Dextrous Manipulation from a Grasping Pose

C. Karen Liu

Georgia Institute of Technology

Abstract

This paper introduces an optimization-based approach to synthesizing hand manipulations from a starting grasping pose. We describe an automatic method that takes as input an initial grasping pose and partial object trajectory, and produces as output physically plausible hand animation that effects the desired manipulation. In response to different dynamic situations during manipulation, our algorithm can generate a range of possible hand manipulations including changes in joint configurations, changes in contact points, and changes in the grasping force. Formulating hand manipulation as an optimization problem is key to our algorithm's ability to generate a large repertoire of hand motions from limited user input. We introduce an objective function that accentuates the detailed hand motion and contacts adjustment. Furthermore, we describe an optimization method that solves for hand motion and contacts efficiently while taking into account long-term planning of contact forces. Our algorithm does not require any tuning of parameters, nor does it require any prescribed hand motion sequences.

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

Keywords: Character animation, physics-based animation

1 Introduction

Sometimes, the factors that break the illusion of reality in synthetic human character animation are the most subtle. Among the most detailed, and hardest motions to convey realistically is dextrous manipulation of an object. While there has been much research in computer animation on rendering a robust grasp of an object, comparatively little work has been done on how to manipulate the object once it is in grasp. The predominant strategy for manipulating in-grasp objects in robotic applications is to alter the grip force, in other words, the internal joint torques. In contrast, people employ several different strategies when manipulating held objects including changing the internal joint torques, shifting the contact points, and altering the hand configuration. While changes in joint torque usually have little visual effect, these alternate strategies can produce a rich variety of natural looking manipulations, such as finger reshaping, sliding and rolling contacts, or changes in contact area.

In this paper, we introduce an algorithm for generating rich detailed hand manipulation motions of an object that is in-grasp. Our algorithm takes as input an initial grasp pose and a desired object trajectory, and generates realistic manipulation motion that effects the desired manipulation. In response to different dynamic situations, our algorithm can generate a range of possible hand manipulations including changes in joint configurations, in contact points, and in the grasping force. To date, designing dynamic controllers that produce changes in contact points or joint configurations once the object is in grasp is a very difficult problem. Consequently, we view our algorithm as an important missing piece in the goal of fully automated generation of end-to-end hand object manipulation motions. It is complementary to existing algorithms that focus on generating robust grasping, and vastly expands upon the range of motions possible once the object is in-grasp. From the user's perspective, our algorithm eases the burden of animating hand motion consistent with the laws of physics while giving the user absolute control to determine the course of manipulation by specifying the movement of the object.

We formulate dextrous object manipulation as a constrained optimization problem where the constraints derive from physical laws and the input trajectory, and the objective function penalizes deviation from the equilibrium joint torques computed at the input grasping pose. Our objective function prefers manipulation strategies that lead to visibly apparent, and as our results show, realistic hand-object interactions. Our approach inherits the advantages that optimization-based approaches exhibit in other character animation applications, most notably the lack of tuning physical parameters, and ease of user input. Dextrous object manipulation poses special challenges for traditional optimization algorithms, however, due to the need to model hand-object contacts. Hand-object contacts vastly increase the state-space and add nonlinear constraints to the problem, rendering this domain intractable to standard optimization algorithms.

To handle the scale of optimization problems generated by our system, we decompose the large nonconvex optimization into a sequence of short-horizon optimizations, each of which yields the hand motion for a small window of time. Often this manner of decomposition will result in unnaturally jerky motions due to the lack of coordination between individual optimization steps. To mitigate this issue, our algorithm informs each short-horizon optimization with the future information extracted from the object motion. The algorithm consists of two interleaving processes: contact force planning and hand motion synthesis. Given the current contact points on the object, the first process plans future contact forces based on the object motion in the future . The second process then uses these projected contact forces as guidance to solve for a short-horizon optimization that yields the current hand pose and the contact positions.

Our algorithm works well for various types of manipulation including moving, rotating, and compressing objects under arbitrary external forces. By combining these basic manipulations, our method can produce interesting bimanual manipulation tasks, such as twist-opening a bottle. However, our method focuses on the continual dynamic adjustments from the grasping pose. Motions that require complex path planning or drastic contact re-planning are beyond the scope of our method.

The key contributions of this work are: 1. A physics-based algorithm capable of synthesizing detailed hand movement during manipulation of objects. 2. A novel solver for a large spacetime optimization that interleaves a simplified long-term planning process with a sequence of short-horizon problems.

