View-dependent dynamics of articulated bodies

Sujeong Kim, Stephane Redon, Young Kim

View-Dependent Dynamics of Articulated Bodies

Sujeong Kim
Ewha Womans University
kimsujeong@ewhain.net

Stephane Redon
i3D-INRIA Rhône-Alpes
stephane.redon@inria.fr

Young J. Kim*
Ewha Womans University
kimy@ewha.ac.kr

Abstract

We propose a method for view-dependent simplification of articulated-body dynamics, which enables an automatic trade-off between visual precision and computational efficiency. We begin by discussing the problem of simplifying the simulation based on visual criteria, and show that it raises a number of challenging questions. We then focus on articulated-body dynamics simulation, and propose a semi-predictive approach which relies on a combination of exact, a priori error metrics computations, and visibility estimations. We suggest several variants of semi-predictive metrics based on hierarchical data structures and the use of graphics hardware, and discuss their relative merits in terms of computational efficiency and precision. Finally, we present several benchmarks and demonstrate how our view-dependent articulated-body dynamics method allows an animator (or a physics engine) to finely tune the visual quality and obtain potentially significant speed-ups during interactive or off-line simulations.

Keywords: view-dependent dynamics, adaptive dynamics, articulated body simulation

1 Introduction

Physically-based simulation has been taking an increasingly important role in numerous graphical applications where a realistic motion is desired, including computer animation, feature films, computer games, and virtual reality. In particular, articulated-body dynamics has been used to realistically simulate the motions of diverse forms of animating characters such as humans, hair, animals, plants, etc.

One of the fundamental problems in articulated-body dynamics is the forward dynamics problem, which computes the motion of an articulated-body when the given forces are exerted to the body. Linear-time, optimal solutions for forward dynamics are well-known (*e.g.* [1, 2]); however, these solutions can be still very costly for simulating numerous or complex articulated bodies in feature films or computer games.

This paper proposes a method for *view-dependent articulated-body dynamics*, which simplifies the forward dynamics of articulated bodies based on visual criteria. We propose a *semi-predictive* approach, which relies on a combination of exact, a priori error computations and visibility estimations. Our method is able to simplify the dynamics of an articulated body not only based on *visibility* criteria (*i.e.* the visible portion of the articulated body on the screen), but also based on the relative importance that the articulated-body motion has to the viewer. We demonstrate our approach on several benchmarks and show how our view-dependent articulated-body dynamics method allows an animator (or a physics engine) to finely tune the visual quality of a simulation, and obtain potentially significant speed-ups during interactive or off-line simulations. As will be shown in the benchmarking results, without incurring visual deterioration (*e.g.* popping), the view-dependent dynamics gracefully simplifies the level of details in the simulation and thus provide visually-plausible simulation to the viewer.

Organization: Section 2 provides an overview of related work on simulation levels of detail and view-dependent dynamics simplification. In Section 3, we discuss the general problem of simplify-

*Corresponding author

ing a simulation based on visual criteria. Section 4 gives an overview of the adaptive articulated-body algorithm, upon which our algorithm is based. Section 5 introduces our semi-predictive motion metrics. Section 6 presents several applications of our approach. Finally, Section 7 concludes this paper and suggests a few future research directions.

2 Related Work

View-dependent simplifications of a dynamics simulation is related to the following research areas.

Simulation levels of detail: A number of approaches exist for adaptive simulation of a number of complex systems, including deformable bodies [3, 4], cloth [5], fluid and smoke [6], hair [7, 8], or objects with a finite but potentially large number of degrees of freedom such as particle systems [9] or articulated bodies [10]. Often, these approaches resort to some type of *adaptivity* to refine the simulation where the current level of discretized simulation cannot appropriately emulate the full dynamics of the system, independently of visual criteria (but possibly with application to view-dependent simplification).

View-dependent simplification: Some authors have specifically addressed the problem of simplifying a simulation based on visual criteria. Carlson and Hodgins [11] use three levels of sophistication to animate creatures in a virtual environment partly based on the distance of the creature to the camera (with arbitrary thresholds, however). Perbet and Cani [12] animate prairies using three levels of detail based on the viewer's position (classified as near, medium and far). O'Brien *et al.* [9] simplifies the dynamics of particle systems by clustering particles into groups, partly based on visual criteria. Chenney and Forsyth [13] discuss a view-dependent culling of dynamic systems and introduce two important criteria that should be satisfied by a dynamics simplification method, namely *consistency* (object re-entering the view satisfy viewers' expectations) and *completeness* (objects that should re-enter the view do so). In particular, they classify the objects based on the viewers expectations, and focus computing resources on the objects for which the viewers have certain expectations. However, they do not discuss how to simplify the dynamics of visible objects. Chenney *et al.* [14] focuses on the consistency problem when the objects motions can be tightly bound. Their method only applies to continuously evolving systems with few degrees of freedom and no external influence, which forbids interactive simulations and collision handling. Beaudoin and Keyser [15] present a method to simplify plant motion, and propose rigorous methods to compute the errors caused by the approximations. The levels of details are pre-computed and a generalization to other objects and external forces does not seem straightforward.

Perceptually-based simplification: How an animation or a simulation is being perceived by humans has recently received research attention. O'Sullivan and Dingliana [16] discuss how viewers perceive collisions in a dynamics simulation of rigid bodies, and in another work, O'Sullivan et al. attempt to evaluate the visual fidelity of such simulations [17]. Harrison *et al.* [18] study how noticeable changes in the lengths of articulated-body links are, depending on viewers attention. O'Sullivan [19] discusses the effect of collisions on attention. One goal of this research is often to define *perceptual metrics*, which would help simplify an animation or a simulation based on perceptual criteria. There have been also research efforts to generate and suggest plausible motions of the physical simulation under different initial conditions [20, 21]. [22] have introduced the *many worlds browsing method*, where the user is allowed to interactively browse and modify the simulation results to match his or her requirements. There are also attempts to understand the error thresholds for plausible motion [17].

