ijrr

The International Journal of Robotics Research
0(0) 1–16

DOI: 10.1177/0278364911426178
ijr.sagepub.com
SAGE

Incremental learning of full body motion primitives and their sequencing through human motion observation

Dana Kulić[1], Christian Ott[2], Dongheui Lee[3], Junichi Ishikawa[4] and Yoshihiko Nakamura[4]

Abstract

In this paper we describe an approach for on-line, incremental learning of full body motion primitives from observation of human motion. The continuous observation sequence is first partitioned into motion segments, using stochastic segmentation. Next, motion segments are incrementally clustered and organized into a hierarchical tree structure representing the known motion primitives. Motion primitives are encoded using hidden Markov models, so that the same model can be used for both motion recognition and motion generation. At the same time, the temporal relationship between motion primitives is learned via the construction of a motion primitive graph. The motion primitive graph can then be used to construct motions, consisting of sequences of motion primitives. The approach is implemented and tested during on-line observation and on the IRT humanoid robot.

Keywords

humanoid robots, learning by demonstration, motion primitive learning, stochastic models

1. Introduction

Learning through observation and imitation is an attractive paradigm for humanoid robots, as it enables the robot to take advantage of the similarity in body structure between humanoids and humans, and avoids the need for explicit programming of complex robot motions. Much of human movement is thought to be composed of motion primitives, which are combined and sequenced to generate more complex behavior (Schaal 2006). Many algorithms have been proposed in the literature for representing and learning motion primitives from demonstration (Breazeal and Scassellati 2002; Schaal et al. 2003; Krueger et al. 2007). However, most of the approaches thus far consider off-line learning, where the data is first collected, segmented and sorted into the motion groups to be learned, and then a one-time, off-line learning process is used. However, a robot operating in the human environment should be capable of continuous learning over its' entire lifespan. The robot should be able to segment and classify demonstrated actions, and learn how those actions may be combined to perform tasks on-line during co-location and possible interaction with the (human) teacher.

In order to extract motion primitives and behaviors during on-line observation, several key issues must be addressed by the learning system: automated motion segmentation, recognition of previously learned motion primitives, automatic clustering and learning of new motion primitives and, finally, learning how motion primitives can be combined into sequences to form behaviors. Since data is being processed autonomously, in an on-line system, later stages of the process must be robust to errors in the preceding steps. In previous work (Kulić et al. 2008b, 2009), we have been developing an approach for on-line segmentation and clustering of whole-body human motion primitives. In this paper, based on the previously proposed framework for learning motion primitives, we propose an approach for also incrementally learning the relationship between the motion primitives for the formation of longer behaviors. A graph model is built representing the sequential relationship between the motion primitives. The graph can then be used to generate new coherent motion sequences for the robot, based on the learned motion primitives and the learned relationship between them. Since the motion primitive graph represents an abstraction of the observed patterns of behavior of the human

[1]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada
[2]Institute of Robotics and Mechatronics, DLR – German Aerospace Center, Wessling, Germany
[3]Department of Electrical Engineering and Information Technology, Technical University of Munich, Munich, Germany
[4]Department of Mechano-Informatics, University of Tokyo, Bunkyo-ku, Tokyo, Japan

Corresponding author:
Dana Kulić, Department of Electrical and Computer Engineering, University of Waterloo, 200 University Avenue West, Waterloo, ON, Canada N2L 3G1.
Email: dkulic@ece.uwaterloo.ca

demonstrator, the motion primitive graph can also further our understanding of human motion. The motion primitive graph can be used to detect normal and abnormal human activity, and to predict future human movements based on the recent history of observations. The complete system is capable of online learning from continuous demonstration.

1.1. Related work

Robot skill learning through observation of human motion is an attractive paradigm for humanoid robots, and has received significant attention in the literature (Kuniyoshi et al. 1989; Kuniyoshi and Inoue 1994; Liu and Asada 1992; Dillmann et al. 1999). Breazeal and Scassellati (2002), Schaal et al. (2003), and Krueger et al. (2007) provide reviews on motion learning by imitation. Many different approaches for representing motion primitives have been proposed, including neural networks (Ogata et al. 2005; Erlhagen et al. 2006; Ito and Tani 2004; Andry et al. 2004), stochastic models such as hidden Markov models (HMMs) (Inamura et al. 2004; Takano et al. 2006, 2005; Billard et al. 2006, 2004; Asfour et al. 2006), phase oscillators for rhythmic motions (Nakanishi et al. 2004), dynamical models (Ijspeert et al. 2002), polynomial functions (Okada et al. 2002) or weighted graph structures (Breazeal et al. 2005). In addition, some researchers focusing on learning the relationships between primitives utilize pre-programmed, static primitive behaviors, for example Nicolescu and Matarić (2001, 2003, 2005), Ilg et al. (2004), and Dominey et al. (2008). Where motion primitives are being learned, as noted by Breazeal and Scassellati (2002), the majority of algorithms proposed to date assume that the motions to be learned are segmented and clustered *a priori*, and that the model training takes place off-line.

A stochastic approach for learning both the motion primitives and the transitions between them is proposed by Taylor et al. (2006), based on the conditional restricted Boltzmann machine (CRBM). Once trained, the system can generate continuous motion sequences, as well as represent and generate the transitions between motions. Once the low-level network consisting of individual motion patterns has been trained, additional higher-order layers can be added to model the higher-order structure of motion patterns. However, training for all primitive level motions must be completed before higher-level training can begin, so that the algorithm is not able to build the model incrementally, during on-line observation.

Jenkins and Matarić (2004a) describe a system for extracting behaviors from motion capture data. In their algorithm, continuous time series data is first segmented using the *kinematic centroid segmentation* algorithm, which is a heuristic algorithm modeling arms and legs as pendulums, and placing segment boundaries at the beginning and end of pendulum swings. The segmented data is then embedded in a lower dimensional space using the spatiotemporal Isomap (ST-Isomap) algorithm (Jenkins and Matarić 2004b). Once the data has been reduced, it is clustered into groupings using the 'sweep-and-prune' technique. Once a model of the primitive behaviors is formed, a higher level re-processing of the data is performed to discover meta-behaviors, i.e. probabilistic transition probabilities between the behaviors. Similarly to primitive behaviors, meta-level behaviors are derived by extracting meta-level feature groups using ST-Isomap. While this system autonomously segments and clusters data, the algorithm cannot operate incrementally, as the entire range of motions is required to form the lower-dimensional space embedding.

Another popular stochastic modeling technique for human motion modeling are HMMs. HMMs have been applied to many different domains requiring human motion modeling, including skill transfer (Dillmann et al. 1999), sign language and gesture modeling (Startner and Pentland 1995) and motion representation (Dillmann et al. 1999; Calinon et al. 2007). In many of the existing approaches using HMMs to implement human motion learning (Billard et al. 2006; Ezaki 2000; Inamura et al. 2004; Lee and Nakamura 2006), humanoid motion primitives have been encoded using HMMs, and subsequently used for motion generation. However, the initial training of the models was carried out off-line, where all of the training examples for each model were grouped manually.

Calinon et al. (2007) proposed a system for programming by demonstration, based on Gaussian mixture models (GMMs). In their approach, the data (including both joint angle data and absolute and relative object position and hand position data) is first analyzed via principal components analysis (PCA) to determine the relevant subspace for the task. The reduced dimensionality data is then temporally aligned and abstracted into a set of Gaussian Mixtures. This system is also extended to an on-line, interactive approach by developing a method for incremental training of the GMM structure (Calinon and Billard 2007b) and developing an interactive training approach combining demonstration and kinesthetic training (Calinon and Billard 2007a). However, it appears that the motion segmentation is performed manually by the trainer.

Kadone and Nakamura (2005, 2006) describe a system for automated segmentation, memorization, recognition, and abstraction of human motions based on associative neural networks with non-monotonic sigmoid functions. Their approach achieves on-line, incremental learning of human motion primitives, and self-organization of the acquired knowledge into a hierarchical tree structure. However, the abstracted motion representation can only be used for subsequent motion recognition, and cannot be used for motion generation.

Kulić et al. (2008b) have been developing algorithms for incremental learning of motion pattern primitives through long-term observation of human motion. Human motion patterns are abstracted into a stochastic model representation, which can be used for both subsequent motion recognition and generation. As new motion patterns are

observed, they are incrementally grouped together based on their relative distance in the model space. The resulting representation of the knowledge domain is a tree structure, with specialized motions at the tree leaves, and generalized motions closer to the root. In this paper, we build on this previously developed approach to incorporate a hierarchical learning structure, where both the motion primitives and their higher-order sequencing are learned at the same time.

