Interactive Animation of Virtual Characters: Application to Virtual Kung-fu Fighting

Nicolas Pronost
Zhejiang Univ., State Key Lab. CAD&CG (China)
nicolas@cad.zju.edu.cn

Franck Multon
Univ. Rennes 2, IRISA - Bunraku (France)
Franck.Multon@irisa.fr

Qilei Li, Weidong Geng
Zhejiang Univ., State Key Lab. CAD&CG (China)
blighli@gmail.com, gengwd@zju.edu.cn

Richard Kulpa
Univ. Rennes 2, M2S Lab (France)
Richard.Kulpa@irisa.fr

Georges Dumont
ENS Cachan, IRISA - Bunraku (France)
Georges.Dumont@irisa.fr

Abstract

This paper aims at proposing a framework for animating virtual humans that can efficiently interact with real users in virtual reality (VR). If the user's order can be modeled as targets and commands, the system searches a database for the most convenient behavior. In order to avoid using a huge database that can deal with any kind of situation, we propose to associate this searching process to an adaptation module. Hence, even if the selected motion is not perfectly suited with the situation, it can be adapted in order to reach accurately the target specified by the user. This framework is illustrated with a kung-fu fighter example. Two people are involved in this example: the user and the supervisor. The user is displacing in the real environment while the position of his head is tracked in real-time thanks to reflective markers. The virtual opponent follows the displacement of the user to stay close to him. At any time, the supervisor can ask the virtual character to kick or punch the user. Our system automatically searches for the convenient motion in an average-size database (75 motions compared to hundreds of motions required for motion graphs) and adapts it to the current situation.

1. Introduction

Real-time motion synthesis of virtual characters is challenging in computer animation productions and interactive simulations in VR. Ideally, the virtual character would act as humans do in a variety of complex situations. First, the motion should be looking natural so that an observer could believe in it and an interactive user could act in a natural way. Second, the virtual character should calculate his motion very rapidly in order to avoid latencies that would compromise the quality of the animation and the interaction. Third, the motion of the virtual human should be adapted to the environment, to his morphology and to the real user's actions. Finally, as the virtual human cannot predict accurately what the user is about to do, it is impossible to use methods that require a complete knowledge of the sequence in order to calculate a convenient motion (such as displacement maps [7] do).

The method presented in this paper aims at animating virtual humans while solving the above problems. The naturalness of the motion is ensured by using a database composed of motion capture clips. A classical approach consists of defining huge databases, generally organized as motion graphs [9], to simulate realistic behaviors in numerous different situations. However, it requires huge precomputations and manual editing. Moreover, the resulting graph is so complex and deep that searching a path to solve all problems listed above requires computation times that do not allow for real-time rendering of the virtual avatars. This database is generally dedicated to one skeleton and may lead to some problems and additional computations when using several characters with various anthropometric properties.

For all these reasons, the approach introduced above is unusable for interactive applications such as VR. An alternative method consists of solving the constraints by using motion adaptation techniques applied on a unique motion clip. However, the resulting sequence may look unrealistic compared to using motion capture clips. Associating a database and motion adaptation techniques seems to be a very promising approach for solving such problems. This paper describes a method based on an average-size database of motion clips which are associated with a small set of parameters. In this database, we propose to search the most convenient motion according to the situation, even if it does not correspond accurately to all the constraints. This motion retrieval algorithm takes the morphology of the actor and the virtual human into account, in order to select the best candidate. The selected motion is then adapted in real-time to the size of the virtual character, to the environment and obeys the supervisor's orders.

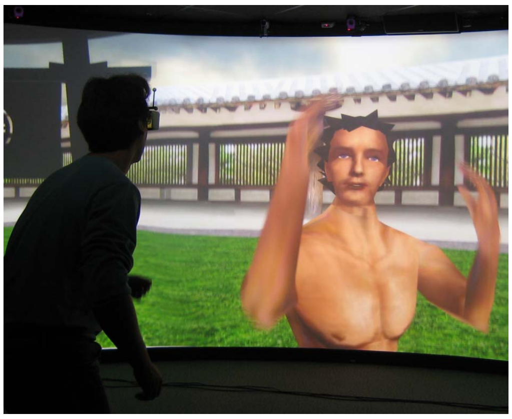

Figure 1. Interaction between a real subject and a virtual kung-fu fighter.

