# A Method for Digital Representation of Human Movements

Vittorio Lippi, Carlo Alberto Avizzano, Emanuele Ruffaldi

***Abstract*— In this work we present a method to produce a model of human motion based on an expansion in functions series. The model is thought to reproduce the learned movements generalizing them to different conditions. We will show, with an example, how the proposed method is capable to produce the model from a reduced set of examples preserving the relevant features of the demonstrations while guaranteeing constraints at boundaries.**

## I. INTRODUCTION

### A. Aim of the work

In this work we present a framework to model human trajectories. Some general ideas have been presented in [1], in this paper we describe in details the steps to apply the method. The key idea is to produce a model that satisfies dynamic constraints of the performed task while maintaining the likelihood to the provided samples. We assume that an internal, at least simplified, model for motion does exist, which has been proven for specific selected motions [10]. We will describe the algorithms applied to obtain the behavior model from observed data and show how the model obtained can be used in real time to interact with a simulated environment.

Several system to model human motion have been designed in literature. As an early example we can cite [7] were the purpose was to design multimodal environments designed for the transfer of human abilities. Since then several, more advanced systems, have been developed e.g. Avizzano [9], Henmi [11], Esen [12] among others. Learning is often performed, to achieve better performances in a 'latent' feature space by most common learning models such as Switching Linear Dynamic Models (SLDM) [13], Gaussian Mixture Regression (GMR) [14], Dynamic Motion Primitives (DMP) [3], Local Weighted Process Regressions (LWPR) [19], Gaussian Processes [16] [17], Stable Estimator Dynamical Systems (SEDS) [18] and Sequenced Linear Dynamical Systems [8].

### B. Design Specifications

The described method is designed to meet the following requirements:

- stability;
- robustness to outliers;
- generalization: ensure trajectory generation that can met with not-shown boundary conditions;
- adaptation: replan ongoing trajectories while maintaining continuity and minimal distortion from examples;
- error tolerance;

The authors are with PERCRO laboratory, TECIP institute, Scuola Superiore Sant'Anna, Pisa. v.lippi@sssup.it

We are interested in solving the problem of motor programming through the creation of a mathematical model of human behavior. This model should be based on the observation of several repetitions of a task. The observation should be allowed to consist in different kinds of movements that are classified as homogeneous by the user. We suppose that this is possible because movements, that are part of the behavior pattern characterizing the expression of a particular skill, show strong similarities as shown in several works such as [5] and [22].

In this work we present a model and a methodology to learn motion patterns from a given (reduced) number of repeated examples and learn the relationships among the variation in the boundary conditions and motion patterns.

The methods we addressed in the previous section are based on an implicit description of the trajectory performed, i.e. a set of differential equations. In this method we use an explicit expression of the trajectory. This implies the stability of the trajectory.

## II. METHOD EXPOSITION

### A. Problem description

We identify a task with the trajectories produced by a human while performing it. The trajectories take place in a space of variables considered relevant for the task itself, e.g. for a reaching task the position of the performer hand in the space could be considered. We assume that the motion pattern that we are going to describe can be segmented in several phases. This method is based on the production of a model for each phase, built on the basis of sample trajectories $Y_i$, a vector function of time. The samples $Y_i$ should be obtained recording the user movements. The model consists in a mapping between a vector of contour conditions $Q$ and a performed trajectory $Y$. The contour conditions consist in the value of some relevant variables that can be different from the ones in $Y$. The model for a phase can hence be represented as the application:

$$Y(t) = \Psi(Q, t) \tag{1}$$

### B. method overview

The performed task can be described as a trajectory, function of time and some contour conditions (e.g. in a reaching task these could be represented by the initial hand positions and the target position). The phases may repeat through the development of the task and hence different segments can be instances of the same phase.

