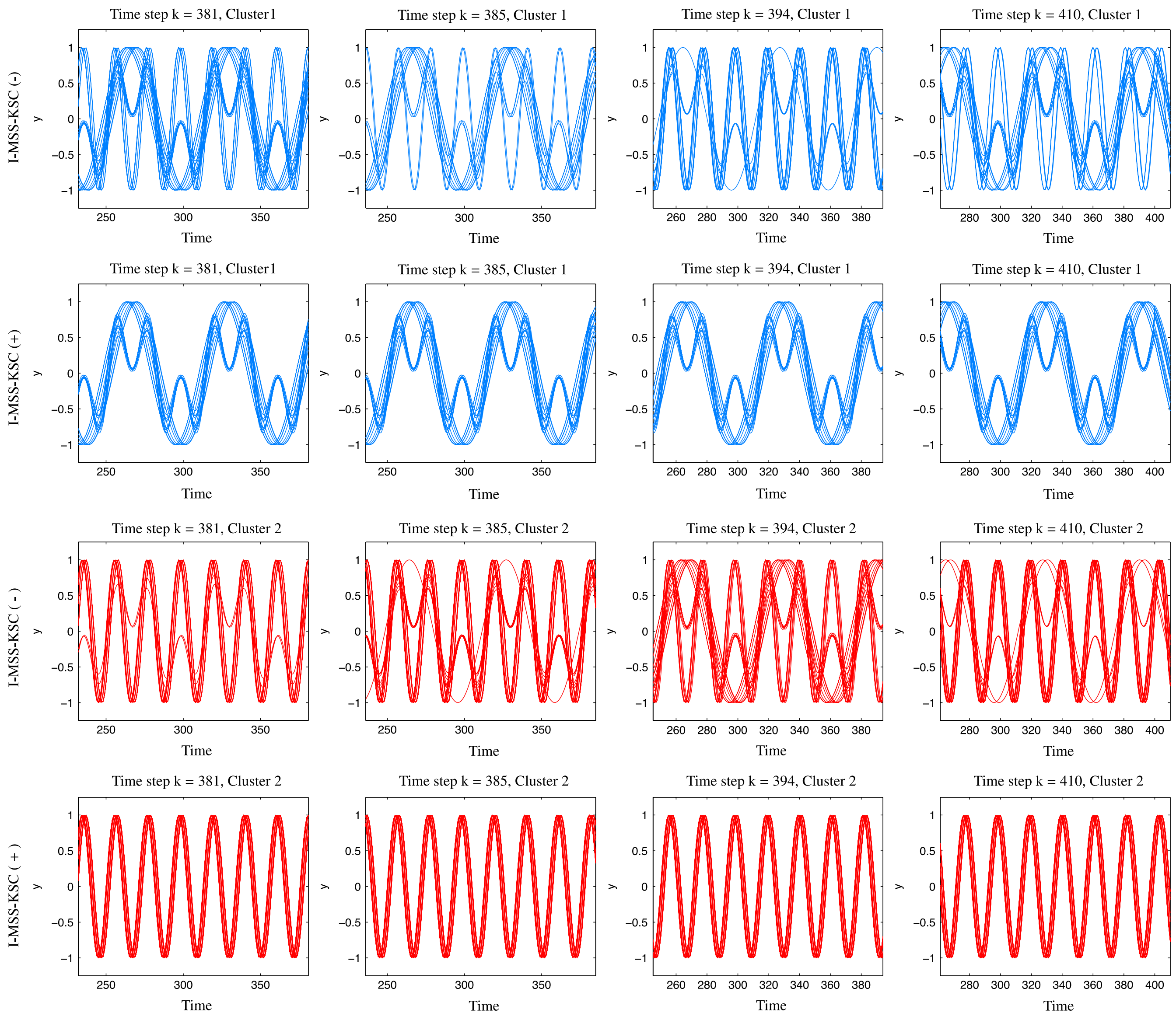

Fig. 9. Synthetic time-series. On-line semi-supervised clustering using the proposed I-MSS-KSC approach implemented in two modes with and without prototypes (I-MSS-KSC(−) and I-MSS-KSC(+)). *First row*: I-MSS-KSC(−): The signals assigned to cluster 1 using the I-MSS-KSC algorithm without the help of any prototypes after the initialization. *Second row*: I-MSS-KSC(+): The signals assigned to cluster 1 using I-MSS-KSC algorithm with the help of the prototypes. *Third row*: I-MSS-KSC(−): The signals assigned to cluster 2 using the I-MSS-KSC algorithm without the help of any prototypes after the initialization. *Fourth row*: I-MSS-KSC(+): The signals assigned to cluster 2 using I-MSS-KSC algorithm with the help of the prototypes.

Table 2
Videos statistics.

Video	width × height	# batch data points	# of frames	Frame rate (frames/s)
Bouncing ball	320 × 180	57 600	139	29
Siamak's hand	320 × 180	57 600	395	29
Dominoes	435 × 343	149 205	121	29
Birds	1280 × 720	921 600	162	29

Table 3
The number of quantization levels, unlabeled/labeled training and validation points used to obtain the initial cluster representatives.