We illustrate our system with a virtual kung-fu fighter whose role is to kick and to punch (according to the supervisor's orders) a target placed on the head of a real user that displaces in the VR system area (3m x 3m area). This application illustrates all the constraints listed above as the animation engine needs to react to unpredictable events such as the target displacement and the supervisor's commands. The experiments are performed in a wide-screen immersive environment, as shown in figure 1.

2 Related works

Thanks to motion capture data, animating virtual humans is a very active field of research. Generally, methods are based on a complete knowledge of future events leading to using optimization techniques, such as space-time constraints optimization [5, 16] or displacement maps [7, 15]. The previous techniques adapt one particular motion to constraints. Inverse kinematics is then widely used to solve these constraints at specific times, by considering, for example, priorities between all of them [12]. All these approaches require a lot of computation time and is not compatible with interactive animation.

Motion graphs [9, 1, 13, 3] have been introduced in order to precompute extrapolated motions according to a wide set of captured data. All the possible transitions between postures of each motion are evaluated thanks to cost function transition metrics [20]. After this precomputation, the resulting graph can be used in real-time to find a path from the current state of the character to a desired one [17]. If the path does not exist in the database the closest state is selected. It has been applied to the guidance of a boxer that has to punch targets in space. To do so, hundreds of short motions are used to build the motion graph so that the space of the target is accurately sampled [14]. However, the large size of the graph leads to numerous computations in order to find a correct path. On the one hand, the larger the number of motions is, the higher the quality of the result is. On the other hand, the larger the number of motions is, the higher the computation time for searching into the graph is.

All the above techniques are generally linked to the use of a unique skeleton while many different characters can be used in interactive animation. Some authors have suggested that motions in a database can be interpolated according to anthropometric parameters in order to compute a plausible locomotion for a biped [19]. However, it cannot be used to adapt motion to additional constraints and to real-time orders. Let us imagine that the user is able to design his own avatar, he may want to use a database of motions to interact with the virtual world without needing to reprocess all the data. Motion retargetting [8] has been introduced to solve this problem. It assumes that the problem is mainly solved by satisfying kinematic constraints. Displacement maps can then be used to solve the constraints but cannot be used in interactive environments where unpredictable events can occur at any time depending on the user's actions. But a morphological-independent representation of motion can be used to simplify the process so that hundreds of characters can be animated in interactive environments [11]. This representation enables the solving of kinematic and kinetic constraints very efficiently, but is not able to decide which motion is the most convenient one according to the situation.

To solve this problem, a first idea could be to couple motion graphs to motion adaptation. Motion graphs could decide which sequences are suitable to deal with the situation while motion adaptation could adjust the resulting poses to the constraints. However, motion graphs are very efficient for a large database but require costly precomputation. The alternative proposed here is to use motion retrieval techniques [18, 22] together with motion adaptation [7] that lead to smaller databases and no precomputation. In our method, we carefully take the retargetting problem into account. Indeed, during the motion retrieval, for each motion clip, we have to take care that the stored motion and constraints are correctly compared to the constraints imposed to a character that is different from the actor.

Compared to motion graphs [14, 2], we use an average-size database because our motion adaptation allows reaching a wider set of constraints without requiring specific motion captures. We also ensure real-time motion retargetting that is impossible with theses approaches unless the database is filled-in with many occurrences of the same motion performed by several actors. Also, Gleicher's approach [8] would require a complete knowledge of the scenery which is impossible because the real opponent is always moving in an unpredictable way. Then, our paper demonstrates fast motion retrieval and real-time motion adaptation and retargetting. Our experiments demonstrate that it runs in real-time for applications where no prediction of the future is feasible.

In the remaining of this paper, we will focus on the kung-fu fighter example to illustrate our method. In this example, a virtual kung-fu fighter has to punch and kick a target placed over a real subject whose motions are captured in real-time. Hence, the real user can displace and crouch in the real world to avoid being touched by the virtual fighter. A supervisor uses a keyboard to order the kung-fu fighter to kick or punch the target. In that case, the fighter should react as fast as possible to reach the moving target. When no order is given, the fighter only moves to follow the displacements of the target so that he could reach it if new orders are given by the supervisor.

