

An Efficient Method for Simulating Flexible Connectors

Frederick Li¹ Jianmin Zhao² Beta Lam³ Rynson Lau^{2,3}

¹ Department of Computer Science, University of Durham, United Kingdom

² College of Mathematics, Physics and Info. Engineering, Zhejiang Normal University, China

³ Department of Computer Science, City University of Hong Kong, Hong Kong

Abstract — Physically-based dynamic simulation has been an active research area for many years. It is an important technique to enhance the realism of interactive multimedia or computer graphics applications. However, while most of these applications require real-time performance to handle user interactions, dynamic simulation is generally expensive computationally, as simulating a flexible (non-rigid) object is typically done by modeling it with a large number of rigid bodies, in which constraints or impulses among the rigid bodies need to be processed continuously. In addition, in an interactive type of application, the motion of rigid bodies may sometimes be intervened. Such interactions, however, cannot be modeled with typical dynamic simulation techniques, such as associating constraints or behavior functions to the rigid bodies, due to its unpredictable nature. In this paper, we present a method for simulating flexible connectors. Our method combines dynamic constraints and impulse-based simulation to support real-time simulation of flexible connectors, such as ropes or chains. It also handles dynamic interventions efficiently.

Index Terms — Physically-based dynamic simulation, dynamic constraints, impulse-based, intervention

I. INTRODUCTION

Multimedia and computer graphics technologies are widely applied in digital entertainment in recent years. Typical examples of digital entertainment include computer games, digital movies / music, and online chatting / instant messaging. Among these applications, computer games are the most demanding one, as users of such application often expect to perceive realistic and high quality output in an interactive manner. In light of this, we present a new real-time simulation method for flexible connectors. It enriches computer games by supporting interactive simulation of a string-like object with rigid bodies at the two ends, such as rope, hose or hanging earring, in a realistic way.

Traditionally, a connector may be modeled by a series of connected rigid bodies with a distance constraint set between each pair of neighboring rigid bodies. The total distance constraint of a connector, which is the sum of all the between-pair distance constraints within the connector, should be kept within a threshold. For instance, if the connector is a joint, the threshold may be set to zero. If the connector is a rope, the threshold may

be set to the length of the rope. When the total distance constraint is violated, constraint forces are applied to the connected rigid bodies throughout the connector. For example, if a rope is in tension, a force will be exerted to each of the connected rigid bodies of the rope; otherwise, the connected rigid bodies would perform free falling provided that no external force is acting on them. In practice, the motion of a flexible connector is difficult to model efficiently.

In this paper, we present a method to simulate the motion of flexible connectors and to handle the interventions with them. Our method models a flexible connector with a series of particles and associates a distance constraint between each pair of neighboring particles. With existing methods, every time when a flexible connector changes its shape, we need to resolve the constraints once, which is very time-consuming. In contrast, our method keeps track of the distance constraints in the flexible connector and resolves the constraint impulses only if the distance constraints are violated. This approach speeds up the simulation process. The main contributions of the paper include:

- providing an efficient way to support real-time simulation of a flexible connector, and
- handling interventions to the flexible connector.

The rest of the paper is organized as follows. Section II discusses related work. Section III lays out the technical preliminaries. Section IV presents the new method. Section V shows some experimental results and Section VI briefly concludes the paper.

II. RELATED WORK

In general, there are three main approaches for simulating physically-based dynamics: constraint-based approach, impulse-based approach, and particle systems. They are briefly summarized as follows.

A. Constraint-based Approach

The constraint-based approach models a flexible object as a collection of physically-based primitive elements, namely rigid bodies, in which each of them is associated with certain geometric constraints, such as position and orientation. The setting up of geometric constraints to the rigid bodies implicitly formulate the configuration, i.e.

the shape, of the flexible object. During a simulation, physical laws will be applied to these rigid bodies to compute their motions. The simulation program will then keep track of and maintain the satisfaction of the geometric constraints continuously. A method of this approach is Dynamic Constraint [12]. The method is flexible as it supports several constraint settings, including fixed positions, navigation paths and orientation. It also allows new types of constraints to be added. On the other hand, [11] simulates the motion of linked figures based on kinematic constraints, behavior functions, and inverse dynamics. The inputs of the method are the physical and behavioral characteristics defined for the linked figures in their initial state, which cannot be changed during the simulation. The kinematic constraints confine the movements of the free portions of the linked figures in motion generation. The behavior function determines how the linked figures should react to the environment, and inverse dynamics provides a means to determine forces required to produce certain motion. In [10], the simulation of rigid body dynamics is treated as point dynamics with constraints. An object is simulated by separating it into many points with constraints. Subsequently, rigidity is added as a constraint to each point of the object to hold the points together to form the object. In general, the constraint-based methods suffer from high computational cost in solving the constraints.

