Controlling Physics-Based Characters Using Soft Contacts

Sumit Jain C. Karen Liu

Georgia Institute of Technology*

Abstract

In this paper, we investigate the impact of the deformable bodies on the control algorithms for physically simulated characters. We hypothesize that ignoring the effect of deformable bodies at the site of contact negatively affects the control algorithms, leading to less robust and unnatural character motions. To verify the hypothesis, we introduce a compact representation for an articulated character with deformable soft tissue and develop a practical system to simulate two-way coupling between rigid and deformable bodies in a robust and efficient manner. We then apply a few simple and widely used control algorithms, such as pose-space tracking control, Cartesian-space tracking control, and a biped controller (SIMBICON), to simulate a variety of behaviors for both full-body locomotion and hand manipulation. We conduct a series of experiments to compare our results with the motion generated by these algorithms on a character comprising only rigid bodies. The evaluation shows that the character with soft contact can withstand larger perturbations in a more noisy environment, as well as produce more realistic motion.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: character animation, articulated rigid body, deformable body, linear complementarity problem

Links: DL PDF

1 Introduction

One of the fundamental simplifications that researchers in physics-based human motion synthesis make, is that motion is the product of an articulated rigid body system with actuated joints representing bones and active skeletal muscles. On the surface this abstraction does capture the most fundamental aspects of the human musculoskeletal system. Utilizing this assumption, researchers have developed several control algorithms that can synthesize movement for various tasks like balance and walking. Although these controllers work well in their specific problem domain, they still cannot achieve the same level of agility the human body displays.

In this paper we revisit the fundamental assumption that an articulated rigid body system, by itself, captures the fundamental properties that enable human-like motion. We focus on one aspect of the motion that is not captured by this simplified model: the contact with the environment primarily occurs through the soft tissue. This factor comes into play in any situation where there is a collision between the character and the environment. Collisions between rigid bodies usually result in sporadic contact points and highly discontinuous pressure distribution. Although a character consist of only rigid bodies is ideal for efficiently simulating human movement, we postulate that the simplified rigid contact model inadvertently increases the difficulty in controller design and results in unrealistic motion.

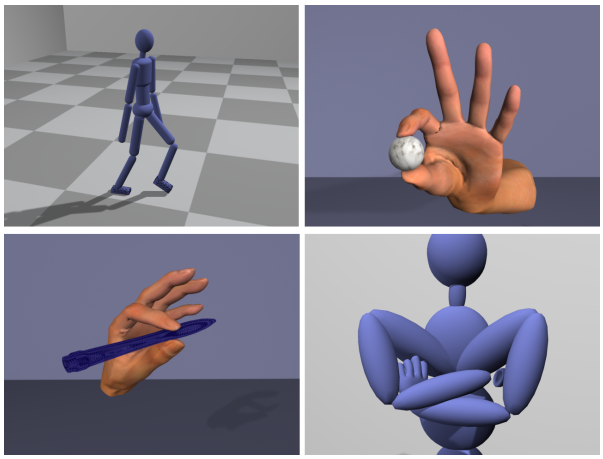

Figure 1: *Various controllers for character animation can be improved by simulating soft tissue deformation at the site of contact.*

The primary contribution of this paper is to demonstrate that simple control strategies coupled with the simulation of soft tissue deformation at the site of contact can achieve very robust and realistic motion. We develop a practical system that allows us to simulate two-way coupling between rigid and deformable bodies in a robust and efficient manner. We then apply a few simple and widely used control algorithms, such as pose-space tracking control, Cartesian-space tracking control, and SIMBICON, to simulate a variety of both full-body locomotion and hand manipulation. The resulting motions are compared with the motion generated by these algorithms on a character comprising only of rigid bodies. These simple controllers demonstrate that the character with soft contacts can withstand larger perturbations in a more noisy environment, without the need of designing more sophisticated control algorithms.

Simulating deformable bodies can be achieved in a few different ways and the design choice often has to balance the required accuracy and performance. We hypothesize that the accuracy offered by sophisticated but expensive methods, such as Finite Element Method (FEM) is unnecessary for our application for two reasons. First, unlike most previous work that simulates deformation of complex volumetric meshes for aesthetic purpose, the primary goal of our work is to produce deformation for more physically correct contacts. Second, average human body deforms marginally due to the support of bones. In particular, the deformation due to contacts is typically small and localized. We take advantage of these properties to design a simple and accurate model that only computes the surface of deformable bodies, rather than the entire volume.

To this end, we introduce a new representation for human skele-

*email:{sumit,karenliu}@cc.gatech.edu

ton consist of an articulated rigid body system, in which each rigid body is surrounded by a set of point masses representing the surface of flesh. Each point mass is attached to the rigid body as a child link with three translational degrees of freedom (DOFs). The dynamics of bones and flesh can be expressed in a unified manner by Lagrangian equations of motion in the generalized coordinates. This compact representation allows us to *soften* or *harden* any part of flesh at any time, by simply adding or removing the translational DOFs of involved point masses, without switching dynamic regimes or introducing any instability. Based on this flexible representation, we develop an efficient system where only the site of collision needs to be simulated as deformable body while the rest of the character remains rigid.

2 Related Work

Recent work in physics-based character animation explored a variety of approaches to develop more robust virtual characters in a dynamically changing environment. Despite the differences in control algorithms, one common focal point of these methods is the improvement of contact control mechanisms. For example, Yin et al.[2007] used the position and the velocity of the center of mass to continuously modulate the contact point of the next step. Abe et al. [2007] formulated a quadratic program to solve for optimal joint torques subject to frictional contact constraints. Muico et al. [2009] developed an online method to adapt the idealized control policy based on the current contact situation. Mordatch et al. [2010] planed ground contact positions and the foot trajectory of the center of pressure. Lee et al. [2010] showed that robust feedback controller can be achieved by carefully synchronizing the reference trajectory at contact changes. Similarly, in hand animation, Kry and Pai [2006] accounted for joint compliances due to contact. Liu [2009] optimized the contact positions and forces to synthesize detailed manipulation. Our motivation is similar in that we seek to create more realistic human motion through a better understanding of the contact phenomenon, but we take a drastically different approach. Instead of improving the control algorithm, our method aims to improve the physical realism of contact by simulating the contact points as deformable bodies rather than rigid bodies.

Creating body deformation for an articulated figure is an important and practical problem in computer animation. Common skeleton-driven techniques, such as skeleton-subspace-deformation [Magnenat-Thalmann et al. 1988; Maya], have been widely adopted by graphics practitioners. These basic methods can be enhanced by adding pose examples [Lewis et al. 2000; Kry et al. 2002] or additional degrees of freedom [Wang and Phillips 2002; Mohr and Gleicher 2003]. Data-driven methods based on scanned data [Allen et al. 2002] or dense motion capture data [Park and Hodgins 2006] can also create detailed body deformation driven by skeletal motion. These interpolation-based methods are able to produce visually appealing secondary motion. In contrast, our goal is to investigate the impact of deformations caused by collisions on control strategies for physically simulated characters.

One promising way to achieve realistic deformation at the site of contact is to apply physics-based modeling and simulation of skin layer around the skeleton. Earlier work has used mass-spring systems to synthesize deformable skin wrapped around the kinematic articulated figure [Gourret et al. 1989; Turner and Thalmann 1993]. Gourret et al. [1989] showed that realistic hand deformation in contact with an elastic object can be computed by a numerical method based on FEM. More recently, Pauly et al. [2004] used a quasi-rigid model to simulate small deformations at the site of contact to improve the robustness of contact resolution for point cloud surface representations. Capell et al. [2002] introduced a framework for simulating skeleton-driven, elastically deformed characters. Their method used a coarse volumetric finite element mesh to represent the deformation of skin, driven by the underlying skeleton motion. Their results highlighted large secondary motion due to inertia rather than the impact of collision. In addition, the skeletal motion was completely pre-scripted and the effect of skin movement did not affect the skeletal motion.

