In Proc. IEEE Int. Conf. Robotics and Automation, San Francisco, California, pp. 2223-2228, April 2000.

Planning of Joint Trajectories for Humanoid Robots Using B-Spline Wavelets

Aleš Ude[1], Christopher G. Atkeson[2,3], Marcia Riley[2,3]

[1]ERATO Kawato Dynamic Brain Project, Japan Science and Technology Corporation
2-2 Hikaridai Seika-cho, Soraku-gun, Kyoto, Japan
[2]College of Computing, Georgia Tech, 801 Atlantic Drive, GA 30332-0280, USA
[3]ATR Human Information Processing Research Laboratories
2-2 Hikaridai Seika-cho, Soraku-gun, Kyoto, Japan

Abstract

The formulation and optimization of joint trajectories for humanoid robots is quite different from this same task for standard robots because of the complexity of the humanoid robots' kinematics. In this paper we exploit the similarity between the movements of a humanoid robot and human movements to generate joint trajectories for such robots. In particular, we show how to transform human motion information captured by an optical tracking device into a high dimensional trajectory of a humanoid robot. We utilize B-spline wavelets to efficiently represent the joint trajectories and to automatically select the density of the basis functions on the time axis. We applied our method to the task of teaching a humanoid robot how to make a dance movement.

1 Introduction

Movements of most of the current robot manipulators can be described by a single open kinematic chain. A standard approach to the specification of movement tasks for such robots is to define a trajectory for the motion of a robot tip. Further constraints are necessary in the case of redundant mechanisms. The specification of a movement task involving the full-body motion of a humanoid robot (Fig. 1) is much more complicated because the underlying kinematic structure of the humanoid robot is more complex. Indeed, our humanoid robot possesses 5 tips: the head, the two hands and the two feet. The motions of its tips are not independent, but are coupled through the robot linkage. Therefore we cannot directly specify a full-body motion by independently specifying the trajectories of these tips. In addition, taken as a whole the robot kinematic structure is redundant.

Our humanoid robot DB consists of 15 rigid parts divided into 6 interconnected kinematic chains: the head; the upper arms, lower arms, and hands; the lower and the upper torso; and the upper legs, lower legs and feet. The trajectories of joints connecting these body parts define the complete motion of the robot as the robot pelvis is fixed in space. DB has 26 degrees of freedom (plus four degrees of freedom for the eyes' movements which are not considered in this paper). The dependencies between trajectories of different body parts are defined by DB's geometric structure. It is thus more appropriate to represent full-body motion trajectories in terms of independent variables, e. g. joint angles, because joint space trajectories automatically conform to the geometric constraints.

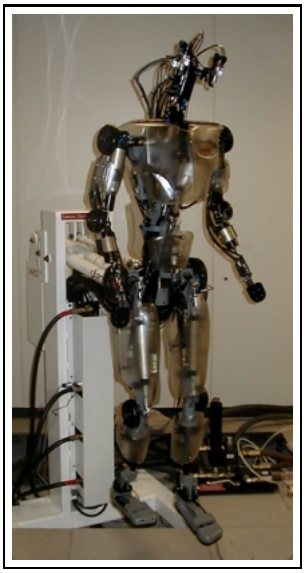

Figure 1: Humanoid robot called DB in our laboratory

There is not much hope of finding a closed form trajectory for full-body motions such as dancing. Therefore we exploit the similarities between humanoid robots and humans to generate appropriate trajectories. We are also taking advantage of the nature of the tasks humanoid robots are asked to perform, which typically involve making human-like motion. We started our investigation with motion capture techniques from the entertainment industry and computer graphics [1], and also borrowed tech-

niques from robot teleoperation, robot programming by direct teaching or showing [5], and virtual reality. We found that the requirements to actually control a physical device required modifications of techniques from the virtual and entertainment applications, and that we could take advantage of the offline nature of our task to more effectively process the teaching data and handle less cumbersome motion capture devices. In this paper we capture full body motions of a human performer using an optical tracking device, which provides the 3D location of identified active markers which are currently in view. We have also experimented with goniometer devices strapped to the performer, and magnetic systems that provide marker orientation as well as location. The techniques presented in this paper can be easily extended to all of these different types of motion capture systems. We are also currently extending our techniques to vision systems where there are no markers, but individual pixels must be matched or accounted for [10].