Video	Quantization level	Q	$\mathcal{D}$		$\mathcal{D}^{val}$	
			$\mathcal{D}_u$	$\mathcal{D}_L$	$\mathcal{D}_u^{val}$	$\mathcal{D}_L^{val}$
Bouncing ball	10	3	1000	4	1500	3
Siamak's hand	8	3	800	3	1500	3
Dominoes	15	3	600	3	1500	3
Birds	13	4	1000	4	1500	3

The third video is a video sequence taken from Berkeley video segmentation data set[6] and is called dominoes video and the fourth video is a high definition video showing birds. Descriptions of the used videos can be found in Table 2.

In order to extract features from a given frame, a local color histogram with a 5 × 5 pixels window around each pixel using minimum variance color quantization is computed. The level of quantization in general depends on the video under study. The number of levels used for each of the videos is reported in Table 3. The χ^2 kernel $K(h^{(i)}, h^{(j)}) = \exp(-\frac{\chi_{ij}^2}{\sigma_\chi^2})$ with parameter $\sigma_\chi \in \mathbb{R}^+$ is used to compute the similarity between two color histograms $h^{(i)}$ and $h^{(j)}$. Here $\chi_{ij}^2 = \frac{1}{2}\sum_{q=1}^{n_q} \frac{(h_q^{(i)}-h_q^{(j)})^2}{h_q^{(i)}+h_q^{(j)}}$ where n_q is the number of quantization levels.

The performance of the proposed I-MSS-KSC model depends on the choice of the tuning parameters. We set the regularization parameters $\gamma_1 = \gamma_2 = 1$ to give equal weights to unlabeled and labeled data points. The initial σ_χ (kernel parameter) is tuned using a grid search in the range $[10^{-3}, 10^1]$. The training and validation data points, i.e. $\mathcal{D}$ and $\mathcal{D}^{val}$, consist of the histograms of the chosen pixel (unlabeled data points) together with some labeled data points. These data points are used for training and validation respectively to obtain the initial cluster representatives for the first frame. Then the solution vectors and cluster representatives are updated in an on-line fashion using Algorithm 2 for the subsequent frames. The number of unlabeled/labeled training and validation data points used to obtain the initial cluster representatives are tabulated in Table 3. We obtain the initial model using the MSS-KSC algorithm trained on the first frame and then I-MSS-KSC is applied to segment the upcoming frames in an on-line fashion. For IKM, we let the algorithm to initialize itself and the maximum number of iterations allowed is set to 100.

Both qualitative and quantitative evaluations of the proposed approaches are provided. For quantitative evaluation of the video segmentation there is not a unique criterion to evaluate the performance of the algorithm under study. Several evaluation criteria are proposed in the literature (Borsotti, Campadelli, & Schettini, 1998; Tan, Mat Isa, & Lim, 2013). Here two criteria are used to evaluate the segmentation results. In the first criterion the segmentation obtained by I-MSS-KSC, EHGB, IKM and SemiStream are compared in Table 4 with the results of the minimum variance quantization method (the number of levels is defined by the user) (Heckbert, 1982) using the Variation of information (VOI) index. This index measures the distance between two segmentations in terms of their average conditional entropy. Low values indicate good match between segmentations (Arbelaez, Maire, Fowlkes, & Malik, 2011).

In the second criterion, the segmentations obtained by the above-mentioned approaches are compared in Table 4 with the original frames using the cluster quality index (CQI) which is empirically defined in the following lines.

Suppose for a given image I, the segmented image has N_c clusters (regions). We define the quality index per cluster as follows:

$$QI_j = 1 - \frac{\sum_{i\in\{R,G,B\}} mean(|P_j^i - m_j^i|)}{3}, \quad j = 1, \dots, N_c,$$

where P_j^i denotes the ith channel of the RGB color for pixels of the original image I that belong to cluster j. m_j^i is the mean value of P_j^i. Next, the cluster quality index (CQI) for a given image I is heuristically defined as a weighted sum of the quality index per cluster i.e.

$$CQI(I) = \sum_{j=1}^{N_c} \theta_j QI_j, \tag{14}$$

where $\sum_{j=1}^{N_c} \theta_j = 1$. In our setting the highest weight is assigned to the cluster with minimum QI index. The CQI takes values in the range [0, 1]. The higher the value of the CQI(I) the better the segmentation is.

The obtained results of the proposed I-MSS-KSC algorithm (with two modes of implementation: I-MSS-KSC(−) and I-MSS-KSC(+)), Incremental K-means, EHGB and SemiStream for some of the

[6] ftp://ftp.cs.berkeley.edu/pub/projects/vision/BVDS_train.tar.gz.

Table 4
Comparison of IKM, SemiStream, EHGB, I-MSS-KSC (−) and I-MSS-KSC (+) in terms of averaged cluster quality and variation of information indices over the number of frames.

Video	Evaluation criterion	Method				
		IKM	SemiStream	EHGB	I-MSS-KSC (−)	I-MSS-KSC (+)
Bouncing ball	CQI	0.906	0.921	0.875	0.895	**0.924**
	VOI	1.17	0.715	0.839	0.912	**0.627**
Siamak's hand	CQI	0.872	0.918	0.890	0.919	**0.925**
	VOI	1.08	0.382	1.118	0.494	**0.344**
Dominoes	CQI	0.843	0.865	**0.880**	0.855	0.866
	VOI	1.552	1.581	1.598	1.584	**1.352**
Birds	CQI	0.848	0.868	**0.874**	0.868	0.868
	VOI	0.564	**0.341**	0.539	**0.376**	0.376

Note: The higher the value of CQI index the better the segmentation is. The lower the value of VOI, the better the segmentation is.