B. Impulse-based Approach

The impulse-based approach models the interactions among objects as collision impulses. Thus, the performance of the collision detection algorithm used is important to the efficiency of the simulation. Similar to the constraint-based approach, this approach also simulates an object as connected rigid bodies. In [8], the dynamics of continuous contacts between objects is solved based on the holonomic constraints. This method also extends the impact analysis to handle continuous and simultaneous contacts of many objects. In [3], all types of contacts are modeled through a series of collision impulses between objects. The method evaluates the collision impulses through three main assumptions: Infinitesimal collision time, Poisson’s hypothesis, and Coulomb friction model. Infinitesimal collision time implies that the position of the objects does not change at the moment when the collision occurs. Poisson’s hypothesis divides the collision into two states, compression and restitution, for analysis. Coulomb friction model is used to simulate the friction when the collision occurs. Other impulse-based methods include [4, 7, 9]. They consider more factors in simulating the motion of an object, such as contact force, friction force and the effect of a collision impulse acting on different parts of an object, to enhance the simulation result. However, dynamics without collisions, such as flexible connector stretching, is difficult to model using the impulse-based approach.

C. Particle Systems

Particle Systems [13] are usually used to simulate non-rigid objects. It formulates an object as a cloud of particles. During a simulation, particles are generated, move, change their properties and are destroyed according to their lifetimes. Such approach may offer an alternative solution for modeling a flexible connector. Typical applications of particle systems include the simulation of natural phenomenon, e.g. fire, smoke, water, clouds and snow, in which each particle is attached with certain appropriate properties, such as mass, velocity, and motion behavior, to model a tiny part of the object. However, it is not straightforward to adopt the particle systems to model a flexible connector. First, with particle systems developed for fire or fluid simulations [2, 13], particles generally have a limited lifetime and are destroyed as the time passes. Such property is not applicable for simulating a flexible connector. Second, objects modeled by a particle system generally do not have a well-defined shape, while we need to know the shape when modeling a flexible connector. To address this issue, a mass-spring model can be set up in a particle system [1, 6]. The shape of an object at any particular state can then be obtained by running the system into an equilibrium state with regard to the springs among particles. However, such process can be time-consuming while the shape of a flexible connector may change continuously. Particle systems may not be practical for some interactive applications.

III. TECHNICAL PRELIMINARIES

We formulate a flexible connector as a collection of particles. To simulate both the shape and the motion of the connector, we keep track of the distance constraints between neighboring particles and evaluate the impulses at the end points of the connector. To do this, we maintain both the position $x(t)$ and the orientation $r(t)$ of each particle at any particular time t , where $x(t) = [p_x \ p_y \ p_z]^T$ is a vector of the 3D coordinates representing the spatial location of the particle. $r(t)$ can be represented by Euler angles and quaternion. Euler angles specify the rotation angle of a particle with respect to the three principle axes, while quaternion specifies the axis and the angle of the rotation. For instance, a rotation of angle θ around axis $[a_x \ a_y \ a_z]^T$ in quaternion is

$$\left[a_x \sin\left(\frac{\theta}{2}\right) \ a_y \sin\left(\frac{\theta}{2}\right) \ a_z \sin\left(\frac{\theta}{2}\right) \ \cos\left(\frac{\theta}{2}\right) \right]^T .$$

To model the motion of the connector, we maintain both the linear velocity, $v(t)$, which equals to $\frac{d}{dt}x(t)$ and can be expressed as $\left[\frac{d}{dt}p_x \ \frac{d}{dt}p_y \ \frac{d}{dt}p_z \right]^T$, and the angular velocity, $\omega(t)$, where the direction and the

magnitude of $\omega(t)$ is the rotation axis and the angular speed, respectively.

To model the impulse of a particle, we consider the relationship between a force, F , and an impulse, J , which is expressed as $J = \int F dt = F \Delta t$. By applying Newton's second law of motion, $F = ma$, where m is the mass and a is the acceleration, the impulse becomes $J = F \Delta t = m \Delta v$.

