A Motion-based User Interface for the Control of Virtual Humans Performing Sports

Xiubo Liang [1], Zhen Wang [1], Weidong Geng [1] and Franck Multon [2, 3]

[1] State Key Laboratory of CAD&CG, Zhejiang University
[2] IRISA - Bunraku, University of Rennes 1
[3] M2S Laboratory, University of Rennes 2

***Abstract*—Traditional human computer interfaces are not intuitive and natural for the choreography of human motions in the field of VR and video games. In this paper we present a novel approach to control virtual humans performing sports with a motion-based user interface. The process begins by asking the user to draw some gestures in the air with a Wii Remote. The system then recognizes the gestures with pre-trained hidden Markov models. Finally, the recognized gestures are employed to choreograph the simulated sport motions of a virtual human. The average recognition rate of the recognition algorithm is more than 90% on our test set of 20 gestures. Results on the interactive simulation of several kinds of sport motions are given to show the efficiency and interestingness of our system, which is easy-to-use especially for novice users.**

***Index Terms*—Virtual human, Motion recognition, Sport simulation, Interactive animation, Virtual reality**

I. INTRODUCTION

Motion-based user interface (MUI) is one kind of perceptual user interface (PUI), whose aim is to make human-computer interaction more like how people interact with each other and with the physical world [1]. As its name implies, MUI employs users' body motion directly to control different programs on computers. Compared with traditional user interfaces, e.g. keyboard, mouse, joystick etc, MUI provides an intuitive way for users to naturally interact with computers. The concept it embodies is to let computers adapt to human beings' interaction habits instead of asking people to comply with some fixed operation modes.

In the fields of virtual reality and computer animation, applications based on MUI are usually referred as performance animation, which means "what you perform and see is what you get as the final motion". Its core technology is how to reconstruct the expressive motions in real world and map them into the characters in virtual world [2]. Animation authoring by direct performance allows intuitive control of characters in the virtual world, which attracts great research interest in recent years. However, most of the existing performance animation systems [5-7] employ the complex vision-based motion capture system as their input device, which requires great professional skills to operate. As a result, they are not suitable for novice users with little experience. Furthermore, the vision-based systems often suffer from the problem of occlusion and can only work in a specified environment, which limits the use of such performance animation systems in everyday surroundings.

Motion sensors (including accelerometer, magnetometer, gyroscope etc) are considered as the ideal alternatives of vision-based motion capture devices for MUI in ordinary applications, because they get rid of the limitation of lighting and can be used in almost everywhere. As a typical application of MUI based on motion sensor, Nintendo Wii surges a revolution in human-computer interaction since its issuance. Many game corporations conform to the trend and release their own MUI console games gradually. Now days, this kind of games are very popular all over the world. However, one of the drawbacks of these systems is that the recognition algorithms can only detect human motions in a rough manner. For example, the swing of the arm and the slight twist of the wrist are both recognized as the swing action in the Wii tennis game. Apparently, such techniques cannot be employed directly to choreograph complex motions, such as sport motions.

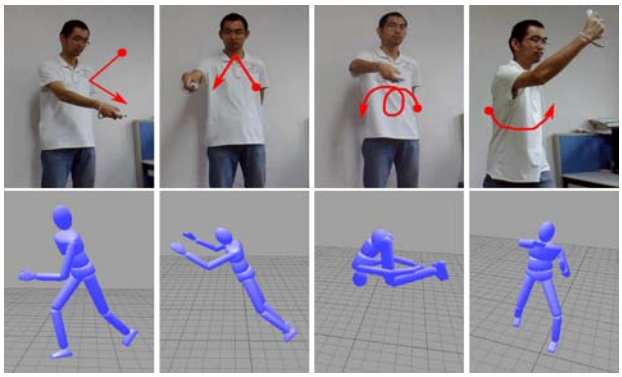

Fig. 1. A user controls the sport motions of a virtual human by his gestures. The motions from left to right are: running forward, jumping forward, front flip and swing punch.

In this paper, we developed a motion-based user interface to choreograph complex sport motions. In order to provide an easy-to-use system for novice users in everyday surroundings, we choose Wii Remote (with an accelerometer embedded) instead of the cumbersome motion capture system as our input device. The overall working process of our system is composed of three major steps. Firstly, the user performs some gestures in the air with Wii Remote. Secondly, our system identifies user's

Manuscript Received on June 10, 2011
E-mail: xiubo.liang@gmail.com

choreographing intents by recognizing the gestures with pre-trained hidden Markov models. Finally, the recognized gestures are employed to choreograph the simulated sport motions of a virtual human. Fig. 1 shows some examples to demonstrate how our system works.

II. RELATED WORK

The two main parts of this paper are a motion-based user interface and an interactive sport motion generation algorithm. In this section, we discuss the related work on these two aspects.

2.1 Motion-based user interfaces for character animation

Computer puppetry is an intuitive way to map the movements of a performer to an animated character in real-time. Its core technology is how to transferring the observations of the motion capture sensors to an animated character whose size and proportion may be different from the performer's [3]. Johnson et al. developed a "sympathetic" interface to control virtual characters in an iconic and intentional manner. The input device in their system is a plush doll embedded with wireless sensors. Over 400 participants successfully used the system and offered positive comments [4]. Dontcheva et al. presented an acting-based animation interface for creating and editing character animation at interactive speeds. Their system creates the connection between the actor and the character through a motion capture system and a set of reflective widgets, which free the animator from the confines of a mouse, a keyboard, and a limited workspace [5].

