Motion Edit with Collision Avoidance

Li Liu
[1] Institution of Computing Technology, Chinese Academy of Sciences
[2] Graduat School of Chinese Academy of Sciences
Beijing China, 1000801
liuli@ict.ac.cn

Zhao-qi Wang
Institution of Computing Technology, Chinese Academy of Sciences
Beijing China, 1000801
Zqwang@ict.ac.cn

Zhu Deng-ming
[1] Institution of Computing Technology, Chinese Academy of Sciences
[2] Graduat School of Chinese Academy of Sciences
Beijing China, 1000801
dmzhui@ict.ac.cn

Shi-Hong Xia
Institution of Computing Technology, Chinese Academy of Sciences
Beijing China, 1000801
xsh@ict.ac.cn

ABSTRACT

The existing motion editing methods don't take collision between the limbs into account potentially producing implausible motions. Facing this problem, in this paper we present an integrated framework of motion editing for producing collision-free motion in real-time. We first provide an efficient scheme for collision detection based on skeleton model, which can fast find the penetration between limbs of human body. Then we provide a novel scheme for constraint generation, which embeds the motion characteristics and inter-frame coherency into the constraint generator, and preserves the qualities of the original motion and motion consistency in the target motion. Finally, we show how to apply the Kalman filter to constraint resolve in a real-time manner. Comparing to other existing methods, our method not only can obtain the collision-free motion in real time, but also can preserve the original characteristics as many as possible and generate the new motion similar to the original motion. Experiment shows that our approach is very useful in producing natural and collision-free motion and efficient enough for application in animation system and game.

Keywords

motion editing; self-collision detection; collision avoidance; constraint-based animation

1. INTRODUCTION

Motion capture technique is an increasing popular approach for generating realistic human motion. However, the captured data is only for special person in performing specific motion. Whenever the model or the environment is changed, the data must be edited account for different anthropometric scales or reaction to environment interactions.

In the past few years, researchers have developed various constraint-based motion editing techniques [Tak05, Gle98, Lee99, Shi01, Cho00]. These techniques mainly focus on deforming the captured data to satisfy the given constraints and generating new natural-looking motion. But they seldom consider the potential collision between limbs of the articulated figure. Behavioral reasonable properties may therefore be lost in the editing process, making motions appear unrealistic. For example, if we simply transfer walk motion of a thin character to a fat man, the hand will likely to interpenetrate the body. So, detection and avoidance self-collision are vital for the human-like animation.

Although the collision-free is so important in motion editing, unfortunately, producing collision-free motion is still a difficult task. Firstly, collision detection algorithm can determine if the interpenetration has occurred in the editing process. Virtual human model is typically represented by thousands of meshes. Since these meshes need to be updated and overlapped test according to the movement of the human body at each frame, the cost of this computation is huge. Secondly, in addition to fast collision detection algorithm is required, a real-time constraint resolve algorithm is also necessary to further accelerate the computational speed. Finally, as we know that the motion editing methods are based on the reference motion, thus it is important to preserve the original motion characteristics in the

Conference proceedings ISBN 80-86943-03-8
WSCG'2006, January 30-February 3, 2006
Plzen, Czech Republic.

editing process without extra computation cost. Unfortunately, current motion editing techniques are largely difficult to satisfy the last two requires at the same time.

Overcoming the challenges outlined above, combining with the techniques of collision detection, constraint generation and constraint resolve, we, in this paper, present a novel approach to generate self-collision-free motion in real time.

The main contributions of our approach as follows:

(1) We select Oriented Bounding Boxes (OBBs) as the coarse hierarchy representation of human body. For the human model, the OBB can easily be constructed and updated. Moreover the OBB can bound the geometry tightly and efficiently eliminate the meshes that are absolute collision-free within a frame.

(2) We provide a fast collision detection method. Unlike the previous methods, we integrated the limit of kinematic reachability for joint into the collision detection algorithm. The pair of bounding boxes that can not close each other won't be checked. Thus, the number of pairs to be detected will substantially reduce. In addition, the method can be applied in other research fields, such as the robotics.

(3) Applying the kalman filter to per-frame resolve the constraints. Different from the previous method, kalman filter not only can accurately deal with the nonlinearity of constraints, but also can minimize the difference between before and after the motion editing. Thus the pose at each time frame can be optimal estimated and the characteristics of the captured motion can be well preserved in the final motion.