Our work is most relevant to two methods that consider two-way coupling between the skin and skeleton. Kim and Pollard [2011a; 2011b] described an efficient coupled system using meshing embedding with FEM. Their method leverages reduced deformable models and linear-time algorithms for rigid body dynamics to achieve real-time performance. The primary focus of their work is to develop an interactive user interface to control skeleton-driven deformable characters. In contrast, our work aims to investigate the effect of deformable contact on robustness of the control algorithms. Therefore, we choose to implement a more accurate LCP contact model instead of the penalty-based contact model used in their work. The computational cost of LCP motivated us to design a more compact representation than FEM for deformable bodies. Galoppo et al. [2007] introduced a fast formulation to compute skin displacement due to dynamic interplay with bones. They proposed a very efficient but specialized contact model by decoupling skeleton and skin computations using an approximated mass matrix. We believe with an additional implementation of a more accurate contact model, either of the above methods for simulating deformable bodies will be suitable for our applications.

Modeling soft bodies has also generated significant interest in robotics due to its wide range of potential applications. When a full-body humanoid robot moves in an unknown environment or interacts with humans in an unstructured setting, the motion of the robot must be stable and compliant to protect the robot's structure and ensure humans' safety. Researchers have demonstrated that integrating a compliant sole or shock absorbing materials under a humanoid robot's foot improves the stability of dynamic biped walking [Chardonnet et al. 2008; Yamaguchi et al. 1995]. Much research effort has also focused on emulating the compliance of anthropomorphic hands. Unlike rigid linked robots, soft robotic hands can conform better to the manipulated objects , enabling more sophisticated tasks and improving dexterity and robustness [Cutkosky and Kao 1989; Xydas and Kao 1998]. Our work is inspired by this line of robotic research, but we aim to create a unified simulation framework for controlling a variety of human movements.

3 Coupled dynamics

Our representation for human skeleton comprises of articulated rigid bodies, each of which is surrounded by a set of point masses representing the surface of the deformable flesh. In this section, we describe the equations of motion coupling the articulated rigid bodies and the deformable surface.

3.1 Articulated rigid body dynamics

The equations of motion of an articulated rigid body system parametrized by generalized coordinates $\mathbf{r}$, are given by:

$$M(\mathbf{r})\ddot{\mathbf{r}} + C(\mathbf{r},\dot{\mathbf{r}})\dot{\mathbf{r}} + \mathbf{g}(\mathbf{r}) = \boldsymbol{\tau} + J_c(\mathbf{r})^T \mathbf{f}_c \tag{1}$$

where M is the mass matrix, C is the Coriolis matrix, $\mathbf{g}$ are the gravitational forces and $\boldsymbol{\tau}$ are the applied generalized forces. The first six values in $\boldsymbol{\tau}$, which correspond to the global rotation and translation of the system, are zero, resulting in an under-actuated system. $\mathbf{f}_c$ and J_c are the contact force and the Jacobian respectively

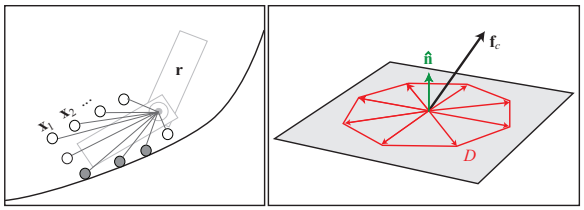

Figure 2: *Left: An articulated rigid body system coupled with deformable surface at the site of contact. Solid dots indicate the active DOFs. Right: A contact force* $\mathbf{f}_c$ *represented by a normal force and a tangent force in coordinates defined by a matrix D*

for a single contact point. M, C, $\mathbf{g}$ and J_c are non-linear functions of $\mathbf{r}$ while C is linear in $\dot{\mathbf{r}}$. It is straightforward to add more contacts to the equation; for clarity, we describe our equations with only one contact.

3.2 Deformable body dynamics

We define the surface of a deformable body as a 3D manifold triangle mesh formed by a set of point masses at the vertices. The elastic forces applied on each point mass are modeled as linear spring forces. We measure two kinds of deformations and their corresponding restoring forces at each vertex v_i:

- Vertex deformation. For a vertex v_i, we measure the deformation from its rest position $\bar{\mathbf{x}}_i$ as $\mathbf{x}_i - \bar{\mathbf{x}}_i$. The corresponding restoring spring force with a spring stiffness k_v is written as:

$$\mathbf{f}_{1,i} = -k_v(\mathbf{x}_i - \bar{\mathbf{x}}_i) \tag{2}$$

- Edge deformation. The deformation of the edge e_{ij} connecting two vertices v_i and v_j is given by $(\mathbf{x}_i - \mathbf{x}_j) - (\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j)$. The corresponding restoring force applied on v_i with a spring stiffness k_e is written as:

$$\mathbf{f}_{2,i} = \sum_{j \in N(i)} -k_e\Big((\mathbf{x}_i - \mathbf{x}_j) - (\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j)\Big) \tag{3}$$

  where $N(i)$ denotes the subset of vertices of the mesh connected to the vertex v_i via an edge.

The force in Equation (2) attempts to keep each vertex at its rest position while the force in Equation (3) tries to maintain the relative position of the vertex with respect to its neighbors. These forces attempt to keep the mesh in its undeformed state and also penalize translation and rotation of the entire mesh in the defined frame of reference.

We collect the positions of all the vertices in a vector $\mathbf{x} \equiv ({\mathbf{x}_1}^T, \ldots, {\mathbf{x}_N}^T)^T$, where N is the number of vertices in the mesh. The restoring forces defined in Equation (2) and Equation (3) for all the vertices can be represented as the product of a sparse stiffness matrix K_x and the deformation $\mathbf{x} - \bar{\mathbf{x}}$ i.e. $\mathbf{f} = K_x(\mathbf{x} - \bar{\mathbf{x}})$. We add a velocity damping force using a damping matrix $K_{\dot{x}}$. The equations of motion for this linear system are written as:

$$M\ddot{\mathbf{x}} = -K_x(\mathbf{x} - \bar{\mathbf{x}}) - K_{\dot{x}}\dot{\mathbf{x}} - \mathbf{g} \tag{4}$$

where M is the diagonal mass matrix with diagonal entries corresponding to each vertex v_i as m_i and $\mathbf{g}$ are the gravitational forces.

3.3 Coupled equations of motion

The deformable body dynamics described above are defined for a fixed frame of reference. To drive the deformable body using a rigid body (node) of the articulated skeleton, we need to *attach* the frame of reference of the deformable body to the rigid node. In addition, we must ensure two-way dynamic coupling between the deformable body and the articulated rigid body system.

We augment the articulated body system with point masses corresponding to the vertices of the deformable body (Figure 2, Left). Each point mass is attached to the rigid node through a 3-degree-of-freedom (DOF) translation joint. The DOFs of this joint are the vertex coordinates $\mathbf{x}_i$. By expressing the DOFs of each point mass as a *child* link of the rigid body, we automatically set the frame of reference of the deformable body to the rigid body. This results in an augmented articulated body system that has a tree structure comprising of rigid nodes and point masses. The new set of DOFs or generalized coordinates is represented as $\mathbf{q} \equiv (\mathbf{r}^T, \mathbf{x}^T)^T$. The dynamics for this articulated system is similar to Equation (1) and can be represented as:

$$M\ddot{\mathbf{q}} + C\dot{\mathbf{q}} + \mathbf{g} = \boldsymbol{\tau} + J_c^T\mathbf{f}_c \tag{5}$$

These new matrices M and C and the vector $\mathbf{g}$ can be expressed as:

$$M = \begin{bmatrix} M_r + \bar{M}_x & M_{rx} \\ M_{rx}^T & M_x \end{bmatrix}, C = \begin{bmatrix} C_r + \bar{C}_x & C_{rx} \\ C_{rx}^T & \mathbf{0} \end{bmatrix}, \mathbf{g} = \begin{pmatrix} \mathbf{g}_r + \bar{\mathbf{g}}_x \\ \mathbf{g}_x \end{pmatrix} \tag{6}$$

where M_r and C_r are the mass and Coriolis matrices and $\mathbf{g}_r$ are the gravitation forces defined in Equation (1), M_x is the mass matrix and $\mathbf{g}_x$ are the gravitation forces defined in Equation (4). Equation (6) also introduces a few new coefficients. M_{rx} and C_{rx} are dense matrices coupling the rigid and deformable DOFs and $\bar{M}_x$, $\bar{C}_x$ and $\bar{\mathbf{g}}_x$ are the contributions of the point masses to the dynamic equations of rigid nodes. These matrices can be derived from standard multi-body system dynamics. The generalized forces $\boldsymbol{\tau}$ are given by:

$$\boldsymbol{\tau} = \begin{pmatrix} \boldsymbol{\tau}_r \\ \boldsymbol{\tau}_x \end{pmatrix} = \begin{pmatrix} \boldsymbol{\tau}_r \\ -K_x(\mathbf{x} - \bar{\mathbf{x}}) - K_{\dot{x}}\dot{\mathbf{x}} \end{pmatrix} \tag{7}$$

where $\boldsymbol{\tau}_r$ are the generalized forces in Equation (1). For an actively controlled articulated system, $\boldsymbol{\tau}_r$ is usually computed by an external controller.