We base our model on a function expansion of trajectories. Function expansion series have proven to be effective in previous human-motion approximation approaches. Traditional

TABLE I
NOTATION

| | |
|---|---|
| $Y_i$ | A motion trajectory (a function of time) |
| $\mathbf{\Phi}$ | The expansion function set |
| $\mathbf{P}_i$ | The function expansion coefficients (a vector) |
| $\mathbf{Q}_i$ | The contour conditions |
| $t$ | time |
| $T_i$ | trajectory duration |
| $\tau$ | Normalized time ($t_i/T_i$) |
| $\mathbf{F}$ | Matrix representing the regression hyperplanes |

polynomial [23] or orthogonal polynomial [25] expansions were applied in several minimum theories related to human arm motions. Several types of orthogonal function are employed, for instance Biess [24] proposed an expansion based on Jacobi polynomials and Fourier series.

We proceed transforming each segment into a set of parameters (the coefficients of the function expansion) then we perform a linear regression between the contour conditions and the parameters. Each variable describing the trajectory is treated separately. The coupling between different variables can consist in the fact that one variable's value can represent the contour condition for another variable.

### C. Parameter Fit

In order to explain the method in details we introduce some important quantities, listed in table II-C

We assume that the each phase is processed separately, hence the index $i$ addresses the $i^{th}$ example of a given phase. The time $t$ is 0 at the beginning of each example. The coefficients vector $P_i$ is not computed directly on $Y_i$ but on a warped version of the trajectory $\tilde{Y}_i = Y_i(t/T_i) = Y_i(\tau)$. In detail the vector $\mathbf{\Phi}$ has the following form:

$$\mathbf{\Phi}(\tau) = \begin{bmatrix} 1 \\ \tau \\ sin(\pi\tau) \\ sin(2\pi\tau) \\ sin(3\pi\tau) \\ \cdots \\ sin(N\pi\tau) \end{bmatrix} \quad (2)$$

The first two components of $\mathbf{\Phi}$ represent a linear translation. The corresponding parameters can be set directly to $Y(0)$ and $Y(1) - Y(0)$. The other parameters represent a description of the function:

$$D(\tau) = \tilde{Y}(\tau) - (Y(0) + \tau(Y(1) - Y(0))) \quad (3)$$

The number of components $N$ can vary to match the requirements of precision and compactness of the representation. It is important to note that, although $\tau$ ranges from 0 to 1 during the development of the example, the period of the first sinus (third component of $\mathbf{\Phi}$ is 2. This is thought to allow the trajectory represented by $P_i$ to have a different slope at the beginning and at the end as shown in figure 1. In particular, for each demonstration, we compute the coefficients $P_i$ of the extended trajectory

$$p(\tau) = \begin{cases} D(\tau) & : 0 \leq \tau \leq 1 \\ -D(1-\tau) & : 1 < \tau \leq 2 \end{cases} \quad (4)$$

so that $P_i = argmin_{P_i} |P_i\phi(\tau) - p(\tau)|$. The first two components of $P_i$ are computed explicitly, while the second one can be computed performing an FFT. The symmetry of $p(\tau)$ assures that just the coefficients associated to the terms in 2 are different from zero.

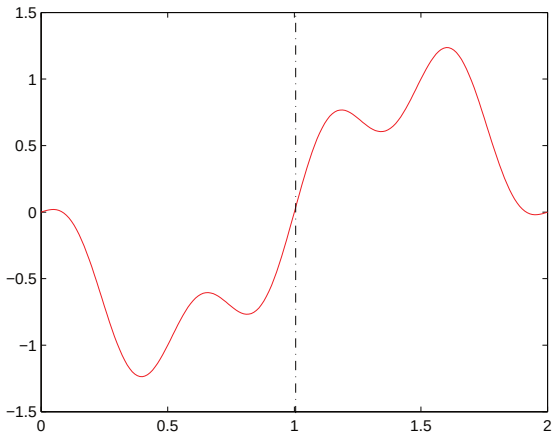


Fig. 1. A trajectory of duration 1 extended to a period of 2. This allow the trajectory to have a different slope for $\tau = 0$ and $\tau = 1$