2 Related work

Synthesis of hand motion is an increasingly active research area in computer animation. Many researchers have utilized physical simulation via dynamic controllers to generate different classes of hand motions. Pollard and Zordan [2005] proposed a grasp controller where the parameters are automatically determined from captured motion sequences. Their method can be integrated seamlessly with ours as we focus on motion adjustment to external changes after the object is in grasp. Kry and Pai [2006] used captured hand motion and contact forces to extract joint compliances. By adjusting the joint compliances, the same captured grasping motion can be adapted to new objects with different properties. Our method does not explicitly compute joint compliances. Instead, we rely on the equilibrium joint torques computed at the input grasping pose to capture the effect of joint compliances for a set of poses similar to the grasping pose. Many researchers have constructed anatomically realistic models to simulate unconstrained hand animation [Albrecht et al. 2003; Tsang et al. 2005; Sueda et al. 2008]. These methods approximate hand anatomy by modeling muscles, tendons, and their interdependency. Due to the complex interaction among various components and computation of muscle activation, these methods have not been applied to hand-object manipulation.

Grasping motions for manipulation can also be generated via forward and inverse kinematics approaches [Aydin and Nakajima 1999; Huang et al. 1995; Koga et al. 1994]. In addition to grasping motions, previous work has explored other types of manipulation such as gesturing or playing musical instrument [Kim et al. 2000; ElKoura and Singh 2003; Majkowska et al. 2006]. Although these methods are able to synthesize detailed finger movements, the motions do not respond to the dynamic changes in the environment. Our method also synthesizes highly detailed hand movements. However, the output motion is completely determined by dynamic equations of motion, rather than dictated by prescribed poses or rule-based algorithms. Consequently, our algorithm can be applied to a larger variety of situations without needing any tuning.

Optimization-based approaches, similar to our algorithm can produce physically realistic motion with active control [Witkin and Kass 1988; Safonova et al. 2004; Liu et al. 2005]. Our work resembles [Liu 2008] in that we synthesize physically correct hand motion directly from a sequence of short-horizon optimizations. In their method, the kinematic goals are realized through kinematics objectives which are optimized along with other dynamic objectives, such as minimizing changes of torques over time. These objectives are often conflicting with each other and rely on careful adjustment of weights to balance the kinematic goals and the dynamic realism. Consequently, the hand motion appeared rigid and relied on global arm movement to achieve the kinematic goals. Our approach interprets kinematic goals in terms of required contact forces derived from a long-term planning process. These contact forces are enforced directly in the dynamic equations of motion, leaving a very clean objective function, whose single goal is to maintain the desired joint actuation.

Selecting the appropriate grasping pose is a crucial problem during the preshaping phase of grasp synthesis. Many rule-based and data-driven algorithms have been proposed in the fields of robotics and computer animation. Our work does not focus on grasp synthesis but is complementary to any existing algorithm that produces physically plausible grasping poses. Once an appropriate grasp is selected, an important problem is to solve for optimal contact forces such that the object is dynamically stable and held by the desired grasp. Previous approaches linearized the friction constraints and solved the contact forces using various optimization techniques [Kerr and Roth 1986; Nguyen 1986; Cheng and Orin 1990; Bicchi 1992]. Our contact force planning algorithm also linearizes the friction constraints on a point contact. We formulate a quadratic program to solve for a sequence of contact forces that balance external forces on the object during manipulation.

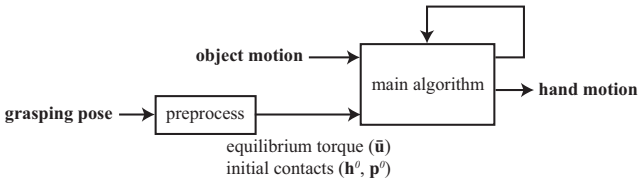

Figure 1: *Overview of the system.*

Previous work in robotics has investigated other manipulation strategies, such as regrasping, finger gaiting, controlled slippage, or rolling contacts [Tournassoud et al. 1987; Fearing 1986; Brost 1988; Cai and Roth 1987]. These methods typically involve high degree of sophistication in designing appropriate dynamic control systems. Our work provides a simple alternative to exploring these manipulation strategies without any effort in designing and fine-tuning the dynamic system.