Adaptive deformable simulation. A mesh with large number of vertices makes the dynamic system very expensive to compute. Since we are interested in simulating the deformable body at the site of contact, we only simulate a subset of the vertices at the site of contact and treat the rest of the surface rigid. Intuitively, this implies that we change the stiffness of the rest of the point masses such that they provide the constraint forces required to keep the surface rigid. Now, given the contact points on the surface, we traverse a p-ring neighborhood of the vertices and mark these visited vertices. We simulate those marked vertices until they are no longer in contact and reach the equilibrium at their rest positions. This adaptive scheme for the deformable body simulation is a practical solution for situations involving small and stiff deformations. However, we also need to consider the dynamic contribution of the point masses not simulated, because their effect on the mass matrix in Equation (6) changes over time due to their dependency on the rigid pose $\mathbf{r}$. To efficiently compute the mass matrix, we precompute the contribution of the point masses in their rest positions relative to the local frame of the parent rigid body. This contribution effectively changes the mass and inertia of the parent rigid body. At each time step during the simulation, since we visit all the simulated point masses that are potentially not at their rest positions, we simply subtract their rest pose contribution to the mass matrix from the computed value.

Discrete equations. In our implementation, we discretize the equations of motion in time. Quantities at any time t_k are subscripted by k. For clarity, we time-subscript the quantities only to disambiguate; otherwise they are assumed to be evaluated at time t_k. We use backward and central differences to discretize velocity and acceleration respectively using the time step h as:

$$\begin{aligned} \dot{\mathbf{q}}_k &\equiv \frac{\mathbf{q}_k - \mathbf{q}_{k-1}}{h} \\ \ddot{\mathbf{q}}_k &\equiv \frac{\mathbf{q}_{k+1} - 2\mathbf{q}_k + \mathbf{q}_{k-1}}{h^2} = \frac{\dot{\mathbf{q}}_{k+1} - \dot{\mathbf{q}}_k}{h} \end{aligned} \tag{8}$$

Substituting the above in Equation (5) and rearranging terms, we arrive at:

$$M(\dot{\mathbf{q}}_{k+1} - \dot{\mathbf{q}}_k) - hJ_c{}^T\mathbf{f}_c = h(\boldsymbol{\tau} - C\dot{\mathbf{q}}_k - \mathbf{g}) \tag{9}$$

Large spring stiffness or damping in Equation (7) may cause instabilities for large time steps. Therefore, we apply the spring forces in an implicit manner i.e. the spring forces are evaluated at time t_{k+1} by using Equation (8) as:

$$\begin{aligned} \hat{\boldsymbol{\tau}}_x &= -K_x(\mathbf{x}_{k+1} - \bar{\mathbf{x}}) - K_{\dot{x}}\dot{\mathbf{x}}_{k+1} \\ &= -K_x(\mathbf{x}_k + h\dot{\mathbf{x}}_{k+1} - \bar{\mathbf{x}}) - K_{\dot{x}}\dot{\mathbf{x}}_{k+1} \\ &= -K_x(\mathbf{x}_k - \bar{\mathbf{x}}) - (hK_x + K_{\dot{x}})\dot{\mathbf{x}}_k - (hK_x + K_{\dot{x}})(\dot{\mathbf{x}}_{k+1} - \dot{\mathbf{x}}_k) \end{aligned} \tag{10}$$

Replacing $\boldsymbol{\tau}_x$ with $\hat{\boldsymbol{\tau}}_x$ in Equation (9) and rearranging terms, we get:

$$\hat{M}(\dot{\mathbf{q}}_{k+1} - \dot{\mathbf{q}}_k) - hJ_c{}^T\mathbf{f}_c = h(\hat{\boldsymbol{\tau}} - C\dot{\mathbf{q}}_k - \mathbf{g}) \tag{11}$$

$$\text{where } \hat{M} = \begin{bmatrix} M_r + \bar{M}_x & M_{rx} \\ M_{rx}^T & M_x + h^2K_x + hK_{\dot{x}} \end{bmatrix}$$

$$\hat{\boldsymbol{\tau}} = \begin{pmatrix} \boldsymbol{\tau}_r \\ -K_x(\mathbf{x}_k - \bar{\mathbf{x}}) - (hK_x + K_{\dot{x}})\dot{\mathbf{x}}_k \end{pmatrix}$$

This semi-implicit system of equations is stable with respect to the deformable forces. Some of other instabilities due to large velocities can be tackled by making the Coriolis term implicit in $\dot{\mathbf{q}}$. Utilizing the linear dependence of C on $\dot{\mathbf{q}}$, we evaluate $C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}$ using the velocities at the next time step $\dot{\mathbf{q}}_{k+1}$:

$$\begin{aligned} C(\mathbf{q}_k,\dot{\mathbf{q}}_{k+1})\dot{\mathbf{q}}_{k+1} &= C(\mathbf{q}_k,\dot{\mathbf{q}}_k + \Delta\dot{\mathbf{q}})(\dot{\mathbf{q}}_k + \Delta\dot{\mathbf{q}}) \\ &= (C(\mathbf{q}_k,\dot{\mathbf{q}}_k) + C(\mathbf{q}_k,\Delta\dot{\mathbf{q}}))\,(\dot{\mathbf{q}}_k + \Delta\dot{\mathbf{q}}) \\ &\approx C(\mathbf{q}_k,\dot{\mathbf{q}}_k)\dot{\mathbf{q}}_k + C(\mathbf{q}_k,\dot{\mathbf{q}}_k)\Delta\dot{\mathbf{q}} + C(\mathbf{q}_k,\Delta\dot{\mathbf{q}})\dot{\mathbf{q}}_k \\ &= C(\mathbf{q}_k,\dot{\mathbf{q}}_k)\dot{\mathbf{q}}_k + 2C(\mathbf{q}_k,\dot{\mathbf{q}}_k)(\dot{\mathbf{q}}_{k+1} - \dot{\mathbf{q}}_k) \end{aligned} \tag{12}$$

We substitute this result in Equation (11). The only change in the equation is the mass matrix:

$$\hat{M} = \begin{bmatrix} M_r + \bar{M}_x & M_{rx} \\ M_{rx}^T & M_x + h^2K_x + hK_{\dot{x}} \end{bmatrix} + 2hC$$

Given the current state $(\mathbf{q}_k,\dot{\mathbf{q}}_k)$, the only unknowns in Equation (11) are $\dot{\mathbf{q}}_{k+1}$ in the absence of contact. We can then directly integrate to the next state after we solve for $\dot{\mathbf{q}}_{k+1}$. To incorporate the effect of contact, however, we need to consider additional constraints described in the next section.

4 Contact model

Similar to the numerous options for modeling deformable bodies, there are many existing methods for modeling collisions as well. We choose a model that accurately simulates unilateral contact forces at the points of contact. In addition, we consider the Coulomb friction model that allows for slipping. These conditions for the contact force and the contact point can be formulated into a linear complementarity problem (LCP), as described in [Anitescu and Potra 1997].