A minimum square error linear regression between the vectors $Q_i$ and each component of $P_i$ is then performed. The vector $Q_i$ is augmented with a 1 so that a linear application can map an affine function from $Q$ to $Y$. The minimized error takes the form:

$$SQE = \sum_{i=1}^{n} (P_i - F; Q_i)(P_i^T - Q_i^T; F^T) \quad (5)$$

where $F$ is the solution produced by the regression:

$$F = [P_1 P_2 \cdots P_n] \begin{bmatrix} Q_1 & Q_2 & \cdots & Q_n \\ 1 & 1 & \cdots 1 & \end{bmatrix}^{\dagger} \quad (6)$$

where the indexes from 1 to $n$ address one of the $n$ trajectories provided as examples. Notice that in general $n >> N$, i.e. the examples are more than the components of the vector $\phi$. the matrix built with the $Q_i$ is hence expected to have full rank (equal to the number of components in $Q$). The pseudoinverse in 6 produces an hyperplane Two sample hyperplanes which were obtained by the regression performed in the example in section III-A are shown in figure 2.

### D. The Training Algorithm

The algorithm performing the parameters adaptation can be summarized into the following steps:

- Choose the variables in $Y$ and $Q$, characterizing the task that is going to be modeled;
- Define segmentation criteria for the sampled data. It is important to consider that, in order to use the obtained

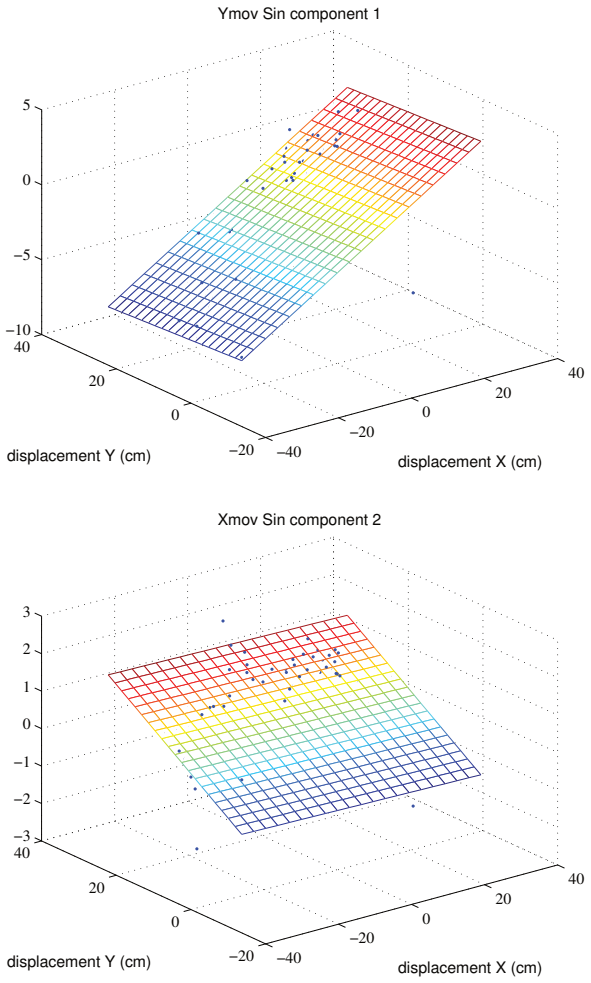


Fig. 2. two motion interpolation planes. the task is performed in a two-dimensional space, where the user moves the hand vertically and horizontally. The 2 graphs represent a component of $P_i$ determining the horizontal motion (top) and a component of $P_i$ determining the vertical motion (bottom) as function of $Q_i$, that, in this case, consists into the horizontal and vertical displacement between the positions at the begin and at the end of the trajectory. Blue dots represents components of the parameter vector $P_i$ computed on the basis of a single example

model to generate a behavior, the segmentation criteria should be based on data available online. Segmentation can be performed on the basis of the variables defining the trajectory (e.g. a position is reached) or some functions of them (e.g. the speed) or some relative to the environment (e.g. catching a given object);

