Automatic and Interactive Key Posture Design by Combing the PIK with Parametric Posture Splicing

Shilei Li, Bing Wu, Jiahong Liang, and Jiongming Su

College of Mechatronic Engineering and Automation, National University of Defense Technology, PhD candidate, Changsha, 410073, P.R. China
leeshileili@gmail.com

Abstract. Key posture design is commonly needed in computer animation. This paper presents an automatic and interactive whole body posture designing technique by combining the PIK (prioritized inverse kinematics) with the proposed parametric human posture splicing technique. The key feature of PIK is that the user can design a posture by adding high level constraints with different priorities. However, the PIK is essentially a numerical IK algorithm which relies on the iterative optimization starting from a good enough initial posture to get the final result. To speed up the running efficiency and ensure the lifelikeness of the final posture, the parametric posture splicing technique is proposed to generate the initial guess of the PIK. According to the set of the high level constraints, the whole body is divided into some partial parts, whose postures are then generated by the parametric posture synthesis from a single posture database. Then an initial posture guess with some main characteristics of the finally acceptable posture can be generated approximately by splicing these partial body postures together. Starting from this initial guess and with all constraints considered at different priority levels, the PIK can be initialized with a bias defined by this particularly initial guess and iterated step by step to get a final posture. The total process of the whole body posture generation is automatic and interactive. The experimental results show that this combination method can not only improve the computation efficiency of the PIK but also can simultaneously ensure the naturalness of the final posture.

Keywords: character animation, posture designing, prioritized inverse kinematics, parametric posture splicing.

1 Introduction

In the computer animation field, key posture design is a fundamental requirement of animators. Today, commercial animation software products, e.g. 3ds Max, Maya, all provide friendly interactive interface for character posture designing. However, key posture design is still a hand-driven technique which requires experience and long-timely humdrum labor.

When the animator designs a particular posture, she always has a rough outline of the desired posture in mind, which usually can be described by a set of high level

A. Nijholt, D. Reidsma, and H. Hondorp (Eds.): INTETAIN 2009, LNICST 9, pp. 119–130, 2009.

constraints. The PIK [1,2] is an efficiently iterative algorithm which can enforce an arbitrary number of strict priorities to arbitrate the fulfillment of conflicting constraints and can run within an interactive environment. Given high level constraints, the whole body posture of complex characters can be generated automatically. However, the PIK is essentially a generalized Jacobian-pseudo-inverse based IK algorithm. The final posture is generated by iterative optimization starting from an initial guess of the posture configuration. Because of the redundancy of the human joints, different initial guesses can generate different final postures. The better the initial guess is, the less computation load and the more lifelikeness of the final result would be realized. Here, the goodness of the initial guess should be measured by the similarity between the initial guess and the final desired posture. In extreme cases, if we can give the final posture directly, there would be no need for the further PIK computation. Although we can never get the exactly final posture directly with only a set of high level constraints, we should set the initial guess close to the final result as much as possible. In this paper the motion capture database is used to generate the initial posture. Nevertheless, the diversity of the possible human postures makes it impossible to establish a posture database that includes all required postures in the future. To reduce the number of sample motions, the parametric splicing technique is purposed to synthesize the initial whole body posture by dividing the body into some partial parts. By combining the PIK with the parametric posture splicing technique, an automatic and interactive key posture designing framework is presented in this paper.

The remainder of this paper is organized as follows. Section *Related works and Contribution* reviews related literatures and outlines our contribution. Next, Section *Overv*iew gives an overview of our posture designing framework. In following two Sections, we detail our method for combing the parametric posture splicing with the PIK algorithm. Our results are shown and discussed in Section *Experimental Results and Discussion*. Finally, in Section *Conclusions and Future Work*, we conclude this paper and discuss some possible directions for the future work.