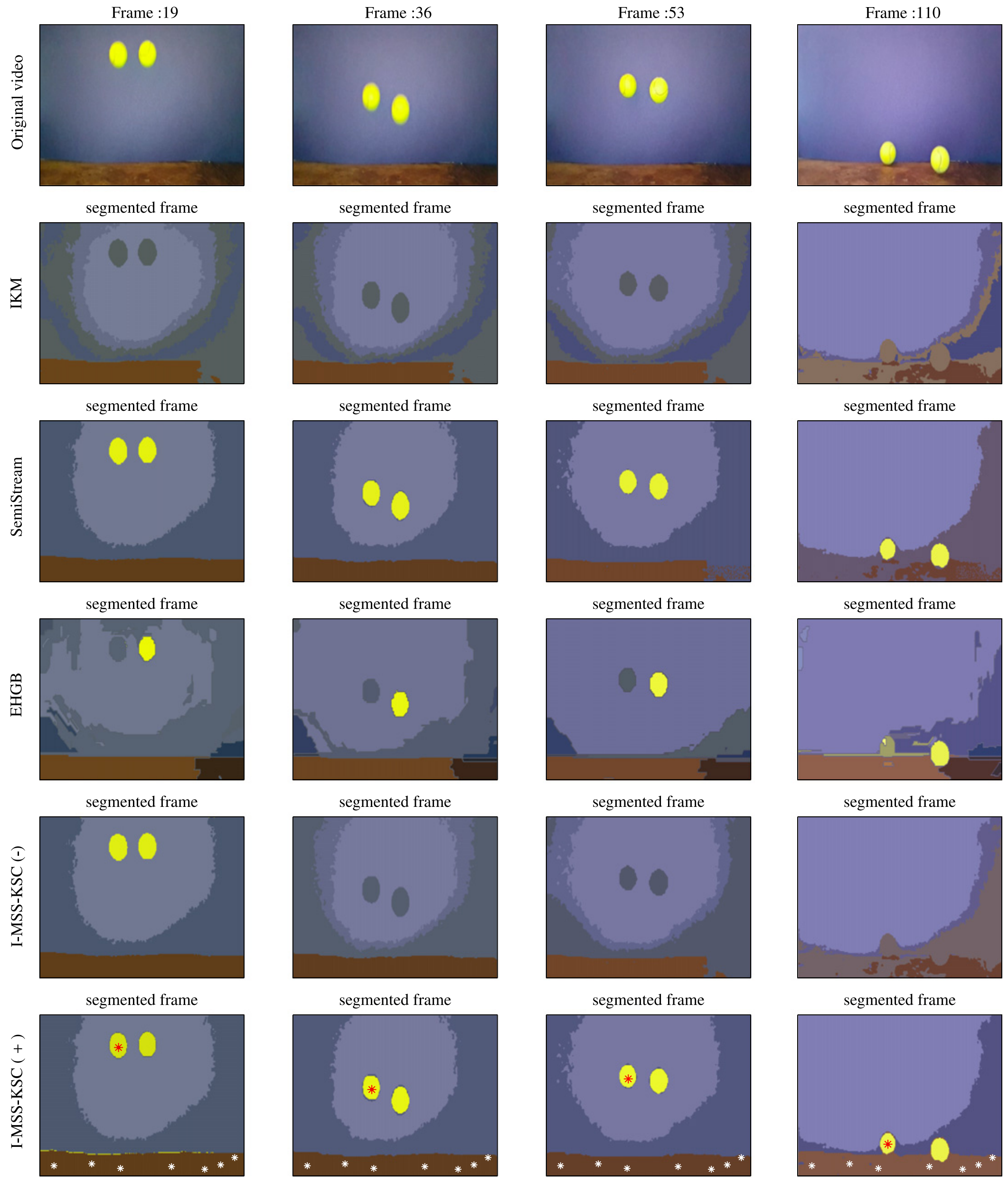

Fig. 10. Bouncing balls video. On-line video segmentation results using the proposed I-MSS-KSC, IKM (Chakraborty & Nagwani, 2011) and EHGB (Grundmann et al., 2010). *First row*: The original frames. *Second row*: The segmentation results obtained by on-line IKM. *Third row*: The segmentation results obtained by SemiStream approach (Halkidi et al., 2012) initialized with MSS-KSC (Mehrkanoon et al., 2015), Notice that the must-link and cannot-link constraints are not shown. *Fourth row*: The segmentation results obtained by EHGB approach (Grundmann et al., 2010) with Min/Max Number of regions $= 10/200$. *Fifth row*: The segmentation results obtained by the proposed I-MSS-KSC algorithm without the help of any labeled pixels after the first frame i.e. I-MSS-KSC($-$) mode. *Sixth row*: The results of the proposed I-MSS-KSC algorithm when labeled pixels for two clusters are provided during on-line segmentation, i.e. I-MSS-KSC($+$) mode.

