Efficient Nonlinear Skin Simulation for Multi-Finger Tactile Rendering

Alvaro G. Perez[1], Gabriel Cirio[1], Daniel Lobo[1], Francesco Chinello[2,3], Domenico Prattichizzo[2,3] and Miguel A. Otaduy[1]

***Abstract*— Recent advances in tactile rendering span, among others, wearable cutaneous interfaces, tactile rendering algorithms, or nonlinear soft skin models. However, the adoption of these advances for multi-finger tactile rendering of dexterous grasping and manipulation is hampered by the computational cost incurred with nonlinear skin models when applied to the full hand. We have observed that classic constrained dynamics solvers, typically designed for contact mechanics, fail to perform efficiently on deformation constraints of nonlinear skin models. In this paper, we propose a novel constrained dynamics solver designed to perform well with highly nonlinear deformation constraints. In practice, we achieve more than $10\times$ speed-up over previous approaches, and as a result we enable multi-finger tactile rendering of manipulation actions that capture the nonlinearity of skin.**

I. Introduction

Humans exploit skin compliance during grasping to smoothly adapt to the surfaces of grasped objects and to expose larger frictional contacts that improve grasp stability [8]. Accurate tactile rendering of the interaction with virtual environments should then rely on soft skin models, but tactile rendering of soft grasping comes with many difficulties: the simulation of a realistic hand should account for skeletal constraints and a soft flesh; skin is highly nonlinear, very soft under low forces, but highly constrained under high forces; and the tactile rendering algorithm itself should account for the distributed nature of contact under soft skin.

To address most of these difficulties, in our work in recent years we have designed novel components of a tactile rendering framework that incorporate soft skin and its characteristic behavior. We have designed a simulation algorithm that accounts for skeletal constraints and a soft flesh with two-way coupling, such that forces are transmitted to and from skeleton and flesh [5]. We have demonstrated efficient simulation of the extreme nonlinearity of skin using a biphasic model with strain-limiting constraints [14]. And we have proposed a tactile rendering algorithm that optimizes the configuration of a wearable cutaneous interface under the objective of contact surface matching [15]. The last two works have been demonstrated interactively only on a single finger.

But, before soft skin models can be adopted for accurate visuo-haptic rendering, one major problem is left to be solved. Simulation of nonlinear skin deformation is a computationally complex problem, and its application to full-hand modeling for multi-finger tactile rendering is currently hampered by the computational cost of constrained dynamics solvers. In this work, we present a fast solver for constrained dynamics that enables multi-finger tactile rendering of the interaction between a nonlinear soft hand model and a virtual environment. As a result, we achieve interactive visual simulation of dexterous manipulations with accurate skin behavior, together with cutaneous rendering of the complex forces involved during soft grasping, as shown in Fig. 1.

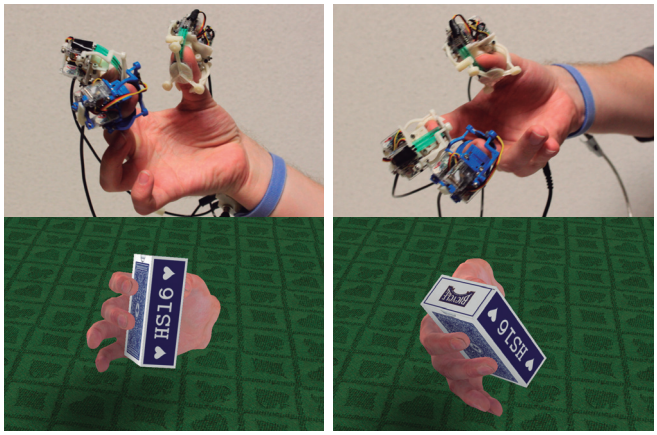

Fig. 1. Tactile rendering of dexterous manipulation of a virtual object. While the user rotates the object, subtle changes in contact orientation and pressure are reproduced by the cutaneous interfaces.

Constrained dynamics is not a novel problem to computational haptics, as it is a corner stone for accurate and robust simulation of frictional contact [4], [11], [16], [19], [12]. However, we have observed that typical solvers for contact constraints do not work well with the deformation constraints of nonlinear skin. We propose a novel solver for constrained dynamics that addresses the highly nonlinear nature of deformation constraints. It is based on a Jacobi relaxation scheme to minimize the computational cost per iteration, linearizes the constraints after each Jacobi iteration to optimize the true nonlinear problem, and performs a line-search optimization per iteration to guarantee a steady error reduction. After the formulation of the simulation problem and a discussion of limitations of classic constrained dynamics solvers in Section II, we describe the details of our solver in Section III.