3. Methods

3.1. Overview

The animation engine comprises two main parts. The first one is a motion retrieval algorithm and the second one is a motion adaptation process. The role of the retrieval algorithm is to search a database to find the motion most suited to the situation. The role of the adaptation process is to adapt the selected motion to the given situation. This situation depends on the current pose of the fighter, on the position of the target according to the morphology of the virtual fighter and on the type of motion ordered by the supervisor. The overall process is depicted in figure 2.

In our experiment, the displacements of the user's head are captured in real-time with an AR-Tracking system composed of 5 infrared 60Hz cameras. The five-markers placed over the head are tracked by the motion capture system in real-time. With these data, the stereoscopic vision and the point of view of the camera are corrected in real-time. The target that the fighter should punch or kick is assumed to be the middle of these 5 reflective markers. A supervisor can interact at any moment thanks to a keyboard in order to command the virtual character to punch or kick the head of the user.

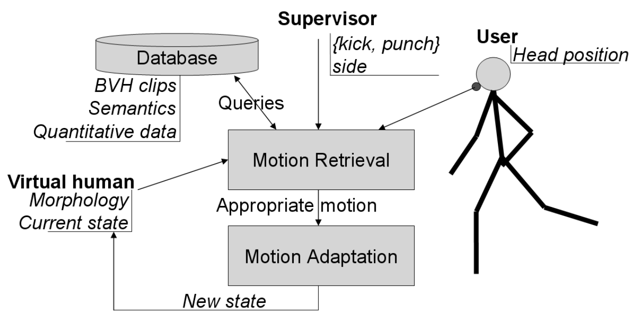

Figure 2. Overview.

The morphology of the virtual fighter is selected by the supervisor into a database composed of various skeletons. At any time, the supervisor can change its selection and immediately visualizes the result. Some kinematic constraints (such as avoiding foot-skating and reaching the target with either one feet or one hand) are solved on the fly. The resulting motion of the virtual fighter is displayed on a 9-meter wide screen for immediate feedback.

3.2. Design of the database

3.2.1 Motion annotation

Since the quality of the resulting motion mainly depends on the content of the database, we carefully selected a set of fighting motions which are suitable for the boxing scenario (like punching and kicking). This selection is preprocessed among a large database composed of motion capture data stored in the BVH standard format.

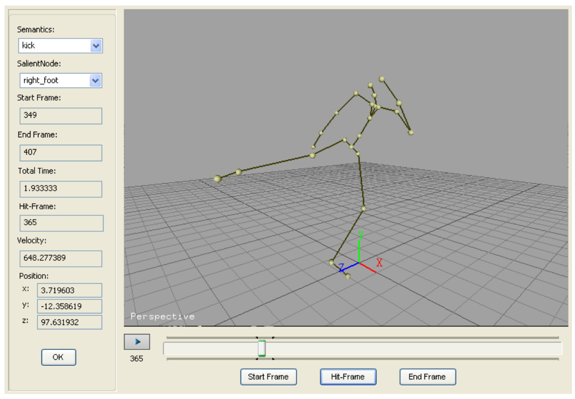
Semantics: kick
SalientNode: right_foot
Start Frame: 349
End Frame: 407
Total Time: 1.933333
Hit-Frame: 365
Velocity: 648.277389
Position:
X: 3.719603
Y: -12.358619
Z: 97.631932

Figure 3. The interface of annotation of the motions. Here, the semantics of this motion clip is defined as a *kick* with the *right foot*.

With toolkits developed in our lab (see figure 3), the user can extract the most interesting subsequences from long kung-fu fighting sequences (such as times during which the actor is punching or kicking). For each resulting subsequence, the user specifies the name of the hit segment (such as the right hand or the left foot) as well as the semantics (such as punching or kicking). The system automatically calculates the trajectory of the hit point. For each motion, the system also automatically calculates the bounding box of this trajectory (see figure 4) which is used to speed up the *Search* algorithm during the retrieval (see section 3.3).

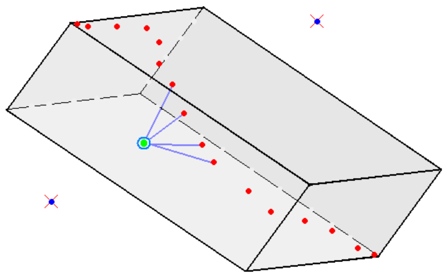

Figure 4. Bounding box of the hit point trajectory modeled with red spots (such as the trajectory of the wrist). The constraint c_r is modeled by the green spot.

