Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2007)
D. Metaxas and J. Popovic (Editors)

Flipping with Physics: Motion Editing for Acrobatics

Anna Majkowska and Petros Faloutsos

University of California, Los Angeles

Abstract

Complex acrobatic stunts, such as double or triple flips, can be performed only by highly skilled athletes. On the other hand, simpler tricks, such as single-flip jumps, are relatively easy to master. We present a method for creating complex, multi-flip ballistic motions from simple, single-flip jumps. Our approach also allows an animator to interact with the system by introducing modifications to a ballistic phase of a motion. Our method automatically adjusts motion trajectories, to assure physical validity of the motion after the modifications. The presented technique is efficient and produces physically valid results without resorting to computationally expensive optimization. To validate our approach we present the results of a study of user sensitivity to errors in angular momentum and take-off angle. The study shows that small changes of these parameters introduced by our method are not perceptible to a viewer.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computing Methodologies]: Computer GraphicsThree-Dimensional Graphics and Realism: Animation

1. Introduction

In 1905 Francis Gouleau of France achieved the first double backward somersault from a stand [V-F59]. Many modern acrobats have tried to repeat this feat and failed. On the other hand, a single backward somersault from a stand can be performed by even a beginning gymnast. In this paper we propose a technique that can recreate Gouleau's performance in a fraction of a second using a motion capture recording of a single backward flip (see Fig 4a). Our algorithm takes as an input a simple ballistic motion, such as a single somersault, and creates a more complex animation, such as a multiple somersault, by rotating and repeating fragments of motion, while maintaining physically-valid momentum profiles.

Our algorithm automatically applies simple motion editing operations to create superhuman motions or acrobatic stunts that would be difficult to record in a motion capture studio. Because it does not employ computationally expensive algorithms, our work can produce complex acrobatic motions in milliseconds after a pre-processing step of a few seconds. Therefore, our technique can be used to create multiple animated characters performing acrobatic motions in real-time. At the same time, by exploiting characteristics of ballistic motions, the proposed technique creates animations that satisfy the laws of linear and angular momentum conservation. Our algorithm can be applied to a variety of ballistic motions but is especially useful in motions with a significant amount of rotation during the flight phase, such as acrobatic flips and twists, skiing and snowboarding tricks, or martial arts moves.

Reusing motion capture data allows us to preserve the unique style of a performer more easily than with other methods, such as physical simulation or optimization techniques. While new optimization methods, for example the one proposed by Liu et al [LHP05] based on joint preferences, go a long way towards explaining biological aspects of motion style, there are still elements of style that cannot be easily characterized and are best captured by data-driven techniques. For example, acrobatic coach and human motion researcher Hartley D. Price compliments Russian athletes on their skills: "Russians were not content to merely perform the skill. They went one step farther and added a breathtaking flavor to the performance, leaving no doubt where the complete understanding was." [V-F59]. While highly skilled athletes can add certain flavors to their performance, beginners often make small errors which also make their motion unique and difficult to create with generative methods.

The challenge of reusing motion capture data lies in the fact that complex, multi-flip jumps differ considerably from their simple, single-flip counterparts. While the take-off and landing poses do not change significantly with a varying number of flips (see Section 5.3), in the multi-flip motions the rotation is faster and the flight phase longer. This results in the need for a more rapid build-up of momentum in the take-off phase and its quicker release during landing. To account for these differences, our technique adjusts motion timing and modifies flight trajectories. The trajectory modifications can introduce small changes in the take-off angle, while numerical approximations during the retiming step can cause small errors in angular momentum computations. However, we found that these changes are not perceptible in the motions generated with our method. To validate this result, we conducted a user study. We knew from previous research that most humans have difficulties noticing even significant discrepancies in angular momentum when observing motions of simple objects [ODGK03]. We were curious to find out if these results could be extended to human motion as well. Our results show that motions with small changes in angular momentum (both smooth and abrupt) as well as changes in take-off angle were not perceived as less correct than the original motions. In fact, surprisingly, sometimes they were perceived as more correct. The detailed results of our study are presented in Section 7.

In this paper we present the following contributions:

- an efficient technique for creating complex acrobatic motions with multiple flips from simple jumps, while obeying momentum conservation laws,
- a method for estimating mass distribution from motion capture data, which increases the accuracy of momentum computation,
- results of a study on human perception which measures sensitivity to errors in angular momentum and take-off angle.

2. Related Work

The generation of physically valid acrobatic motions has been a subject of intensive study. Control mechanisms have been successfully applied to create a wide range of of acrobatic motions. These include: somersaults [HWBO95, LvdPF00], high dives [WH00], kip stunts [FvdPT01, LvdPF00] as well as ski and snowboard tricks [ZvdP05]. While control algorithms create physically valid (though sometimes robotic-looking) motions, creation of controllers is still a difficult and time-consuming process.

Optimization techniques with physics constraints have become a prominent method for generating realistic highly dynamic motions [WK88, LGC94, PA00, LP02, FP03, SHP04]. The complexity and running time of the optimization methods have been significantly reduced in recent works: [LP02, FP03, SHP04]. However, even with these improvements, generating motions with optimization techniques typically takes a few minutes, which limits their applicability in real-time and interactive applications. Many works apply spacetime optimization to the problem of editing ballistic motions [WK88, SHP04, RGBC96, SP05, ALP06, MPS06]. These techniques allow animators to synthesize new, physically valid motions while preserving realism and style of motion. While many of these techniques could be applied to the problem of creating complex acrobatic motions, this paper shows that for such motions time-consuming optimization techniques are not necessary. High-effort ballistic motions can be synthesized and edited extremely efficiently using only basic motion editing operations.

The results of our perceptual study contribute to the area of human perception of motion. Reitsma and Pollard [RP03] studied human sensitivity to changes in vertical and horizontal acceleration and modifications of gravity. Safanova and Hodgins [SH05] observed that motion interpolation can cause large fluctuations in angular momentum which are often unnoticed if the resulting changes in angular velocities are small. O'Sullivan et al. [ODGK03] studied perception of angular distortions and velocity changes in post-collision trajectories for simple objects such as spheres, blocks and T-shapes. We contribute to this body of knowledge by measuring human sensitivity to distortions in angular momentum and take-off angle in human acrobatic motions.