3 Overview

Our algorithm, illustrated in Figure 1, takes as input an initial grasping pose and partial object trajectory, and produces as output a physically plausible hand animation that effects the desired manipulation. The partial object trajectory input comprises the position and orientation of the object and possibly location of environment contacts, at important time frames. The detail level of the trajectory can range from a few keyframes of the object, to the entire object motion sequence for the duration of the animation. The input grasping pose can be provided by any grasp synthesis algorithms or manually designed by the user. Prior to running the main algorithm, we preprocess the input grasping pose to detect initial contacts with the object. Furthermore, we compute the required joint torques that maintain the grasping pose against gravity. These equilibrium torques, denoted by $\bar{\mathbf{u}}$, will be used later in synthesizing the hand motion.

Figure 2 illustrates how the main algorithm produces a hand pose for iteration k via interleaving two processes, contact force planning and hand motion synthesis. The contact force planning phase solves for a small window of contact forces $(\mathbf{F}^k, \cdots, \mathbf{F}^{k+n-1})$ that achieve the next n frames of the object's motion, assuming the contact positions $\mathbf{p}^{k-1}$, solved by the previous iteration, remain the same for the next n frames. If the object's motion cannot be achieved under the laws of physics, we use an iterative algorithm to generate additional hand-object contacts until the problem becomes feasible.

In the hand motion synthesis phase, we take as input the current contact forces $\mathbf{F}^k$ and possibly additional contacts from the contact-planning phase and synthesize the hand configuration for the current time step. We formulate a short-horizon optimization that solves for the hand pose $\mathbf{q}^k$, hand torques $\mathbf{u}^\mathbf{k}$, contact positions in the object coordinates $\mathbf{p}^k$, and the matching position in the hand coordinates $\mathbf{h}^k$ for the time frame k. Along with the geometric and dynamics constraints, we introduce an objective that maintains the same torque usage as the equilibrium torques $\bar{\mathbf{u}}$. To complete the iteration, we update the contact positions with new $\mathbf{p}^k$ for the next contact force planning phase, and advance the window by one frame to the next iteration. In iteration $k+1$, the contact force from $\mathbf{F}^{k+1}$ to $\mathbf{F}^{k+n}$ will be re-planned based on the new contact locations $\mathbf{p}^k$.

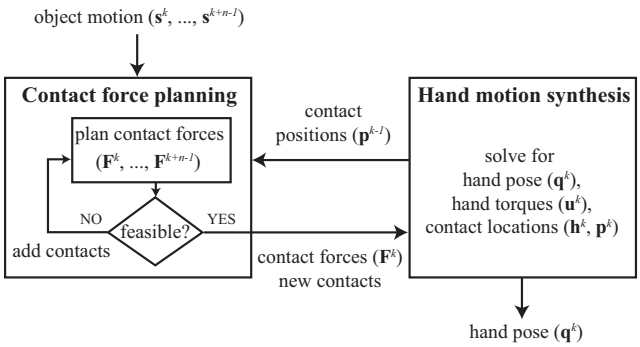

Figure 2: *The main algorithm synthesizes the hand motion via an iterative process. This figure illustrates the operations at time frame k.*

4 Contact force planning

To achieve the desired object motion, we must determine a set of suitable contact forces according to the laws of physics. Furthermore, the hand-object contact forces should be as smooth as possible over time, as human hand tends to avoid generating abrupt changes in the manipulative forces. To this end, we formulate an optimization to plan contact forces that avoid abrupt movements and anticipate the changes of the object in the near future.

At each iteration k, the goal of the contact force planning is to compute the contact forces ($\mathbf{F}^k, \cdots, \mathbf{F}^{k+n-1}$) that realize the object motion in the next n frames ($\mathbf{s}^k, \cdots, \mathbf{s}^{k+n-1}$), given the contact positions in the object coordinates , $\mathbf{p}^{k-1}$, from the previous iteration. $\mathbf{F}^t = \{\mathbf{f}_1^t, \cdots, \mathbf{f}_m^t\}$ consists of all the hand-object contact forces at a particular time instance. m indicates the number of contact points at frame t. If the user specifies some environment contacts at frame t, we also include object-environment contact forces as free variables in $\mathbf{F}^t$. Because the optimization window slides one frame forward at each iteration, the contact forces $\mathbf{F}^k, \cdots, \mathbf{F}^{k+n-2}$ are re-planned in iteration k. The re-planning of contact forces is necessary because a new event occurring at frame $k+n-1$ could make the previously planned contact forces suboptimal. Furthermore, the hand motion synthesis step could change the contact positions used in the contact force planning at iteration $k-1$.