On highly constrained scenarios, which are those critical for maintaining an interactive simulation at all times, we achieve speed-ups of more than $10\times$ over previous

[1] URJC Madrid, c/ Tulipán s/n, 28933 Móstoles (Madrid), Spain.
[2] Department of Information Engineering and Mathematics, University of Siena, Via Roma 56, 53100 Siena, Italy.
[3] Department of Advanced Robotics, Istituto Italiano di Tecnologia, Via Morego 30, 16163 Genova, Italy.

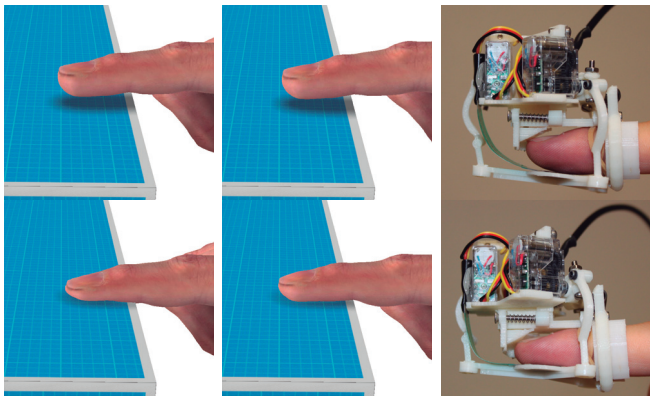

Fig. 2. Impact of the nonlinear skin model under contact. Left: Without strain-limiting constraints, the finger collapses when pressing hard. Middle: With strain-limiting constraints, the finger retains a correct shape. Right: Tactile rendering of the same contact with a wearable cutaneous device.

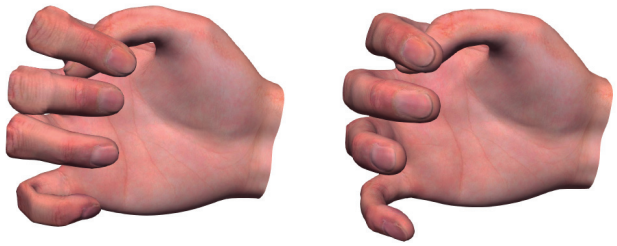

Fig. 3. Impact of the nonlinear skin model on hand deformations. Left: Without strain-limiting constraints, finger joints deform excessively under large rotations. Right: With strain-limiting constraints, joints are compliant under small rotations, yet they retain a correct shape under large rotations.

constrained dynamics solvers. The interactive solution of nonlinear skin deformation is employed to drive a tactile rendering algorithm, described in Section IV. In Section V we discuss performance results of our solver, as well as interactive grasping and manipulation actions rendered using three wearable cutaneous interfaces.

II. Nonlinear Skin as Constrained Dynamics

In this section, we first describe the simulation of a deformable hand with nonlinear skin, which serves as the central component of a model-based tactile rendering algorithm. We describe all the components of the simulation, focusing on the formulation of the hand's deformation as a constrained dynamics problem. Then, we discuss typical linear complementarity problem (LCP) solutions for constrained dynamics, based on relaxation methods.

A. Hand Simulation

The deformation of a hand is dominated by a combination of skeletal constraints and soft skin deformation. Taking the tracked configuration of the user's hand as input, we simulate skeletal motion using an algorithm for articulated-body dynamics with linear cost in the presence of joint stiffness and implicit integration [6]. The bones of the hand's skeleton are linked to their corresponding tracked configuration using 6D virtual coupling springs [9]. Due to the lack of a kinesthetic haptic interface, the motion of the user's hand cannot be stopped by virtual objects. Then, the deviation between the real and virtual hand configurations is absorbed by the 6D virtual coupling springs.

Skin is highly nonlinear, and the impact of this nonlinearity in visuo-haptic simulation is shown in two examples in Fig. 2 and Fig. 3. One possible solution would be to use efficient methods for hyperelastic materials with contact [3]. Instead, we simulate nonlinear skin deformation using a linear corotational FEM formulation augmented with strain-limiting constraints. We refer to [14] for details on the formulation of constraints and their Jacobians. This approach approximates the extreme nonlinearity of skin using a biphasic model, very soft under low forces, and completely constrained under large forces. On each simulation frame, we integrate dynamics using implicit Euler with force linearization, and in this way the solution to unconstrained dynamics turns into the solution to a sparse linear problem. However, we enforce nonlinear strain-limiting constraints, and then the solution to deformation dynamics turns into a nonlinear constrained optimization problem. This optimization problem consists of finding velocities that minimize the deviation from unconstrained velocities, subjct to the strain-limiting constraints. Specifically, this nonlinear constrained optimization problem can be cast as:

$$\min \frac{1}{2}(\mathbf{v}-\mathbf{A}^{-1}\mathbf{b})^T\mathbf{A}(\mathbf{v}-\mathbf{A}^{-1}\mathbf{b}), \tag{1}$$

$$\text{s.t. } \mathbf{C}(\mathbf{x}_0+h\mathbf{v}) \geq \mathbf{0},$$

$$\text{with } \mathbf{A}=\mathbf{M}-h\frac{\partial \mathbf{F}}{\partial \mathbf{v}}-h^2\frac{\partial \mathbf{F}}{\partial \mathbf{x}} \tag{2}$$

$$\text{and } \mathbf{b}=\left(\mathbf{M}-h\frac{\partial \mathbf{F}}{\partial \mathbf{v}}\right)\mathbf{v}_0+h\mathbf{F}(\mathbf{x}_0,\mathbf{v}_0). \tag{3}$$