Our method for estimating mass distribution is similar to the approach proposed in [BSP02]. In their work, Bhat and colleagues apply optimization to estimate motion and physical parameters of a rigid body in free flight from video, given the body's mass distribution as an input. In contrast, our technique uses 3D motion data to estimate not only motion parameters, but the mass distribution as well.

3. Overview

Our algorithm works in two stages: a pre-processing stage, which needs to be executed only once when a new motion is added to the system, and a run-time jump generation stage (see Figure 1). In the pre-processing stage, we first find a ballistic phase in the input motion. Next, taking advantage of linear momentum conservation, we estimate our character's mass distribution and inertia tensors for each body part. Finally, given the masses and the inertias, we compute the linear and angular momentum in each motion frame.

In the run-time jump generation stage (see Figure 2), we first employ a search algorithm to identify fragments of motion that can be rotated and joined together to create a more complex performance. Next, we retime the resulting motion to maintain continuity of momentum curves. As a last step, our algorithm repositions the character so its center of mass follows physically correct trajectories. We also provide functionality for user motion editing: an animator can, for example, change the position of the legs from tucked to straight, to create a more difficult jump. When legs are straightened, their distance from the center of mass increases and the jump rotation must slow down. Our algorithm can automatically adjust the edited motion to assure physical correctness. Similarly, different jumps can be joined together during the flight and adjusted to ensure correct momentum profiles. We de-

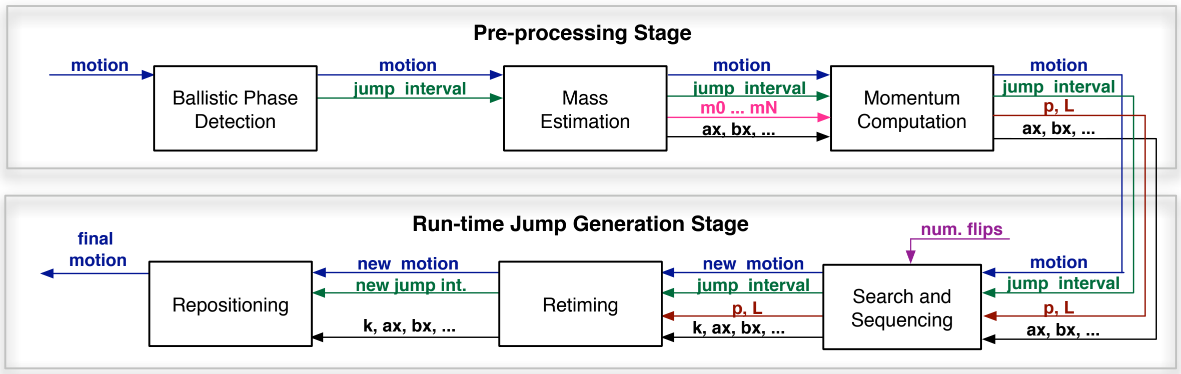

Figure 1: *Schematic diagram of our method's pre-processing stage (top) and the run-time jump generation stage (bottom).*

scribe the preprocessing steps of our algorithm in Section 4 and the run-time jump generation stage in Section 5.

4. Pre-processing Stage

In the pre-processing stage, our algorithm computes motion parameters such as mass distribution, linear and angular momentums, and jump duration (see Figure 1). These parameters need to be computed only once for each motion and can then be used multiple times in the run-time jump generation stage to create various ballistic motions.

4.1. Ballistic Phase Detection

In this step of our algorithm we determine where ballistic phases occur in the input motion. This information is first used to estimate the character's masses, then to run the search algorithm, and finally to create a new motion and retime it correctly. Our jump phase detection algorithm is based on the constraint method proposed by [LP02]. That is, we recognize that during the take-off and landing phases the character's feet or hands remain planted on the ground for a certain amount of time. We don't know the exact position of the ground in our motions (there can be multi-level environments). However, we can assume that intervals where at least one end effector is not moving significantly are good candidates for border frames of the ballistic motion.

Sometimes during the ballistic phase, one of the limbs can remain in the same place for significant time. Therefore, we add an additional condition, based on the pattern of linear momentum build-up and release. Typically, during take-off, linear momentum slightly drops and then suddenly increases. Similarly, during landing, there is first a sudden drop in momentum and then a slight increase. We used the momentum change conditions (first down, then up) together with the constraint method to find the longest ballistic interval in each motion. With this technique we managed to correctly find jump phases in all of our motion clips. Because in this step we do not yet know the character's estimated mass distribution, we use an average mass distribution of a human body (obtained from [DG67]) to compute the linear momentum. In our results this approximation did not affect the accuracy of the output.

4.2. Estimating Mass Distribution

Applying momentum conservation laws to motion capture data is a key element of our approach. The knowledge of a character's mass distribution, though typically not provided with motion capture data, is necessary for the computation of linear and angular momentums. Since we are working with motions with long ballistic phases, we have the advantage of being able to compute the mass distribution from the input motion alone. During a flight phase, the character's center of mass (COM) over time follows a line in the x and z dimensions and a parabola in the y (vertical) dimension due to gravity. Using this property we compute the masses of the body parts which minimize the distance between the character's COM calculated from the motion data and the correct COM trajectory, which is a line in x, z dimensions and a parabola in y.

To achieve this we formulate a simple linear least-squares optimization problem:

$$\min_{\substack{m_1,\ldots,m_n\\ a_x,b_x,a_z,b_z\\ a_y,b_y,c_y}} \left\| \left(\sum_{i=1}^{n} \frac{m_i}{M} pos_i(t) \right) - COM(t) \right\|^2 \quad \text{for } t = 0 \ldots T$$

with additional conditions that the sum of the masses is constant: $\sum_{i=1}^{n} m_i = M$ and that $m_i \geq 0$. In the above equation T is the duration of the flight phase, $pos_i(t)$ is the position of i-th body part at time t obtained from the motion data and $COM(t) = [a_x t + b_x, a_y t^2 + b_y t + c_y, a_z t + b_z]^T$ is the correct COM trajectory. Note that in our optimization we search both for mass distribution and the COM trajectory parameters a_x, b_x, a_y, b_y, c_y, a_z, b_z. We do not use the real-world gravity coefficient in our model because in motion capture data the character's joint lengths are often scaled compared to the real-world values. Using the results of our optimization we can compute the scaling factor as: $s = 2a_y/g$, where g is the gravity coefficient.