The motion database is so composed of elements including the motion clip, the definition of the skeleton, the semantic information and some necessary quantitative data (such as the hit time and the position of the segment at this time).

3.2.2 Clustering

Since the whole process must be in real time, computation time for motion retrieval is a key issue. We propose to organize the database into clusters in a precomputation process. The goal is to group poses of several motions that are compatible from the constraint c_r point of view. Hence, the method should gather poses of motions if the position of the hit point is similar. To solve this problem, we choose the K-mean clustering algorithm. The goal of this algorithm is to gather poses of various motions into clusters in order to minimize the total intra-cluster variance.

In order to compare the similarity of the poses, a distance is introduced. Since the frames of the motion clips have variety of orientation and absolute positions on the horizontal plane, defining this distance is not obvious. A solution that we use, proposed by Kovar et al. [10] consists of minimizing the distance between the two poses by tuning the alignment parameters of the two motions.

Let us now consider the two modules involved in the real-time process.

3.3. Motion retrieval

As the simulated character is certainly different to the actor who performed the motions included in the database, some adjustments should be performed. Firstly, the position of the constraint should be scaled to fit the size of the character. For example, a constraint placed at 1-meter in front of the virtual fighter does not have the same influence if the latter is 2m or 1.5m tall. We divided the relative position of the constraint c (expressed in the root reference frame) by the size of the character $size_c$. The result is an adimensional constraint which is multiplied by the size of the actor $size_a$:

$$c_r = c \times \frac{size_a}{size_c} \qquad (1)$$

Secondly, the current pose of the virtual fighter $q(t)$ is retargetted to the actor belonging to the database, providing q_r. Consequently, the distance calculated between q_r and a pose in each clip is based on the same skeleton. Motion retargetting is performed thanks to the method described in section 3.4.1 and used for the *Motion Adaptation* module (see section 3.4). Hence, the inputs of the motion retrieval algorithm are c_r and q_r, as shown in figure 5.

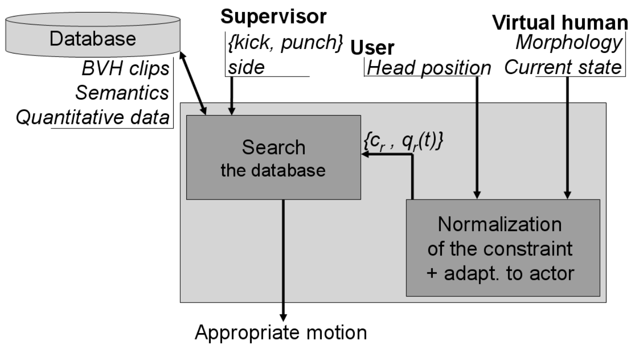

Figure 5. Overview of the motion retrieval process.

After clustering, the database is organized as a set of groups G associated with a typical pose (or centroid). The principle for the retrieval process then consists of selecting a cluster which best corresponds to the initial pose q_r and in associating a distance d_{q_r} to each pose of this cluster. To this end, the centroids are compared with the initial pose q_r.

The selection only takes the initial pose into account but solving constraint c_r is also an important problem. The resulting candidate poses need to be filtered in order to eliminate those which corresponding trajectory of the hit point is not compatible with c_r. This task is performed by eliminating motions associated with the bounding box that does not contain q_r. Each of the remaining poses are associated with the minimum distance d_{c_r} between the trajectory of the hit point and c_r. The pose (and the corresponding motion) that is finally selected by the algorithm is the one that minimizes the weighted sum d of the two distances d_{c_r} and d_{q_r}.

The motion associated with the lowest distance d is then rotated and translated to fit the initial pose q_r. The resulting motion is provided to the *Motion Adaptation* module.

3.4. Motion adaptation

The goal of this module is to adapt the selected motion M_s to the current situation. It involves: 1) synchronizing and blending M_s with the current active motion, 2) retargetting the motion to the animated character and 3) solving all the constraints (such as guiding the wrist to the target and correcting feet artifacts). This process is depicted in figure 6.

