Simple feedforward control for responsive motion capture-driven simulations

Rubens F. Nunes[1,2], Creto A. Vidal[1], Joaquim B. Cavalcante-Neto[1], and Victor B. Zordan[2]

[1]CRAb – Federal University of Ceará, Brazil
[2]rgl – University of California, Riverside, USA
{rubens,cvidal,joaquim}@lia.ufc.br, vbz@cs.ucr.edu

Abstract. Combining physically based simulation and motion capture data for animation is becoming a popular alternative to large motion databases for rich character motion. In this paper, our focus is on adapting motion-captured sequences for character response to external perturbations. Our technique is similar to approaches presented in the literature, but we propose a novel, straightforward way of computing feedforward control. While alternatives such as inverse dynamics and feedback error learning (FEL) exist, they are more complicated and require offline processing in contrast to our method which uses an auxiliary dynamic simulation to compute feedforward torques. Our method is simple, general, efficient, and can be performed at runtime. These claims are demonstrated through various experimental results of simulated impacts.

1 Introduction

Demand for realistic animation of characters is increasing in game and entertainment industries, as well as in various other areas, such as biomechanics, robotics and rehabilitation; and important advances have occurred in recent decades [1]. Motion capture is a popular tool for generating realistic animation. However, adapting the data easily to unpredicted interactions in the environment without introducing artifacts is still a challenge. In contrast, dynamic simulation produces interactive responses to perturbations caused by the environment directly by applying collision forces to the model. Nevertheless, the difficulty of designing controllers for complex models, such as humanoids, which are capable of producing motions as realistic as the ones obtained by motion capture, limits its applicability.

A growing body of work proposes methods for combining these two techniques [2–7] (among others.) In 2002, Zordan and Hodgins [2] introduced feedback controllers for reactive characters capable of tracking captured motions. In their technique, proportional derivative (PD) feedback controllers calculate torques for tracking data while also allowing the simulations to react to external stimulus. One issue with this technique is that stiff feedback controllers are required for good tracking. According to [3], biological systems use feedforward as the predominant control input and only use low-gain feedback to correct small

deviations from the desired trajectory. They show that low-gain feedback control is capable of tracking the captured motions faithfully by adding a feedforward term to the control equation. Thus, during impacts, reactions are simulated with low stiffness without the need to update the gains (as done in [2]).

In this paper, our focus is on adapting motion-captured sequences for character response to external perturbations. Our solution is based on the framework proposed by [3] but we replace the method of computing the feedforward term with a simple and efficient method, which can be performed at runtime. Yin et al. [3] calculate their feedforward term by using inverse dynamics in a pre-processing stage. Our method calculates the feedforward term through use of high-gain feedback controllers applied to an 'auxiliary' simulation executed in parallel with the main simulation of the character. Unexpected impacts are considered only in the main simulation, which uses low-gain feedback controllers. Our method is effective for handling of small external disturbances that do not lead to changes in balance and does not depend on the balance strategy employed. In addition, we introduce several novel contributions throughout our investigation, e.g. an automatic method for scaling control gains and adding a purposeful delay to make more natural reactions as well as the introduction of a forward internal model (as described in [8]) to aid in correcting disturbances from *expected* impacts.

2 Related work

Several researchers have proposed methods for combining dynamics and motion capture. Shapiro et al. [4] created a framework based on [9] able to manage both dynamic and kinematic controllers. Zordan et al. [5] used dynamic simulation to produce transitions between captured motions in order to simulate reactions to impacts. Recent papers have built controllers for tracking captured motions while maintaining balance [6, 10, 7].