The $3(T+1)$ equations constructed from the above formula don't always have a unique solution. To minimize the number of optimized variables, we assume perfect symmetry between right and left limbs. Even with this modification, we found that the optimization still commonly produced solutions unrealistic for human body mass distribution. Although our optimization algorithm allowed us to specify a reasonable mass distribution $m_1^0, \ldots, m_n^0$ (obtained from [DG67]) as a starting point, the results of the optimization still suffered from overfitting. Overfitting occurred because the optimization algorithm attempted to "explain" inherent errors in the motion data. This caused large shifts in mass distribution, which resulted in only small improvements in the optimization results. To alleviate this problem, we added another equation to our problem formulation, which made the optimization algorithm favor solutions close to the initial distribution:

$$\min_{m_i} W \left\| m_i - m_i^0 \right\|^2 \qquad \text{for} \qquad i = 0 \ldots n$$

where W is a small constant. This formulation removes the overfitting problem as only large improvements in the optimization can sway the results significantly from the initial solution. Given the mass distribution and the skeleton proportions (obtained from mocap), we compute inertia tensors by approximating each body part with an ellipsoid of a constant density, as proposed in [LHP05]. We obtain typical densities of body parts from [DG67].

In our method we take advantage of the regular shape of COM trajectories during the flight phase due to the linear momentum conservation. We also tried to utilize the angular momentum conservation property to increase the number of constraints in the optimization problem. Unfortunately, computed derivatives contained too much noise, due to numerical errors in our motion data, to be useful in our optimization.

Our technique of mass estimation is useful for momentum computations as it reduces the deviation of linear momentum during the flight phase from the physically correct pattern. In our computations the reduction was 30% compared to the momentum values computed with the initial mass distribution. It is difficult to evaluate how well our method estimates the real-world mass distribution of a performer, since we do not know the actual masses of our performer's body parts (hence the need for estimation). However, a video recording of the motion capture session showed that the performer of our motions was tall. Tall males typically exhibit lower weight of limbs and higher weight of the trunk as the percentage of total weight, compared to a person of an average height [CM69]. The results of our optimization showed the same pattern in the estimated masses compared to the average mass distribution.

4.3. Momentum Computation

Given the mass distribution, computing the momentum is straightforward. The linear momentum $\mathbf{p}$ at time t can be computed as:

$$\mathbf{p}(t) = \sum_{i=1}^{N} m_i \mathbf{v}_i(t),$$

where m_i is the mass of i-th body part and $\mathbf{v}_i(t)$ is its velocity at time t. Angular momentum $\mathbf{L}$ for a system of rigid bodies can be computed as follows:

$$\mathbf{L}(t) = \sum_{i=1}^{N} \mathbf{R}_i(t)\mathbf{I}_i\mathbf{R}_i(t)^T + m_i(pos_i(t) - COM(t)) \times \mathbf{v}_i(t),$$

where $\mathbf{R}_i(t)$ is the rotation matrix for i-th body part in world coordinates, $pos_i(t)$ and $\mathbf{I}_i$ are the i-th body position and inertia matrix respectively, and $COM(t)$ is the character's center of mass at time t.

5. Run-Time Jump Generation Stage

In the run-time jump generation stage, our algorithm creates a complex multi-flip motion from an easier, single-flip jump using basic motion editing operations (see Figures 1 and 2). These operations and their impact on momentum are described in Section 5.1. The jump generation stage starts with a search, which finds and glues together fragments of motion to produce a longer sequence (Section 5.2). Next, the resulting motion is retimed to assure continuity of angular and linear momentums during take-off, flight, and landing phases (Section 5.3). Finally, the motion frames are repositioned to maintain correct COM trajectories in the ballistic phase (Section 5.4). Additionally, we provide functionality for user motion editing: an animator can modify character's poses during flight, and our algorithm then automatically adjusts the new motion to assure conservation of linear and angular momentums (Section 5.5).

5.1. Editing Operations and Their Impact on Momentum

In our algorithm, we employ three basic motion editing operations: repositioning, rotation and retiming by interpolation. In this section we analyze their impact on linear and angular momentums. We also show how these operations are used in our method to maintain the momentum conservation property during flight and assure continuity of momentum during take-off and landing.

Repositioning of the center of mass. Changes in the COM trajectory do not affect the character's angular momentum, which is computed with respect to the character's COM. We use this property in the last step of the run-time jump generation stage, when adjusting the COM trajectory to the longer flight phase.

Rotation around the angular momentum vector. Rotating a fragment of a ballistic motion around the axis defined by a

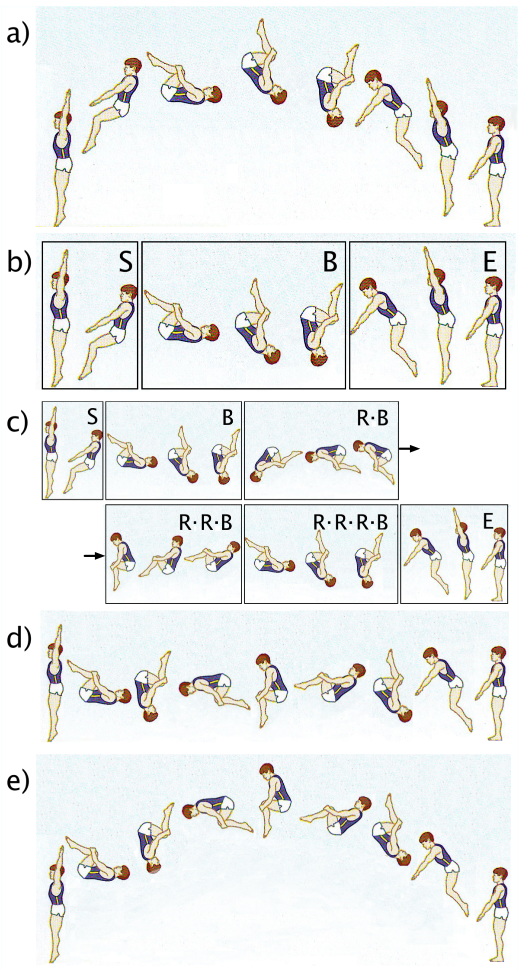

Figure 2: *Outline of the run-time jump generation stage of our algorithm. a) Input motion. b) First, the search algorithm finds a self-looping sequence B; root positions are ignored. c) Next, the sequence B is rotated around the angular momentum vector and repeated to obtain a double-flip. d) The resulting sequence is retimed to assure continuity of linear momentum during take-off and landing. e) In the final step, the character's COM is repositioned to follow correct trajectories under gravity.*