Where $\mathbf{x}$ and $\mathbf{v}$ are vectors containing all nodal positions and velocities, respectively, $\mathbf{x}_0$ and $\mathbf{v}_0$ denote these positions and velocities at the end of the previous time step, $\mathbf{F}$ is a vector with all nodal forces, $\mathbf{M}$ is the mass matrix, $\mathbf{A}$ and $\mathbf{b}$ are the matrix and right-hand side of the unconstrained linear system, $\mathbf{C}$ is the vector of strain-limiting constraints, and h is the time step.

We implement two-way coupling between the skeleton and the skin using zero-rest-length springs, one per skin node, with the skeletal end of each spring defined using linear blend skinning. We follow the numerical integration algorithm of Garre et al. [5] to avoid solving a coupled system under implicit integration. This algorithm first solves the skeletal motion approximating the effect of the skin, and then solves the skin deformation.

To conclude the description of the simulation model, we have implemented collision response using the penalty method, with anchored springs for Coulomb friction [20].

B. LCP Formulations and Relaxation Methods

Nonlinear optimization problems such as the one in (1) are common in constraint-based collision response, and are

solved using sequential quadratic programming (SQP) strategies [4], [7], [13]. Velocities are initialized as unconstrained velocities $\mathbf{v}_0 \leftarrow \mathbf{A}^{-1}\,\mathbf{b}$, and positions as unconstrained positions $\mathbf{x}_0 \leftarrow \mathbf{x}_0 + h\,\mathbf{v}_0$. Then, the constraints are linearized, e.g., at the unconstrained positions, as $\mathbf{C}(\mathbf{x}) \approx \mathbf{C}(\mathbf{x}_0) + \frac{\partial \mathbf{C}}{\partial \mathbf{x}}\,(\mathbf{x} - \mathbf{x}_0)$, and the problem in (1) becomes an LCP. Note that, for computational efficiency reasons, this LCP is formed using only the constraints that are violated at the unconstrained positions. The LCP is solved until convergence (of the linearized constraints), and constraints are again linearized for a new SQP iteration. The process continues until convergence of the nonlinear constraints.

Given positions $\mathbf{x}_{i-1}$ and velocities $\mathbf{v}_{i-1}$ at the previous iteration, each SQP iteration can be formulated as:

$$\begin{aligned} &\min \frac{1}{2}\,(\mathbf{v}_i - \mathbf{v}_0)^T \mathbf{A} (\mathbf{v}_i - \mathbf{v}_0), \\ &\text{s.t. } \mathbf{C}(\mathbf{x}_{i-1}) + h\,\mathbf{J}\,(\mathbf{v}_i - \mathbf{v}_{i-1}) \geq \mathbf{0}. \end{aligned} \tag{4}$$

In practice, we approximate this SQP formulation, and on each iteration we minimize the change in velocities:

$$\begin{aligned} &\min \frac{1}{2}\,(\mathbf{v}_i - \mathbf{v}_{i-1})^T \mathbf{A} (\mathbf{v}_i - \mathbf{v}_{i-1}), \\ &\text{s.t. } \mathbf{C}(\mathbf{x}_{i-1}) + h\,\mathbf{J}\,(\mathbf{v}_i - \mathbf{v}_{i-1}) \geq \mathbf{0}. \end{aligned} \tag{5}$$

This yields the following LCP per SQP iteration:

$$\mathbf{0} \leq \lambda_i \quad \perp \quad \mathbf{J}\,\mathbf{A}^{-1}\,\mathbf{J}^T\,\lambda_i + \frac{1}{h}\,\mathbf{C}(\mathbf{x}_{i-1}) \geq 0. \tag{6}$$

Such LCPs are typically solved using relaxation methods based on matrix splitting [2], e.g., projected Gauss-Seidel (PGS) or projected Jacobi (PJ), on the LCP matrix $\mathbf{B} = \mathbf{J}\,\mathbf{A}^{-1}\,\mathbf{J}^T$. Solving the LCP yields the Lagrange multipliers λ_i, and next the velocities and positions are updated for the next SQP iteration:

$$\mathbf{v}_i = \mathbf{v}_{i-1} + \mathbf{A}^{-1}\,\mathbf{J}^T\,\lambda_i, \tag{7}$$

$$\mathbf{x}_i = \mathbf{x}_{i-1} + h\,(\mathbf{v}_i - \mathbf{v}_{i-1}). \tag{8}$$