However, puppeteering requires a specialized input device and a significant amount of expertise, which limits its range of use. In order to overcome the shortcoming, researchers begin to explore performance animation interfaces which allow users to control virtual characters with their body motion directly. Chai and Hodgins employed two video cameras and a small set of retro-reflective markers on actor's body to create an easy-to-use system for character animation. The low-dimensional control signals are transformed into full-body motions by constructing a series of local models from a motion capture database [6]. Similarly, Ishigakis et al. developed their system with the same device. The advantage is that they integrate the prerecorded motions with both online performance and dynamic simulation. As a result, their system can synthesize motions with both physical realism and user's personal style [7].

Recently, as micromechanical technology became mature, the size and price of motion sensors drop dramatically. Many researchers begin to employ them to develop new animation interfaces. Slyper et al. created an action capture system with accelerometers, whose readings are continuously matched against existing motion capture data with a sliding window, and an avatar is then animated with the closest matched motion [8]. However, their simple matching metric cannot extract the semantics of motions which may be important in some interactive applications. Xiubo et al. solved this problem with their pattern recognition approach [2]. Shiratori and Hodgins proposed a Wiimote-based user interfaces for the control of a physically simulated character through the motion of user's arms, wrists, or legs [9]. But their system can only deal with simple swing motions of arms and legs, while our system can recognize much more gestures and generate more complex motions.

2.2 Responsive motion generation for animated characters

A simple way of generating new motions is firstly breaking the whole motion clips into small pieces and then rearranging them according to the constraints from users' input or virtual environments [10-12]. However, natural body configurations can't be created with such methods when the new environment is quite different from the environment when capturing. Physically-based method is considered as an alternative of the above methods to generate natural motions. Honda Motor Corporation successfully controlled the movements of real robots while keeping balance using zero-moment-point (ZMP) [13]. Macchietto et al. demonstrated a real-time motion simulation system capable of keeping balance while tracking a reference motion and responding to external perturbations by changing centers of pressure (COP), which seeks to control the linear and angular momentum of the character [14].

There are a lot of researches focusing on controllers. The benefits of such methods are effectiveness, robust, and easy transfer to new topology. Yin et al. developed robust controllers tracking motion trajectories with balance strategy involved, which also responded to external pushes [15]. Shiratori et al. created controllers for the trip recovery responses that occurred during walking [16]. Lee et al. presented a dynamic controller to synthesize under-actuated 3d full body biped locomotion, which takes motion capture reference data to reproduce realistic human locomotion through real-time physically-based based simulation [17].

More and more methods of this kind are inclined to hybrid with captured motion data. Zordan and Hodgins presented a system to synthesize hit and react motions based on captured data, which used inverse kinematic for hit and controlled forward dynamics for react [18]. Arikan et al. described an algorithm to animate characters being pushed by external forces by picking and modifying recorded motions [19]. Shapiro et al. introduced hybrid methods which usually run with kinematic models but can change to dynamics to react to external force, and change back once reaction is over. [20]. Komura et al. modified motions to maintain balance in response to external influences in animating reactions for locomotion bipeds [21]. Zordan et al. incorporated unexpected impacts into a motion capture-driven animation system through combination of physical simulation and best plausible re-entry motion clips [22].

III. OVERVIEW

In the field of virtual reality and video games, users usually prefer intuitive and natural user interfaces to interact with

virtual environments. Thanks to Wii Remote, our system can perceive the acceleration signals of users' natural gestures, which will be used to detect users' interaction intents. In order to generate the responsive animation, we captured a series of sport motions in advance and segmented them manually into distinct actions. As a result, the recognized gestures can be mapped to the separated actions. That is to say, the gestures are employed to choreograph sport motions. However, as the force properties of a gesture are different when the gesture is performed at different time, it will be uninteresting if we just replay the pre-captured motion. Therefore, we employ stunt motion simulation and responsive motion simulation techniques to generate new expressive motions according to the force properties embodied in the performed gestures. The schematic block diagram of our system is shown in Fig. 2.

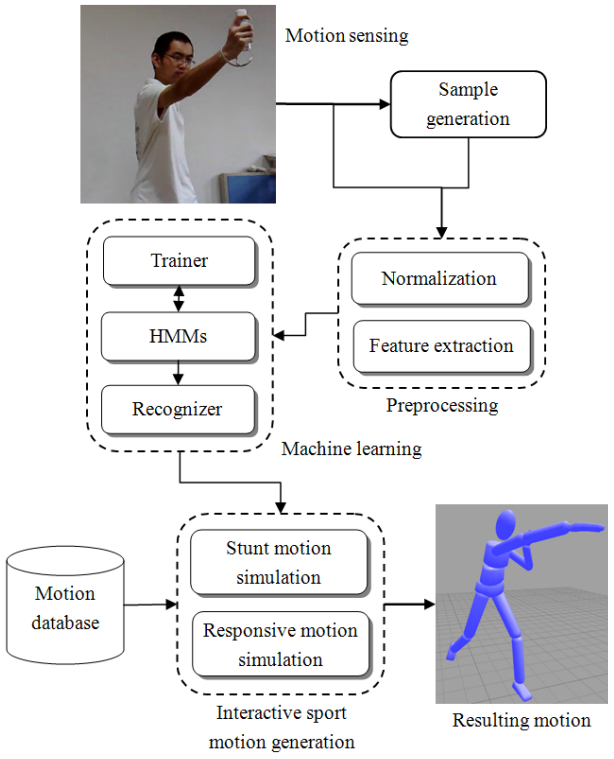

Fig. 2. The framework and pipeline of our system.