character's center of mass and the angular momentum vector will not affect the angular momentum within this motion fragment. Intuitively, angular momentum is a vector that remains constant in the given reference frame during the jump duration. If we rotate our reference frame around the axis defined by that vector and the COM, then all points on that axis will remain unchanged. More formally, angular momentum of a system around its COM can be defined as

$$\mathbf{L} = \sum_i m_i(\mathbf{r}_i \times \mathbf{v}_i) = \sum_i m_i(|\mathbf{r}_i||\mathbf{v}_i|\sin\theta_i\,\hat{\mathbf{n}}),$$

where m_i is the mass of a particle, $\mathbf{r}_i$ is the distance from the COM, $\mathbf{v}_i$ is the particle's velocity w.r.t. the COM, θ_i is the angle between $\mathbf{r}_i$ and $\mathbf{v}_i$, and $\hat{\mathbf{n}}$ is a unit vector perpendicular to $\mathbf{r}_i$ and $\mathbf{v}_i$. If we rotate the reference frame by some angle ϕ around the axis defined by the center of mass and the angular momentum vector the new angular momentum will be:

$$\begin{aligned}\mathbf{L}^{new} &= \sum_i m_i(|\mathbf{r}_i||\mathbf{v}_i|\sin\theta_i R(\phi)\hat{\mathbf{n}}) = R(\phi)\sum_i m_i(\mathbf{r}_i \times \mathbf{v}_i)\\ &= R(\phi)\cdot\mathbf{L} = \mathbf{L},\end{aligned}$$

because the axis of rotation is parallel to $\mathbf{L}$. We use this property in the sequencing step to increase the number of flips, while conserving angular momentum.

Motion retiming. Uniform retiming scales the linear and angular momentum linearly. A motion fragment with linear momentum $\mathbf{p}$ and angular momentum $\mathbf{L}$ retimed by a factor of n has momentums equal to $n\mathbf{p}$ and $n\mathbf{L}$. This is due to scaling of velocities during retiming:

$$\mathbf{v}_i^{new} = \frac{\mathrm{d}\,(r_i(nt))}{\mathrm{d}t} = \frac{\mathrm{d}\,(r_i(nt))}{\mathrm{d}(nt)}\,\frac{\mathrm{d}(nt)}{\mathrm{d}t} = n\mathbf{v}_i.$$

While this is true in the continuous case, applying this rule to motion capture data, where derivatives are computed numerically, causes errors. However, for $\frac{1}{2} \le n \le 2$, these errors in our motions are within 20% range and according to our study (Section 7) are not perceptible in the resulting motion. We use retiming to assure continuity of linear momentum during take-off and landing and to adjust momentums to motion changes made by an animator.

5.2. Search and Sequencing

The search algorithm attempts to find a fragment of ballistic motion, which can be repeated in the new sequence after rotation around the axis defined by the angular momentum vector $\mathbf{L}$ and the character's COM. In other words, it finds a sequence of frames $\mathbf{F_B} = f_m, f_{m+1}, \dots f_n$ and an angle ϕ, such that

$$R(f_m, \mathbf{L}, \phi) \approx f_n.$$

That is, frame f_m after rotation by ϕ around $\mathbf{L}$ is similar to f_n (we describe the similarity measure in Appendix A). We call the jump fragment $\mathbf{F_B}$ with this property a *self-looping sequence*. $\mathbf{F_B}$ is then progressively rotated and repeated multiple times to create a longer jump with more flips. For example, in Figure 2c the self-looping motion is repeated four times to create a double flip.

In other words, we use the initial motion sequence:

$$\mathbf{F} = [\mathbf{F_S},\ \mathbf{F_B},\ \mathbf{F_E}],$$

where $\mathbf{F_S} = f_1, \dots, f_{m-1}$ and $\mathbf{F_E} = f_{n+1}, \dots, f_{end}$, to create a new, longer sequence with more flips:

$$\mathbf{F}^{new} = [\mathbf{F_S},\ \mathbf{F_B},\ \mathbf{R_B}(\phi),\ \mathbf{R_B}(2\phi),\ \dots,\ \mathbf{R_B}(r\phi)],$$

where $\mathbf{R_B}(\phi)$ is the sequence $\mathbf{F_B}$ with all frames rotated by ϕ around $\mathbf{L}$. The parameter r denotes the number of repetitions of $\mathbf{F_B}$ in $\mathbf{F}^{new}$ and depends on the number of flips we want to obtain in the new motion. For example, in Figure 2 we construct a double flip, which involves the root rotation of 4π around $\mathbf{L}$. The rotation around $\mathbf{L}$ in $\mathbf{F_B}$ is $\phi \approx \frac{3\pi}{4}$ and rotations in $\mathbf{F_S}$ and $\mathbf{F_E}$ are close to $\frac{\pi}{2}$. Therefore we need to repeat the self-looping sequence four times. We compute r as $r = \lceil (2\pi \cdot \text{num\_flips} - \phi_S - \phi_E)/\phi \rceil$, where ϕ_S and ϕ_E are amounts of rotation around $\mathbf{L}$ in $\mathbf{F_S}$ and $\mathbf{F_E}$ respectively. Note that since we are rotating the self-looping sequence around its angular momentum vector, the angular momentum in $[\mathbf{F_B}, \mathbf{R_B}(\phi), \ldots, \mathbf{R_B}(r\phi)]$ will remain approximately constant when computed with respect to the COM.