Outside the robotics community, in the computer graphics domain, there has also been a long-standing research effort to develop algorithms for realistic human-like motion generation for animated characters. Kovar et al. (2002) first proposed the *motion graph* technique. In this approach, a directed graph is constructed encapsulating the relationships between postures extracted from a motion capture data set. The graph can then be used to generate extended sequences of realistic looking motions. Yamaguchi et al. (2008) and Yamane et al. (2009) developed an algorithm for building a motion graph via a binary tree clustering technique. Sidenbladh et al. (2002) also propose an approach for probabilistically modeling 3D human motion for character synthesis and tracking. In their approach, a low-dimensional linear model of the human motion is learned, and then used to structure the database into a binary tree. Each leaf node in the tree represents a fixed length (1/3 of a second) of time series data of the human movement. The binary tree structure of the motion database allows for fast search enabling on-line motion tracking and synthesis. However, similar to the work of Jenkins and Matarić (2004a) both the motion graph and the binary tree approaches proposed in the graphics literature (Kovar et al. 2002) require that the entire database of motions is available *a priori*, it is not possible to incrementally add new motions to the database during on-line observation.

1.2. Proposed approach

The aim of our research is to develop robots which can learn motion primitives and higher-level behaviors on-line while observing and interacting with a human partner over extended periods of time. In our previous work (Kulić et al. 2008b, 2009), we have been developing algorithms for automatic extraction of motion primitives from continuous motion data. In this paper, we extend the previously developed framework to autonomously extract the temporal and sequencing relationships between the motion primitives concurrently with learning the motion primitives themselves. We also integrate the learning of the motion primitives and the learning of the primitive sequencing into a single framework and demonstrate the complete system during on-line observation of human motion, as well as the application of the learned motion primitives and their sequencing to humanoid robot motion generation. We model motion primitives as stochastic models, either HMMs or factorial HMMs. Motion primitives can be represented using either Cartesian data of human motion (for example, the locations of key body parts), or using joint angle data of a kinematic model of the human. In this paper, we focus on full body motions, however, the framework proposed herein can be utilized for any type of motion primitive, such as motion primitives involving arms or legs only. For tasks involving interaction with the environment, the observation vector could also consist of relative position data, such as the distance between the hand and the object to be grasped, and other interaction data, such as contact or grasping forces. Once the appropriate task representation is selected, using continuous time-series data as the input, we first segment the data into potential motion primitives, using a modified version of the Kohlmorgen and Lemm algorithm (Kohlmorgen and Lemm 2001) for unsupervised segmentation (Kulić et al. 2009). Next, the extracted segments are input into an automated clustering and hierarchical organization algorithm (Kulić et al. 2008b). The segmentation algorithm uses a HMM to represent the incoming data sequence, where each model state represents the probability density estimate over a window of the data. The segmentation is implemented by finding the optimum state sequence over the developed model. Individual motion patterns are then clustered in an incremental fashion, based on intra-model distances. The resulting clusters are then used to form a model of each abstracted motion primitive, which can be used for subsequent motion generation. Concurrently with the learning of the motion primitives, the relationship between primitives is learned by forming a directed graph of the motion primitives. Each time a new motion primitive is added, a new node is added to the graph. Each time a consecutive pair of known motion primitives is recognized, a transition is added to the graph, incrementally forming a transition matrix among the motion primitives. The motion primitive graph can then be used for subsequent motion sequence generation on a humanoid robot. An overview of the combined approach is shown in Figure 1.

This paper is organized as follows: In Section 2, we summarize the previously proposed segmentation and clustering framework (Kulić et al. 2008b, 2009), which is used to extract motion primitives used in subsequent sections. Section 3 describes the formation of the motion primitive graph, which forms the model of the sequential relationships of the motion primitives. In Section 4, the results of experiments verifying the algorithm during on-line operation, and on a humanoid robot trained with a continuous stream of human motion capture data are reported. Section 5 concludes the paper.

2. Automated segmentation and clustering

In this section we review our framework for autonomously extracting motion primitives from continuous online demonstration data, initially reported in Kulić et al. (2008b, 2009). In the proposed approach, the continuous data stream is first segmented into motion primitive segments (Kulić and Nakamura 2008), and then these segments are

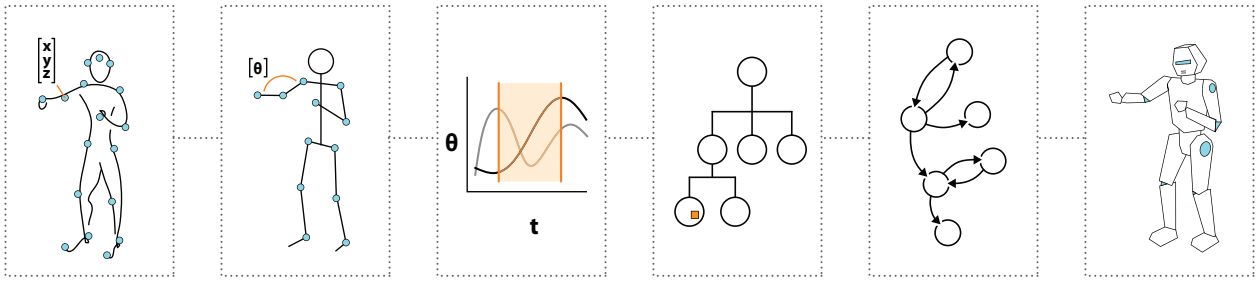

Fig. 1. Overview of the integrated system. $[x, y, z]$ marker position data is first converted into joint angle data for the humanoid kinematic model using on-line inverse kinematics. The joint angle data is next segmented using on-line stochastic segmentation (Section 2.1, initially reported in Kulić et al. (2009)). Segments are incrementally clustered and organized in a tree structure (Section 2.2, initially reported by Kulić et al. (2008b)). At the same time, the temporal relationships between segments are learned via the motion primitive graph (Section 3). Walks on the motion primitive graph are used to generate command trajectories for the humanoid robot. The last two modules are the focus of this paper.

incrementally clustered to extract models of motion primitives (Kulić et al. 2008b, 2009). We provide a basic overview of the algorithms required for understanding how these motion primitives are subsequently used for identifying and generating sequences of behaviors; for a detailed description and analysis of the segmentation and clustering algorithms, the reader is referred to the original publications (Kulić and Nakamura 2008; Kulić et al. 2008b, 2009).

2.1. Probabilistic segmentation

A modified version of the Kohlmorgen and Lemm segmentation algorithm (Kohlmorgen and Lemm 2001; Janus and Nakamura 2005) is used to automatically segment the continuous time series data into motion primitive candidates (Kulić and Nakamura 2008). We provide a brief overview of the segmentation approach here, detailed analysis of the performance and parameter sensitivity of the stand-alone segmentation algorithm is provided in Kulić and Nakamura (2008). The same parameter settings as specified in Kulić and Nakamura (2008) are used herein.

The segmentation approach was selected due to its ability to handle observation data with a large number of degrees of freedom in the observation vector, and the ability to begin segmenting without any *a priori* knowledge about the type of motion to be observed. The Kohlmorgen and Lemm segmentation algorithm (Kohlmorgen and Lemm 2001) is based on the assumption that data belonging to the same motion primitive will have the same underlying probability distribution. The incoming data stream is first embedded into a higher-dimensional space, as suggested by Kohlmorgen and Lemm (2001). This embedding is achieved by concatenating temporally adjacent incoming data vectors. The resulting data vector $\mathbf{x}$ has dimensionality $N_d * T$ where N_d is the dimensionality of the incoming data (typically the number of degrees of freedom (DoFs) of the demonstrator), and T is the embedding duration. The density distribution of the embedded data is next estimated over a sliding window of length W, via a standard (Parzen) density estimator with multivariate Gaussian kernels, centered on the data points in the window $\{\mathbf{x}_{t-w}\}_{w=0}^{W-1}$:

$$p_t(\mathbf{x}) = \frac{1}{W} \sum_{w=0}^{W-1} \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{(\mathbf{x} - \bar{\mathbf{x}}_{t-w})^2}{2\sigma^2}\right), \quad (1)$$

where $p_t(\mathbf{x})$ is the density function at time t, $\mathbf{x}$ is the embedded data at time t, $\bar{\mathbf{x}}_{t-w}$ is the average of the data points in the sliding window, and σ is a smoothing parameter calculated proportional to the mean distance between each $\mathbf{x}$ and its n nearest neighbors.