The contact force is represented by its normal and tangential components, $\mathbf{f}_c = f_n\hat{\mathbf{n}} + D\boldsymbol{\beta}$, where $\hat{\mathbf{n}}$ is the normal vector at contact and D is a matrix with its columns as the vectors that span the tangent plane at contact. The contact force is parametrized by a scalar f_n and a vector $\boldsymbol{\beta}$ in the normal and the tangential space (Figure 2, Right).

Let $\mathbf{p}(\mathbf{q})$ be the coordinates of the contact point. The point velocity is linearly related to the generalized velocity as $\dot{\mathbf{p}} = J_c\dot{\mathbf{q}}$. In the normal direction, the projection of the point velocity and the normal force have to satisfy the following condition to prevent penetration of the surface and enforce work-less and unilateral properties of the contact force:

$$\begin{aligned} &(\dot{\mathbf{p}}_{k+1}\cdot\hat{\mathbf{n}}) \perp f_n \\ \text{or } &(\hat{\mathbf{n}}^T J_c\dot{\mathbf{q}}_{k+1}) \perp f_n \end{aligned} \tag{13}$$

where the notation $\mathbf{a} \perp \mathbf{b}$ implies the complementarity condition $\mathbf{a}^T\mathbf{b} = 0$ with $\mathbf{a},\mathbf{b} \geq \mathbf{0}$. Note that non-negativity ensures component-wise complementarity $a_i \perp b_i$. In the tangential direction, we introduce a parameter λ that represents the relative tangential movement for the point at contact. If there is slipping ($\lambda > 0$), the tangential force should lie on the boundary of the polyhedra in the direction opposite to the movement. If the contact is static, the magnitude of the tangential force should be bound by the normal force modulated by friction coefficient. These two conditions can be enforced by:

$$\begin{aligned} &(\lambda\mathbf{e} + D^T J_c\dot{\mathbf{q}}_{k+1}) \perp \boldsymbol{\beta} \\ \text{and } &(\mu f_n - \mathbf{e}^T\boldsymbol{\beta}) \perp \lambda \end{aligned} \tag{14}$$

where μ is the friction coefficient and $\mathbf{e}$ is a vector of ones. We refer the reader to [Stewart and Trinkle 1996; Anitescu and Potra 1997] for a more detailed explanation of the complementarity conditions.

Let $\mathbf{z} = (f_n,\boldsymbol{\beta}^T,\lambda)^T$. Rewriting Equation (11) with the parametrized contact force, we get:

$$\begin{bmatrix} \hat{M} & -(\hat{\mathbf{n}}^T J_c)^T & -(D^T J_c)^T & 0 \end{bmatrix} \begin{pmatrix} \dot{\mathbf{q}}_{k+1} \\ \mathbf{z} \end{pmatrix} + \mathbf{b} = \mathbf{0} \tag{15}$$

where $\mathbf{b} = -\hat{M}\dot{\mathbf{q}}_k - h(\hat{\boldsymbol{\tau}} - C\dot{\mathbf{q}}_k - \mathbf{g})$. Combining the above constraint with the complementarity conditions in Equation (13) and Equation (14), we get:

$$\begin{bmatrix} \hat{M} & -(\hat{\mathbf{n}}^T J_c)^T & -(D^T J_c)^T & 0 \\ \hat{\mathbf{n}}^T J_c & 0 & \mathbf{0} & 0 \\ D^T J_c & 0 & \mathbf{0} & \mathbf{e} \\ 0 & \mu & -\mathbf{e}^T & 0 \end{bmatrix} \begin{pmatrix} \dot{\mathbf{q}}_{k+1} \\ \mathbf{z} \end{pmatrix} + \begin{pmatrix} \mathbf{b} \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{w} \end{pmatrix}$$

$$\text{with } \mathbf{w} \perp \mathbf{z} \tag{16}$$

Equation (16) represents a mixed LCP problem with $\mathbf{w},\mathbf{z}$ and $\dot{\mathbf{q}}_{k+1}$ as unknowns. Given the current state of the character $(\mathbf{q}_k,\dot{\mathbf{q}}_k)$ and the contact points, we solve this problem to get the generalized velocities $\dot{\mathbf{q}}_{k+1}$ for the next time step t_{k+1} and compute the new state as $(\mathbf{q}_k + h\dot{\mathbf{q}}_{k+1},\dot{\mathbf{q}}_{k+1})$.

5 Implementation of controllers

Without active control forces $\boldsymbol{\tau}_r$ applied at the actuated joints of the skeleton, the system described in previous sections is completely passive. To demonstrate the utility of our coupled dynamic system, we implement the following control algorithms based on existing controllers widely adopted in computer animation and robotics. Their applications in locomotion and manipulation are demonstrated in Section 6.

5.1 Pose-space tracking control

We formulate an implicit PD control for tracking a given pose or a sequence of poses for the articulated skeleton. Let $(\mathbf{r}_k, \dot{\mathbf{r}}_k)$ denote the current state for the rigid skeleton and $\bar{\mathbf{r}}$ be the desired pose. Further, let K_r and $K_{\dot{r}}$ be diagonal stiffness and damping matrices respectively. To provide more stability in the system, we calculate the feedback force based on the deviation of the *next* state from the desired pose, similar to Equation (10):

$$\begin{aligned}\boldsymbol{\tau}_r &= -K_r(\mathbf{r}_{k+1} - \bar{\mathbf{r}}) - K_{\dot{r}}\dot{\mathbf{r}}_{k+1} \\ &= -K_r(\mathbf{r}_k - \bar{\mathbf{r}}) - (hK_r + K_{\dot{r}})\dot{\mathbf{r}}_k - (hK_r + K_{\dot{r}})(\dot{\mathbf{r}}_{k+1} - \dot{\mathbf{r}}_k) \quad (17)\end{aligned}$$

Plugging $\boldsymbol{\tau}_r$ into Equation (11) and rearranging the equation, we modify $\hat{M}$ and $\hat{\boldsymbol{\tau}}$ in Equation (11) as:

$$\begin{aligned}\hat{M} &= \begin{bmatrix} M_r + \bar{M}_x + h^2K_r + hK_{\dot{r}} & M_{rx} \\ M_{rx}^T & M_x + h^2K_x + hK_{\dot{x}} \end{bmatrix} + 2hC \\ \hat{\boldsymbol{\tau}} &= \begin{pmatrix} -K_r(\mathbf{r}_k - \bar{\mathbf{r}}) - (hK_r + K_{\dot{r}})\dot{\mathbf{r}}_k \\ -K_x(\mathbf{x}_k - \bar{\mathbf{x}}) - (hK_x + K_{\dot{x}})\dot{\mathbf{x}}_k \end{pmatrix} \quad (18)\end{aligned}$$

5.2 Cartesian-space tracking control

Sometimes it is more effective to track a position in Cartesian-space rather than in the pose space. Let $\mathbf{p}_i(\mathbf{r})$ represent the world coordinates of a point on a rigid body. Let the desired position of this point be $\bar{\mathbf{p}}_i$. For small deviations, we can approximate it as $\Delta\mathbf{p}_i \equiv \mathbf{p}_i - \bar{\mathbf{p}}_i \approx J_i\Delta\mathbf{r}$, where J_i is the Jacobian evaluated at $\mathbf{p}_i$. With this approximation, we can simply use the pose-space PD control as described in Equation (18) to track the new desired pose as $\bar{\mathbf{r}} + \Delta\mathbf{r}$. The same approximation can be applied to tracking multiple Cartesian points:

$$\begin{bmatrix} J_1^T, & \ldots & , J_m^T \end{bmatrix}^T \Delta\mathbf{r} = \begin{pmatrix} \Delta\mathbf{p}_1^T, & \ldots & , \Delta\mathbf{p}_m^T \end{pmatrix}^T \quad (19)$$

Similarly, body orientation in Cartesian-space can be controlled using the same mechanism: $\Delta\omega_b = J_{\omega_b}\Delta\mathbf{r}$, where $\Delta\omega_b$ is the change in orientation of the body and J_{ω_b} is the Jacobian defined as $\omega_b = J_{\omega_b}\dot{\mathbf{r}}$.