On deformation problems with implicit integration, such as ours, the solution to the LCP (6) is computationally expensive. With PGS, the cost is dominated by the dense nature of the matrix $\mathbf{B}$, both on its initialization and during each LCP iteration. PJ requires only the diagonal $\mathbf{D}$ of $\mathbf{B}$, but suffers worse convergence. Convergence can be guaranteed dividing the Jacobi update by a weight equal to the degree of the incidence graph of the $\mathbf{B}$ matrix [1], [18]. With a dense $\mathbf{B}$ matrix, this weight is just the number of constraints. In our experience, this weighting scheme is overly conservative, and a line search works better in practice. More elaborate methods also exist, such as iterative constraint anticipation [13], which requires more iterations but enjoys a much lower cost per iteration, and applies a relaxation method jointly to the system matrix $\mathbf{A}$ and the LCP matrix $\mathbf{B}$.

Despite the high cost of each LCP solve, the SQP approach is appropriate for contact mechanics problems because it converges in few iterations, i.e., few LCP solves (less than 4 in the vast majority of cases as reported in [13]). For strain-limiting constraints, on the other hand, we have observed that the nonlinear constraints improve very slowly on each LCP solve. Fig. 5 shows the prototypical convergence behavior of PGS and PJ on the nonlinear constrained optimization problem (1) with strain-limiting constraints. The residual (of the linearized constraints) is quickly reduced within each LCP, but the evaluation of the full nonlinear residual at the beginning of each SQP iteration denotes very slow overall convergence. In practice, a large effort is spent in vain solving expensive LCPs. In the next section, we introduce our new solver to overcome this slow convergence and enable interactive hand simulation with nonlinear skin deformations.

III. Nonlinear Jacobi Solver

In this section, we present our new solver for nonlinear constrained dynamics, particularly designed for the simulation of nonlinear skin using strain-limiting constraints. Our solver is based on two principles: the linearization of the nonlinear constraints on each iteration, and the minimization of the computational cost per iteration. By following these two principles, we achieve a speed-up of more than $10\times$ w.r.t. standard constrained dynamics solvers, on deformable hand simulation in the context of multi-finger tactile rendering.

We start the section with a detailed description of the PJ line-search solver designed for each SQP iteration. Then, we discuss computational optimizations that exploit Cholesky factorizations and parallelism.

A. Projected Jacobi Line-Search

As discussed in the previous section, we linearize the nonlinear constrained dynamics problem by formulating SQP iterations in the form of (5), and each SQP iteration yields the LCP (6). Our nonlinear solver formulates one single PJ iteration on this LCP, which leads to the following Jacobi update for the Lagrange multipliers:

$$\mathbf{D}\,(\lambda_i - \lambda_{i,0}) + \mathbf{B}\,\lambda_{i,0} + \frac{1}{h}\,\mathbf{C}(\mathbf{x}_{i-1}) \geq 0. \tag{9}$$

With Lagrange multipliers initialized as $\lambda_{i,0} = 0$, the Jacobi update simplifies to:

$$\mathbf{D}\,\lambda_i + \frac{1}{h}\,\mathbf{C}(\mathbf{x}_{i-1}) \geq 0. \tag{10}$$

Jacobi relaxation is not guaranteed to converge, and instead we execute a line search [10] along the direction of the Jacobi update. With $\lambda_{i,0} = 0$, this search direction is simply:

$$\mathbf{s} = \max\left(0, \quad -\frac{1}{h}\,\mathbf{D}^{-1}\,\mathbf{C}(\mathbf{x}_{i-1})\right). \tag{11}$$

Lagrange multipliers are reparameterized based on a step along the search direction, $\lambda_i = \alpha\,\mathbf{s}$, with $0 \leq \alpha \leq 1$ to avoid reprojection. The step α is computed by minimizing the QP equivalent to the LCP in (6) (see [2]):

$$\begin{aligned} &\min \frac{1}{2}\,\lambda_i^T\,\mathbf{B}\,\lambda_i + \frac{1}{h}\,\lambda_i^T\,\mathbf{C}(\mathbf{x}_{i-1}) = \\ &\min \frac{1}{2}\,\mathbf{s}_i^T\,\mathbf{J}\,\mathbf{A}^{-1}\,\mathbf{J}^T\,\mathbf{s}_i\,\alpha^2 + \frac{1}{h}\,\mathbf{s}_i^T\,\mathbf{C}(\mathbf{x}_{i-1})\,\alpha \Rightarrow \\ &\qquad \alpha = \frac{-\frac{1}{h}\,\mathbf{s}_i^T\,\mathbf{C}(\mathbf{x}_{i-1})}{\mathbf{s}_i^T\,\mathbf{J}\,\mathbf{A}^{-1}\,\mathbf{J}^T\,\mathbf{s}_i}. \end{aligned} \tag{12}$$

Followed by a projection of α to the interval $[0,1]$. Note that, even though α is unique for all constraints, their forces are not equal, as the search direction scales differently the various constraints.