frames of the bouncing-ball and Siamak's hand videos are depicted in Figs. 10 and 11 respectively (the videos of these simulations are presented in the supplementary material (see Appendix A) of the paper). Figs. 10 and 11, show that it is possible to improve the performance of the video segmentation by incorporating prototypes. Note that for the first video sequence, one of the ball and the table are the objects of interest. Since the table is static, the labels are provided by the user and they are fixed through out the video sequence. Whereas the ball's prototype is provided by a Kalman filter. Here one may notice that I-MSS-KSC($+$) makes it possible to improve the performance by carrying the object labeled through out the video sequence. The labeled pixels of the objects are shown by red and white asterisks ($*$). The obtained results of the proposed method (I-MSS-KSC($+$)), IKM, EHGB and SemiStream for the third video are shown in Fig. 12 (the video of this simulation is provided in the supplementary material (see Appendix A) of the paper). Fig. 12 indicates that the on-line segmentation results can be improved when the labels are incorporated into the algorithm. In Fig. 12, the labeled pixels of the objects are shown by yellow and white asterisks ($*$). The segmentation of the Birds video using the above-mentioned algorithms is shown in Fig. 13. In this video at each time instant 921 600 data points are analyzed.

6. Particular case: one-by-one

In case that the data points arrive one by one, the proposed algorithm 2 is still applicable but few modifications are needed. Assuming that a new data point x_{new} is fed to the algorithm, then

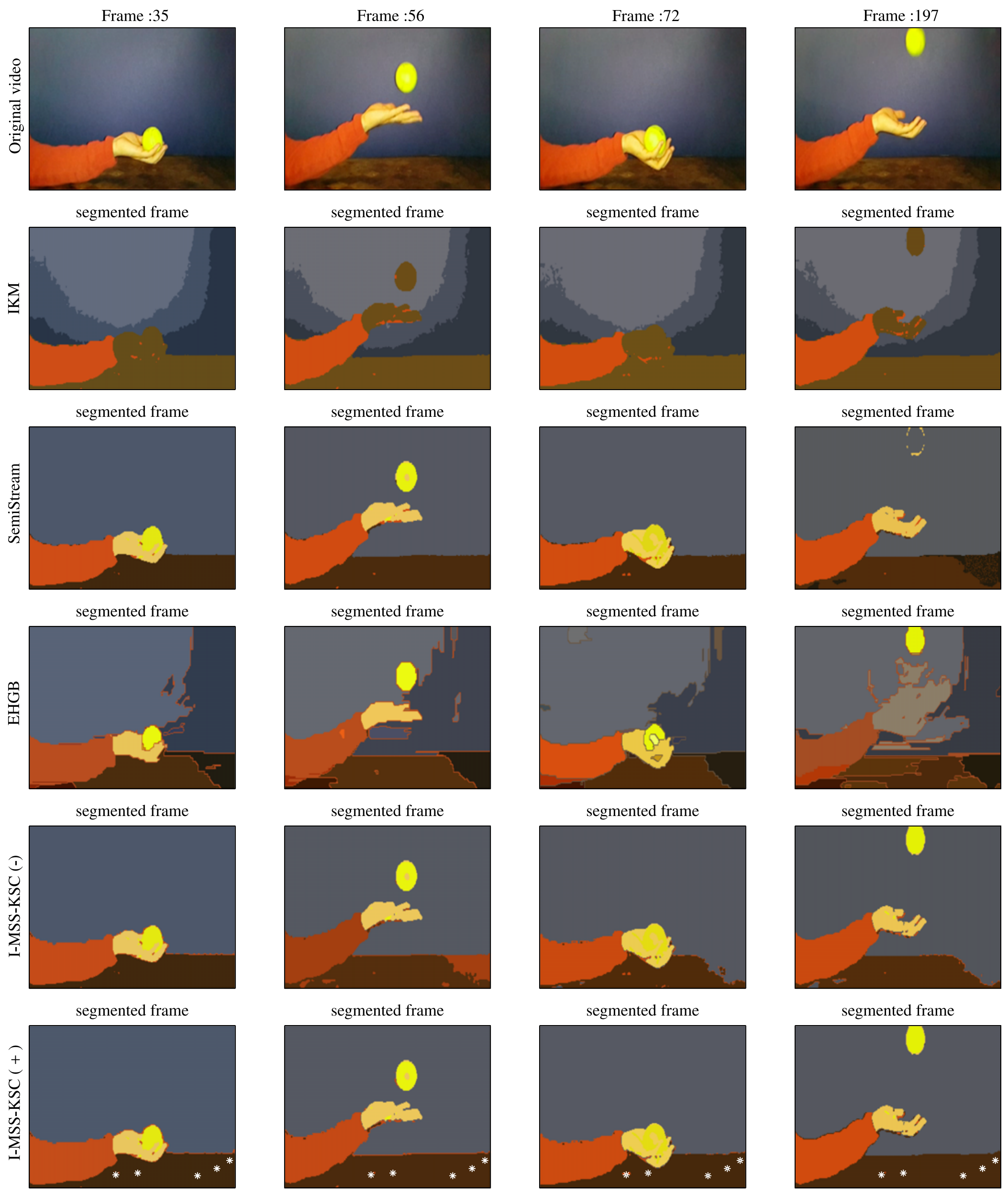

Fig. 11. Siamak's hand video. On-line video segmentation results using the proposed I-MSS-KSC, IKM (Chakraborty & Nagwani, 2011) and EHGB (Grundmann et al., 2010). *First row*: The original frames. *Second row*: The segmentation results obtained by on-line IKM. *Third row*: The segmentation results obtained by SemiStream approach (Halkidi et al., 2012) initialized with MSS-KSC (Mehrkanoon et al., 2015), Notice that the must-link and cannot-link constraints are not shown. *Fourth row*: The segmentation results obtained by EHGB approach (Grundmann et al., 2010) with Min/Max Number of regions = 10/200. *Fifth row*: The segmentation results obtained by the proposed I-MSS-KSC algorithm without the help of any labeled pixels after the first frame, i.e. I-MSS-KSC(−) mode. *Sixth row*: The results of the proposed MSS-KSC algorithm when labeled pixels for two clusters are provided during on-line segmentation, i.e. I-MSS-KSC(+) mode.