2 Related Works and Contribution

Key posture design is widely needed in computer animation. The motion quality of the traditionally hand-driven keyframing technique is basically determined by key postures. Nowadays, it is common to leverage on motion capture databases to generate virtual character animations as motion capture technique provides the most lifelike results by replaying the real human motions. However, motion capture data often need some adaptions to reuse in different environments. The adaption process often introduces artifacts and key posture design is still widely needed so as to correct and adjust any undesirable postures. Recently, to relieve the animator from the burden of enforcing physical plausibility, physical interpolation [3,4] is proposed to generate highly detailed and physically realistic motions. However, realistic key postures are still needed as the foundation of the final results.

While designing key postures can be done by directly given parameter values of different joints, it becomes rapidly tedious and time consuming as soon as the total number of animated object's DoFs (degrees of freedom) increases. For character

animation, this is particularly noticeable as typical virtual characters may contain up to fifty DoFs even without considering the fingers. For this reason, specific algorithms have been developed to ease key posture design. IK (Inverse kinematics) is a process for determining the configuration of a character's parameters based on specifications of resulting features of the pose, such as end-effector positions. It can calculate the mid joints angles automatically, providing a goal-directed method in the human posture generation. The control of complex articulated figures using IK often requires that we can simultaneously apply multiple constraints, which may lead to conflicts between tasks because some are not achievable at the same time, whilst they can be separately. In references [1,2], a priority based numerical IK solver(PIK) is presented. The priority strategy ensures that the most important ones are satisfied first and the less import ones are satisfied as much as possible without disturbing the vital constraints. In references [5,6,7], the PIK is used for motion editing. In reference [7], to realize one particular key frame editing, the PIK is combined with the low dimension motion model constructed from a motion database using PCA. This approach provides faster and better quality results with less tuning from the user side compared with references [5,6], which use the PIK to realize motion editing based on the per-frame PIK paradigm. In this paper, the PIK algorithm is adopted to realize the process of conflicting constrains in key posture design. We also combine the PIK with a motion database but here the motion database are used to provide posture samples rather the motion model of one particular motion type like [7].

Our work belongs to the example-based IK methods. Style-based IK [8] is a robust and powerful technique for character posing using a statistical model learned from small datasets. However, it does not guarantee that poses still looks natural when desired poses are far away from the training data. Just like the parametric motion synthesis techniques [9,10,11] which generate new motions exactly corresponding to user-specified parameters, we propose the parametric posture splicing so that the combination of partial postures can be used as the initial guess of one particular set of high level constraints. To reduce the motion samples required for the parametric motion synthesis technique, Ha et al. [12] combined the upper and lower body splicing with the parametric motion synthesis. Here, we also use the splicing to reduce the required posture samples. But unlike these papers which generate motion sequences, we only generate a single posture without the need of the complex preprocess operation to determine the time, space and constraint correspondences between a set of motion primitives. Splicing is a technique to produce a new motion by cutting some part of one motion and attaching it to another. By combining different partial motions, the original motion database can be enriched without capturing new motions. Heck et al.[13] proposed a detailed method to splice the upper body of a motion with the lower body of another. Majkowska et al. [14] proposed a technique that splices hand gestures with body motions. Ikemoto et al. [15] introduced a way of enriching a motion database by changing the limb associated with a motion, and suggest rules for synthesizing motions that look natural. Recently Jang et al. [16] suggested an analogous combination technique which selects and combines coherent partial motions. In these papers, splicing is made just to ensure the naturalness of the final results without considering the exact feature of the finally combined results beforehand. In this paper, splicing is also used

but we focus on the generation a posture entailing some main characteristics of user constraints so that it can be used as the initial guess of the PIK.

Compared with other example-based IK algorithm, the main contribution of our work is the high efficiency and quality of the final results by sequential apply the parametric splicing and PIK. Through the parametric splicing, a natural whole body posture can be generated without complex computation and used later as the initial starting point of the PIK algorithm. By combining the parametric splicing with the PIK, a more robust and efficient IK framework is presented.