Once Lagrange multipliers are computed, we update velocities and positions as in (7)-(8), linearize the constraints again, and start a new SQP iteration. To test convergence, we use as residual function the norm of the constraints $\|\min(\mathbf{C}(\mathbf{x}),0)\|$.

In contrast to classic solvers, we avoid wasted effort by not solving to convergence the LCP in each SQP iteration, we minimize computational cost per iteration thanks to the Jacobi relaxation scheme, and we guarantee a steady error reduction thanks to the line-search optimization. These features are the key ingredients that result in a speed-up of more than $10\times$ in practice. Next, we discuss further optimizations to the solver.

B. Cholesky Factorization

With just one relaxation step per SQP iteration, the cost of the solver is largely dominated by the computation of the LCP matrix. Then, PJ is far more efficient than PGS, as it requires only the diagonal $\mathbf{D}$, not the full matrix $\mathbf{B}$. Moreover, we further optimize the efficiency of the solver by precomputing a Cholesky factorization of the system matrix, $\mathbf{A} = \mathbf{L}\mathbf{L}^T$

The diagonal matrix $\mathbf{D}$ is formed using the Jacobians of individual constraints, $\mathbf{J}_k$, as:

$$\mathbf{D} = \mathrm{diag}(\mathbf{J}_k \mathbf{L}^{-T} \mathbf{L}^{-1} \mathbf{J}_k^T). \tag{13}$$

Each term of the diagonal can be computed by executing just one back-substitution $\mathbf{L}^{-1}\mathbf{J}_k^T$, and then a dot product of this vector with itself.

Algorithm 1 Constrained dynamics step

procedure CONSTRAINEDDYNAMICS($\mathbf{x}_0, \mathbf{v}_0$)
  Formulate $\mathbf{A}$ and $\mathbf{b}$ using (2)-(3)
  Compute Cholesky factorization $\mathbf{A} = \mathbf{L}\mathbf{L}^T$
  Initialize unconstrained velocities $\mathbf{v}_0 \leftarrow \mathbf{A}^{-1}\mathbf{b}$
  Initialize unconstrained positions $\mathbf{x}_0 \leftarrow \mathbf{x}_0 + h\mathbf{v}_0$
  $i = 1$
  [–P–] Evaluate constraints $\mathbf{C}(\mathbf{x}_{i-1})$
  while $\|\min(\mathbf{C}(\mathbf{x}_{i-1}),0)\| >$ tolerance **do**
    Evaluate constraint Jacobians $\mathbf{J} = \frac{\partial \mathbf{C}}{\partial \mathbf{x}}$
    [–P–] Compute LCP diagonal $\mathbf{D}$ as in (13)
    Compute search direction $\mathbf{s}$ as in (11)
    Compute step α as in (12)
    $\alpha \leftarrow \max(\min(\alpha,1),0)$
    Compute $\lambda_i = \alpha\mathbf{s}$
    Update velocities and positions using (7)-(8)
    $i \leftarrow i+1$
    [–P–] Evaluate constraints $\mathbf{C}(\mathbf{x}_{i-1})$
  end while
end procedure

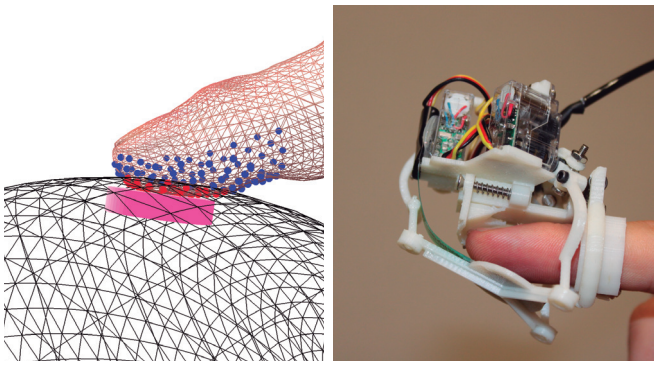

Fig. 4. Tactile rendering results using *contact surface matching* [15]. Left: Test points on the virtual finger pad, in contact (red) and not in contact (blue), with a visualization of the device configuration resulting from the optimization (in pink). Right: The actual cutaneous device, rendering the optimal contact location and orientation on the finger pad.

The Cholesky factorization is of course reused also in the computation of the line-search step in (12) and the velocity update in (7).