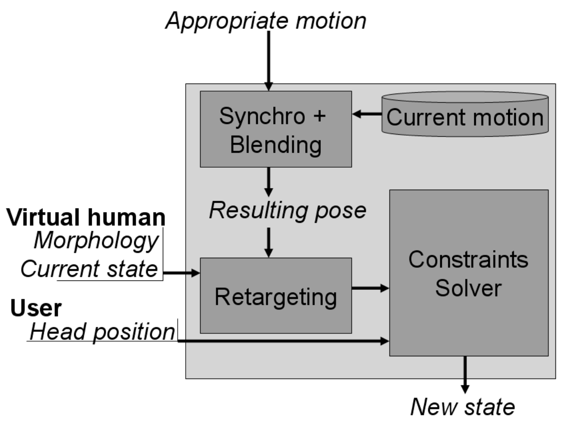

Figure 6. Overview of the motion adaptation process.

Retargetting mainly consists of multiplying adimensional data of the skeleton by the new dimensions of the character, and correcting the feet artifacts (see paragraph 3.4.1). The resulting pose is compatible with the skeleton and the ground but does not satisfy the other constraints. Paragraph 3.4.2 explains the method used to satisfy these constraints very rapidly.

3.4.1 Retargetting

Retargetting is a major issue of this paper because the morphology of the character is a key point in the selection of the motion and of its adaptation to the scenery. Retargetting is used at two different phases. Firstly, it adapts the current pose of the kung-fu fighter to the skeleton of the actor that participated in the motion capture session. Secondly, when a motion is retrieved from the database, it is supposed to adapt the sequence of poses to the morphology of the kung-fu fighter. To solve these problems, people usually apply joint angles from character#1 to character#2 while dealing with geometric constraints [8] in order to avoid foot-skating and other unwanted artifacts. However, this conversion is generally time-consuming and implies a complete knowledge of the constraints in advance which is not possible in interactive environments.

An alternative consists of using a morphological-independent representation of motion, as suggested in [11]. As motion retargetting generally involves solving geometric constraints, this representation is based on Cartesian data instead of joint angles. Hence, for example, the arm is modeled as a line between the shoulder and the wrist (denoted l_i for limb i) and an angle α_i to locate the elbow. Finally, only the relative position of the wrist in the shoulder reference frame and the angle are stored. The same way, all the intermediate body parts are modeled with relative position of the distal point in the proximal reference frame, divided by the total length.

3.4.2 Constraints solver

The method described above aims at adapting a pose to a character which dimensions are different from the actor. It mainly focuses on solving geometric constraints for the feet. However, other constraints should also be satisfied: mainly hitting the target with either a wrist or a foot. Let us recall here that the database is small compared to motion graphs or other classical approaches. Consequently, the selected motion is certainly not accurately adapted to the constraint and has to be modified. This modification must be fast to avoid increasing the size of the database which would lead to high computations to retrieve the convenient motion.

In our case, when the supervisor provides an order, the system should react in less than 1/30s. We consequently propose a frame-by-frame solver driven by continuous constraints. In the selected motion M, the duration between the current frame t_0 and the one corresponding to the contact with the target t_c is known. At t_c, the constraint corresponds to the vector $d(t_c)$ between $M(t_c)$ and the actual target X. At t_0, this distance is $d(t_0) = 0$. In between, $d(t)$ is an interpolation between $d(t_0)$ and $d(t_c)$.

At any time t, we thus adapt the joint angles of the character in order to make the wrist or the foot reach the new position $p(t) + d(t)$, where $p(t)$ is the initial position without adaptation. However, in many cases, the constraint is not reachable by only using the corresponding limb. In that case, we use an iterative algorithm derived from the Cyclic Coordinate Descent method [21].

The further the constraint is, the longer the time the algorithm requires to converge. If only the limb is involved, this method only requires $88\mu s$, as for retargetting. For reachable constraints that need to use three body parts (such as the limb, the clavicle and the torso), computation time is approximately $515\mu s$. For unreachable constraints, in the worst case, computation time is $1240\mu s$ which is still compatible with a 30Hz frame-rate, which is commonly used in computer animation.

3.5. Experimental set-up

Several virtual characters (see figure 7) with various sizes were designed using Avatar Studio (product of Canal NuMedia). At the beginning, the supervisor selects one of them, but he can change of fighter at any time by just hitting a key on the keyboard. Not only the global size but also the proportions between the body segments are different from one character to another.

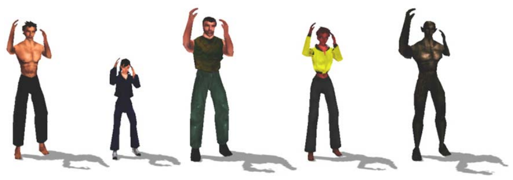

Figure 7. The five characters used for the demo. The dimensions and proportions of the body segments are significantly different one from each other.

