Neurocomputing 173 (2016) 1471–1484

Contents lists available at ScienceDirect

Neurocomputing

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

Incremental episodic segmentation and imitative learning of humanoid robot through self-exploration

Farhan Dawood, Chu Kiong Loo*

[a] Department of Artificial Intelligence, University of Malaya, Lembah Pantai, 50603 Kuala Lumpur, Malaysia

ARTICLE INFO

Article history:
Received 27 November 2014
Received in revised form
29 June 2015
Accepted 13 September 2015
Communicated by Bin He
Available online 9 October 2015

Keywords:
Robot behaviour learning
Adaptive Resonance Map
Gaussian distribution
Hidden Markov Model
Incremental learning
Topological map

ABSTRACT

Imitation learning through self-exploration is an essential mechanism in developing sensorimotor skills for human infants as well for robots. We assume that a primitive sense of self is the prerequisite for successful social interaction rather than an outcome of it. During imitation learning, a crucial element of conception involves segmenting the continuous flow of motion into simpler units – motion primitives – by identifying the boundaries of an action. Secondly, in realistic environment the robot must be able to learn the observed motion patterns incrementally in a stable adaptive manner without corrupting previously learned information. In this paper, we propose an on-line and unsupervised motion segmentation method allowing the robot to imitate and perform actions by observing the motion patterns performed by other partner through Incremental Kernel Slow Feature Analysis. The segmentation model directly operates on the images acquired from the robots vision sensor (camera) without requiring any kinematic model of the demonstrator. After segmentation, the spatio-temporal motion sequences are learned incrementally through Topological Gaussian Adaptive Resonance Hidden Markov Model. The learning model dynamically generates the topological structure in a self-organizing and self-stabilizing manner. Each node represents the encoded motion element (i.e. joint angles). The complete architecture was evaluated by simulation experiments performed on DARwIn-OP humanoid robot.

1. Introduction

Imitation plays a major role in human development and especially for learning of new unexampled motor skills. Several studies indicate that imitation ability emanates early in life, and is considered as the major mode of learning for infants and likewise for adults [1]. Even at very young ages the infants learn to control their bodies and execute tasks by watching others performing these tasks. Unlike humans, robots have limited capabilities to learn from their environment and it can only be embedded by explicitly programming the robots according to specific applications. Due to the multifariousness of actions to be executed and the range of possible interaction with objects and humans, it would be not feasible and tedious to explicitly pre-program a robot with such capabilities. For robotic perspective, imitation is of great interest because it provides a potential means of automatic programming complex systems without extensive trials.

Self-exploration establishes the basic building block for almost all forms of learning based on action. Starting from day one, the new-born babies engage themselves in an active world where they observe and interact with others. Using self-exploration, the infants actively get involved in random acts and observe the consequent changes in the perceptual world. Developmental psychology theorists point out two opposing hypotheses concerning the origins of self-concepts in infants. One view considers the fact that the infant learns about itself primarily through interaction with others. Infants have no prior self-concepts and learn about the possibilities and powers of their own actions through observing the reactions of others to their behavior. On the contrary, other theorists assert that rudimentary concept about self exist prior to such social experience [2,3]. In this view, infants have a proprioceptive sense of self that derives in part from their own body movements which the authors have called 'body babbling' [3]. According to this hypothesis, a primitive sense of self is the prerequisite for successful social interaction rather than an outcome of it.

Body babbling is the process of learning how specific muscle movements achieve various elementary body configurations. The concept of self-learning put forward by Meltzoff and Decety [4] arguing about the developmental model of learning, stating that human infants monitor their own body via proprioception and associates their acts-as-felt to the acts-as-seen in others. They call this hypothesis as 'like-me' framework proposing that "*the other acts like me and I can act like the other*". Exteroception (perception

* Corresponding author. Tel.: +603 7967 6381; fax: +603 7957 9249.
E-mail address: ckloo.um@um.edu.my (C.K. Loo).

http://dx.doi.org/10.1016/j.neucom.2015.09.021
0925-2312/© 2015 Elsevier B.V. All rights reserved.

of the acts of others) and Proprioception (perception of one's own acts) are not one undifferentiated whole.

A robot can be more than a passive observer of the world as it learns and develops. Body image is fundamental for manipulation and it is extremely adaptive in animals. Body image pertains to a collective representation that embodied agents preserve in order to plan and execute actions in the environment, perceive and comprehend the behaviours of others, and search and develop episodic memory commands. Kinaesthetic-visual correspondence is the recognition of resemblance amongst the notion of the one's own body's extent and movement and how it appears.

The approach adopted in our method is to develop an imitative mechanism for learning primitive concepts through self-exploration similar to what the human infant does. Its proposed model is designed to rigorously investigate the feasibility of the hypothesis that self-exploration could be a foundational step in developing social cognition. In our mirror experiment, a humanoid robot stands in front of a glass mirror in order to obtain the visual features of the self-executed actions. In designing such incremental learning system several key issues must be addressed: automatic segmentation of the observed actions, automatic clustering, incremental learning and organization of the learned data for easy and efficient retrieval.

The main contribution of our work is in developing a novel on-line segmentation and incremental learning architecture through self-exploration capable of solving the following problems: (1) During a continuous interaction with human/demonstrator how a robot can autonomously determine the start and end of the action using on-board vision sensors without utilizing any kinematic model of the demonstrator? (2) How can the model retain previously learned data and also acquire new observed knowledge in a self-organizing manner? (3) How to select the structure for the probabilistic model, i.e., estimating the number of states to efficiently encode the sensory data without restricting the learning capabilities of the robot?

For this purpose, we developed an algorithm for incremental learning and automatic segmentation based on the image sequences captured from the robot's on-board vision sensors. The complete proposed architecture is shown in Fig. 1. The learning system begins with determining the primitives from continuous movements using on-line segmentation. For this purpose, we have developed an algorithm for the automatic segmentation based on the image sequences captured from the robot's on-board vision sensors. The observed actions are segmented into episodes of different actions determined by the start and end of actions through Incremental Kernel Slow Feature Analysis (Inc-KSFA) algorithm which extracts slowly varying features from input signals. The variation of slowly changing features is exploited to determine the occurrence of different activities in an incremental fashion. These segmented boundaries of action assists the robot to cluster the observed own actions as primitive actions using TGARM.

In addition to obtaining the visual images of self performed actions, the joint angle values of the self for different behavioral actions are also acquired from robot sensors as motion elements. These motion elements are mapped onto the behavior space. For learning the motion features we proposed a probabilistic incremental learning algorithm called Topological Gaussian Adaptive Resonance Hidden Markov Model (TGAR-HMM). In contrast to conventional HMM, the developed probabilistic learning algorithm is based on incrementally learning spatio-temporal behavioral sequences by developing the graph based structure of the behavior patterns in the form of topological map using Topological Gaussian Adaptive Resonance Map (TGARM). The topological model incrementally updates the number of states and parameters required by the probabilistic model to encode the observed motion elements. This compactly describes the environment as a collection of nodes linked by edges.

Based on segmented data, the learning algorithm is triggered; whenever an action starts, the learning algorithm incrementally encodes the motion elements. The learned motion elements are also labelled using the segments defined during segmentation.

In order to evaluate the performance of our proposed architecture, we have tested our algorithm on different action sequences performed by the robot. These tests were performed in simulation and on Darwin-OP humanoid robot. The rest of the paper is organized as follows: The next section reviews the related work. Later in Section 3 we will explain the segmentation algorithm performed though Incremental Kernel Slow Feature Analysis. In Section 4 we explain our proposed incremental learning algorithm through TGAR-HMM. The experimental results are discussed and analysed in Section 6. Finally, Section 7 concludes our work along with some discussion in the proposed model and future work.

2. Related work

Gold and Scassellati [44] have developed a model based on the classical mirror test, for self-recognition through expecting motion in the visual field utilizing an action perception loop. During experimentation, the robot can only view the limited part of the body which limits the observation of the whole body. The mirror in this experiment is only utilized to differentiate between self and other.

The behaviour patterns are considered as a sequence of motion primitives [5], atomic parts of a behavioural sequence. For example, if the demonstrator is performing a fighting action, then each motion sequence may correspond to a simple move, such as kick or punch. These basic actions, which combines together to form a variety of actions, are specified as motion primitives. The

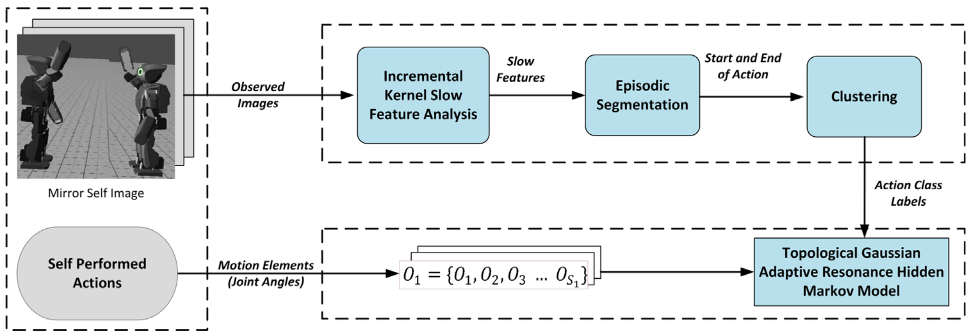
Mirror Self Image
Observed Images
Incremental Kernel Slow Feature Analysis
Slow Features
Episodic Segmentation
Start and End of Action
Clustering
Action Class Labels
Self Performed Actions
Motion Elements (Joint Angles)
$O_1 = \{O_1, O_2, O_3 \ldots O_{S_1}\}$
Topological Gaussian Adaptive Resonance Hidden Markov Model

Fig. 1. Architecture for autonomous motion segmentation and incremental imitative learning.

automated segmentation of observed action sequences into plausible meaningful behaviours is a main problem in imitation learning.

Most of the motion segmentation algorithms for robot learning rely on joint angles and very less focus was given to segment motion patterns directly from the vision. Takano and Nakamura [6] utilises the correlation among actions for segmentation of motion patterns. The observed movements are first divided into smaller sequences and encoded into HMM. If the value of the correlation is smaller for the moments, they are defined as boundaries of the actions.

Kulic et al. [7] presented a segmentation and incremental learning algorithm through stochastic methods. The segmentation algorithm uses Hidden Markov Model to represent the observed data into motion primitives which are then clustered based on intra model distances measured using the relative log likelihood. The incremental learning is performed by arranging these clusters in tree structure. Each node in the tree represents a motion primitive. In addition to the tree structure, a directed graph is built representing the relationship between the motion primitives which is then used for motion generation [7].

Meier et al. [8] presented a sequential segmentation and recognition algorithm for movements based on Dynamic Movement Primitives (DMP). This reformulation first finds the known primitives from the library, and learn new primitives if no match can be found. The segmentation is performed through stochastic DMP which finds the minima points from the velocity of acceleration profile. The model operates off-line considering the development of motion library a priori.

Mohammad et al. [9,10] provide an algorithm for automatic segmentation of actions and commands from continuous motion streams using constrained motif discovery. They used a general change point detection algorithm called Robust Singular Spectrum Transform (RSST) [11] and solves the segmentation problem by finding the boundaries of motifs. However, the algorithm operates on the motion captured data instead of directly relying on the visual data for the segmentation process.

For segmentation using only the vision data, Kulic et al. [12] developed an autonomous motion primitive segmentation algorithm. Their model is based on detecting the features through optical flow. The suitable feature points which show significant optical flow values are extracted from the image and tracked from frame to frame. These feature points were clustered together for coherent movements.