IV. OUR METHOD

In our method, we model a flexible connector as a series of connected particles. To simulate the motion of the connector, each particle is simulated individually as a free object. The simulation is divided into two parts. The first part is to evaluate the interaction at the end points of the flexible connector by considering both the constraint impulse acting on these points and the distance constraint of the connector (Ref. Section IV.A). The second part is to reposition the particles that representing the flexible connector using the particle system approach (Ref. Section IV.B). Here, we also show how to extend the simulation to handle interventions to the flexible connector (Ref. Section IV.C).

A. End Points Interaction

Preliminaries: A flexible connector is dynamic in nature and its shape may change over time. To enhance the simulation performance, the constraint impulse exerted on any particle of the connector will not be evaluated if the distance constraint between the particle and its neighbor is not violated. Technically, this means that the distance, $d(x_i, x_j)$, between x_i and x_j , does not violate the distance constraint, $l_{i,j}^c \leq d(x_i, x_j) \leq l_{i,j}^t$, where x_i and x_j are the i th and j th particles of the connector, respectively, and $d()$ is the distance function. $l_{i,j}^c$ and $l_{i,j}^t$ are the lengths of the segment between x_i and x_j started to be compressed and started to be stretched, respectively. Under such situation, the following assumptions are held for x_i and x_j :

- only free falling motion is applied to x_i and x_j if there is no external forces applied to them,
- no interaction is imposed between x_i and x_j , and
- positions of x_i and x_j do not change at the moment when the constraint impulse is applied.

End Point Interactions: Figure 1 shows a stretched flexible connector. The two end points, **A** and **B**, of the connector are attached to two different rigid bodies. If we pick up the rigid body attached to end point **B** and move it to the right while letting the one attached to end point **A** to move freely, a constraint impulse J will be exerted between the end points along \overline{BA} to avoid the connector

violating the distance constraint. Mathematically, the constraint impulse J can be expressed as follows:

$$J \times \overline{BA} = 0$$

Let \vec{v}_r be the relative velocity of end point **A** to end point **B, \vec{v}_n and \vec{v}_t be the normal and the tangent components of \vec{v}_r , respectively. As the constraint impulse J is applied to stop the length between the end points from increasing, it implicitly eliminates \vec{v}_n , while \vec{v}_t is not affected by J . In practice, the elasticity of the connector should generate a small amount of "rebounces" along \vec{v}_n . Hence, \vec{v}_n may not be zero eventually.**

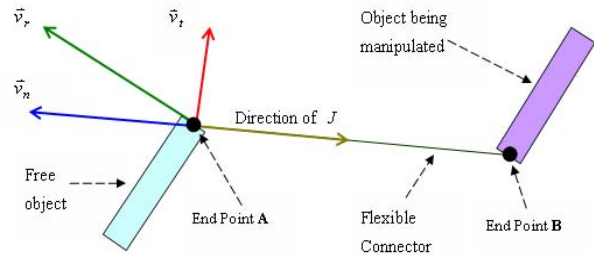


Figure 1: Constraint impulse J would eliminate \vec{v}_n . It also produces the "rebound" effect in the direction of $-\vec{v}_n$ and the damping effect on \vec{v}_t .

Similar to [5], we evaluate the end point velocity of the connector from the composition of the linear velocity and the angular velocity of the rigid body attached to the end point. Hence, the velocity of end point **A**, v_a , is:

$$v_a = v_{com} + r\omega \tag{1}$$

where v_{com} is the linear velocity of the centre of mass of the rigid body attached to end point **A**, r is the distance between end point **A** and the centre of mass of the rigid body attached to end point **A**, and ω is the angular velocity of the rigid body attached to end point **A**.

In addition, constraint impulse J would affect both the linear velocity and the angular velocity of the rigid body. Let m_a and I be the mass and the moment of inertia, respectively, of the rigid body attached to end point **A**. The relationship can be expressed as follows:

For linear velocity:

$$J = m_a \Delta v_{com}$$

$$\frac{dv_{com}}{dJ} = \frac{1}{m_a} \tag{2}$$

For angular velocity:

$$Jr = I \Delta \omega$$

$$\frac{d\omega}{dJ} = \frac{r}{I} \tag{3}$$

By substituting Eq. (2) and (3) into Eq. (1), we obtain the following equation:

$$\frac{dv_a}{dJ} = \frac{dv_{com}}{dJ} + r \frac{d\omega}{dJ} = \frac{1}{m_a} + r \frac{r}{I}$$

