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

Farhan Dawood, Chu Kiong Loo

*Department of Artificial Intelligence, University of Malaya, Lembah Pantai, 50603 Kuala Lumpur, Malaysia

A R T I C L E   I N F O

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

A B S T R A C T

Imitation learning through self-exploration is an essential mechanism in developing sensorimotor skills for human infants as well as 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.

© 2015 Elsevier B.V. All rights reserved.

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...
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 online 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 actions assist 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

![Fig. 1. Architecture for autonomous motion segmentation and incremental imitative learning.](image-url)
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 is created once and cannot grow incrementally. 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). Tscherpanow [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 modeling the observation probabilities, the TGER-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 be invariant to frequent transformations. SFA is originally designed to represent the salient features on an input stream of data in a way slowly over time. In other words, the slowness learning principle, which is governed by the slow information contained in a signal does not change suddenly, but slowly over time. In other words, the slowness learning principle.

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., \( g(x) = Wx \). However, for the real-time applications the non-linear input features are expanded through expansion function, \( h(\cdot) \), such that \( z = h(x) \), 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 \( y_j(t) = g(x(t)) = W_j h(x(t)) = W_j z(t) \). 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 \( z = h(x) - h_o \), where \( h_o = \bar{h}(x) \). Assuming that the signal has unit variance such that \( W_j z_j^T w_j = 1 \), where \( z_j^T = I \), the optimization problem can be written in more convenient way using matrix notations

\[
\min_{W} \text{tr}(W^T Z Z^T W), \quad \text{s.t.} \quad W^T Z Z^T W = I \tag{5}
\]


where \( Z = [z_1, \ldots, z_n] \) contains the input features and \( Z \) represents the temporal derivation matrix, and \( \text{tr}(\cdot) \) computes the trace of a matrix. The matrix \( S \) represents the whitening matrix to fulfil the unit variance constraint such that \( S^T Z Z^T S = I \). Then, the directions of least variance in the derivative signals \( Z \) are found on the derivative covariance matrix \( Z^T Z \) and represented by an orthogonal matrix \( R \) to obtain the projection \( W = SR \) which solves (5). For this we compute the eigenvalue decomposition of \( Z \) as \( Z = Q \Lambda Q^T \). For high dimensional data we find the singular value decomposition (SVD) [35] of the centred data matrix \( Z \) as \( U \Sigma V^T \).

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

\[
\min_{R} \text{tr}(R^T S Z Z^T SR), \quad \text{s.t.} \quad R^T S Z Z^T SR = I \tag{6}
\]

The temporal derivatives are computed from \( Z \) using the forward temporal differences \( \dot{Z} = Z_{k+1} - Z_k \) where \( Z_k \) is the centred data matrix such that \( Z = Z - \frac{1}{n} \mu_Z \) where \( \mu_Z \) is the mean of the data patterns, where \( 1_{n \times 1} \) is an \( n \times 1 \) matrix with all elements set to 1.

Later to find the output of SFA, in the second step, the mean ED of \( Z^T \) is computed such that \( S Z Z^T S = R^T R \). Here \( Z \) represents the centred derivative data matrix computed by subtracting \( z \) from its mean \( \mu_z \) such that \( \dot{Z} = Z - \mu_z \). The output of the SFA is given by

\[
y_j = R^T (S^T \dot{Z}_j - S^T \mu_z) = W^T (z_j - \mu_z) \tag{7}
\]

The ordering, in terms of slowness, of the functions \( y_j \), is provided by the order of the components in \( R \) which is governed by the eigenvalues in \( H \). 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. Incremental Kernel Slow Feature Analysis needs to update the mean of data and the whitening projections.
Suppose we have a data matrix $X_\ell = \{x_i, \ldots, x_n\} \in \mathbb{R}^{m \times n}$. In principle we non-linearly map $X_\ell$ to a higher dimensional space $\mathcal{F}$ using the function $\Phi: \mathbb{R}^m \rightarrow \mathcal{F}$. Using $\Phi$, we transform $X_\ell$ into $\Phi_\ell = \{\phi(x_i), \ldots, \phi(x_n)\}$. The map $\Phi$ is induced by a kernel function $\kappa(\cdot, \cdot)$ that allows us to evaluate inner products in new space $\mathcal{F}$.