As in [2, 3], discussed in the introduction, some other works also proposed strategies to handle the specific problem addressed in this article. Pollard and Zordan [11] use an approach similar to that proposed by Yin et al. to control hand grasps. A feedforward term is pre-computed by inverse dynamics to offset the influence of the arm movement and gravity. In 2007, Yin et al. [6] proposed to calculate the feedforward term by using a variation of Feedback Error Learning (FEL) which is specific to cyclic movements such as walking, but did not address more general movements such as the non-cyclic ones we use in our example. In contrast to our approach, general FEL requires multiple passes to accumulate a stable feedback error value and it is therefore less conducive to online motion generation tasks. Yin et al. get around this by exploiting the repetition of cyclic motions. Da Silva et al. [7] calculate a feedforward term by formulating and solving a Quadratic Programming (QP) problem. Because of the complexity of the optimization, they update the feedforward term one to two orders of magnitude more slowly than the simulation dynamics. In contrast, our feedforward term is computed in lockstep with the simulation at interactive rates.

Some related works offer alternatives to selecting gains for torque calculations. In order to track the captured motions, Wrotek et al. [12] propose applying external feedback torques directly at the bodies, based on world-space error, instead of joint-space error. They report that with this technique the gains could be defined in an easier and more flexible way. However, the approach requires the gains to be updated using the same strategy presented in [2] in order to address reactions to impacts. Allen et al. [13] presented an automatic method to calculate the gains, by considering time information (synchronization) provided by the user. However, when an impact occurs, their approach to guarantee timing yields a character that changes its stiffness indirectly, based on the size of the disturbance. This effect is not readily observed in biological systems and thus we opt to maintain stiffness in lieu of guaranteeing the length of the resulting response during impact. Kry and Pai [14] presented a technique in which contact forces also are captured along with the movement. This information is used to estimate the tension on the joints. In contrast, our method does not require access to such information.

3 Approach overview

The proposed method in this paper estimates the feedforward control present in biological systems through an auxiliary dynamic simulation performed simultaneously with the main simulation. The auxiliary simulation uses high-gain feedback controllers in order to produce torques that make the virtual human track the captured motions faithfully. This process corresponds to the pre-processing done by biological systems when performing well-trained motions. In a sense, the auxiliary simulation acts as an internal model (see [8]) which perfectly matches the dynamics of the character. The feedforward terms used in the main simulation are exactly those torques produced in the auxiliary simulation. Thus, the torques applied to the simulated character during the main simulation consist of the sum of the feedforward terms with the torques produced by the low-gain feedback controllers used in the main simulation.

The general scheme shown in Figure 1 illustrates the steps of the proposed method: 1) Obtaining the model's desired state, $q_d(\mathrm{t})$, from the captured motion; 2) Calculation, using high-gain feedback controllers, of the torques, τ_{aux}, used in the auxiliary simulation to correct for the error between the current state, $q_{aux}(\mathrm{t})$, and the desired state, $q_d(\mathrm{t})$; 3) Calculation, using low-gain feedback controllers, of the torques, τ, used in the main simulation to correct for the error between the current state, $q(\mathrm{t})$, and the desired state, $q_d(\mathrm{t})$; 4) Application of the torques, τ_{aux}, to the model in the auxiliary simulation and sending τ_{aux} (feedforward torques, ff) to the main simulation; 5) Application of the sum $\tau + ff$ to the model in the main simulation; 6) Integration in time of the two simulations, considering the external perturbations, and updating the current states of the model.

Note that the high-gain feedback controllers only have access to the current state of the auxiliary simulation. Therefore, the resulting torques calculated by