- segment data obtaining several chunks, representing samples for the task's phases;
- scale each chunk in time so that it is warped in a interval of time of unitary duration. A number of samples to represent it is defined. This emphasizes the shape of the performed trajectory. We preferred an uniform scaling (formally expressed by the parameter $T_i$ in the previous formulas) over other more complex time warping systems because it is easily applied in real time: we rely on dataset segmentation to obtain chunks of data representing trajectories with the same shape;
- for each sample chunk $Y_i$ compute the sample parameters $P_i$ vector. Our particular choice of $\Phi$ allows us to compute the first two components of P analytically on the basis of the initial and the final position of the trajectory and then to obtain the series expansion of the residuals representing the rest of the parameters.
- Perform a linear regression to obtain $F$, a mapping from $Q$ to $P$.

### E. Generating the Trajectories

After the training phase the model is defined by the vectors $F$. To generate the trajectories we compute the parameter vector at the beginning of each phase:

$$P = F \begin{bmatrix} Q \\ 1 \end{bmatrix} \tag{7}$$

then, during the phase $Y$ is computed as:

$$Y(t) = P\Phi(t/T) \tag{8}$$

The first two components of $P$ are set directly to match the initial and the final value of $Y$. The trajectory is then performed until a segmentation criterion is met and the phase is considered ended. It is important to point out that, in order to generate the behavior $Q_i$ and $T_i$ (or directly $\tau$) should be available in real time. The parameter $T$ should be computed on the basis of the task and the state of the environment.

The stability of the trajectory is guaranteed by the closed form in which it is expressed.

The computation of the trajectory is very fast, consisting basically in two matrix product to compute respectively $P$ and $Y$. The system is hence suitable to be used in real time. An example of generalized trajectories, together with the sample data is shown in figure 3.

### F. Imposing Constraints

It is possible to impose a constraint on the derivative of the state variables at the end of the phase, the final velocity $v_{constrained}$. We performed this by first computing the parameters for the reaching task then correcting it, and hence, in the general case, obtaining a different final position. Given the parameters vector $P$ representing the trajectory without the constraint we have a final velocity $v_{free} = \frac{d\Phi^T}{d\tau}P$ at time $T$ the correction to $\Delta P$ to $P$ can be expressed as:

$$\Delta P = F \left[ \frac{d\Phi^T}{d\tau} F \right]^{\dagger} [v_{constrained} - v_{free}] \tag{9}$$

This formulation finds the closest point lying on the regressed hyperplane (equation 7) verifying the constraint.

The convenience of this choice can be shown through an example. In figure 4 we show the difference in the trajectory produced by the trajectory correction explained in the equation 9 and the one produced relaxing the constraint of lying on the hyperplane: although closer in the space of parameters to the reaching task the latter solution produces an unnatural movement, in contrast to the former that preserves

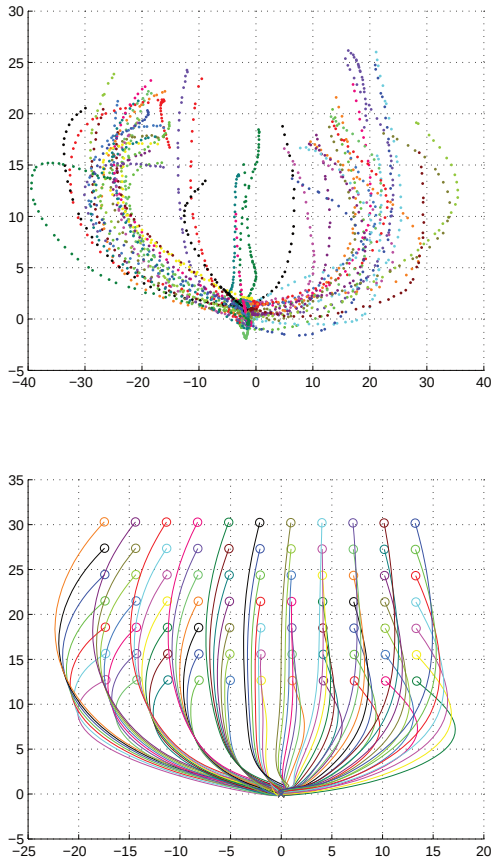

Fig. 3. Sample movement for a dynamic reaching task in virtual reality i.e. catching a falling ball (top) and the generalization obtained through the proposed modeling. $Q_i$ consists into the displacement between the position at the beginning and at the end of the trajectory. Trajectories are plotted with different colors to improve readability

the characteristic shape of the gesture. Moreover this example shows that the regressed hyperplane is highly descriptive about the nature of the movement. The example has been produced with data from the task described in the section III-A