As more data are observed, the distance between successive data windows can be calculated based on the integrated square error between two probability density functions (the L_2-norm):

$$d(p_{t1}, p_{t2}) = \int (p_{t1}(\mathbf{x}) - p_{t2}(\mathbf{x}))^2 \, d\mathbf{x}. \quad (2)$$

The segmentation analysis is carried out by defining a HMM over a set S of sliding windows. Each window corresponds to a state of the HMM. For each state, the observation probability distribution is defined as

$$p(p_t(\mathbf{x})|s) = \frac{1}{\sqrt{2\pi}\varsigma} \exp\left(-\frac{d(p_s(\mathbf{x}), p_t(\mathbf{x}))}{2\varsigma^2}\right), \quad (3)$$

where $p(p_t(\mathbf{x})|s)$ is the observation probability distribution, i.e. the probability that the data represented by $p_t(\mathbf{x})$ is observed while in state s, $p_s(\mathbf{x})$ is the probability distribution associated with the time window corresponding to state s, d is the distance function based on the integrated square error (Equation (2)), and ς is a free parameter in the algorithm, which determines the variance of the observation probability distribution.

The initial state distribution of the HMM is given by the uniform distribution, and the state transition matrix is designed such that transitions to the same state are k times more likely than transitions to any of the other states:

$$a_{ij} = \begin{cases} \frac{k}{k+N-1} & \text{if } i = j, \\ \frac{1}{k+N-1} & \text{if } i \neq j, \end{cases} \quad (4)$$

where a_{ij} represents the transition probability that the HMM will transition from state i to state j, with both i and j ranging from $1 : N$, where N is the number of states of the HMM. The Viterbi algorithm (Rabiner 1989) can then be used to find the optimum state sequence given the current set of observations. This state sequence represents the segmentation result, with each change of state corresponding to a segment point. We use the on-line variant of the Viterbi algorithm (Kohlmorgen and Lemm 2001), which incrementally builds the state path likelihoods as each new state is observed, by re-using the estimate of the likelihood and optimal state sequence from the previous time step. The on-line formulation of the Viterbi algorithm proposed by Kohlmorgen and Lemm subsumes the two free parameters of the algorithm, ς and k, into a single parameter which can be interpreted as the cost of state switching. This parameter can be determined based on the expected magnitude of change between different motions relative to noise (Kohlmorgen and Lemm 2001). To prevent the state list from growing to infinity as the number of observed data points increases, Kohlmorgen and Lemm (2001) propose removing states following a segment away from that state. However, Janus (2006) has found that this approach leads to over segmenting, as the considered data range becomes too small (of the order of $5W$) and therefore the algorithm becomes more prone to local minima. Instead, Janus (2006) proposes that the algorithm runs in batch mode over a larger, fixed number of windows, and that windows should be discarded in a first in first out (FIFO) manner. The Janus approach is adopted herein.

This approach is capable of segmenting motion data with no *a priori* information about the type of motion to be observed, and is used at the start of the learning phase. Using a HMM to model the time series data, enables the system to model the temporal and spatial variability in human motion data, allowing the segmentation algorithm to segment human movement in the face of variability and noise. The basic segmentation algorithm works well for finding boundaries of two different motions, but is not as effective at correctly identifying the exact time of the switching point for each action, for example, the transition from arm raise to arm lower, especially for those actions where few joints are moving. By adding information about which joints are active, the performance of the basic algorithm can be improved. As motion primitive models are extracted from the demonstration (described in the following section), knowledge about previously observed motion primitives can be used to further improve the segmentation, by modifying the state transition model, and by modifying the distance function based on the knowledge of which joints are active during the known motion primitive (Kulić et al. 2009). This modification to the basic Kohlmorgen and Lemm algorithm improves the segmentation performance, resulting in more accurate segmentation as more motion primitives become known. The algorithm has been validated and found to perform reliably with a variety of motion sequences and human kinematic models (Kulić et al. 2009; Kulić and Nakamura 2009). A detailed analysis and results of the segmentation algorithm performance can be found in Kulić et al. (2009).

2.2. Incremental motion primitive learning

The segmentation algorithm described in Section 2.1 decomposes the incoming continuous data stream into short segments of time series data, each corresponding to a potential motion primitive. The task of the motion primitive learning component is to group these segments together based on their similarity, and extract a model for each group of similar segments. We consider this group model the *motion primitive*. Once the incoming time series data has been segmented, each segment is sequentially passed to the clustering module. In the proposed clustering approach (Kulić et al. 2007c), a hierarchical tree structure is incrementally formed representing the motions learned by the robot. Each node in the tree represents a motion primitive, which can be used to recognize a similar motion, and also to generate the corresponding motion for the robot. Within each local area of the motion space, a standard clustering technique (Jain et al. 1999) is used to subdivide motion primitives. A HMM is used to abstract the observation sequences. HMMs were selected to model motion primitives because this type of model can encapsulate both spatial and temporal variability, both of which are a feature of human motion obtained during demonstration. In addition, as HMMs are a generative model, the same model can be used for both motion recognition and generation, allowing the model to be used both for discriminating between new and known motions, and for generating the motion to be executed by the humanoid robot.

In the tree structure to be constructed, each node represents a group of similar motion primitives, which is represented by a group HMM. Note that the HMM model used to model an individual motion is different from the HMM used for segmentation. In the segmentation HMM, described in Section 2.1, each state describes the data distribution over a window of motion data. The HMMs used for clustering describe the motion at a finer grained level, where each state can be considered to describe a key posture in a motion primitive. The algorithm initially begins with one group (the root node). Each time a motion is observed, it is encoded into a HMM and compared with existing group models via a tree search algorithm, and placed into the closest group. The size of the HMM model can be selected either by use of the Akaike information criterion (Kulić et al. 2007a), or by adaptively adding chains to the HMM based on the representation requirements (Kulić et al. 2008b). The distance between two models is computed using the Kullback–Leibler distance:

$$D(\lambda_1, \lambda_2) = \frac{1}{T}[\log P(O^{(2)}|\lambda_1) - \log P(O^{(2)}|\lambda_2)], \quad (5)$$

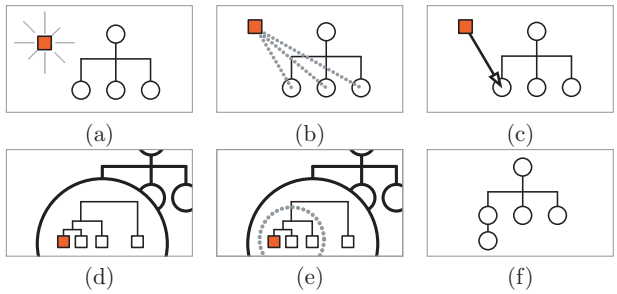

Fig. 2. Overview of the incremental clustering algorithm (Kulić et al. 2008b) (a square represents a data sequence and a circle represents a group): (a) A new observation sequence is observed and encoded as a HMM; (b) the observation sequence is compared with existing groups via tree search; (c) the new sequence is placed in the closest existing group; (d) local clustering is performed on the modified group (zoomed in view of modified group); (e) a new subgroup is formed from similar motions in the modified group; (f) the subgroup is added to the tree as a child of the modified group.

where λ_1, λ_2 are two models, $O^{(2)}$ is an observation sequence *generated* by λ_2 and T is the length of the observation sequence. Since this measure is not symmetric, the average of the two intra distances is used to form a symmetric measure.

The new motion is placed into the nearest node. The repository of known groups is organized in a tree structure, so the new observation sequence does not need to be compared with all known behaviors, instead the comparison procedure is implemented as a tree search. The new observation sequence is placed in the closest node, based on the maximum node distance, otherwise it is placed in the parent node of the closest node:

$$D_{\text{thresh}} = K_{\text{maxGD}} D^G_{\text{max}}. \tag{6}$$

Here D_{thresh} is the distance threshold at which a new observation sequence is considered for inclusion to a node, K_{maxGD} is the multiplication factor applied and D^G_{max} is the maximum intra-observation distance for the given node. If no similar motions are found, the new motion is placed in the root node. Once a new observation sequence is added to a group, local clustering is performed within the exemplars of the group, using the complete link hierarchical clustering algorithm (Jain et al. 1999). If a cluster with sufficiently similar data is found, a child group is formed with this data subset. Two criteria are used for forming clusters: minimum number of elements in the subgroup, and the maximum proximity measure of the potential subgroup. The maximum proximity measure is calculated as a function of the distribution function:

$$D_{\text{cutoff}} = \mu_g - K_{\text{cutoff}} \sigma_g, \tag{7}$$

where D_{cutoff} is the distance (proximity) cutoff value (i.e. only clusters where the maximum distance is less than this value will be formed), K_{cutoff} is a user-defined parameter of the algorithm, μ_g is the average distance between observed models in the group, and σ_g is the standard deviation among all of the distances. If a new subgroup is formed based on these criteria, a new group model is trained using the observation sequences from all of the identified subgroup elements. The generated model is subsequently used by the robot both to generate the movement of this motion primitive, and to subsequently recognize when this motion is performed again. Through continuous observation of multiple demonstrations, the algorithm incrementally learns and organizes the motion primitive space, based on the robot's lifetime of observations. The algorithm is illustrated in Figure 2.

