Humanoid human-like reaching control based on movement primitives

Minh Tuan Tran
Universite Paul Sabatier

Philippe Souères
LAAS-CNRS Toulouse France

Michel Taix
Universite Paul Sabatier

Manish Sreenivasa
University of Wollongong, manishs@uow.edu.au

Christophe Halgand
LAAS-CNRS Toulouse France
Humanoid human-like reaching control based on movement primitives

Abstract
This paper deals with the problem of generating realistic human-like reaching movements from a small set of movement primitives. Two kinds of movement databases are used as reference. The first one is obtained numerically, by applying biological principles of motor control on the dynamic model of the robot arm. The second one is obtained by recording reaching movements of human subjects. From these databases, primitives are extracted and analyzed by using Principal Component Analysis. An original generalization method is then proposed for generating movements that did not belong to the initial database. We show that twenty primitives allow to produce new movements, having characteristics similar to that of humans. Experiments on the humanoid robot HRP-2 are presented to illustrate the result.

Keywords
primitives, humanoid, human-like, control, movement, reaching

Disciplines
Engineering | Science and Technology Studies

Publication Details
Humanoid human-like reaching control based on movement primitives

Tran Minh Tuan, Philippe Souères, Michel Taïx, Manish N. Sreenivasa and Christophe Halgand

Abstract—This paper deals with the problem of generating realistic human-like reaching movements from a small set of movement primitives. Two kinds of movement databases are used as reference. The first one is obtained numerically, by applying biological principles of motor control on the dynamic model of the robot arm. The second one is obtained by recording reaching movements of human subjects. From these databases, primitives are extracted and analyzed by using Principal Component Analysis. An original generalization method is then proposed for generating movements that did not belong to the initial database. We show that twenty primitives allow to produce new movements, having characteristics similar to that of humans. Experiments on the humanoid robot HRP-2 are presented to illustrate the result.

I. INTRODUCTION

One of the main theories of motor control in biology conjectures that the Central Nervous System (CNS) of primates uses a finite set of elementary motor components, called primitives, to generate movements. This theory was suggested by numerous biological experiments. At the kinematic level, the primitives are sometimes described as joint covariations. For example, covariations of ankle, knee and hip joints during bending movements and walking were described in [1] and [2]. A similar coactivation between whole-body links during reaching movements was reported in [3] and [4]. Motor primitives or synergies were also pointed out at the muscular level, from EMG measurements. For instance, five muscular synergies involved in postural control were described in [5]. The encoding of motor primitives was also shown at the neural level. The well known experiment by Mussa-Ivaldi and Bizzi [6], showing that local stimulation of the spinal chord induce different leg movements in frogs, is a good illustration of this. Recently, an attempt at modeling these experimental results in the control theory framework was proposed in [7]. For roboticists, the idea of reducing the complexity of control by using a finite set of movement primitives is very attractive. It offers a promising alternative to the computation of inverse kinematics and cost minimization to cope with the high redundancy of anthropomorphic structures. This idea has already motivated some applications. Two primitives were used by Hauser et al. [8], for controlling the balance of a small size humanoid robot. Lim et al. [9], used Principal Component Analysis (PCA) to extract primitives from the captured movement of a human arm modeled as 4 Degrees Of Freedom (DOF) chain. These primitives were used as basis functions for parameterizing new realistic robot movements. In the study of Chhabra et. al. [10], the authors used a nonnegative matrix factorization method to extract primitives from a database of control signals to control a 2 DOF arm. Interestingly, it was shown that new movements can be learned faster in the primitive space than in the control space [11] . In order to introduce feedback in the control, Todorov et al. [12] considered sensorimotor primitives. This kind of primitive was applied to the control a 2 DOF arm.