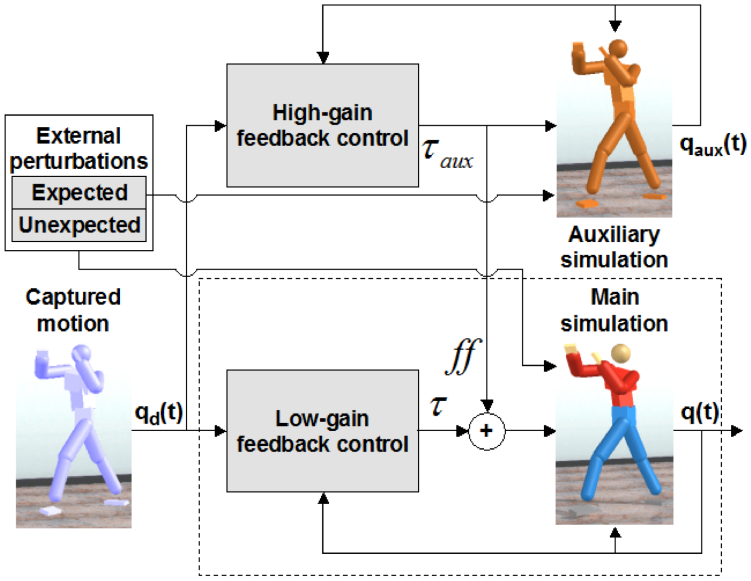

Fig. 1. Overview of the proposed method.

these controllers are *feedback* torques for the auxiliary simulation, but *feedforward* torques for the main simulation, since these controllers have no access to the state of the main simulation.

4 Feedforward and feedback torque calculations

Non-linear PD controllers are used both in the auxiliary and in the main simulation to correct for the error between the current state and the desired state of the model, through the following expression:

$$\tau = I\left[k_s f(\theta_e)\left(\theta_d - \theta\right) + k_d\left(\dot{\theta}_d - \dot{\theta}\right)\right] \tag{1}$$

where θ and θ_d are the current and desired joint angles; $\dot{\theta}$ and $\dot{\theta}_d$ are the current and desired joint velocities; k_s and k_d are the proportional (spring) and derivative (damper) gains; I is the inertial matrix of the outboard bodies (that is, the body chain affected by the joint) for each joint; $\theta_e = (\theta_d - \theta)$; and $f(\theta_e)$ is a non-linear factor defined as a function of θ_e. To limit the strength of the character, τ is capped during the dynamic simulation.

The values of θ_d used in both simulations are obtained directly from the captured motions. The values of $\dot{\theta}_d$ can be estimated by finite differences also from the captured data. However, in the auxiliary simulation, due to high-gain feedback, these values can be reduced to zero without reducing the quality with

which the auxiliary model tracks the captured motions. On the other hand, in the main simulation, we found that, due to low-gain feedback, the information about desired velocity is important. In our implementation, desired velocity information for the main simulation is obtained conveniently and directly from the auxiliary simulation, since the auxiliary model tracks the captured data faithfully. Therefore, the values of $\dot{\theta}_d$ from the auxiliary simulation are less sensitive to noise than finite difference estimates obtained with the captured data.

Selecting and scaling gain values. Using the inertia scaling, introduced by Zordan and Hodgins [2], allows only a pair of gains (k_s and k_d) to be specified for the whole body, eliminating the laborious process of specifying gain values for each joint. Allen et al. [13] describe the details of the calculation of this factor, which must be updated at each simulation step. With this scale, one pair of gains (k_s and k_d) has to be defined by the animator for each simulation (auxiliary and main). As in [13], we note the influence of the ratio $k_d : k_s$ on the simulated motions: 1) if $k_d : k_s$ is low, the model tends to oscillate in the neighborhood of the desired configuration; and 2) if $k_d : k_s$ is high, the model tends to be overdamped and move slowly, as if in water. Considering these observations, it is ideal that the ratio $k_d : k_s$ be high in the neighborhood of the desired configuration, to prevent oscillations; and low, when the model is far from the desired configuration, to allow it to return to tight tracking in a timely fashion. To accomplish these two goals simultaneously, we add the scalar $f(\theta_e)$ to the error calculation. Fattal and Lischinski [15] use non-linear PD controllers in which the ratio is updated by modifying k_d. We update the ratio $k_d : k_s$ by scaling the term k_s instead of k_d which intuitively has the desired effect of increasing the stiffness as the error increases, like when an impact occurs.