4.1 Enforcing physical realism

The generated contact forces must satisfy both the dynamic equations of object motion and frictional constraints. The equation governing object motion is shown in equation 1.

$$\mathbf{M}(\mathbf{s})\ddot{\mathbf{s}} + n(\mathbf{s}, \dot{\mathbf{s}}) - \sum_{i=1}^{m} \mathbf{J}_i^T \mathbf{f}_i = \mathbf{0} \tag{1}$$

We drop the superscript indicating the time frame for clarity. The first two terms combine the inertial and the gravitational forces applied on the object. The third term sums the currently active contact forces from both the hand and the environment. $\mathbf{J}_i$ is a 3×6 Jacobian matrix evaluated at the global positions of contact $\mathbf{p}_i$. We implicitly enforce frictional constraints by representing contact forces as an additive function of basis vectors approximating Coulomb's friction cone.

$$\mathbf{f}_i = \mathbf{B}_i \boldsymbol{\lambda}_i, \quad \boldsymbol{\lambda}_i \in \mathbf{R}^4, \, \mathbf{B}_i \in \mathbf{R}^{3\times 4} \tag{2}$$

The columns of $\mathbf{B}_i$ represent the basis vectors with nonnegative matching coefficients $\boldsymbol{\lambda}_i$. $\mathbf{f}_i$ in Equation 1 can then be represented by free variables $\boldsymbol{\lambda}_i$ via Equation 2. Depending on whether the state of contact is resting or sliding, the basis vectors can span a pyramidal space or one dimensional direction, as shown in (Figure 3). We set the limits on $\boldsymbol{\lambda}_i$ such that it can not generate gripping force more than 30N.

We use the following procedure for determining whether the state of contact is resting or sliding for the current and near future time frames. For the environment-object contacts, the input object motion provides full information about the contact states. For the hand-object contacts, we assume no slippage will occur in the next n frames and use a static contact model to approximate contact forces. However, if the relative velocity at the contact i is nonzero at iteration $k-1$, we use a dynamic contact model to compute $\mathbf{f}_i^k$, the first contact force in the window. If the hand motion continues to slide after the motion synthesis phase, the contact-planning in iteration $k+1$ will use an updated $\mathbf{p}_i^k$ and model $\mathbf{f}_i^{k+1}$ as a sliding contact.

4.2 Convex quadratic program

In addition to physical realism, as represented by Equation 1, we also want to favor hand motions that are smooth over time. To reduce the discontinuity in contact forces, we define a simple objective function that minimizes the changes in contact forces over time:

$$G(\boldsymbol{\lambda}_i, t) = \|\mathbf{B}_i(\boldsymbol{\lambda}_i^t - \boldsymbol{\lambda}_i^{t-1})\|^2 \tag{3}$$

Since Equation 1 is linear in variables $\boldsymbol{\lambda}$, we can formulate a simple convex quadratic program (QP) that solves the force planning problem efficiently.

$$\underset{\boldsymbol{\lambda}}{\operatorname{argmin}} \sum_{t=k}^{k+n-1} \sum_{i=1}^{m} G(\boldsymbol{\lambda}_i, t) \quad \text{subject to} \begin{cases} \text{Equation 1} \\ \boldsymbol{\lambda} \text{ limits} \end{cases} \tag{4}$$

4.3 Additional contacts

When the QP is infeasible, it indicates that the current contacts cannot generate enough forces to manipulate the object with given friction coefficients and the strength of the hand. To handle these situations, we rely on the observation that humans often increase the contact forces by adding contact points at the most convenient location to the existing grasp, rather than replanning the entire grasp configuration. Since we can efficiently test whether a given set of contact points is feasible by solving a LP feasibility problem (Equation 4 without the objective function), our algorithm can afford to iteratively add new contacts until the problem becomes feasible.

We consider the current hand pose and the contact forces when adding the new contacts. Our algorithm cycles through the list of existing contact points sorted by the magnitude of their contact forces in a decreasing order. Suppose $\mathbf{h}_i$ is an existing contact point on the hand and S_i is the link of the hand skeleton where $\mathbf{h}_i$ resides. The new contact point on the hand, $\mathbf{h}_{new}$, is selected as the closest point to $\mathbf{h}_i$ that is not on S_i, while the corresponding contact point on the object, $\mathbf{p}_{new}$, is computed as the closest point from the object to $\mathbf{h}_{new}$. The initial position of the new contact point might not be achievable by the hand, but it will be recomputed at the subsequent hand motion synthesis step. Therefore, the algorithm for adding contact points is essentially only choosing which link the new contact point resides. The exact position of the new point is determined in the hand motion synthesis.