A natural way of representing observed behaviours by the robot is through the use of probabilistic models. Hidden Markov Models (HMM) [13] have been widely used for human motion recognition and generation. The ability of HMMs to generalize human demonstrations has led to the development of several methods for HMM based movement recognition and generation [14,15]. However, most of these methods operate off-line where the model structure is static and determined beforehand. In off-line or batch learning method, the motion patterns are processed sequentially. Such interaction might seem to be a bit artificial since the patterns are predefined by the designer and cannot accommodate the new properties of various tasks. Therefore, the stochastic model should allow new information to be incorporated incrementally without corrupting previously acquired data.

Another issue associated with the well-established learning algorithm based on stochastic models such as HMM or Gaussian Mixture Models (GMM) [16,17] is in estimating the structure of the model. If there are less number of states than the observed trajectory components, the model cannot explain the patterns substantially. On the contrary, if the model states are large in number, the system will require too much training data and excessive computational requirement. In the literature the structure of HMM i.e. the number of HMM states is either to be chosen manually or using other methods such as Bayesian Information Criterion (BIC) [16] or Akaike Information Criterion (AIC) [17]. This limits the number of motion patterns to be learned a priori and result in a trade-off between model fitness and number of parameters.

The above-mentioned issues with estimating the structure of HMM have been addressed by Niekum et al. [18]. They have developed a Beta Process Autoregressive HMM (BP-AR-HMM) to solve the issues of segmentation and defining the structure of HMM in a flexible way. The model segments the demonstration into motion categories. These segments are further subdivided into semantically grounded states for finite-state automation. Butterfield et al. [19] developed a method by extending Hierarchical Dirichlet Processes Hidden Markov Models (HDP-HMM) to deal with perceptual aliasing and discovering an suitable number of skills.

Calinon and Billard [20] presented two incremental learning approaches: direct and generative method. These methods are used to update the model parameters when novel skills are demonstrated. The probabilistic model is first learned using joint angles trajectories captured through motion sensors and is progressively refined using kinaesthetic process. For direct update method the idea is to adapt the EM algorithm by separating the parts dedicated to the data already used to train the model and the one dedicated to the newly available data. The model is first created with data points and updated iteratively during EM step, until convergence to the set of parameters. For the generative method, an initial GMM model is created using the classic EM algorithm. When a new data are available, regression is used to generate stochastically new data by considering the current model.

Seyhan et al. [21] developed a behaviour learning model for simple and complex actions in robot using HMM and a correlation based ART (CobART) network. CobART is a type of ART 2 network. The motion primitives acquired from the CobART are modelled through HMM to represent a relation between these motion primitives. The model generates different categories for the same behaviours but with slight variation between them, thus providing a correlation between the motion patterns. A similar model was presented by [22] through hierarchically integrating CobART networks. The model learns the spatio-temporal sequences, however, the structure of the BehaviourHMM proposed by them is fixed and cannot grow incrementally.

Vasquez et al. [23] have developed an incremental model for learning and predicting through growing HMM. The structure of HMM model was updated through the use of topological mapping algorithm called Instantaneous Topological Mapping (ITM). The model was designed for vehicle motion learning and prediction. Similar to our approach, the growing HMM architecture uses the topological mapping structure to define the HMM structure. In an earlier version of the algorithm they have employed growing neural gas (GNG). Tscherepanow [24] developed a topological structure for unsupervised learning using Fuzzy ART. Using TopoART algorithm a stable representation of the data is created. The model was only used for clustering and is not suitable for spatio-temporal learning. A similar architecture was introduced by Okada et al. [25] utilizing the HMM and SOINN. In contrast to our proposed method for incremental learning, HMM-based SOINN uses HMM to convert time-series data to a fixed-length vector. Later, SOINN is used for incremental clustering of the data.

The proposed incremental learning approach is related to the growing HMM (GHMM) model presented by Vasquez et al. [23]. However, unlike the GHMM algorithm, in the proposed algorithm we used Gaussian distribution for adding and updating new nodes. Furthermore, instead of using fixed covariance matrix for modelling the observation probabilities, the TGAR-HMM model uses the covariance matrix which is updated based on the input patterns.

3. Episodic segmentation-incremental kernel slow feature analysis

Slow feature analysis (SFA) is a technique that extracts slowly varying features from raw input stream of sensory signals [26]. The main intuition behind SFA is based on the presumption that the information contained in a signal does not change suddenly, but slowly over time. In other words, the slowness learning principle represents the salient features on an input stream of data in a way invariant to frequent transformations. SFA is originally designed to learn invariances in the model of primate's visual system. For systems based on vision the sensory inputs are rapidly changing and encode the visual information which is relevant to the behaviours. Several studies have been conducted to show the relevance of slowness principle to the biological sensory cortex [27]. SFA have been widely used in various machine learning applications such as classification [28], feature extraction [29], data segmentation [30] and dimensionality reduction.

When dealing with real-time video sequences, the limit or length of data is not known a priori, therefore, an incremental algorithm is needed. One incremental version of SFA was proposed by Luciw et al. [31] which learns the estimation of the features extracted resulting loss of accuracy. Another variant of SFA was developed based on the kernel methods for data expansion and introduced a regularized sparse kernel SFA algorithm for solving the optimization problem. A different approach was employed by Luciw et al. [32]. They developed a hierarchical extension of SFA composed of multiple layers of SFA representing a receptive field dealing with non-linear expansion through spatial partitioning, reducing computational cost.

In this paper, we study the application of SFA for unsupervised segmentation of motion primitives from a continuous stream of visual data. More precisely, we demonstrate that it is possible to discover the dynamics of motion primitives in an unsupervised manner using Inc-KSFA. Our motivation is based on the close relationship between human perception and SFA. In this paper we used the idea of SFA for video segmentation developed by Liwicki et al. [33]. However, instead of using complex gradient based kernel computed in Krein space [34], we have utilized the method of reproducing kernel in Hilbert space, to reduce computational complexity. Incremental Kernel SFA algorithm does not rely on any predefined training images for processing. It discovers the temporal variations in a video stream online.

3.1. Slow feature analysis-formulation

Slow Feature Analysis (SFA) is an unsupervised approach which searches for a set of mappings g_i from I–dimensional input data $\mathbf{x}(t) = [x_1(t), \dots, x_I(t)]^T$ to generate J-dimensional output signal $\mathbf{y}(t) = [y_1(t), \dots, y_J(t)]^T$ with components $y_j(t) := g_j(\mathbf{x}(t))$ such that $j \in \{1, \dots, J\}$. The optimization problem of SFA is defined by minimizing the temporal variations of the output signal, mathematically

$$\Delta(y_j) := \langle \dot{y}_j^2 \rangle_t \tag{1}$$

Under the constraints

$$\langle y_j \rangle_t = 0 \quad \text{(Zero Mean)} \tag{2}$$

$$\langle y_j^2 \rangle_t = 1 \quad \text{(Unit Variance)} \tag{3}$$

$$\forall_{i<j}, \langle y_i y_j \rangle_t = 0 \quad \text{(Decorrelation)} \tag{4}$$

where $\langle \cdot \rangle$ and $\dot{y}$ represent the temporal averaging and the derivative of y, respectively. The constraints (2)–(4) are inserted to exclude trivial solutions. The unit variance constraint prevents constant signals from emerging and avoids trivial solution, the decorrelation constraint imposes distinctness among data patterns and the zero mean constraint is introduced for convenience and simplicity.

Often, the mapping between the input and output data is assumed to be linear such that the input–output transformation is the weighted sum i.e., $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x}(t)$. However, for the real time applications the non-linear input features are expanded through expansion function, $h(\cdot)$, such that $\mathbf{z}_i := h(\mathbf{x}_i)$, yields the non-linearly expanded signal. After expansion, the problem can be treated as linear and the j-th output signal component is given by $y_j(t) = g_j(\mathbf{x}(t)) = \mathbf{w}_j^T\mathbf{h}(\mathbf{x}(t)) = \mathbf{w}_j^T\mathbf{z}(t)$. Generally, the expanded signal may not have zero mean, however, without the loss of generality we can compute the centred data matrix by subtracting the mean over time such that $\mathbf{z} := \mathbf{h}(\mathbf{x}) - \mathbf{h}_0$, where $\mathbf{h}_0 = \langle \mathbf{h}(x) \rangle_t$. Assuming that the signal has unit variance such that $\mathbf{w}_i^T \langle \mathbf{z}\mathbf{z}^T \rangle_t \mathbf{w}_j = 1$, where $\mathbf{z}\mathbf{z}^T = \mathbf{I}$, the optimization problem can be written in more convenient way using matrix notations

$$\min_{\mathbf{W}} \operatorname{tr}(\mathbf{W}^T\dot{\mathbf{Z}}\dot{\mathbf{Z}}^T\mathbf{W}), \quad \text{s.t.} \quad \mathbf{W}^T\mathbf{Z}\mathbf{Z}^T\mathbf{W} = \mathbf{I} \tag{5}$$

where $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_n]$ contains the input features and $\dot{\mathbf{Z}}$ represents the temporal derivation matrix, and $\operatorname{tr}(\cdot)$ computes the trace of a matrix. The matrix $\mathbf{S}$ represents the whitening matrix to fulfil the unit variance constraint such that $\mathbf{S}^T\mathbf{Z}\mathbf{Z}^T\mathbf{S} = \mathbf{I}$. Then, the directions of least variance in the derivative signals $\dot{\mathbf{Z}}$ are found on the derivative covariance matrix $\dot{\mathbf{Z}}\dot{\mathbf{Z}}^T$ and represented by an orthogonal matrix $\mathbf{R}$ to obtain the projection $\mathbf{W} = \mathbf{SR}$ which solves (5). For this we compute the eigenvalue decomposition of $\overline{\mathbf{Z}}$ as $\overline{\mathbf{Z}}^T\overline{\mathbf{Z}} = \mathbf{Q}\Lambda\mathbf{Q}^T$. For high dimensional data we find the singular value decomposition (SVD) [35] of the centred data matrix $\overline{\mathbf{Z}}$ as $\mathbf{UDV}^T$.

We only keep the non-singular dimensions with eigenvalues above a certain threshold and obtain the whitening projection provided by $\mathbf{S} = \mathbf{UD}^{-1}$. Then SFA can be represented as:

$$\min_{\mathbf{R}} \operatorname{tr}(\mathbf{R}^T\mathbf{S}^T\dot{\mathbf{Z}}\dot{\mathbf{Z}}^T\mathbf{SR}) \quad \text{s.t.} \quad \mathbf{R}^T\mathbf{S}^T\mathbf{Z}\mathbf{Z}^T\mathbf{SR} = \mathbf{I} \tag{6}$$

The temporal derivatives are computed from $\mathbf{Z}$ using the forward temporal differences $\dot{\mathbf{Z}} = \mathbf{ZP}_n$ [36] where $\mathbf{P}_n$ is an $n \times (n-1)$ matrix with the elements $\mathbf{P}_n(i,i) = -1$ and $\mathbf{P}_n(i+1,i) = 1$. The zero mean constraint can be obtained by computing the centred data matrix such that $\overline{\mathbf{Z}} = \mathbf{Z} - \mathbf{1}_{n\times 1}\boldsymbol{\mu}_{\mathbf{Z}}$ where $\boldsymbol{\mu}_{\mathbf{Z}}$ is the mean of the data patterns, where $\mathbf{1}_{a\times b}$ is an $a \times b$ matrix with all elements set to 1.