the following formula is used to update the cluster representatives in both $\mathcal{X}$ and α-spaces (line 11 of Algorithm 2):

$$rep_{\mathcal{X}}(\mathcal{A}_i(k)) = \mu \cdot rep_{\mathcal{X}}(\mathcal{A}_i(k-1)) + (1-\mu)x_{\text{new}}$$
$$rep_{\alpha}(\mathcal{A}_i(k)) = \mu \cdot rep_{\alpha}(\mathcal{A}_i(k-1)) + (1-\mu)\alpha_{\text{new}},$$

where $\mu \in [0, 1]$. In addition, line 10 of Algorithm 2 which corresponds to cluster elimination is deactivated. We applied the algorithm to two real data sets Iris and Wine from UCI repository. Wine data set contains three types of wine described by three classes. These data are the results of a chemical analysis of wines produced in the same region in Italy but derived from three different cultivators. The Iris data set is composed of three types of iris plant. One class is linearly separable from the other two; the latter are not linearly separable from each other (Asuncion & Newman, 2007).

Initially the model is trained only using the data points (labeled and unlabeled) from the first two classes. When an unlabeled new data point arrives, the algorithm decides whether the new point belongs to one of the existing classes or a new class should be created. For the experiment conducted on the Wine and Iris data sets, all the arriving new points were unlabeled (I-MSS-KSC(−)).

The number of training/test data points and the obtained results over 5 simulation runs are tabulated in Table 5. These data sets are not linearly separable and there is an overlap between two of the classes (class 2 and 3). The creation of the new class is based on the user defined threshold ϵ (see Definition 4). However as we are