In this paper, we propose a method that allow to characterize a set of kinematic primitives from a database of movements and use them to produce realistic human-like movements with a 6 DOF humanoid robot arm (Fig. 1-b). The movements are expected to have the main kinematic characteristics of human movements, namely: almost straight hand trajectories with a possible weak regular curvature, and bell-shaped velocity profiles. In order to develop a sufficiently generic method that allow an easy transfer from the human movement to the robot motion, we based our study on two different databases. The first database was obtained by simulation by applying a biological model of motor control on the dynamic model of HRP-2’s arm [13], [14]. The second database was constructed by recording human reaching movement on different subjects using a motion capture system. From both databases, primitives are extracted by using PCA, and the reconstruction process, which allows to express each movement as a linear combination of these primitives, is analyzed. Then, an original generalization method is proposed to generate new movements from this basis of primitives. The proposed method consists in solving a low-dimensional minimization problem to determine the
weighting coefficients that allow to approach the minimum-
jerk solution with the best precision. Depending on the
number of primitives, the computation-time and the preci-
sion of the reconstruction processes, are discussed. Finally,
examples of reaching movements computed by this method
are presented on the humanoid robot HRP-2.

II. MOVEMENT DATABASE GENERATION

The first part of the work was devoted to the synthesis
of two databases of reaching movements. The first database
was obtained in simulation, by applying the biological model
of motor control proposed by Guigon et al. [13], on the
dynamical model of HRP-2’s arm. The application of this
model to the control of HRP-2’s reaching movements was
described in a previous work [14]. This result is briefly
recalled here. Guigon’s model is based on the idea that the
CNS processes dynamic efforts (inertial, velocity dependent)
and static efforts (elastic, gravitational) separately, and that
the energy of motoneurons is continuously minimized during
movement. In order to apply these principles to the control
of HRP-2, a global model was considered, which contained
the dynamics of the 6 DOF of the robot arm. For each DOF, an
additional filter simulating the dynamics of a pair of virtual
antagonist muscles was included. From this model, we
determined the reaching trajectories minimizing the energy of
the virtual motoneurons, while disregarding the gravitational
term in the model, according to the separation principle. The
robot arm movements obtained using this approach turns out
to be very realistic and exhibit the main kinematic features
of human reaching motions. However, the main drawback
of this approach is the computation time. Indeed, several
minutes were necessary to generate a movement, making the
method unusable for on-line control. This motivated our wish
to determine a finite set of primitives from which the optimal
solutions could be determined faster.

The second database was obtained by recording human
movements using a motion capture technology. Arm mo-

tion of 3 participants were recorded using infrared markers
attached to the shoulder, elbow and wrists. Accuracy of
capture was less than 1mm, with a frequency of 100Hz. Participants
were asked to perform a set of reaching move-
ments while standing up. We considered 32 target positions,
regularly-spaced in two parallel grids located in front of the
participant’s shoulder (Fig. 1). The reaching target was a
small ball at the end of a stick that was manipulated by the
experimenter. The target was placed at one of the 32 position,
and the participants heard a sound to indicate that they should
start moving their hand to the target. Upon reaching the
current target, the next target was randomly chosen from
the other 31 positions, and the experimenter moved the stick
to this new position. Movements were executed such that all
32 targets were reached from each other, giving a total of
993 movements per participant.

The same sequence of task was used for simulating the
reaching movements with the dynamic model of HRP-2, in
order to construct the first database. However, in the case
of the robot, a scaling factor was applied to the dimension
of the target setup and its distance to the robot in order to
obtain comparable values. For both the human subjects and
the robot, the database were constructed for a model of arm
including six joints: three at the shoulder, two at the elbow
and one at the wrist. Note that the remaining DOF of the
human arm were not considered for this application. In order
to simplify the primitive extraction, the robot movements
were all executed with a duration of 1s discretized in 100
time values. In the same way, the human reaching trajectories
were normalized to 1s within the same sampling of 0.01s. So,
for both the robot and the humans, each reaching movement
was described by 600 values encoding the variation of 6
angles.

III. PRIMITIVES EXTRACTION

We are looking for open-loop primitives with no sensory
feedback. In the literature, such primitives are defined as
time functions to be modulated in amplitude by weighting
coefficients [9], [8], [15]. The most general and simplest way
of modeling the problem is to consider that movements can
be express as linear combination of these primitives. For our
problem, each movement of the 6 DOF arm can be repres-
ented by a joint trajectory \( u(t) \in \mathbb{R}^6 \). A database of \( M \)
movements is then described by a set of such trajectories:
\[
U_m(t) = (U_m^1(t), U_m^2(t), \ldots, U_m^6(t)) \in \mathbb{R}^6, \quad m = 1, \ldots, M, \quad t \in [0, T].
\]
For this database, the primitive extraction problem is to deter-
mine \( K \) time-functions \( \Phi_k(t) = (\Phi_k^1(t), \Phi_k^2(t), \ldots, \Phi_k^6(t)) \in \mathbb{R}^6, k = 1, \ldots, K, t \in [0, T] \), such that, for all \( m = 1, \ldots, M \), it
is possible to determine \( K \) real coefficients \( \alpha_k^m \), verifying:

\[
U_m(t) = \sum_{k=1}^{K} \alpha_k^m \Phi_k(t)
\]