(4) Presenting a constraint generation technique. Per-frame method requires acquire the end-effector kinematics constraints for each frame in advance, However the existing methods of constraint generation, such as key frame interpolation [Bin98] and reusing the end-effector position of the performer to the target character [Cho00, Bad93], only can guarantee the interaction between the target character and the environment, the motion characteristics of original motion will be lost. For this reason, we present a reasonable and efficient method of constraint generation by integrating the captured motion characteristics and inter-frame coherence into the constraint generator.

The rest of this paper is organized into several sections: We begin our discussion with related work in section 2, and then provide an overview of our scheme in section 3. Technique of constraint generation is provided in section 4. In section 5, kalman filter is applied to per-frame resolve the constraints in real time. In section 6 we describe our method of self-collision in detail. We demonstrate some experimental results in Section 7. Finally we conclude the paper and discuss the future work in section 8.

2. RELATED WORK

2.1 Collision Avoidance

To our knowledge there is little work in human motion animation that edits captured motion considering the collision avoidance. However, many algorithms in other research fields have been designed with the same goal.

In the robotics field, collision avoidance and path planning are very crucial techniques, and many exact and heuristic solutions have been developed [Can88]. With the increasing number of the DOF (Degrees Of Freedom), however, the computing complexity of these methods grows exponentially. Obviously, these solutions can not satisfy the requirement of collision avoidance in human animation.

In the physically based animation field, collision was detected and dynamic equations were applied to simulate collision response [Ban95]. However, in human motion animation, when some key frames are changed to avoid collision, the whole duration of the motion, rather than local, should be modified correspondingly. So for the human motion animation, instant reaction is not a good choice.

In the inverse kinematics field [Hua96, Zha94], they use the sensors to detect the collision, and collision avoidance is integrated into the inverse kinematics equation. But the method is progressive: each time a collision is detected, the inverse kinematics brings the end-effector to a particular place. Thus, the motion goes from one place to another place in the end, which does not ensure a coherent or realistic motion.

Close to our problem is the work of J-C.Nebel[Neb00], who was interested in generating collision-free motion with key-frame interpolation. After collision has been detected, his method creates new sub-key frames between the given key frames. Then these key and sub-key frames are interpolated to create new collision-free motion. However, the key-frame interpolation may severely lose the original characteristics of captured motion, thus makes very hard to generate a new motion similar to the captured motion. Facing this problem, we present an integrated framework of motion editing for generate collision-free motion. Comparing with Nebel's work, our method can well preserve the characteristics of the captured motion. We also

develop a new self-collision detection algorithm special for the articulated model in our method, which can significantly accelerate the detection rate.

2.2 Motion Editing

Motion editing has been well-studied in the past few years and some of commonly used methods are based on spacetime optimization and per-frame methods. Gleicher [Gle98] presented a method using the spacetime optimization technique within the whole motion duration. In his method, inter-frame cohenrence can well be preserved in the target motion, but the computation is huge and only can deal with short motion sequence at interactive speed. For the per-frame method, it can handle infinitely long motion sequence in real time, but the inter-frame cohenrence is difficult to be guaranteed in the editing process. Lee and Shin [Lee99] divided the motion editing problem into per-frame inverse kinematics following by B-Spline fitting. Tak[Tak05] applied kalman filter and least-squares filter. Both techniques use per-frame method with a post filter to smooth the motion. But the two phases will interact in some cases. For example, post filter will ruin the kinematics constraints done by per-frame resolve. Recently, Choi and Ko [Cho00] developed an on-line retargeting method based on per-frame inverse rate control, Shin et al [Shi01] also presented an on-line technique based on IK and the notion of dynamic importance of end-effectors. Without the post filter, the two methods maintain the inter-frame coherence by inputting the continuous path. However the existing per-frame techniques do not introduce an acceptable method to generate constraint trajectory, which not only can satisfied specified constraints, but also can contain the captured motion characteristics. For this problem, we provide an elegant solution to develop a reasonable constraint generator.