The major algorithmic steps of our system are explained below:

(1) Motion sensing. The real time acceleration signals are transferred from Wii Remote to our system through Bluetooth communication.

(2) Sample generation. For each gesture, the user only needs to perform it one time; the system will generate a series of new samples by adding noise.

(3) Preprocessing of samples. Resample the samples to a fix length; normalize the amplitude and extract the main features of them.

(4) Training and recognition. Offline, train the HMMs for each gesture with its training samples. Online, recognize the acceleration signals with the trained HMMs.

(5) Stunt motion simulation. Generate the stunt motions with motion exaggeration techniques based on the scaling of linear and angular velocity.

(6) Responsive motion simulation. Generate the responsive motions of a character when being hit with physical simulation techniques.

IV. GESTURE RECOGNITION

In order to bring good user experience, a gesture interaction system should meet three conditions: small training workload, high recognition rate and short response time. In this part, we describe the details of the gesture recognition algorithm to show why our system satisfies the above conditions.

4.1 Automatic generation of training samples

The traditional machine learning methods require users to gather a lot of samples for training. This process is quite boring and will greatly reduce users' interest in using the gesture interaction system. In this paper, we explored to find and validate the method of automatic generation of training samples. Kay pointed out that adding noise could improve the recognition ability of machine learning models under some conditions [23]. Therefore, we generate a series of new samples based on an original sample by adding noise. A motion sensing sample is essentially time sequence of 3D acceleration signals, which can be represented by a high-dimensional vector: $S_i = \{s_1, s_2, ..., s_J\}$, where $s_j \in R^3$ is the 3D acceleration data at frame j. Then a new sample can be generated by equation (1):

$$S_i^{'} = S_i + N_i \tag{1}$$

where $N_i = \{n_1, n_2, ..., n_J\}$, $n_j \in R^3$ is the 3D noise distortion. We employed two ways to generate the noise distortions:

- Uniform noise. To generate random values between $[-a, +a]$ satisfying uniform distribution, whose mathematical expectation $\mu = 0$ and variance $\sigma^2 = a^2/3$.
- Guassian noise. To generate random values satisfying the Guassian distribution with parameter b, whose mathematical expectation $\mu = 0$ and variance $\sigma^2 = b$.

The control parameter for the noise adding algorithm is Signal to Noise Ratio (SNR), which is the ratio of signal variance to noise variance. The best value of SNR could be acquired by a series of experiments (See Part Ⅵ for detailed discussion). Fig. 3 shows some examples of the sample generation algorithm.

4.2 Preprocessing of training samples

When an action is performed by different users or even if the action is performed by the same user at different time, its speed and size may vary greatly. Therefore, we should normalize the training samples. First, filter out the redundant data at the start and end of the sample. Then, interpolate between any two frames to get a c2 continuity cubic spline and resample it to obtain the new sample of specified length. At last, normalize

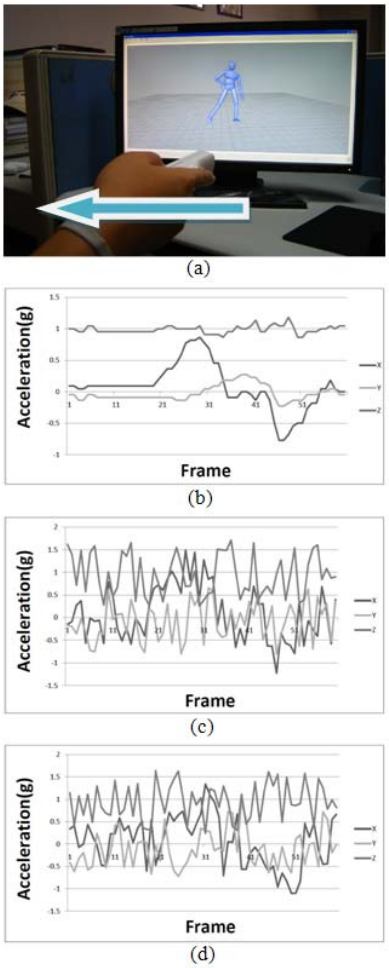

Fig. 3. Examples of the sample generation algorithm. (a) The gesture of controlling the character to walk backwards. (b) The original acceleration sample. (c) The generated sample with uniform noise distortion. (d) The generated sample with Gaussian noise distortion.

the amplitude of the sample with the normalization formula for random variables:

$$A^* = (A - \mu)/\sigma \tag{2}$$

Because of the multiplicity of the gestures, there is a great deal of irrelevant information in the samples and the main features are submerged in a great deal of useless data. Therefore, we employ the Principle Component Analysis (PCA) algorithm [24] to extract the key features of the samples, which are then fed to our machine learning model for training and recognition.