In this expression, the functions \( \Phi_k(t) \) represent the expected
primitives and the coefficients \( \alpha_k^m \) are weighting
the contribution of each primitive \( \Phi_k(t) \) into the movement
\( U_m(t) \). Each \( \Phi_k(t) \) has then the same dimension as \( U_m(t) \), and
constitutes a particular joint trajectory. The determination
of Eqn. (1) states two problems. The first one is to determine
the number, \( K \), of primitives that are necessary to reach a
given precision and the second one is to characterize these \( K \)
primitives. To reduce the complexity of the control problem,
\( K \) needs to be as small as possible, compared to the number
of parameters necessary to encode the trajectory \( U_m(t) \). For
a continuous-time problem, this number is infinite. However,
in practice, the problem is solved by considering a sampling
of the time interval \([0, T]\) in \( J \) elements. For the 6 DOF arm
we consider, \( K \) needs to be small with respect to the 6J angle
values that encode each movement. In other terms, once the
\( K \) primitives are determined, it is sufficient to compute \( K 
\) coefficients \( \alpha_k^m \), \( k = 1, \ldots, K \), to generate the movement \( U_m(t) \)
over \([0, T]\), instead of 6J variables. The second problem is to compute the canonical time-functions \( \Phi_k \). Considering
the notation of Eqn. (1), each movement \( U_m \) and each primitive
\( \Phi_k \) is defined by a \( 6 \times J \) matrix as follows:
The curve corresponding to the three human subjects and the robot arm are represented. In addition, the blue curve represents the mean of the three human subjects.

\( U_m = \begin{bmatrix} u_{m1}^1 & u_{m2}^1 & \ldots & u_{mJ}^1 \\ u_{m1}^2 & u_{m2}^2 & \ldots & u_{mJ}^2 \\ \vdots & \vdots & \ddots & \vdots \\ u_{m1}^M & u_{m2}^M & \ldots & u_{mJ}^M \end{bmatrix}, \Phi_k = \begin{bmatrix} \phi_k^1 & \phi_k^2 & \ldots & \phi_k^J \end{bmatrix} \) (2)

where, \( u_{mj}^m, h = 1, \ldots, 6, m = 1, \ldots, M, \ j = 1, \ldots, J \) are real numbers. On this basis, the reconstruction error relative to the \( M \) movements of the database, \( U_m, m = 1, \ldots, M \) is defined by:

\[
E = E(\alpha_m^m, \Phi_k) = \sum_{m=1}^{M} \sum_{h=1}^{6} \sum_{j=1}^{J} (u_{mj}^m - \sum_{k=1}^{K} \alpha_k^{m} \phi_k^j)^2
\]

Solving this problem is then reduced to the determination of \( K \) primitives, with \( K \) as small as possible, \( K << 6J \), but sufficiently large to guarantee that the reconstruction error will be lower than the expected precision threshold. This compromise will be discussed in the next section. Among the existing techniques that allow to cope with this kind of problem (see [18] for an overview), Principal Component Analysis (PCA), is intrinsically well adapted, simple and well-performing. It has already been successfully tested in neurosciences and in robotics [1], [3], [16], [9]. In order to apply PCA, it is more convenient to use a vector notation of data. So, instead of using the matrix notation (Eqn. (2)), each movement \( U_m, m = 1, \ldots, M \) will be described by the following vector expression:

\[
U_m = (u_{11}, u_{12}, \ldots, u_{1J}, u_{21}, \ldots, u_{2J}, \ldots, u_{61}, \ldots, u_{6J})^T
\]