The segmentation and clustering approaches can be combined to allow for fully automated motion primitive extraction (Kulić et al. 2009). The primitives extracted through this approach generally correspond well to motion primitives as they would be extracted by a human observer, such as arm raise, arm lower, etc. The algorithm allows the robot to incrementally learn and classify motion primitives observed during continuous observation of a human demonstrator. The generation of a hierarchical structure of the learned behaviors allows for easier retrieval, and the automatic generation of the relationships between behaviors based on their similarity and inheritance. In addition, the robot's knowledge is organized based on the type of training received, so that the robot's knowledge will be most specialized in those areas of the motion primitive space where the most data has been observed.

The constructed tree structure bears some similarity to the binary tree proposed by Sidenbladh et al. (2002). However, in the approach developed by Sidenbladh et al., a fixed time sequence is always used for the motion primitives. Here, we temporally segment the data into variable length motion primitives, based on the data properties of the motion, so that the leaf nodes correspond to recognizable motion primitives. In addition, since we only consider the distance between any two models, our approach allows for the incremental building of the motion database. On the other hand, Sidenbladh et al. use the variance over the entire data set to construct the tree structure, implying that the entire training data must be available prior to the start of the tree construction.

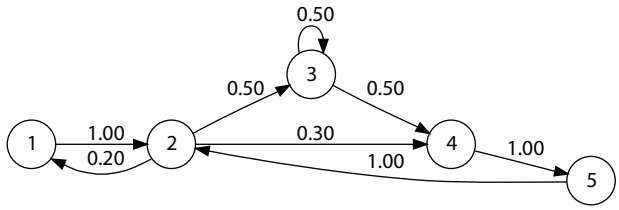

Fig. 3. Example motion primitive graph. Each node represents a motion primitive, while each edge represents an observed sequential ordering between primitives. In this example graph, motion primitive 1 is always followed by motion primitive 2, which can be followed by primitives 1, 3, or 4.

3. Motion primitive graph

Concurrently with the construction of the hierarchical tree structure representing the motion primitives, we learn the relationship between the primitives by constructing a directed graph representing the observed transitions between the primitives. Each node in the motion primitive graph represents a motion primitive, while each edge represents an observed transition between two motion primitives. An example motion primitive graph is illustrated in Figure 3.

Initially, the graph is empty, as no motion primitives are known at initialization. Each time a new motion primitive is abstracted by the clustering algorithm as a leaf node (described in Section 2.2), a corresponding node is added to the motion primitive graph. The incremental clustering algorithm also performs motion recognition. When a newly observed motion segment is placed in an existing (non-root) node of the tree, this indicates that the motion segment has been recognized as the motion primitive corresponding to the selected node. A motion primitive transition model is built incrementally by monitoring for instances when a sequence of two motion primitives are recognized by the incremental clustering algorithm. Each time a recognized motion primitive transition is detected, the corresponding edge is incremented. In this way, the robot incrementally learns how motion primitives may be combined during behavior execution.

The constructed graph can then be used to generate sequences of primitives by concatenating a set of nodes connected in the graph, for example, by searching the graph for a valid path given a starting and target position. The graph can also be used to generate novel sequences of primitives, not observed from the demonstrator. In this way, the robot can generate novel behaviors based on its known motion primitives and their relationships. In addition, the motion graph represents an abstracted model of the observed human behavior. It can also be used to detect and monitor human activity, and to predict future movement of the observed human, based on the sequence of primitives executed thus far.

The general problem of learning transition matrices is a well-studied topic in machine learning, and particularly reinforcement learning (Sutton and Barto 1998). The proposed approach is similar to the motion graph approach employed for graphics character animation (Kovar et al. 2002; Yamaguchi et al. 2008; Yamane et al. 2009). However, a key difference from the motion graph approach is that, in the approach proposed herein, the motion data is encapsulated in two hierarchical levels: the level of the motion primitive and the level of the motion primitive graph. At the level of the motion primitive, each node (each state of the HMM) can be thought to correspond to a posture. At the level of the motion primitive graph, each node corresponds to a motion primitive, rather than an individual posture. Conversely, with the motion graph approach, only a single layer of hierarchy is used, such that each node corresponds to a posture only. An alternative way of conceptualizing the motion primitive approach is that the motion primitives discover the postures which only transition to each other, and these are grouped into nodes at the motion primitive graph level. This means that the resulting graph is much smaller and therefore more easily searchable, and requires less computational effort for generating appropriate paths. The motion primitive graph is at a higher abstraction level, since motion primitives serve to abstract continuous motions into a single logical concept.

Unlike the motion graph approaches, the algorithm proposed herein also allows the graph structure to be built incrementally rather than requiring the data to be available prior to the start of training.

In the current implementation, only the leaf nodes generated by the clustering algorithm are used to form the motion primitive graph. Higher abstractions could also be achieved by constructing the primitive motion graph at higher levels of the motion primitive hierarchy, by making use of the hierarchy of motion primitives identified by the tree structure of the motion primitives generated by the clustering algorithm. As the number of motion primitives becomes large, higher-level motion primitive graphs could be used to further reduce search time by searching first at the higher level, where the number of nodes is fewer, and searching at the lower level within the subspace identified by the initial higher-level search. For example, a large tree of motion primitives might include a higher-level node which encapsulates reaching motions, and child nodes which model different types of reaching motion. A motion primitive graph at the higher level could first be used to identify that a reaching primitive is required, and then a local search within the lower-level graph could be employed to determine the most appropriate type of reaching motion.

4. Realtime implementation and experimental results

The proposed approach was tested in two experimental scenarios. In the first scenario, described in Section 4.1, the system was tested using a 40-DoF human kinematic model, during on-line, interactive demonstration. In the second

scenario, described in Section 4.2, a humanoid kinematic model corresponding to a 30-DoF humanoid robot was used, to allow the motion to be directly generated on the humanoid robot.

4.1. On-line learning during interactive motion capture

In the first set of experiments, the system is tested during on-line, interactive demonstration. For this purpose, a visualization system has also been developed (Kulić et al. 2009), which allows the user to visualize the state of the system's knowledge as it is being acquired. This system provides a visual overview of the motion database, the motion primitive graph, and individual motion primitives. The human demonstrator is outfitted with 34 reflective markers located on various parts of the body, and the marker $[x, y, z]$ position is captured and computed by the motion capture system online. The marker data is then passed to the combined motion extraction and visualization system, which performs the on-line inverse kinematics (Yamane and Nakamura 2003) to generate joint angle data, displays an animation character performing the demonstrator motions, and simultaneously passes the data to the segmentation, clustering and motion primitive extraction module for processing. In previous work, a simplified inverse kinematics model was used (20 DoFs) (Kulić et al. 2009), however, here we use a more realistic 40-DoF model, which is better able to capture the full range of human motion. In addition, for this demonstration, Cartesian data is used, as this data is not being re-targeted for humanoid motion generation, and we have found that Cartesian data produces better segmentation results when large DoF models are used (Kulić and Nakamura 2009). During the demonstration, the demonstrator can view the current status of the learning system on the large video screen located in front of the demonstrator, to determine when a motion primitive has been learned and the demonstration can move on to the next set of primitives.

In the experiment shown here, the demonstrator first teaches the system a set of four motion primitives (both arms raise motion, both arms lower motion, bend down motion, bend return motion), and then teaches the system the correct sequencing of the primitives. During the first part of the demonstration sequence, the demonstrator executes multiple examples of the motion primitives. These are consistently segmented so that a group model for each primitive can be formed. During the second part of the sequence, the demonstrator performs a longer sequence of combinations of the motion primitives, which are not necessarily in the same order as the order observed in the first part. These are again correctly segmented and classified to enable the motion primitive graph to be learned.

The experimental setup and frames extracted from the video of the user performing the motion primitives are shown in Figure 4. As can be seen from the figure, the user can observe the motions being abstracted and the current state of the knowledge base, so that the demonstrator is able to easily determine when the demonstrated action or sequence of actions has been acquired. Figure 5 shows the system output as seen by the demonstrator during the sequence, showing the evolution of the motion primitive graph and the tree structure during the demonstration. Figure 6 shows the final tree structure at the end of the demonstration, and Figure 7 shows the final motion primitive graph.

4.2. Motion generation for a humanoid robot

4.2.1. Humanoid robot platform and low-level control. The proposed approach was tested on the IRT platform humanoid robot (shown in Figure 14). The IRT robot is a human-size humanoid robot, with 38 DoFs. The robot has three joints actuating the head, seven joints in each of the arms, six joints in each of the legs, and the one joint at the hip, with an additional eight DoFs in the fingers and toes which were not used during these experiments.