5.3 Locomotion control

In addition to basic tracking controllers, we also apply our coupled dynamic system to a biped controller, SIMBICON [Yin et al. 2007], which has been adopted widely by other researchers in computer graphics community [Coros et al. 2008; Coros et al. 2010; Wang et al. 2009; Wang et al. 2010; Lee et al. 2010].

SIMBICON uses maximal coordinates to compute joint torques and employs Open Dynamics Engine (ODE) to solve for the simulation time step. The algorithm for advancing one time step in original SIMBICON can be summarized as:

STEP 1: Compute joint torques $\boldsymbol{\tau}_s$
STEP 2: Detect collisions
STEP 3: Create contacts to be solved for in ODE
STEP 4: Apply $\boldsymbol{\tau}_s$ to character in ODE
STEP 5: Advance one time step in ODE to get next state

Our method follows the SIMBICON algorithm except for the contact force handling. To compute the contact forces for our coupled dynamic system, we first need to convert the torques and the state from maximal coordinates used in SIMBICON to our generalized coordinates. We then solve for contact forces via Equation (16) and use them to override the contact forces solved by ODE. Our contact handling, summarized as follows, replaces the STEP 3 of SIMBICON algorithm.

STEP 3.1 Convert $\boldsymbol{\tau}_s$ to generalized torques $\boldsymbol{\tau}_r$
STEP 3.2 Convert state to generalized coordinates $(\mathbf{r}_k, \dot{\mathbf{r}}_k)$
STEP 3.3 Solve $(\mathbf{q}_{k+1}, \dot{\mathbf{q}}_{k+1})$ and $\mathbf{f}_c$ using Equation (16)
STEP 3.4 Apply $\mathbf{f}_c$ to character in ODE

Performance improvement. Although our contact handling for the coupled dynamic system inevitably increases the computation time, our semi-implicit scheme in $\hat{\boldsymbol{\tau}}$ potentially allows for a larger time step than the one used by SIMBICON (0.5ms), resulting in less frequent computation of Equation (16). However, using a smaller frequency for the contact handling poses problems to the rest of the simulation algorithm; infrequent updates of the deformable state and the contact forces can lead to inaccurate collision detection and dynamic inconsistency. To address this problem of asynchronous time updates, we introduce a mixed-frequency simulation algorithm.

Let h_s be the SIMBICON time step used in STEP 5 and $h_d = nh_s$ be the time step for solving Equation (16) in STEP 3.3 for some integer n. At time t_0, we execute STEP 3.1 to STEP 3.4; i.e. we solve Equation (16) and record the contact forces $\mathbf{f}_c$ and the new deformable state $(\mathbf{x}_{k+1}, \dot{\mathbf{x}}_{k+1})$. The new deformable state and contact forces are applied at time $t_0 + h_d$. For any time $t \in (t_0, t_0 + h_d)$, we interpolate the deformable state as $(\mathbf{x}_t, \dot{\mathbf{x}}_t) = (1-u)(\mathbf{x}_k, \dot{\mathbf{x}}_k) + u(\mathbf{x}_{k+1}, \dot{\mathbf{x}}_{k+1})$ where $u = (t - t_0)/h_d$. For the contact force during $t \in (t_0, t_0 + h_d)$, we could simply treat the deformable as a rigid body and use STEP 3 to compute the contact forces. However, this treatment largely reduces the overall effect of the coupled dynamic system. Instead, we take into account the impact of deformable body on the rigid DOFs as an *additional generalized force*, $\bar{\boldsymbol{\tau}}_x$, applied to the character before solving the ODE time update:

$$\bar{\boldsymbol{\tau}}_x = -(\bar{M}_x\ddot{\mathbf{r}} + M_{rx}\ddot{\mathbf{x}} + \bar{C}_x\dot{\mathbf{r}} + C_{rx}\dot{\mathbf{x}} + \bar{\mathbf{g}}_x) \quad (20)$$

Equation (20) can be derived from Equation (5) and Equation (6):

$$\begin{aligned}&(M_r + \bar{M}_x)\ddot{\mathbf{r}} + M_{rx}\ddot{\mathbf{x}} + (C_r + \bar{C}_x)\dot{\mathbf{r}} + C_{rx}\dot{\mathbf{x}} + (\mathbf{g}_r + \bar{\mathbf{g}}_x) = \boldsymbol{\tau}_r + J_c^T\mathbf{f}_c \\ \Rightarrow\ & M_r\ddot{\mathbf{r}} + C_r\dot{\mathbf{r}} + \mathbf{g}_r = \boldsymbol{\tau}_r + J_c^T\mathbf{f}_c - (\bar{M}_x\ddot{\mathbf{r}} + M_{rx}\ddot{\mathbf{x}} + \bar{C}_x\dot{\mathbf{r}} + C_{rx}\dot{\mathbf{x}} + \bar{\mathbf{g}}_x)\end{aligned}$$

Leaving out the last term in the RHS (i.e. $\bar{\boldsymbol{\tau}}_x$), the above equation represents the equations of motion of an articulated rigid body system with DOFs $\mathbf{r}$ (Equation (1)). The accelerations $\ddot{\mathbf{r}}$ and $\ddot{\mathbf{x}}$ for the duration $(t_0, t_0 + h_d)$ can be computed using their discretizations (Equation (8)) since we have both the velocities, $\dot{\mathbf{q}}_k$ and $\dot{\mathbf{q}}_{k+1}$ at times t_0 and $t_0 + h_d$ respectively. When the LCP is solved at time t_0, the new solved state is valid for time $t_0 + t_d$. For all the intermediate simulation time steps of SIMBICON, the state for the deformable body is linearly interpolated based on the states at time t_0 and $t_0 + t_d$. The LCP computation is much more expensive as compared to the SIMBICON simulation; hence solving LCP every n frames helps speed up the computation substantially.

Stability improvement. Because the applied torques $\boldsymbol{\tau}_s$ computed by SIMBICON are based on a very small time step, directly applying them to our formulation with a much larger time step can

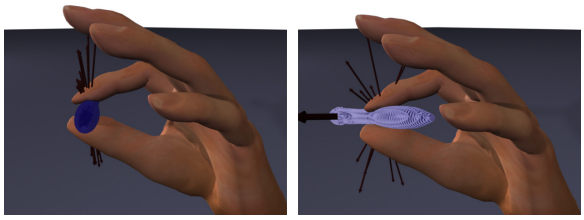

Figure 4: *Soft fingers enable a more robust pinch-grasp.*

sometimes lead to instability. To counter this, we add one term to the converted generalized torques $\boldsymbol{\tau}_r$ to approximate the effect of an implicit integration scheme. This modified $\boldsymbol{\tau}_r$ are only used to compute more stable contact forces in STEP 3.3, and we still use the original $\boldsymbol{\tau}_s$ in ODE forward simulation (STEP 4). We define a vector $\mathbf{k}_\tau$ such that $\boldsymbol{\tau}_r = K_\tau(\mathbf{r}_k + h_d\dot{\mathbf{r}}_\mathbf{k}) + \mathbf{k}_\tau$ for some chosen positive definite matrix K_τ. We now make an approximation by replacing $\dot{\mathbf{r}}_k$ with $\dot{\mathbf{r}}_{k+1}$ thus making the forces implicit since $\dot{\mathbf{r}}_{k+1}$ is unknown: $\boldsymbol{\tau}_r \approx K_\tau(\mathbf{r}_k + h_d\dot{\mathbf{r}}_{\mathbf{k+1}}) + \mathbf{k}_\tau$. The resulting deviation added to $\boldsymbol{\tau}_r$ equals $h_d K_\tau(\dot{\mathbf{r}}_{\mathbf{k+1}} - \dot{\mathbf{r}}_\mathbf{k})$ or $h_d^2 K_\tau \ddot{\mathbf{r}}$. Introducing this additional term produces more stable contact forces without altering the original torques generated by the SIMBICON controller.

6 Results

We tested results on controllers for biped locomotion and hand manipulation. All the results were produced on a single core of 2.7 GHz Intel i7. For each behavior, we applied the identical control algorithm to a character with soft tissue at the site of contact (soft character) and a character comprising only rigid bodies (rigid character). The motions of these two characters were compared. The results can be viewed in the accompanying video.