To scale k_s, we employ the term $f(\theta_e)$. In addition, we use this term to introduce a purposeful delay in the reaction. After receiving an impact, biological systems have a small delay in reacting due to the neural flow of information and delay of the synapses [3]. The delay is included in the definition of $f(\theta_e)$,

$$f(\theta_e) = \begin{cases} 1, & \text{if } |\theta_e| \leq \lambda \\ \dfrac{|\theta_e|}{\lambda}, & \text{if } |\theta_e| \leq M\lambda \\ M, & \text{otherwise} \end{cases} \tag{2}$$

where λ corresponds to the delay in the reaction and M is the maximum allowable value of the factor. θ_e is measured in radians, λ must be positive and M must be greater than 1. Analyzing the function, we have: $f(\theta_e) \geq 1$; $f(\theta_e)$ is linear, for $\lambda \leq |\theta_e| \leq M\lambda$; $|\theta_e|/\lambda = 1$, when $|\theta_e| = \lambda$; and $|\theta_e|/\lambda = M$, when $|\theta_e| = M\lambda$. In all tests, $\lambda = 0.1$ and $M = 5$.

5 Forward internal model for expected reactions

Based on concepts introduced by Kawato in [8], we propose an additional use of the auxiliary simulation. Kawato discussed the existence of two forms of internal

model, an *inverse internal model* which computes motor commands (in our case torques) and a *forward internal model* which predicts the effect of those motor commands based on an approximation of the dynamic state. We can draw interesting parallels between our auxiliary simulation and Kawato's internal models, but one insight that comes from this breakdown is that if the character knows about the dynamics of an impact, an internal forward model could be used for predicting and correcting for errors resulting from the disturbance.

Following this perspective, we introduce the notion of an *expected* impact which is "anticipated" by applying the collision to the auxiliary model. For an expected reaction, as when a person braces for a known impending impact, our auxiliary simulation accounts for the disturbance by simulating the impact and incorporating its correction into the feedforward terms, τ_{aux}. The effect in the main simulation is an automatic, anticipatory response of specifically the joints neighboring the impact (which see an increase in error in the auxiliary simulation.) Of course there are other forms of anticipation and trained reactions (for example, see [16]) but this simple adjustment allows us to automatically generate an important aspect of anticipation to expected interactions.

Without such a strategy, the problem of tuning the character (as done in [2]) poses a number of difficult issues: which joints should have their gains increased?; how much should the increase of each gain be?; and at which instant should the gain values return to the original low values? Consider, in practice, there could be situations where both unexpected and expected impacts occur simultaneously, and, it would be laborious to derive answers to these questions in order to handle all cases properly for those types of situations. In the method proposed here, the impacts expected by the model result in reactions that are automatically, intentionally more rigid in the region of the disturbance, without the need of modifying the gains.

6 Implementation

Our system uses the Open Dynamics Engine (ODE) [17] to perform the dynamic simulation and collision detection. Our humanoid model consists of 14 rigid bodies connected with ball joints, totaling 39 internal degrees of freedom plus 6 global degrees of freedom at the model's root (the pelvis). The animation obtained from the system achieves interactive rates using OpenGL and hardware rendering. On a 1.8 GHz PC with 512 MB of RAM and an NVIDIA video card FX 5200, the average frame rate is 10 fps, when rendering one frame at each thirtieth of a second of the simulation.

The character's perceived compliance and motion capture tracking quality both depend on the choice of gain values (k_s and k_d) assigned to the two simulations. Although manually defined, the choice of the small number of gain values (four total) is not difficult and need only be done once. The values are fixed in all test results in this paper. We found useful gain ratios relating the feedback and feedforward torque controllers as $k_{smain} = 0.05 * k_{saux}$ and $k_{dmain} = k_{daux}$.

Note, the gain values in the auxiliary simulation can be defined first since the gains in the main simulation do not affect that choice.