Hence,
$$\frac{dv_a}{dJ} = \frac{1}{m_a} + \frac{r^2}{I} \tag{4}$$

As \vec{v}_n should be eliminated and we need to simulate the “rebound” effect, the velocity change of end point **A**, Δv_a , would become $-\vec{v}_n \varepsilon - \vec{v}_n$, where ε is the elasticity coefficient of the connector. The negative sign of $\vec{v}_n \varepsilon$ indicates that the velocity is in the opposite direction of \vec{v}_n . Finally, the ratio of the mass of the end point to that of the connector, $\frac{m_a}{m_c + m_a}$, where m_c is the mass of the

connector, is introduced to simulate different types of connectors. Including this factor in the simulation is important. As an example, a single rope is typically lighter than a chain. The rope would then absorb less energy than the chain. Hence, the end points should get a higher velocity if the connector is a rope. The constraint impulse applies to an end point is modified as follows:

$$J = \frac{\Delta v_a}{\frac{dv_a}{dJ}} \times \frac{m_a}{m_c + m_a} = \frac{-v_n \varepsilon - v_n}{\frac{1}{m_a} + \frac{r^2}{I}} \times \frac{m_a}{m_c + m_a}$$

Hence,
$$J = -\frac{v_n(\varepsilon + 1)}{\frac{1}{m_a} + \frac{r^2}{I}} \times \frac{m_a}{m_c + m_a} \tag{5}$$

By solving these constraints, the interaction between the end points can be determined.

B. Simulating the Connector

Our method computes the end point positions of a flexible connector before simulating the connector itself. This helps eliminate the complicated computations on the constraints for the connected particles of the connector. With the end point positions of the connector computed, the remaining task is to ensure that the simulated result is physically close to the correct one. In particular, the following constraints should be satisfied:

- the end points of the connector should be at the correct positions,
- the shape of the connector should not change too much between any two consecutive frames, in order to produce a more realistic effect, and
- the length between each neighboring pair of particles should not exceed the length constraint.

The flexible connector is simulated by particle system. It is modelled as a set of particles. There are three main factors determining the simulation result:

- **Mass of the connector:** It affects how each particle of the connector is pulled towards or apart from its neighboring ones in the simulation. Such pulling effect reduces as the mass of a particle increases. In the simulation, the particles representing the connector end points are set to have infinite mass.
- **Length of the connector:** Given that a fixed number of particles are used to simulate the connector, increasing the length of the connector increases the distance between neighboring particles.

- **Number of particles to represent the connector:** Increasing the number of particles enhances the “smoothness” of the connector, as the length of each connector segment will be shorten. This helps the connector bend much softly.

During the simulation, the updated position of a particle would be evaluated if the distance constraint of one of its segments, r_{nei} , is violated. r_{nei} is evaluated as:

$$r_{nei} = \frac{k}{n-1}$$

where k is the length of the connector and n is the number of particles used to model the connector. In practice, we revise the distance constraint by including a user acceptable error threshold, h . The new distance constraint is set between $r_{nei} - h$ and $r_{nei} + h$. Hence, given two neighboring particles p_i and p_j with mass m_i and m_j , respectively, they would be repositioned only if $|p_i - p_j| < r_{nei} - h$ or $|p_i - p_j| > r_{nei} + h$. Repositioning p_i and p_j , which are in the directions of $p_j - p_i$ and $p_i - p_j$, respectively, would be set to allow $|p_i - p_j| = r_{nei}$. In addition, the magnitude of the repositioning is set according to the mass of the particles. Provided that a fixed force is acting on an object, according to Newton's first laws of motion, a heavier object would be accelerated less than a lighter object. Hence, the displacement, Δd_i , of p_i becomes:

$$\frac{m_j}{m_i + m_j} \left(r_{nei} - |p_j - p_i| \left(\frac{p_j - p_i}{|p_j - p_i|} \right) \right)$$

After repositioning every particle according to their neighboring ones, we reposition each particle according to the particle-to-end point distance as shown in Figure 2. The repositioning only occurs if such distance violates the distance constraint r_{end} :

$$r_{end} = \frac{k(q-1)}{n-1}$$

where k is the length of the connector, q is the particle-to-end-point distance of the q^{th} particle, and n is the number of particles used to model the connector.