We captured the motion of a kung-fu master using a "Motion Analysis" motion capture system. The kung-fu master was asked to perform:

- Various kicks for each side of the human body in order to have various styles and position in space (high kick, low kick, middle kick and another free-style kick),
- Various punches also for each side.

For the right side of the body, some punches and kicks are depicted in figure 8 when the strike occurs.

Each motion was stored using the standard BVH format including information for both skeleton and motion. Hence, 75 motion clips (limited to a unique hit per clip) were stored into the database.

The motion selected in the *Motion Retrieval* module is blended with the one currently performed by the character by continuously tuning a weight. The *Motion Adaptation* module is then applied to ensure to reach the target which is generally not the case if we apply the motion as stored. When a punch or a kick is currently performed, new orders from the supervisor are ignored. Hence, punches and kicks are only blended with displacements and the rest motion. As a consequence, for kicks, the *Motion Adaptation* module has to synchronize the foot-contacts of the two motions

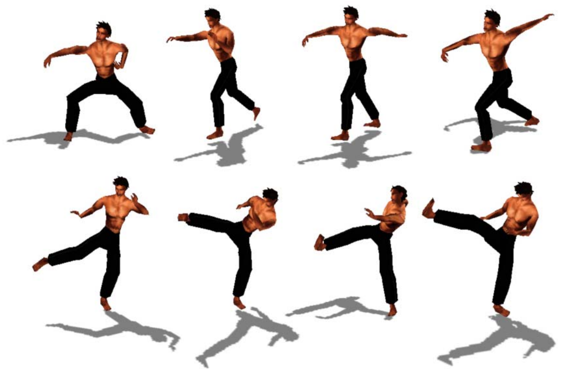

Figure 8. Four right punches and four right kicks available in the database (picture corresponding to the contact time).

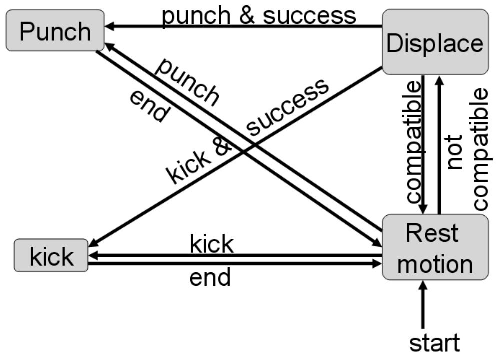

Figure 9. State machine describing the simple behavior of the kung-fu fighter.

(kicking and rest motion) in order to avoid some impossible cases. For example, kicking with the left leg while it's the support leg leads to unrealistic motions.

When the user enters the room, the reflective markers are placed over the head. The camera is translated to the position of the user's head. The height of the camera is set to the height of the user's eyes. The stereoscopic view is adapted to his displacements at each frame. At the beginning, the virtual fighter is placed 2-meters in front of the user and is animated with the rest motion. The virtual fighter starts moving to the user's direction while staying in an area compatible with kicking or punching the target. If he reaches this position, it again plays the rest motion unless the supervisor asks for a punch or a kick. The simple behavior of the kung-fu fighter is depicted in figure 9. Although this behavior and these interactions are not complex, they are sufficient to show the effectiveness of our method. More sophisticated methods of interactions [4, 6] could be considered as perspective.

Each time the supervisor selects a kick or a punch, the position of the head is stored (let $target$ be this position). After that, if the user moves, $target$ is not refreshed so that he can avoid the fighter's attacks. On the opposite, if the user does not move, we can check if the attack is successful or not.

4. Results

A subject was asked to participate in this study. He was equipped with stereo glasses (Crystal Eyes glasses for active stereovision). 5 reflective markers were attached to the glasses on a rigid body. The subject took position in the immersive environment in order to interact with virtual fighters (as shown in figure 1).

When the subject moves into the environment, the virtual fighter also moves to stay at a reachable distance for punching or kicking him. In figure 10, the head of the subject is modeled with a blue sphere and the motions (selected into the database) are adapted to reach the sphere's position.