The newly created sequence $\mathbf{F}^{new}$ contains a take-off phase and multiple rotations. Next, we perform a second search, to add the landing phase to $\mathbf{F}^{new}$. We find two frames $f \in [\mathbf{R_B}((r-1)\phi), \mathbf{R_B}(r\phi)]$ and $g \in \mathbf{F_E}$, such that $f \approx g$. The resulting sequence $\mathbf{F}^*$ will consist of $\mathbf{F_S}$, $[\mathbf{F_B}, \mathbf{R_B}(\phi), \ldots, \mathbf{R_B}(r\phi)]$ truncated to end at frame f, and $\mathbf{F_E}$, truncated to start at frame g. As a last step, to eliminate small discontinuities in transitions between motion fragments, we smooth joint angles by slerping over small windows of frames.

If the ballistic motion does not contain self-looping sequences (there are no similar frames f_m and f_n), our algorithm can still be applied. The transition between any two chosen frames of ballistic motion can be created by an animator or automatically by smoothing. Although during the transition the angular momentum will not be constant, we can adjust it by applying non-uniform motion retiming as described in Section 5.5. The same algorithm can be easily modified to join two different motions during the jump phase. As long as the two motions have the same angular momentum direction, the magnitudes can be adjusted by retiming.

5.3. Retiming

In the search step we constructed a complex motion with multiple rotations and a longer ballistic phase compared to the original motion. Differences in the flight phase duration and COM trajectories affect the momentum build-up during take-off and its release during landing. One of challenges here is to adapt the take-off and landing phases to reflect the changes in the flight phase and to assure continuity of linear and angular momentums.

For low-effort jumps, changes in momentum build-up are associated with significant changes in take-off poses. For example, a character might bend the knees more to achieve a longer or higher jump. Surprisingly perhaps, the same is not true for the high-effort ballistic motions, such as acrobatic flips or long jumps, where the athletes are operating close to the limit of their physical abilities. Research suggests that the approach speed is the main factor affecting the performance in high-effort motions, while take-off poses vary little with changing performance levels. Bridgett and Linthorne [BL06] evaluate long jumps performed by professional athletes with maximum effort and conclude that variations in run up speed account for 96% of the variation in jump distance. Seyfarth, Blickhan and van Leeuwen [SBvL00] report that the optimal jumping technique defined by leg angles is not affected by changes in approach speed or muscle design. In acrobatic motion research, King and Yeadon [KY03, KY04] present data on joint angles during take-off for a single and double backflip from a round-off (see Table 1). Differences in joint angles are negligible: it would take less than 0.02 second to transition between these two take-offs, due to high angular velocities of body parts.

Joint	Single flip (deg) a	Double flip (deg) b	Ang. vel. (deg/sec) ω	Time (sec) $\lvert a-b \rvert/\omega$
ankle	125	125	821	0.00
knee	174	168	278	0.02
hip	175	202	715	0.02
trunk	78	99	931	0.02
shoulder	154	153	157	0.01

Table 1: *Comparison of take-off poses for a single and double backflip. Differences between the joint angles in the two take-offs are negligible; with given angular velocities, it takes less than a duration of one frame to transition between them. Data adapted from [KY03] and [KY04].*

While there is less scientific data on the landing phase, our conversations with gymnastic coaches [Bel, Nel] reveal that a perfectly executed double flip will have a very similar landing to a correctly carried out single flip, but will be performed faster.

Since in high-effort motions take-off and landing do not vary significantly with performance level, we can achieve natural-looking motions and eliminate momentum discontinuities by simply retiming motion fragments so that they are performed faster. We retime the take-off, flight and landing phases by the same factor k. Retiming will scale the angular momentum by a factor of k in all phases of motion. During flight, linear momentum will be adjusted by repositioning of the character's COM, but retiming also increases linear momentum k times during take-off and landing, and reduces the duration of the ballistic phase by a factor of k. In Appendix B we show that there exists a single scaling factor k^* which assures the continuity of the linear momentum curve. In practice however, small discontinuities in linear momentum are not perceptible to the human eye. Therefore we can adjust k to obtain different jump heights and rotation speeds. In our results we typically chose $k \approx 0.7k^*$, as we found higher jumps with slower rotations to be more aesthetically pleasing.

5.4. Repositioning

In this step we adjust the character's center of mass in each flight frame of the newly constructed motion, so that it follows physically valid trajectories. Computing the COM position along x and z axes is simple, given values a_x, b_x, a_z and b_z computed in the mass estimation step (Section 4.2) and retiming factor k chosen in the retiming step (Section 5.3):

$$\begin{aligned} COM_x^{new}(t) &= k \cdot a_x t + b_x \\ COM_z^{new}(t) &= k \cdot a_z t + b_z \\ t &= 0 \ldots T^{new}, \end{aligned}$$

where T^{new} is the duration of the ballistic phase in the new motion.We also offset the COM trajectory after the jump to maintain the continuity of motion:

$$\begin{aligned} \text{offset}_x &= COM_x^{new}(T^{new}) - COM_x(T) \\ \text{offset}_z &= COM_z^{new}(T^{new}) - COM_z(T). \end{aligned}$$

In the case of the y (vertical) axis, we need to modify the parabolic trajectory by scaling the initial velocity b_y at $t = 0$:

$$COM_y^{new}(t) = a_y t^2 + k \cdot b_y t + c_y \quad \text{for} \quad t = 0 \ldots T^{new}.$$

The change in initial velocity causes a (typically small) change in the take-off angle. However, our study indicates (Section 7) that such small changes are not perceptible.

Using the above formulas, we can also make non-trivial modifications to the character's trajectory by modifying c_y and adjusting the scaling factor k (see Appendix B). For example in Figure 4c we changed the height from which the character takes off. In the original motion the take-off and landing were at the same level.