3 View-dependent simulation

Although several methods have been proposed to take advantage of visibility to simplify simulations, it appears that very few authors have formally discussed *how* to choose an appropriate simulation level of detail based on visual criteria. In this section, we discuss this problem and identify a number of relevant issues. We find that the problem of quantifying and, most importantly, *predicting* the number of perturbed pixels due to an approximation in a simulation is surprisingly difficult, for a number of reasons. This may explain why the problem of simplifying a simulation based on its appearance has been relatively unexplored compared to, for example, the problem of view-dependent geometric simplification [23].

3.1 Defining an error measure

View-dependent simplification of dynamics first raises the problem of defining an appropriate error

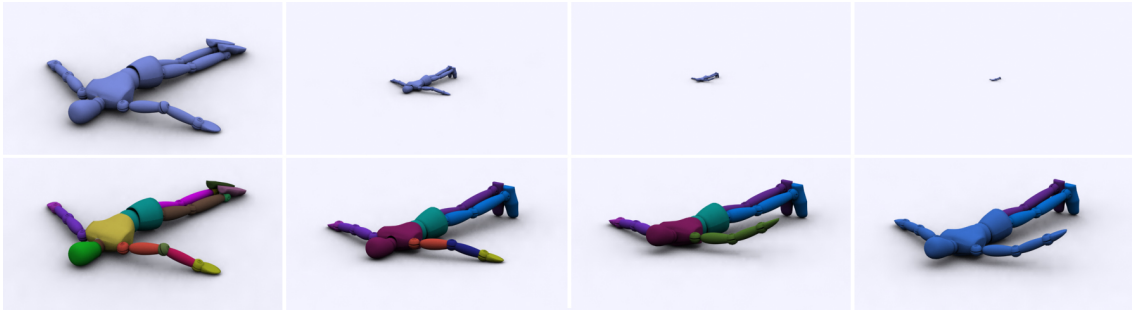

Figure 1: **View-dependent dynamics simplification.** **Top**: *Our algorithm automatically simplifies the dynamics of a falling character as its distance to the viewer increases.* **Bottom**: *Corresponding rigidification at this time step (one color per rigid group). See also Fig. 7 for the corresponding motion strips (see also the attached video).*

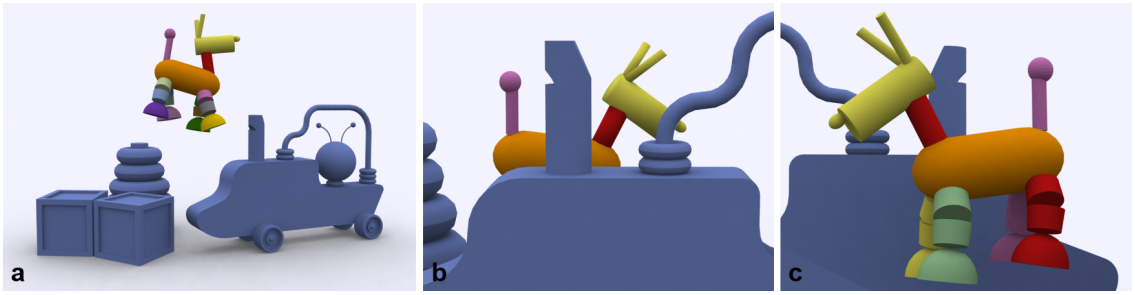

Figure 2: **View-dependent dynamics simplification of a toy-like dog model in an interactive application (offline rendering).** **a**: *the complete environment. The user controls the model (sixteen rigid bodies) with a haptic interface.* **b**: *the user places the dog behind the environment. Our algorithm automatically rigidifies the legs of the model, resulting in a total of eight rigid groups.* **c:** *this back view shows the rigid groups corresponding to the position shown in* **b** *(one color per rigid group — see attached video).*

measure with which the quality of a simplification can be judged.

3.1.1 Static case

Consider first the *static* problem, *i.e.* simplifying a simulation at a single instant in time. In the static problem, we would like to simplify the dynamics of the objects in the scene at a given time step and still obtain, at the next time step, an image "close" to the one we would have obtained if the full dynamics had been simulated. We thus need an objective measure of the similarity between the simplified frame and the fully simulated one, at the next time step.

Fortunately, this question has already been raised within the graphics and scientific visualization research communities for geometric simplification (see [24] for an extensive overview). Often, the error between two images is measured in terms of the number of pixels that differ between the two images [23]. More generally, the characteristics of the visual system should be accounted for, in order to take advantage of *e.g.* attention [19], mesh saliency [25], etc. Although much progress has been made, the problem of defining a "perceptual distance" between two images is still largely open.

Note that the simulated objects may go through a complex, non-linear rendering stage, involving complex lighting and material models as well as various post-processing stages (*e.g.* motion blur, editing, etc.). Ideally, a completely integrated system would take the full simulation and rendering pipeline into account, to avoid spending too much time on motions that would later be hidden by post-process.

3.1.2 Dynamic case

The *dynamic* problem, *i.e.* considering the long-term impact of a simplification, raises additional questions. Indeed, we should probably consider the impact of a simplification at a given time step on *all* subsequent time steps. Ideally, all simplifications should be invisible to the viewer. One way to measure the error could thus be to compare

the final images of the simplified segment and the original animation segment.