Biped locomotion controller Using the SIMBICON-based controller described in the previous section, we designed three experiments to evaluate the impact of soft contacts. We directly used the source code of SIMBICON [2007] and ODE for the control force computation and the forward simulation. The only modification by our method was the contact force handling.

In the first experiment, we applied large push forces to the character and compared the motion with and without soft contact simulation. SIMBICON is known for its robustness to external perturbations. However, when a strong push that induces a large torque, the rigid character tends to lose contact easily and fails to recover. In contrast, the deformable bodies allow our character to maintain more contact points on the ground with more evenly distributed pressure (Figure 3). Losing a few contact points due to the perturbation does not critically affect the balance state.

The second and third experiments focus on evaluating the controller under different sources of uncertainty. We first considered the noise in the motor control system of the character. We implemented the same simplified, biologically-inspired model as Wang et al. [2010]. This model adds noise to the joint torques produced by the controller at every time step. The noise is drawn from a Gaussian density with zero mean and a standard deviation depending on the magnitude of the joint torque and the strength of the joint. We used middle noise level (i.e. $\beta = 75$ defined in their paper) to test our controllers. As a result, the rigid character quickly becomes unstable when small pushes are applied. The soft character, on the other hand, is still able to maintain balance and withstand large pushes.

The last experiment evaluates the biped controller operating on a noisy surface. We first segmented the floor into small tiles of 5×5 cm^2. For each vertex, we added a random offset, uniformly sampled from a range of [0,2] cm, to its vertical and horizontal positions. We then reconstructed a bumpy surface based on the modified vertices. The controllers were tested on several randomly generated different floors from which we demonstrate three in the video. The uncertainty on the surface greatly affects the rigid character's ability to walk. In some cases it wanders to various locations and other cases it simply falls. The soft character is able to stay in the original course for all cases with small variations. A close-up observation shows that the deformable foot mesh of the soft character conforms better to the noisy surface and maintains more contact points when the character is pushed.

Although we have not implemented our contact model on biped controllers other than SIMBICON, we expect that our formulation can work with other biped control algorithms that employ LCP-like contact models. Based on our experiments with SIMBICON, our method can be applied in such way that the underlying control strategies, model coefficients, and numerical parameters all remain intact, while the controller enjoying more contact points and smoother transition of contact area. These benefits are particularly important for controlling under-actuated DOFs for biped characters. However, our method might not have significant improvement on biped controllers that already exploited contact force and timing [Muico et al. 2009; Lee et al. 2010].

Cartesian-space tracking controller We designed a manipulation controller based on tracking the center of pressure on the finger. The intended function of the controller is to flick a marble ball in the desired direction using the distal phalanx of the index finger and the thumb. The controller attempts to match the relative center of pressure between the thumb and the index finger to a desired vector, in addition to tracking an equilibrium pose.

As shown in the video, the rigid hand clearly has no control over the direction of the ball. In some cases, it fails to launch the ball because the loss of contact points occurs too early. The soft hand shows much more accurate control in different directions and the ball never slips off the fingers before the launch. We also tested the robustness of the controller by adding noise to the surface of the ball. By applying the same control forces several times, the fingers with soft contact manage to flick the ball in the similar directions, but the rigid fingers produce motions with huge variance. The key difference that leads to better dexterity is that the soft hand maintains more contact points and smoother movement of the center of pressure at all times.

Pose-space tracking controller We tested pose-space tracking control strategy on a human upper body and a human hand. For the human upper body, our goal is to track an arm-folding pose (Figure 1 lower right). Although it is not a difficult control problem, this particular pose is very difficult to simulate due to a large area of contact against multiple body parts. In our results, the rigid character immediately gets stuck when the hand collides into the opposite upper arm. In contrast, the soft character is able to fold her arms with the hands and upper arms brushing against each other and smoothly moving into their own desired positions.

We also developed a controller capable of pinch-grasping a thin object. We employed PD controllers to track an equilibrium pose such that a pen can be held horizontally in between the index finger and the thumb (Figure 4). Without any perturbation, both the rigid hand and the soft hand can successfully hold the pen. However, when external forces are applied to the pen, the rigid hand quickly loses contacts and drops the pen while the soft hand can withstand perturbation in any direction. The total contact forces applied on the pen are comparable between two hands, but the pressure distribution is

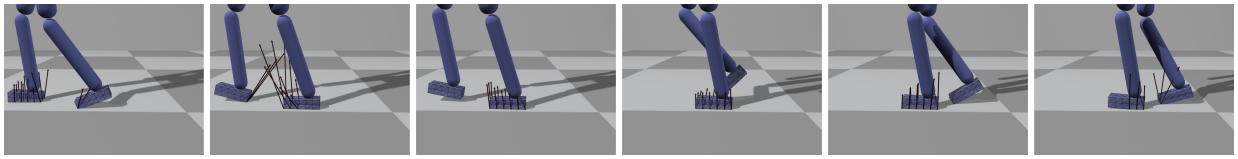

Figure 3: *Simulating deformable body at the site of contact results in more contact points and smoother center of pressure.*

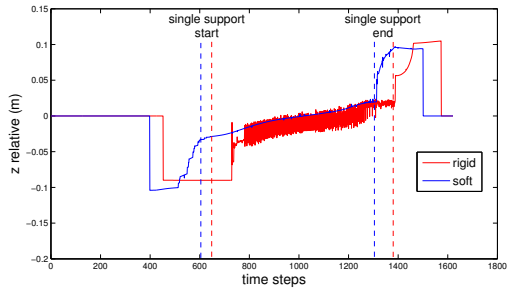

Figure 5: *Comparison of the center of pressure on the left foot in the direction of heel to toe for one step (frames 400-1500)*

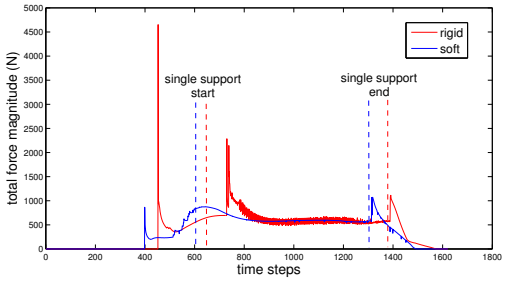

Figure 6: *Comparison of the magnitude of the total contact force on the left foot for one step (frames 400-1500).*

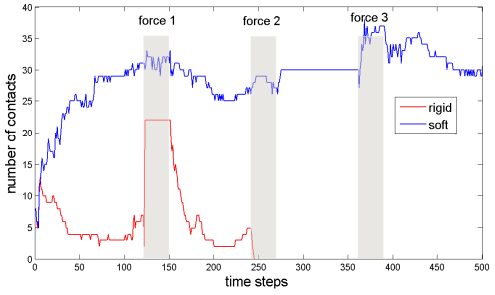

Figure 7: *Comparison of the number of contacts from the pinch-grasp example.*

much smoother on the soft hand.

We summarize the performance and the parameters used in our simulations in Table 1. The stiffness parameter k_m is the representative stiffness for k_v and k_e in Equation (2) and Equation (3). We define the values $k_v = 0.4k_m$ and $k_e = 0.6k_m$ for all our examples. In addition, we define the damping matrix $K_{\dot{x}} = 0.1k_m\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. For the locomotion examples, we define $K_\tau = 10^3$ (Section 5.3). The bottleneck in the computation is the number of contacts solved by LCP for every time step. We use the publicly available PATH LCP solver (http://www.cs.wisc.edu/cpnet/cpnetsoftware/) to solve for Equation (16) for both the cases of rigid and soft contacts.

7 Evaluation and Limitations

In addition to the comparisons described in Section 6, we performed quantitative analyses to show the impact of soft contact, the effect of a few key parameters, and comparisons with alternative design decisions. We also address a few limitations of the current system in this section.

Center of pressure. We plotted the center of pressure of the character's foot from one of the biped examples (Figure 5). The comparison shows that a soft contact has a much smoother center of pressure than the rigid contact. Because the center of pressure is crucial to maintaining the angular momentum, this result is consistent with our observation that the rigid character loses balance more easily when large external torque is applied. Similarly, we compared the magnitude of the total contact force from the same example (Figure 6). The average contact force is usually similar between a rigid and a soft contact, but the rigid contact has a larger variance in the magnitude of contact force. Note that more sophisticated biped controllers might achieve similar results, but we deliberately choose very simple control algorithms to highlight the effect of soft body contact.