3 Overview

The key feature of our posture designing framework is that the parametric posture splicing is combined with the PIK to realize automatic and interactive posture generation. Basically, our approach is divided into two main steps:

1. *Parametric posture splicing*: Posture databases are used to generate the initial guess of the PIK. To reduce the required posture samples, the splicing technique is used. Here, the main problem we need to solve is that how we can ensure the combined whole body posture can be used as an initial guess of the final desired posture. According to the certain set of high level constraints, we divide the whole body into some partial joint groups and generate these partial postures using the weighted interpolation of corresponding parts of different whole body postures. Then the posture of the whole body is spliced and used as the initial guess of the PIK.
2. *Further posture refining using PIK*: The feature of the PIK is its priority strategy. Using the PIK, the complex task with an arbitrary number of priority levels and parametric inputs can be processed. Using the parametric posture splicing results as the initial configuration, the PIK solves for the final results by considering all of user specified constraints with different priorities. As the initial guess generated through the partial interpolation of a posture database, the quality of the final results and the calculation efficiency can be improved compared to situations where only a random or a fixed initial posture configuration is used.

4 Parametric Posture Splicing

Our basic idea is to subdivide the body of a character into several parts, such as the lower and upper body. New postures can then by generated from the combination of partial postures. However, human is a complex life system which moves in a highly coordinated way and combining partial postures in an arbitrary way can not produce the harmony of real human posture. Furthermore, to synthesize a posture that can be used as an initial guess for a particular situation, we must ensure the combined posture have a certain similarity with the desired posture. In summary, we have two questions to solve. First, what division should we use to ensure the lifelikeness of the spliced posture? Second, how to use the same posture database to approximately generate different partial postures we want? In this section, we propose the parametric posture splicing to solve these two questions.

4.1 Posture Database Setup

To do the parametric posture splicing operation, first a posture database needs to be established as a preprocessing step. We use the motion capture data to setup the posture samples. A particular character posture is generally represented as a state vector P , described by the global position and orientation of the root node and a set of joint angles:

$$P = [p_0, q_0, q_1 \cdots q_n] \tag{1}$$

Where p_0 and q_0 represent the 3D global position and orientation of the root joint and q_i is the local transformation of the i^{th} joint expressed in its local coordinate system. It should be noted that a posture is unchanged if we translate it along the floor plane or rotate it about the vertical axis. Most papers adopt the similarity matrix proposed by Kovar et al.[17] to compare different postures. The optimized 2D transformation is used to get the optimal sum of squared distances between corresponding point clouds and the readers may refer to this paper for the details. However, when the posture database is used for the initial posture generation, the virtual character geometry may be different from the actual performer of the motion data. Consequently, when comparing the similarity between two postures, they are joint angles rather than some position values that determine the similarity level. For example, if an adult man and a child have the same joint angles of all of their joints, we think their postures are same in spite of their great differences in positional constraints. Although it is not always true that joint angles rather than position values determine the feature of a pose, for example, if we have a particular positional requirement of a posture (e.g. touch the nose, scratch the back of the head etc), it should be noted that here our purpose is to establish a posture database which can be used to generate the initial guess of the PIK algorithm to ensure the naturalness of the final posture result. The positional constraints of the particular posture will be further processed by the PIK component.

Before the similarity is computed, one of the two poses should be translated and rotated in the horizontal plane to remove differences in absolute position and facing direction. After removing differences in absolute positions and facing directions from the state of the pose, we calculate the similarity of the two postures simply using the following equation:

$$s = \sum_{i=0}^{n} [ws_i \times \left| q_i^{'} - q_i \right|] \tag{2}$$

Where s is the score for the posture similarity and the weighting vector ws_i is used because the angle variation of a joint located at the beginning of the human body kinematics chain has a greater influence on the generated posture than the same angle variation of a joint located at the end of the chain.

To generate the different posture samples, first different types of motion with a complete cycle, such as walking, running, kicking, reaching, sitting, etc. are collected. Then the posture database was established from these motions by adding a new pose

only if the similarity score between that pose and all the poses currently in the database is greater than a threshold. Consequently, we finally get a database containing a set of unique postures, each stored as the joint angles.