Because general systems are aperiodic, however, and may be extremely sensitive to perturbations, a simplified simulation might rapidly diverge from a non-simplified one. Thus, a better way to measure the error between two animation segments might thus be as the sum of (or bound on) successive static errors. This is typically how the quality of an integration method is evaluated, *i.e.* by computing a bound on the error occurring at each time step. Such an error measure might be able to solve the consistency and completeness problems defined by Chenney and Forsyth [13], by choosing sufficiently low error thresholds.

3.2 Simplifying the dynamics

Adaptive methods can be roughly classified according to the way they evaluate (or estimate) and use error measures.

Ideally, the exact error should be computed *a priori*, *i.e. without* having to compute the exact solution to the problem. This is often extremely difficult, however, and a priori methods often *estimate* the error (*e.g.* [3]).

Note that another approach would be to design an *a posteriori* simplification method, similar to adaptive time-stepping integration methods. This would prove relatively easy: two images would be produced, one using full dynamics, and the other with view-dependent dynamics. If the images differ by less than a user-defined threshold (in terms of the number of pixels, for example), then the view-dependent simplification would be declared acceptable.

There are, however, at least three problems with such an approach. First, to make the scheme computationally worthwhile, we would have to assume that temporal coherency is high, so that the simplification test can be performed only once in a while (for example every one hundred frames). This might not be valid in numerous dynamics scenarios, especially when discontinuities resulting from collisions, external forces or user interactions may occur (moreover, regular "peaks" in computational costs might be inappropriate in interactive applications such as games and virtual environments, which favor consistent frame rates). Second, performing a full dynamics step might well be too slow when too many degrees of freedom are involved. Finally, it might be difficult to decide what simplification should be attempted. Should the degrees of freedom be removed because they have little variation? Should they be grouped together because they have *similar* variation? Because of the exponential number of combinations, some a priori assumptions have to be made to simplify the system before comparing it to the fully simulated one (*e.g.* merging and splitting degrees of freedom based on their acceleration [8]).

In this paper, we propose a *semi-predictive* approach, which combines a priori computations of joint accelerations errors with visibility estimations.

4 Preliminaries

The predictive component of our view-dependent method relies on the adaptive dynamics (AD) method introduced by Redon *et al.* [10]. For completeness, we briefly describe their approach here. We refer the reader to their paper for a detailed exposition.

4.1 Divide-and-conquer articulated-body dynamics

The AD algorithm can be seen as a generalization of the Divide-and-Conquer Algorithm (DCA) proposed by Featherstone [1, 2]. In this algorithm, an articulated body is recursively defined by connecting two articulated bodies. A *binary assembly tree* describes the sequence of assembly operations, in which the leaf nodes represent rigid bodies, and the root node corresponds to the complete articulated body (see Figure 3). Each non-leaf node thus represents both a sub-articulated body and the joint used to connect its two child nodes.

Featherstone [1, 2] shows that the dynamics of an articulated body can be described by the following *articulated-body equation*:

$$\mathbf{a} = \mathbf{\Phi}\mathbf{f} + \mathbf{b}, \tag{1}$$

which is similar to the Newton-Euler equation describing the motion of a rigid body. Here, $\mathbf{a}$ is the composite acceleration of the articulated body (a vector which concatenates the bodies accelerations), $\mathbf{\Phi}$ is the composite inverse inertia of the articulated body, $\mathbf{f}$ is a composite kinematic constraint force (which holds the articulated body together), and $\mathbf{b}$ is a composite bias acceleration, due to external forces and torques (inertial effects are zero under the quasi-statics assumption). Featherstone's DCA essentially consists in two passes over the complete assembly tree. The *main pass* is a bottom-up traversal, in which the DCA

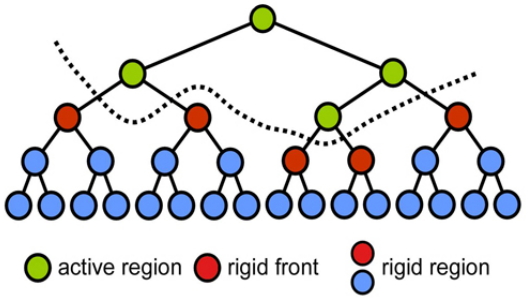

Figure 3: **Assembly tree of an articulated body.** Adaptive articulated-body dynamics simulates only some of the joints in the articulated bodies, which form the *active region*.

determines the inverse inertias $\mathbf{\Phi}$ and bias accelerations $\mathbf{b}$ for each node in the assembly tree from those of its children, and the external forces and torques applied on the articulated body. The top-down *back-substitution pass* computes, for each internal node starting from the root node, the kinematic constraint forces $\mathbf{f}$ (which enforce the kinematic constraint described by the node) and the acceleration $\ddot{\mathbf{q}}$ of the joint represented by the node. This algorithm is applied repeatedly to simulate the motion of an articulated body.

4.2 Adaptive articulated-body dynamics

4.2.1 Hybrid bodies

In order to speed up articulated-body dynamics simulation, Redon *et al.* [10] *approximately* solve the problem by computing joint accelerations in a limited *sub-tree* of the assembly tree (*cf* Figure 3), called the *active region*. The remaining joints are *inactive*, and their accelerations are being implicitly set to zero. An articulated body with at least one inactive joint is called a *hybrid body*. The motion of a hybrid body is simulated by "rigidifying" the inactive joints. This results in *hybrid* inverse inertias and bias accelerations $\overline{\mathbf{\Phi}}$ and $\overline{\mathbf{b}}$, that can also be computed from the bottom up. The complexity of simulating a hybrid body is then $O(n_a + n_f \log(n/n_f))$, where n_a is the number of active joints, n is the total number of joints, and n_f is the number of nodes where an external force of a torque is updated. This results in potentially significant performance speed-ups when the number of active joints and external forces updates are low.