To relate human motion to robot motion, we model the kinematics of the performer using a similar kinematic model as the robot's, but scaled to the performer. The performer's kinematic model should contain at least those degrees of freedom which are relevant for the desired movement task. Sometimes it is advantageous to augment the performer's kinematic model by additional degrees of freedom to facilitate the measurement process. This approach can of course work only for a humanoid robot having a kinematic structure similar to a human.

2 Setting up the criterion function

The placement of a human body in Cartesian space is determined by the position and orientation of a global body coordinate system rigidly attached to one of the body parts and by the values of the joint angles. We use twist coordinates [6] to model the performer's and DB's kinematics. If the coordinates of a marker in a local coordinate system of a rigid body part to which it is attached are given by $\boldsymbol{y}_j$, then its 3-D position at body configuration $(\boldsymbol{r}, \boldsymbol{d}, \theta_1, \ldots, \theta_n)$ can be calculated as follows

$$\begin{aligned} \tilde{\boldsymbol{y}}_j &= \boldsymbol{g}(\boldsymbol{r}, \boldsymbol{d}) \cdot \exp(\theta_{i_1} \boldsymbol{\xi}_{i_1}) \cdot \ldots \cdot \exp(\theta_{i_{n_j}} \boldsymbol{\xi}_{i_{n_j}}) \cdot \boldsymbol{G}_j \cdot \boldsymbol{y}_j \\ &= \boldsymbol{h}_j(\boldsymbol{r}, \boldsymbol{d}, \theta_1, \ldots, \theta_n). \end{aligned} \quad (1)$$

Here $\exp$ is the function mapping twists $\boldsymbol{\xi}_i$ representing the body kinematics into rigid body transformations, $\boldsymbol{G}_j$ is the homogeneous matrix combining the position and orientation of the local body part coordinate system to which the marker is attached with respect to the global body coordinate system at zero configuration, $\boldsymbol{r}$ and $\boldsymbol{d}$ are the orientation [1] and position of a global body coordinate system with respect to the world coordinate system, and $\boldsymbol{g}(\boldsymbol{r}, \boldsymbol{d})$ denotes the homogeneous matrix corresponding to $\boldsymbol{r}$ and $\boldsymbol{d}$. Note that the set of twists affecting the motion of a body point varies according to the identity of the body part to which the marker is attached.

[1] We use rotation vectors to represent the orientation.

Our trajectory planning method should generate motions which are perceptually similar to the motion of the performer. This can be achieved by recovering joint angles that results in marker positions close to the measurements. Thus we should favor configurations minimizing

$$\sum_{j=1}^{N} \|\boldsymbol{h}_j(\boldsymbol{r}(t_k), \boldsymbol{d}(t_k), \theta_1(t_k), \ldots, \theta_n(t_k)) - \boldsymbol{y}_j^*(t_k)\|^2, \quad (2)$$

where $\boldsymbol{y}_j^*(t_k)$ denotes the measured markers at time t_k.

A straightforward approach to the generation of motion trajectories is to sequentially minimize criterion (2) at each measurement time. A continuous trajectory can be generated afterwards by interpolating the recovered joints. This approach has, however, several deficiencies. First of all, optical motion capture systems often cannot recover the positions of all markers due to occlusions. This can result in underconstrained linear systems causing the optimization process to break down. Moreover, experiments showed that even when the positions of all markers can be recovered, the optimization process can still break down because of the singularities in the kinematic model. Finally, just moving like the human is not the only criterion relevant for the robot motion.

The optimization process can be made more reliable by recovering the complete trajectory instead of single configurations and by the regularization of the objective function (2). Writing the full-body trajectory as $\boldsymbol{f}(t) = (\boldsymbol{r}(t), \boldsymbol{d}(t), \theta_1(t), \ldots, \theta_n(t))$, we look for a function that minimizes

$$s(\boldsymbol{f}) = \frac{1}{2} \sum_{k=1}^{M} \sum_{j=1}^{N} \|\boldsymbol{h}_j(\boldsymbol{f}(t_k)) - \boldsymbol{y}_j^*(t_k)\|^2, \quad (3)$$

over all possible trajectories. The regularization can be achieved by minimizing the amplitude of physical quantities such as acceleration (2nd derivative) or jerk (3rd derivative)

$$r(\boldsymbol{f}) = \frac{1}{2} \int_0^1 \|\boldsymbol{f}^{(m)}(t)\|^2 \mathrm{d}t, \; m = 2 \text{ or } 3. \quad (4)$$

