2008 8th IEEE-RAS International Conference on Humanoid Robots
December 1 ~ 3, 2008 / Daejeon, Korea

Incremental Learning of Full Body Motion Primitives for Humanoid Robots

Dana Kulić, Dongheui Lee, Christian Ott, Yoshihiko Nakamura

[1] *University of Tokyo, Japan* $\{dana, dhlee, ott, nakamura\}@ynl.t.u-tokyo.ac.jp$

***Abstract*—This paper describes 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. Motion segments are next 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 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 on the IRT humanoid robot.**

I. 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. Many algorithms have been proposed in the literature [1], [2]. 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. We propose an approach for on-line segmentation and clustering of whole body human motion primitives, and incremental learning of the relationship between the motion primitives for the formation of longer behaviors. The observed motion time series data stream is first stochastically segmented into potential motion primitive segments, based on the assumption that data belonging to the same motion primitive will have the same underlying distribution. The segmented motions are then passed to an incremental clustering algorithm which forms a tree representation of the learned motions, and abstracts each motion type into a generative model. Concurrently, 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 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.

A. Related Work

Breazeal and Scasellati [1] and Schaal et al. [2] provide reviews on motion learning by imitation. As noted by Breazeal and Scasellati, the majority of algorithms discussed in the literature assume that the motions to be learned are segmented and clustered a-priori, and that the model training takes place off-line.

Jenkins and Matarić [3] 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 spatio-temporal Isomap algorithm [4]. 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.

Hidden Markov Models have been a popular technique for human motion modeling, and have been used in a variety of applications, including skill transfer [5], sign language and gesture modeling [6] and motion representation [5], [7]. A common paradigm is *Programming by Demonstration* (PbD)

978-1-4244-2822-9/08/$25.00

[5]. While PbD is a general paradigm, in many of the systems demonstrated thus far, the number of motions is specified a-priori, the motions are clustered and trained off-line, and then a static model is used during the recognition phase. In previous research [8], [9], [10], [11], humanoid motion primitives have been encoded using Hidden Markov Models, and subsequently used for motion generation. However, the initial training of the models was carried out off-line, where all the training examples for each model were grouped manually.

Kulić et al. [12] 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.

Outside the robotics community, in the computer graphics domain, there has also been a long standing research effort to develop algorithms for realistic human-line motion generation for animated characters. Kovar et al. [13] 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. [14] develop an algorithm for building a motion graph via a binary tree clustering technique.

B. 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. 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 [15] algorithm for unsupervised segmentation. Next, the extracted segments are input into an automated clustering and hierarchical organization algorithm [16], [17]. The segmentation algorithm uses a Hidden Markov Model 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. Section 2 summarizes the segmentation algorithm, Section 3 overviews the clustering method, while Section 4 describes the formation of the motion primitive graph. In Section 5, the results of experiments verifying the algorithm on a humanoid robot trained with a continuous stream of human motion capture data is reported. Section 6 concludes the paper.

II. Probabilistic Segmentation

A modified version of the Kohlmorgen and Lemm segmentation algorithm [15] is used to automatically segment the continuous time series data into motion primitive candidates [18]. We provide a brief overview of the segmentation approach here, detailed analysis of the performance and parameter sensitivity of the standalone segmentation algorithm is provided in Kulić and Nakamura [18]. The same parameter settings as specified in [18] are used herein. The Kohlmorgen and Lemm segmentation algorithm [15] 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, and the density distribution of the embedded data is estimated over a sliding window of length W, via a standard density estimator with multivariate Gaussian kernels, centered on the data points in the window $\{\vec{x}_{t-w}\}_{w=0}^{W-1}$.

$$p_t(\mathbf{x}) = \frac{1}{W}\sum_{w=0}^{W-1}\frac{1}{(2\pi\sigma^2)^{n/2}}exp(-\frac{\|\mathbf{x}-\vec{x}_{t-w}\|^2}{2\sigma^2}), \qquad (1)$$

where σ is a smoothing parameter calculated proportional to the mean distance between each $\vec{x}_t$ 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. This distance can be calculated analytically in the case of mixtures of Gaussian density functions. The segmentation analysis is carried out by defining a Hidden Markov Model 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:

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

where $p(p_t(\mathbf{x})|s)$ is the probability of observing the window represented by $p_t(\mathbf{x})$ in state s, and d is the distance function based on the integrated square error. The initial state distribution 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.

$$a_{ij} = \begin{cases} \frac{k}{k+N-1} & \text{if } i = j; \\ \frac{1}{k+N-1} & \text{if } i \neq j \end{cases} \qquad (3)$$

where N is the number of states of the HMM. The Viterbi algorithm [19] can then be used to find the optimum state

sequence given the current set of observations. An on-line variant of the Viterbi algorithm is also developed [15], which incrementally builds the state path table as each new state is observed, by re-using the estimate of the likelihood and optimal state sequence from the previous time step.

To prevent the state list from growing to infinity as the number of observed data points increases, Kohlmorgen and Lemm [15] propose removing states following a segment away from that state. However, Janus [20] has found that this approach leads to over-segmenting, as the considered data range becomes too small (on the order of $5W$) and therefore the algorithm becomes more prone to local minima. Instead, Janus propose that the algorithm runs in batch-mode over a larger, fixed number of windows, and that windows be discarded in a FIFO manner. The Janus approach is adopted herein.

III. Incremental Motion Primitive Learning

Once the incoming time series data has been segmented into potential primitives, each segment is sequentially passed to the clustering module. In the proposed clustering approach [16], [17], 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 [21] is used to subdivide motion primitives. A Hidden Markov Model is used to abstract the observation sequences. The parameters of the model form the feature set of the data. These features are then used to define a distance measure between observation sequences, which is used for clustering.

The algorithm initially begins with one group (the root node). Each time a motion is observed, it is encoded into an HMM and compared to existing groups via a tree search algorithm, and placed into the closest group. Each time a group is modified, local clustering is performed within the exemplars of the group. If a cluster with sufficiently similar data is found, a child group is formed with this data subset. Therefore the algorithm incrementally learns and organizes the motion primitive space, based on the robot's lifetime observations. The tree formation process is illustrated in Fig. 1.

This 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.

A. The Clustering Approach

Each newly acquired observation sequence is encoded into a Hidden Markov Model. It is then compared to existing groups (if any). Here, the distance between two models can be calculated [19] by Equation 4.

$$D(\lambda_1, \lambda_2) = \frac{1}{T}[logP(O^{(2)}|\lambda_1) - logP(O^{(2)}|\lambda_2)], \quad (4)$$

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. This distance measure is based on the relative log likelihood that a generated sequence is generated by one model, as compared to a second model. It represents a Kullback-Leibler distance between the two models.

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

$$D_{thresh} = K_{maxGD} D^G_{max}. \quad (5)$$

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.

Once a new observation sequence is added to a group, a clustering procedure is invoked on that group, to determine if a subgroup may be formed. The complete link hierarchical clustering algorithm is used to generate the hierarchical tree structure within a group [21]. Clusters are formed based on two criteria: number of leaves in the subgroup, and the maximum proximity measure of the potential subgroup. To calculate the maximum distance measure, the average and standard deviation of the inter motion distances in the cluster is calculated. The distance cutoff is then calculated as a function of the distribution function:

$$D_{\text{cutoff}} = \mu - K_{\text{cutoff}}\sigma \quad (6)$$

where D_{cutoff} is the distance 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 which is discussed later, μ is the average distance between observations, and σ is the standard deviation among all the distances in the node.

If a new subgroup is generated, a new group model is trained using the raw observation sequences from all the group elements. The generated model is subsequently used by the robot to generate behaviors. The group model replaces the individual observations in the parent node. If one of the group elements allocated to the new cluster is already a group model, the generated motion sequence based on that model is used for the training. In this case, a modified form of the re-estimation formulas for multiple observation sequences [19] is used. The

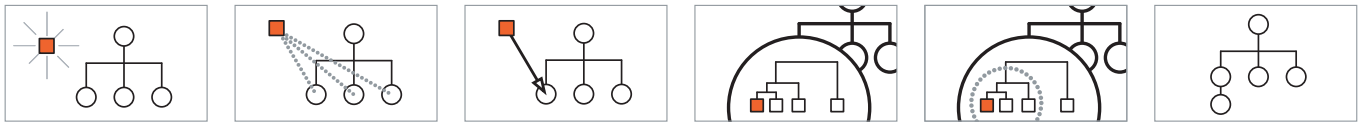

Fig. 1. Overview of the Incremental Clustering Algorithm (A square represents a data sequence, and a circle represents a group). (a) a new observation sequence is observed and encoded as an HMM; (b) the observation sequence is compared to existing groups via tree search; (c) the new sequence is placed 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.

algorithm is modified by over-weighting the group models, in order to account for the fact that there are multiple observation sequences stored in the generated model, and therefore more weight should be given to the group model, as compared to the individual observation sequences.