C. Parallelization

Even after exploiting the Cholesky factorization, the formulation of matrix $\mathbf{D}$ is still one of the major bottlenecks in the solver, although now on par with the evaluation of constraints, which requires the computation of SVDs [14]. Fortunately, these two steps in our algorithm can be easily parallelized. We use a CPU multi-core parallelization.

Algorithm 1 shows pseudocode for one step of nonlinear constrained dynamics simulation using our nonlinear Jacobi solver. Bottleneck steps that are parallelized are highlighted with the symbol [–P–]. Several other steps may also be parallelized, but they barely affect performance.

IV. TACTILE RENDERING

We render tactile stimuli by modulating the local contact surface (position and orientation) on each finger pad, using a wearable cutaneous interface [17]. This device touches the finger pad with a disk-like surface, has three degrees of freedom (pitch, roll, and displacement normal to the finger pad), and is actuated using a parallel mechanism. In our examples, we have used three interfaces, worn on the thumb, index, and middle fingers.

On each simulation frame, we track the user's hand using a Leap Motion device, simulate hand dynamics in contact with virtual objects, and finally compute optimal configurations of the cutaneous interfaces to mimic the contact configuration in the virtual environment. For this purpose, we adopt the *contact surface matching* tactile rendering algorithm of Perez et al. [15]. For each finger, we test for collision with the virtual environment roughly 100 points sampled on the finger pad area, perform the contact surface matching optimization to compute the degrees of freedom of the corresponding cutaneous interface, and send them as a rendering command to the driver of the interface. Fig. 4 shows an example frame depicting the points sampled on the finger pad, the device

configuration resulting from the optimization, and the actual device acting on the user's finger.

Tactile rendering runs in just over 1 ms per finger on our system, and we have executed the optimizations for all three fingers in parallel. The cutaneous rendering loop is intrinsically passive, hence no stability considerations need to be accounted for.

V. Results

We have executed all our experiments on a PC with a quad-core Intel Core-i7 2600 (3.4GHz) and 8GB of RAM. Multi-core parallelization as discussed in Section III-C is implemented using OpenMP, and we rely on the Eigen library for efficient linear algebra operations.

In our tactile rendering framework, we simulate the deformable hand using an articulated skeleton with 16 bones and a deformable mesh with 597 tetrahedra. This coarse model guarantees interactive computation of robust and smooth deformations even when large parts of the skin reach strain-limiting constraints. The deformation is then smoothly interpolated to a textured triangle mesh with 23640 triangles, which is used for visual rendering. For tactile rendering, we also detect contacts on sample vertices of the finger pads of this high-resolution mesh, as discussed in Section IV.

Fig. 2 and Fig. 3 show two examples where the impact of nonlinear skin deformations becomes evident. Without strain-limiting constraints, fingers deform excessively under pressing contact. With constraints, instead, they appear soft at initial contact, allowing a compliant adaptation of the finger pads to grasped objects, but they correctly retain their shape when the user presses harder. Without constraints, the hand suffers deformation artifacts at joints. With constraints, instead, deformable joints are compliant under small rotations, but they become very stiff under large rotations.

We have compared the performance of our nonlinear Jacobi solver to standard solvers for nonlinear constrained optimization discussed in Section II-B, specifically, SQP with PGS for the LCP on each SQP iteration, and SQP with projected Jacobi line-search (PJLS) for the LCPs. During a 10-frame interval centered on the hand configuration shown in Fig. 3, the number of active constraints is high, 335 on average. SQP-PGS runs at an average of 313 ms per time step, SQP-PJLS at 282 ms, and our nonlinear Jacobi solver at 22 ms. This yields an average speed-up of 14.2× w.r.t. SQP-PGS and 12.8× w.r.t. SQP-PJLS on highly constrained scenarios. In all three solvers we exploit the Cholesky factorization and parallelizations discussed in Section III; the speed-up is thanks to the design of our solver, not the optimizations.

Fig. 5 shows the prototypical convergence rate for the three solvers, obtained for one of these test frames. With SQP-PGS and SQP-PJLS, the residual of linearized constraints is quickly reduced within each LCP, but is not a good indicator of the full nonlinear residual, which is evaluated at the beginning of each SQP iteration and exhibits much slower convergence. Our solver reaches a good compromise, with a steady reduction of the nonlinear residual.

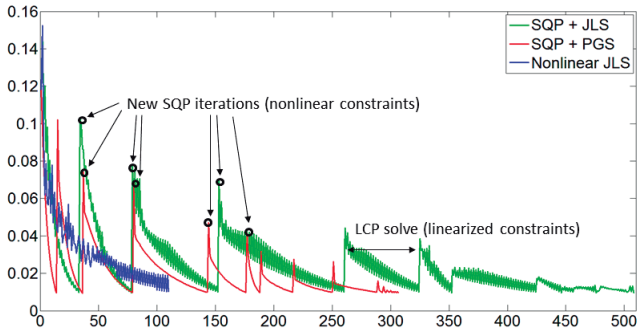

Fig. 5. Comparison of convergence rate of our solver (Nonlinear JLS, in blue) vs. SQP with projected Gauss-Seidel (SQP-PGS, in red) and SQP with projected Jacobi line-search (SQP-JLS, in green). The plot shows the constraint residual vs. iteration count on a prototypical frame of hand simulation. SQP-PGS and SQP-JLS exhibit a fast reduction of the residual of linearized constraints during LCP solves, but slow reduction of the full nonlinear residual, which is evaluated at the beginning of new SQP iterations.