where, each \( u_{hj}, h = 1, \ldots, 6, j = 1, \ldots, J \) is the angular value of joint \( h \) at time \( j \). The \( M \) column vectors \( U_m \) of dimension \( N = 6J \) are then gathered to compose a \( M \times N \) matrix. The \( N \times N \) covariance matrix is then computed. The eigenvectors \( e_i, i = 1, \ldots, N \) and eigenvalues \( \lambda_i, i = 1, \ldots, N \) of this covariance matrix are then determined and ordered. The eigenvalue \( \lambda_i \) represents the variance of data in the direction of the corresponding eigenvector \( e_i \). The larger \( \lambda_i \) is, the more its associated component \( e_i \) is dominant in the representation of data. The question is then to determine the number \( K \), such that the first \( K \) principal components \( e_k, k = 1, \ldots, K \) constitute a basis of primitives from which the whole database can be expressed with the required precision. Note that each primitive \( e_k \) is itself the representation of a six-joints trajectory. Finally, for each movement \( U_m \), the coefficient corresponding to the primitive \( e_k \) is determined by computing the scalar product between these two vectors:

\[
\alpha_k^m = \langle U_m, e_k \rangle, \quad m = 1, M, k = 1, \ldots, K.
\]

Note that the application of PCA requires the data to be centered. This condition was almost exactly satisfied for each database.

PCA was separately applied to the robot database and the human databases described in section II. In each case, we observed that the variance accounted for (VAF) by the \( K = 8 \) first primitives was more than 96\%, while it was more than 99.5\% with the \( K = 20 \) first primitives. Using the previously described approach, we determined the coefficients that allowed us to reconstruct the whole set of movements by a linear combination of these primitives. These reconstructed trajectories were then compared with the original ones. To compute the reconstruction error for a whole database, we used the Root Mean Squared Error (RMSE), which is expressed as follows:

\[
RMSE = \sqrt{\frac{E}{(M \times 6 \times J)}}
\]

where, \( E \) is defined by Eqn. (3). Fig. 2 shows the decay of the RMSE as a function of the number \( K \) of primitives, for each database. The decay rate of the RMSE is similar for the three human subjects, while it is slightly different for the robot arm. For human subjects, beyond the first 8 primitives, each addition of a new primitive induces almost the same error reduction. Whereas, for the robot model, the important decay rate between the 8th and the 16th primitive shows that these primitives still contain an important part of information. Interestingly, the curves corresponding to the human subjects and the robot become roughly parallel after the 18th primitive, showing a certain level of similarity between both databases. Beyond this level, the regular gap between the curves seems to be due to the difference in the kinematics and the dynamics of the arm structures. As human subject are taller and heavier than the robot, the effect of inertia and masses are higher on the human arm than on the robot arm. This might induce a higher reconstruction error for human movements. In each case, the primitives computed with PCA allow to represent the original data with a good precision. Depending on the expected level of precision, the number of primitives can be easily chosen. For instance, \( K = 16 \) primitives allows to reproduce the robot database with a mean error of 0.02 radians, \((1.2^\circ)\). The same number of primitives allows to represent the database of subject 1 with a mean error of 0.03 radians, \((1.7^\circ)\). As an example, Fig. 3 shows two trajectories reconstructed from these primitives and Fig. 4 shows the first 8 primitives for each database: the robot, subject 1, 2 and 3. The primitives represent the principal variations of the movements, classified by decreasing order of dominance variance. The first
primitives are rather regular, while the last more oscillatory one, allows to capture subtle variations of movements. Recall that each primitive describes the 6 joint trajectories. It is interesting to remark in Fig. 4, that the curves of primitives corresponding to the different database are not necessarily similar. This difference is due to the definition of primitives given by (1). Indeed, according to this relation, each primitive represents the simultaneous temporal evolution of the six arm joints, and these six trajectories are weighted by the same coefficient $\alpha^m_k$. This choice is biologically plausible, but leads to a different primitive basis of each database. Indeed, even though the subjects do the reaching movements in a similar way, the variations due to each subject may radically modify the expression of primitives.

It is important to note that some authors used PCA to extract primitive for each link separately [9]. As a consequence, the modulation of primitives is decoupled on each joint. In that case, it was shown that a lower number $K$ of primitives is sufficient for each link (usually between 2 and 4), but the number of coefficients is multiplied by $N$. Thus, if 3 primitives are necessary for each link of a 6 DOF arm, then 18 primitives will be necessary for representing the arm movements. We also tested this approach for our problem and we found that the variance accounted for by 3 primitives on each link was more than 96%. In particular, we found a certain similarity between the primitives for the subject and the robot arm. However, there is no fundamental difference between the two methods of primitive extraction. The number of variables are almost identical in both cases. By looking for primitives containing the 6 arm joints, our goal was to capture the joints co-activation. In this way, we expected to reduce the complexity, increase the precision of reconstructed movements, and find a basis of primitives well adapted to generalization. This last step is considered in the next section.