Once a cluster node has been formed, the group model for the node constitutes the abstraction of the motion primitive. To generate a motion trajectory for the robot from the group model, the deterministic motion generation method is used [22]. In this method, at each time step, the state duration is first estimated from the state transition model, and the subsequent state is selected by a greedy policy. The output observation vector is then generated by a greedy policy on the output model. The resulting reference trajectory is then low-pass filtered and passed to a low level controller, to ensure that dynamic and stability constraints are satisfied.

The clustering performance and the shape of the resulting motion primitive tree structure is dependent on the K_{cutoff} parameter, which controls cluster formation. Selecting a large K_{cutoff} value results in very similar cluster groupings, and reduces tree structure error, but makes clusters slower to form. A detailed analysis of the performance of the standalone clustering algorithm, and its parameter sensitivity is performed in Kulić et al. [12]. The same parameter values are used herein. It is also possible to reduce the sensitivity to the K_{cutoff} parameter by performing online correction of clustering errors in a slower outer loop [23].

IV. 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.

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 III), 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 proposed approach is similar to the motion graph approach employed for graphics character animation [13], [14]. However, a key difference from the motion graph approach is that the graph is composed of motion primitives, rather than individual postures. 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.

In the current implementation, only the leaf nodes 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. 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 if fewer, and searching at the lower level within the subspace identified by the initial higher level search.

V. Experiments

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. A total of 34 markers was used. The data set consists of 17 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. The subject performs a total of 751 motion segments. 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 coarticulation, i.e., motions which may partially overlap when executed sequentially.

The motion capture system [24] captures the Cartesian position of markers located on the body (for example, shoulder, elbow, wrist, hip, knee, etc.) with a sampling rate of 10ms. In order to map this trajectory to our humanoid robot, we perform an inverse kinematics computation based on a simplified kinematic model.

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. 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 algorithm correctly extracts 22 of the 25 motion primitives. For the extracted motion primitives, the false positive error rate is 3.5%, and the false negative rate is 6.7%. In this case, a false positive error indicates that a motion is mis-recognized as a different motion, for example a Left Kick Extend motion is misclassified as a Left Kick Retract motion. A false negative error indicates that a known motion was not recognized. The resulting tree structure is shown in Figure 3. 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 motion primitive graph resulting from the extracted primitives is shown in Figure 4.

Following motion primitive and motion primitive graph extraction, the motions were generated and implemented on the 38 degrees of freedom IRT humanoid robot. During the experiments we used only the 3 joints actuating the head, the 7 joints in each of the arms, the 6 joints in each of the legs, and the 1 joint at the hip, while the additional 8 degrees of freedom in the fingers and toes were not used.

The robot is controlled from two real-time computers connected via a network, one for the upper and one for the lower body. The desired motion trajectory is commanded in real-time to the upper-body computer via a UDP socket interface. In the current implementation, the upper body controller implements position control to follow the arms and waist joint trajectory after applying simple filtering and velocity constraints. The resulting motion is also transmitted to the lower body computer, where a position based balancing controller aims at maintaining the center of gravity of the robot at a fixed point, and at following the desired orientation and height of the hip of the humanoid robot. The current controller implementation does not require measurements of the ground reaction forces, but can only follow motions where both feet maintain contact with the ground. The balancing control algorithm is briefly shown in Figure 2. It is based on a velocity level inverse kinematics for the center of gravity and the base link orientation (see also [25]). In particular, the COG displacement due to the upper body motion should be compensated. Therefore, a simple optimization is used for computing a compensating ”COG velocity” taking account of the current upper body configuration and the desired height of the base link. Since both feet maintain contact with the ground, we only have to solve for the the joint angles of one leg, while the configuration of the other leg can be obtained in a second step from the closed kinematic loop.