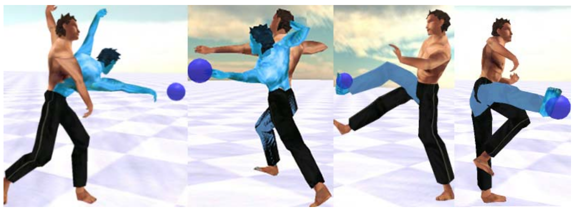

Figure 10. Adaptation of various punches and kicks selected into the database. The modified posture is depicted in light blue and the target is modeled as a blue sphere.

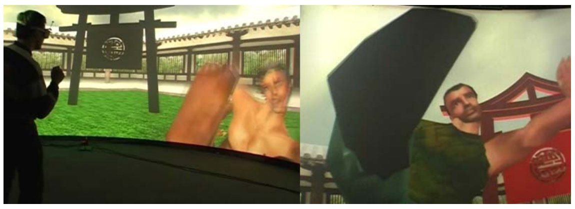

Figure 11. External view of the interaction between the user and the virtual fighter (Left). Same kind of scenery but viewed by the user (Right).

In the virtual environment, the parameters of the camera are modified in real-time according to the displacement of the head of the subject. Hence, for each punch and kick, the strike is always placed in the middle of the screen (corresponding to the position of the camera in the virtual world). As wished, this situation is correct for the user but may be perceived as false for people placed not exactly at the same position. Left part of figure 11 shows a right kick that reaches the middle of the screen, but the point of view of the real camera was far from the user. In that case, from the camera point of view, the kicks seems missing the target. The right part of this figure shows what the user actually see: the foot is placed in the middle of the screen and it looks credible for him.

We evaluated the performance of our method on a computer, with the following configuration : Pentium 4 3.6 GHz, 2Gb of memory. The *Search* algorithm was running on a separate thread to the remainder of the method. The latency between the supervisor keystroke and the beginning of the search is approximately 3ms. In the *Search* thread, the time from the beginning of the search algorithm and the return of the selected motion is almost 0.28s. As we used two separated threads, the system continues animating the virtual human during the selection of the motion. This motion is provided thanks to a BVH file written on the disk. This file must be loaded and converted into the morphological independent representation to retarget it to the virtual character. This total process spends about 40ms which is too much for 25Hz-animation. However, the main part of this time is spent reading the file from the disk. This time could be easily decreased if we use memory instead of hard disk. Let us recall here that motion retargetting and adaptation is very fast (only spends 88μs).

5. Conclusion

We have proposed a method for animating virtual humans that are able to interact with real users in real-time. The main issue of this paper is to associate two techniques used in computer animation for animating virtual humans: motion retrieval and motion adaptation. This association raises some challenges such as searching a database of motions captured on actors while the virtual humans and the constraints are different. Compared to motion graphs classically used to solve this problem, our method is able to deal with characters of various sizes in real-time. The difference of morphology is directly considered in the motion retrieval process which enables to find the most appropriate reaction to real-time constraints, such as kicking or punching moving targets.

Computation time is low for two main reasons. Firstly, the size of the database is low, especially compared to methods based on motion graphs. Consequently, searching this database is not so time consuming. Moreover, we proposed an organization of the database that speeds up the search process while taking the various constraints into account: dealing with the current pose of the character and with the target. The lack of data in database is compensated by the motion adaptation algorithm. Secondly, motion retargetting and adaptation is based on a morphological-independent representation that also speeds up the solving of kinematics constraints. The performance of the global experiment reported in this paper could be significantly improved by using memory instead of hard disk to transfer motions from the two threads (one for the *Search* algorithm and one for the remainder).

In this paper, we focused on the kung-fu fighting example but this work could be used for several other applications involving interactions between real and virtual humans in VR. Because of the low computation time, it could be used to enable collaborative works between virtual and real humans, as in virtual plants. However, the simple behavioral model based on finite state machines should be improved. We could imagine associating this module with behavioral models in order to deal with complex situations involving several virtual humans.

As perspective, we wish to increase the capabilities of virtual human by providing them with numerous different behaviors. The idea is to give more autonomy to virtual humans without dealing with huge databases that are impossible to manage in real-time. The same way, we wish to more accurately evaluate how computation time is affected when the database is filled-in with motions captured on numerous different actors. More generally, we wish to validate this work by comparing real behaviors to those obtained with our method. As we propose to work with a limited set of motions, the choice of these few motions is certainly very important. We will now experiment with methods for automatically selecting a minimum set of motions in a large database in order to produce realistic behaviors without large computation time.