Later to find the output of SFA, in the second step, the mean ED of $\dot{\overline{\mathbf{Z}}}\dot{\overline{\mathbf{Z}}}^T$ is computed such that $\mathbf{S}\dot{\overline{\mathbf{Z}}}\dot{\overline{\mathbf{Z}}}^T\mathbf{S} = \mathbf{RHR}^T$. Here $\dot{\overline{\mathbf{Z}}}$ represents the centred derivative data matrix computed by subtracting $\dot{\mathbf{Z}}$ from its mean $\boldsymbol{\mu}_{\dot{\mathbf{Z}}}$ such that $\dot{\overline{\mathbf{Z}}} = \dot{\mathbf{Z}} - \mathbf{1}_{n\times 1}\boldsymbol{\mu}_{\dot{\mathbf{Z}}}$. The output of the SFA is given by

$$\mathbf{y}_j = \mathbf{R}^T\left(\mathbf{S}^T\overline{\mathbf{z}}_j - \mathbf{S}^T\boldsymbol{\mu}_{\dot{\mathbf{z}}}\right) = \mathbf{W}^T\left(\overline{\mathbf{z}}_j - \dot{\boldsymbol{\mu}}_{\mathbf{z}}\right) \tag{7}$$

The ordering, in terms of slowness, of the functions y_j, is provided by the order of the components in $\mathbf{R}$ which is governed by the eigenvalues in $\mathbf{H}$. The slowest function is related to the smallest eigenvalue and the next larger eigenvalue gives the second slowest function, etc.

Unfortunately, this batch processing approach of SFA is not suitable for online applications because of its expensive storage limitation. Therefore, an incremental version of SFA is used to overcome this issue.

3.2. Incremental kernel slow feature analysis

Incremental Kernel Slow Feature Analysis updates slow features, incrementally so that it can process new input data. Inc-KSFA needs to update the mean of data and the whitening projections.

Suppose we have a data matrix $\mathbf{X}_A = [\mathbf{x}_1 \cdots \mathbf{x}_n] \in \mathbb{R}^{m \times n}$. In principle we non-linearly map $\mathbf{X}_A$ to a higher dimensional space $\mathcal{F}$ using the function $\mathbf{\Phi} : \mathbb{R}^m \longrightarrow \mathcal{F}$. Using $\mathbf{\Phi}$, we transform $\mathbf{X}_A$ into $\mathbf{\Phi}_A = [\phi(\mathbf{x}_1) \cdots \phi(\mathbf{x}_n)]$. The map $\mathbf{\Phi}$ is induced by a kernel function $\kappa(\cdot,\cdot)$ that allows us to evaluate inner products in new space $\mathcal{F}$

$$\kappa(\mathbf{a},\mathbf{b}) = \phi(\mathbf{a}) \cdot \phi(\mathbf{b}), \quad \text{with } \mathbf{a},\mathbf{b} \in \mathbb{R}^m \tag{8}$$

Let the considered mappings be elements of a reproduced kernel Hilbert space (RKHS) [37]. Consider the matrix $\boldsymbol{K} = \mathbf{\Phi}_A^T \mathbf{\Phi}_A$. By using $\kappa(\cdot,\cdot)$, $\mathbf{\Phi}_A^T \mathbf{\Phi}_A$ can be evaluated without having to perform the mapping $\mathbf{\Phi}$ since $\mathbf{\Phi}_A^T \mathbf{\Phi}_A$ contains only dot products between the $\phi(\mathbf{x}_i)$s. Through the application of kernel trick the optimization problem of SFA can be reformulated as

$$\min_R \operatorname{tr}(\mathbf{R}^T \mathbf{S}^T \dot{\overline{\mathbf{K}}} \dot{\overline{\mathbf{K}}}^T \mathbf{S}\mathbf{R}) \ \text{ s.t. } \ \mathbf{R}^T \mathbf{S}^T \overline{\mathbf{\Phi}}_A \overline{\mathbf{\Phi}}_A^T \mathbf{S}\mathbf{R} = \mathbf{I} \tag{9}$$

where $\dot{\overline{\mathbf{K}}}$ is the derivative of the centered kernel matrix $\overline{\mathbf{K}}$ such that $\dot{\overline{\mathbf{K}}} = \overline{\mathbf{K}}\mathbf{P}_n \mathbf{M}_{n-1}$. c $\mathbf{M}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_{n \times n}$.

Let us assume we are given a new data matrix $\mathbf{X_B} \in \mathbb{R}^{m \times i}$ where $\mathbf{\Phi}_B = \phi(\mathbf{X}_B)$. We want to incrementally find the whitening projections and update the slow features to incorporate new data patterns such that the whole information is represented by the concatenation of $\mathbf{\Phi}_A$ and $\mathbf{\Phi}_B$ as $\mathbf{X}_C = [\mathbf{\Phi}_A \ \ \mathbf{\Phi}_B]$. Let $\boldsymbol{\mu}_A$ and $\boldsymbol{\mu}_B$ be the mean of $\mathbf{\Phi}_A$ and $\mathbf{\Phi}_B$ respectively such that

$$\boldsymbol{\mu}_A = \mathbf{\Phi}_A \left(\frac{1}{n_A} \mathbf{1}_{n_A \times 1} \right) \ \text{ and } \ \boldsymbol{\mu}_B = \mathbf{\Phi}_B \left(\frac{1}{n_B} \mathbf{1}_{n_B \times 1} \right) \tag{10}$$

where n_A and n_B are the number of data samples contained in $\mathbf{\Phi}_A$ and $\mathbf{\Phi}_B$, respectively. Trivially, we can update the mean $\boldsymbol{\mu}_C$ of overall data $\mathbf{X}_C$ as [38]

$$\boldsymbol{\mu}_C = \frac{n_A}{n_A + n_B} \boldsymbol{\mu}_A + \frac{n_B}{n_A + n_B} \boldsymbol{\mu}_B \tag{11}$$

Let $\overline{\mathbf{\Phi}}_A$ and $\overline{\mathbf{\Phi}}_B$ be the centered data matrix of $\mathbf{\Phi}_A$ and $\mathbf{\Phi}_B$, respectively computed through subtracting the data from its mean [38]. Similarly, we update the centered matrix of overall data matrix $\mathbf{X}_C$ as: $\overline{\mathbf{X}}_C = \mathbf{X}_C - \boldsymbol{\mu}_C$.

Unit variance and incremental whitening: Consider the matrix $\overline{\mathbf{K}}$ such that $\overline{\mathbf{K}} = \overline{\mathbf{\Phi}}_A^T \overline{\mathbf{\Phi}}_A$. Suppose we have computed the eigenvalue decomposition of $\overline{\mathbf{K}}$ as $\mathbf{P}\mathbf{\Lambda}\mathbf{P}^T$. Via kernel SVD, we compute the singular value decomposition of $\overline{\mathbf{\Phi}}_A$ i.e.$[\overline{\mathbf{\Phi}}_A \mathbf{P} \mathbf{\Lambda}^{-\frac{1}{2}}][\mathbf{\Lambda}^{\frac{1}{2}}][\mathbf{P}^T] = \mathbf{U}\mathbf{D}\mathbf{V}^T$.

We want to compute the matrix $\mathbf{S}$ which whitens the overall data matrix. For this we need to incrementally compute the SVD of the concatenated matrix such that $\overline{\mathbf{X}}_C : [\overline{\mathbf{\Phi}}_A \ \ \overline{\mathbf{\Phi}}_B] = \mathbf{U}'\mathbf{D}'\mathbf{V}'^T$. Let $\tilde{\mathbf{\Phi}}_B$ be the component of $\mathbf{\Phi}_B$ orthogonal to $\mathbf{U}$ and $\mathbf{U}' = [\mathbf{U} \ \ \tilde{\mathbf{\Phi}}_B]$ (computed through QR decomposition [39]). The concatenated matrix can be represented in partitioned form as [40]

$$\overline{\mathbf{X}}_C : [\overline{\mathbf{\Phi}}_A \ \ \overline{\mathbf{\Phi}}_B] = \left[\mathbf{U} \ \ \tilde{\mathbf{\Phi}}_B \right] \begin{bmatrix} \mathbf{D} & \mathbf{U}^T \overline{\mathbf{\Phi}}_B \\ 0 & \tilde{\mathbf{\Phi}}_B (\overline{\mathbf{\Phi}}_B - \mathbf{U}\mathbf{U}^T \overline{\mathbf{\Phi}}_B) \end{bmatrix} \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^T \tag{12}$$

Let

$$\mathbf{\Psi} = \begin{bmatrix} \mathbf{D} & \mathbf{U}^T \overline{\mathbf{\Phi}}_B \\ 0 & \tilde{\mathbf{\Phi}}_B (\overline{\mathbf{\Phi}}_B - \mathbf{U}\mathbf{U}^T \overline{\mathbf{\Phi}}_B) \end{bmatrix} \tag{13}$$

which is a square matrix of size $k+i$ where k is the number of singular values in $\mathbf{D}$. Computing the SVD of $\mathbf{\Psi} = \hat{\mathbf{U}}\hat{\mathbf{D}}\hat{\mathbf{V}}^T$ to diagonalize the matrix and substituting into (12) yields the SVD of $\overline{\mathbf{X}}_C : [\overline{\mathbf{\Phi}}_A \overline{\mathbf{\Phi}}_B]$

$$\overline{\mathbf{X}}_C = \left(\left[\mathbf{U} \tilde{\mathbf{\Phi}}_B \right] \hat{\mathbf{U}} \right) \hat{\mathbf{D}} \left(\left[\hat{\mathbf{V}}^T \begin{bmatrix} \mathbf{V}^T & 0 \\ 0 & \mathbf{I} \end{bmatrix} \right] \right) = \mathbf{U}'\mathbf{D}'\mathbf{V}'^T \tag{14}$$

Since, we are only interested in computing $\mathbf{U}'$ and $\mathbf{D}'$, $\mathbf{V}'$, whose scales with the number of observed data, need not to be computed. Thus, we only need to calculate the SVD of matrix $\mathbf{\Psi}$ for the incremental update of the corresponding eigen-spectrum which is defined as: $\mathbf{U}' = \left[[\mathbf{U} \ \ \tilde{\mathbf{\Phi}}_B] \hat{\mathbf{U}} \right]$ and $\mathbf{D}' = \hat{\mathbf{D}}$. Therefore, the projection which whitens the signal is computed as $\mathbf{W} = \mathbf{U}'\mathbf{D}'^{-1}$.

Slow feature update: Suppose $\dot{\mathbf{\Phi}}_A$, $\dot{\mathbf{\Phi}}_B$ and $\dot{\mathbf{X}}_C$ represent the time derivatives of $\mathbf{\Phi}_A$, $\mathbf{\Phi}_B$ and $\mathbf{X}_C$, respectively. Let us assume there are $\dot{n}_A$ samples represented in $\dot{\mathbf{\Phi}}_A$. Similarly, we assume that new dataset $\dot{\mathbf{\Phi}}_A$ consists of $\dot{n}_B$ samples. We first find the centred data matrix of the newly observed elements represented by $\dot{\overline{\mathbf{\Phi}}}_B$. Thus from [41] we can update the overall data matrix as

$$\dot{\overline{\mathbf{X}}}_C \dot{\overline{\mathbf{X}}}_C^T = \dot{\overline{\mathbf{\Phi}}}_A \dot{\overline{\mathbf{\Phi}}}_A^T + \dot{\overline{\mathbf{\Phi}}}_B \dot{\overline{\mathbf{\Phi}}}_B^T + \frac{n_A n_B}{n_A + n_B} (\dot{\boldsymbol{\mu}}_A - \dot{\boldsymbol{\mu}}_B)(\dot{\boldsymbol{\mu}}_A - \dot{\boldsymbol{\mu}}_B)^T \tag{15}$$

where $\dot{\boldsymbol{\mu}}_A$ and $\dot{\boldsymbol{\mu}}_B$ represent the mean of time derivative patterns $\dot{\mathbf{\Phi}}_A$ and $\dot{\mathbf{\Phi}}_B$, respectively. The updated mean of overall data matrix is calculated analogous to (11). Finally we calculate the new feature function by computing the Eigen decomposition of $\mathbf{S}^T \dot{\overline{\mathbf{X}}}_C \dot{\overline{\mathbf{X}}}_C^T \mathbf{S}$ as $\mathbf{R}\mathbf{H}\mathbf{R}^T$. Via the kernel trick, the above process is rendered practicable without explicitly evaluating the mapping $\phi(\cdot)$.