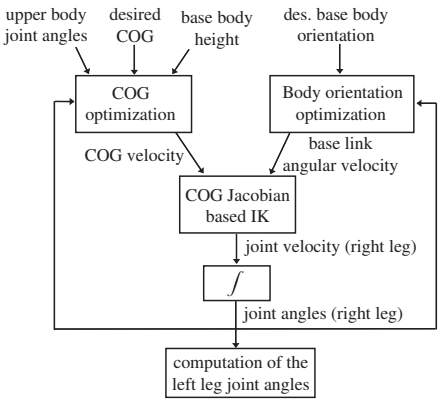

Fig. 2. Balancing Algorithm

Due 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. 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 model. The concatenated sequence is then low pass filtered and interpolated prior to being passed to the robot as a trajectory command. Due to the velocity limits of the robot, the generated trajectory was interpolated to generate motions at half speed compared to the human execution.

Figure 5 shows frames from a motion primitive sequence executed by the robot. In this particular 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”, ”Right Arm Raise”. As can be seen from the sequence and from the extracted motion graph (see Figure 4), 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”.

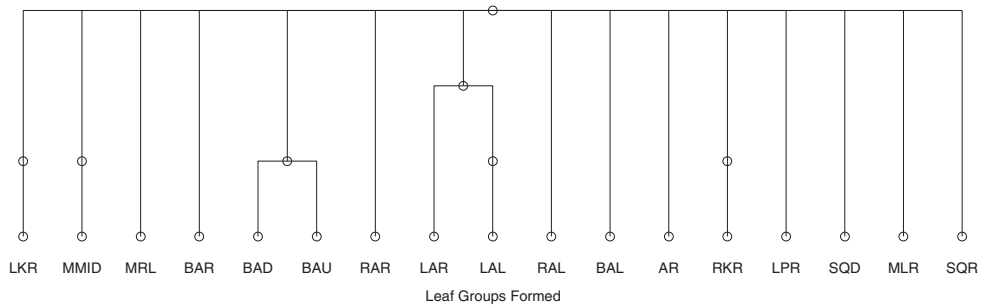

Fig. 3. Tree diagram of the extracted motion primitives following 15 minutes of observation. 'LKR' = Left Kick Raise, 'MMID' = March mid step, 'MRL' = March Right Leg Raise, 'BAR' = Both Arms Raise, 'BAD' = Bow Down, 'BAU' = Bow Up, 'RAR' = Right Arm Raise, 'LAR' = Left Arm Raise, 'LAL' = Left Arm Lower, 'RAL' = Right Arm Lower, 'BAL' = Both Arms Lower, 'AR' = Arms Ready, 'RKR' = Right Kick Retract, 'LPR' = Left Punch Retract, 'SQD' = Squat Down, 'MLR' = March Left Raise, 'SQR' = Squat Raise