4.2.2 Active region determination

To approximate the motion that would have been obtained if full dynamics were computed, Redon *et al.* [10] periodically updates the active region using *motion metrics*. If C is an articulated body, the *acceleration metric* $\mathscr{A}(C)$ of C is a weighted sum of its joint accelerations:

$$\mathscr{A}(C) = \sum \ddot{\mathbf{q}}_\mathbf{i}^T \mathbf{A}_i \ddot{\mathbf{q}}_\mathbf{i}, \tag{2}$$

where the $\mathbf{A}_i$ are symmetric, positive definite matrices. Similarly, the *velocity metric* $\mathscr{V}(C)$ of C is a weighted sum of its joint velocities:

$$\mathscr{V}(C) = \sum \dot{\mathbf{q}}_\mathbf{i}^T \mathbf{V}_i \dot{\mathbf{q}}_\mathbf{i}, \tag{3}$$

where the $\mathbf{V}_i$ are symmetric, positive definite matrices. Redon *et al.* [10] shows that the acceleration metric value of an articulated body is a quadratic function of the kinematic constraint forces:

$$\mathscr{A}(C) = (\mathbf{f}^C)^T \mathbf{\Psi}^C \mathbf{f}^C + (\mathbf{f}^C)^T \mathbf{p}^C + \eta^C, \tag{4}$$

where the *acceleration metric coefficients* $\mathbf{\Psi}^C$, $\mathbf{p}^C$ and η^C can be computed from the bottom up (similar to the articulated-body coefficients). To update the active region, the acceleration metric is used to restrict the back-substitution pass to the most important sub-tree of the assembly tree. Then, the velocity metric is used to determine the new set of most important joints.

5 View-dependent metrics

We can now present our approach for view-dependent simplification of articulated-body dynamics. Our method can be regarded as *semi-predictive*, since we are able to predict the exact error in joint accelerations before actually computing all of them (using the adaptive dynamics framework), but we make some assumptions about how the visual error is affected by errors in joint accelerations.

5.1 Simplifying the simplification problem

In order to make the view-dependent simplification practical, we make two fundamental assumptions.

First, we assume that the motion of a joint only has a *local* effect on the motions of the rigid bodies (in cartesian coordinates). This is generally the case when there is little correlation in neighboring joints, and cross-coupling inverse inertias tend to have low ranks and not transmit applied

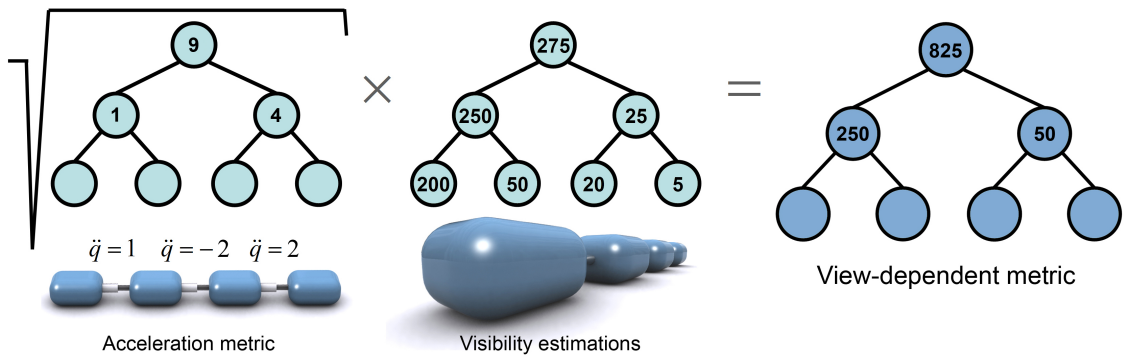

Figure 4: **View-dependent motion metric.** *The view-dependent motion metric is obtained by combining a priori motion metrics with visibility estimations.*

torques and forces [26]. This allows us to examine each sub-assembly of the articulated body independently of the others, by assuming that the visual impact of a joint acceleration error in a sub-assembly is approximately restricted to this sub-assembly.

Second, we assume that the visual error caused by the rigidification of a sub-assembly can be roughly obtained by decoupling the contribution of the sub-assembly motion (acceleration or velocity) from the visibility of objects under such motion (*e.g.* the number of rasterized pixels). The major benefit of this assumption is that we can easily customize the motion metrics in the adaptive dynamics framework, and exploit well-established measures of visibility used in other types of applications, in particular in rendering.

These two assumptions allow us to formulate view-dependent motion metrics which can be computed efficiently, while still providing reasonable estimations of the visual error caused by partial rigidifications (*cf* Section 6).

5.2 Semi-predictive metrics

Let us call $\mathscr{N}_{\mathbf{M}}(C)$ the projected area of an articulated body C onto the screen, under the viewing transformation $\mathbf{M}$. Then, the view-dependent acceleration and velocity metrics, $\mathscr{A}_{\mathbf{M}}(C)$ and $\mathscr{V}_{\mathbf{M}}(C)$, are defined as follows (*cf* Figure 4):

$$\begin{aligned}
\mathscr{A}_{\mathbf{M}}(C) &= dt^2 \mathscr{N}_{\mathbf{M}}(C) \sqrt{\sum \ddot{\mathbf{q}}_{\mathbf{i}}^T \mathbf{A}_i \ddot{\mathbf{q}}_{\mathbf{i}}} \\
&= dt^2 \mathscr{N}_{\mathbf{M}}(C) \sqrt{\mathscr{A}(C)}, \\
\mathscr{V}_{\mathbf{M}}(C) &= dt \mathscr{N}_{\mathbf{M}}(C) \sqrt{\sum \dot{\mathbf{q}}_{\mathbf{i}}^T \mathbf{V}_i \dot{\mathbf{q}}_{\mathbf{i}}} \\
&= dt \mathscr{N}_{\mathbf{M}}(C) \sqrt{\mathscr{V}(C)}, \\
\mathbf{A}_i &= \mathbf{V}_i = \mathbf{D}_i,
\end{aligned}$$

where dt is the size of the time step, and $\mathbf{D}_i$ is a diagonal weight used to homogenize joint types (to mix ball-socket joints and prismatic joints, for example — $\mathbf{D}_i$ can be set to the identity matrix if all degrees of freedom are of the same type).