Another interesting issue is to deal with bidirectional interaction in which the virtual human could react to kicks and punches of the user, for example. In a virtual plant, the user could give tools to virtual humans. This latter should be able to grasp this tool and use it for complex tasks in close interaction with the user.

References

[1] O. Arikan and D. Forsyth. Interactive motion generation from examples. *Proceedings of SIGGRAPH 2002*, 2002.

[2] B. Chiu, V. Zordan, and C. Wu. State-annotated motion graphs. In *Proceedings of the 2007 ACM Symposium on Virtual Reality Software and Technology*, pages 73–76, 2007.

[3] M. Choi, J. Lee, and S. Shin. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Transactions on Graphics*, 22(2):182–203, 2003.

[4] P. Chua, R. Crivella, B. Daly, N. Hu, R. Schaaf, D. Venturaa, T. Camill, J. Hodgins, and R. Pausch. Training for physical tasks in virtual environments: Tai chi. In *Proceedings of the IEEE Virtual Reality 2003*, pages 22–26, 2003.

[5] M. Cohen. Interactive spacetime control for animation. *Proceedings of ACM SIGGRAPH '92*, .26:293–302, July 1992. Chicago, Illinois.

[6] L. Emering, R. Boulic, and D. Thalmann. Interacting with virtual humans through body actions. *IEEE Computer Graphics and Applications*, 18(1), 1998.

[7] M. Gleicher. Motion editing with spacetime constraints. In *In Proceedings of Symposium on Interactive 3D Graphics*, pages 139–148, 1997.

[8] M. Gleicher. Retargetting motion to new characters. In *Proc. of ACM SIGGRAPH*, pages 33–42, July 1998.

[9] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. *ACM Transactions on Graphics*, 21(3):473–482, 2002.

[10] L. Kovar, M. Gleicher, and J. Schreiner. Footstake cleanup for motion capture. *ACM Siggraph Symposium on Computer Animation 2002*, 2002.

[11] R. Kulpa, F. Multon, and B. Arnaldi. Morphology-independent representation of motions for interactive human-like animation. *Computer Graphics Forum, Eurographics 2005 special issue*, 24(3):343–352, 2005.

[12] B. LeCallennec and R. Boulic. Interactive motion deformation with prioritized constraints. In D. P. R. Boulic, editor, *Proceedings of ACM/Eurographics SCA*, pages 163–171, Grenoble, France, august 2004.

[13] J. Lee, J. Chai, P. Reitsma, J. Hodgins, and N. Pollard. Interactive control of avatars animated with human motion data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 491–500. ACM Press, 2002.

[14] J. Lee and K. Lee. Precomputing avatar behavior from human motion data. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, pages 79–87, Grenoble, France, August 2004.

[15] J. Lee and S. Shin. A hierarchical approach to interactive motion editing for human-like figures. *Proceedings of ACM SIGGRAPH 99*, pages 39–48, Aug. 1999.

[16] Z. Liu, S. Gorther, and M. Cohen. Hierarchical spacetime control. *Proceedings of ACM SIGGRAPH '94*, pages 35–42, July 1994. Orlando, Floride.

[17] J. McCann and N. Pollard. Responsive characters from motion fragments. *ACM Trans. on Graphics (ACM SIGGRAPH 2007)*, 26(3), 2007.

[18] M. Muller, T. Rober, and M. Clausen. Efficient content-based retrieval of motion capture data. *ACM Transactions on Graphics*, 24(3):677–685, 2005.

[19] N. Pronost, G. Dumont, G. Berillon, and G. Nicolas. Morphological and stance interpolations in database for simulating bipedalism of virtual humans. *Visual Computer*, 22:4–13, 2006.

[20] J. Wang and B. Bodenheimer. An evaluation of a cost metric for selecting transitions between motion segments. *Eurographics/SIGGRAPH Symposium on Computer Animation 2003*, pages 232–238, 2003.

[21] L.-C. T. Wang and C. C. Chen. A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. *IEEE Trans. On Robotics and Applications*, 7(4):489–499, August 1991.

[22] T. Yu, X. Shen, Q. Li, and W. Geng. Motion retrieval based on movement notation language. *Computer Animation and Virtual Worlds*, 16(3-4):273–282, 2005.