4.3 Gesture training and recognition

A Hidden Markov Model (HMM) consists of an underlying Markov chain having a finite number of states and a finite set of observation probabilities, one of which is associated to each state. The probability based mechanism of HMM makes it ideal for the processing of time series data. HMM was successfully used in the field of speech recognition [25]. Similar to speech, motion signals are also time sequence data. HMM should also have the ability to model our data very well. Therefore, we employ HMM as our recognition model.

There are two main forms of HMM: the continuous one and the discrete one. The former has more powerful modeling ability than the latter [26]. Therefore, we choose the continuous HMM for motion recognition, which can be represented as followings:

$$\lambda = \{S, \pi, A, \{b_i(o_t)\}\} \tag{3}$$

where S is the number of states; π is the initial probability of each state; $A = \{a_{ij}\}$ is the state transition matrix and each element $a_{ij} = P(q_{t+1} = j \mid q_t = i)$ is the probability of transition from state i to state j; $b_i(o_t)$ represents the observation probability density function (PDF) for state i at time t. In our system, we adopt the left-to-right topology of HMMs, each of which has 6 hidden states and each state contains a 3-component mixture of Gaussian. Supposing that we have K kinds of gestures and each of which has a corresponding HMM (represented by λ_k). Then, the recognition model is composed of a K-length vector of HMMs (See Fig. 4).

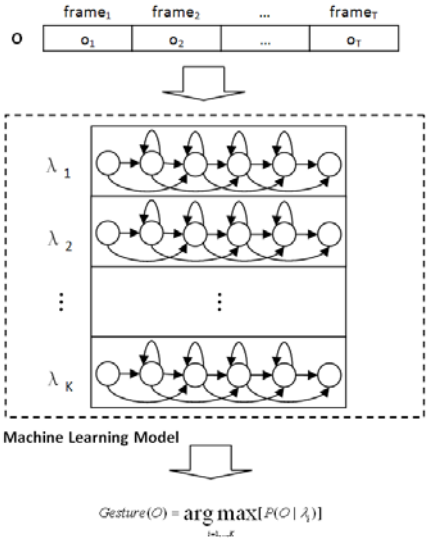

Fig. 4. Gesture recognition process based on HMMs.

Assuming that after preprocessing, a sample is represented by a feature sequence $O = (o_1 o_2 ... o_T)$. Once λ_k is trained, the probability of the feature sequence O can be computed with Forward-Backward algorithm [25]. The gesture with the maximum probability in the K-length vector is selected as the recognition result (see equation (4)).

$$Gesture(\boldsymbol{O}) = \arg\max_{i=1,...,m}[P(\boldsymbol{O} \mid \lambda_i)] \tag{4}$$

V. INTERACTIVE SPORT MOTION GENERATION

In this part, we present stunt motion simulation and responsive motion simulation to synthesize new sport motions based the captured data.

5.1 Stunt motion simulation

We propose a method to make changes to the velocity curve of each joint and achieve more faithful and smooth result. The only parameter for motion exaggeration in our system is acquired automatically from the force properties embodied in the acceleration signals. The more powerful movement the user performs, the more exaggerated the resulting motion will be. We use the average of the acceleration readings as the power parameter, denoted by B. The fade-in and fade-out principle is employed to process the velocity curve. Let $v_{ko}(t)$ denote the original velocity curve of the k^{th} joint, with the time parameter $t \in [0,T]$. Then the exaggerated velocity $v_{ke}(t)$ can be computed as $v_{ke}(t) = F(t)v_{ko}(t)$, and $F(t)$ is the function to exaggerate the original motion, which satisfies the following condition:

$$\int_0^T F(t)v_{ko}(t)dt = \int_0^T v_{ko}(t)dt = A \qquad (5)$$

where A is the displacement between the beginning position and the ending position. In general, $F(t)$ can be a linear, quadratic, or exponential function etc. We choose the quadratic form as our exaggeration function, as the quadratic function has a definite turning point. This could be intuitively utilized to control the amplitude parameter for the exaggeration. Therefore, $F(t) = at^2 + bt + c$, where c is the constant value of $F(t)$ at $t = 0$ which can be specified by the user and $B = (4ac - b^2)/4a$ is the parameter we get from the acceleration signals which corresponds to the maximum exaggeration of the motion. Then we can calculate a, b and the exaggerated velocity $v_{ke}(t)$. Finally, the trajectory of each joint is obtained by the integration of the exaggerated velocity. Fig. 5 gives an example of exaggerated action and its reactive motion.

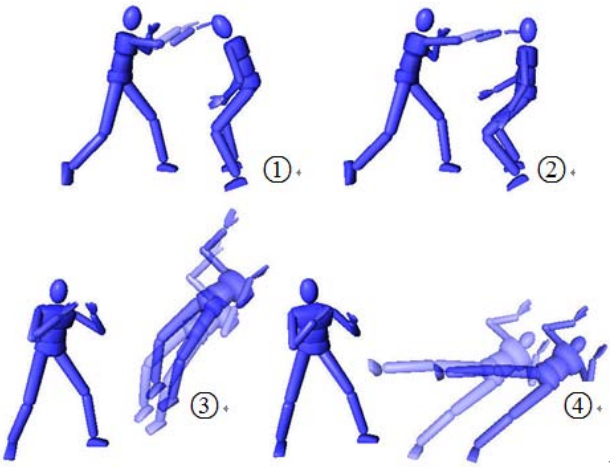

Fig. 5. Motion exaggeration based on velocity scaling. The original (light blue) and exaggerated (deep blue) hit and reactive motions: the stretched right arm of the hitter and the longer falling distance of the reactive character imply more powerful motions.