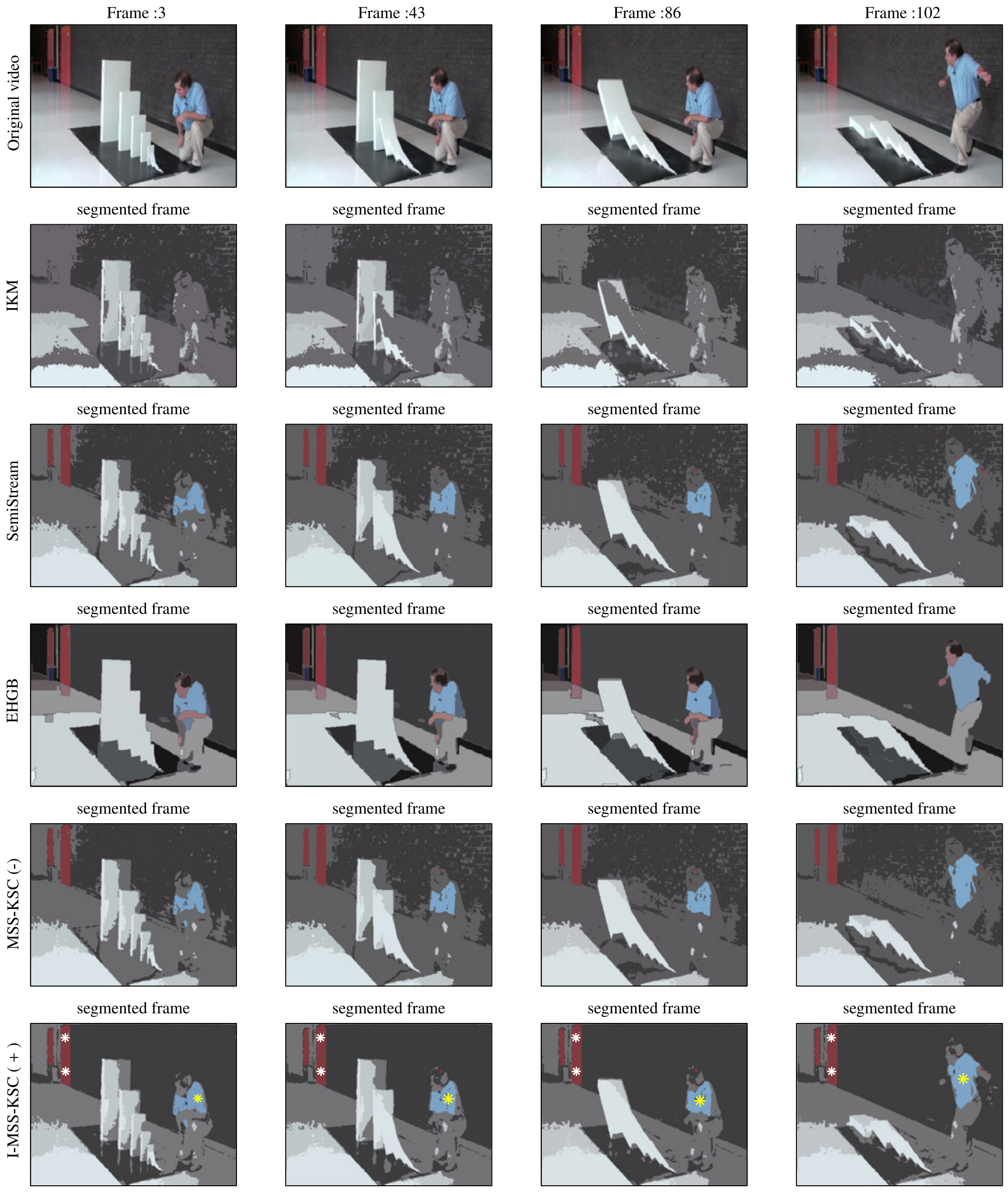

Fig. 12. Dominoes video. On-line video segmentation results using the proposed I-MSS-KSC, IKM (Chakraborty & Nagwani, 2011) and EHGB (Grundmann et al., 2010). *First row*: The original frames. *Second row*: The segmentation results obtained by on-line K-means. *Third row*: The segmentation results obtained by SemiStream approach (Halkidi et al., 2012) initialized with MSS-KSC (Mehrkanoon et al., 2015), Notice that the must-link and cannot-link constraints are not shown. *Fourth row*: The segmentation results obtained by EHGB approach (Grundmann et al., 2010) with Min/Max Number of regions $= 10/100$. *Fifth row*: The segmentation results obtained by the proposed I-MSS-KSC algorithm without the help of any labeled pixels after the first frame i.e. I-MSS-KSC($-$) mode. *Sixth row*: The results of the proposed I-MSS-KSC algorithm when labeled pixels for two clusters (objects) are provided during on-line segmentation. Note that one object is static and therefore its labels will be static and can be provided by the user.

Table 5
The number of unlabeled/labeled training points used to obtain the initial cluster representatives. The averaged accuracy on the test is reported.

Dataset	# classes	Dimension	$\mathcal{D}$		$\mathcal{D}^{test}$	Accuracy
			$\mathcal{D}_u$	$\mathcal{D}_L$		
Iris	3	4	17	33	100	0.81 ± 0.04
Wine	3	13	22	43	113	0.90 ± 0.08

in the semi-supervised setting, the possibility of incorporating the user labels of the third class into the algorithm is also an option.

7. Conclusions

In this paper, a new incremental multi-class semi-supervised algorithm is proposed. It uses the multi-class semi-supervised kernel spectral clustering (MSS-KSC) as core model. The update of the solution vectors and the memberships are obtained using the out-of-sample solution property of the MSS-KSC approach. The user labels or labels provided by a Kalman filter are incorporated into the algorithm in an on-line fashion to improve the performance of the I-MSS-KSC. The validity and applicability of the proposed method is shown on synthetic data sets