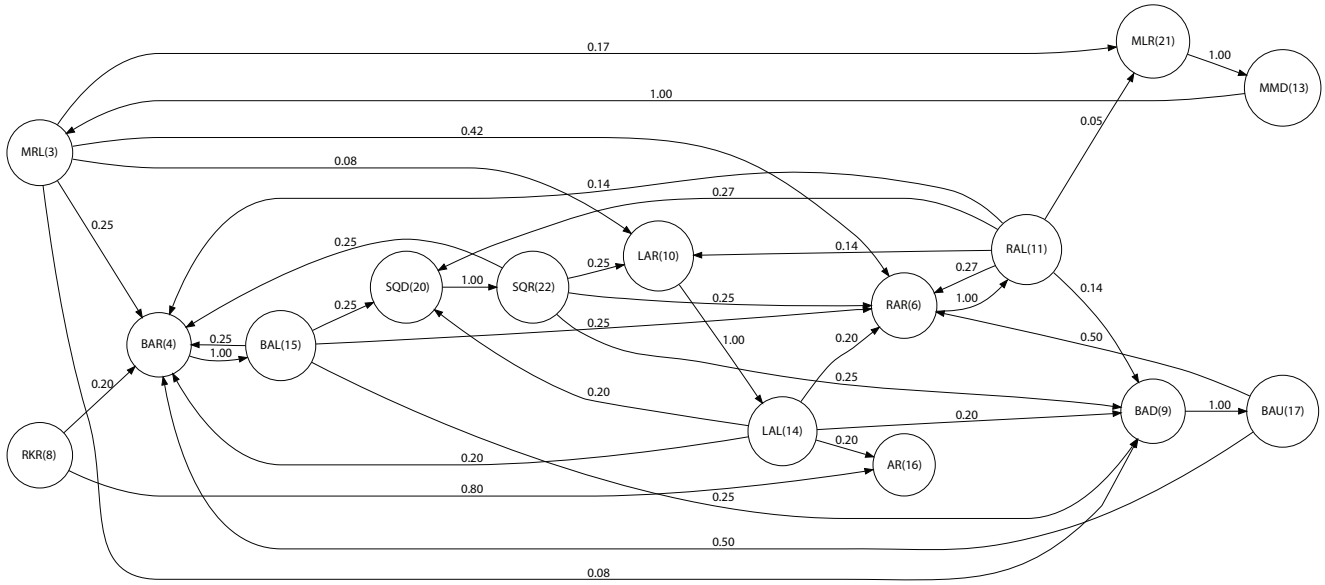

Fig. 4. Generated Motion Primitive Graph. The node label corresponds to the tree node in Figure 3, while the number in brackets indicates the node formation order.

A second key result is that 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.

VI. 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 observed human motion is first converted into robot joint angle data via online inverse kinematics. The joint angle data is then autonomously segmented into potential motion primitives using stochastic segmentation [15], [18]. Segmented motion primitive candidates are incrementally clustered to abstract generative models of the motion primitives [26]. 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.

Acknowledgment

The authors gratefully acknowledge the assistance of Mr. Akihiko Murai and Dr. Wataru Takano with the collection

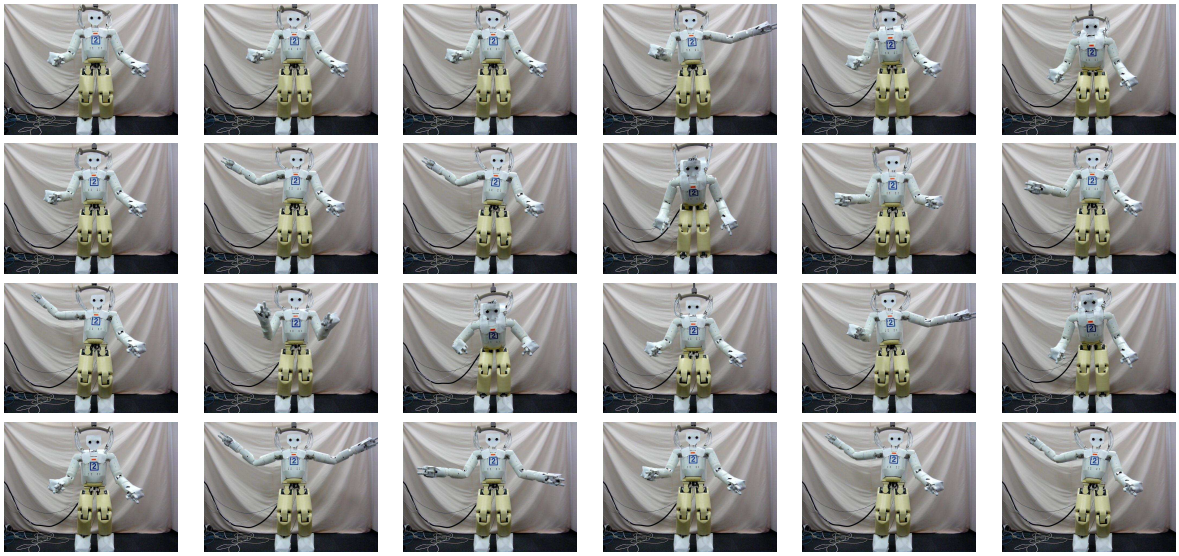

Fig. 5. Frames from the video capturing the experiment. A frame is extracted once for every 2s of the video sequence.

of the motion capture data. This work is supported by the Japanese Society for the Promotion of Science grant 18.06754 and Category S Grant-in-Aid for Scientific Research 20220001. This research is partly supported by Special Coordination Funds for Promoting Science and Technology, IRT Foundation to Support Man and Aging Society.