Angular velocity scaling is suitable to generate motions that rotate in air like flip and twist turn. Unlike the method presented by [27], which searches a fragment that can be repeated from a ballistic motion and copies it and blends the copy with the original motion, we modulate the rotation directly by scaling the angular velocity of the original motion. The rotations are described with quaternion. We first extract the angular velocity of the j^{th} joint by the following equation [28]:

$$\omega_j(i) = \frac{2\log(q_j^{-1}(i)q_j(i+1))}{h} \qquad (6)$$

where $\omega_j(i)$ and $q_j(i)$ are the angular velocity and rotation of the j^{th} joint at time i respectively, h is the time interval between i and $i+1$.

Then, a constant scaling factor is used to scale such original angular velocity to obtain the exaggerated one. After that, the new angular displacement could be calculated as follows:

$$q_j^*(n) = q(0)\prod_{i=0}^{n-1}\exp(\frac{h}{2}\omega_j^*(i)) \qquad (7)$$

where $\omega_j^*(i)$ is the exaggerated angular velocity of the j^{th} joint at time i, and $q_j^*(n)$ is the exaggerated angular displacement of the j^{th} joint at time n. Fig. 6 shows an example of stunt motion generated based on angular velocity scaling.

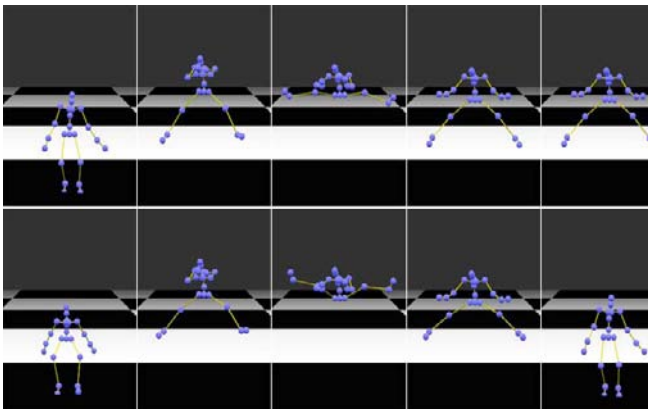

Fig. 6. Jump with legs expanding. The top row is the original motion and the bottom row is the exaggerated result.

5.2 Responsive motion simulation

We consider cases where no protective steps are taken. The whole responding process can be divided into two stages generally[29][30]. Experiments in Physiology show that it takes the central nervous system about 100-250ms to act since the moment of hit. People behave greatly different in the two stages, and we treat the simulations separately. We call response in the first stage passive response, and the next active response, intuitively.

5.2.1 Passive response

When part of body is hit, we dynamically simulate its moving way. We model the body as connected rigid bodies, but with springs adhere to neighbor bones, since muscle mostly react to weaken the force coming from outside. Mia's work[31] also gives the suggestion in agree with our comprehension. We formulate it as a resistance factor k_s which prevents the changes of body's original state θ_0 :

$$\tau = -k_s(\theta - \theta_0) \tag{7}$$

where θ is the current joint angle, θ_0 the original, and k_s the resistance factor.

The stiffness of each spring k_s varies, and generally speaking, ones at the end are smaller than those near the torso, for example spring at the elbow and the one at shoulder. And, in our implementation, we not only exert impulse on those collision parts, but break the total impulse into more portions, to influence adjacent body parts according to their distance from the contact area. Formally, we write: $\mathrm{M} = [m_1 m_2 \cdots m_k]$, where m_i is the component of M corresponding to i^{th} joint, $m_i = me^{-\omega d_i^2}$,where d_i is the distance between the i^{th} joint and the position on the character being pushed, m is the impulse amount and ω is a parameter that governs how fast the impulse declines. This is out of the same consideration with [32], and makes the hit more plausible.

5.2.2 Active response

The passive stage lasts a short time, and then the character will realize his own situation and make his active response, which is his real purpose. The passive response is relatively easy to simulate, but the active one is much harder considering thousands of different acts a real person may take. We simply assume the character wants to regain his initial pose.

Character's state, i.e. his pose, is represented by joints' configuration and the root's. Suppose current character's state is s_0 , and his initial pose s_d . We will take our character from state s_0 to s_d along a valid and nature trajectory.

An inertia-scaled PD-servo [18] is used in both steps, it is formulated as:

$$\tau = \mathrm{I}\left(\mathrm{k}_p(\theta_d - \theta) + k_d(\dot{\theta} - \dot{\theta}_d)\right) \tag{8}$$

where I is the inertial matrix of the outboard body for each joint, θ the current joint angle, θ_d the target, k_d and d_p the offset gain and damping gain we manually set. The $\dot{\theta}_d$ term helps in tracking the target trajectory with minimal time lag.

Setting θ_d the angles in s_d , we step the time, and thus get a simulated trajectory $\Theta = [s_0 s_1 ... s_M]$, where s_M is close enough to s_d .To make it better, we also add balance consideration following the algorithm suggested by Wooten and Hodigins [33]. An offset is calculated for in hips and ankles as:

$$\lambda = k_p(x_d - x) - k_d(\dot{x}) \tag{9}$$

where x is position of COM, x_d the target, and k_p , k_d the offset gain and damping terms for hips and ankles respectively. The offsets are then added to θ_d in equation (8) above. The Fig. 7 gives an example of the responsive motion simulation for hit and reaction.