A low-level controller takes the generated trajectories as input and generates motor commands for the robot. Our control approach currently aims at the on-line imitation of the human's upper body motion, while both feet remain in contact with the ground. The joints of the lower body are used for balancing the robot and implementing the hip height and orientation as described in Ott et al. (2008).

The controller design is divided into the control of the upper and lower body (Figure 8) according to the realtime controller hardware of our humanoid robot, in which one realtime computer is used for controlling the upper body joints, and a second realtime computer is used for controlling the lower body joints. The joint angles q and the desired frame for the robot's base link (hip) h_b are transmitted from the upper body computer to the lower body computer via a UDP socket interface. Following simple filtering and the utilization of a linear joint interpolator for limiting the joint velocity to the robot's velocity constraints, we command the joint angles of the upper body directly to the joint position controllers. The commanded trajectory is also transferred to the controller of the lower body where it influences the balancing task.

For the balancing, we aim at keeping the center of gravity (COG) at a constant position $\boldsymbol{c}_d$ inside the support polygon of the feet. From the viewpoint of the popular mass-concentrated 'inverted pendulum model', this has the consequence, that the zero moment point also stays at the same point such that the constraints on the ground reaction force can be satisfied (Kajita 2005).

In addition to the balancing, the joints of the legs should also implement the orientation of the base link as a second objective. The optimization of the two criteria can be performed separately by gradient descent. Thereby, we obtain a desired COG velocity v_d and a desired angular velocity ω for the body frame, which are executed by means of a velocity based inverse kinematics (see Figure 8).

Fig. 4. Frames from the video sequence of the interactive demonstration. The top row shows the both arms raise and lower motions, the bottom row shows the bend down and bend retract motions.

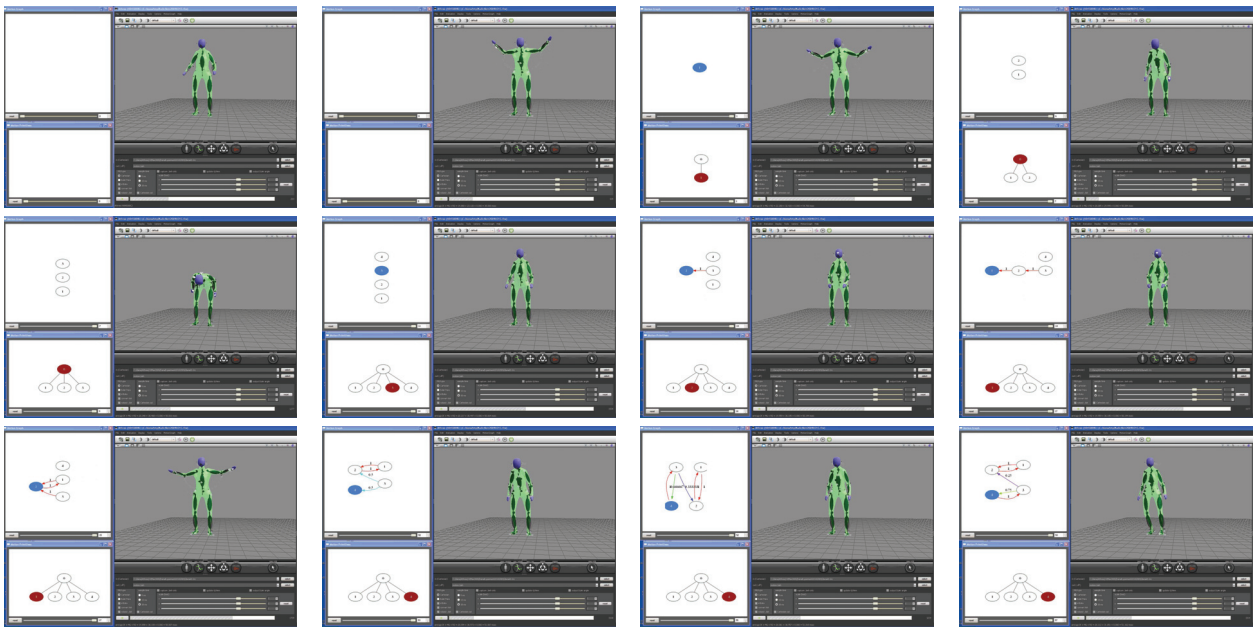

Fig. 5. Frames from the visualization system output as the motion primitives and the motion primitive graph are acquired. The demonstrator first demonstrates the four motion primitives, and then teaches the motion primitive sequencing. Here, motion primitive 1 = both arms lower, motion primitive 2 = both arms raise, motion primitive 3 = bend retract, and motion primitive 4 = bend down.

The current controller assumes that both feet always remain on the ground. In order to extend the system to stepping motions, one possible way would be to identify the occurrence of these motions in the commanded motion primitive and perform an online step planning for generating a trajectory which can be executed with the humanoid.

4.2.2. Data collection and analysis. A large data set was collected in a motion capture studio to test the proposed algorithms on a lengthy continuous sequence of a variety of whole-body motions. Figure 9 shows the motion capture setup and marker locations. A total of 34 markers was used. The data set consists of 16 minutes of continuous whole-body motion data of a single human subject. During the data sequence, the subject performs a variety of full body motions, including a walk in place motion, a squat motion, kicking and arm raising. There are approximately the same number of examples of each motion type in the dataset. In some cases, there is a pause between motions, while other motions are fluidly connected. Therefore, the learning system is exposed to both motions executed in isolation, as well as motions which are sequenced together, and may exhibit co-articulation, i.e. motions which may partially overlap when executed sequentially.

The video of the motion sequence is also observed by a human observer and manually segmented and labeled. The human observer was guided to segment the data into segments which would be visually recognizable as a discrete movement, no further criteria was used. Qualitatively, the human observer segmentation corresponds very closely to the zero velocity crossing segmentation output (Fod et al. 2002), where a segment is located each time there is

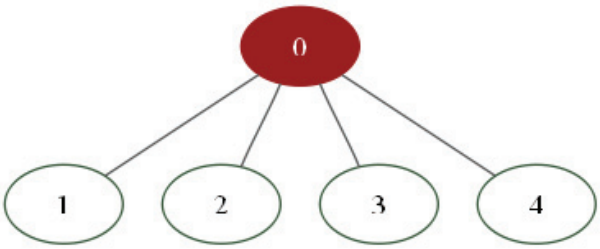

Fig. 6. Resulting tree structure. Here, motion primitive 1 = both arms lower, motion primitive 2 = both arms raise, motion primitive 3 = bend retract, and motion primitive 4 = bend down.

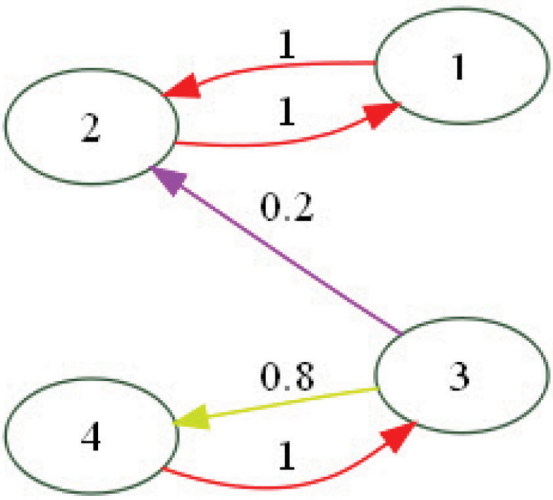

Fig. 7. Resulting motion primitive graph structure. Here, motion primitive 1 = both arms lower, motion primitive 2 = both arms raise, motion primitive 3 = bend retract, and motion primitive 4 = bend down.