The new contacts are usually added to the last frame of the window because any infeasibility in previous frames were resolved in the

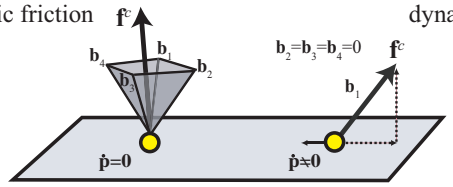

Figure 3: *Friction basis vectors for static and dynamic friction forces.*

previous iterations. On a rare occasion, the change of contact location from the hand synthesis might cause the infeasible situations for the entire window of frames. In that case, our algorithm deems the grasp pose provided by the user unsuitable for the desired object motion.

5 Hand motion synthesis

Given the current contact forces and future additional contact points from the planning phase, the process of hand motion synthesis optimizes the hand pose $\mathbf{q}$, hand torques $\mathbf{u}$, and contact positions in the hand coordinates $\mathbf{h}$ and in the object coordinates $\mathbf{p}$ at the current time instance. Although each optimization is solved independently, the current set of contact forces are generated from the prior object motion.

The hand motion must obey the laws of physics when interacting with the object through the planned contact forces. We allow the contact locations on the hand and on the object to change as long as proper contact is maintained via geometric constraints. Among all the hand motions that satisfy the dynamic and geometric requirements, our algorithm favors solutions that accentuate detailed hand motion and contact changes.

5.1 Dynamics of the hand

The dynamic equations of hand motion in the generalized coordinates can be expressed as follows.

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + n(\mathbf{q},\dot{\mathbf{q}}) + \sum_{i=1}^{m} (\mathbf{J}_i(\mathbf{q},\mathbf{h}_i))^T \mathbf{B}_i(\mathbf{p}_i)\boldsymbol{\lambda}_i - \mathbf{u} = \mathbf{0} \qquad (5)$$

The first two terms compute the inertial and the gravitational forces applied on the hand. Since our motion has very little acceleration, the inertial effect is relatively small comparing to other forces. The third term computes the contact forces from the object using the same linearized friction model as described in Equation 2. Unlike Equation 1, the Jacobian matrix now depends on the free variables $\mathbf{q}$ and $\mathbf{h}$. The basis vectors $\mathbf{B}$ of contact forces depend on free variables $\mathbf{p}$, as the orientation of the friction cone depends on the normal vector at $\mathbf{p}$. The last term $\mathbf{u}$ indicates the joint torques applied internally by the hand.

There are many ways to balance Equation 5 when the contact forces change due to the manipulation or external perturbations. The most straightforward way is to simply change the joint torques $\mathbf{u}$ within a defined range. This strategy works well in robotics applications but is suboptimal for generating expressive hand animation as the results have little visual effect on the hand joint angles or contact points. To highlight the detailed hand movement and contact adjustment, our algorithm prefers to adjust the Jacobian $\mathbf{J}$ through the changes of joint configuration $\mathbf{q}$ and contact points on the hand $\mathbf{h}$, rather than changing the joint torques internally. We accomplish this through an objective function that minimizes the deviation of the hand torques from the equilibrium torques $\bar{\mathbf{u}}$, computed at the input grasping pose: $G(\mathbf{u}) = \|\mathbf{u} - \bar{\mathbf{u}}\|^2$.

5.2 Geometric constraints

In addition to dynamic constraints, each contact point requires a contact constraint to maintain the geometric relation between the hand and the object.

$$\mathbf{E}(\mathbf{q})\mathbf{h}_i - \mathbf{E}(\mathbf{s})\mathbf{p}_i = \mathbf{0} \qquad (6)$$

where $\mathbf{E}$ denotes the transformation from the hand or the object coordinates to the world coordinates. If new contacts are added to the next n frames from the contact planning phase, we enforce Equation 6 with a receding slack that reaches zero at the frame when the contact is established.