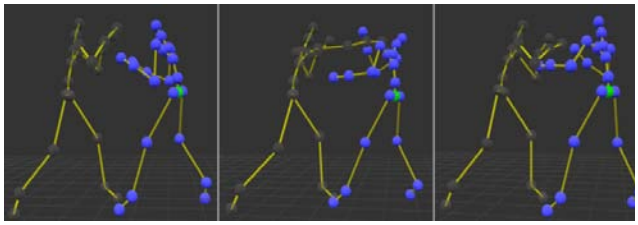

Fig. 7. The process of responsive motion simulation. The process from left to middle is passive response and the process from middle to right is active response.

VI. EXPERIMENTAL RESULTS

We developed a prototype system of motion-based control interface to choreograph sport motions with a Wii Remote, a Bluetooth device, a PC and a projector. The development kits are QT 4.5 and GHMM-0.7.0. To evaluate our system, we captured a series of sport motions, including walking, running, jumping, leaping, flipping, fighting etc. 20 gestures were designed as the choreographing orders (See Fig. 8).

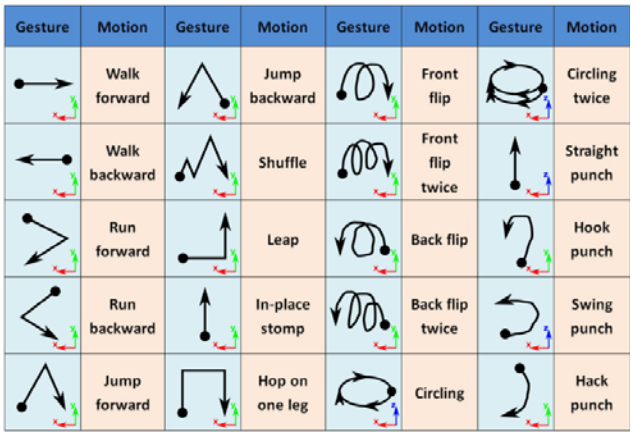

Fig. 8. The 20 gestures for the control of virtual humans.

To get the best value of SNR, we did a series of experiments on the recognition accuracy under different SNRs. It shows that result is best (over 95%) when SNR is about 4 (see Fig. 9). In the future, when using this module, we directly generate samples with SNR of value 4 which will insure good recognition accuracy

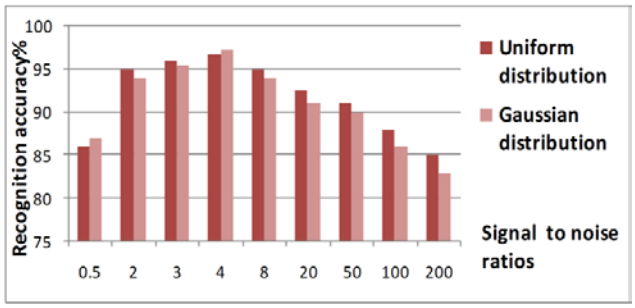

Fig. 9. The recognition accuracies for various noise levels of the two distributed noise.

We also did some experiments on the flexibility of the recognition algorithm. Acceleration samples were generated with the best SNR acquired in the last experiment. The average recognition accuracy is over 90% for various magnitudes of the same gesture (see Fig. 10). That means our algorithm is robust to the size of the gestures performing by users.

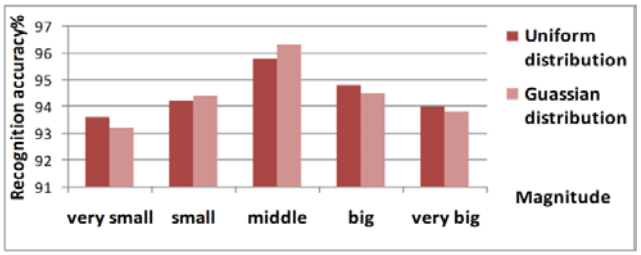

Fig. 10. The recognition accuracies for various magnitudes of the same gesture.

For sport motion generation, our system can create not only the truly simulated motions, but the vividly exaggerated ones based on pre-captured motions. That is to say our system can synthesize the artistic motions that go beyond the ability of humans and cannot be captured from actors. For example, when doing motion capture, our actor can only do the forward flip and twist turn with one round, while the user gives the gesture commands of "front flip twice" and "circling twice". In such cases, our system still works well in converting user's choreographing intent to the resulting motions due to the stunt motion simulation techniques. See Fig. 11 and Fig. 12 for two examples of the synthesized results from our system.

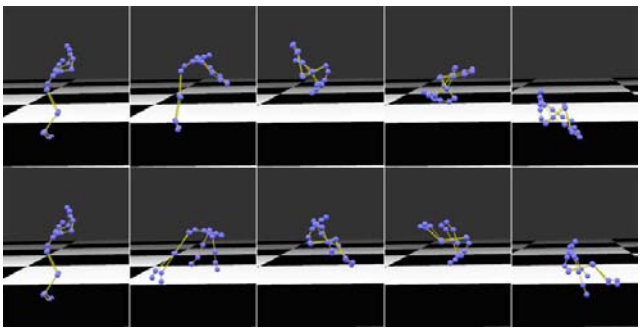

Fig. 11. Forward flip. The double forward somersault generated by our approach (bottom) based on the original single-flip motion (top).