2.3 Collision Detection

Current approaches to collision detection seldom deal with self-collision detection of chain structure, but rather propose general methods. These approaches are usually divided into two main classes: one is based on features, and the other is based on bounding volume (BV) and spatial decomposition techniques. The feature-based approaches [Van97, Bro01] exploit the spatial and temporal coherence and compute minimal separation distance. However they are only suited for the case of simple convex polyhedron. Therefore, the method can not handle self-collision detection of human body. Human body is a very complex polyhedron containing thousands of meshes. Typical examples of BV include axis aligned bounding boxes (AABB) [Van97] and oriented bounding boxes (OBB) [Got96b]. Hierarchical structure include sphere [Qiu94], cone tree, k-d tree and octrees [Sam89], other special representation also include binary space partitions (BSP) [Nat90] etc. Although these methods can be used to detect self-collision of human body, but that is very wasteful. According to analyses above, we apply the OBB hierarchy to represent human body, combine the separate axis theorem and the method of triangle-triangle test to detect self-collision, together with the topology information and joint reachability limit to avoid checking collision-free limb pairs.

3. OVERVIEW

In this study, we present a method to generate realistic and collision-free motion in real time. The method combines our previous work [Liu05] with collision detection technique. Figure 1 demonstrates the flow chat of our method. The constraint generator is first applied to generate constrains, which satisfy the given constraints. Then kalman filter is used to edit the captured motion according to the given constraints. Next the edited motion is sent to a collision avoidance system and self-collision between limbs is detected. If there is no collision, which means the result is correctly. Otherwise, based on the detected result, we interactively correct some key frames to avoid collision in the next editing step. That is to say, adding new constraints to the original motion. Finally the corrected key-frame is fed back to constraint generator to reconstruct a new constraint trajectory for the end-effector of human body. The procedure will iterate until the final motion is collision-free and satisfies the specified constraints.

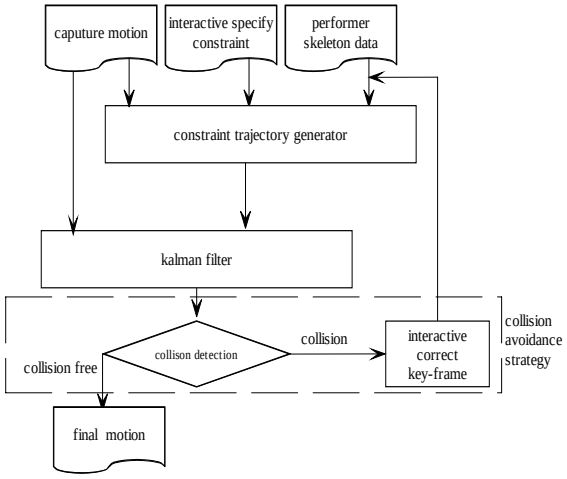


Figure 1.The flow chart of motion editing with collision avoidance

4. CONSTRAINT GENERATOR

Constraints are some features that are to be preserved or changed in the target motion. They contain three kinds of data: the constraint type, the scope of the constraint and constraint value. Specification of

constraint is an important step, because constraints will directly control the properties or details of the target motion. For example, if we specify the restriction of elbow in advance, it won't bend backwards or obtain unrealistic target motion.

Our constraint resolver is based on per-frame kalman filter approach. Since per-frame method need specify constraints at each frame, we can write the constraint as $M(t)$, $t \in [t_0, t_1]$, where $M(t)$ is the constraint value at time *t*, $[t_0, t_1]$ is the scope of the constraint. For the different type constraint, $M(t)$ has different expression.

4.1 Kinematics Constraint

As we stated earlier, our motion editing method is based on reference motion. So original motion pattern should be well preserved in the target motion, otherwise the existence of the reference motion has no sense. Up to this point, we not only need consider satisfying the given spacial constraints, but also must integrate the characteristics of captured motion into constructing kinematics constraint. (See detail in [Liu05])

How to contain the motion characteristics into the kinematic constraint is a great challenge. In our method, we solve it from the following two steps:

(1) Since the joint angle data of captured motion contains some important motion features, and the forward kinematic expression can restore such features effectively, we construct a base end-effector constraint trajectory by entering the captured data q^{org} and skeleton information of target character l^{target} into the forward kinematic equation f_{fk}.

$$M^{org}(t) = f_{fk}(q^{org}, l^{target}) \quad t \in [t_0, t_1]$$

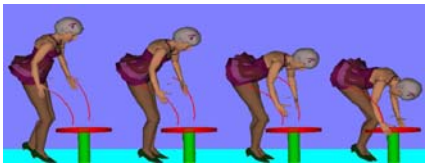
Although captured motion characteristics have been well preserved in this trajectory $M^{org}(t)$, it still can not completely satisfy the given constraints. In the next step, we will further modify the base constraint trajectory according to the given constraints; moreover, we should avoid losing motion characteristics as few as possible.