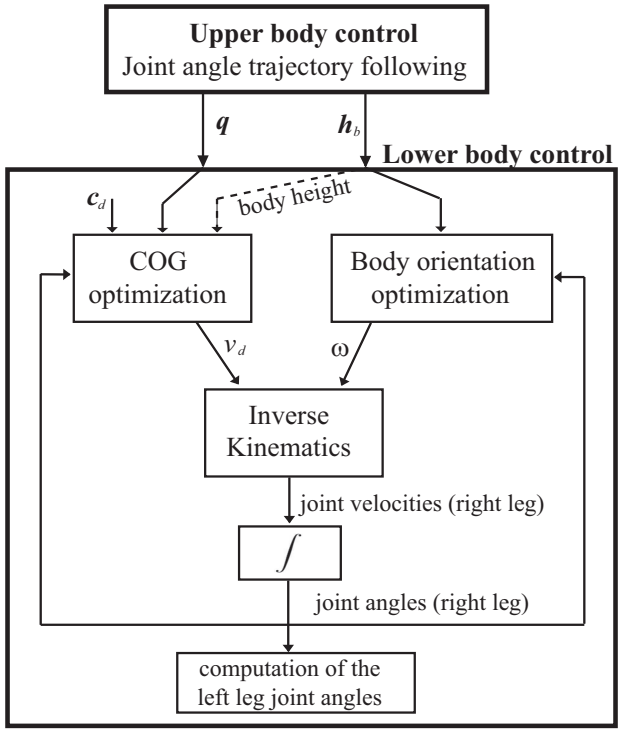

Fig. 8. Control system: the control task is split up into joint tracking control for the upper body and a balancing control task for the lower body. The lower body joints also implement the hip height and orientation, while stepping motions are currently not considered in the controller. The balancing algorithm is based on a separate optimization of center of gravity position and the hip orientation.

a stop in movement, or a change in movement direction. The human observer data is considered as the ground truth, against which the automated system is measured. The manual labeling identifies a total of 751 motion segments in the entire 16 minute sequence.

The motion capture system (Kurihara et al. 2002) captures the Cartesian position of markers located on the body (for example, shoulder, elbow, wrist, hip, knee, etc.) with a sampling rate of 10 ms. In order to map this trajectory to our humanoid robot, we perform an inverse kinematics computation (Yamane and Nakamura 2003) based on a kinematic model. Unlike in previous works (Kulić et al. 2008b,a), where a generic 20-DoF human model was used, here the specific humanoid kinematic model corresponding to the IRT robot was used. To correct for the difference in height between the demonstrator and the robot, a scaling was applied to the vertical dimension of all marker data. This allows the joint angle trajectory to be learned directly, so that the learned trajectory can be easily applied to the humanoid robot.

The continuous time series data consisting of the robot base body and joint angles was used to learn the motion primitives and the motion primitive graph. A data flow diagram showing an overview of the entire process is shown in Figure 1. While in this case the learning was performed following data acquisition, the data was fed to the algorithm incrementally, simulating on-line acquisition, and the learning was performed at a rate faster than the real-time length of the data sequence, demonstrating the suitability of the proposed method for on-line learning.

Following a single presentation of the data sequence, the segmentation and clustering algorithm correctly extracts 17 of the 22 motion primitives, requiring 6–8 demonstrations of a motion primitive prior to building a model for that primitive. No spurious, incorrect motion primitives are extracted. The resulting tree structure is shown in Figure 10. Following a second presentation of the data sequence, all motion primitives are extracted correctly. For the extracted motion primitives following the first presentation, the false-positive error rate is 3.5% and the false-negative rate is 6.7%. Here, the false-positive error rate is defined as the number of motion primitives recognized incorrectly and the

Fig. 9. Motion capture and marker setup.

false-negative rate is defined as the number of motion primitives not recognized, even though there is a known model. Note that the algorithm does not attach word labels to the extracted motion primitives, these are generated manually by inspection of the extracted primitives. The motions generated from the both arm raise (BAR) and the right kick raise (RKR) motion primitive models are shown in Figures 11 and 12, respectively.

The resulting motion primitive graph consists of 17 nodes, encapsulating the majority of the 751 motion segments in the entire 16 minute sequence. The generated representation encapsulates 77% of the observed sequence. This number represents the percentage of motions recognized out of all motions presented.

To compare the proposed approach to existing techniques in the literature, the motion graph approach proposed by Yamane et al. (2009) was implemented and tested on the same dataset. Note that the approach of Yamane et al. does not require pre-segmentation, so the Kohlmorgen and Lemm segmentation was not applied to the dataset prior to graph formation. The system was tested on a computer with 32 GB of memory, but as the Yamane approach requires that all frames be stored, this memory size was not sufficient to handle the dataset. Owing to these memory constraints, it was not possible to generate a tree using the motion graph approach with the entire database, so the dataset was first downsampled to reduce the memory requirements. Using every second frame of the dataset, the resulting motion graph, using the parameter values recommended in Yamane et al. (2009), results in 1,306 nodes. The size of the motion primitive graph highlights a key difference between the proposed method and previous approaches (Kovar et al. 2002; Yamane et al. 2009). By using two levels of hierarchy to describe the motion, i.e. a motion primitive level and a higher-level graph, rather than a flat structure based on postures alone, a significant reduction in the graph size can be achieved. As can be seen from these results, the approach of Yamane et al. already encountered memory constraints with this size dataset, requiring downsampling, while the proposed method has significantly smaller memory requirements.

To verify performance on unseen data, the method is also tested by separating the training sequence into a training and testing set. In this experiment, the full learning algorithm is run on the first half of the data set. During this first half of the sequence, 10 leaf nodes are formed representing motion primitives. The training is then stopped, and recognition performance is tested on the second half of the dataset. The second half of the dataset is processed by applying the segmentation only. The primitive models formed during the first half of the dataset are then used to recognize motions in the second half, as data is processed on-line. No further updating of the primitive models takes place during the testing stage. The false-positive error rate is 4.8%, while the false-negative error rate is 19.9%.

Following motion primitive and motion primitive graph extraction, the motions were generated and implemented on the IRT humanoid robot, using the lower-level controller as described in Section 4.2.1.

Owing to the current hardware implementation, motions involving foot raising (such as kicking or walking) are manually removed from the motion graph prior to generating motion sequences. The final motion primitive graph resulting from the extracted primitives (with the foot raising motions removed) is shown in Figure 13. Once these motion primitives are removed, motion sequences are generated by random sampling of the output edges from each subsequent node on the motion primitive graph. Each motion primitive is generated using deterministic generation from the associated HMM (Kulić et al. 2007b). The concatenated sequence is then low-pass filtered and interpolated prior to being passed to the robot as a trajectory command. The low-pass filtering is necessary due to the deterministic generation approach, which results in discontinuous edges in the joint trajectory at state transitions. Using a different generation method, such as generating by sampling and averaging (Takan 2006) or based on the nearest example (Lee and Nakamura 2006) would eliminate the need for this step. Owing to the velocity limits of the robot, the generated trajectory was interpolated to generate motions at half the human execution speed.

Figure 14 shows frames from a motion primitive sequence executed by the robot. In this particular sequence,

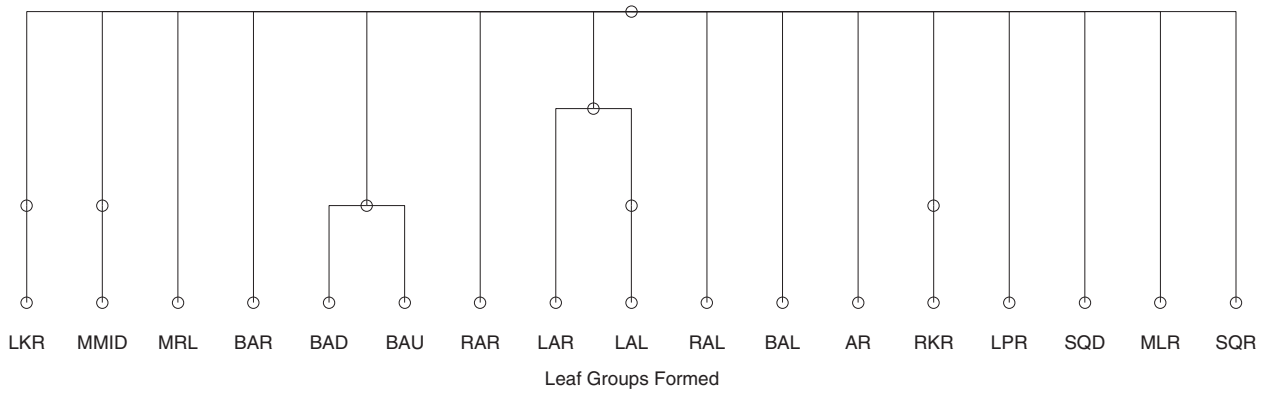

Fig. 10. Tree diagram of the extracted motion primitives following 16 minutes of observation. 'LKR' = left kick raise, 'MMID' = march mid step, 'MRL' = march right leg raise, 'BAR' = both arms raise, 'BAD' = bow down, 'BAU' = bow up, 'RAR' = right arm raise, 'LAR' = left arm raise, 'LAL' = left arm lower, 'RAL' = right arm lower, 'BAL' = both arms lower, 'AR' = arms ready, 'RKR' = right kick raise, 'LPR' = left punch retract, 'SQD' = squat down, 'MLR' = march left raise, 'SQR' = squat raise

Fig. 11. Frames from the generated sequence of the both arm raise (BAR) abstracted motion primitive.