As the mass of the end point is set to be infinity, the displacement, Δd_i , of p_i becomes:

$$\left(r_{nei} - |x - p_i| \right) \left(\frac{x - p_i}{|x - p_i|} \right)$$

In each iteration, the constraint checking and repositioning of the end points are scheduled at the last step. Hence, the particles of the connector would not be mis-repositioned to go beyond the end points due to the distance error between neighboring particles. The velocity of each particle can be computed using inverse dynamics after each iteration. Such information would be used as the input to the next iteration. Figure 3 shows the workflow of our method for simulating a flexible connector.

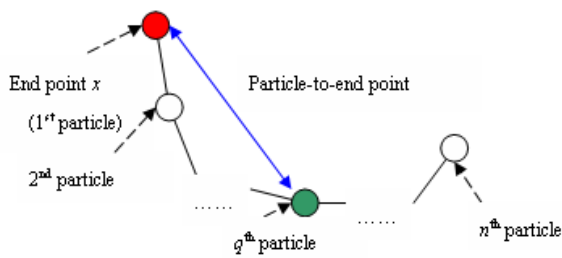


Figure 2: Particle-to-end-point distance.

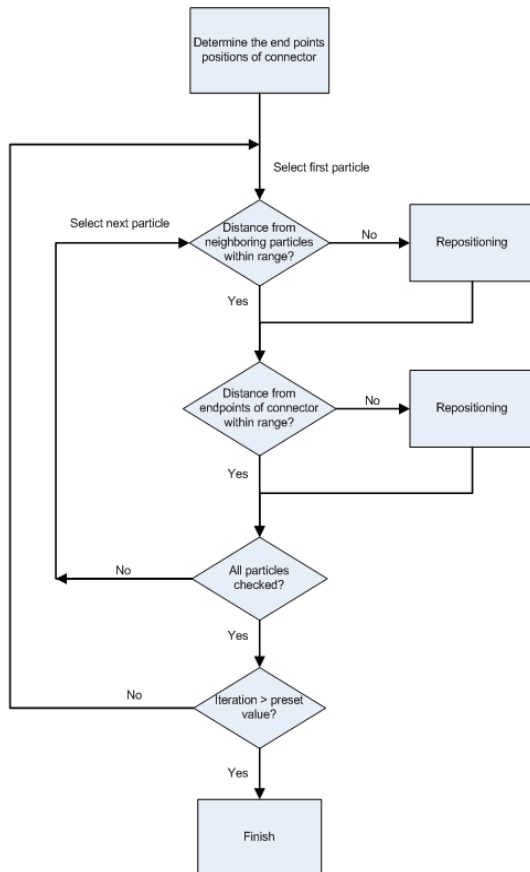


Figure 3: The workflow of our method.

C. Intervention Handling

In an interactive environment, a flexible connector may be disturbed by certain interventions, such as putting a pin to hang a rope. To keep the simulation running interactively, we process interventions by dividing a connector into parts rather than by conducting complex physical simulation. Figure 4 illustrates the idea by considering a rope with one end attached to a fixed point and the other end to a free object. Suppose a force is acted on a particular point of the rope, we logically break the rope into two parts at this point. One part will be responsible for simulating the rope from the fixed point to the point receiving the intervention (segment within the blue box), while another part simulates the rope from the point receiving the intervention to the end point attaching the free object (segment within the red box). If there is not a particle representing the break point, a new particle will be added to represent the point. Then, the properties

of the divided connectors, such as the connector length, need to be adjusted. Finally, an individual simulation will be run on the divided connectors separately.

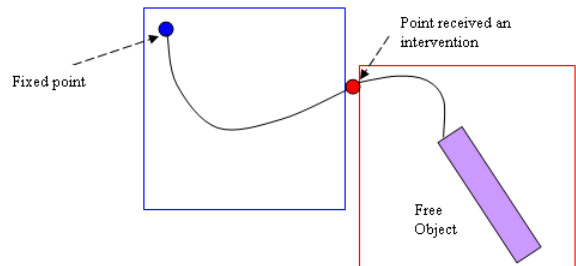


Figure 4: One connector may be divided into two independent connectors.

This technique of handling intervention to flexible connectors is unique to our method. First, intervention to flexible connectors is difficult to model in typical rigid body dynamic simulation techniques. As the time of the intervention is generally not predictable, such intervention cannot be modeled by behavior functions or constraints, which are the basis of typical rigid body dynamic simulation techniques. Second, conventional simulation methods cannot be used to divide a connector into parts for handling intervention to flexible connectors, as it would then change the system setting and hence would affect the correctness of the simulation result.