3.3. Episodic segmentation

We have applied the above explained Incremental Kernel SFA algorithm for the segmentation of actions observed. SFA minimizes the slowness features present in the signal. Eq. (1) can be interpreted as the sum of squared Euclidean distance of the slow features computed between consecutive images. Suppose we have data $\mathbf{z}_i$, define δ as change detected in data after time t, as the squared Euclidean difference in slow features among its previous data

$$\delta_{l_t}(\mathbf{z}_i) = (\mathbf{z}_i - \mathbf{z}_{i-1})^T \mathbf{W}_{l_t} \mathbf{W}_{l_t}^T (\mathbf{z}_i - \mathbf{z}_{i-1}) \tag{16}$$

In order to compare the change between the current frame and the previous time frames, we need to compute the average change without keeping the previous signals in the memory. The sum of k largest eigenvalues in $\mathbf{H}$ is nearly equivalent to $\sum_1^{t-1} \delta_{l_{t-1}}(\mathbf{z}_i)$. The difference is caused by $\dot{\boldsymbol{\mu}}_{\mathbf{z}_{t-1}}$ and thus we compute the average as

$$\mu_{l_{t-1}} = \frac{\operatorname{tr}(\mathbf{H}_{l_{t-1}})}{\dot{n}_{\mathbf{z}_{t-1}}} + \dot{\boldsymbol{\mu}}_{\mathbf{z}_{t-1}}^T \mathbf{S}_{t-1} \mathbf{R}_{l_{t-1}} \mathbf{R}_{l_{t-1}}^T \mathbf{S}_{t-1}^T \dot{\boldsymbol{\mu}}_{\mathbf{z}_{t-1}} \tag{17}$$

The significant ratio (ζ) of the current and mean change is calculated to judge how substantial the change at current step is

$$\zeta = \frac{(t-1)\delta_{l_{t-1}}(\mathbf{z}_t)}{\mu_{l_{t-1}}} > \tau \tag{18}$$

where τ is the threshold value determined manually. Since the calculation of eigenvalues is done as a part of incremental Kernel SFA, thus, we do not require previous samples. The significance ratio of the current and average change is calculated to judge how substantial the change at current step is. The frames with large variations in the slow features are used to segment the data.

4. Topological Gaussian adaptive resonance hidden Markov model for incremental learning

In our proposed architecture, an HMM can be considered as graph whose nodes represent states attainable by the object and whose edges represent transitions between these states. The system is assumed to be at a particular state and to develop stochastically at discrete time steps by following the graph edges according to a transition probability. TGAR-HMM is described as a time evolving HMM with continuous observation variables, where the number of HMM states, structure and probability parameters are updated every time that a new observation sequence is available. Structurally, TGAR-HMM are similar to the standard HMMs, besides the fact that the transition structure and the

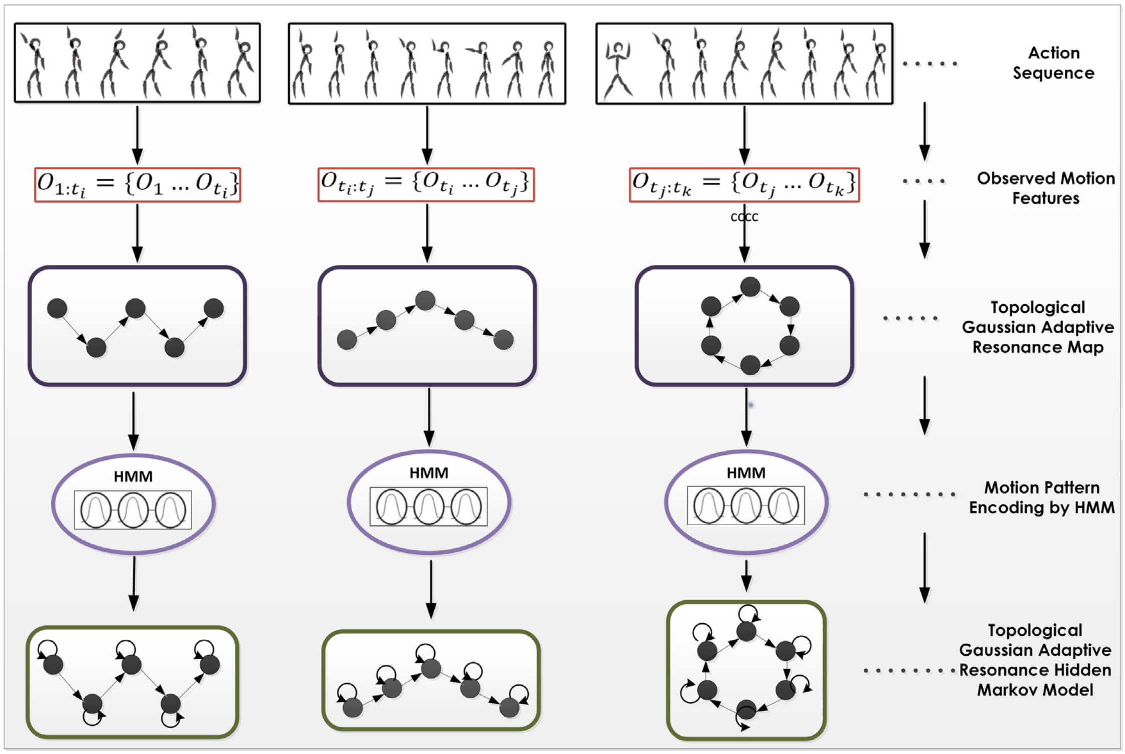
Action Sequence
$O_{1:t_i} = \{O_1 \dots O_{t_i}\}$
$O_{t_i:t_j} = \{O_{t_i} \dots O_{t_j}\}$
$O_{t_j:t_k} = \{O_{t_j} \dots O_{t_k}\}$
cccc
Observed Motion Features
Topological Gaussian Adaptive Resonance Map
HMM
Motion Pattern Encoding by HMM
Topological Gaussian Adaptive Resonance Hidden Markov Model

Fig. 2. Overview of TGAR-HMM learning architecture.

number of states are not constant but vary as more input observation sequences are processed. In addition to that the learning algorithm is able to incrementally update the model.

The main idea for developing the proposed probabilistic model is that the structure of the model should consider the spatial structure of the state-space discretization, where the transition among discrete states are only permitted if the corresponding regions are neighbors. Hence, structure learning essentially consists of estimating the suitable space discretization from the observed data and identifying neighboring regions. We have addressed this problem by proposing a topological map using the Topological Gaussian Adaptive Resonance Map (TGARM) algorithm. For parameter learning, we have utilized the incremental expectation-maximization (EM) approach to handle the changing number of states and with continuous observation.

The proposed learning algorithm is based on incrementally learning spatio-temporal behavioural sequences by developing the graph based structure of the behaviour patterns in the form of topological map. Fig. 2 shows the graphical representation of the learning architecture. The observed motion elements (joint angle values) are first organized through topological map consisting of nodes and edges. This map is then used to update the state structure for estimating the optimal number of states and the transition probabilities among the states. The on-line segmentation and learning algorithms together help in grouping different actions in memory.

4.1. Motion primitive modelling – topological Gaussian adaptive resonance structure

In this phase we extract useful motion sequences which are represented as the nodes linked with each other through edges. The function of the topological map is to develop a discrete structure of the continuous environment. The input to the learning algorithm consists of a series of discrete observation (i.e. joint angle values from sensor reading) describing the motion features.

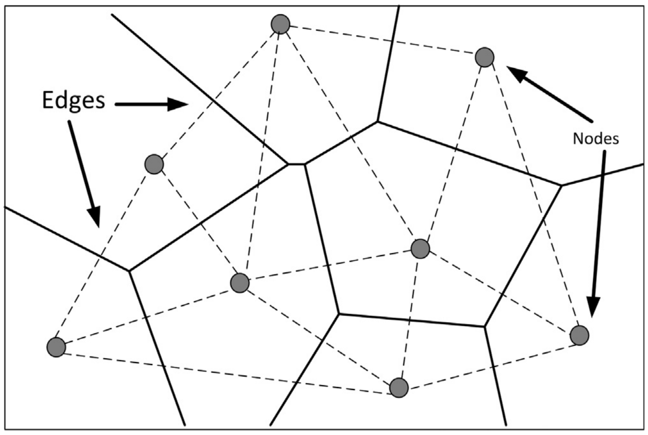

Fig. 3. Example of space distribution through topological mapping showing nodes connected through edges.

In addition, the observations are arranged in sequences $O_{1:T} = \{O_1, \dots, O_T\} = O_t$ such that every sequence described the trajectory of action.

We adopt the competitive Hebbian rule proposed by Martinez and Schulten [42] in topology preserving networks to build connections between neural nodes. The competitive Hebbian rule can be described as: for each input signal, connect the two closest nodes (measured by distance) by an edge. It has been proved that each edge of the generated graph belongs to the Delaunay triangulation corresponding to the given set of reference vectors, and the graph is optimally topology-preserving in a very general sense (Fig. 3).

We designed a topological map of the observed data through a novel algorithm called TGARM. The TGARM model is based on the Gaussian mixture model of the input space where each Gaussian component represents a category node. TGARM has following properties:

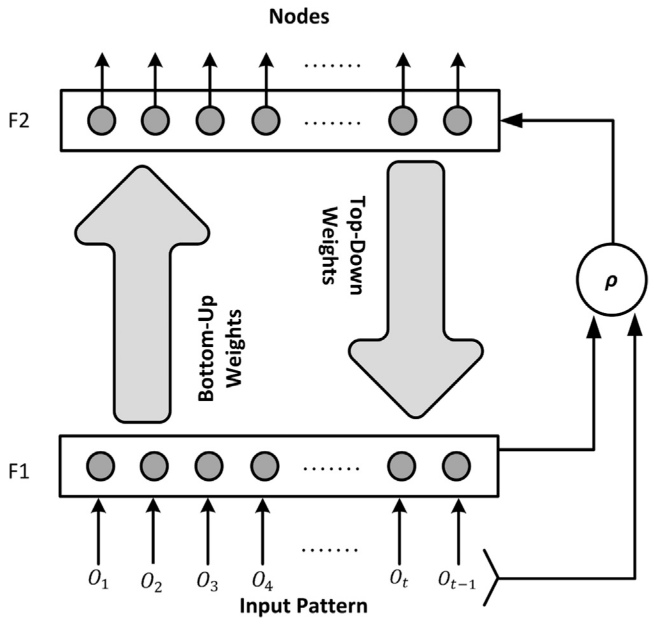

Fig. 4. Architecture for creation of topological map.

- The structure grows incrementally by incorporating new knowledge without defiling the previously learned data and adaptively responds to the information acquired from the environment. This avoids catastrophic forgetting i.e., retain previously learned knowledge effectively.
- Each node in the topological map is defined as a Gaussian distribution, with mean and covariance.
- The parameters of each node in the topological map are updated for each input sample observed rather than acquiring an entire data set.