4.2 Parametric Partial Posture Generation

Given a certain set of high level constraints, to reduce the number of required posture samples, we first need to divide the whole body into a certain number of partial parts and then generate these partial postures individually using the same posture database. Here unlike traditional methods [13,15,16] which simply divide the whole body into the same division, e.g. the upper and lower body parts, all the time, we divide the body part according to different situations. When a certain set of high level constraints is given, we first determine the underlying joint recruiting level of each constraint. For example, when a desired position of the hand is defined, we need to know if only the upper limb, the spine and the upper limb or even the lower body is needed to finish the task. This can be done automatically according to the relative positional constraints. For example, one constraint may require the character's hand in a certain position relative to its body. Then according to the particular geometrical length of its body model, we can approximately determine the joint recruiting level of this constraint. According to the character geometrical model and joint coupling relations, we establish a set of simple rulers of the recruiting level of different constraints. For example, for the reaching task, only the total stretching length is used to determine whether only the upper limb or the upper limb and other body parts is used for the posture generation. The users can also directly set the recruiting level of different constraints. For two different constraints, if they have common joints for the participation of their realization, they are put into the same partial body part. Here, by dividing the whole body into joint parts differently according to the exact set of constraints, we can ensure the coordination of the whole body posture as much as possible when splicing them together.

After the whole body is divided according to the constraints, we need to generate partial posture of these partial joint groups. For each sample in the database, we do the forward kinematics calculation to get the corresponding constraints value. Using these values, for each joint group JP_i ($1 \le i \le p$, p is the number of the partial body parts by dividing the whole body), we do the following calculation:

$$\begin{pmatrix} c_1^1 & c_1^2 & \cdots & c_1^{np_i} \\ c_2^1 & c_2^2 & & c_2^{np_i} \\ \vdots & \vdots & & \vdots \\ c_{nc_i}^1 & c_{nc_i}^2 & & c_{nc_i}^{np_i} \\ 1 & 1 & \cdots & 1 \end{pmatrix} w_i = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{nc_i} \\ 1 \end{pmatrix} \tag{3}$$

Here, nc_i is the number of constraints which belong to the same joint group JP_i and np_i is the number of the used sample postures. The used partial posture samples

are searched in the database through the k *nearest neighbors* according to the distance between the required constraints value and actually constraints value of each posture samples. The number of posture samples should satisfy $np_i \geq nc_i + 1$. c_{nc_i} is the representation of the value of required constraints and $c_{nc_i}^{np_i}$ is the representation of the actual constraints value of each sample postures. The equation is solved by using a singularity-robust inverse [18] to compute the weight for each sample postures. After the weight vector w_i is got, the joint angles in the partial joint group JP_i can be calculated as:

$$q_i(k) = \sum_{j=1}^{np_i} w_i(j) q_j \tag{4}$$

Where $w_i(j)$ is the j^{th} element of w_i, k is the total number of the joints in the joint group JP_i and q_j denotes the angle of the corresponding joint k of the j^{th} sample posture. In other words, the partial posture is generated by the weighted combined of similar postures in the joint space.

The calculation process above is repeated for each joint group JP_i $(1 \leq i \leq p)$ so that the posture of every partial body parts is parametrically synthesized.

4.3 Whole Body Posture Splicing

After the joint angles of all body parts are calculated, we can combine them together directly to generate the whole body posture. Unlike the complex motion splicing operator [13,14], here we need no further processing of the temporal or spatial correlation between the postures of different body parts. Because we divide the whole body into partial body parts according to the exact constraints and the interpolation of joint angles rather than the joint positions are used for the partial posture generation, the whole body posture looks natural at most of the times. Nevertheless, combining partial postures together directly can never always ensure the lifelikeness of the whole body posture. However, through the parametric posture splicing technique proposed above, we realize the most important requirement that the initial guess should have a certain global similarity with the desired posture. It should be noted that here the spliced body posture is only used as an initial guess of the final posture and it will be refined by the PIK later. Furthermore, the animators can interactively change the whole body posture at this stage.