Since Equation 6 allows the contact locations to change in both the local coordinates of the object and of the hand, our algorithm allows for slipping and rolling on the surface. To ensure that the slippage is consistent with the planned contact forces, we constrain the relative velocity at the contact according to the contact force.

$$\mathbf{v}^r \equiv \mathbf{T}^{-1}(\mathbf{p}_i^{t-1}) \left[\dot{\mathbf{E}}(\mathbf{q})\mathbf{h}_i^{t-1} - \dot{\mathbf{E}}(\mathbf{s})\mathbf{p}_i^{t-1} \right] \qquad (7)$$

where $\dot{\mathbf{E}}$ indicates the change of the transformation between the current frame and the previous frame (e.g. $\dot{\mathbf{E}} = \frac{1}{\Delta t}(\mathbf{E}(\mathbf{q}^t) - \mathbf{E}(\mathbf{q}^{t-1}))$). $\mathbf{T}^{-1}$ is a matrix that transforms a vector from the world space to the contact surface coordinates whose axes are illustrated in Figure 4(a). Equation 7 yields a 3×1 vector $\mathbf{v}^r$ indicating the relative velocity at the contact in the surface coordinates. If the contact force is well within the static friction cone, we enforce constraints that eliminate the slippage but allow for rolling by setting the tangential components of $\mathbf{v}^r$ to zero: $v_1^r = 0$, $v_3^r = 0$. If the previous state of contact is sliding or the contact force lies on the boundary of the static friction cone, we relax the constraints such that sliding is allowed in the negative direction of the friction force: $v_1^r \leq 0$, $v_3^r = 0$.

5.3 Optimization

Finally, we impose limits on the joint angles and the joint velocity. We combine the geometric and dynamic motion constraints with limits on the joint angles and joint velocity to formulate the optimization problem for the current time frame. This nonconvex optimization is solved by Sequential Quadratic Programming method [Gill et al. 1996]:

$$\underset{\mathbf{q},\mathbf{u},\mathbf{h},\mathbf{p}}{\operatorname{argmin}} \|\mathbf{u} - \bar{\mathbf{u}}\|^2 \quad \text{subject to} \begin{cases} \text{Equation 5, 6, 7} \\ \mathbf{q} \text{ and } \dot{\mathbf{q}} \text{ limits} \end{cases} \qquad (8)$$

We derive the limits on $\mathbf{q}$ and $\dot{\mathbf{q}}$ from a captured motion sequence of finger exercise. The limits are approximately set to cover the entire range of motion.

6 Results

We apply our method to a variety of dextrous manipulations with different grasping poses on different objects. Our hand model consists of 35 degrees of freedom (DOFs): six for the shoulder, two for the elbow, and the rest for the wrist and fingers (Figure 4(b)). Although the algorithm is not intended for online applications, since the object motion needs to be provided in advance, our current implementation can run at 5 frames per second on average using a 2.8G Hz Intel Core 2 Duo processor. Every motion sequence in the supplementary video took less than 15 seconds to generate.

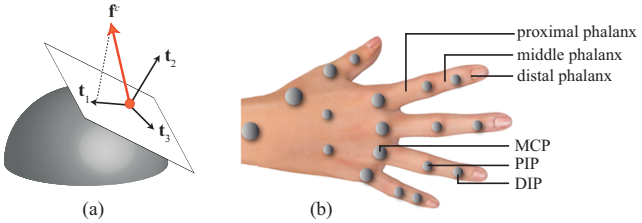

Figure 4: *(a) The surface coordinates are described by three axes: $\mathbf{t}_1$ is in the direction of the friction force, $\mathbf{t}_2$ is the normal vector at the contact point, and $\mathbf{t}_3 = \mathbf{t}_2 \times \mathbf{t}_1$. (b) The hand model contains 6 DOFs for the shoulder, 2 DOFs for the elbow, and 27 DOFs for the wrist and fingers.*

Our algorithm does not require any tuning of the parameters. The variability of the output motion directly comes from the input variables, such as different grasping poses, desired object movement, object mass, or surface materials. The only tunable parameter in the algorithm is the window size n in the contact forces planning phase. The window size reflects the duration of the anticipation in hand motion. For the examples we showed, we set $n = 6$, but any value from 5 to 10 can produce reasonable results.

Translate. Basic translational manipulation moves the object from point A to point B. We synthesize dragging and lifting motion from a grasping pose with three fingers in contact with the object (Finger 5(a)). When dragging a light weight box (0.5 kg) resting on the surface, the hand appears relaxed and similar to the input grasping pose. When we increase the object weight to 1.5kg, the hand starts to change its shape by flexing the MCP joint (Figure 4(b)) and extending the PIP joint of the index and middle fingers. The adjustment is more evident when the object accelerates or decelerates. We also apply the same input pose to generate lifting motion. By increasing the contact area, the hand can generate more grip forces to lift a heavy object. For example, when lifting a 1.5kg box, the hand has to add additional contacts at middle phalanx of the index finger and proximal phalanx of both the index finger and the thumb (Figure 6 left).