We always normalize the time of our trajectory to 1. In general we could penalize any weighted combination of kinematic variables such as acceleration, jerk, and violation of joint space or Cartesian soft limits. Because we have the full trajectory available and can analytically compute velocities and accelerations (for spline based trajectory representations), we could use inverse dynamics to compute

torques or actuator commands from a physically feasible trajectory, and use the computed values to penalize any function of torque or actuator commands such as squared torque magnitude or actuator energy dissipation.

The trajectory planning problem thus becomes

$$\min_{\boldsymbol{f}} \{s(\boldsymbol{f}) + \lambda r(\boldsymbol{f})\}, \tag{5}$$

where λ is the parameter governing the tradeoff between the two objective functions. Apart from the above mentioned computational issues, it is advantageous to generate smooth trajectories also in order to reduce the wear and tear of the mechanical system, avoid exciting higher order dynamics, and because real actuators often have limits on their torque output or on the rate of change of output. Furthermore, jerky motions do not look natural which is distracting for movement tasks like dancing.

3 Trajectory Generation

Since parametric forms of complex body movements are normally not known, we applied the finite element method to represent the trajectory. In particular, we could use B-splines that were used before for the generation of joint trajectories of industrial robots [9] as basis functions. Unfortunately, an optimal set of fixed B-splines often cannot be determined in advance. If there are not enough basis functions, the generated motion may be far from the desired motion. If there are too many basis functions, the computational complexity is increased unnecessarily due to the larger number of variables as well as the resulting ill-conditioning of the linear subproblems that arise in the optimization process. A solution is to use a wavelet basis, in which the trajectory is represented hierarchically [4, 8], instead of a B-spline basis.

3.1 B-Spline Wavelets

Let $V^j(m)$ be the set of $2^j + m$ endpoint-interpolating B-splines of degree m constructed from knot sequence

$$\frac{1}{2^j}[\underbrace{0,\ldots,0}_{m+1 \text{ times}}, 1, 2, \ldots, 2^j - 2, 2^j - 1, \underbrace{2^j,\ldots,2^j}_{m+1 \text{ times}}]. \tag{6}$$

Linear spaces spanned by these splines are nested, i. e.

$$\mathcal{L}(V^0(m)) \subset \mathcal{L}(V^1(m)) \subset \mathcal{L}(V^2(m)) \subset \ldots, \tag{7}$$

where $\mathcal{L}(X)$ denotes the linear space spanned by members of X. Each orthogonal complement of $\mathcal{L}(V^j(m))$ in $\mathcal{L}(V^{j+1}(m))$ is called a wavelet space and its basis is denoted by $W^j(m)$. The members of $W^j(m)$ are

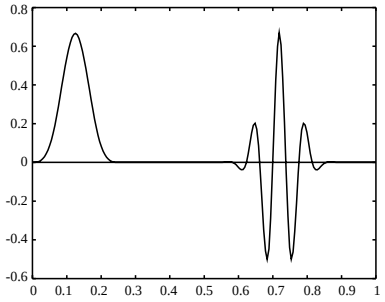

Figure 2: A typical B-spline and a typical wavelet

called semiorthogonal wavelets because they are orthogonal only to B-splines but not to each other. By definition $\mathcal{L}(W^j(m)) \subset \mathcal{L}(V^{j+1}(m))$, thus wavelets are piecewise polynomials of the same degree as the underlying B-splines. A desirable property for wavelets is to have small support. Let ϕ_i^j and ψ_i^j be members of $V^j(m)$ and $W^j(m)$, respectively. Since $\mathcal{L}(V^j(m))$ and $\mathcal{L}(W^j(m))$ are subsets of $\mathcal{L}(V^{j+1}(m))$, we can write

$$\phi_i^j = \sum_k p_{ik}^j \phi_k^{j+1}, \; \psi_i^j = \sum_k q_{ik}^j \phi_k^{j+1}. \tag{8}$$

A construction for semiorthogonal wavelets with the smallest number of consecutive nonzero q_{ik}^j is given in [8]. Note that the same property is true for B-splines, hence matrices $\boldsymbol{P}^j = \{p_{ik}^j\}$ and $\boldsymbol{Q}^j = \{q_{ik}^j\}$ are banded. We relate the reader to [8] for the explicit formulae for $\boldsymbol{Q}^j$ and $\boldsymbol{P}^j$. A typical B-spline and a typical wavelet are shown in Fig. 2.