### G. Trajectory Correction

As stated in the introduction we are interested into the development of an adaptable system. With adaptation we mean the ability to change and replan the motion strategies in consequence of the results of the actions on the environment or in consequence of external changes coming from the environment. This capability is typically captured in motion control system through the use of closed loop controllers that monitor a tracking error in order to produce correlated changes in the motion. For example suppose, at relative normalized time $\tau = 0$, to have determined a motion vector $P_i$ associated to given conditions $Q_i$ on the begin and the end of the trajectory. We consider two cases in which the trajectory should be corrected:

- replan due to motion control errors;
- adaptation due to target changes.

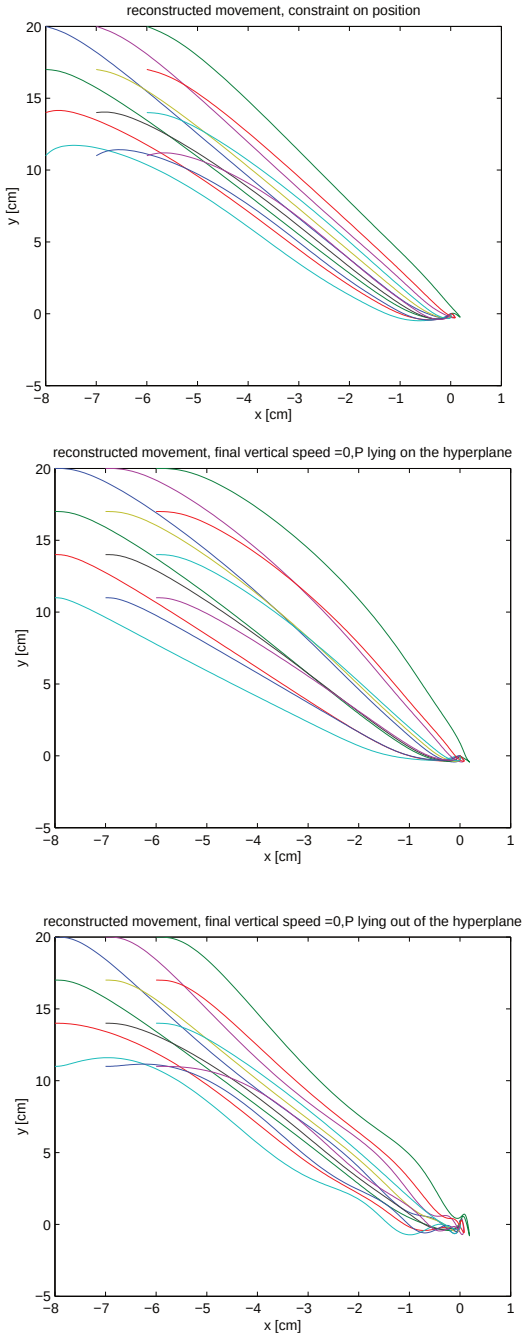


Fig. 4. Reconstructed trajectories (top) with the constraint of reaching a given position and with a forced null vertical speed at the end of the trajectory with the constraint that $P$ should belong tho the regressed hyperplane (middle) and without this constraint, on the basis of minimum square error(bottom).