Intuitively, these metrics estimate the visual error caused by zeroing accelerations or velocities by assuming the worst possible case: all joints in the sub-assembly have correlated motions, and contribute to the displacement of the whole visible surface. Once the acceleration and velocity metrics have been obtained as in the adaptive dynamics framework, these view-dependent metrics can be obtained in constant time, provided we know the values of $\mathscr{N}_{\mathbf{M}}(C)$. We now present different ways of obtaining $\mathscr{N}_{\mathbf{M}}(C)$ and discuss their relative advantages in terms of precision and computational efficiency.

5.3 Calculation of $\mathscr{N}_{\mathbf{M}}$ using bounding-volume hierarchies

For a given sub-assembly C, we can quickly approximate $\mathscr{N}_{\mathbf{M}}(C)$ using bounding volumes (BVs) such as spheres, oriented bounding boxes [27] or axis-aligned bounding boxes. Before the simulation begins, we compute a bounding volume for each rigid body in the articulated body, that we store in the local reference frame attached to the rigid body. We also associate a bounding volume to each internal node of the assembly tree. These internal bounding volumes are updated at runtime, so as to enclose the bounding volumes of their children. In order to approximate $\mathscr{N}_{\mathbf{M}}(C)$, we can then use either the bounding volume associated to C, or the bounding volumes associated to the rigid bodies composing C. The resulting approximations of $\mathscr{N}_{\mathbf{M}}(C)$, denoted by $\widetilde{N}_{\mathbf{M}}(C)$, is taken as the minimum of these two projections (see also

Figure 5):

$$\widetilde{\mathcal{N}}^1_{\mathrm{M}}(C) = \int\int_{\mathrm{M}(\partial \mathrm{BV}(C))} dxdy \quad (5)$$

$$\widetilde{\mathcal{N}}^2_{\mathrm{M}}(C) = \sum_{i \in C} \int\int_{\mathrm{M}(\partial \mathrm{BV}(C_i))} dxdy \quad (6)$$

$$\widetilde{\mathcal{N}}_{\mathrm{M}}(C) = \min\{\widetilde{\mathcal{N}}^i_{\mathrm{M}}(C) | i = 1,2\} \quad (7)$$

Even though we need to sacrifice additional time to update the internal bounding volumes at run-time, we may obtain a tighter estimation of $\mathcal{N}_{\mathrm{M}}$, depending on the configuration of the articulated body and the camera position. For example, if an articulated body forms a long but folded chain, then $\widetilde{\mathcal{N}}^1_{\mathrm{M}}$ will be smaller than $\widetilde{\mathcal{N}}^2_{\mathrm{M}}$. On the other hand, if this articulated body is stretched, then $\widetilde{\mathcal{N}}^1_{\mathrm{M}}$ will be larger.

The choice of the bounding volume also affects the theoretical complexity of the visibility estimation. If we use spheres or oriented bounding boxes, we can store their parameters in the local reference frame associated to the internal nodes. As a result, we do not have to update these bounding volumes in the rigid region, and the complexity of updating the bounding-volume hierarchy is linear in the number of *active* joints. In order to update the internal bounding volumes in constant time, however, we need to compute their parameters directly from those of their children, and not from the bounded geometry. This might result in overly conservative bounding volumes. In such a case, axis-aligned bounding volumes can be preferable despite the need to update all of them (*i.e.* including in the rigid region).

5.4 Calculation of $\mathcal{N}_{\mathrm{M}}$ using GPUs-based occlusion queries

Estimating visibility using bounding volumes has the following disadvantages:

- Without a proper clipping procedure or visible surface determination, the $\widetilde{\mathcal{N}}_{\mathrm{M}}(C)$ value in Eq. 7 is always positive.
- Depending on the choice of bounding volumes, the projected area can be quite conservative.

In order to address these issue, we rely on occlusion queries supported by commodity graphics hardware [28]. Using occlusion queries, one can quickly obtain the number of visible pixel coverage of an articulated body C on the screen. However, this GPUs-based approach takes a linear time with respect to the number of links in C. More specifically, we get the number of visible pixels

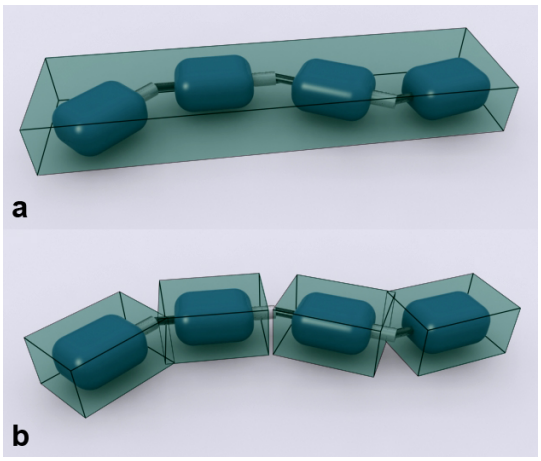

Figure 5: **Visibility estimation from bounding volumes** *The visibility estimation $\widetilde{N}_{\mathrm{M}}(C)$ is taken as the minimum of two projections.* **a***: Projected bounding volume of the entire articulated body.* **b***: Sum of the projected bounding volumes of individual rigid bodies.*