To maintain the model balance in the performed tests, external forces and torques are applied to the model's root (pelvis) and feet (when they contact the floor) to track the captured motions. These forces and torques are generated by high-gain feedback controllers, which are used in both simulations. Although this strategy is not physically correct, the proposed method does not depend on the balance control approach that is used and therefore allows for other strategies as well. Yin et al. [6] suggest a promising balance strategy for locomotion. However, maintaining balance during highly dynamic (athletic) motions like those presented in this paper, remains an open problem.

7 Results

In order to demonstrate the effectiveness of the proposed method, we perform several experiments. For our examples, we use fighting motions, such as punches and kicks motions. As the character tracks these captured motions, we create disturbances by throwing balls that collide with the character's body parts causing physically simulated impacts – the impacts yield corresponding external forces on the model. This type of interaction is very common in games. The accompanying videos allow a visual assessment of the movements and the method's quality.

In our first test, several objects are collided with the model while it tracks the captured motions. The model reacts to the external disturbances and returns to track the captured motion. Figure 2 shows an example where the model reacts to an impact on the arm and then another on the head. The auxiliary model does not receive the impacts and can be used as a reference for comparison. The original captured motion is shown in the accompanying videos. Our next result compares the capability of the low-gain feedback controllers to track the captured motions: without using the feedforward term (Figure 3a); and using the proposed feedforward term (Figure 3b), obtained from the auxiliary simulation, which uses high-gain feedback controllers. The feedforward term computed using the proposed method is effective and allows low-gain feedback controllers to track the captured motions faithfully, with quality similar to the tracking done by the high-gain feedback controllers used in the auxiliary simulation. Finally, we show that the proposed method can handle expected external disturbances easily. Figure 4a illustrates a situation where the model receives an unexpected impact on the back of the head and, simultaneously, an expected impact on the leg. The expected impact on the leg is also considered on the auxiliary model, as illustrated in the figure. In Figure 4b, the same two impacts are both considered unexpected for purposes of comparison. In the first case, the reaction on the leg occurs in a rigid way as desired, while in the second case the knee acts compliant.

The proposed method has presented effective and visually pleasing reactions to impacts in all parts of the model, even when more than one impact occur

simultaneously or several impacts occur in sequence. Moreover, the new method handled expected external disturbances, in a simple and effective manner.

8 Discussion and conclusions

This paper addresses the problem of simulating realistic reactions to external disturbances in captured motions, in an easy and automatic fashion. The proposed method employs an auxiliary simulation which uses high-gain feedback control to determine feedforward torque for the character (main) simulation. The controller allows the feedforward term to be obtained at runtime in a general way. The proposed method handles natural reactions to both unexpected and expected external disturbances.

Advantages of the proposed technique. Our method presents some absolute advantages and some advantages shared with certain previous methods. Advantages of our method can be listed as follows:

- Simple computation of the feedforward term. The computation done by the proposed method is exactly the same as the computation of the feedback torques but in an auxiliary model, thus very minimal implementation is required beyond feedback control alone.
- Tracking general trajectories. As opposed to inverse dynamics, feedback controllers produce appropriate torques for general trajectories, including trajectories that do not respect the laws of physics – such as transitions between motions [18, 9, 2, 6] and keyframe inputs (e.g. pose control) [19, 6, 13]. Using inverse dynamics on these trajectories can lead to unnatural large or undefined torque values.
- Online feedforward control. The proposed method is carried out at runtime. This feature allows the system to handle situations in which the trajectories cannot be predicted, such as trajectories synthesized by higher level controllers based on sensors or user input [9]. These trajectories cannot be pre-processed in order to obtain feedforward torques using offline techniques.
- Reactions to expected impacts. The proposed method handles expected reactions in a unified manner that is consistent with findings in motor control. By considering expected impacts in the auxiliary simulation the feedforward term automatically anticipates with a behavior visually similar to bracing for an impact.