The multiresolution finite element approach assumes that the optimal trajectory can be written as a linear combination of B-spline wavelets:

$$\boldsymbol{f} = \sum_i \boldsymbol{C}_i \phi_i^L + \sum_j \sum_i \boldsymbol{D}_i^j \psi_i^j. \tag{9}$$

Here B-splines ϕ_i^L are fixed at the lowest possible resolution L, while the optimal set of wavelets ψ_i^j, $L \le j \le K$, should be determined automatically by the optimization procedure. By definition B-spline wavelets ψ_i^j have unit norm. This forces their amplitude to become higher and higher to retain the unit norm as they become narrower and narrower. Since this is counterproductive for optimization [4], we replaced ψ_i^j with $2^{L-j}\psi_i^j$ to obtain proper scaling.

3.2 Large-Scale Optimization

By replacing $\boldsymbol{f}$ in the optimization problem (5) with the above linear combination of B-splines and wavelets, which are fixed in this section, we obtain a classic unconstrained optimization problem. Instead of minimizing over all functions from some function space, we can minimize over parameters $\boldsymbol{C}_i$, $\boldsymbol{D}_i^j$. Note, however, that the number of unknown parameters is very high. The full-body motion of

DB involves 26 joints plus 6 degrees of freedom for the displacement in space. If we fix the highest resolution space of piecewise polynomials to $V^7(3)$, there are altogether 67 basis functions. Thus in this case the highest possible number of variables is $67 * 32 = 2144$ which clearly makes our optimization problem large. The highest resolution used in our experiments was eight which results in over 4000 unknown variables.

Trust region methods are suitable for solving large-scale optimization problems [2]. Let $\boldsymbol{H}$ and $\boldsymbol{g}$ respectively be the Hessian and the gradient of the objective function (5) at the current estimate for unknown variables $\boldsymbol{x} = (\boldsymbol{C}_i, \boldsymbol{D}_i^j)$. The main idea of the trust region approach is to approximate the criterion function with a quadratic function in the neighborhood around the current estimate. The next approximation is thus computed by minimizing

$$\min_{\boldsymbol{x}} \left\{ \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{g} \text{ such that } \|\boldsymbol{S}\boldsymbol{x}\| \leq \Delta \right\}, \quad (10)$$

where $\boldsymbol{S}$ is a diagonal scaling matrix and Δ is a positive scalar setting the size of the neighborhood.

The trust region approach assumes that the gradient and the Hessian of the criterion function can be calculated. Let us define

$$\boldsymbol{c}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{h}_1\Big(\sum_i \boldsymbol{C}_i\phi_i^L(t_1) + \sum_j\sum_i \boldsymbol{D}_i^j\psi_i^j(t_1)\Big) - \boldsymbol{y}_1^*(t_1) \\ \vdots \\ \boldsymbol{h}_N\Big(\sum_i \boldsymbol{C}_i\phi_i^L(t_M) + \sum_j\sum_i \boldsymbol{D}_i^j\psi_i^j(t_M)\Big) - \boldsymbol{y}_N^*(t_M) \end{bmatrix}. \quad (11)$$

Comparing (11) with the objective function (3) we note that $s(\boldsymbol{f}) = \frac{1}{2}\|\boldsymbol{c}(\boldsymbol{x})\|^2$. Let $\boldsymbol{J}$ be the Jacobian of $\boldsymbol{c}$ at the current estimate. It is easy to verify that the gradient of s is equal to $\boldsymbol{J}^T\boldsymbol{c}(\boldsymbol{x})$ while the Hessian of s is given by $\boldsymbol{J}^T\boldsymbol{J}+$ second order terms. It is a common practise in nonlinear least squares problems to neglect the second order terms, which are expensive to calculate, and to approximate the Hessian by $\boldsymbol{J}^T\boldsymbol{J}$.