Rotate. We can also synthesize hand motion for manipulating the orientation of the object. To rotate and lift a box resting on the table, the hand applies just enough contact forces to pivot around the environment-object contacts because our algorithm favors the motion with equilibrium torque usage. When we apply a two-finger pinch grasp to rotate a small circular object (Figure 5(b)), the hand exploits rolling contacts with the opposed thumb and the index finger moving in the opposite directions. Using rolling contact is considered an optimal strategy by our objective function, because the deviation of torque usage is localized at the thumb and the index finger, while the rest of the arm maintains closely to the equilibrium torque usage. To generate repetitive rolling motion, we disable constrains described in Equation 6 and Equation 7 when the object is not rotating. Without these constraints, the hand will naturally move back to a pose similar to the input grasp due to the objective function.

Press. Our algorithm allows the hand to manipulate simple deformable objects. Similar to other types of manipulation, the required contact forces are computed based on the desired deformation. Although we only use a simple linear spring to model the deformable object, the hand configuration changes realistically when it attempts to flatten the object using a pinch grasp with three fingers

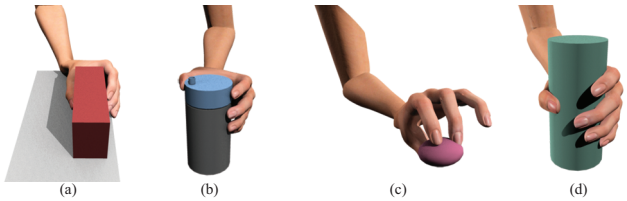

Figure 5: *Different input grasping poses.*

Figure 6: *Different types of manipulation.*

(Figure 5(c)). When applying large contact forces to the deformable object, the index and the middle fingers tend to align the distal phalanx with the surface normal direction and use the finger tips to contact the object (Figure 6 right). The Jacobian matrix formed by this hand configuration allows the same finger torques to generate larger contact forces.

Withstand external forces. The user can also specify the external forces applied on the object during the manipulation. For example, the hand that holds down the bottle while the other hand is twist-opening the lid needs to counteract external forces and torques applied to the bottle (Figure 6 middle). By varying the direction or the magnitude of the external forces and torques, we can synthesize different responses of the hand from a single grasping pose.

Without maintaining equilibrium torque usage. To demonstrate the importance of our objective function, we synthesize a few types of manipulation without maintaining the equilibrium torque usage. In most cases, very little adjustment of the hand can be observed visually as most changes are made in joint torques internally. In the example of rotating a small circular object, the motion without the objective function prefers to rotate the entire arm along with the object, rather that using rolling contacts.

7 Discussion

We introduced an algorithm that synthesizes hand motion from a single initial grasping pose and desired object trajectory. Our optimization-based method synthesizes detailed, varied hand movements, as well as realistic contact phenomena, such as rolling, sliding, and adding more contacts. In addition, our algorithm for solving the optimization problem results in more realistic looking hand motions due to the ability to incorporate future planned contact forces and desired object trajectories into account. Our algorithm is able to synthesize manipulation motions in situations when the object is in the hand grasp, and the desired object trajectory can be achieved without significant changes in contact points, such as complete regrasping.

Our method suffers from a few limitations. If the desired manipulation requires significantly different actuation from the equilibrium torque usage, our algorithm does not yield plausible results. Moreover, the unconstrained fingers occasionally look awkward when

the manipulation involves large contact forces since we do not take into account interdependency among fingers.

Many real-world manipulations involve complex contact planning, such as regrasping and finger gaiting, which cannot be handled by our current algorithm. The simple treatment of additional contact usually results in increasing contact area around the contact point with the largest contact force. In some situations, a better solution would be adding contacts on initially unconstrained fingers or repositioning the initial contacts. As a future work, we would like to investigate more intelligent algorithms to explore other strategies which consider variables like stability, efficiency, or comfort [Rosenbaum and Jorgensen 1992].

Our current algorithm requires all the variables to be differentiable due to the optimization methods we chose. Therefore, the optimization of contact locations is confined to a geometry with continuous representation. For example, the solution for $\mathbf{h}$ must lie on the same ellipsoidal segment of the hand as the initial contact, $\mathbf{h}^0$. To extend our algorithm to polygonal geometries, a discrete search algorithm can be considered when the current point is at the boundary of a smooth surface. The sampling method described in [Popović et al. 2000] can be a promising direction.