Table 1: **Comparisons** *From top to bottom: the original Featherstone's DCA, BVH- and occlusion query (OQ)-based methods.*

Method	Timing (msec)	# of Active joints
Featherstone	0.218	19
BVH	0.136	10
OQ	0.108	8

$\mathcal{N}_{\mathrm{M}}(C)$ for an articulated body C in two passes as follows:

1. Render the entire simulation scene including C and other objects in the environment.
2. Render each rigid body C_i (*i.e.* leaf-level node in the assembly tree) in C and use occlusion queries to determine the number of its visible pixels $\mathcal{N}_{\mathrm{M}}(C_i)$.
3. Recursively add up $\mathcal{N}_{\mathrm{M}}(C_i)$'s to get the number of visible pixels for the parent node of C_i's until we get $\mathcal{N}_{\mathrm{M}}(C)$.

By using a two-pass rendering method, we can get the number of visible pixels for C, occluded by the objects in the environment as well as by some of its own links in C (self-occlusion).

5.5 Comparisons between the two methods for calculating $\mathcal{N}_{\mathrm{M}}$

We have presented two methods to calculate $\mathcal{N}_{\mathrm{M}}$ earlier. Each method has its own cons and pros. The bounding-volume hierarchy-based

method has a sublinear time complexity with respect to the number of links in an articulated body to calculate $\mathcal{N}_{\mathrm{M}}$; it is the same as that of the original adaptive dynamics [10]. Moreover, since this method relies completely on CPU-based computation, the dynamics process is completely decoupled from the rendering process. However, in this case, it is not straightforward to take into account occlusion between links. Moreover, $\mathcal{N}_{\mathrm{M}}$ can be overly conservative depending on the relative configurations between links. On the other hand, the GPU-based method provides a tight estimation of $\mathcal{N}_{\mathrm{M}}$ and can easily handle different types of visibility including occlusion and screen space clipping. However, it requires a linear time complexity with respect to the number of links. But, in practice, the linear time complexity is almost negligible on modern graphics hardware thanks to its rapid rasterization capability.

We evaluate these methods using the pendulum model consisting of 20 rigid bodies and 19 joints. In Table 1 (also in the accompanying video), we summarize the average timings and the number of active joints during the simulation using the original Featherstone's algorithm, BVH-based (Eq. (6)) and occlusion queries-based methods. Compared to the original Featherstone's algorithm, our methods consume 40-50% less time. The occlusion queries-based method further reduces the number of active joints by taking into account the inter-visibility between links.

6 Implementation and Results

We implemented our view-dependent dynamics algorithm using C++ and OpenGL graphics library (for occlusion queries). In this section, we demonstrate our algorithm on scenarios running on a 2.26GHz Intel Pentium M processor laptop with 2GB RAM under Windows XP. Notice that our view-dependent dynamics performs exactly like adaptive dynamics [10] except that the adaptivity is automatically determined by the visibility of the simulated bodies.

6.1 Applications

Swinging pendulum : a pendulum model consisting of three hundred rigid bodies swings because of gravity. View-dependent dynamics is applied to the swinging motion of a pendulum in two series of tests: one by varying the threshold value for motion metrics (Figure 8) and one by varying the viewer's distance to the pendulum (Figure 9).

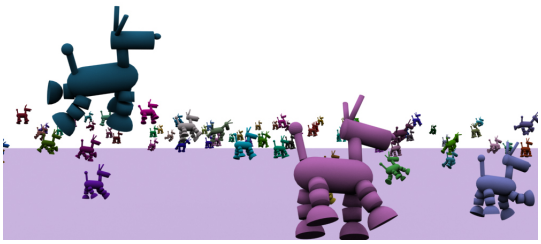

Figure 6: **View-dependent Simulation of 100 Swinging Toy Dogs.** *100 toy-like dogs consisting of 1600 rigid bodies are attached to virtual springs (not shown in the image) in space and simulate dynamics in a view-dependent manner. In this scene, as many dogs are occluded by other dogs, clipped against the viewport, or seen far from the viewer, the number of simulated, active joints is reduced from 1500 to 493 on average without incurring visual deterioration in the simulation. Each simulation frame takes 14 msec on average.*

As can be seen from the graphs, our method allows the user to finely tune the performance of the view-dependent dynamics by changing the error threshold or, for a given threshold, to benefit from the automatic simplification and corresponding speed-up when the distance to the viewer varies.

Haptic-enabled dog puppet: a toy-like dog model consisting of sixteen rigid bodies is interactively manipulated using a haptic interface (Figure 2.**a**). We use Sensable's Omni haptic device and map its end-effector to a virtual control stick attached to the toy dog by virtual strings. As the user interactively controls the toy dog, some links of the dog can be hidden by objects in the environment (Figure 2.**b**). Our view-dependent algorithm automatically rigidifies these links (Figure 2.**c**).

Hanging toy dogs: 100 toy-like dog models are attached to springs whose other ends are fixed in space. Initially, the dogs are twisted from the equilibrium state in order to create an initial rotational velocity. Then, the dogs are released and create dynamics simulation (Figure 6). During the simulation, a random torque is intermittently applied to the dogs.

Falling character: a character consisting of twenty-nine rigid bodies falls on a floor due to gravity. As the viewer moves away from the character, the view-dependent dynamics automatically rigidifies some joints (Figure 1) but preserves the overall look of the motion (Figure 7).