The calculation of the gradient and the Hessian of the criterion function (4) involves the calculation of inner products of derivatives of basis functions

$$\int_0^1 \phi_i^{(m)}\phi_j^{(m)}\mathrm{d}t,\ \int_0^1 \phi_i^{(m)}\psi_j^{(m)}\mathrm{d}t,\ \int_0^1 \psi_i^{(m)}\psi_j^{(m)}\mathrm{d}t.$$

Let $\boldsymbol{\Omega}$ be the matrix of these inner products. It is easy to see that (4) can be rewritten as $r(\boldsymbol{f}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Omega}\boldsymbol{x}$. Thus the gradient of the objective function r is given by $\boldsymbol{\Omega}\boldsymbol{x}$ and its Hessian is simply equal to $\boldsymbol{\Omega}$. Note that $\boldsymbol{\Omega}$ is independent of $\boldsymbol{x}$, thus it should be calculated only once at the beginning of the optimization process. We employed the Gaussian quadrature formulae, which are exact for polynomials up to a certain degree, to evaluate these integrals exactly.

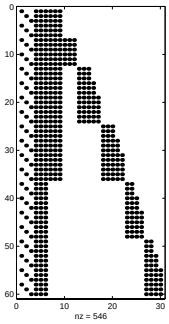

Figure 3: Sparsity pattern of the kinematics Jacobian

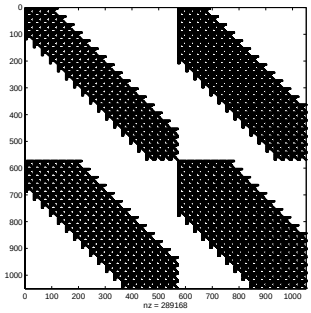

Figure 4: Sparsity pattern of the Hessian $\boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{\Omega}$

The row dimension of $\boldsymbol{J}$ is even larger than its column dimension, hence its calculation should be designed with care. The bulk of the computing time for the calculation of $\boldsymbol{J}$ is spent on the calculation of the Jacobian of the kinematic model for the human body motion at all measurement times. Due to the structure of our model, the kinematics Jacobian is sparse (see Fig. 3). Moreover, due to the minimal support property of B-splines and wavelets, $\boldsymbol{J}$, $\boldsymbol{J}^T\boldsymbol{J}$, $\boldsymbol{\Omega}$ and the resulting combined Hessian $\boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{\Omega}$ are also sparse (see Fig. 4). In the example in Fig. 4, all B-splines and all wavelets at a particular resolution level were included in the calculation. The sparsity pattern is a bit more complicated when only some of the wavelets from different resolution levels are used (each resolution level contributes one large block in the Hessian), but the resulting matrices are nevertheless sparse.

We showed that the gradient and the Hessian of the combined criterion function (5) can be estimated as

$$\boldsymbol{g} = \boldsymbol{J}^T\boldsymbol{c}(\boldsymbol{x}) + \lambda\boldsymbol{\Omega}\boldsymbol{x},\ \boldsymbol{H} \approx \boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{\Omega}. \quad (12)$$

Thus the next estimate for our trajectory can be computed by solving the trust region subproblem (10). However, standard algorithms for solving (10) are not appropriate for large-scale problems because they require the computation of a full eigensystem [2]. This is resolved in large-scale optimization approaches by restricting the solution space to a two-dimensional subspace spanned by the direction of the gradient and the direction of the negative curvature. The calculation of the next approximation is then trivial. Finally, the trust radius Δ is adjusted to a new value. We were able to use the MATLAB Optimization Toolbox implementation of a trust region method for large scale optimization problems and the sparse matrix capabilities of MATLAB for the resolution of sparse linear systems. Therefore we relate the reader to [2] and the references therein for further information.

3.3 Detecting the Optimal Resolution

Criteria similar to the ones proposed in [4] for variational geometric modeling were applied to choose the optimal resolution: add more wavelets to better approximate the trajectory and remove the unneeded wavelets to obtain a solution with lower energy (defined by the objective function (4)). While the first principle can be realized using hierarchical B-splines, the second criterion is much easier to realize in a wavelet basis because the necessary density is reflected in the magnitude of wavelet coefficients.