5.5. User Modifications

Our method allows an animator to modify a motion during the flight phase, for example by changing the leg position from tucked straight. When legs are straightened, their distance from the center of mass increases, which causes an increase in the overall body inertia. Since the momentum must remain constant, the jump rotation must slow down. Our algorithm can automatically adjust the edited motion to assure the conservation of angular momentum. As long as the modifications do not introduce significant changes in the direction of the angular momentum vector $\mathbf{L}$, changes in the magnitude of $\mathbf{L}$ can be eliminated by first retiming the motion and then repositioning the character's COM during the flight phase.

Since momentum in the modified motion changes over time, the retiming factor must change as well. We create a new, retimed frame sequence by computing which frame should be displayed next, given the number of the previous frame in the retimed sequence:

$$\text{next\_frame} = \text{last\_frame} + \frac{1}{ratio(\text{last\_frame})},$$

where $ratio(f) = |\mathbf{L}_{modif}(f)|/|\mathbf{L}(f)|$ is a ratio between the angular momentum magnitude $|\mathbf{L}_{modif}|$ in the modified motion and the angular momentum magnitude $|\mathbf{L}|$ in the original motion at a frame f. For in-between frames, we linearly interpolate the joint angles and compute the angular momentums from the interpolated values. While retiming does not eliminate all angular momentum errors, it reduces them significantly (see Figure 3). Before the retiming, angular momentum error was over 50% and could be detected by most viewers (see Table 2). Our method reduces the error to 20% or less, so that according to our study it is no longer perceptible.

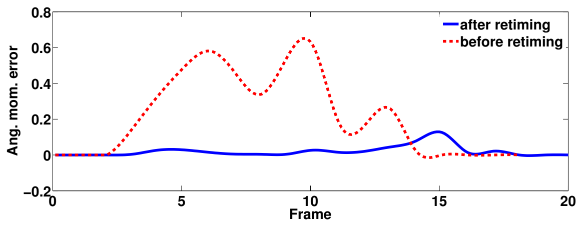

Figure 3: *Errors in the angular momentum in a motion modified by an animator before and after retiming relative to the original motion. Retiming causes significant reduction in the relative error.*

6. Results

We applied our technique to a variety of acrobatic motions from the CMU motion capture database [cmu] and commercial databases. Figure 4a presents a double backward somersault from a stand generated with our technique and the comparison with the original single-flip motion. The generated motion exhibits faster rotation and more rapid take-off and landing compared to the original jump, to accommodate changes in momentum due to the longer ballistic phase. Figure 4b shows the results of a user's editing operations. A tucked forward flip is modified by an animator to become a straight-leg jump. Our system retimes and repositions the motion to adapt to inertia changes. This causes the flight phase to be longer with slower rotation and greater jump height compared to the original motion. Take-off and landing phases are also modified due to the change in length of the flight phase. Figure 4c presents the results of joining together two acrobatic motions, an aerial cartwheel and a forward flip, to create a new stunt. The motions are retimed so that the angular momentums during the flight phase are equal in both of them. This example also shows the capability of our technique to easily change the height from which the character takes off.

Running time. The pre-processing stage of our algorithm took about 3 seconds. The run-time jump generation stage for a two-flip jump took 3.69 ms, a three-flip jump 4.60 ms and a four-flip jump 5.09 ms.

7. Experimental Evaluation

We studied human sensitivity to errors in angular momentum and take-off angle, to find out how the approximations introduced by our method affect viewers' perception of the generated motions.

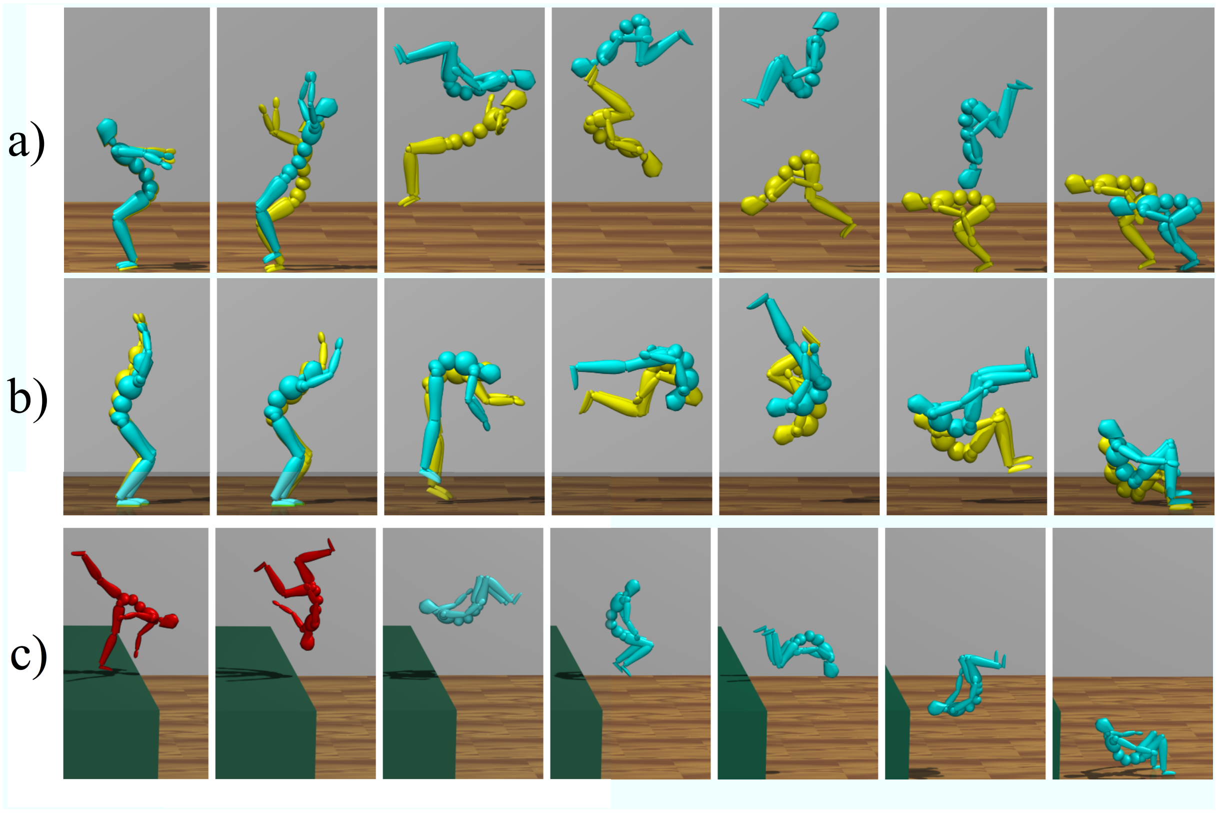

Figure 4: *a) A double backward somersault from a stand generated with our technique (in turquoise) compared with the original single-flip motion (in yellow). b) The results of a user's editing operation compared with the original motion. c) The results of joining together an aerial cartwheel (in red) and a forward flip (in turquoise).*