Fig. 4 shows the structure of topological map model. The inputs are received from the robot sensors (for example joint angle values). Each input neuron is connected to the output neurons through the bottom-up weights; conversely, each neuron in the output layer is connected to the input layer through the top-down weights. The bottom-up weights or the activation value provides likelihood that an input pattern is a probable candidate for being a node or a category whereas the matching function provides a confidence measure about the top-down weights. This confidence measure is defined by the vigilance parameter, ρ. The output layer creates a topological structure of the input data.

Each node weights are defined by a vector M_j and matrix Γ_j representing its mean and covariance, respectively. Another associated parameter with each node is the node count or learning rate n_j representing the number of nodes or number of input patterns learned by TGARM. The network is initialized with two parameters: the baseline vigilance parameter $\overline{\rho}$ which takes the values within interval $(0,1)$ and the initial covariance matrix, γ. The variables that define the contents of the nodes are summarized in Table 1. At the beginning of learning, the mean value is initialized with the input parameter and the covariance matrix is initialized with some suitable value. This algorithm allows the observer to incrementally learn and update the structure of the model based on the observed motion patterns.

During learning a winning node ω_J is selected from an input pattern based on the highest probability. Since each node is represented by Gaussian components defined by the mean values, and the covariance matrix Γ_j, therefore, the conditional density of O_t given the winning node j or the bottom-up input activation value for a node is calculated as

Table 1
Topological Gaussian Adaptive Resonance Hidden Markov Model (TGAR-HMM) parameters.

Parameters	Meaning
ω_J	Winning node
M_j	Mean value – weight parameter
Γ_j	Covariance matrix initialized by γ – weight parameter
ρ	Vigilance parameter initialized by $\overline{\rho} \in (0,1)$
n_j	Node count or learning rate
π_i	State prior probability
a_{ij}	State transition probability
m_i	Mean value for Gaussian in HMM – $(m_i = M_i)$
C_i	Covariance matrix for each node in HMM – $(C_i = \Gamma_i)$
B	Observation probability distribution – $N(O_t \mid m_i, C_i)$

$$p(O_t \mid j) = \frac{1}{(2\pi)^{N/2} |\Gamma_j|^{1/2}} \exp\left[-\frac{1}{2}(O_i - M_j)^T \Gamma_j^{-1}(O_i - M_j)\right] \tag{19}$$

where N is the dimensionality of the input motion patterns.

For each input pattern the activation value is calculated using Eq. (19) and the neuron with highest activation value is selected which determines the node with the highest probability

$$J = \arg\max_j p(O_t \mid j) \tag{20}$$

But the node represented by its weights $(w_J = (M_J, \Gamma_J))$ is only allowed to be updated if the vigilance criterion or matching between the given input and the selected winner node is fulfilled. A node w_J passes the vigilance criterion if its match function value exceeds the vigilance parameter value ρ, that is if

$$\exp\left[-\tfrac{1}{2}(O_t - M_j)^T \Gamma_j^{-1}(O_t - M_j)\right] \geq \rho \tag{21}$$

The vigilance is a measure of similarity between the input and the node's mean relative to its standard deviation. If the winning node fails to pass the vigilance test (Eq. (21)), the current winner node is disqualified and its activation value is reset. Then, the observation pattern is searched for the new winning best-matching neuron. If no satisfactory neuron is found, a new neuron representing the input pattern with $n_J = 0$ is integrated satisfying the vigilance criterion.

When the winning neuron, satisfying the vigilance condition representing the input pattern is selected, its parameters i.e. count, mean, and variance is updated

$$n_J = n_J + 1 \tag{22}$$

$$\boldsymbol{\xi}_J = \left(1 - \frac{1}{n_J}\right)\boldsymbol{\xi}_J + \left(\frac{1}{n_J}\right)\mathbf{O}_t \tag{23}$$

$$\boldsymbol{\Gamma}_J = \left(1 - \frac{1}{n_J}\right)\boldsymbol{\Gamma}_J + \left(\frac{1}{n_J}\right)(\mathbf{O}_t - \boldsymbol{\xi}_J)(\mathbf{O}_t - \boldsymbol{\xi}_J)^T \tag{24}$$

The learning algorithm grows its neural structure starting with the first node. The algorithm for topological mapping is summarized in Algorithm 1. Each time a behaviour pattern is observed, it is encoded as a Gaussian node in the structure. This algorithm allows the observer to incrementally learn and update the structure of model based on the observed motion patterns. When the resonating neuron is determined, a lateral connection or an edge is established between the current and previous winner node. This mechanism will provide a stable architecture for providing a link between the previously learned knowledge and also integrate newly observed data in to map temporal correlation between them.

The performance of the TGARM depends on two parameters: the vigilance parameter ρ, and the initial covariance matrix γ. The vigilance parameter directly influences the formation of new nodes when novel information is detected. For higher values of the vigilance parameter, the system becomes more sensitive to the changes in the input and the network becomes complicated. On the other hand, for the lower values of the vigilance parameter the system becomes less sensitive and also becomes less complicated and faster. Therefore, the decision about the vigilance parameter value greatly influences the convergence and recognition properties of the system. Furthermore, the generalization performance of the network is greatly affected by the selection of vigilance parameter.

4.2. Incremental learning

After updating the structure of the model, motion patterns are learned through the probabilistic module. This is done through Hidden Markov Model (HMM). HMM is a doubly stochastic model that consists of states which are not directly observed. HMM explicitly include time resulting in efficient learning of temporal sequences and compensate for uncertainties. Each state in the HMM emits an observation as output which infers the most likely dynamical system. Each state is connected by transitions between the states and generates an output pattern. Each motion pattern is encoded in the HMM model. In order to select the appropriate structure of the HMM or selecting the optimum numbers of HMM states, Topological Gaussian Adaptive Resonance is employed. After updating the model structure and encoding the observed motion patterns, the remaining parameters of HMM, such as transition probability and prior probabilities is updated using Expectation Maximization (EM) algorithm.

In our model the transition between different states is characterized by the edges connecting neighbouring states, thus, the system can only make transition between the neighbouring states only. Since behaviour patterns are ordered sequences composed of different atomic actions or motion primitives, each motion element is encoded as a Gaussian. Therefore, we used left-to-right HMM model structure for representing observed motion patterns to allow the data to flow in a sequential order in forward direction of time. In left-to-right HMM the self-transition loop is also allowed.

An HMM is characterized by the following parameters:

- *State prior probabilities* – $\pi_i = P(s_0 = i)$: Represents the prior probability for the corresponding state.
- *State transition probability matrix* – $a_{ij} = P(s_{t+1} = j \mid s_t = i)$: Represents the probability of transition from state i to state j.
- *Observation probability distribution* – $B = P(O_t \mid s_t = i)$. The probability distribution of observation vector from state i. This distribution is represented by Gaussian function denoted by the parameters $N(O_t \mid m_i, C_i)$, where m_i and C_i is mean vector and the covariance matrix for the i-th state in HMM.

These HMM parameters are denoted as $\lambda = \{\pi, A, B\} = \{\pi, A, m, C\}$. Each hidden state in HMM encodes and abstracts an observed motion pattern where a sequence of motion patterns is estimated using the transition between these hidden states.

Algorithm 1. Algorithm for Topological Gaussian Adaptive Resonance Structure.

```
Require:
    Observation Vector $O_t$
    Initial Covariance Matrix $\Sigma$
    Baseline Vigilance Parameter $\overline{\rho}$
Ensure:
    Nodes $\mathcal{N}$
    Edges $\mathcal{E}$
Require:
1: Input the observation vector $O_t$.
2: if there is no node in the network then
3:     Add $O_t$ in the network as new node. $\mathcal{N} \leftarrow \mathcal{N}_i \cup \{O_t\}$
4:     Update the weights of the node $\mathcal{N}$ using (22)–(24)
5: else
6:     Determine the winner node $\omega_J$ (19)
7:     Determine the vigilance criterion for $\omega_J$ (21)
8:     if calcVig $< \overline{\rho}$ i.e. the observation vector fails the vigilance test then
9:         Reset the winner node and find a new winner from observation vector.
10:        Update the weights of previous winner node.
11:        Obtain the new observation vector $O_t$.
12:        If the learning is not completed, go to Step 6 to process the next observation.
13:    else
14:        if calcVig $> \overline{\rho}$ i.e. the node passed the vigilance test then
15:            Add as a new node $\mathcal{N} \leftarrow \mathcal{N} \cup \{O_t\}$
16:            Update the weights of the $\omega_J$ (winner node) $\mathcal{N}(n_J, M_J, \Gamma_J)$ using Eqs. (22)-(24)
17:            Add edge between the previous winner and current winner nodes $\mathcal{E} \leftarrow \mathcal{E} \cup \{(prevWinner, \omega_J)\}$
18:        end if
19:    end if
20: end if
```

4.3. Updating structure and parameters of HMM

In this section we will explain how the parameters and structure of HMM is updated through the use of TGARM. After updating the topological map, the structure of HMM is also updated based on the added nodes and edges. The structure of HMM is updated whenever a new behaviour is observed. Corresponding to every node added in the topological map, a state in the HMM is also added. Each added state is initialized with the prior probability $\pi_i = \pi_0$ and self-transition probability $a_{i,i} = a_0$, where i represents the new node. Similarly, for addition of every new node and the new edges (i,j) connecting these nodes, the transition probabilities are also initialized with state transition probability value $a_{i,j} = a_0$.

After updating the HMM structure, the parameters of HMM are also updated. The mean and the covariance values related to each Gaussian observation are updated during the structure (topological map) updating process discussed in previous section. The same values are used by the HMM. However, the remaining parameters such as transition probability and state prior probabilities need to be re-estimated. These parameters are updated using the Expectation Maximization algorithm. Traditionally, Baum Welch algorithm [13] which is a type of EM algorithm is used for learning the initial state probability distribution and the sate transition model.

Re-esimate the transition probability and state prior probability using

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i) \beta_t(i)} \tag{25}$$

$$\overline{\pi}_i = \frac{\alpha_1(i)\beta_1(i)}{P(O \mid s_i)} \tag{26}$$

In Eqs. (25) and (26) the α_i and β_i represent the forward and

Table 2
Recursive computation of forward and backward variables.

Algorithm	Computation
Forward variable	$\alpha_i(1) = \pi_i b_i(O_1)$
	$\alpha_j(t+1) = [\sum_{i=1}^{N} \alpha_i(t) a_{ij}] b_j(O_{t+1})$
Backward variable	$\beta_i(T) = 1$
	$\beta_i(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_j(t+1)$

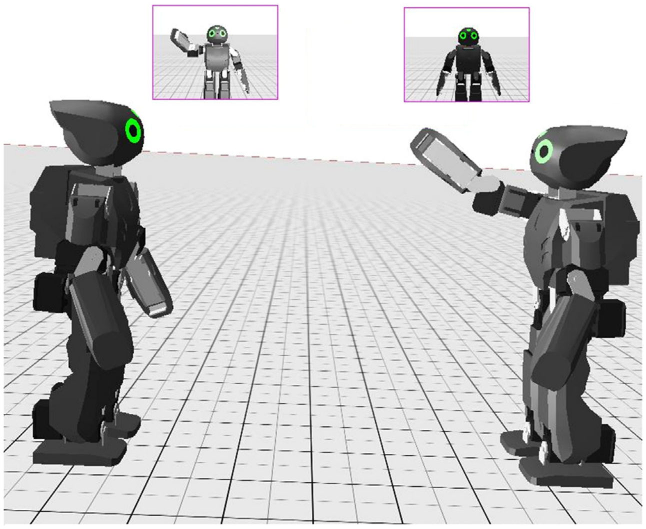
Fig. 5. Simulation environment for experimentation consisting of two robots. One acts as demonstrator (left robot), while second acts as observer (right robot).