Contribution to robustness. The main reason that soft bodies can generate contact forces resulting in a more robust motion is due to the increase in the number of contact points. In all our examples, the number of contacts of soft bodies is significantly greater than that of rigid bodies. For example, we compared the number of contact points of rigid and soft fingers in a pinch-grasp motion (Figure 4). Figure 7 suggests that soft fingers maintain a large number of contacts at all times and do not abruptly change the number of contacts under external forces. In contrast, the number of contacts on rigid fingers is much smaller and fluctuates drastically when an external force is applied, leading to unpredictable and less stable results. We also verified that the robustness of the motion was not due to the implicit formulation. In this experiment, we replaced all the implicit formulations with explicit ones and simulated the same biped examples with a smaller time step. Both implicit and explicit characters recover from the push in a similar manner.

System parameters. Our framework performs robustly for a wide range of parameters such as the stiffness of the mesh and time steps. We use a time step of 4ms for locomotion controllers (see Table 1). We also performed experiments with an larger time step of 8ms. The resulting controller was still robust to large forces as compared to the original SIMBICON controller. In addition, we experimented with stiffness of the foot meshes increased by a factor of 10-20 and the controller performed robustly.

Different LCP solutions. In theory, the mixed LCP problem in Equation (16) can have multiple solutions depending on the initial point given to the PATH solver. In practice, we empirically showed

Examples	total DOFs	simulated DOFs	num contacts	fps	LCP time (%)	Time step (ms)	Stiffness k_m
finger flick	2573	576±88	39±6	3.9±3.2	86±6	1.7	1.5×10^4
arm fold	2802	322±89	33±10	3.5±1.7	68±10	8.3	10^4
pinch-grasp	1427	258±22	29±4	5.2±3.2	85±6	1.7	1.5×10^4
biped walk	334	197±43	16±3	18.5±4.5	63±5	4.0	10^3

Table 1: *Performance and parameters of the examples. "total DOFs" is the number of DOFs that can be potentially simulated while "simulated DOFs" is the number of DOFs in adaptive simulation. "fps" is the frame rate for our simulations and "LCP time" is the percentage of the simulation time to solve Equation (16). For biped walk, the LCP is solved every 8 SIMBICON time steps (SIMBICON time step is 0.5 ms).*

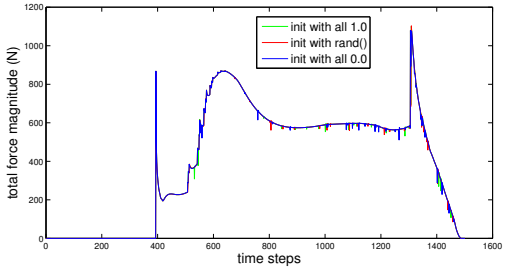

Figure 8: *Comparison of the magnitude of contact forces for simulations with different initial points for the LCP solver.*

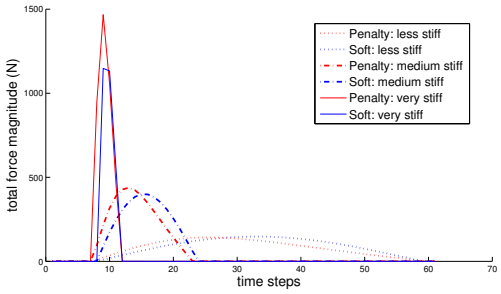

Figure 9: *Comparison of the contact force magnitudes from penalty method and soft contacts with LCP.*

that our particular LCP, i.e. the contact force problem, is not sensitive to the initial point. We repeatedly simulated the same motion in Figure 3 by solving the LCP with different initial points: a vector with all zeros, a vector with all ones, and a vector of different random values at every frame. Figure 8 shows little variation in the force magnitudes and the simulated motions are close to identical.

Flexible rigid foot vs. deformable foot. One possible approach to generate robust locomotion is to represent the foot with many rigid links, allowing for more flexible foot motion. Coupling the penalty methods for contact with multiple smaller links may add more compliance at the sites of contact. We therefore increased the complexity of the rigid foot to four links, each of which is connected to its parent link by a hinge joint along the center line of the foot. The simulation results shown in the accompanying video show that a rigid foot represented by four links results in more stable motion when comparing with a rigid foot with fewer links. However, our deformable feet still exhibit more stability than the four-linked rigid feet due to more continuous changes of contacts. There are a couple of disadvantages associated with breaking down a rigid foot into smaller rigid links. As the number of links increase, the mass and the size of each link decreases thereby potentially causing instability due to increased ratio of the maximum to minimum mass in the skeleton. This results in more strict time step restrictions and more careful selection of stiffness parameters. In addition, modeling the feet with multiple links increases the complexity of controller design, as more parameters need to be tuned for the additional actuators on the foot.

Penalty methods. Penalty-based methods are commonly used to approximate the compliance at the site of contact due to their simple formulation and less intensive computation. However, our approach is advantageous due to the following reasons. First, LCP formulation models more accurate contact by enforcing work-less normal force, no penetration, and realistic slipping. Second, our method explicitly deforms the geometry of the body parts, introducing more contact points which, in turn, increase the robustness of the control algorithms. Third, using low stiffness for penalty methods may lead to frequent penetration artifacts. For a detailed, contact-rich motion, such as hand manipulation, the artifacts can be very visible. We compared the penalty method with our soft contacts for a simple case of a box falling on the ground (see video and Figure 9). We modeled the penalty forces in the normal direction as described in [Yamane and Nakamura 2006] and vary the stiffness and damping parameters to model different levels of softness and the duration of contact. For the deformable box with soft contacts, we chose the mesh stiffness such that the duration of contact is similar to the corresponding penalty method case. When the stiffness of the box is high, the large contact force computed by the penalty method causes the hard box to bounce elastically because the velocity cannot be damped out quickly enough given a very short contact duration. In contrast, our method correctly generates just enough force to stop the hard box and produces no extra bounce. When the stiffness of the box is low, the contact force profiles look similar for both cases in Figure 9. However, the motion generated by the penalty method shows large penetration between the box and the ground while our approach does not. In addition, visible deformation that stores and releases potential energy can be observed using our approach.

Limitations. There are a few limitations in the current implementation of the algorithm. Our implementation does not handle a deformable body across multiple rigid bones. For motions with large areas of contact, such as rolling on the floor or sitting on a chair, it is essential to allow the contact to move continuously across the bones. With moderate effort to convert between frames of reference, we can modify our implementation such that two connected point masses can be parented to different rigid bones.

Our algorithm is not optimized for the performance. The computation of a few components, such as collision detection and handling can be largely reduced by decoupling the surface mesh resolution and the number of contacts such as using volume contacts as described in a recent work [Allard et al. 2010]. In addition we can incorporate existing, efficient methods that simulate a fine surface mesh embedded in a coarse control mesh to improve the perfor-

mance [Kim and Pollard 2011b].

The stiffness coefficients in our contact model are arbitrarily chosen and do not resemble human skin and muscle materials in the real world. To achieve more realistic deformation, we need to acquire accurate material coefficients and design more anatomically correct models. Better biomechanical modeling of the musculoskeletal system is our ultimate aim for simulating human behaviors. However, balancing between model complexity and computational tractability remains a challenging problem that requires further exploration.

We have noticed some self-intersection between legs in the biped examples. Our algorithm can handle collision between two body parts at the deformable layer, but when the intersection is deep in the bone level, our algorithm does not handle the collision. This type of deep self-intersection is usually due to the control force generated by a controller that does not take into account self collision. Having a better collision detection routine can improve the situation (we use ODE's build-in collision detection routine), but a more effective solution is probably to design a controller that considers "proprioception" when computing the control forces.