Participants. The study involved 19 student volunteers (3 women and 15 men) ranging in age from 20 to 25 years (3 participants did not specify their age). None of the volunteers had significant experience in computer animation or gymnastics.

Stimuli. Animations of single-flip forward and backward somersaults were created as a stimuli. All animations were rendered in the same style with the same camera configuration and the same take-off position. Original motion capture clips were used as base motions. Errors in angular momentum were created by the motion retiming during the flight phase. We tested both abrupt changes - sudden introduction of retiming in the middle of the flight phase, and smooth changes in angular momentum - retiming factor growing linearly and achieving maximum value just before landing. Both abrupt and smooth errors ranged from -50% to 50% of the original angular momentum magnitude. The errors didn't change the direction of the angular momentum vector. Errors in take-off angle were introduced by modifying the linear velocity in the flight phase along the direction of the jump and ranged from 0 to 40%.

Procedure. Participants were told that they were about to see a sequence of motions, some of which were physically correct and some of which were not. They were not informed how many of the motions contained errors or what kind of errors were introduced, but they were shown examples of correct motions to give them a reference point. Participants were instructed to mark each of the test motions either as correct or incorrect. The test motions were arranged randomly within each error group and then interleaved to reduce the learning effect. Each motion was shown twice.

Results. Similar to findings reported by [ODGK03] for simple objects, our subjects were not sensitive to even significant changes in angular momentum during ballistic motion. For example, for both smooth and abrupt changes, a 25% increase in angular momentum was imperceptible and as likely to be classified physically valid as the original motion. Surprisingly, motions with smooth decreases in momentum often scored higher than the original motions. This might be caused by humans overestimating the effect of air friction on the motion.

Likewise, our subjects proved not to be sensitive to errors in the take-off angle. A 30% change was perceived as physically valid as often as the original motion. Perhaps even

Ang. mom. modif.	Error level 0%	17 %	25%	37%	50%
smooth incr.	72 (.00)	71 (.05)	72 (.00)		41 (.81)
smooth decr.	72 (.00)	79 (-.22)	78 (-.18)	78 (-.18)	62 (.27)
abrupt incr.	72 (.00)	76 (-.13)	78 (-.18)	48 (.63)	44 (.75)
abrupt decr.	72 (.00)	83 (-.38)	68 (.11)	33 (1.0)	22 (1.4)
Take-off ang. modif.	**Error level 0%**	**10 %**	**20%**	**30%**	**40%**
incr.	63 (.00)	81 (-.53)	50 (.34)	63 (.00)	18 (1.3)

Table 2: *Results of our study: mean ratings (in percent) and sensitivity levels (in parenthesis). Sensitivity less or equal to zero means that participants could not detect errors.*

more surprising is the fact that the higher error didn't necessarily cause increased sensitivity: a 10% change scored significantly higher than the original motion and a 30% change was viewed as correct more often than a 20% change. This shows that most humans have relatively little experience with high-effort ballistic motions and find it difficult to estimate their correctness. The results of the study are summarized in Table 2.

8. Conclusions and Future Work

Our method for creating complex ballistic motions from simpler jumps is efficient because it uses only basic editing operations. Our approach works especially well with high-effort motions, which do not exhibit significant changes in take-off and landing poses with varying levels of effort. For low-energy jumps, it would be interesting to explore the possibilities of using our technique jointly with optimization methods, such as the ones proposed in [SP05] or [LP02], in order to adapt take-off and landing poses to different jump lengths. Another possible extension to this work would be a method which allows an animator to introduce changes to in-flight motion which significantly alter the direction of the angular momentum vector. The property of angular momentum conservation could then be restored by rotating the character appropriately.

Acknowledgments

The work in this paper was partially supported by NSF under contract CCF-0429983. We would like to thank coaches Ron Bell and Marshall Nelson for their assistance in this project and Brian Allen, Sung-Hee Lee, Shawn Singh, Andrew Sorensen and Victor Zordan for their helpful comments and discussion. We would also like to thank Intel Corp., Microsoft Corp., ATI Corp. and AGEIA Corp. for their support through equipment and software grants.

Appendix A: Similarity Measure

In our search algorithm we use a comparison function which computes the distance between two poses while ignoring their positions in space and global rotations around $\mathbf{L}$. Our comparison method is similar to previous approaches [LCR*02, KGP02, AF02, ZMCF05]. However, in contrast to the above methods, we first rotate the skeleton around the momentum vector before the comparison to minimize differences in body orientation between compared frames. In other words, given two frames f_1 and f_2, we first compute rotation angle ϕ around $\mathbf{L}$, by projecting the angle between root orientations in f_1 and f_2 on a plane perpendicular to $\mathbf{L}$. Next, we rotate f_2 by ϕ to remove the differences in global orientations around $\mathbf{L}$. Finally, we compute the distance between the two frames as a weighted sum of differences between global orientations, joint angles and angular velocities.

Similarly to [ZMCF05], we assign higher weights to joints close to the trunk, as differences in limb positions can be easily smoothed by blending. We used the same set of weights ($w_{root} = 2$, $w_{trunk} = w_{thigh} = 0.7$, $w_{knee} = 0.5$, $w_{shoulder} = 0.3$, $w_{elbow} = 0.1$) for all of our motions.