(2) How to modify the constraint trajectory is another important problem. The Result of the motion frequency analysis [Unu95] has proved that low frequency components of a motion stand for the basic ingredient of a motion, such as motion amplitude, while high frequency components denote the motion details. Consequently, in order to preserve the original motion characteristics as many as possible, we apply the technique of displacement mapping to construct kinematic constraint trajecotry, thus will avoid adding new high frequency information to the target motion. So the final constraint expression can be written as

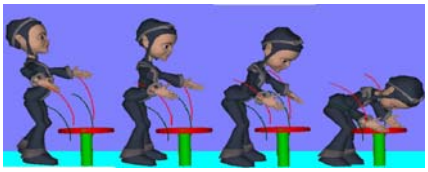
$$M(t) = M^{org}(t) + d(t) \qquad t \in [t_0, t_1]$$

Where $d(t)$ is a displacement trajectory, which can be obtained by linear or cubic interpolation based on the displacement value d_k. The d_k can be computed by subtracting the $M^{org}(t_k)$ from $c(t_k)$ at time k, that is: $d_k = c(t_k) - M^{org}(t_k)$

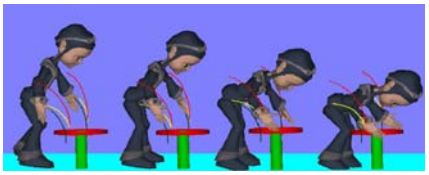
(a) Two different character models



(b) The original motion of bending over



(c) The edited result based on the end-effector trajectory of the performer



(d) The edited result based on our constraint generator

Figure 2: Compare the edited result based on the different constraint trajectory

Figure 2 proves the advantage of our method. The red curve is the end-effector of performer, the black curve shows the trajectory of the motion that is generated by directly transferring the captured data to the target model, and the yellow curve is the trajectory constructed by our method. Compare the edited result following on different constraint trajectory, we can see, that our result (d) is more similar to the original motion (b) than result (c) does.

4.2 Joint Limit Constraint

A joint value of a human is generally bounded to a valid domain, $[\theta_{\min}, \theta_{\max}]$. Once the limitation is broken, the motion will look unrealistic or uncomfortable. To solve this problem, we add joint limit constraint into our constraint generator. So the joint angle of the target motion is within the specified limits.

When the joint angle q exceeds the specified limit $[\theta_{\min}, \theta_{\max}]$, we reduce the q to the given limits. Thus, the corresponding constraint expression can be written as follows:

$$M(t) = \begin{cases} q & if\ q \in [\theta_{\min}, \theta_{\max}] \\ \theta_{\max} & if\ q > \theta_{\max} \\ \theta_{\min} & if\ q < \theta_{\min} \end{cases}$$

5. KALMAN FILTER RESOLVER

To obtain the reasonable and realistic motion, both the constraint generation and the constraint resolve are two equally important aspects. An efficient resolver need to adjust the pose according to the specified end-effector constraints in real time manner, while keep the characteristic of the original motion. For this purpose, we use the UKF [Wan97] (unscented kalman filter) to solve the constraint in real time.

UKF is a method of optimal state estimation, which can accurately handle the non-linear constraint, and minimize the difference of pose between before and after motion editing. In our method, the inputs are the captured data q_i^{org} and the specified constraint $M(t_i)$ at ith frame. Here, the captured data q_i^{org} is regard as state value and the constraint $M(t_i)$ as actual observation value. The output is the new desired pose. During the editing process, we can further control the final pose based on the input value of the process mode and the constraint expression [Liu05]. The algorithm can be described as follows:

Algorithm:

For each frame i,

1) Process model

The process model is defined as: $x_i^- = q_i^{org}$

Where x_i^- is the process model, equaling to the captured motion data at ith frame. Since the original motion contains excellent motion quality, we select the q_i^{org} as the process model. Thus we can easily estimate the similar target motion. In this step, if x_i^- is a n-dimension vector, we need to set a n×n diagonal matrix, P_i^-, as the input value. Here P_i^- relates to the displacement of each DOF in the editing process. Moreover, it can further control whether some DOF is changed before and after editing.