There are several exciting areas of future work for this project. It would be interesting to test the proposed method on synthesized motion, for example, derived from simple transitions between captured motions or generated by traversing motion graphs [20]. Also, it would be interesting to define a general criterion for determining whether or not an impact should be expected (noticed in advance) by the model. Similarly, another direction would be to allow the model to have deliberate anticipatory reactions as in [16], to prevent impacts to vulnerable parts of the body.

References

1. Magnenat-Thalmann, N., Thalmann, D.: Virtual humans: thirty years of research, what next? The Visual Computer **21** (2005) 997–1015
2. Zordan, V.B., Hodgins, J.K.: Motion capture-driven simulations that hit and react. In: Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, San Antonio, Texas, USA (2002) 89–96
3. Yin, K., Cline, M.B., Pai, D.K.: Motion perturbation based on simple neuromotor control models. In: Proceedings of IEEE Pacific Conference on Computer Graphics and Applications, Canmore, Alberta, CAN (2003) 445–449
4. Shapiro, A., Pighin, F., Faloutsos, P.: Hybrid control for interactive character animation. In: Proceedings of IEEE Pacific Conference on Computer Graphics and Applications, Canmore, Alberta, CAN (2003) 455–461
5. Zordan, V.B., Majkowska, A., Chiu, B., Fast, M.: Dynamic response for motion capture animation. ACM Transactions on Graphics **24** (2005) 697–701
6. Yin, K., Loken, K., van de Panne, M.: Simbicon: Simple biped locomotion control. ACM Transactions on Graphics **26** (2007) article 105.
7. Da Silva, M., Abe, Y., Popovic, J.: Simulation of human motion data using short-horizon model-predictive control. Computer Graphics Forum **27** (2008)
8. Kawato, M.: Internal models for motor control and trajectory planning. Current Opinion in Neurobiology **9** (1999) 718–727
9. Faloutsos, P., van de Panne, M., Terzopoulos, D.: Composable controllers for physics-based character animation. In: Proceedings of ACM SIGGRAPH, Los Angeles, CA, USA (2001) 251–260
10. Sok, K.W., Kim, M., Lee, J.: Simulating biped behaviors from human motion data. ACM Transactions on Graphics **26** (2007) article 107.
11. Pollard, N.S., Zordan, V.B.: Physically based grasping control from example. In: Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, New York, NY, USA (2005) 311–318
12. Wrotek, P., Jenkins, O.C., McGuire, M.: Dynamo: Dynamic data-driven character control with adjustable balance. In: Proceedings of ACM SIGGRAPH Video Games Symposium, Boston, USA (2006) 61–70
13. Allen, B., Chu, D., Shapiro, A., Faloutsos, P.: On the beat! timing and tension for dynamic characters. In: Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, San Diego, CA, USA (2007) 239–247
14. Kry, P.G., Pai, D.K.: Interaction capture and synthesis. ACM Transactions on Graphics **25** (2006) 872–880
15. Fattal, R., Lischinski, D.: Pose controlled physically based motion. Computer Graphics Forum **25** (2006) 777–787
16. Zordan, V.B., Macchietto, A., Medina, J., Soriano, M., Wu, C., Metoyer, R., Rose, R.: Anticipation from example. In: Proceedings of ACM Virtual Reality Software and Technology, Newport Beach, CA, USA (2007) 81–84
17. ODE: Open dynamics engine (2008) http://www.ode.org/.
18. Wooten, W.L., Hodgins, J.K.: Simulation of leaping, tumbling, landing, and balancing humans. In: Proceedings of IEEE International Conference on Robotics and Automation, San Francisco, USA (2000) 656–662
19. Van de Panne, M.: Parameterized gait synthesis. IEEE Computer Graphics and Applications **16** (1996) 40–49
20. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. ACM Transactions on Graphics **21** (2002) 473–482