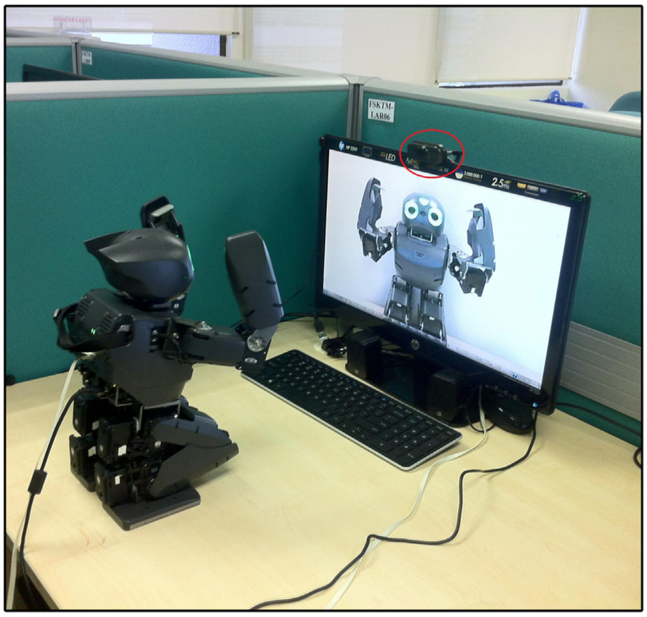

Fig. 6. Robot Setup.

backward variables [13]. Table 2 explains the recursive computation of these variables. $P(O|S_i)$ in Eq. (26) determines the joint observation probability.

Table 3
Summary of different types of actions performed and their identification accuracy.

Motion description	Label	Segmentation accuracy (%)	
		Start of action	End of action
Right Arm Raise 180°	RAR180	99.199	98.281
Right Arm Lower 180°	RAL180	99.316	97.642
Left Arm Raise 180°	LAR180	99.419	98.835
Left Arm Lower 180°	LAL180	89.611	97.721
Both Arms Raise 180°	BAR180	99.601	88.872
Both Arms Lower 180°	BAL180	99.633	98.961
Left Arm Raise 90°	LAR90	99.623	98.843
Left Arm Lower 90°	LAL90	99.376	99.125
Right Arm Raise 90°	RAR90	99.688	96.341
Right Arm Lower 90°	RAL90	99.809	99.918
Both Arms Raise 90°	BAR90	99.765	99.496
Both Arms Lower 90°	BAL90	99.791	99.576
Raise Both Arm Front	RBAF	99.814	99.829
Lower Both Arms Front	LBAF	99.791	99.363
Raise Left Arm Front 90°	RLAF90	99.874	99.051
Lower Left Arm Front 90°	LLAF90	89.909	99.263
Raise Right Arm Front	RRAF	99.883	99.524
Lower Right Arm Front	LRAF	99.832	99.437
Raise Right Arm Front 90°	RRAF90	99.886	99.739
Lower Right Arm Front 90°	LRAF90	99.891	89.444
Raise Left Arm Front	RLAF	99.892	99.841
Lower Left Arm Front	LLAF	99.924	99.479
Raise Both Arms Front 90°	RBAF90	99.995	99.601
Lower Both Arms Front 90°	LBAF90	99.995	99.476
Bow Down	BOD	99.875	99.877
Bow Up	BOU	99.939	99.880
Squat Down	SQD	99.767	99.942
Squat Up	SQU	100	99.850
Right Kick Extend	RKE	99.775	99.889
Left Kick Extend	LKE	99.834	99.089

In order to update the parameters incrementally for new observed data, an incremental learning rule is applied as follows:

$$\overline{a}_{ij} = \frac{\overline{a}_{ij} + (N_p - 1)a_{ij}}{N_p} \tag{27}$$

$$\overline{\pi}_i = \frac{\overline{\pi}_i + (N_p - 1)\pi_i}{N_p} \tag{28}$$

where N_p is the number input patterns that has been observed until the current time.

5. Experimental setup

The tests of proposed algorithm were performed through simulation on open humanoid platform DARwIn-OP,[1,2] developed by Robotis Co. Ltd. For simulation purposes we have used the Webots [43] simulator. DARwIn-OP has 20 degrees of freedom with 3-axis accelerometer.

The first set of experiments was conducted in a simulation environment to test the efficacy of the proposed framework. The validation of our mirror image based self-learning approach was performed on a test-bed consisting of two DARwIn-OP robots (Fig. 5). Just as humans perceive their reflection in the mirror similar to themselves, similarly, in our simulation environment, one robot acts as a demonstrator while the other robot observed these demonstrated actions as the mirror image reflection of the demonstrator. The algorithm was tested on video sequences of different actions captured by the robot's camera. At the same time the joint angle values of the demonstrator (self-performed actions)

[1] Dynamic Anthropomorphic Robot with Intelligence (DARwIn-OP).
[2] http://www.robotis.com/xe/darwin_en

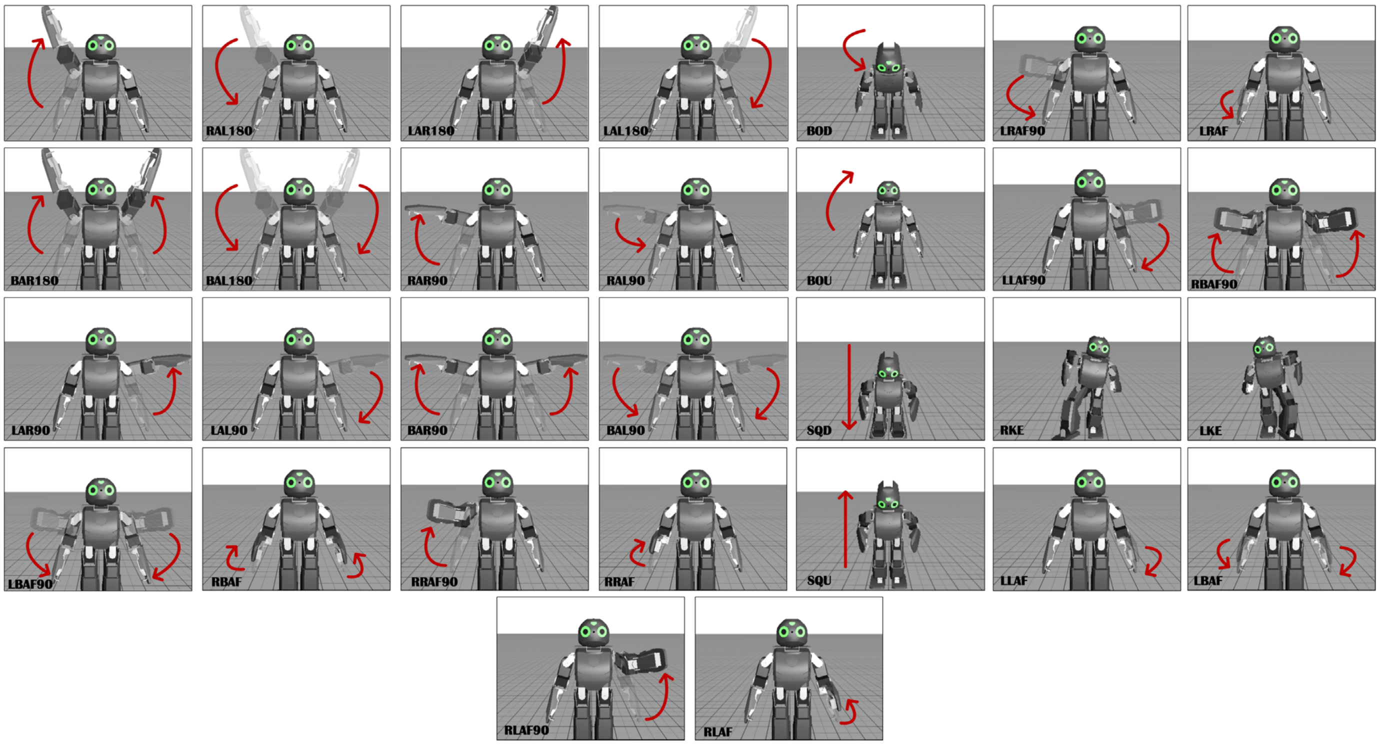

Fig. 7. Different samples of actions performed by the robot during experimentation.

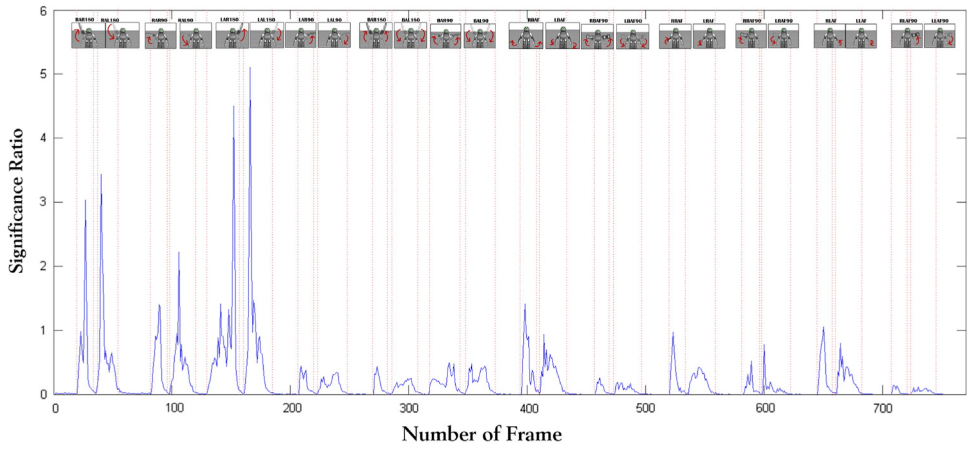

Fig. 8. Output of the segmentation algorithm through Incremental Kernel Slow Feature Analysis.

are also recorded by the observer. These joint angle values are used for learning purposes.

To assess the application for computational framework in real-life situations, we used a real robot environment consisting of real DARwIn-OP humanoid robot (Fig. 6). A camera is positioned on the monitor screen to create the mirror reflection environment. The camera on the monitor projects the robot's action on the screen, while the robot's own monocular camera observes these projected self-image. During body babbling phase, the robot observes its own projected image on the screen during the random generation of actions, and process the observed self-images for segmentation. In parallel the joint angles are also learned by the robot for each action performed.

6. Results and discussion

In this section we will discuss the experimental analysis of the proposed approach. Table 3 summarizes the types of actions performed for testing. The dataset is divided into two groups of multiple actions. The first group of actions is performed with fixed interval between the actions, while, the second group is executed fluidly. These two groups are performed with varying speeds. The experiments were performed with three varying speeds of 1.0 rad/s, 2.5 rad/s and 4.0 rad/s. The camera captured data at 30 frames per second. Fig. 7 shows the visualization of actions performed by the demonstrator. Each image in the figure shows different frames extracted from the action sequences.