Fig. 12. Frames from the generated sequence of the right kick raise (RKR) abstracted motion primitive.

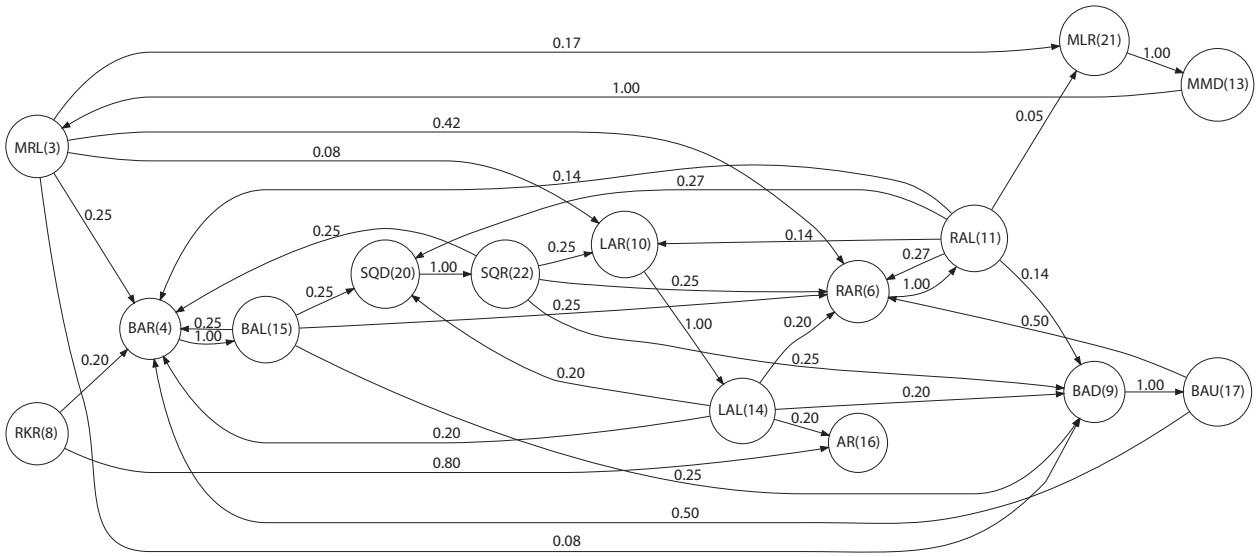

Fig. 13. Generated motion primitive graph. The node label corresponds to the tree node in Figure 10, while the number in brackets indicates the node formation order.

Fig. 14. Frames from the video capturing the experiment. A frame is extracted once for every second of the video sequence.

the robot executes the following sequence of primitives: 'left arm raise', 'left arm lower', 'bow down', 'bow up', 'right arm raise', 'right arm lower', 'bow down', 'bow up', 'right arm raise', 'right arm lower', 'squat down', 'squat up', 'left arm raise', 'left arm lower', 'bow down', 'bow up', 'both arm raise', 'both arms lower', and 'right arm raise'. As can be seen from the sequence and from the extracted motion graph (see Figure 13), the system correctly extracts sequencing information about primitives which must be consecutive, such as 'bow down' must be followed by 'bow up', as well as those motion primitives which can be followed by multiple other primitives, such as 'left arm down' can be followed by 'bow down' or 'squat down'. The full autonomous segmentation and clustering approach generates extracted primitives which are sufficiently accurate, such that they can be concatenated according to the learned motion primitive graph and used to generate smooth continuous motions, with only a simple low-pass filter. The sequence of motion primitives to be executed is determined in realtime based on the likelihood computed from the motion primitive graph of Figure 13. Since the motion primitive graph is static in this case, the realtime determination of the sequence may seem trivial and not critically important. In more realistic applications in the future, however, the motion primitive graph will be continuously updated and therefore will not be static but dynamic. Furthermore the sequence of motion primitives could be determined not only based on the motion primitive graph, but also by including other contextual information, such as sensory information, linguistic information, object/environment information, and so on to compute conditional probabilities for executing

a given sequence. Although these experiments are still within the rather simple information framework, they show the feasibility of computation of motion sequences based on probability and its use in realtime humanoid robot control.

A key advantage of the proposed approach is the significant reduction in size of the motion primitive graph. The smaller graph facilitates robot motion generation, as it reduces the search time for motion planning, as well as the memory requirements for graph storage. A key issue with the proposed approach is the granularity of the motion primitives. Longer motion primitives (for example, considering both an arm raise and lower as a single primitive) have the potential to further reduce the size of the graph, but may limit the variety of motions that the robot could perform. This tradeoff can be controlled by the performance of the segmentation algorithm, which can generate longer or shorter motion primitive candidates for clustering. One approach would be to generate shorter candidates for initial node formation, and then after sufficient data is observed, apply node merging for those nodes which are only connected to one other node. For example, in Figure 13, the node SQD is always followed by node SQR (transition probability of 1.0), and there are no other incoming paths to SQR, nor outgoing paths from SQD. These nodes could be merged into a single SQ node.

5. Conclusions and future work

This paper presents an approach for online, continuous learning of full body motion primitives and allowable motion primitive sequencing through observation of a human demonstrator. The proposed approach is general and can be applied to any type of observation data, including Cartesian data for the position of the human limbs and/or relevant objects in the environment, or joint angle data for a humanoid robot. The selected observation data is first autonomously segmented into potential motion primitives using stochastic segmentation (Kohlmorgen and Lemm 2001; Kulić and Nakamura 2008). Segmented motion primitive candidates are incrementally clustered to abstract generative models of the motion primitives (Kulić et al. 2008a). As each motion primitive is learned, it is also added to a motion primitive graph, which is incrementally updated to learn the relationship and sequencing rules of the motion primitives. The generated motion graph can then be used to generate extended motion sequences composed of motion primitives.

Future work will focus on implementing foot raising and walking motions on the humanoid robot, as well as motions involving interaction with the environment. In addition, techniques for generating motions with a desired goal or execution criteria based on the motion primitive graph will be developed, as well as the use of the motion primitive graph for human activity detection and motion prediction. We are also working on collecting a larger and more varied dataset of human motions, which will enable the learning and abstraction of a larger set of human motion primitives. This larger dataset will enable the construction of a larger motion primitive graph, so that the use of hierarchical motion primitive graphs can be further investigated.

Acknowledgments

The authors gratefully acknowledge the assistance of Mr Akihiko Murai and Dr Wataru Takano with the collection of the motion capture data. This work was conducted while all the authors were at the Nakamura Lab at the University of Tokyo. An early version of this work was presented at the HUMANOIDS2008 conference.

Funding

This work is supported by the Japanese Society for the Promotion of Science (Category S Grant-in-Aid for Scientific Research number 20220001). This research is also partially supported by Special Coordination Funds for Promoting Science and Technology, 'IRT Foundation to Support Man and Aging Society'.

References

Andry P, Gaussier P, Nadel J and Hirsbrunner B (2004) Learning invariant sensorimotor behaviors: a developmental approrach to imitation mechanisms. *Adaptive Behavior* 12: 117–140.

Asfour T, Gyarfas F, Azad P and Dillmann R (2006) Imitation learning of dual-arm manipulation tasks in humanoid robots. In *Proceedings of the IEEE International Conference on Humanoid Robots*, pp. 40–47.

Billard A, Calinon S and Guenter F (2006) Discriminative and adaptive imitation in uni-manual and bi-manual tasks. *Robotics and Autonomous Systems* 54: 370–384.

Billard A, Epars Y, Calinon S, Schaal S and Cheng G (2004) Discovering optimal imitation strategies. *Robotics and Autonomous Systems* 47: 69–77.

Breazeal C, Buchsbaum D, Gray J, Gatenby D and Blumberg B (2005) Learning from and about others: towards using imitation to bootstrap the social understanding of others by robots. *Artificial Life* 11: 1–32.

Breazeal C and Scassellati B (2002) Robots that imitate humans. *Trends in Cognitive Sciences* 6: 481–487.

Calinon S and Billard A (2007a) Active teaching in robot programming by demonstration. In *Proceedings of the IEEE International Conference on Robot and Human Interactive Communication*, pp. 702–707.

Calinon S and Billard A (2007b) Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*, pp. 255–262.

Calinon S, Guenter F and Billard A (2007) On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 37: 286–298.

Dillmann R, Rogalla O, Ehrenmann M, Zollner R and Bordegoni M (1999) Learning robot behaviour and skills based on human demonstration and advice: The machine learning paradigm. In *Proceedings of the International Symposium on Robotics Research*, pp. 229–238.