After calculating the estimate for a trajectory at a certain resolution, we first check the magnitude of wavelet coefficients. If they are below a certain threshold, we remove the corresponding wavelets from the trajectory. The next step is the addition of higher resolution wavelets on time intervals where the model marker positions generated by the recovered trajectory poorly match the measured markers. Wavelets centered on such intervals are added to the solution and their initial coefficients are set to zero. This procedure is repeated until a stable solution is found.

4 Experiments and Discussion

We captured several motion trajectories involving full-body motion of a human performer. A marker-based measurement system Optotrak (see web page http://www.ndigital.com/opto.html) was used for this purpose. Optotrak uses identifiable active markers which is advantageous because full-body movements often cause some of the markers to be occluded. Systems using passive markers are more prone to matching errors in such cases. In our experiments, we used 22 markers distributed over the performer's body.

Experiments showed that our approach offers significant advantages compared to a straightforward approach of sequentially minimizing the criterion function (2) at consecutive measurement times. Firstly, the sequential minimization requires that a sufficient number of markers is visible at each measurement time. This can make the optimization process fail when too many of them are occluded. In the sequential approach, we were able to recover part of the trajectory only by interpolating the marker positions from the times when they were not occluded to the intervals of occlusion. This reduces the quality of the recovered trajectory. These problems were alleviated by the batch recovery of a complete trajectory. Secondly, it turned out that even when the positions of all markers can be measured, the recovery of some of the joint angles becomes sensitive at some configurations due to the kinematic structure of the body as well as the choice of the marker placement on the body. Indeed, we were unable to estimate the torso degrees of freedom and some of the degrees of freedom at

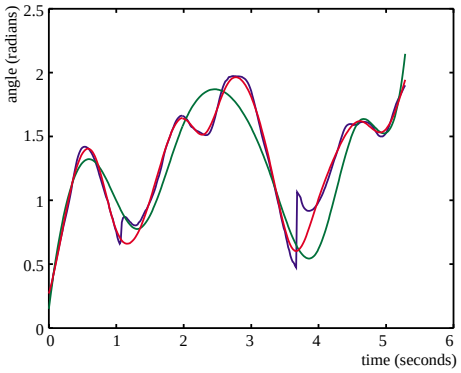

Figure 5: Trajectory of the right arm flexion/extension

the performer's hand using the sequential approach. These problems were resolved in the proposed approach with the introduction of the regularization factor (4) in the optimization criterion. Thirdly, the recovered trajectory is smooth and can avoid discontinuities that can arise in the sequential approach because of the switching between local minima of the objective function.

Our trajectories are high dimensional (over 30 degrees of freedom), so due to space limitations we are unable to present all of their components. Some of the above findings are nicely demonstrated on the recovery of the right shoulder flexion/extension degree of freedom (see Fig. 5). The motion capture process lasted a bit more than 5 seconds and we took 60 measurements per second. The blue trajectory shows the trajectory recovered by the sequential approach. It is more noisy than the other two trajectories and it contains discontinuities. The green trajectory shows the recovered trajectory using an initial cubic B-spline basis (third resolution level, 11 basis functions). While it is less noisy than the trajectory generated by the sequential approach and it does not contain discontinuities, it only poorly follows the measured markers. The red trajectory shows our final result after adding some wavelet functions. It is both smooth and continuous and it follows the measured markers better. The average error between the measured and the estimated marker positions, i. e. the square root of criterion (2) divided by the number of markers, for the last two trajectories is shown in Fig. 6 (dashed: initial B-spline trajectory, solid: final trajectory). We should note here that a significant part of these errors is caused by systematic errors due to inaccuracies in our model. If these errors are larger than the predefined tolerance for the measurement error, then the wavelet adding process should stop supplying new wavelets. Currently, this is done by limiting the upper resolution level.

Our models are built by measuring the performer and the positions of markers attached to her/him. Work on automatic model generation, which should make the model building process more accurate and less cumbersome, is