The raw image sequences acquired from robot camera are processed for motion primitive segmentation. Initially, the demonstrator is standing still and no feature points exhibit significant change. As soon as the robot starts moving the joints, change in feature values is recorded and the significant ratio is computed. Based on the significant ratio, the start and end of an action are computed. Fig. 8 shows the result of the Incremental Kernel SFA algorithm. Whenever a significant change in the

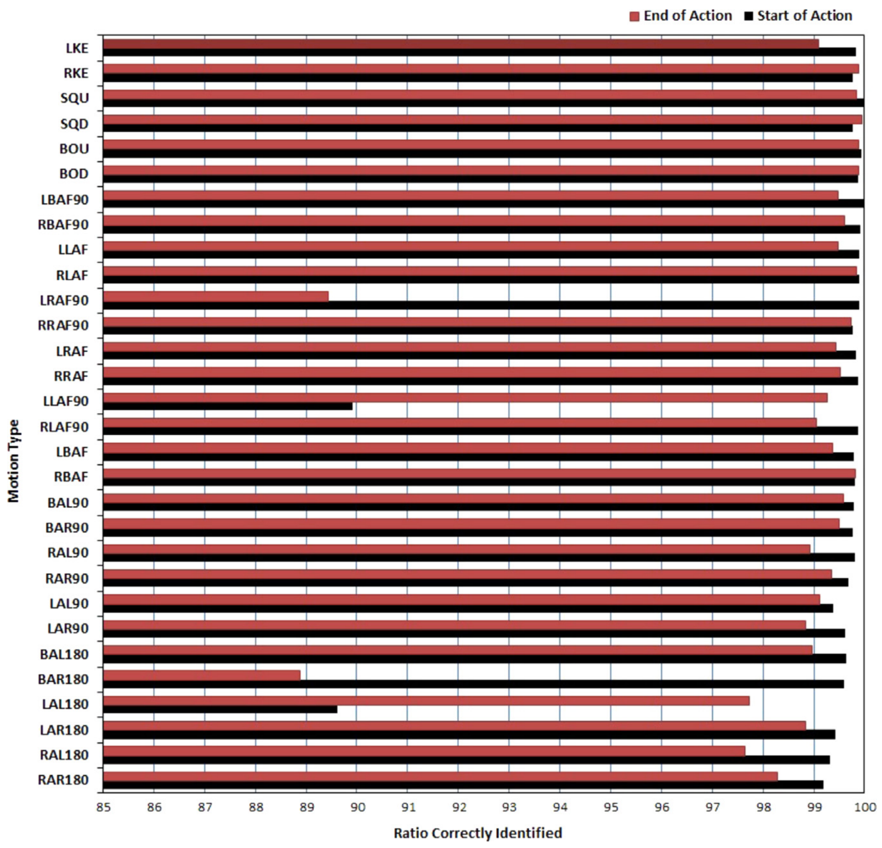

Fig. 9. Average accuracy of segmentation results between the joints based segmentation and Incremental Kernel SFA.

captured frames is detected, the significant ratio value is increased when the action is completely performed, thus performing the segmentation on-line. The threshold value is selected to be $\tau = 0.0125$. Fig. 8 shows the change in significant ratio along with the number of frames to segment the observed motion patterns into episodes of action. The segmentation algorithm commences with no a-priori knowledge of the motion patterns and the observed data is being segmented on-line by analysing the incoming data stream. The experiments were performed with varying speeds. Since, the segmentation process is based on processing the captured images, therefore, segmentation is not effected by the change in the speed of joints.

Initially, the demonstrator is standing still. As soon as the robot starts moving the joints, change in feature values are recorded and the significant ratio is computed. Based on the significant ratio, the start and end of an action is computed. Fig. 8 shows the result of Incremental Kernel SFA algorithm. Whenever a significant change in the captured frames is detected, significant ratio value is increased until the action is completely performed, thus performing the segmentation on-line. Fig. 8 shows the change in significant ratio along with the number of frames to segment the observed motion patterns into episodes of action. In this figure the dotted line shows the manual segmentation.

To compare the performance of the proposed algorithm, segmentation is performed based on the change in recorded joint angle values. Fig. 9 shows the accuracy of segmentation output for different types of actions. The average segmentation ratio is computed for each action performed multiple times and summarized in Table 3. As can be seen from these results, the segmentation of the actions performed produces less error even at the critical points where the actions transit from one motion to other.

Table 4
Comparison of segmentation results between Optical Flow based [12] and proposed Inc-KSFA based segmentation.

Action type	Segmentation accuracy (%)			
	Optical flow [12]		**Inc-KSFA** [Proposed]	
	Start of action	**End of action**	**Start of action**	**End of action**
LAL180	98.708	100	99.510	98.956
LAR180	69.298	100	90.161	98.372
BAL90	91.250	95.152	99.871	100
BAR90	86.103	94.215	100	99.711
RAL90	95.647	88.051	99.901	100
RAR90	82.558	94.421	99.595	99.454
BAL180	98.569	82.868	99.888	98.837
BAR180	79.202	99.843	99.715	99.832
LKE	62.638	77.081	99.549	98.281
RKE	98.089	35.661	99.440	97.524
LAL90	99.956	98.765	99.515	99.905
LAR90	70.080	97.198	99.798	98.508
BOU	100	48.671	97.842	99.990
BOD	68.266	100	99.520	98.374
RAL180	95.672	80.842	99.671	99.356
RAR180	80.442	85.035	99.472	99.671

In order to assess the effectiveness of the proposed architecture we have performed a comparison of our approach against the most similar approach developed by Kulic et al. [12]. Comparing heterogeneous techniques, even if they solve a common problem, is a difficult task. Often their theoretical bases are too different, making it difficult to evaluate them fairly. Here we used segmentation accuracy a measure, despite its simplicity, still provide useful indicator about the parsimony of the models produced by the different approaches.

The segmentation by Kuic et al. [12] is based on the analysis of optical flow in the video sequence. The proposed algorithm searches for coherent clusters of optical flow, and generates segmentation points in those frames where there are significant changes to the coherent clusters. The result for the comparison for accuracy of motion primitive segmentation is shown in Table 4. As can be seen in Table 4, the Inc-KSFA algorithm achieves a correct segmentation rate comparable the optical flow based approach. For evaluation purposes we have used the data set consists of 17 min of continuous whole body motion data of a single human subject. The video of the dataset can be accessed on-line: https://ece.uwaterloo.ca/dkulic/TRO2009SuppMaterial.html

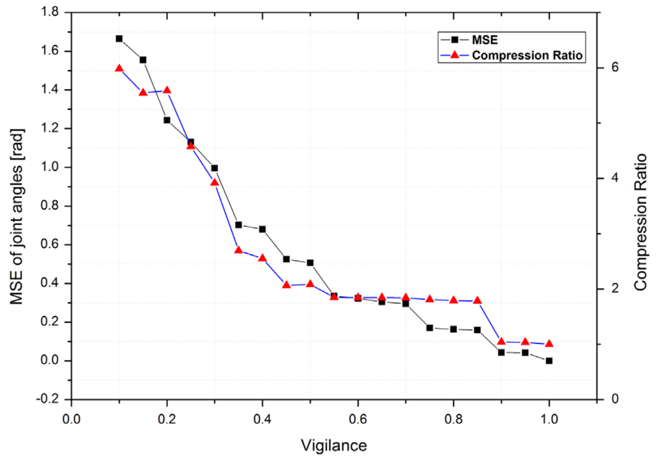

Fig. 10. Plot for compression ratio and average mean square error for different values of vigilance parameter.

For learning part of the algorithm the joint angle values are utilized. The joint angles values are clustered based on the start and end of an action obtained from the Incremental Kernel SFA. Once the start of an action is detected, TGAR-HMM starts adding the joint angle values as motion primitives in the form of nodes linked with edges. The learning of that particular action is completed when the end of that action is detected. We used left-to-right HMM model structure for representing observed motion patterns to allow the data to flow in a sequential order in forward direction of time. In left-to-right HMM the self-transition loop is also allowed. After learning, the specific episode of action is clustered according to its label. Thus, for each cluster, the observed action is learned by the observer in an incremental manner.

As discussed earlier that the performance of TGARM greatly depends on the selection of values for the vigilance parameter and initial covariance matrix. For vigilance parameter, the value is chosen to be $\rho = 0.85$, for fast learning and utilizing labelled nodes. Similarly, the initial covariance matrix determines the isotropic spread in feature space of a new nodes distribution. For large values of γ, the learning will be slow with fewer nodes, while for smaller values of γ, the training will be faster with more number of nodes. The initial covariance matrix is selected in an adhoc fashion through trial and error.

$$CompressionRatio = \frac{No.\ of\ Samples}{No.\ of\ Nodes} \tag{29}$$

Fig. 10 shows the effect of selecting different values of vigilance parameter on the compression rate and generalization error. As the value of vigilance parameter is increased, the mean square error among the observed and learned values decreases by adding more number of nodes to the network. For higher values of vigilance parameter, the value of compression ratio (29) is decreased

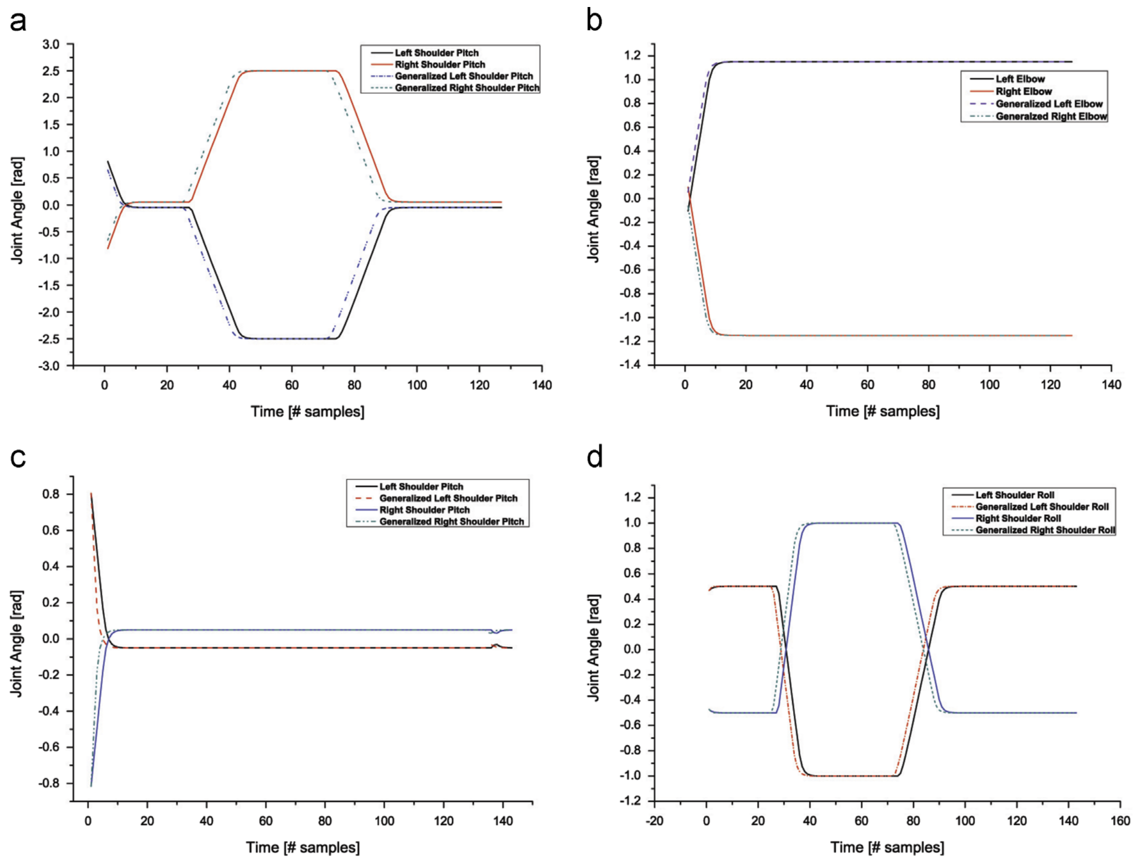

Fig. 11. Plot of original and learned motion patterns for (a) and (b) Raising and Lowering Both Arms (RBA180 – LBA180); (c) and (d) Raising and Lowering Both Arms (RBA90 – LBA90).

resulting in encoding motion patterns as close as possible to the observed motion.