5 Automatic and Interactive Posture Generation

After the initial guess of the whole body posture is generated, we can refine the initial guess to get the desired whole body posture with the PIK. In this section, we first give an overview of the PIK algorithm and then we use the PIK to get the final posture with all constraints satisfied as much as possible.

5.1 Overview of the Prioritized Inverse Kinematics Algorithm

The PIK algorithm presented by Paolo Baerlocher et al.[1,2] is an extension to the Jacobian pseudo-inverse based numerical inverse kinematics method. The main originality is that they presented a recursive algorithm to speed up the traditional null space projectors computation and generalized the previous priority based approach to an arbitrary number of tasks with multiple levels of priority. Here, we give the final formulation directly. Considering t tasks T_p ordered from the highest priority ($p=1$) to the lowest ($p=t$), the algorithm can be summarized as follows:

$$\Delta q_p = \Delta q_{p-1} + (\tilde{J}_p)^{+\lambda_p}(\Delta x_p - J_p \Delta q_{p-1}) \tag{5}$$

$$\Delta q_1 = (\tilde{J}_1)^{+\lambda_1}\Delta x_1 \tag{6}$$

Where $\tilde{J}_p = J_p P_{N(J^A_{p-1})}$, $P_{N(J^A_p)} = I - (J^A_p)^+ J^A_p = P_{N(J^A_{p-1})} - (\tilde{J}_P)^+ \tilde{J}_P$ and $P_{N(J^A_0)} = I$. $J^A_p = \begin{bmatrix} J_1 \\ J_2 \\ \vdots \\ J_p \end{bmatrix}$ is the so-called augmented Jacobian because obviously the whole null space of these tasks (for tasks with the priority from *1* to p):

$$N(J_1, J_2 \cdots, J_p) = N(J_1) \cap N(J_2) \cap \cdots N(J_{p-1}) \cap N(J_p) = N(J^A_p) \tag{7}$$

In the formulation above, J^+ is the pseudo-inverse defined by $J^+ = J^T(JJ^T)^{-1}$ and $J_i^{+\lambda_i} = J_i^T(J_i J_i^T + \lambda_i^2 I)^{-1}$ is the so-called damped least squares inverse using the damping factor λ_i to deal with singularities.

In addition, linear equality or inequality constraints, e.g. relative position constraints and joint limits, can also be processed in the framework of the PIK algorithm. Readers may refer to references [1,2] for further details.

5.2 Using the PIK Algorithm for Further Refining of the Spliced Posture

Using the PIK, we firstly need to prioritize all of the constraints. After the priority levels of different constraints are set, we can use the PIK algorithm directly. Here, besides using the spliced whole body as the initial guess, we further add an optimization item at the lowest priority level to make the final result resemble the initial guess as much as possible. We define a cost function as follows just like those used in motion editing [5,6]:

$$g(q) = \frac{1}{2}(p - p_s)^T K_w (p - p_s) \tag{8}$$

Where K_w is a weighting definite diagonal matrix setting relative importance of different joints and p_s is the initial posture. With this cost function, we can solve the finally desired joint increments by setting this optimization term as the lowest task using the same null space projection principle. At the PIK stage, the joints near the root node are often adjusted more than the joints at the end of the chain to exactly satisfy some positional constraints. Consequently, here we set very large values on the joints at the end of the chain and zero of the spine joints which near the root node. By setting the weighting matrix this way, we can ensure the final posture resemble the initially spliced posture as much as possible. During the priority loop, after the lowest priority constraint T_t ($p = t$) is processed, we can get the finally desired joints increments Δq by adding Δq_t with the following optimization term:

$$\Delta q = \Delta q_t - P_{N(J_t^A)} \alpha \nabla g \tag{9}$$

Where α is a positive constant and $\nabla g = \frac{\partial g}{\partial q}$. Here, we choose the negative gradient of the cost function $-\alpha \nabla g$ to ensure the final Δq ensemble the certain joint angles as close as possible. By adding this optimization item as the lowest task, we can ensure the lifelikeness of the final result by solving the redundancy with the certain joint angles of the initial guess.