Please refer to the video to see the corresponding

motions.

6.2 Limitations

The approach presented in this paper has essentially two limitations:

- Our view-dependent metric is only semi-predictive. We do not currently have a (not overly) conservative way to bound the visual error caused by a simplification. We only estimate this error a priori, by making an assumption on the effect of a joint acceleration on the global articulated-body motion.
- Because of the way we combine acceleration metrics with visibility estimations, the error threshold set by the user is not as intuitive as it should be. However, the graceful, "continuous" degradation of the dynamics produced by our algorithm when the threshold or distance vary makes it relatively easy to choose this parameter.

7 Conclusion and Future Work

In this paper, we have introduced a method for view-dependent simulation of articulated-body dynamics. We have first discussed the general problem of simplifying a simulation based on visual criteria, a question which seems to have received relatively little attention in the past. We have showed how this problem raises a number of new challenges, even though it is strongly related to the well-known geometric simplification problem. Focusing on articulated-body dynamics, we have proposed *semi-predictive motion metrics*, which combine predictive error metrics based on joint accelerations with visibility estimations. The metrics, based on bounding volume hierarchies or graphics hardware visibility queries, allow us to automatically simplify a simulation based on visual criteria. We have demonstrated our approach in several scenarios and showed that our approach allows a user to finely trade between visual quality and performance. The dynamics are gracefully simplified as the distance to the viewer, or the error threshold, is increased.

Besides addressing the limitations mentioned above, we would now like to consider perceptual factors (*e.g.* [29]). In particular, we would like to perform user studies and examine how these studies could lead to more general view-dependent simulation methods.

References

[1] Roy Featherstone. A divide-and-conquer articulated body algorithm for parallel o(log(n)) calculation of rigid body dynamics. part 1: Basic algorithm. *International Journal of Robotics Research 18(9):867-875*, 1999.

[2] Roy Featherstone. A divide-and-conquer articulated body algorithm for parallel o(log(n)) calculation of rigid body dynamics. part 2: Trees, loops, and accuracy. *International Journal of Robotics Research 18(9):876-892*, 1999.

[3] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr. Dynamic real-time deformations using space and time adaptive sampling. *In Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001.

[4] E. Grinspun, P. Krysl, and P. Schroeder. Charms: a simple framework for adaptive simulation. *ACM Transactions on Graphics, 21(3)*, 2002.

[5] L. Li and V. Volkov. Cloth animation with adaptively refined meshes. *In Proceedings of the Twenty-eighth Australasian conference on Computer Science*, 2005.

[6] F. Losasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. *ACM Transactions on Graphics (SIGGRAPH 2004 Proceedings)*, 2004.

[7] K. Ward, M.C. Lin, L. Joohi, S. Fisher, and D. Macri. Modeling hair using level-of-detail representations. *In Proceedings of Computer Animation and Social Agents*, 2003.

[8] F. Bertails, T.-Y. Kim, M.-P. Cani, and U. Neumann. Adaptive wisp tree - a multiresolution control structure for simulating dynamic clustering in hair motion. *In Proceedings of ACM-SIGGRAPH/Eurographics Symposium on Computer Animation*, 2003.

[9] D. O'Brien, S. Fisher, and M. C. Lin. Automatic simplification of particle system dynamics. *In Proceedings of Computer Animation*, 2001.

[10] S. Redon, N. Gallopo, and M. C. Lin. Adaptive dynamics of articulated bodies. *In ACM Transactions on Graphics (SIGGRAPH 2005), 24(3)*, 2005.

[11] D. A. Carlson and J. K. Hodgins. Simulation levels of detail for real-time animation. *In Proceedings of Graphics Interface*, 1997.

[12] F. Perbet and M.-P. Cani. Animating prairies in real-time. *In Proceedings of the 2001 symposium on Interactive 3D graphics*, 2001.

[13] S. Chenney and D. Forsyth. View-dependent culling of dynamic systems in virtual environments. *Proceedings of the 1997 symposium on Interactive 3D graphics*, 1997.

[14] S. Chenney, J. Ichnowski, and D. A. Forsyth. Dynamics modeling and culling. *IEEE Computer Graphics and Applications*, 1999.

[15] J. Beaudoin and J. Keyser. Simulation levels of detail for plant motion. *In Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2004.

[16] C. O'Sullivan and J. Dingliana. Collisions and perception. *ACM Transactions on Graphics, 20(3)*, 2003.

[17] Carol O'Sullivan, John Dingliana, Thanh Giang, and Mary K. Kaiser. Evaluating the visual fidelity of physically based animations. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 527–536, New York, NY, USA, 2003. ACM Press.

[18] J. Harrison, R. A. Rensink, and M. Van De Panne. Obscuring length changes during animated motion. *ACM Transactions on Graphics 23(3)*, 2004.

[19] C. O'Sullivan. Collisions and attention. *ACM Transactions on Applied Perception*, 2005.

[20] Ronen Barzel, John F. Hughes, and Daniel N. Wood. Plausible motion simulation for computer graphics animation. In *Proceedings of the Eurographics workshop on Computer animation and simulation '96*, pages 183–197, New York, NY, USA, 1996. Springer-Verlag New York, Inc.

[21] Stephen Chenney and D. A. Forsyth. Sampling plausible solutions to multi-body constraint problems. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 219–228, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[22] Christopher D. Twigg and Doug L. James. Many-worlds browsing for control of multibody dynamics. *ACM Trans. Graph.*, 26(3):14, 2007.

[23] H. Hoppe. View-dependent refinement of progressive meshes. *ACM Transactions on Graphics (SIGGRAPH 1997 Proceedings)*, 1997.