We evaluated the performance of the system using error between the demonstrated and learned motion to determine the appropriate adapting learned motion. The mean error is used as a metric to evaluate the sustainability of the learned motion with respect to the demonstrated motion. This error metric provides a measure for the evaluation of generalization capability of proposed learning model. Fig. 11 shows generalization results for the action of raising both arms (RBA) and lowering both arms (LBA). Fig. 11 shows the results for shoulder movements.

7. Conclusion

In this paper we have developed the on-line segmentation method utilizing visual data only instead of relying on the kinematic data such as joint angles. The proposed Incremental Kernel SFA algorithm searches for the suitable slow features in the images. After finding these features the significant ratio among the captured frames was calculated. Incremental Kernel SFA need not to calculate this significant ratio on all the previous data, which makes it suitable for online applications. After segmentation the observed motion patterns are encoded through TGAR-HMM. The structure of model aggregates the information as observed and organizes the information in an efficient growing and self-organizing manner. The TGAR-HMM model efficiently learns and encodes the spatio-temporal patterns and computes the probability that observation sequences could be generated. Secondly, the novel HMM architecture adaptively selects the models structure based on the observed data and is not pre-defined based on some prior knowledge. The proposed model has been tested for different kinds of behaviour on an open humanoid platform DARwIn-OP. The proposed algorithm achieves the better performance both in segmentation and generalizing the observed motion patterns.

The architecture presented here is inevitably limited in scope, allowing for improvement in future work. First this system assumes that an observer always stands face to face with a demonstrator, and this system does not have concept about the translation or rotation to the ground of the demonstrator. A key piece of our learning system is the selection of vigilance parameter which effects the performance of the system. In future work, the testing of the proposed system using the human–robot experiment will be considered.

Acknowledgements

This research is supported by High Impact Research MoE Grant UM.C/625/1/HIR/MOHE/FCSIT/10 (Mirror Neuron and Quantum Mechanics Based Neural Architecture for Humanoid) from the Ministry of Education Malaysia.

References

[1] G. Aschersleben, Early development of action control, Psychol. Sci. 48 (4) (2006) 405–418.
[2] G. Butterworth, Imitation in Infancy – Cambridge Studies in Cognitive Perceptual Development, Cambridge University Press, New York, NY, US, 1999, 'Neonatal imitation: existence, mechanisms and motives', pp. 63–88.
[3] A. Meltzoff, The like me framework for recognizing and becoming an intentional agent, Acta Psychol (Amst) 124 (1) (2007) 26–43.
[4] A.N. Meltzoff, J. Decety, What imitation tells us about social cognition: a rapprochement between developmental psychology and cognitive neuroscience, Philos. Trans. R. Soc. London B: Biol. Sci. 358 (1431) (2003) 491–500.
[5] S. Schaal, Is imitation learning the route to humanoid robots? Trends Cognit. Sci. 3 (6) (1999) 233–242.
[6] W. Takano, Y. Nakamura, Humanoid robot's autonomous acquisition of proto-symbols through motion segmentation, in: 2006 6th IEEE-RAS International Conference on Humanoid Robots, 2006, pp. 425–431.
[7] D. Kulic, D. Lee, C. Ott, Y. Nakamura, Incremental learning of full body motion primitives for humanoid robots, in: 8th IEEE-RAS International Conference on Humanoid Robots, 2008. Humanoids 2008, 2008, pp. 326–332.
[8] F. Meier, E. Theodorou, S. Schaal, Movement segmentation and recognition for imitation learning, in: Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS), 2012, pp. 761–769.
[9] Y. Mohammad, T. Nishida, S. Okada, Unsupervised simultaneous learning of gestures, actions and their associations for human–robot interaction, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009, 2009, pp. 2537–2544.
[10] Y. Mohammad, T. Nishida, Fluid imitation, Int. J. Soc. Robot. 4 (4) (2012) 369–382.
[11] Y. Mohammad, T. Nishida, On comparing SSA-based change point discovery algorithms, in: 2011 IEEE/SICE International Symposium on System Integration (SII), IEEE, 2011, pp. 938–945.
[12] D. Kulic, D. Lee, Y. Nakamura, Whole body motion primitive segmentation from monocular video, in: IEEE International Conference on Robotics and Automation, 2009. ICRA'09, 2009, pp. 3166–3172.
[13] L. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (2) (1989) 257–286.
[14] T. Inamura, I. Toshima, H. Tanie, Y. Nakamura, Embodied symbol emergence based on mimesis theory, Int. J. Robot. Res. 23 (4–5) (2004) 363–377.
[15] S. Calinon, A. Billard, Stochastic gesture production and recognition model for a humanoid robot, In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings, vol. 3, 2004, pp. 2769–2774.
[16] A.G. Billard, S. Calinon, F. Guenter, Discriminative and adaptive imitation in uni-manual and bi-manual tasks, Robot. Auton. Syst. 54 (5) (2006) 370–384.
[17] D. Kulic, W. Takano, Y. Nakamura, Incremental on-line hierarchical clustering of whole body motion patterns, in: The 16th IEEE International Symposium on Robot and Human Interactive Communication, 2007. RO-MAN 2007, 2007, pp. 1016–1021.
[18] S. Niekum, S. Chitta, B. Marthi, S. Osentoski, A.G. Barto, Incremental semantically grounded learning from demonstration, in: Robotics: Science and Systems 2013, 2013.
[19] J. Butterfield, S. Osentoski, G. Jay, O. Jenkins, Learning from demonstration using a multi-valued function regressor for time-series data, In: 2010 10th IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2010, pp. 328–333.
[20] S. Calinon, A. Billard, Incremental learning of gestures by imitation in a humanoid robot, in: 2007 2nd ACM/IEEE International Conference on Human–Robot Interaction (HRI), 2007, pp. 255–262.
[21] S.S. Seyhan, F.N. Alpaslan, M. Yavas, Simple and complex behavior learning using behavior hidden Markov model and Cobart, Neurocomputing 103 (0) (2013) 121–131.
[22] M. YavaÅ, F.N. Alpaslan, Hierarchical behavior categorization using correlation based adaptive resonance theory, Neurocomputing 77 (1) (2012) 71–81.
[23] D. Vasquez, T. Fraichard, C. Laugier, Incremental learning of statistical motion patterns with growing hidden markov models, IEEE Trans. Intell. Transp. Syst. 10 (3) (2009) 403–416.
[24] M. Tscherepanow, Topoart: a topology learning hierarchical art network, in: K. Diamantaras, W. Duch, L. Iliadis (Eds.), Artificial Neural Networks ICANN 2010, Lecture Notes in Computer Science, vol. 6354, Springer, Berlin, Heidelberg, 2010, pp. 157–167.
[25] S. Okada, Y. Kobayashi, S. Ishibashi, T. Nishida, Incremental learning of gestures for human–robot interaction, AI Soc. 25(2) (2010) 155–168.
[26] L. Wiskott, T.J. Sejnowski, Slow feature analysis: unsupervised learning of invariances, Neural Comput. 14 (4) (2002) 715–770.
[27] M. Franzius, H. Sprekeler, L. Wiskott, Slowness and sparseness lead to place, head-direction, and spatial-view cells, PLoS Comput. Biol 3 (8) (2007) e166, http://dx.doi.org/10.1371/journal.pcbi.0030166.
[28] P. Berkes, Pattern recognition with slow feature analysis (February 2005) http://cogprints.org/4104/.
[29] Z. Zhang, D. Tao, Slow feature analysis for human action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 436–450.
[30] F. Nater, H. Grabner, L.V. Gool, Temporal relations in videos for unsupervised activity analysis, in: British Machine Vision Conference, 2011.
[31] M.D. Luciw, V.R. Kompella, S. Kazerounian, J. Schmidhuber, An intrinsic value system for developing multiple invariant representations with incremental slowness learning, Front. Neurorobot 7 (9) (2013).
[32] M. Luciw, V.R. Kompella, J. Schmidhuber, Hierarchical incremental slow feature analysis, in: Workshop on Deep Hierarchies in Vision (DHV, Vienna), 2012.
[33] S. Liwicki, S. Zafeiriou, M. Pantic, Incremental slow feature analysis with indefinite kernel for online temporal video segmentation, in: K. Lee, Y. Matsushita, J. Rehg, Z. Hu (Eds.), Computer Vision ACCV 2012, Lecture Notes in Computer Science, vol. 7725, Springer, Berlin, Heidelberg, 2013, pp. 162–176.
[34] S. Liwicki, S. Zafeiriou, G. Tzimiropoulos, M. Pantic, Efficient online subspace learning with an indefinite kernel for visual tracking and recognition, IEEE Trans. Neural Netw. Learn. Syst. 23 (10) (2012) 1624–1636.
[35] G.H. Golub, C.F. Van Loan, Matrix Computations, 3rd ed., Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[36] W. Bhmer, S. Grnewlder, H. Nickisch, K. Obermayer, Regularized sparse kernel slow feature analysis, in: D. Gunopulos, T. Hofmann, D. Malerba, M. Vazirgiannis (Eds.), Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Computer Science, vol. 6911, Springer, Berlin, Heidelberg, 2011, pp. 235–248.

[37] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, New York, NY, USA, 2004.

[38] T.-J. Chin, D. Suter, Incremental kernel principal component analysis, IEEE Trans. Image Process. 16 (6) (2007) 1662–1674.

[39] A. Levey, M. Lindenbaum, Sequential Karhunen–Loeve basis extraction and its application to images, IEEE Trans. Image Process. 9 (8) (2000) 1371–1374.

[40] M. Brand, Incremental singular value decomposition of uncertain data with missing values, in: A. Heyden, G. Sparr, M. Nielsen, P. Johansen (Eds.), Lecture Notes in Computer Science, 2350, Springer, Berlin, Heidelberg, 2002, pp. 707–720.

[41] D. Ross, J. Lim, R.-S. Lin, M.-H. Yang, Incremental learning for robust visual tracking, Int. J. Comput. Vis. 77 (13) (2008) 125–141.

[42] T. Martinetz, K. Schulten, Topology representing networks, Neural Netw. 7 (3) (1994) 507–522.

[43] Webots, Commercial mobile robot simulation software, Cyberbotics Ltd. http://www.cyberbotics.com.

[44] K. Gold, B. Scassellati, Using probabilistic reasoning over time to self-recognize, Robotics and Autonomous Systems 57 (4) (2009) 384–392.

Farhan Dawood received the Master's degree in Electronic Engineering from International Islamic University, Islamabad, Pakistan, in 2009. He is currently a Research Assistant at Advanced Robotic Lab, University of Malaya, Kuala Lumpur, Malaysia and also working towards the Ph.D. degree. His research interests include human–robot interaction, computer vision and machine learning.

Chu Kiong Loo obtained his Ph.D. (University Sains Malaysia), B.Eng. (First class Hons in Mechanical Engineering from University Malaya). Formerly he was a Design Engineer in various industrial firms in different capacities as well as he is the founder of Advanced Robotics Lab in University of Malaya. He has been involved in the application research of Perus's Quantum Associative Model and Pribram's Holonomic Brain Model in humanoid vision projects. Currently he is the Professor of Computer Science and Information Technology, University of Malaya, Malaysia. He has completed many funded projects by Ministry of Science in Malaysia and High Impact Research Grant from Ministry of Higher Education, Malaysia. Loo's research experience includes brain inspired quantum neural network, constructivism inspired neural network, synergetic neural networks and humanoid research.