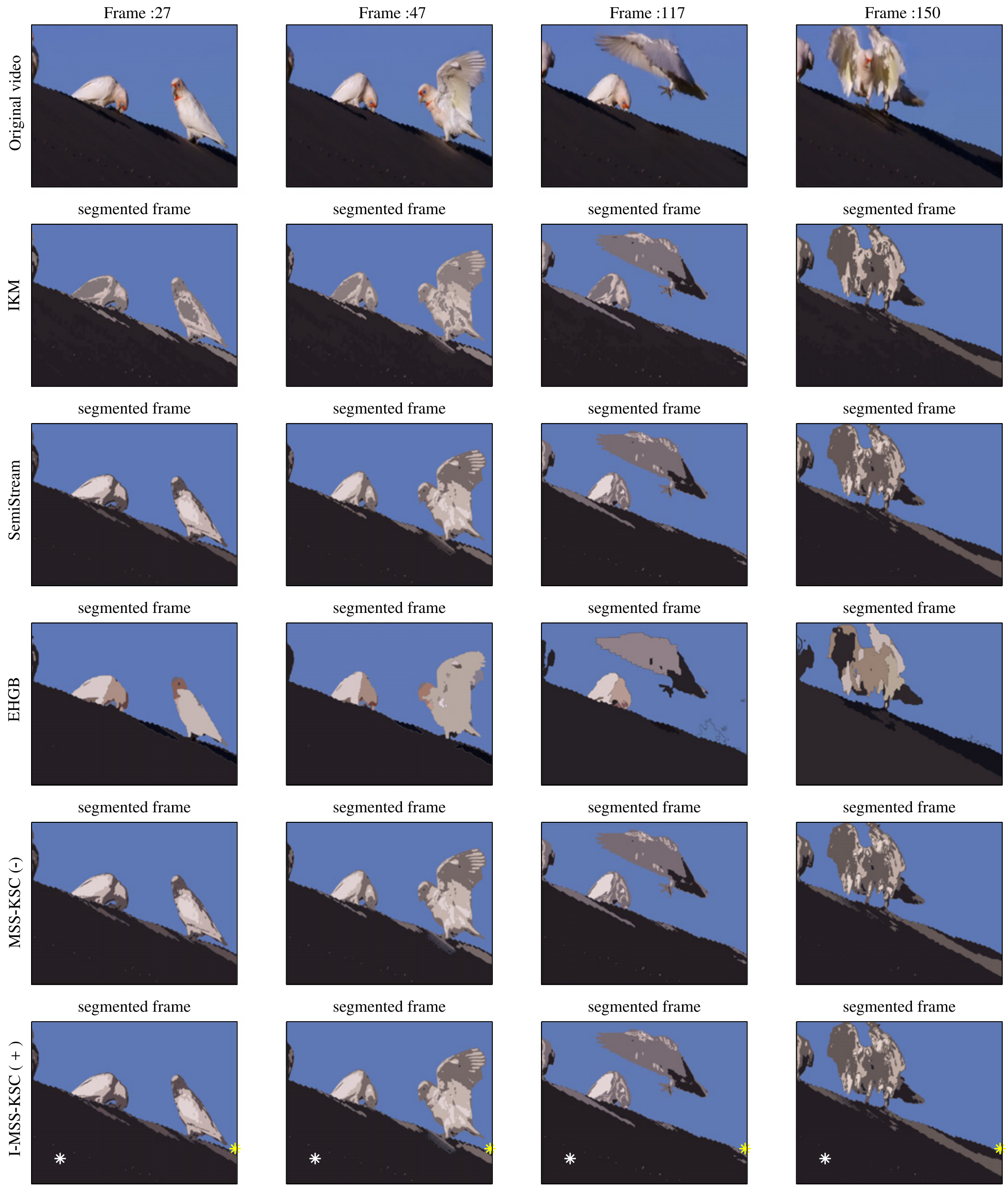

Fig. 13. Birds video. On-line video segmentation results using the proposed I-MSS-KSC, IKM (Chakraborty & Nagwani, 2011) and EHGB (Grundmann et al., 2010). *First row*: The original frames. *Second row*: The segmentation results obtained by on-line K-means. *Third row*: The segmentation results obtained by SemiStream approach (Halkidi et al., 2012) initialized with MSS-KSC (Mehrkanoon et al., 2015), Notice that the must-link and cannot-link constraints are not shown. *Fourth row*: The segmentation results obtained by EHGB approach (Grundmann et al., 2010) with Min/Max Number of regions = 10/100. *Fifth row*: The segmentation results obtained by the proposed I-MSS-KSC algorithm without the help of any labeled pixels after the first frame i.e. I-MSS-KSC(−) mode. *Sixth row*: The results of the proposed I-MSS-KSC algorithm when labeled pixels for two clusters are provided during on-line segmentation.

and some real-life videos sequences. For the video segmentation test cases, the results obtained by the proposed I-MSS-KSC algorithm is compared with those of the incremental K-means (IKM) (Chakraborty & Nagwani, 2011), the Efficient Hierarchical Graph-Based Video Segmentation algorithm (EHGB) (Grundmann et al., 2010) and the semi-supervised incremental clustering algorithm (SemiStream) (Halkidi et al., 2012). In general our results obtained by the proposed I-MSS-KSC approach outperforms the incremental K-means (IKM) (Chakraborty & Nagwani, 2011) and comparable with the ones of the EHGB approach. In addition as opposed to SemiStream that uses many must-link and cannot-link constrains, we work with few user defined labeled data points (prototypes) which act as attractors for other unlabeled data points.

Acknowledgments

• EU: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC AdG A-DATADRIVE-B (290923). This paper reflects only the authors' views, the Union is not liable for any use that may be made of the contained information. • Research Council KUL: GOA/10/09 MaNet, CoE PFV/10/002 (OPTEC), BIL12/11T; PhD/Postdoc grants • Flemish Government: ◦ FWO: projects: G.0377.12 (Structured systems), G.088114N (Tensor based data similarity); PhD/Postdoc grants ◦IWT: projects: SBO POM (100031); PhD/Postdoc grants ◦ iMinds Medical Information Technologies SBO 2014 • Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012–2017). Johan Suykens is a professor at the KU Leuven, Belgium.

Appendix A. Supplementary material

The experimental findings and the demonstrative videos associated with this paper can be found in the on-line version at ftp://ftp.esat.kuleuven.be/stadius/siamak/155-spt.zip.

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.neunet.2015.08.001.

References

Adankon, M. M., Cheriet, M., & Biem, A. (2009). Semi-supervised least squares support vector machine. *IEEE Transactions on Neural Networks*, *20*(12), 1858–1870.

Alzate, C., & Suykens, J. A. K. (2010). Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(2), 335–347.

Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. (2011). Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(5), 898–916.

Asuncion, A., & Newman, D.J. (2007). UCI machine learning repository.

Badrinarayanan, V., Budvytis, I., & Cipolla, R. (2013). Semi-supervised video segmentation using tree structured graphical models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(11), 2751–2764.

Barrero, O. (2005). *Data assimilation in magnetohydrodynamics systems using Kalman filtering*. (Ph.D. dissertation), Leuven, Belgium: Katholieke Universiteit Leuven (KU Leuven).

Belkin, M., Niyogi, P., & Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *The Journal of Machine Learning Research*, *7*, 2399–2434.

Borsotti, M., Campadelli, P., & Schettini, R. (1998). Quantitative evaluation of color image segmentation results. *Pattern Recognition Letters*, *19*(8), 741–747.

Chakrabarti, D., Kumar, R., & Tomkins, A. (2006). Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 554–560). ACM.