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

$$\min_{K} \text{tr}(S^T K K^T S) \text{ s.t. } R^T S K A S^T R = 1$$

where $K$ is the derivative of the centered kernel matrix $X_c$ such that $K = \Phi_\ell \Phi_\ell^T$. Let us assume we are given a new data matrix $X_{\ell^i} = \{x_i, \ldots, x_n\}$ where $\Phi_{\ell^i} = \phi(X_{\ell^i})$. 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 $\Phi_\ell$ and $\Phi_{\ell^i}$ as $X_c = \{\Phi_\ell, \Phi_{\ell^i}\}$. Let $\mu_A$ and $\mu_B$ be the mean of $\Phi_A$ and $\Phi_B$, respectively such that

$$\mu_A = \frac{1}{n_A} \sum_{i=1}^{n_A} x_i$$

and

$$\mu_B = \frac{1}{n_B} \sum_{i=1}^{n_B} x_i$$

where $n_A$ and $n_B$ are the number of data samples contained in $\Phi_A$ and $\Phi_B$, respectively. Trivially, we can update the mean $\mu_c$ of overall data $X_c$ as $\mu_c = \mu_{\ell} - \mu_c$.

Unit variance and incremental whitening: Consider the matrix $K$ such that $K = \Phi_\ell \Phi_\ell^T$. Suppose we have computed the eigenvalue decomposition of $K$ as $K = U D V^T$. Via kernel SVD, we compute the singular value decomposition of $\Phi_\ell$, i.e., $[\Phi_\ell A] = [D V]$.

We want to compute the matrix $S$ which whitens the overall data matrix. For this we need to incrementally compute the SVD of the concatenated matrix such that $X_c = [\Phi_A, \Phi_B] = U D V^T$. Let $\Phi_B$ be the component of $\Phi_B$ orthogonal to $U$ and $U = [U, \Phi_B]$ (computed through QR decomposition [39]). The concatenated matrix can be represented in partitioned form as

$$X_c = [\Phi_A, \Phi_B] = \begin{bmatrix} D & U \Phi_B^T \\ 0 & \Phi_B (\Phi_B^T U U^T \Phi_B)^{-1} \Phi_B^T \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^T$$

Let

$$\Psi = \begin{bmatrix} D & U \Phi_B^T \\ 0 & \Phi_B (\Phi_B^T U U^T \Phi_B)^{-1} \Phi_B^T \end{bmatrix}$$

which is a square matrix of size $k + i$ where $k$ is the number of singular values in $D$. Computing the SVD of $\Psi = U D V^T$ to diagonalize the matrix and substituting into (12) yields the SVD of $X_c = [\Phi_A, \Phi_B]$

$$X_c = \left[\begin{bmatrix} U \Phi_B^T \\ 0 \Phi_B (\Phi_B^T U U^T \Phi_B)^{-1} \Phi_B^T \end{bmatrix} \right] \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^T = U D V^T$$

Since we are only interested in computing $U$ and $D^T$, whose scales with the number of observed data, need not to be computed. Thus, we only need to calculate the SVD of matrix $\Psi$ for the incremental update of the corresponding eigen-spectrum which is defined as $U = \begin{bmatrix} U & \Phi_B \end{bmatrix}$ and $D^T = U D^T$.

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. TGR-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, TGR-HMM are similar to the standard HMMs, besides the fact that the transition structure and the
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.

In addition, the observations are arranged in sequences $O_{1:T} = \{O_1,\ldots,O_T\}$ 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:
The parameters of each node in the topological map are updated adaptively 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 structure grows incrementally by incorporating new knowledge without defiling the previously learned data and adaptively responds to the information acquired from the environment. The structure grows incrementally by incorporating new variables that 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 evidence measure about the top-down weights. This configuration provides a confidence measure about the top-down weights. This confidence measure is defined by the vigilance parameter, \( \rho \). The output layer creates a topological structure of the input data.

Each node weights are defined by a vector \( M_j \) and matrix \( \Gamma_j \) 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 \( \bar{\rho} \) which takes the values within interval (0,1) and the initial covariance matrix, \( \gamma \). 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 \( \omega_j \) 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 \( \Gamma_j \), therefore, the conditional density of \( \Omega_t \) given the winning node \( j \) or the bottom-up input activation value for a node is calculated as

\[
p(O_t | j) = \frac{1}{(2\pi)^{d/2}|\Gamma_j|^{1/2}} \exp \left[-\frac{1}{2}(O_t - M_j)^T \Gamma_j^{-1} (O_t - M_j) \right]
\]

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

\[
J = \arg \max_j p(O_t | j)
\]

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 \( \rho \), that is if

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

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

\[
n_j = n_j + 1
\]

\[
\xi_j = \left(1 - \frac{1}{n_j}\right) \xi_j + \frac{1}{n_j} \Omega_t
\]

\[
\Gamma_j = \left(1 - \frac{1}{n_j}\right) \Gamma_j + \frac{1}{n_j} (\Omega_t - \xi_j)(\Omega_t - \xi_j)^T
\]