In the first case, we assume that, during the execution of a motion, a displacement among desired and effective position is detected; in the latter, we assume that the final point has changed during the execution of the motion. In both cases we request that any replan does not change the past history

of the motion, while it intervenes on the remaining (future) trajectory to compensate for the variations. At time $\tau_1$ the variable is found the state to be $\tilde{Y}_{\tau_1}^{error}$, that can be different from $\tilde{Y}(\tau_1)$, while the desired final state is $\tilde{Y}_f$, that can be different from $\tilde{Y}(1)$. The correction in the parameter vector $P$, as following from the definition of $P$ in the equation 7, is:

$$\Delta P^T = [\Delta\tilde{Y}_i(\tau_1)\ \Delta\tilde{Y}_f]\,[\Phi(\tau_1)\ \Phi(1)]^\dagger \qquad (10)$$

where the $\Delta$ indicates a deviation of the variable from the one associated to the original trajectory. Also in this case, the correction forces $P$ to lie on the hyperplane defined by the equation 7.

### H. The Importance of Segmentation

It is important to address the importance of the segmentation in this system. Computing an unique $P$ for all the process we lose several desiderable characteristics:

- the possibility to define different scaling in time for different part of the process;
- the possibility to manage the transition between phases with a state machine, allowing complex interaction with the environment and the possibility to change the order in which the phases are presented;
- The possibility to specify different constraints for different situations occouring during the task.

It is important to define in advance a segmentation criterion that is possible to apply in real time. This can be defined explicitly on the basis of the knowledge of the task and of the contour conditions implied by the breakpoints chosen. Segmentation in a complex task involving a large number of variables can be obtained using machine learning techniques such as neural networks to model the transition between phases as presented in [20]. This would work, again, in case the user provides explicitly a segmentation of the sample to be used to train the neural networks. An optimized segmentation for this method is still object of research.

## III. Experimental Results

### A. Test Task

In order to test the system we set up a simplified version of ball juggling in virtual reality inspired by [4], involving the catch and the toss of one ball with the right hand. The examples have been produced recording the vertical and horizontal component of the hand position of a user interacting with the virtual environment. We used a Polhemus electromagnetic tracker sampled at 200 Hz. The set up of the system is shown in figure 5 The user was asked to perform the following exercise:

- toss a ball towards the left portion of the screen;
- reach a given point at the bottom of the screen with the hand;
- catch the ball
- return to the starting point and start again

The task takes place totally in virtual reality. The interaction is based only on hand tracking: the toss is triggered when vertical hand acceleration reaches a given negative threshold

Fig. 5. The system used during the experiment.The user stands in front of a monitor where a a virtual hand and a virtual ball are shown. The user controls the position of the virtual hand moving a Polhemus® tracker.

and the ball detaches with the speed of the hand at the end of the toss. This requirement to perform a trajectory starting with an acceleration an ending in a deceleration fast enough to hit the threshold when the speed is the desired one for the toss. A catch is triggered when the virtual hand touches the virtual ball and hence the catch can be considered a dynamic reaching task. We segmented the task in four phases: *tossing*, *reaching the bottom of the screen*, *catch*, and *return*. The phase transition has been based on the toss for the tossing phase, the position reached for the going-down and the return phases, and the successful ball catch for the catch phase. In this case, the test can not be reduced to a two points reaching problem [10] since in this case complex start/end point boundary values alter the dynamics of the gesture.

In this example the conditions $Q_i$ determining the behavior are the initial and the final position of the hand. In the case of the *catch* the final position should be computed to assure that the hand reaches the ball.

We used random targets based on the distribution of user movements for the reaching tasks (going-down and return), the catch phase was driven setting as constraint to reach the ball. The $T_i$ times were generated on the basis of a uniform distribution covering the range of times observed during the training. The toss phase was driven in the same way, but applying later a condition of vertical speed equal to zero (as explained in II-F) this was performed by the user to invert the motion.

### B. Data Regression

To regress the parameters of the model we proceed actually performing two regressions. First we regress a vector $P$ from every trajectory associated to a phase, computing the coefficients (i.e. the first two) representing the linear translation analytically to achieve the desired position at the beginning and at the end of the trajectory, we compute the expansion of the residuals minimizing the square error. Finally we regress the Hyperplanes using the obtained $P$ vectors as samples representing the relation between the

conditions $Q_i$ and the movement $Y$. Since the harmonic decomposition of the residual can be made arbitrary precise increasing the number of components $N$ we concentrate then in an estimation of the goodness of the linear regression. We check, at least within the context of the presented case study, the hypothesis that a linear model can describe the relation between a human movement and its boundary conditions.