Appendix B: Computing Scaling Factor k^*

The retiming process affects the linear momentum in two directions: it increases the momentum during take-off and landing, and also decreases the necessary momentum build-up by reducing the flight duration. Therefore, the take-off velocity is scaled: $\mathbf{v}_0^* = k^*\mathbf{v}_0$, and the duration of the flight is shortened: $T^* = T/k^*$, where $\mathbf{v}_0$ is the take-off velocity along the y axis in the original motion and T is the duration of the flight phase before retiming. Given the parabola equation: $y^*(t) = -\frac{1}{2}gt^2 + \mathbf{v}_0^*t + y_{start}$ and the end condition: $y^*(T^*) = y_{end}$ we solve for the factor k^* which assures the continuity of the linear momentum curve:

$$k^* = \sqrt{\frac{\frac{1}{2}gT}{\mathbf{v}_0 - \frac{\Delta y}{T}}},$$

where g is the gravity coefficient, and $\Delta y = y_{end} - y_{start}$ is the difference in COM heights between take-off and landing.

References

[AF02] Arikan O., Forsyth D. A.: Interactive motion generation from examples. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002).

[ALP06] Abe Y., Liu C. K., Popović Z.: Momentum-based parameterization of dynamic character motion. *Graph. Models 68*, 2 (2006), 194–211.

[Bel] Bell R.:. Personal communication. Gymnastic coach and competition judge, Olympic gymnast.

[BL06] Bridgett L., Linthorne N.: Changes in long jump take-off technique with increasing run-up speed. *Journal of Sports Sciences 24*, 8 (2006), 889–897.

[BSP02] Bhat K. S., Seitz S. M., Popovic J.: Computing the physical parameters of rigid-body motion from video. In *ECCV (1)* (2002), pp. 551–565.

[CM69] Clauser C. E., McConville J. T.: *Weight, Volume, and Center of Mass of Segments of the Human Body*. National Technical Information Service, 1969.

[cmu] CMU graphics lab motion capture database. http://mocap.cs.cmu.edu/.

[DG67] Dempster W. T., Gaughran G. R. L.: Properties of body segments based on size and weight. *American Journal of Anatomy 120*, 1 (1967), 33–54.

[FP03] Fang A. C., Pollard N. S.: Efficient synthesis of physically valid human motion. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers* (2003), pp. 417–426.

[FvdPT01] Faloutsos P., van de Panne M., Terzopoulos D.: The virtual stuntman: dynamic characters with a repertoire of autonomous motor skills. *Computers and Graphics 25*, 6 (2001), 933–953.

[HWBO95] Hodgins J. K., Wooten W. L., Brogan D. C., O'Brien J. F.: Animating human athletics. In *In Proceedings of SIGGRAPH 95* (August 1995), pp. 71 – 78.

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), pp. 473–482.

[KY03] King M. A., Yeadon M. R.: Coping with perturbations to a layout somersault in tumbling. *Journal of Biomechanics 36*, 7 (2003), 921–927.

[KY04] King M. A., Yeadon M. R.: Maximising somersault rotation in tumbling. *Journal of Biomechanics 37*, 4 (2004), 471–477.

[LCR*02] Lee J., Chai J., Reitsma P. S. A., Hodgins J. K., Pollard N. S.: Interactive control of avatars animated with human motion data. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), pp. 491–500.

[LGC94] Liu Z., Gortler S. J., Cohen M. F.: Hierarchical spacetime control. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (1994), pp. 35–42.

[LHP05] Liu C. K., Hertzmann A., Popović Z.: Learning physics-based motion style with nonlinear inverse optimization. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (2005), pp. 1071–1081.

[LP02] Liu C. K., Popović Z.: Synthesis of complex dynamic character motion from simple animations. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), pp. 408–416.

[LvdPF00] Laszlo J., van de Panne M., Fiume E.: Interactive control for physically-based animation. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (2000), pp. 201–208.

[MPS06] McCann J., Pollard N. S., Srinivasa S.: Physics-based motion retiming. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2006), pp. 205–214.

[Nel] Nelson M.:. Personal communication. Gymnastic coach, former gymnast (three-time NCAA champion).

[ODGK03] O'Sullivan C., Dingliana J., Giang T., Kaiser M. K.: Evaluating the visual fidelity of physically based animations. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers* (2003), pp. 527–536.

[PA00] Pandy M. G., Anderson F. C.: Dynamic simulation of human movement using large-scale models of the body. In *Proc. IEEE Intl. Conference on Robotics and Automation* (2000), pp. 676–681.

[RGBC96] Rose C., Guenter B., Bodenheimer B., Cohen M. F.: Efficient generation of motion transitions using spacetime constraints. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), pp. 147–154.

[RP03] Reitsma P. S. A., Pollard N. S.: Perceptual metrics for character animation: sensitivity to errors in ballistic motion. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers* (2003), pp. 537–542.

[SBvL00] Seyfarth A., Blickhan R., van Leeuwen J. L.: Optimum take-off techniques and muscle design for long jump. *J. Exp. Biol 203* (2000), 741–750.

[SH05] Safonova A., Hodgins J. K.: Analyzing the physical correctness of interpolated human motion. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005), pp. 171–180.

[SHP04] Safonova A., Hodgins J. K., Pollard N. S.: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers* (2004), pp. 514–521.

[SP05] Sulejmanpasić A., Popović J.: Adaptation of performed ballistic motion. *ACM Transactions on Graphics 24*, 1 (2005), 165–179.

[V-F59] V-Five Association of America: *Gymnastics and tumbling, prepared by Hartley D. Price [and others] for the V-Five Association of America*. Annapolis : United States Naval Institute, 1959.

[WH00] Wooten W. L., Hodgins J. K.: Simulating leaping, tumbling, landing, and balancing humans. In *ICRA* (2000), pp. 656–662.

[WK88] Witkin A., Kass M.: Spacetime constraints. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques* (1988), pp. 159–168.

[ZMCF05] Zordan V. B., Majkowska A., Chiu B., Fast M.: Dynamic response for motion capture animation. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (2005), pp. 697–701.

[ZvdP05] Zhao P., van de Panne M.: User interfaces for interactive control of physics-based 3d characters. In *I3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games* (2005), pp. 87–94.