[24] D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner. Level of detail for 3d graphics. *Morgan Kaufmann Publishers*, 2003.

[25] C. H. Lee, A. Varshney, and D. W. Jacobs. Mesh saliency. *In ACM Transactions on Graphics (SIGGRAPH 2005), 24(3)*, 2005.

[26] S. Redon and M. C. Lin. An efficient, error-bounded approximation algorithm for simulating quasi-statics of complex linkages. *In Computer-Aided Design, 38, pp. 300-314, Elsevier*, 2006.

[27] S. Gottschalk, M. C. Lin, and D. Manocha. Obbtree: a hierarchical structure for rapid interference detection. *In ACM Transactions on Graphics (SIGGRAPH 1996)*, 1996.

[28] NVIDA. *SDK White Paper - Occlusion Query, Checking for Hidden Pixels*. NVIDIA, 2004.

[29] P. S. A. Reitsma and N. S. Pollard. Perceptual metrics for character animation: Sensitivity to errors in ballistic motion. *ACM Transactions on Graphics (SIGGRAPH 2003 Proceedings)*, 22(3):537–542, 2003.

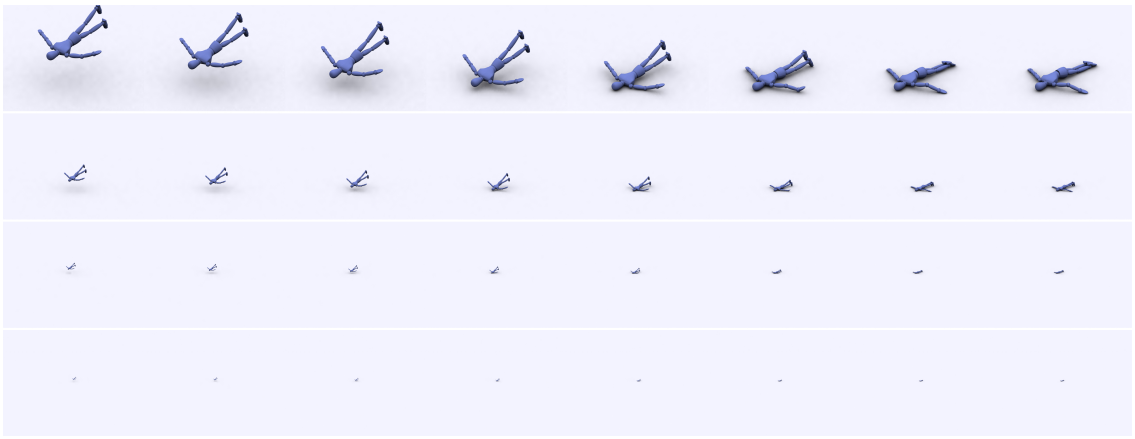

Figure 7: **View-dependent dynamics simplification of a falling character.** *Our algorithm automatically simplifies the dynamics of the falling character while preserving the overall visual aspect of the impact* (e.g. *legs motion, see also Fig. 1 for a close-up on the final frames.).*

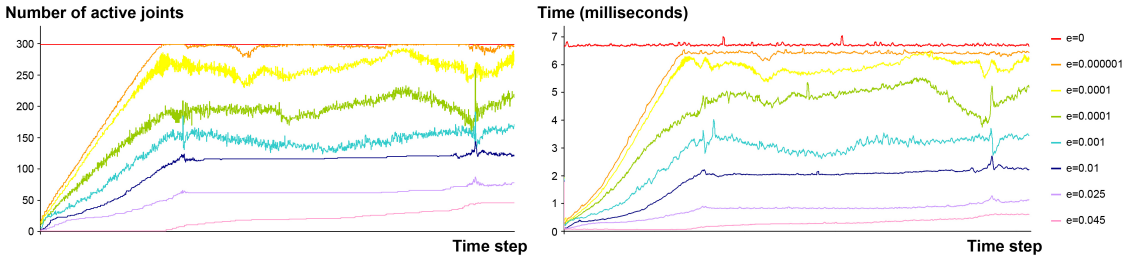

Figure 8: **Performance of our algorithm depending on visual error thresholds.** *As the visual error threshold e increases, the number of active nodes is automatically decreased by the view-dependent algorithm (***left***), and the computational cost is reduced (***right***).*

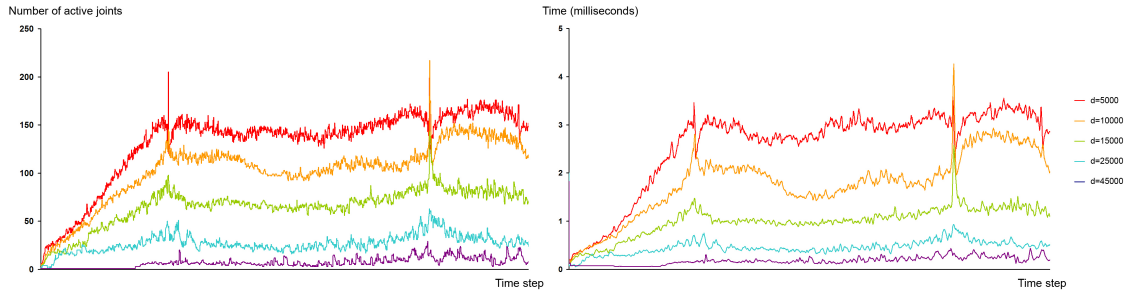

Figure 9: **Performance of our algorithm depending on the viewer's distance.** *Our algorithm automatically decreases the number of active joints when the distance d between the viewer and the pendulum increases (***left***), reducing the computational cost of the simulation (***right***).*