Dominey PF, Metta G, Nori F and Natale L (2008) Anticipation and initiative in human-humanoid interaction. In *Proceedings of the IEEE International Conference on Humanoid Robots*, pp. 693–699.

Erlhagen W, Mukovskiy A, Bicho E, Panin G, Kiss C, Knoll A, et al. (2006) Goal-directed imitation for robots: A bio-inspired approach to action understanding and skill learning. *Robotics and Autonomous Systems* 54: 353–360.

Ezaki H (2000) *Understanding Actions of Others Through Self-action Components*. Master's thesis, University of Tokyo, 2000 (in Japanese).

Fod A, Matarić MJ and Jenkins OC (2002) Automated derivation of primitives for movement classification. *Autonomous Robots* 12: 39–54.

Ijspeert AJ, Nakanishi J and Schaal S (2002) Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1398–1403.

Ilg W, Bakir GH, Mezger J and Giese MA (2004) On the representation, learning and transfer of spatio-temporal movement characteristics. *International Journal of Humanoid Robotics* 1: 613–636.

Inamura T, Toshima I, Tanie H and Nakamura Y (2004) Embodied symbol emergence based on mimesis theory. *The International Journal of Robotics Research*, 23: 363–377.

Ito M and Tani J (2004) On-line imitative interaction with a humanoid robot using a dynamic neural network model of a mirror system. *Adaptive Behavior* 12: 93–115.

Jain AK, Murty MN and Flynn PJ (1999) Data clustering: A review. *ACM Computing Surveys* 31: 264–323.

Janus B (2006) *On-line motion segmentation algorithm for mimesis model*. Master's thesis, University of Tokyo.

Janus B and Nakamura Y (2005) Unsupervised probabilistic segmentation of motion data for mimesis modeling. In *Proceedings of the IEEE International Conference on Advanced Robotics*, pp. 411–417.

Jenkins OC and Matarić M (2004a) Performance-derived behavior vocabularies: Data-driven acquisition of skills from motion. *International Journal of Humanoid Robotics* 1: 237–288.

Jenkins OC and Matarić M (2004b) A spatio-temporal extension to isomap nonlinear dimension reduction. In *Proceedings of the International Conference on Machine Learning*, pp. 441–448.

Kadone H and Nakamura Y (2005) Symbolic memory for humanoid robots using hierarchical bifurcations of attractors in nonmonotonic neural networks. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pp. 2900–2905.

Kadone H and Nakamura Y (2006) Segmentation, memorization, recognition and abstraction of humanoid motions based on correlations and associative memory. In *Proceedings of the IEEE International Conference on Humanoid Robots*, pp. 1–6.

Kajita S (2005) *Humanoid Robot*. Ohmsha, 2005 (in Japanese).

Kohlmorgen and Lemm S (2001) A dynamic hmm for on-line segmentation of sequential data. In *NIPS 2001: Advances in Neural Information Processing Systems*, pp. 793–800.

Kovar L, Gleicher M and Pighin F (2002) Motion graphs. In *Proceedings of the ACM SIGGRAPH*, pp. 473–482.

Krueger V, Kragic D, Ude A and Geib C (2007) The meaning of action: A review on action recognition and mapping. *Advanced Robotics* 21: 1473–1501.

Kulić D, Imagawa H and Nakamura Y (2009) Online acquisition and visualization of motion primitives for humanoid robots. In *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, pp. 1210–1215.

Kulić D and Nakamura Y (2008) Scaffolding on-line segmentation of full body human motion patterns. In *Proceedings of the IEEE/RJS International Conference on Intelligent Robots and Systems*, pp. 2860–2866.

Kulić D and Nakamura Y On-line segmentation of whole body human motion data for large kinematic models. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4300–4305.

Kulić D, Takano W and Nakamura Y (2007a) Incremental online hierarchical clustering of whole body motion patterns. In *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, pp. 1016–1021.

Kulić D, Takano W and Nakamura Y (2007b) Representability of human motions by factorial hidden markov models. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 2388–2393.

Kulić D, Takano W and Nakamura Y (2007c) Towards lifelong learning and organization of whole body motion patterns. In *Proceedings of the International Symposium of Robotics Research*, pp. 113–124.

Kulić D, Takano W and Nakamura Y (2008a) Combining automated on-line segmentation and incremental clustering for whole body motions. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2591–2598.

Kulić D, Takano W and Nakamura Y (2008b) Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden Markov chains. *International Journal of Robotics Research* 27: 761–784.

Kulić D, Takano W and Nakamura Y (2009) On-line segmentation and clustering from continuous observation of whole body motions. *IEEE Transactions on Robotics* 25: 1158–1166.

Kuniyoshi Y, Inaba M and Inoue H (1989) Teaching by showing: Generating robot programs by visual observation of human performance. In *Proceedings of the International Symposium on Industrial Robots*, pp. 119–126.

Kuniyoshi Y and Inoue H (1994) Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE Transactions on Robotics and Automation* 10: 799–822.

Kurihara K, Hoshino S, Yamane K and Nakamura Y (2002) Optical motion capture system with pan–tilt camera tracking and realtime data processing. In *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1241–1248.

Lee D and Nakamura Y (2006) Stochastic model of imitating a new observed motion based on the acquired motion primitives. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 4994–5000.

Liu S and Asada H (1992) Transferring manipulative skills to robots: representation and acquisition of tool manipulative skills using a process dynamics model. *Journal of Dynamic Systems, Measurement and Control* 114: 220–228.

Ott Ch, Lee D and Nakamura Y (2008) Motion capture based human motion recognition and imitation by direct marker control. In *Proceedings of the IEEE International Conference on Humanoid Robots*, pp. 399–405.

Nakanishi J, Morimoto J, Endo G, Cheng G, Schaal S and Kawato M (2004) Learning from demonstration adn adaptation of biped locomotion. *Robotics and Autonomous Systems* 47: 79–91.

Nicolescu MN and Matarić MJ (2001) Learning and interacting in human–robot domains. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 31: 419–430.

Nicolescu MN and Matarić MJ (2003) Natural methods for robot task learning: instructive demonstrations, generalization and practice. In *Proceedings of the International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pp. 241–248.

Nicolescu MN and Matarić MJ (2005) Task learning through imitation and human–robot interaction. In Dautenhahn K and Nehaniv C (eds), *Imitation and Social Learning in Robots, Humans and Animals: Behavioral, Social and Communicative Dimensions*. Cambridge: Cambridge University Press.

Ogata T, Sugano S and Tani J (2005) Open-end human–robot interaction from the dynamical systems perspective: mutual adaptation and incremental learning. *Advanced Robotics* 19: 651–670.

Okada M, Tatani K and Nakamura Y (2002) Polynomial design of the nonlinear dynamics for the brain-like information processing of whole body motion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1410–1415.

Rabiner LR (1989) A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77: 257–286.

Schaal S (2006) Dynamic movement primitives - a framework for motor control in humans and humanoid robotics. In Kimura H, Tsuchiya K, Ishiguro A and Witte H (eds), *Adaptive Motion of Animals and Machines*. Tokyo: Springer, pp. 261–280.

Schaal S, Ijspeert A and Billard A (2003) Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 358: 537–547.

Sidenbladh H, Black MJ and Sigal L (2002) Implicit probabilistic models of human motion for synthesis and tracking. In *Proceedings of the European Conference on Computer Vision*, pp. 784–800.

Startner T and Pentland A (1995) Visual recognition of american sign language using hidden markov models. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pp. 189–194.

Sutton RS and Barto AG (1998) *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Takano W (2006) *Stochastic segmentation, proto-symbol coding and clustering of motion patterns and their application to signifiant communication between man and humanoid robot*. PhD thesis, University of Tokyo.

Takano W, Yamane K and Nakamura Y (2005) Primitive communication of humanoid robot with human via hierarchical mimesis model on the proto symbol space. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pp. 167–174.

Takano W, Yamane K, Sugihara T, Yamamoto K and Nakamura Y (2006) Primitive communication based on motion recognition and generation with hierarchical mimesis model. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3602–3608.

Taylor GW, Hinton GE and Roweis S (2006) Modeling human motion using binary latent variables. In *Proceedings of the Conference on Neural Information Processing Systems*, pp. 1345–1352.

Yamaguchi Y, Yamane K and Nakamura Y (2008) A large-scale human motion database and its application to character animation. In *Proceedings of the JSME Conference on Robotics and Mechatronics*, pp. 1P1–J18 (in Japanese).

Yamane K and Nakamura Y (2003) Natural motion animation through constraining and deconstraining at will. *IEEE Transactions on Visualization and Computer Graphics* 9: 352–360.

Yamane K, Yamaguchi Y and Nakamura Y (2009) Human motion database with a binary tree and node transition graphs. In *Proceedings of Robotics: Science and Systems 2009*.