Chakraborty, S., & Nagwani, N. (2011). Analysis and study of incremental k-means clustering algorithm. In *High performance architecture and grid computing* (pp. 338–341). Springer.

Chang, C. C., Pao, H. K., & Lee, Y. J. (2012). An RSVM based two-teachers–one-student semi-supervised learning algorithm. *Neural Networks*, *25*, 57–69.

Chen, S. (2012). Kalman filter for robot vision: a survey. *IEEE Transactions on Industrial Electronics*, *59*(11), 4409–4420.

Chi, Y., Song, X., Zhou, D., Hino, K., & Tseng, B. L. (2007). Evolutionary spectral clustering by incorporating temporal smoothness. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 153–162). ACM.

Franklin, G., Powell, J., & Workman, M. (1990). *Digital control of dynamic systems* (2nd ed.). Addison-Wesley.

Golub, G. H., & Van Loan, C. F. (2012). *Matrix computations*. Johns Hopkins University Press.

Grundmann, M., Kwatra, V., Han, M., & Essa, I. (2010). Efficient hierarchical graph-based video segmentation. In *IEEE conference on computer vision and pattern recognition* (pp. 2141–2148). IEEE.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On clustering validation techniques. *Journal of Intelligent Information Systems*, *17*(2–3), 107–145.

Halkidi, M., Spiliopoulou, M., & Pavlou, A. (2012). A semi-supervised incremental clustering algorithm for streaming data. In *Advances in knowledge discovery and data mining* (pp. 578–590). Springer.

He, X. (2004). Incremental semi-supervised subspace learning for image retrieval. In *Proceedings of the 12th annual ACM international conference on Multimedia* (pp. 2–8). ACM.

Heckbert, P. (1982). *Color image quantization for frame buffer display. Vol. 16*. ACM, no. 3.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME. Series D, Journal of Basic Engineering*, *82*, 34–45.

Kamiya, Y., Ishii, T., Furao, S., & Hasegawa, O. (2007). An online semi-supervised clustering algorithm based on a self-organizing incremental neural network. In *International joint conference on neural networks*, (IJCNN 2007). (pp. 1061–1066). IEEE.

Langone, R., Agudelo, O. M., De Moor, B., & Suykens, J. A. K. (2014). Incremental kernel spectral clustering for online learning of non-stationary data. *Neurocomputing*, *139*, 246–260.

Mehrkanoon, S., Alzate, C., Mall, R., Langone, R., & Suykens, J. A. K. (2015). Multiclass semisupervised learning based upon kernel spectral clustering. *IEEE Transactions on Neural Networks and Learning Systems*, *26*(4), 720–733.

Mehrkanoon, S., & Suykens, J. A. K. (2012). Non-parallel semi-supervised classification based on kernel spectral clustering. In *The 2013 international joint conference on neural networks (IJCNN)* (pp. 2311–2318). IEEE.

Mehrkanoon, S., & Suykens, J. A. K. (2014). Large scale semi-supervised learning using KSC based model. In *Proceedings of the 2014 IEEE world congress on computational intelligence (IEEE WCCI/IJCNN 2014),July 6-11, 2014, Beijing, China* (p. 8). IJCNN.

Ning, H., Xu, W., Chi, Y., Gong, Y., & Huang, T. S. (2010). Incremental spectral clustering by efficiently updating the eigen-system. *Pattern Recognition*, *43*(1), 113–127.

Sarlin, P. (2013). Self-organizing time map: an abstraction of temporal multivariate patterns. *Neurocomputing*, *99*, 496–508.

Suliman, C., Cruceru, C., & Moldoveanu, F. (2010). Kalman filter based tracking in an video surveillance system. *Advances in Electrical and Computer Engineering*, *10*(2), 30–34.

Tan, K. S., Mat Isa, N. A., & Lim, W. H. (2013). Color image segmentation using adaptive unsupervised clustering approach. *Applied Soft Computing*, *13*(4), 2017–2036.

Teichman, A., & Thrun, S. (2012). Tracking-based semi-supervised learning. *The International Journal of Robotics Research*, *31*(7), 804–818.

Vapnik, V. (1998). *Statistical learning theory*. Wiley.

Wang, Y., Chen, S., & Zhou, Z.-H. (2012). New semi-supervised classification method based on modified cluster assumption. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(5), 689–702.

Warren Liao, T. (2005). Clustering of time series data—a survey. *Pattern Recognition*, *38*(11), 1857–1874.

Weng, S.-K., Kuo, C.-M., & Tu, S.-K. (2006). Video object tracking using adaptive Kalman filter. *Journal of Visual Communication and Image Representation*, *17*(6), 1190–1208.

Xiang, S., Nie, F., & Zhang, C. (2010). Semi-supervised classification via local spline regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(11), 2039–2053.

Yang, Y., Nie, F., Xu, D., Luo, J., Zhuang, Y., & Pan, Y. (2012). A multimedia retrieval framework based on semi-supervised ranking and relevance feedback. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(4), 723–742.

Zhong, J., & Sclaroff, S. (2003). Segmenting foreground objects from a dynamic textured background via a robust Kalman filter. In *Ninth IEEE international conference on computer vision, 2003. Proceedings* (pp. 44–50). IEEE.

Zhu, X. (2006). Semi-supervised learning literature survey. Computer Science, University of Wisconsin–Madison.