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 \( \rho \), and the initial covariance matrix \( \gamma \). 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 state 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** – \( a_{ij} = P(S_t = j | S_{t-1} = i) \): Represents the probability of transition from state \( i \) to state \( j \).
- **Observation probability distribution** – \( B = P(O_t | S_t = i) \): The probability distribution is represented by a Gaussian function denoted by the parameters \( N(O_t | m_i, C_i) \). 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 \( \rho \)

**Ensure:**
- Nodes \( N \)
- Edges \( E \)

**Require:**
- Observation vector \( O_t \)
- If there is no node in the network then
- Add \( O_t \) in the network as new node, \( N \leftarrow N \cup \{O_t\} \)
- Update the weights of the node \( N \) using (22)-(24)
- If \( \text{calcVig} < \rho \) i.e. the observation vector fails the vigilance test then
- Reset the winner node and find a new winner from observation vector.
- Update the weights of previous winner node.
- Obtain the new observation vector \( O_t \)
- If the learning is not completed, go to Step 6 to process the next observation.
- Else
- If \( \text{calcVig} > \rho \) i.e. the node passed the vigilance test then
- Add as a new node \( N \leftarrow N \cup \{O_t\} \)
- Update the weights of the \( O_t \) (winner node) \( N(\eta_f, M_f, \Gamma_f) \) using Eqs. (22)-(24)
- Add edge between the previous winner and current winner nodes \( E \leftarrow E \cup \{\text{prevWinner}, O_t\} \)
- End if
- End if
- 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 \( \pi_l = \pi_0 \) and self-transition probability \( a_{ij} = a_{0l} \), where \( l \) represents the new node. Similarly, for addition of every new node and the new edges \( \{ij\} \) connecting these nodes, the transition probabilities are also initialized with state transition probability value \( a_{ij} = a_{0l} \).

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-estimate the transition probability and state prior probability using

\[
\pi_{lj} = \frac{\sum_{t=1}^{T-1} a_{l}(i)a_{j}(O_{t+1})\beta_{j,l}(i)}{\sum_{t=1}^{T-1} a_{l}(i)\beta_{l,l}(i)} \tag{25}
\]

\[
\pi_{li} = \frac{a_{l}(i)\beta_{l,l}(i)}{P(O_{l} | S_l)} \tag{26}
\]

In Eqs. (25) and (26) the \( a_l \) and \( \beta_l \) represent the forward and
backward variables [13]. Table 2 explains the recursive computation of these variables. $P(O|S_i)$ in Eq. (26) determines the joint observation probability.

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

$$\pi_j = \frac{\pi_j + (N_p - 1)\alpha_{ij}}{N_p}$$

(27)

$$\pi_i = \frac{\pi_i + (N_p - 1)\beta_{ij}}{N_p}$$

(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)
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
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 affected 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

<table>
<thead>
<tr>
<th>Action type</th>
<th>Segmentation accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Optical flow [12]</td>
</tr>
<tr>
<td></td>
<td>Start of action</td>
</tr>
<tr>
<td>LAL180</td>
<td>98.708</td>
</tr>
<tr>
<td>LAR180</td>
<td>69.298</td>
</tr>
<tr>
<td>BAL90</td>
<td>91.250</td>
</tr>
<tr>
<td>BAR90</td>
<td>86.103</td>
</tr>
<tr>
<td>RAL90</td>
<td>95.647</td>
</tr>
<tr>
<td>RAR90</td>
<td>82.558</td>
</tr>
<tr>
<td>BAL180</td>
<td>98.569</td>
</tr>
<tr>
<td>LKE</td>
<td>62.638</td>
</tr>
<tr>
<td>LAL90</td>
<td>99.956</td>
</tr>
<tr>
<td>LAR90</td>
<td>90.089</td>
</tr>
<tr>
<td>BAL90</td>
<td>99.996</td>
</tr>
<tr>
<td>LAR180</td>
<td>70.080</td>
</tr>
<tr>
<td>BOU</td>
<td>100</td>
</tr>
<tr>
<td>BOD</td>
<td>68.266</td>
</tr>
<tr>
<td>BAL180</td>
<td>95.672</td>
</tr>
<tr>
<td>RAR180</td>
<td>80.442</td>
</tr>
</tbody>
</table>

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 to 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

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 $\gamma$, the learning will be slow with fewer nodes, while for smaller values of $\gamma$, the training will be faster with more number of nodes. The initial covariance matrix is selected in an ad-hoc fashion through trial and error.

\begin{equation}
\text{CompressionRatio} = \frac{\text{No. of Samples}}{\text{No. of Nodes}}
\end{equation}

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 affects 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 UML/C/625/1/HIR/ MOHE/FCGSIT/10 (Mirror Neuron and Quantum Mechanics Based Neural Architecture for Humanoid) from the Ministry of Education Malaysia.

References


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.