8 Conclusion

In this paper, we investigate the impact of the deformable bodies on existing control algorithms for physically simulated characters. The primary contribution of this paper is to demonstrate that simple control strategies coupled with the simulation of soft tissue deformation at the site of contact can achieve very robust and realistic motion. In constructing, we develop a practical system that simulates two-way coupling of rigid and deformable bodies efficiently and robustly. We then apply a few simple and widely used control algorithms, such as pose-space tracking control, Cartesian-space tracking control, and SIMBICON, to simulate a variety of behaviors for both full-body locomotion and hand manipulation. The results were compared with motions of a character comprising only of rigid bodies.

A controller can always be improved by designing better control strategies as evidenced by recent research in biped control. However, in this paper, we depart from the approach of developing a new control algorithm and demonstrate that superior robustness can be achieved for existing simple controllers without necessarily adding more complexity. Our soft contacts formulation can be used in conjunction with any existing controllers for a variety of tasks such as biped locomotion or hand manipulation. We lay stress on the simplicity of the generic soft contacts formulation that does not demand for reimplementation of most of the existing controllers and may facilitate development of advanced control methods that use the deformation in the control design.

There are a few interesting avenues for future research. We would also like to investigate other biped controllers that do not use PD feedback control [da Silva et al. 2008; Muico et al. 2009; Ye and Liu 2010]. These controllers are usually more "aware" of contact situations when planning the control forces. The simulation of deformable bodies at the site of contact could have greater impact on those controllers.

Recent advent in biped controllers primarily focused on robust locomotion. Motion with large impact due to collisions has not been demonstrated on physically simulated character. One exciting future direction is to design new controllers that exploit soft contact to achieve motion with frequent high-speed collisions, such as fighting, parkour, or American football.

A related topic that remains relatively un-explored is the control in the presence of collisions between human characters. Successful rigid body collision algorithms do not lend themselves well to collisions for human motion because human is made of neither passive nor rigid bodies. We believe that the problem of collision with human body is unique and needs to be considered together with the problem of motion control. Our work takes a step toward designing a better collision algorithm specifically for modeling human motion.

Acknowledgements

This work was supported by NSF CAREER award CCF 0742303 and Alfred P. Sloan fellowship.

References

ABE, Y., DA SILVA, M., AND POPOVIĆ, J. 2007. Multiobjective control with frictional contacts. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 249–258.

ALLARD, J., FAURE, F., COURTECUISSE, H., FALIPOU, F., DURIEZ, C., AND KRY, P. G. 2010. Volume contact constraints at arbitrary resolution. *ACM Trans. Graph. 29*.

ALLEN, B., CURLESS, B., AND POPOVIĆ, Z. 2002. Articulated body deformation from range scan data. *ACM Transactions on Graphics 21*, 3 (July), 612–619.

ANITESCU, M., AND POTRA, F. A. 1997. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics 14*, 231–247.

CAPELL, S., GREEN, S., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2002. Interactive skeleton-driven dynamic deformations. *ACM Transactions on Graphics 21*, 3 (July), 586–593.

CHARDONNET, J.-R., DAVID, A., KHEDDAR, A., AND YOKOI, K. 2008. Interactive dynamic simulator for humanoid robotswith deformable soles. In *26th Annual Conference of the Robotics Society of Japan (RSJ)*.

COROS, S., BEAUDOIN, P., YIN, K., AND VAN DE PANNE, M. 2008. Synthesis of constrained walking skills. *ACM Trans. Graph 27*, 5, 113.

COROS, S., BEAUDOIN, P., AND VAN DE PANNE, M. 2010. Generalized biped walking control. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 1–9.

CUTKOSKY, M., AND KAO, I. 1989. Computing and controlling the compliance of a robotic hand. *IEEE Transactions on Robotics and Automation 5*, 2.

DA SILVA, M., ABE, Y., AND POPOVIĆ, J. 2008. Interactive simulation of stylized human locomotion. *ACM Trans. Graph. (SIGGRAPH) 27*, 3, 1–10.

GALOPPO, N., OTADUY, M. A., TEKIN, S., GROSS, M. H., AND LIN, M. C. 2007. Soft articulated characters with fast contact handling. *Comput. Graph. Forum 26*, 3, 243–253.

GOURRET, J.-P., THALMANN, N. M., AND THALMANN, D. 1989. Simulation of object and human skin deformations in a grasping task. In *ACM SIGGRAPH*, 21–30.

KIM, J., AND POLLARD, N. S. 2011. Direct control of simulated non-human characters. *IEEE Computer Graphics and Applications (In Press)*.

KIM, J., AND POLLARD, N. S. 2011. Fast simulation of skeleton-driven deformable body characters. *ACM Transactions on Graphics (In Press)*.

KRY, P. G., AND PAI, D. K. 2006. Interaction capture and synthesis. *ACM Trans. on Graphics 25*, 3 (Aug.), 872–880.

KRY, P., JAMES, D. L., AND PAI, D. K. 2002. Eigenskin: Real time large deformation character skinning in hardware. In *Eurographics/SIGGRAPH Symposium on Computer Animation*.

LEE, Y., KIM, S., AND LEE, J. 2010. Data-driven biped control. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 1–8.

LEWIS, J. P., CORDNER, M., AND N., F. 2000. Pose space deformations: A unified approach to shape interpolation and skeleton-driven deformation. *ACM Trans. on Graphics (SIGGRAPH) 19*, 3 (July).

LIU, C. K. 2009. Dextrous manipulation from a grasping pose. *ACM Transactions on Graphics (SIGGRAPH) 28*, 3.

MAGNENAT-THALMANN, N., LAPERRIÈRE, R., AND THALMANN, D. 1988. Joint-dependent local deformations for hand animation and object grasping. In *Graphics Interface*, 26–33.

MAYA, A. *Autodesk Maya, http://usa.autodesk.com/*.

MOHR, A., AND GLEICHER, M. 2003. Building efficient, accurate character skins from examples. *ACM Trans. on Graphics (SIGGRAPH) 22*, 3 (July).

MORDATCH, I., DE LASA, M., AND HERTZMANN, A. 2010. Robust physics-based locomotion using low-dimensional planning. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 1–8.

MUICO, U., LEE, Y., POPOVIĆ, J., AND POPOVIĆ, Z. 2009. Contact-aware nonlinear control of dynamic characters. In *ACM Trans. Graph. (SIGGRAPH)*, 1–9.

PARK, S. I., AND HODGINS, J. K. 2006. Capturing and animating skin deformation in human motion. *ACM Transactions on Graphics 25*, 3 (July), 881–889.

PAULY, M., PAI, D. K., AND GUIBAS, L. J. 2004. Quasi-rigid objects in contact. In *ACM SIGGRAPH/Eurographics Symposium on Computer animation*, 109–119.

SIMBICON, 2007. `www.cs.ubc.ca/~van/simbicon_cef`.

STEWART, D. E., AND TRINKLE, J. C. 1996. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering 39*, 15, 2673–2691.

TURNER, R., AND THALMANN, D. 1993. The elastic surface layer model for animated character construction. In *Proceedings of Computer Graphics International*.

WANG, X. C., AND PHILLIPS, C. 2002. Multi-weight enveloping: Least-squares approximation techniques for skin animation. In *Eurographics/SIGGRAPH Symposium on Computer Animation*.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2009. Optimizing walking controllers. *ACM Trans. Graph. (SIGGRAPH Asia) 28*, 5, 1–8.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2010. Optimizing walking controllers for uncertain inputs and environments. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 1–8.

XYDAS, N., AND KAO, I. 1998. Modeling of contact mechanics with experimental results for soft fingers. In *IEEE/RSJ International Conference on Intelligent Robots and System*.

YAMAGUCHI, J., TAKANISHI, A., AND KATO, I. 1995. Experimental development of a foot mechanism with shock absorbing material for acquisition of landing surface position information and stabilization of dynamic biped walking. In *IEEE International Conference on Robotics and Automation*.

YAMANE, K., AND NAKAMURA, Y. 2006. Stable penalty-based model of frictional contacts. In *ICRA*, 1904–1909.

YE, Y., AND LIU, C. K. 2010. Optimal feedback control for character animation using an abstract model. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 1–9.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. Simbicon: simple biped locomotion control. *ACM Trans. Graph. (SIGGRAPH) 26*, 3, 105.