6 Experimental Results and Discussion

This section presents some postures generated for a driving posture design task. The articulated character has 40 degrees of freedom (7 per upper limb, 7 per leg, 3 for back and waist respectively and 6 for the root joint). All postures are generated based on the same posture database. Tests were run on a single-core 3.5 GHz Intel Pentium 4 processor with 2GB memory under Windows XP operating system.

In this example, we use our framework to design a particular diving posture. We define five positional constraints: one for the hip, two for the hand and two for the foot. The hip constraint is given the highest priority level. The two hand constraints are given the second priority level and the two feet constraints are lastly considered. According to the recruiting level of each constraint, the whole body is divided into the upper and lower part. Two different geometrical character models are used. The initially spliced whole body posture of the character one is shown on the left of the figure 1 and the corresponding final whole body posture refined by the PIK is shown on the middle left. The initially spliced whole body posture of character two is shown on the middle right of the figure 1 and the corresponding final whole posture refined by the PIK is shown on the right. It can be seen that the initial guesses of the whole body posture for the two different characters are almost the same because the

parametric posture splicing technique use the same posture database to generate them. These small differences come not only from the difference between geometrical models but also from the nonlinear relationship between the joint angles and the end-effectors' positions. The final whole body postures of the two characters have bigger differences because the PIK are used to generate the posture with positional constraints are satisfied as much as possible.

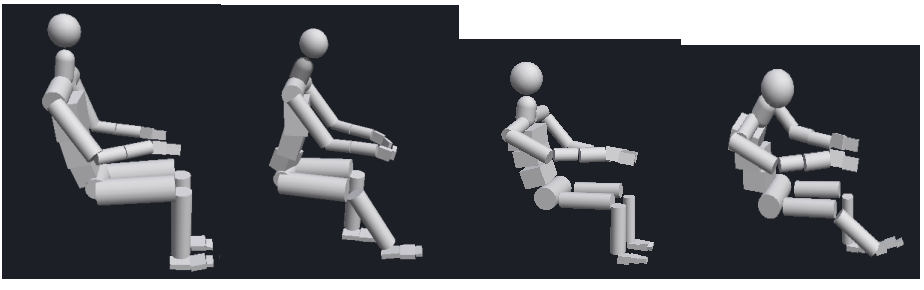

Fig. 1. The results of whole body posture: form left to right (a) initial guess of the character one (b) final result of the character one (c) initial guess of the character two (d) final result of the character two

From this experimental result, it can be seen that the final whole body postures are natural by using the initial guesses generated from the posture database. We also generate the whole body posture from the standing posture. For this experiment, the whole body posture generated by only using the PIK also looks natural. However, the computation time is approximately three times longer than our methods. For more complex tasks, we think the benefits of our method would be greater.

Currently, the main limitation of our method is that we search through all poses in the database according to the constraints and consequently the database needs to be small to use our technique in real-time situations. However, our method is more robust and intuitive than those using a statistical model learned for motion samples [8].

7 Conclusions and Future Work

In this paper, we have developed a whole body posture design framework that relies on the PIK, the parametric posture splicing technique and a library of example postures. At the preprocessing stage, we setup a posture library based on the motion capture data. To process the difference in the human's geometrical models, joint angles rather that joint positions are used. During the posture designing stage, after giving a set of high level constraints, we first divide the whole body into several body parts. Then a simple and efficient partial posture synthesis method is proposed to generate the partial body postures. The advantage of our method is that the posture samples need not be a perfect match for the constraints and the new whole body posture with the main characteristics of the constraints can be generated efficiently. Finally, the PIK is used to generate the final whole body posture. In essence, by combing the PIK with the parametric posture splicing, we divide the traditional PIK

into two steps. The first step is to generate the initial guess using a posture database. The second step is to refine the initial guess using the PIK. Consequently, the whole body posture generation framework is automatic and interactive. Furthermore, the burden of the posture design is greatly reduced and simultaneously the quality of the final posture is improved.