REFERENCES

[1] C. Breazeal and B. Scassellati, "Robots that imitate humans," *Trends in Cognitive Sciences*, vol. 6, no. 11, pp. 481–487, 2002.

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

[3] O. C. Jenkins and M. Matarić, "Performance-derived behavior vocabularies: Data-driven acquisition of skills from motion," *International Journal of Humanoid Robotics*, vol. 1, no. 2, pp. 237–288, 2004.

[4] ——, "A spatio-temporal extension to isomap nonlinear dimension reduction," in *International Conference on Machine Learning*, 2004, pp. 441–448.

[5] R. Dillmann, O. Rogalla, M. Ehrenmann, R. Zollner, and M. Bordegoni, "Learning robot behaviour and skills based on human demonstration and advice: The machine learning paradigm," in *International Symposium on Robotics Research*, 1999, pp. 229–238.

[6] T. Startner and A. Pentland, "Visual recognition of american sign language using hidden markov models," in *International Conference on Automatic Face and Gesture Recognition*, 1995, pp. 189–194.

[7] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 286–298, 2007.

[8] A. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robotics and Autonomous Systems*, vol. 54, pp. 370–384, 2006.

[9] H. Ezaki, "Understanding actions of others through self-action components," Master's thesis, University of Tokyo, 2000, in Japanese.

[10] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *The International Journal of Robotics Research*, vol. 23, no. 4–5, pp. 363–377, 2004.

[11] D. Lee and Y. Nakamura, "Stochastic model of imitating a new observed motion based on the acquired motion primitives," in *IEEE International Conference on Intelligent Robots and Systems*, 2006, pp. 4994–5000.

[12] D. Kulić, W. Takano, and Y. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains," *International Journal of Robotics Research*, vol. 27, no. 7, pp. 761–784, 2008.

[13] L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," in *ACM SIGGRAPH*, 2002, pp. 473–482.

[14] Y. Yamaguchi, K. Yamane, and Y. Nakamura, "A large-scale human motion database and its application to character animation," in *JSME Conference on Robotics and Mechatronics*, 2008, pp. 1P1–J18, in Japanese.

[15] J. Kohlmorgen and S. Lemm, "A dynamic hmm for on-line segmentation of sequential data," in *NIPS 2001: Advances in Neural Information Processing Systems*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds., vol. 14, 2002, pp. 793–800.

[16] D. Kulić, W. Takano, and Y. Nakamura, "Incremental on-line hierarchical clustering of whole body motion patterns," in *IEEE International Symposium on Robot and Human Interactive Communication*, 2007, pp. 1016–1021.

[17] ——, "Towards lifelong learning and organization of whole body motion patterns," in *International Symposium of Robotics Research*, 2007, pp. 113–124.

[18] D. Kulić and Y. Nakamura, "Scaffolding on-line segmentation of full body human motion patterns," in *IEEE/RJS International Conference on Intelligent Robots and Systems*, 2008, pp. 2860–2866.

[19] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[20] B. Janus, "On-line motion segmentation algorithm for mimesis model," Master's thesis, University of Tokyo, 2006.

[21] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[22] D. Kulić, W. Takano, and Y. Nakamura, "Representability of human motions by factorial hidden markov models," in *IEEE International Conference on Intelligent Robots and Systems*, 2007, pp. 2388–2393.

[23] D. Kulić and Y. Nakamura, "Incremental learning and memory consolidation of whole body motion patterns," in *International Conference on Epigenetic Robotics*, 2008, pp. 61–68.

[24] K. Kurihara, S. Hoshino, K. Yamane, and Y. Nakamura, "Optical motion capture system with pan-tilt camera tracking and realtime data processing," in *IEEE International Conference on Robotics and Automation*, vol. 2, 2002, pp. 1241–1248.

[25] Ch. Ott, D. Lee, and Y. Nakamura, "Motion capture based human motion recognition and imitation by direct marker control," in *IEEE International Conference on Humanoid Robots*, 2008.

[26] D. Kulić, W. Takano, and Y. Nakamura, "Combining automated on-line segmentation and incremental clustering for whole body motions," in *IEEE International Conference on Robotics and Automation*, 2008, 2591–2598.