### C. Performace

To train the system described in the previous section we produced dataset for each phase composed from 40 to 60 different examples containing each from 150 to 200 different sample points. Assuming that the deviation from the plane can be represented with a Gaussian noise, a linear regression analysis was carried out on the full dataset. A *Lilliefor* Normality test on the regression residuals confirmed the hypothesis of 'Gaussian Like'.

The trained system is capable of performing the exercise steadily: this requires to perform reaching tasks between different points keeping the trajectory shape and to reproduce the trajectory dynamics producing the tosses. Besides reproducing the task we are interest into doing in a style that is similar to the oneperformed by the human. To asses this an analysis of the 95% confidence interval on regression parameters was performed by partitioning samples in different test and to assess the relevance of the parameters and of the model hyperplanes. In the case of relevant parameters, we found a very high stability of the F-test conducted on regression results that confirmed the accuracy of estimation. A jacknife method applied as cross validation check shown that, the variance in the estimated planes parameters was one or two order of magnitude less than the estimated values.

## IV. Conclusions

We presented a programming by demonstration framework. We based our method on the construction of a model for a complex task. We described how the system can give a great flexibility in modeling motor tasks involving the interaction with the environment and we shown an example of how the system can work. We have described a trajectory correction procedure that can be used explained how the model can compensate tracking error and react to target changes. Experimental statistical analyses have demonstrated the effectiveness of the approach.

### Future Work

The modeling of human movement through linear functions between a task phase contour conditions and trajectory parameters should be formally tested on a wider set of cases on different tasks and sampling different users). Even if the decomposition in translation and harmonics is general enough to reproduce human movements, we are planning to test different $\Phi$ vectors, mainly to explore the descriptive properties of the vector $P$ about the studied movement. Another interesting improvement of the described system would consist in the design of a method to obtain segmentation (phase transition) rules from the examples.

## V. Acknowledgments

The authors are grateful the EU FP6 and the whole SKILLS consortium for having supported this research. EU partially funded the research activities within the IP-SKILLS research project and the consortium provided us with valuable test sets and feedback to refine the method herein presented.

## References

[1] Carlo Alberto Avizzano, Vittorio Lippi, A regression model for Digital Representation of Juggling, In *Proc. of SKILLS conference*, In printing (2011). DOI: 10.1051/bioconf/20110100006

[2] C.G. Atkenson 1990, Using Local Models to Control Movement, In *Neural Information Processing Systems* v. 2, pp.316-323, ISBN-1-55860-100-7(1990).

[3] J.A. Ijspert, J. Nakanishi, S. Schaal, Learning Rythmic Movements by Demonstration Using Nonlinear Oscillators, In *Proc of. Int. Conf. on Robotics and Automation*, (2000).

[4] Lagarde, J., Zelic, G., Avizzano, C. A., Lippi, V., Ruffaldi, E., Zalmanov, H., Erev-Yehene, V., Gillian, N., OModrain, S., Mottet, D. Gopher, D. The Light Weight juggling training system for juggling International Conference on Multimodal Interfaces for Skills Transfer, 2009

[5] A. Filippeschi E. Ruffaldi, Expert rowers motion analysis for synthesis and technique digitalization, In *Proc. of SKILLS conference*, In printing (2011).

[6] J. Abbot, A. Okamura, Pseudo-admittance bilateral telemanipulation with guidance virtual fixtures, In *International Journal of Robotics Research* vol. 26:865-71. (2007).

[7] Y. Yokokohji, R. Hollis, T. Kanade, K. Henmi, T. Yoshikawa, Toward machine mediated training of motor skills. Skill transfer from human to human via virtual environment. In *Proceedings of IEEE International Workshop on Robot and Human Communication Proceedings* pp. 32-37, (1996).