2) Calculation of the sample points

Given an n-dimension vector x_i^-, we can calculate m sample points χ_m, m = 2n +1. These sample points completely capture the mean and covariance of the vector x_i^-.

$$\chi_0 = x_i^-$$

$$\chi_m = x_i^- + (\sqrt{(n+\lambda)P_i^-})_m \quad m = 1, \cdots, n$$

$$\chi_m = x_i^- - (\sqrt{(n+\lambda)P_i^-})_{m-n} \quad m = n+1, \cdots, 2n$$

In which λ is a scaling parameter (see [Van01]).

3) Measurement model

Since the given constraints must be satisfied in the editing process, we define the constraint expression as the measurement model. That is,

$$Y_m = f(\chi_m) \quad m = 0, ..., 2n$$

Y_m is the predicted measurement value of the mth sample point. The mean y_i^- is approximated by a weighted sample mean:

$$y_i^- = \sum_{m=0}^{2n} W_m^m Y_m$$

The weight is given by:

$$W_0^m = \lambda/(\lambda + n)$$

$$W_0^c = \lambda/(\lambda + n) + 1 - \alpha^2 + \beta$$

$$W_m^m = W_m^c = 1/2(\lambda + n) \quad m = 1, ..., 2n$$

4) The gain of Kalman filter

$$P^{yy} = \sum_{m=0}^{2n} W_m^e (Y_m - y_i^-)(Y_m - y_i^-)^T$$

$$P^{xy} = \sum_{n=0}^{2n} W_m^c (\chi_m - x_i^-)(Y_m - y_i^-)^T$$

$$k_i = P^{xy}(P^{yy})^{-1}$$

5) Update of state value

Based on the specified constraint $M(t_i)$, new pose $\hat{x}_i$ at i-frame can be computed by:

$$\hat{x}_i = x_i^- + k_i(M_i^{des} - y_i^-)$$

6. SELF-COLLISION DETECTION

In this section, we present an efficient and real-time self-collision detection method. The method can be divided into two phase: the coarse phase and the fine phase. The coarse phase quick eliminates the meshes that are absolutely collision-free within a frame based on OBB. While the fine phase identifies how many meshes are indeed overlapping in all potential collision meshes.

6.1 Hierarchical Representation of OBB

6.1.1 Construction of Bounding Box

Before building the hierarchical representation of the human body, we need to decide what type of Bounding Volume (BV) to be used. Here, we apply a cost equation [Got96] to select a reasonable BV. For a human body, we believe that OBBs is a good choice. There are two reasons why we choose the OBB against other kinds of bounding volume:

(1) OBB can fit the original model as tightly as possible, thus the number of checking overlap in the fine phase will reduce significantly.

(2) The human model can be described as kinematic chain consisted of limbs, and every limb is nearly symmetrical, so the OBB hierarchy can be easily constructed and updated without any extra cost. Since the kinematic chain of human body is generally subdivided into limbs corresponding to the bones, and each limb includes a subset of triangle meshes relative to the whole triangle meshes of body, so we can construct an OBB for each limb and compute the size of OBB according to the subset of triangle meshes. An OBB is a rectangle box defined by three axes, three radiuses and a centre superposed to the centre of the corresponding limb. Because every limb is nearly symmetrical, we select the main axis of OBB along the connection orientation between the child joint and the parent joint, and then the other two axes can be obtained by the covariance matrix of the subset of the triangle mesh. As for the three radiuses, they can be computed by maximum and minimum extend of mesh subset along each axis. The construction of OBB is shown in figure 3. For the special limbs, such as the hip and shoulder, we note that they have two symmetrical and relative fixed bones, so the two bones can be replaced by a new bone to construct bounding box, as shown in figure 4.

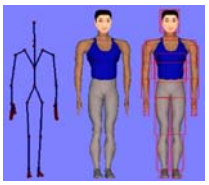


Figure 3. Skeleton, human body and OBB hierarchy model

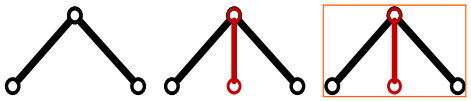


Figure 4. The OBB of hip: the black line means the bones of left hip and the right hip, respectively; the red line stands for the replacement bone; the yellow rectangle means the bounding box of the hip

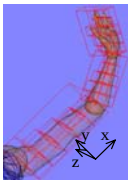


Figure 5. The OBB hierarchy of upper limb with five subdivisions along the bone, the coordinate is selected in terms of the upper arm, and x axis is the main axis

In order to eliminate substantially the collision-free meshes, we further subdivide each OBB into several parts along the main axis (the orientation of bone). Therefore the limbs is divided several parts by the sub boxes and these sub boxes can bound the limb more tightly than its parent box. Because we only need to test the pair of triangle meshes falling in the sub-box in the fine phase, the computational complexity is further lower. An example is shown in figure 5.