V. EXPERIMENT RESULTS

Our experiments were conducted on a PC with an Intel® Core™2 Duo 2.13 GHz CPU and 2GB RAM. We have compared the performance of our new method with the Dynamic Constraint method [12], a physically-based dynamic simulation package SD/FAST, and a simple particle system.

Experiment 1: In this experiment, we simulate a rope, which is attached to a fixed point at one end and a free object at the other end in our simulation. We run this simulation with the aforementioned methods and measure their performance for 100 time frames. Figure 5 shows the experimental results. From the results, we can see that both the Dynamic Constraint method and the SD/FAST package cannot support real-time simulation. This is because the Dynamic Constraint method needs to perform a series of complicated matrix computations, which involve finding the inverse of singular matrices. Such computations are very time-consuming to perform. The SD/FAST package performs even worse, as it may needs to run several iterations of such computations in each time frame. Among all existing methods, only the particle system runs fast enough to support real-time simulation. In addition, its performance is very stable in terms of the computation time spent in each time frame, as it performs the same amount of calculations in each frame. Nevertheless, our method performs even better than the particle system. On average, our new method runs 2.76 times faster than the particle system. Unlike the particle system, which spends time on sorting out the forces acting on individual particles of the rope and evaluating

the motions of the particles, our method reduces the simulation process to the repositioning of the particles representing the rope rather than.

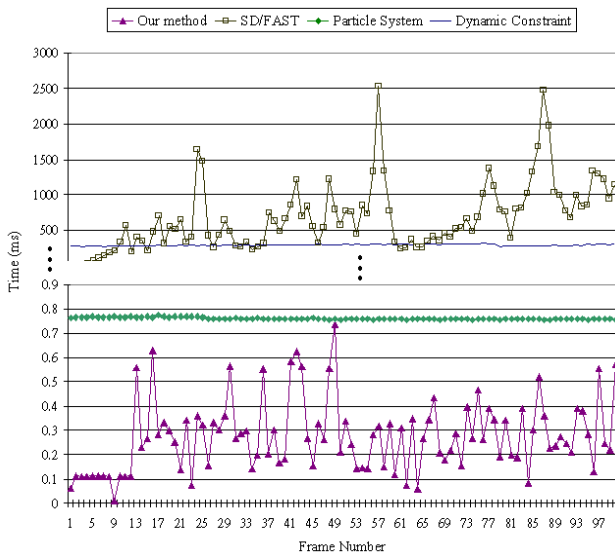


Figure 5: The graph shows the simulation time spent on different methods.

Experiment 2: In this experiment, we simulate the free falling motion of a rope in a cube space with 1.2m on each side. The rope is of 0.2m in length, which is attached to the center of the cube at one end and a handle at the other end. Initially, the rope is picked towards the top of the cube space using the handle. Then, the rope is dropped down by releasing the handle. We regard the result obtained from the Dynamic Constraint method as the ideal one and use it as the reference method. Consequently, we evaluate the correctness of each studied method by measuring the average position difference of the rope segments obtained from the method against the reference one. The results are shown in Figure 6. By observation, the result obtained from the particle system is the worst. Since in the particle system, the rope is modeled by connecting the particles with hard springs, which may cause the particles to receive large forces easily, the rope segments may generally vibrate with large magnitudes. This is especially true for those segments from the heavy particles, e.g., those representing the handle. This introduces large errors to the particle system based simulation. In contrast, SD/FAST produces a better result since it models flexible connectors using similar constraints as the reference method does, even though the two methods solve the constraints through different algorithms. Although our method models flexible connectors using the particle system, our method still produces a better result than the generic particle systems and the SD/FAST method. This is because our method includes a particle repositioning mechanism, as mentioned in Section IV.B, to correct the simulation results.

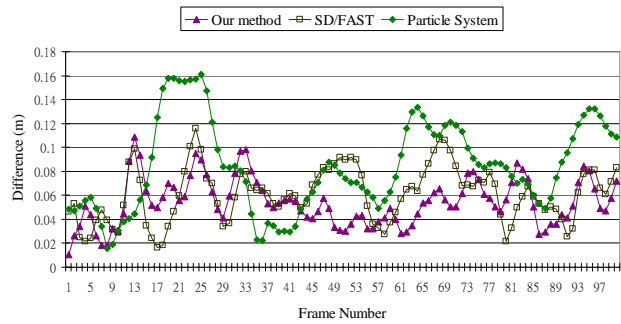


Figure 6: The rope segment position difference between “Dynamic Constraint” and other methods.