[8] Kevin R. Dixon and Pradeep K. Khosla,Trajectory Representation Using Sequenced Linear Dynamical Systems in Proceedings of the IEEE International Conference on Robotics and Automation (2004)

[9] C.A. Avizzano, J. Solis, A. Frisoli, M. Bergamasco, Motor Learning Skills Experiments using Haptic Interface Capabilities, In *Proc. of 11th IEEE International Workshop on Robot and Human Interactive Communication* (2002).

[10] T. Flash, N. Hogan, Coordination of Arm Movements: an experimentally confirmed mathematical model, *Neurology* (1985).

[11] K. Henmi, T. Yoshikawa, Virtual lesson and its application to virtual calligraphy system. In *Proc. of IEEE International Conference on Robotics and Automation*, pp. 1275-1280 (1998).

[12] H. Esen, K. Yano, M. Buss, Force skill training with hybrid trainer model. In *Proc. of IEEE International Workshop on Robot and Human Interactive Communication*, pp. 9-14, ISBN: 978-1-4244-2212-8 (2008).

[13] V. Pavlovic, J.M. Rehg, J. MacCormick, Learning Switching Linear Models of Human Motion, *Advances in Neural Information Processing Systems*, pp.981-987, issn=1049-5258 (2001).

[14] S. Calinon, F. Guenter, A. Billard, On learning, representing and generalizing a task in a humanoid robot, In *IEEE Transactions on Systems, Man and Cybernetics*, vol. 37:286-98 (2006).

[15] M. Da Silva, Y. Abe, J. Popovic, Simulation of Human Motion Data Using short-horizon Model Predictive Control, In *Proc. Eurographics* (2008).

[16] J.M. Wang, D.J. Fleet, Gaussian Process Dynamical Models for Human Motion, In *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 30:2 (2008).

[17] Y. Ye, K. Liu, Synthesis of responsive motion using a dynamic model. In *Proc. of Eurographics* (2010).

[18] S.M. Khansari-Zadeh, A. Billard, Imitation Learning of Globally Stable Non-Linear Point-to-point Robot Motions using Nonlinear Programming, In *Proc. of IEEE/RJS Int. Conference on Intelligent Robots and Systems*, ISBN-978-1-4244-6676-4, pp 2676-2682 (2010).

[19] S. Vijayakumar, A. Souza, S. Schaal, Incremental online learning in High dimensions, Neural Computation 17(12), pp. 2602-2634, (2005).

[20] Otniel Portillo-Rodriguez and Oscar O. Sandoval-Gonzalez and Emanuele Ruffaldi and Rosario Leonardi and Carlo Alberto Avizzano and Massimo Bergamasco. Real-Time Gesture Recognition, Evaluation and Feed-Forward Correction of a Multimodal Tai-Chi Platform. Haptic and Audio Interaction Design, Third International Workshop, HAID 2008, Jyvskyl, Finland, September 15-16, 2008, Proceedings.

[21] P. Kormushev, S. Calinon, D. Caldwell, Robot Motor Skill Coordination with EM-based Reinforcement Learning, In *IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, ISBN: 978-1-4244-6676-4 (2010).

[22] M. Vasilescu, Human motion signatures: Analysis, synthesis, recognition. In *Proceedings of International Conference on Pattern Recognition* 3:456-60, ISBN: 0-7695-1695-X, (2002).

[23] S.E. Engelbrecht, J.P. Fernandez,Invariant characteristics of horizontal-plane minimum-torque-change movements with one mechanical degree of freedom, In *Biological cybernetics*, vol 76:5 pp. 321–329 (1997).

[24] A. Biess, M. Nagurka, T. Flash, Simulating discrete and rhythmic multi-joint human arm movements by optimization of nonlinear performance indices, In *Biological cybernetics*, vol 95 pp. 31–53 (2006).

[25] G.E. Forsyth, Generation and the use of orthogonal polynomials for data fitting with a digital computer, in *Jour. Soc. Indust. Appl. Math.* vol 5, pp. 74-88 (1957).