Our framework relies on the user to provide an object motion achievable by the input grasping pose. The requirement is reasonable if the object is under full control of the hand and the environment during manipulation. In a dynamic scenario, the object motion is subject to unexpected perturbations that affect the object's trajectory and the responses of the hand. With minor modification, our current algorithm can handle small perturbations by adding object states as free variables and minimizing the deviation from the desired object trajectory. When the perturbations are large, the desired object trajectory might become completely invalid. We would like to explore more sophisticated trajectory planning algorithms to expand the scope of our method.

Acknowledgments

This work was supported by NSF CAREER award CCF #0742303.

References

ALBRECHT, I., HABER, J., AND SEIDEL, H.-P. 2003. Construction and animation of anatomically based human hand models. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 98–109.

AYDIN, Y., AND NAKAJIMA, M. 1999. Database guided computer animation of human grasping using forward and inverse kinematics. *Computers and Graphics 23*, 1, 145–154.

BICCHI, A. 1992. Optimal control of robotic grasping. In *Proc. American Control Con.*, 778–779.

BROST, R. C. 1988. Automatic grasp planning in the presence of uncertainty. *Int. J. Robotics Research 7*, 3–17.

CAI, C., AND ROTH, B. 1987. On the spatial motion of a rigid body with point contact. In *Proc. IEEE Int. Con. Robotics and Automation*, 686–695.

CHENG, F. T., AND ORIN, D. E. 1990. Efficient algorithm for optimal force distribution-the compact dual lp method. *IEEE Trans. Robot Automat. 6* (Apr.), 178–187.

ELKOURA, G., AND SINGH, K. 2003. Handrix: Animating the human hand. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 110–119.

FEARING, R. 1986. Simlified grasping and manipulation with dexterous robot hands. *IEEE Jour. on Robotics and Automation 2*, 188–195.

GILL, P., SAUNDERS, M., AND MURRAY, W. 1996. Snopt: An sqp algorithm for large-scale constrained optimization. Tech. Rep. NA 96-2, University of California, San Diego.

HUANG, Z., BOULIC, R., AND THALMANN, D. 1995. A multisensor approach for grasping and 3-D interaction. In *Computer Graphics International '95*.

KERR, J., AND ROTH, B. 1986. Analysis of multifingered hands. *Int. J. Robotics Research 4*, 3–17.

KIM, J., CORDIER, F., AND MAGNENAT-THALMANN, N. 2000. Neural network-based violinists hand animation. In *Conference on Computer Graphics International*, 37–44.

KOGA, Y., KONDO, K., KUFFNER, J., AND LATOMBE, J.-C. 1994. Planning motions with intentions. In *SIGGRAPH*, 395–408.

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

LIU, C. K., HERTZMANN, A., AND POPOVIĆ, Z. 2005. Learning physics-based motion style with nonlinear inverse optimization. *ACM Trans. on Graphics 24*, 3 (July), 1071–1081.

LIU, C. K. 2008. Synthesis of interactive hand animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

MAJKOWSKA, A., ZORDAN, V., AND FALOUTSOS, P. 2006. Automatic splicing for hand and body animation. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

NGUYEN, V. D. 1986. Constructing force-closure grasps. In *Proc. IEEE Int. Con. Robotics and Automation*, 1368–1373.

POLLARD, N. S., AND ZORDAN, V. B. 2005. Physically based grasping control from example. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 311–318.

POPOVIĆ, J., SEITZ, S. M., ERDMANN, M., POPOVIĆ, Z., AND WITKIN, A. P. 2000. Interactive manipulation of rigid body simulations. 209–218.

ROSENBAUM, D. A., AND JORGENSEN, M. J. 1992. Planning macroscopic aspects of manual control. *Hum. Mov. Sci. 11*, 61–69.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensinal, behavior-specific spaces. *ACM Trans. on Graphics 23*, 3, 514–521.

SUEDA, S., KAUFMAN, A., AND PAI, D. K. 2008. Musculotendon simulation for hand animation. *ACM Trans. on Graphics 27*, 3 (Aug.).

TOURNASSOUD, P., LOZANO-PEREZ, T., AND MAZER, E. 1987. Regrasping. In *Proc. IEEE Int. Con. Robotics and Automation*, 1924–1928.

TSANG, W., SINGH, K., AND FIUME, E. 2005. Helping hand: An anatomically accurate inverse dynamics solution for unconstrained hand motion. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 1–10.

WITKIN, A., AND KASS, M. 1988. Spacetime constraints. In *SIGGRAPH*, vol. 22, 159–168.