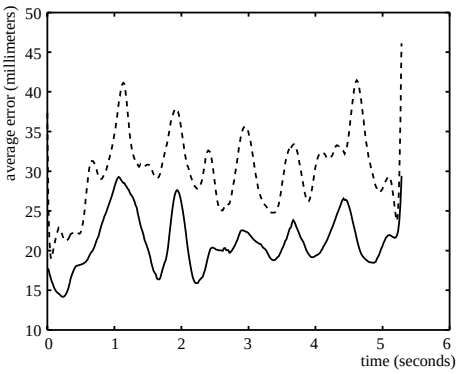

Figure 6: The average error in marker positions

currently under way. Another problem is the mismatch between human and robot. Especially, DB's joint limits are more restrictive than human's. In this work we handled this problem by scaling and translating the reconstructed joint trajectories into the range of DB's joint limits. The regularization term in (5) was used to prevent jumps in velocities and accelerations and thus ensure the generation of motions that can be executed by the robot. This makes robot motion planning different from programming a virtual movie actor or interactive game agent that need only approximate rather than follow real-world physics. See [3] for a computer graphics approach to motion adaptation. Another extension would be the development of a cross-validatory procedure for the automatic determination of the smoothing parameter in criterion (5), as it is standard in biomechanics [12].

A possible field for further research is the development of criterion functions that improve the style of motion [11] and can be added to the objective function (5). Although the objective function (4) does improve the style of motion by reducing its acceleration or jerk, its primary function is to make the trajectory feasible to execute. Our scheme is suitable for the generation of a library of movement primitives. Techniques for an on-line interpolation between such primitives based on perceptual inputs should be developed in the future. Another interesting area of research would be to develop control strategies to combine the learned primitives into longer movement tasks [7]. The humanoid robotics is an attractive test case for such research.

5 Summary and Conclusions

This paper presents a new approach to the formulation and optimization of joint trajectories for humanoid robots using B-spline wavelets. We first outlined the difficulties arising in the specification of full-body motions for humanoid robots and proposed to exploit the similarity between human and humanoid robot motions to generate such trajectories. We demonstrated that B-spline wavelets are suitable for the formulation and optimization of humanoid robots' trajectories at different resolution levels. Finally, we showed how to resolve the resulting large-scale optimization problems to compute such trajectories. The ability to treat large-scale optimization problems that need to be solved to generate optimal full-body motions and to automatically infer the appropriate resolution level draws a distinction between our approach and other approaches proposed for human motion capture in the literature.

Videos showing our humanoid robot performing some dance movements can be seen on DB's home page http://www.erato.atr.co.jp/DB/home.html.

Acknowledgment: Aleš Ude is on leave from the Department of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, Ljubljana, Slovenia.

References

[1] B. Bodenheimer and C. Rose. The process of motion capture: Dealing with the data. In *Computer Animation and Simulation '97, Proc. Eurographics Workshop*, pages 3–18, Budapest, Hungary, 1997. Springer Verlag.

[2] T. Coleman, M. A. Branch, and A. Grace. *Optimization Toolbox User's Guide*. The MathWorks, Natick, MA, 1999.

[3] M. Gleicher. Retargeting motion to new characters. In *Computer Graphics, Proc. SIGGRAPH '98*, pp. 33–42, 1998.

[4] S. J. Gortler and M. F. Cohen. Hierarchical and variational geometric modeling with wavelets. In *Proc. 1995 Symp. on Interactive 3D Graphics*, pp. 35–42, Monterey, California, 1995.

[5] T. Lozano-Perez. Robot programming. *Proc. IEEE*, 71(7):821–841, 1983.

[6] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Boca Raton, New York, 1994.

[7] C. Rose, B. Guenter, B. Bodenheimer, and M. F. Cohen. Efficient generation of motion transitions using spacetime constraints. In *Computer Graphics, Proc. SIGGRAPH '96*, pp. 147–154, 1996.

[8] E. J. Stollnitz, T. D. DeRose, and D. H. Salesin. *Wavelets for Computer Graphics: Theory and Applications*. Morgan Kaufmann Publishers, San Francisco, 1996.

[9] S. E. Thompson and R. V. Patel. Formulation of joint trajectories for industrial robots using B-splines. *IEEE Trans. Industrial Electronics*, 34(2):192–199, 1987.

[10] A. Ude. Robust estimation of human body kinematics from video. In *Proc. IEEE/RSJ Conf. Intelligent Robots and Systems*, Kyongju, Korea, 1999.

[11] A. Witkin and M. Kass. Spacetime constraints. In *Computer Graphics, Proc. SIGGRAPH '88*, pp. 159–168, 1988.

[12] H. J. Woltring. A FORTRAN package for generalized, cross-validatory spline smoothing and differentiation. *Advances in Engineering Software*, 8(2):104–113, 1986.