Demonstration: We have implemented a simple virtual environment, where a virtual character is playing a nunchaku. The nunchaku is formed by two sticks, which are connected by a flexible connector. Screen shots of the demonstration are shown in Figure 7. In the demonstration, the virtual character holds and moves the nunchaku by one of its stick, and leaves the other stick of the nunchaku moving freely (Ref. Figure 7(a)). Then, the virtual character stops moving the nunchaku (Ref. Figure 7(b)). The free moving stick moves upward for a while as affected by the former manipulation from the virtual character (Ref. Figure 7(c)) and falls down later due to gravity (Ref. Figure 7(d)). As the free moving stick keeps falling, it eventually receives impulses because the distance constraints of the flexible connector become violated (Ref. Figure 7(e)). When the virtual character moves the nunchaku again, the free moving stick performs free falling again at certain time, when the distance constraints of the flexible connector is not violated (Ref. Figure 7(f)).

V. CONCLUSION

In this paper, we have presented a new method for simulating flexible connectors. Our method is simple as it does not require deriving a separate set of equations or behavior functions for different constraints or for flexible connectors with different structures, while the constraint-based methods typically requires this. According to our experimental results, the new method performs better than some existing methods in terms of computational costs and accuracy.

ACKNOWLEDGMENT

The work described in this paper was partially supported by a SRG grant from City University of Hong Kong (Project Number: 7002309) and an UK EPSRC grant (Project Number: EP/G009635/1).

REFERENCES

[1] A. Kilian and J. Ochsendorf, “Particle-Spring Systems for Structural Form Finding,” *Journal of the Int’l Association for Shell and Spatial Structures*, **46**(2):77-84, 2005.
 [2] A. Lamorlette and N. Foster, “Structural Modeling of Flames for a Production Environment,” *ACM Trans. on Graphics*, **21**(3):729-735, Jul., 2002.

- [3] B. Mirtich and J. Canny, "Impulse-Based Simulation of Rigid Bodies," *Proc. ACM Symp. on Interactive 3D Graphics*, pp. 181-188, Apr., 1995.
- [4] D. Baraff, "Analytical Methods for Dynamic Simulation of Non-Penetrating Rigid Bodies," *Proc. ACM SIGGRAPH*, pp. 223-232, Jul., 1989.
- [5] D. Baraff, "Physically Based Modeling: Rigid Body Simulation," <http://www.pixar.com/companyinfo/research/pbm2001/pdf/notesg.pdf>.
- [6] E. Hergenrother and P. Dähne, "Real-Time Virtual Cables Based On Kinematic Simulation," *Proc. WSCG*, pp. 402-409, Feb., 2000.
- [7] J. Bender and A. Schmitt, "Fast Dynamic Simulation of Multi-Body Systems Using Impulses," *Proc. VRIPhys*, pp. 81-90, Nov., 2006.
- [8] J. Hahn, "Realistic Animation of Rigid Bodies," *Proc. ACM SIGGRAPH*, pp. 299-308, Aug., 1988.
- [9] K. Kawachi, H. Suzuki, and F. Kimura, "Simulation of Rigid Body Motion with Impulsive Friction Force," *Proc. IEEE Int'l Symp. on Assembly and Task Planning*, pp. 182-187, Aug., 1997.
- [10] K. van Overveld and B. Barenbrug, "All You Need is Force: A Constraint-Based Approach for Rigid Body Dynamics in Computer Animation," *Proc. Computer Animation and Simulation*, pp. 80-94, 1995.
- [11] P. Isaacs, and M. Cohen, "Controlling Dynamic Simulation with Kinematic Constraints," *Proc. ACM SIGGRAPH*, pp. 215-224, Jul., 1987.
- [12] R. Barzel, and A. Barr, "A Modeling System Based On Dynamic Constraints," *Proc. ACM SIGGRAPH*, pp. 179-188, Aug., 1988.
- [13] W. Reeves, "Particle Systems - A Technique for Modeling a Class of Fuzzy Objects," *ACM Trans. on Graphics*, 2(2):91-108, Apr., 1983.