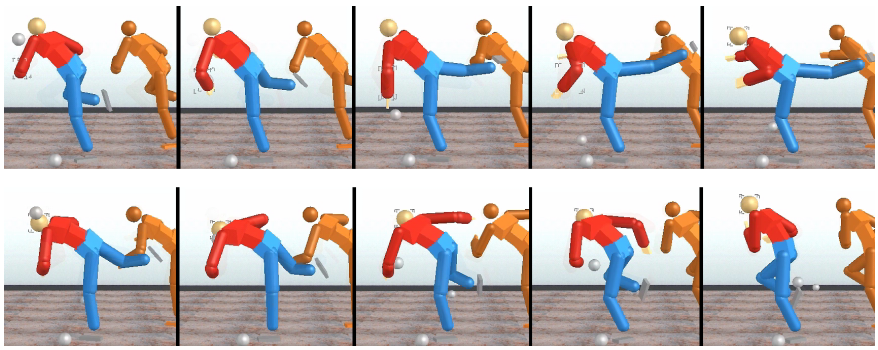

Fig. 2. The model is unexpectedly hit on the arm and after on the head, reacting to the impacts and returning to tracking the captured motion in a flexible and natural way.

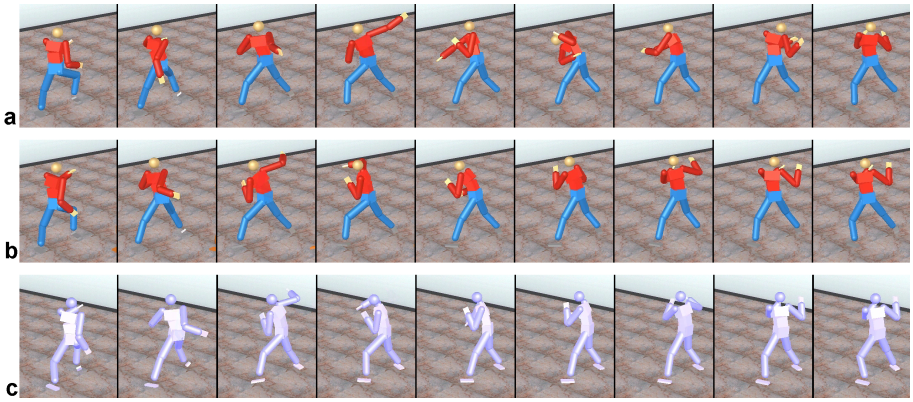

Fig. 3. (a) Low-gain feedback control without the feedforward term; (b) low-gain feedback control with the feedforward term; (c) original captured motion.

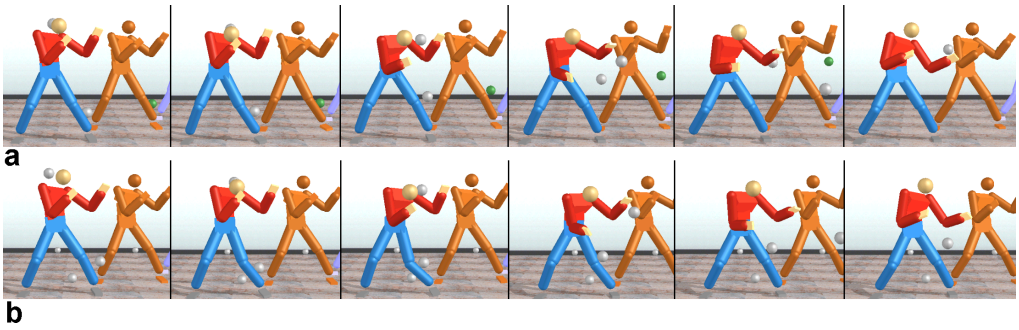

Fig. 4. The model receives, simultaneously, an unexpected impact behind the head and an (a – expected; b – unexpected) impact on the leg.