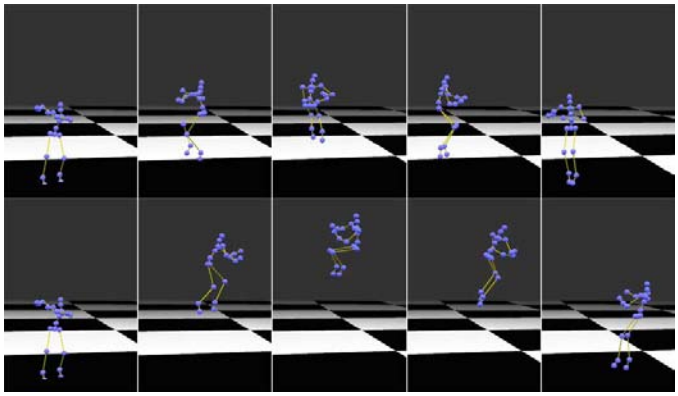

Fig. 12. Twist turn. The exaggerated motion (bottom) turns two rounds while the original motion (top) turns only one.

VII. CONCLUSION AND FUTURE WORK

In this paper, we present a natural user interface for the control of sport motions based on user's live performance. The subjective feedback from the users during our experiments shows that it's very attractive to them. The physical and artistic motion simulation techniques promote the interestingness of the system greatly.

The main technical contributions of the paper are as followings:

- We present a novel approach to choreographing sport motions by live performance captured with Wii Remote, which can be used in everyday surroundings.
- Thanks to the noise-adding mechanism, motion recognition in this system is user-independent and the online training is not required.
- Our system can generate new expressive motions by stunt motion simulation and responsive motion simulation techniques, which is quite attractive to novice users.

However, there are some limitations of the current method. As there is only an accelerometer in the Wii Remote, the acceleration readings are in the local coordinates of the device. If user holds Wii Remote in a different orientation when making the gestures, the recognition accuracy may decrease significantly. Furthermore, it's very difficult to segment the acceleration signals into different actions automatically. Therefore, we ask the user to push a button at the beginning of a gesture and release it at the end of the gesture. In the future, we will try to overcome the problem by integrating more motion sensors (such as gyroscope and magnetometer) to our system. Another drawback of our system is that the scaling-based motion exaggeration method may generate some artificial results in some cases. In the future, more physical constraints such as balance-control may be taken into consideration to ensure appropriate physical plausibility in the exaggerated motion.

It's our hope that this method and its future variations will play a significant role in creating expressive motions with a more intuitive and general way.

REFERENCES

[1] M. Turk. Perceptual user interfaces, *Communications of the ACM*, vol. 43, pp. 33–34, 2000.

[2] X. Liang, Q. Li, X. Zhang, S. Zhang, and W. Geng. Performance-driven motion choreographing with accelerometers, *Computer Animation and Virtual Worlds*, 20(2-3), pp. 89–99, 2009.

[3] H. J. Shin, J. Lee, S. Y. Shin and M. Gleicher. Computer puppetry: an importance-based approach. *ACM Transactions on Graphics*, 20(2), pp. 67 – 94, 2001.

[4] M. P. Johnson, A. Wilson, C. Kline, B. Blumberg and A. Bobick. Sympathetic interfaces: using a plush toy to direct synthetic characters, in *Proc. of CHI '99*, New York, pp. 152–158, 1999.

[5] M. Dontcheva, G. Yngve and Z. Popović. Layered acting for character animation, *ACM Trans. Graph.*, vol. 22, pp. 409–416, 2003.

[6] J. Chai and J. K. Hodgins. Performance animation from low-dimensional control signals, *ACM Trans. Graph.*, vol. 24, pp. 686–696, 2005.

[7] S. Ishigakis, T. White and V. B. Zordan and C. K. Liu. Performance-based control interface for character animation, *ACM Trans. Graph.*, vol. 28, pp. 61:1–61:8, 2009.

[8] R. Slyper, J. K. Hodgins. Action capture with accelerometers, in *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 193–199, 2008.

[9] T. Shiratori, J. K. Hodgins. Accelerometer-based user interfaces for the control of a physically simulated character, *ACM Trans. Graph.*, vol. 27, pp. 1–9, 2008.

[10] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs, *ACM Trans. Graph.*, 21(3), pp. 473–482, 2002.

[11] O. Arikan, and D. A. Forsyth. Interactive motion generation from examples, *ACM Trans. Graph.*, 21(3), pp. 483–490, 2002.

[12] Y. Li, T. Wang, and H. Shum. Motion texture: a two-level statistical model for character motion synthesis, *ACM Trans. Graph.*, 21(3), pp. 465–472, 2002.

[13] Honda Motor Co. Studies of leg/foot functions of robot, http://world.honda.com/asimo/, 2006.

[14] A. Macchietto, V. Zordan, and C.R. Shelton. Momentum control for balance, *ACM Trans. Graph.*, 28(3), article 80, 2009.

[15] K. Yin, K. Loken, and M. van de Panne. SIMBICON: simple biped locomotion control, *SIGGRAPH '07*, article 105, 2007.