6.1.2 Obb Update

Whenever the change is applied to the freedom of human body, all bounding boxes that contain the affected joint need to be updated, so efficient update of OBB is crucial for the effectiveness of the collision detection. In the process of updating, the vertex position P_B of the bounding box and its sub-box are unchanged in the local coordinate system *B*, and the origin of the local coordinate system can be presented as a rotation matrix R_A^B and a translation vector T relative to the global coordinate system A. So for each update, the vertex position P_A in the global coordinate system can be computed by;

$$P_A = R_A^B \cdot P_B + T$$

So, our method can update the OBB at linear cost.

6.2 Collision Detection

In the coarse phase, we first detect collision between different OBBs, if the two OBBs have overlapped, we continue down the OBB hierarchy detecting collision between different sub-boxes, until all potential collision sub-boxes is detected. The phase can quick eliminate the collision meshes.

Collision detection between two OBB can be handled by the theorem of separating axis [Got96a]. For two OBB in space, there are 15 potential separating axes to be tested. Every OBB is projected onto these axes to form an interval, if the two intervals have not overlap, which means the two OBBs are collision-free.

For the human body containing N parts and N-1 joints, the number of OBB pairs that must be check is: $P = (N^2 - N)/2$. For example, in our model, N = 23, the number of limb pairs to be checked is 253. So the computation is huge. However, the human body is also different from the general chain, and the movements of its joints are all limited in a specified range. That means some pair of limbs are absolutely collision-free, for example the neck will never penetrate the thigh or the head. Based on this idea, we can utilize the limit of joint to pre-compute all potential collision-free limbs. Thus, the number of limb pairs which must be checked can be significantly reduced.

The above method works well in "rejection test", but if two objects have collision, the method needs to divide the OBBs continuously to detect potential collision. With the increasing of the number of sub-box, the performance of detection will slow down severely. For this reason, we use triangle-triangle detection [Tom97] to detect collision among triangles meshes in the fine detection phase.

7. Example

The articulated figure in our experience has a total of 23 limbs and each has 3 DOFs. The number of pairs should be checked is 253. Considering the kinematic reachability limitations, the number of remaining pairs will lower to 127. The computation reduces significantly.

In this experience we reuse the jump motion data to a target model. Figure 6 shows the original motion of the performer, and figure 7 shows the result that we transfer directly the captured data of straight jumping to the target model. Since the limb length of the virtual character is different from that of the performer, collision occurred: the hands penetrate his head. In figure 8, we drag the hand to a desired position, feed back the new constraint to constraint trajectory generator, and then build a new constraint trajectory

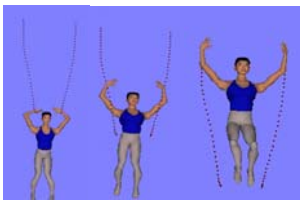


Figure 6. The original motion

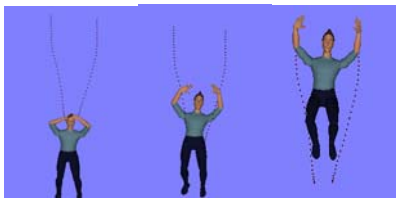


Figure 7 Transferring the original data to the target character, collision occurred

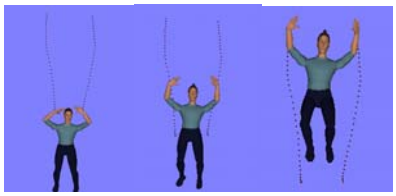


Figure 8: The final motion with collision free

8. CONCLUSION

In this paper, we provide a new method for editing the motion-captured data to produce the collision-free motion in real time. The central idea of our method is to integrate the issues of collision detection, constraint generation and constraint resolve to produce the collision-free motion. Further more, for each of these issues, we all provide an elegant solution. Comparing to other existing methods, our method not only can obtain the collision-free motion in real time, but also can preserve the original characteristics as many as possible and generate the new motion similar to the original motion.