Frederick Li received both a BA (Hons.) in Computing Studies and an MPhil. in Computer Science from The Hong Kong Polytechnic University, and a Ph.D. degree in Computer Graphics from the City University of Hong Kong. He is currently a Lecturer (Assistant Professor) at the University of Durham. Prior to the current appointment, he taught at The Hong Kong Polytechnic University. Before that, he was the project manager of a Hong Kong Government Innovation and Technology Fund. His research interests include computer

graphics, virtual reality, surface modeling, distributed virtual environment and multimedia systems.

Jianmin Zhao received a B.Sc. degree in Physics from Zhejiang Normal University and M.Sc. degree in Computer Science from Zhejiang University, China. He is a Professor and Dean of College of Mathematics, Physics and Information Engineering at Zhejiang Normal University. He has served as Vice-Director of a number of departments, including National Higher Pedagogical Colleges, Universities Computer Education Research Board, Zhejiang Province Computer Academic Society, Zhejiang Province Computer Education and Application Academic Society, executive member of the council of Chinese Computer Academic Society Theory Computer Specialized Committee, the council of national CBE Academic Society and East China University Computer Elementary Education Research Board, assistant director of Zhejiang Province computer education Direction committee, expert of the State Natural Sciences Foundation and Zhejiang Province natural sciences funding project, and editorial member of core publication "Microelectronics and Computer" committee. He has also served as Conference Co-chair of ICWL 2008.

Beta Lam received both a B.Sc. in Computer Engineering and Information Technology and an MPhil in Computer Science from City University of Hong Kong. His research interests include computer graphics and dynamic simulation.

Rynson Lau received a B.Sc. degree in Computer Systems Engineering from the University of Kent and a Ph.D. degree in Computer Science from the University of Cambridge. He has been on the faculty of University of Durham, City University of Hong Kong, and the Hong Kong Polytechnic University. He is currently with the City University of Hong Kong. He has served as the Guest Editor of a number of journal special issues, including *ACM Trans. on Internet Technology*, *IEEE Trans. on Multimedia*, *IEEE Trans. on Visualization and Computer Graphics*, *Presence*, *IEEE Computer Graphics & Applications*, and *IEEE Internet Computing*. He has also served in the conference committee of a number of conferences, including Program Co-chair of *ACM VRST 2004*, *ICAT 2006*, *ICEC 2007*, *IEEE U-Media 2009*, and Conference Co-chair of *ACM VRST 2005*, *CASA 2005*, *IDET 2008*.

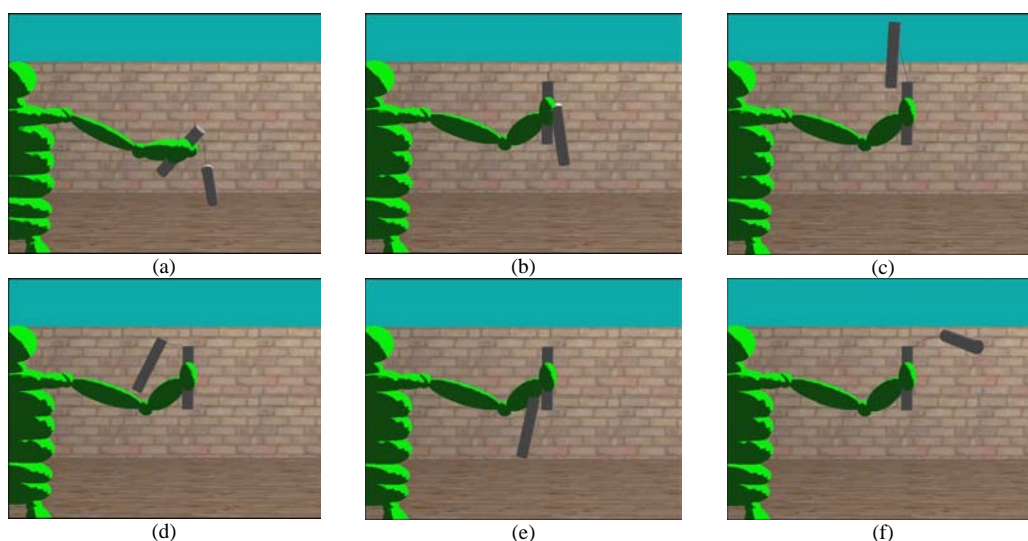


Figure 7: Scene shots of a virtual character playing nunchaku.