All in all, with our nonlinear Jacobi solver, we have ensured that the simulation of the hand remains interactive at all times. It runs at an average of 119 fps during free motion, at an average of 89 fps on moderately constrained situations (less than 100 active constraints), and at an average of 54 fps on highly constrained situations (more than 100 active constraints).

Finally, Fig. 1 and Fig. 6 show the application of our solver in the context of dexterous manipulation and grasping with multi-finger tactile rendering. Please see also the accompanying video. In Fig. 1, an object is rotated using three fingers. During the manipulation, the relative orientations of the finger pads and the object's surface change, and this subtle change is captured by the orientation of the cutaneous interface. In Fig. 6, the user attempts to lift an object. In the two left columns, with a weak grip, the object slides and the user fails to lift it. In the two right columns, with a strong grip, the user succeeds to lift the object. The strength of the grip is conveyed by the cutaneous interfaces through their normal displacements w.r.t. the finger pads.

VI. Discussion

In this work, we have completed a framework for visuo-haptic rendering of soft grasping and manipulation. Building on previous components for skeleton-flesh coupling, nonlinear skin modeling, and tactile rendering for cutaneous interfaces, we now incorporate an efficient solver for constrained dynamics that enables interactive simulation of a full hand with nonlinear skin for multi-finger tactile rendering. As discussed throughout the paper, this solver differs from classic solvers for contact mechanics, which fail to offer a satisfactory convergence rate under strain-limiting constraints.

The major limitation of the solver, unfortunately mutual to constrained optimization solvers, is that its cost is superlinear w.r.t. the size of the problem. The cost per iteration is bilinear in the number of degrees of freedom and active constraints. In addition, the iteration count of relaxation approaches does

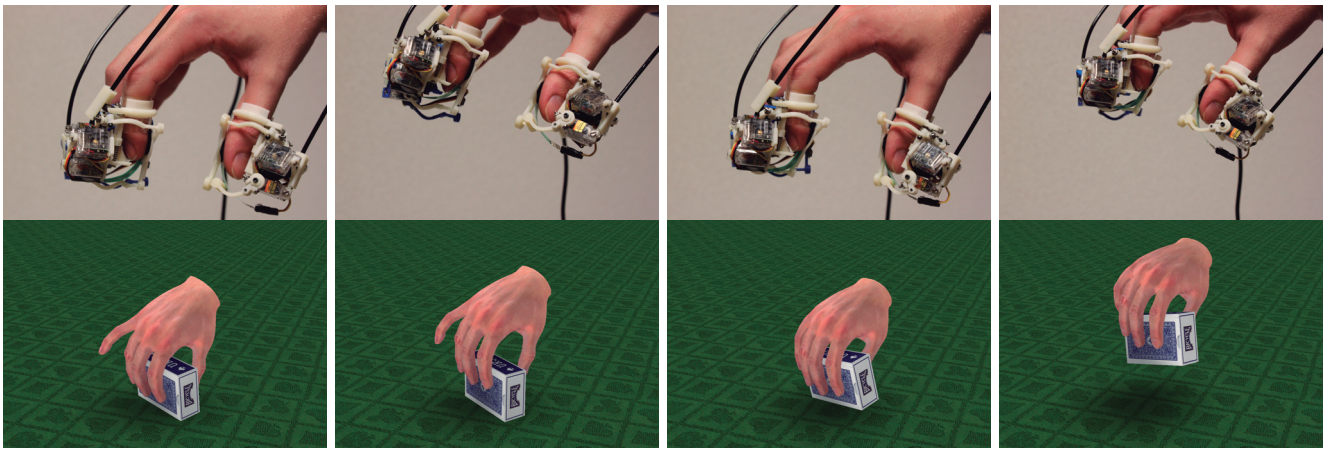

Fig. 6. Lifting an object with multi-finger tactile rendering. With a weak grip (2 left columns), the object slides and the user fails to lift it. With a strong grip (2 right columns), the user succeeds to lift the object. Please pay attention to the subtle (but visible) differences in the normal pressure exerted by the cutaneous devices, which conveys the strength of the grip to the user.

not scale well with the size of the problem. Multiresolution strategies might address this limitation.