9. ACKNOWLEDGEMENTS

This research is supported by National 973 project, Grant(2002CB312104); Key Cooperation Project of International Science and Technology (2005D-FA11060); Key Project of NSF (60533070); NSF of China, Grant (60573162, 60403042, 60473002); 863 Plan of China, Grant (22004AA115130, 2005AA-114010);National Special Item for Olympics, Grant (Z0004027040331,Z0004024040231); Beijing Natural Science Foundation (4051004, 4062032); and Knowledge Innovation Project 20056380 of Institute of Computing Technology, Chinese Academy of Sciences. We thank for the contributions to this work from all our colleagues in the Digital laboratory ICT of CAS.

10. REFERENCES

[Bad93] Badler, N.I, Hollick, M.J. and Granieri, J.P. Real-time control of a virtual human using minimal sensors. PRESENCE,:pp: 82-86, 1993.

[Brown01] Brown, J., Sorkin, S, Bruyns, C. Latombe, J., Montgomery, K., and Stephanides, M. Real-time simulation of deformable objects: Tools and application. In Comp. Animation, 2001.

[Bin98] Bindiganavale, R. and Badler, N.I.. Motion abstraction and mapping with spatial constraints. in CAPTECH'98 con.proc, pp: 70-82, 1998.

[Ban95] Bandi, S. and Thalmann, D. An adaptive spatial subdivision of the object space for fast collision of animated rigid bodies. in Eurographics'95, pp: 259-270, 1995.

[Can88] Canny, J.F. The complexity of robot motion planning. MIT Press, 1988.

[Cho00] Choi, K. and Ko, H. On-line motion retargeting. Journal of Visualization and Computer Animation 11, 5, pp: 223-235, 2000.

[Gle98] Gleicher, M. Retargetting motion to new haracter. Com.of ACM, pp. 33-42, 1998.

[Gle01] Gleicher, M. Comparing contraint-based motion editing methods. Graphical Models.63,2, pp. 107-134, 2001.

[Got96a] Gottschalk, S. Separating axis theorem. Technical Report TR96-024, Department of Computer Science, UNC Chapel Hill, 1996

[Got96b] Gottschalk, S. Lin, M.C. and Manocha, D. OBBTree: A hierarchical structure for rapid interference detection. Conp. Graphics, pp: 171-180, 1996.

[Hua96] Huang, Z. Motion control for human animation. PhD thesis. EPFL-DI-LIG, 1996

[Liu05] Liu Li, Wang Zhao-qi, Zhu Deng-ming, Xia Shi-hong. An interactive motion retargeting method. in CASA'05 Conf.proc 2005.

[Lee99] Lee, J. and Shin, S.Y. A hierarchical approach to interactive motion editing for human-like figures. Com.of ACM, pp. 39-48, 1999.

[Mol97] Moller, T. A fast triangle-triangle intersection Test. Journal of graphics tools, pp: 25-30, 1997.

[Nat90] Naylor, B. Amanatides, J. and Thibault, W. "Merging bsp trees yield polyhedral modelling results", I Proc. of ACM Siggraph, , pp.: 115-241990.

[Neb00] Nebel, J-C. Realistic collision avoidance of upper limbs based on neuroscience models. Eurographics. 19(3). 2000.

[Qui94] Quinlan, S. Efficient distance computation between non-convex objects. in proceedings of international conference on robotics an automation. pp:3324-3329, 1994 .

[Sam89] Samet, H. Spatial data structures: quadtree, octrees and other Hierarchical Methods. Addison Wesley, 1989

[Shi01] Shin, H.J., Lee, J., Shin, S.Y., and Gleicher, M. Computer puppertry: an importance-based approach. ACM Transactions on Graphics 20, 2, pp. 67-94, 2001.

[Tak05] Tak, S. and Ko, H. A physically based motion retargeting filter. ACM Transactions on Graphics, Vol. 21, Issue 3, 2005

[Tom97]M.Tom. A fast triangle-triangle intersection test. Journal of Graphics Tools, 2(2):25-30, 1997

[Unu95] Munetoshi U. Ken A. and Ryozao T. Fourier principles for emotion-based human figure animation. conf.proc of ACM'95, pp: 91-96.1995

[Van97]Van, G. and Bergen, D. Efficient collision detection of complex deformable medels using AABB trees. J.of Graphics Tools, 2(4), pp: 1-13, 1997.

[Wan01] Wan, E.A. and Merwe, R.V.D. Kalman filter and neural netwroks. John Wiler&Sons. 2001.

[Zha94] .Zhao, J. and Badler, N.I. Interactive body awareness. Computer-aid design, 26(12), pp:861-867, 1994