IV. GENERALIZATION

So far, we have shown that large databases of reaching movements can be expressed as linear combination of a small number of primitives. The question is now to determine if it is possible, from these primitives, to generate new movements having the characteristics of human movements. In other terms, we want to solve a generalization problem which can be stated as follows: considering a set of primitives $\Phi_k$, given the initial arm configuration and a reaching task defined by the the target position and the movement duration, determine the coefficients $\alpha_k$ that allow to express the trajectory as a linear combination of the primitives.

However, recall that the objective of this work is to generate realistic reaching movements rapidly, without having to solve a complex optimization problem as we did previously in [14]. Our first attempt to solve the generalization problem was to apply learning methods in order to characterize fast input-output interface. We tested different architectures of multi-layer feedforward neural networks which received as input the six values of initial arm joints, the three Cartesian coordinates of the target and the movement duration, and were expected to give the $\alpha_k$ weighting coefficients of the $K$ primitives as output. For some of these networks, the learning process on the 993 input-output pairs of each database took several hours. However, none of them succeeded in characterizing the $\alpha_k$ in a sufficiently generic way. In many cases, the reaching error between the final hand position and the target was too large and the movements were often not realistic. Increasing the number of primitives $K$ did not improve the quality of the result. It seemed that the information learned by the networks was not sufficiently rich to capture the movement characteristics. Furthermore, feedforward neural network does not seem to be well appropriate to cope with the sensibility of the weighting of motor primitives. Then, as the learning techniques we considered did not provide us
with satisfying results, we developed an original approach which consists in constraining the hand trajectory in the operational space. This method is described in the following section.

A. Method of trajectory constraint in the operational space

We want to determine the \( \alpha_k \) that allow to generate new movements that still have the characteristic of human movements. To this end, we propose to determine the \( \alpha_k \) which guarantee that the hand trajectory satisfies the minimum jerk condition [17]. In order to implement this reasoning, we formulate an optimization problem whose objective is to characterize the \( \alpha_k \) coefficients that allow to minimize the mean error between the hand trajectory and the minimum-jerk curve in the Cartesian space. As before, the time interval \([0,T]\) is sampled in \( J \) elements \( 0 = t_1, \ldots, t_J = T \). For \( j = 1, \ldots, J \), let us denote by \( q_j = (q_{1j}, q_{2j}, q_{3j}, q_{4j}, q_{5j}, q_{6j})^T \) the arm configuration at time \( t_j \) and \( H_j = f(q_j) \) the corresponding hand position in the Cartesian space, where \( f \) represents the direct kinematics associated to the 6 DOF arm model. Using this notation, the optimization problem can be stated as follows: Given \( K \) primitives \( \Phi_k \), \( k = 1, \ldots, K \), each one representing the six arm joint trajectories, the initial arm configuration \( q_1 \), and a reaching task defined by the target position \( H_f \), find the value of \( K \) real coefficients \( \alpha_k \), \( k = 1, \ldots, K \), minimizing the function \( E_{\text{jerk}} \) defined by:

\[
E_{\text{jerk}} = \sum_{j=1}^{J} ||f(q_j) - g(\tau_j, H_1, H_f)||
\]  

(5)

Recall that, according to the definition of primitives, the angular value \( \phi_{hj} \) of joint \( h \) at time \( t_j \), that appear in the above definition, is computed from the relation \( \phi_{hj} = \phi_{h1} + \sum_{k=1}^{K} \alpha_k \phi_{hjk} \), where \( \phi_{hjk} \), \( j = 1, \ldots, J \), is the angular value of the \( h^{th} \) joint of the \( k^{th} \) primitive at time \( t_j \), and \( g \) is the time-function representing the hand position along the reference minimum-jerk trajectory expressed by:

\[
g(\tau) = H_1 + (H_1 - H_f)(15\tau^4 - 6\tau^5 - 10\tau^3),
\]