In the future, firstly we want to integrate our framework into the commercial animation software products so as to provide a friendly interface to the users. Secondly we do not yet have a way of measuring how much the spliced posture is appropriate to the certain constraints or how many posture samples are sufficient to produce natural looking posture. We need do more research on how the whole body can be coordinately combined and search for more reliable methods for deriving good initial postures from observed motion samples.

Acknowledgments. The data used in this paper was obtained in part from mocap.cs.cmu.edu. This database was created with funding from NSF EIA-0196217.

References

1. Baerlocher, P., Boulic, R.: An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. The Visual Computer 20(6), 402–417 (2004)
2. Baerlocher, P.: Inverse kinematics Techniques for the Interactive Posture Control of Articulated Figures. PhD thesis, Ecoles Polytechniques fédérale de Lausanne, Swiss Federal Institute of Technology (2001)
3. Fattal, R., Lischinski, D.: Pose controlled physically Based Motion. Computer Graphics Forum 25(4), 777–787 (2006)
4. Allen, B., Chu, D., Shapiro, A., Faloutsos, P.: On the Beat! Timing and Tension for Dynamic Characters. In: Proceeding of Eurographics/ ACM SIGGRAPH Symposium on Computer Animation, pp. 239–247 (2007)
5. Boulic, R., Le Callennec, B., Herren, M., Bay, H.: Motion Editing with Prioritized Constraints. In: Proceedings of 1st International Workshop on Interactive Rich Media Content Production - Architectures, Technologies, Applications, Tools (2003)
6. Le Callennec, B., Boulic, R.: Interactive motion deformation with prioritized constraints. Graphical Models 68(2), 175–193 (2006)
7. Carvalho, S.R., Boulic, R., Thalmann, D.: Interactive low-dimensional human motion synthesis by combining motion models and PIK. Computer Animation and Virtual World 18(4-5), 493–503 (2007)
8. Grochow, K., Martin, S.L., Hertzmann, A., Popovic, Z.: Style-based inverse kinematics. ACM Transactions on Graphics 23(3), 522–531 (2004)
9. Rose, C., Cohen, M.F., Bodenheimer, B.: Verbs and adverbs: multidimensional motion interpolation. IEEE Computer Graphics and Applications 18(5), 32–40 (1998)
10. Kovar, L., Gleicher, M.: Flexible automatic motion blending with registration curves. In: SCA 2003: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 214–224 (2003)
11. Kovar, L., Gleicher, M.: Automated Extraction and Parameterization of Motions in Large Data Sets. ACM Transactions on Graphics 23(3), 559–568 (2004)
12. Ha, D., Han, J.: Motion synthesis with decoupled parameterization. The Visual Computer 24(7), 587–594 (2008)

13. Heck, R., Kovar, L., Gleicher, M.: Splicing upper-body actions with locomotion. Comput. Graph. Forum. 25(3), 459–466 (2006)
14. Majkowska, A., Zordan, V.B., Faloutsos, P.: Automatic splicing for hand and body animations. In: SCA 2006: Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 309–316 (2006)
15. Ikemoto, L., Forsyth, D.A.: Enriching a motion collection by transplanting limbs. In: SCA 2004: Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 99–108 (2004)
16. Jang, W.S., Lee, W.K., Lee, I.K., Lee, J.: Enriching a motion database by analogous combination of partial human motions. The Visual Computer 24(4), 271–280 (2008)
17. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. ACM Transactions on Graphics 21(3), 473–482 (2002)
18. Nakamura, Y., Hanafusa, H.: Inverse Kinematics Solutions with Singularity Robustness for Robot Manipulator Control. J. Dynamic Sys., Meas., and Control, 108, 163–171 (1986)