[16] T. Shiratori, B. Coley, R. Cham, and J.K. Hodgins. Simulating balance recovery responses to trips based on biomechanical principles, in *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 37–46, 2009.

[17] Y. Lee, S. Kim, and J. Lee. Data-driven biped control, *SIGGRAPH '10*, article 129, 2010.

[18] V.B. Zordan, and J.K. Hodgins. Motion capture-driven simulations that hit and react, in *Proc. Of ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 89–96, 2002.

[19] O. Arikan, and D.A. Forsyth. Synthesizing constrained motions from examples, *ACM Trans. Graph.*, 21(3), pp. 483–490, 2002.

[20] A. Shapiro, F. Pighin, and P. Faloutsos. Hybrid control for interactive character animation, in *Proc. of the 11th Pacific Conference on Computer Graphics and Applications*, pp. 445–461, 2003.

[21] T. Komura, H. Leung, and J. Kuffner. Animating reactive motions for biped locomotion, in *Proc. of the ACM symposium on Virtual reality software and technology*, pp. 32–40, 2004.

[22] V.B. Zordan, A. Majkowska, B. Chiu, and M. Fast. Dynamic response for motion capture animation, *ACM Trans. Graph.*, 24(3), pp. 697–701, 2005.

[23] S. Kay. Can detectability be improved by adding noise? *IEEE Signal Processing letters*, 7(1), pp. 8-10, 2000.

[24] L. I. Smith. A tutorial on principal components analysis, *Cornell University*, USA, 2002.

[25] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition, in *Proc. of the IEEE*, 77(2), pp. 257–286, 1989.

[26] K. Sanna, K. Juha, K. Panu, and M. Jani. User independent gesture interaction for small handheld devices, *IJPRAI*, 20(4), pp. 505–524, 2006.

[27] A. Majkowska, P. Faloutsos. Flipping with physics: motion editing for acrobatics, in *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 35–44, 2007.

[28] K. Shoemake. Animating rotation with quaternion curves, in *Proc. of ACM SIGGRAPH*, pp. 245–254, 1985.

[29] A.P. Georgopoulos, J.F. Kalaska, and J.T. Massey. Spatial trajectories and reaction times of aimed movements: effects of practice, uncertainty, and change in target location, *Journal Of Neurophysiology*, 46(4), pp. 725-743, 1981.

[30] R.C. Miall, D.J. Weir, D.M. Wolpert, and J.F. Stein. Is the cerebellum a Smith predictor? *Journal Of Motor Behavior*, 25(3), pp. 203-216, 1993.

[31] R.C. Miall. Motor control, biological and theoretical, pp. 597-600, 1998.

[32] O. Arikan, D.A. Forsyth, and J.F. O'Brien. Pushing people around, Proceedings of the *2005 ACM SIGGRAPH*/Eurographics symposium on Computer animation, pp. 59-66, 2005.

[33] W.L. Wooten, and J.K. Hodgins. Simulating leaping, tumbling, landing and balancing humans, Proceedings of *IEEE International Conference on Robotics and Automation*, pp. 656-662, 2000.

Xiubo Liang is currently a PhD candidate at the Department of Computer Science, Zhejiang University, China. He received the BE degree from the Department of Computer Science, Zhejiang University, China, in 2006. His research interests include computer animation and game development, digital entertainment and natural interaction, virtual reality, and intelligent information processing.

Zhen Wang is currently a PhD candidate at the Department of Computer Science, Zhejiang University, China. He received the BE degree from the Department of Computer Science, Nanjing University of Science and Technology, China, in 2009. His research interests include character animation (physics based animation, especially), digital entertainment and natural interaction, and intelligent information processing.

Weidong Geng is currently a professor of College of Computer Science, Zhejiang University, China. He received the BSc degree from the Computer Science Department in Nanjing University, China, in 1989, and a M.Sc. degree from the Computer Science Department of National University of Defense Technology in 1992. In 1995, he received his PhD from the Computer Science and Engineering Department of Zhejiang University, China. From 1995 to 2000, he was in Zhejiang University, where he took charge of a number of projects about CAD/CG, and intelligent systems. He joined Fraunhofer Institute for Media Communication (former GMD.IMK), Germany, as a research scientist in 2000. In 2002, he worked in Multimedia Innovation Center, The Hong Kong Polytechnic University, Hong Kong. Since 2003, he works in State Key Laboratory of CAD&CG, and the research interests are computer aided design, computer animation, multi-modal interface, interactive media and digital entertainment.

Franck Multon is Professor in University Rennes2 in France. He is performing his research in biomechanics in M2S Lab and in character simulation in Bunraku/INRIA Rennes. His research interests are biomechanics, character simulation, and interaction between real and virtual humans. He defended his PhD in 1998 in INRIA Rennes on motion control of virtual humans. Since 1999 he was Assistant Professor in University Rennes2, has defended his "authorization to supervise research" in 2006 and has been hired as full Professor in 2008. He published 20 journal papers and 23 conference papers in several domains including computer animation, robotics, virtual reality, biomechanics, neurosciences and anthropology. He is member of ACM SIGGRAPH, IEEE, and the European Society of Biomechanics. He has reviewed papers in several conferences and journals in the above domains, and was member of the international program committee of ACM SIGGRAPH SCA, CASA, IEEE-VR, GRAPP and ACM SIGGRAPH VRST.