(6)

where \( t \) is the normalized time and \( \tau_j = t_j/T \) the discrete normalized value that corresponds to time \( t_j \) in \([0,T]\). In this problem, the joint limits on the \( q_{hj} \) during the movement are also considered. They are defined by an inequality of the type:

\[
l_h \leq q_{hj} \leq u_h \quad \text{for} \ h = 1, \ldots, 6, \ j = 1, \ldots, J,
\]

(7)

where \( l_h \) and \( u_h \) are the upper and lower bounds of joint \( h \) respectively. Note that this problem is different from the one considered in [9]. Here, the objective is not to minimize a kinematic or dynamic criterion to determine the control signals, but instead to follow a reference curve, having the characteristics of human movement, in the Cartesian space. This problem is also different from the usual minimum-jerk problem in the sense that the joint trajectory of the arm is directly deduced from the knowledge of the \( \alpha_k \) coefficients. No additional computation is then required to compute the joint trajectories; a step which usually requires to solve the inverse kinematics problem with the minimization of some additional criterion to cope with redundancy. Furthermore, note that the generalization problem described by Eqn. (5), (6) and (7) is a simple optimization problem involving \( K \) real variables and \( 6J \) linear constraints, which can be solved by using standard techniques. For this work, we used the Matlab fmincon solver. The result we obtained is described in the next section.

B. Results

In order to test the proposed generalization approach, a large number of new movements were generated by considering the model of the robot arm. To this end, a shift
between $-10cm$ and $+10cm$ was applied in random in each direction of the initial target grid (see Fig. 1) to generate new targets, and angular increments between $-5^\circ$ and $+5^\circ$ were randomly added to each arm joint to specify new initial arm configurations. This allowed to generate a new large set of reaching tasks with different arm configurations. Note that the larger amplitude movements were about 1 meter long. In order to evaluate the compromise between rapidity and reaching precision, different values of $K$ between 6 and 20 were considered. Table I shows the average reaching error and computation time on all tested movements for each value of $K$. Logically, for a higher $K$ the computation time increases, while the reaching error decreases. It can be noted that the reaching error defined by the distance between the target position and the hand position at final instant $T$ can reflect the error at each time instant on the whole trajectory because the term $E_{jerk}$ in definition (5) is the sum of all instantaneous errors. Fig. 5 shows the characteristics of movements obtained with the robot arm, for $K = 6, 8, 10, 12, 20$. Clearly, the hand trajectories and the velocity curves are not realistic for $K = 6$ and 8. However, as $K$ increases, the trajectory becomes straighter and the velocity curve becomes smoother and more regular. For $K = 12$ the movement starts being very realistic. The improvement obtained by adding a new primitive is very small. For $K = 20$ the velocity profiles are perfectly bell-shaped and the hand trajectories are almost straight with a continuous weak curvature. Fig. 6 shows HRP-2 performing the motion corresponding to $K = 20$. For the sake of safety, we slowed down the motion to 2 s instead of the original 1 s. The human and humanoid experimental trials can be seen in the video submitted with this paper.

Compared to our preceding result, where reaching trajectories were obtained by solving a complex optimization problem defined by Guigon’s model [14], the computation time reduced by a factor of 6, i.e. from $2min$ to $20s$. However it is important to note that the method proposed in [14], was encoded in C whereas, so far, the current method is implemented in Matlab. Therefore, we can claim that the computation time can be further reduced by at least a factor 5, by using the C programming language for encoding the current method. In this condition, less that 5s would be necessary to generate a new realistic reaching movement.

### V. CONCLUSION

This result shows that the use of movement primitives coupled with an appropriate generalization process provides an efficient way to store the complex information of human movement, in order to quickly generate new trajectories. Although it has been suggested that the CNS plans movements by solving complex optimization problems - such as minimizing the energy of motoneurons - it is likely that the use of a reduced number of canonical variables allows to simplify the computation. The application of this kind of approach to humanoid robotics is interesting as it offers an alternative to the computation of inverse kinematics with additional cost minimization. In future works, we plan to address the problem of combining such open-loop primitives to produce whole body movements.

### VI. ACKNOWLEDGMENTS

This work was supported by the Zeuxis project funded by the EADS foundation and by the ROMEO project funded by the French FUI.

### REFERENCES