Concerning the overall visuo-haptic rendering framework, several elements could be improved. With our coarse deformation model it is not possible to resolve high-resolution deformations, such as the skin indentation produced when touching an edge. The Leap Motion tracker is not sufficiently robust, as it is designed for tracking a bare hand, not a hand wearing cutaneous interfaces. And the tactile rendering algorithm builds on a hypothesis of geometric contact matching, which does not support correct rendering of friction forces or interaction with soft bodies.

To conclude, now that a full visuo-haptic rendering framework of multi-finger interaction is in place, it is time to evaluate the impact of the various components on grasping and manipulation task performance, and refine them accordingly.

Acknowledgments

The authors wish to thank José San Martín for his help with the devices, and Carlos Garre for his help with hand simulation. This project was supported in part by grants from the EU FP7 (project no. 601165 WEARHAP), the European Research Council (ERC Starting Grant no. 280135 Animetrics), and the Spanish Ministry of Economy (TIN2012-35840). The work of Gabriel Cirio was funded in part by the Spanish Ministry of Science and Education through a Juan de la Cierva Fellowship.

References

[1] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact and friction for cloth animation. *Proc. of ACM SIGGRAPH*, 2002.

[2] R. Cottle, J. Pang, and R. Stone. *The Linear Complementarity Problem*. Academic Press, 1992.

[3] H. Courtecuisse, Y. Adagolodjo, H. Delingette, and C. Duriez. Haptic rendering of hyperelastic models with friction. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 591–596, 2015.

[4] C. Duriez, F. Dubois, A. Kheddar, and C. Andriot. Realistic haptic rendering of interacting deformable objects in virtual environments. *Proc. of IEEE TVCG*, 12(1), 2006.

[5] C. Garre, F. Hernandez, A. Gracia, and M. A. Otaduy. Interactive simulation of a deformable hand for haptic rendering. In *Proc. of World Haptics Conference*, 2011.

[6] F. Hernández, C. Garre, R. Casillas, and M. A. Otaduy. Linear-time dynamics of characters with stiff joints. In *Proc. of V Ibero-American Symposium on Computer Graphics (SIACG 2011)*, June 2011.

[7] D. M. Kaufman, S. Sueda, D. L. James, and D. K. Pai. Staggered projections for frictional contact in multibody systems. *Proc. of ACM SIGGRAPH Asia*, 2008.

[8] S. Kim, C. Laschi, and B. Trimmer. Soft robotics: a bioinspired evolution in robotics. *Trends in Biotechnology*, 31(5):287 – 294, 2013.

[9] W. A. McNeely, K. D. Puterbaugh, and J. J. Troy. Six degrees-of-freedom haptic rendering using voxel sampling. In *Proc. of SIGGRAPH 99*, Computer Graphics Proc., pages 401–408, Aug. 1999.

[10] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[11] M. Ortega, S. Redon, and S. Coquillart. A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):458–469, 2007.

[12] M. Otaduy, C. Garre, and M. Lin. Representations and algorithms for force-feedback display. *Proceedings of the IEEE*, 101(9):2068–2080, Sept 2013.

[13] M. A. Otaduy, R. Tamstorf, D. Steinemann, and M. Gross. Implicit contact handling for deformable objects. *Computer Graphics Forum*, 28(2):559–568, Apr. 2009.

[14] A. G. Perez, G. Cirio, F. Hernandez, C. Garre, and M. A. Otaduy. Strain limiting for soft finger contact simulation. In *World Haptics Conference (WHC), 2013*, pages 79–84, 2013.

[15] A. G. Perez, D. Lobo, F. Chinello, G. Cirio, M. Malvezzi, J. San Martin, D. Prattichizzo, and M. A. Otaduy. Soft finger tactile rendering for wearable haptics. In *World Haptics Conference (WHC), 2015 IEEE*, pages 327–332, 2015.

[16] I. Peterlik, M. Nouicer, C. Duriez, S. Cotin, and A. Kheddar. Constraint-based haptic rendering of multirate compliant mechanisms. *Haptics, IEEE Transactions on*, 4(3):175–187, 2011.

[17] D. Prattichizzo, F. Chinello, C. Pacchierotti, and M. Malvezzi. Towards wearability in fingertip haptics: a 3-dof wearable device for cutaneous force feedback. *IEEE Transactions on Haptics*, 6(4):506–516, 2013.

[18] R. Tonge, F. Benevolenski, and A. Voroshilov. Mass splitting for jitter-free parallel rigid body simulation. *ACM Trans. Graph.*, 31(4):105:1–105:8, 2012.

[19] D. Wang, X. Zhang, Y. Zhang, and J. Xiao. Configuration-based optimization for six degree-of-freedom haptic rendering for fine manipulation. *Haptics, IEEE Transactions on*, 6(2):167–180, 2013.

[20] K. Yamane and Y. Nakamura. Stable penalty-based model of frictional contacts. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*, pages 1904–1909, 2006.