# Synthesis of Detailed Hand Manipulations Using Contact Sampling

Yuting Ye*
Georgia Institute of Technology

C. Karen Liu†
Georgia Institute of Technology

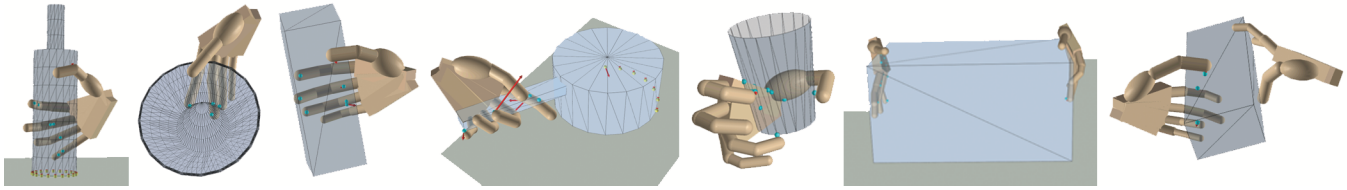


**Figure 1:** *Our algorithm synthesizes detailed hand movements for a wide variety of objects. (Cyan and yellow dots indicate contacts between the object and the hand and between the object and the environemnt respectively. Red arrows indicate contact forces.)*

## Abstract

Capturing human activities that involve both gross full-body motion and detailed hand manipulation of objects is challenging for standard motion capture systems. We introduce a new method for creating natural scenes with such human activities. The input to our method includes motions of the full-body and the objects acquired simultaneously by a standard motion capture system. Our method then automatically synthesizes detailed and physically plausible hand manipulation that can seamlessly integrate with the input motions. Instead of producing one "optimal" solution, our method presents a set of motions that exploit a wide variety of manipulation strategies. We propose a randomized sampling algorithm to search for as many as possible visually diverse solutions within the computational time budget. Our results highlight complex strategies human hands employ effortlessly and unconsciously, such as static, sliding, rolling, as well as finger gaits with discrete relocation of contact points.

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

**Keywords:** physics-based simulation, motion planning, motion capture

**Links:** DL PDF

## 1 Introduction

Synthesis of full-body motion with detailed hand-object manipulation is a very challenging problem in computer animation. The perception of realism depends not only on motions on a grand scale, but also on subtle movements of the hand as it interacts with the environment. As a recent study by Joerg et al. [2010] shows, even very subtle desynchronization errors in body-and-hand motions can be detected by the human eye. As a result, the level of accuracy required for generating believable body-and-hand motion sequences raises significant challenges for existing methods of motion tracking. Existing optical motion capture systems are unsuitable for simultaneous tracking of full-body motion and detailed hand-object manipulation. The camera parameters and placement optimal for full-body tracking do not produce accurate hand motions due to insufficient resolution and occlusion. The most popular method in film industry of multi-resolution tracking of body and hand is to capture the full-body motion of the actors and manually animate the hand. This process usually takes enormous effort, and highly depends on the animators' skills. An alternative approach is to capture the body and the hand separately with different mocap settings, and synchronize the motions as a post-process [Majkowska et al. 2006]. However, performing the same scene multiple times can be difficult, especially when the scene involves incidental contacts with objects. Another option is to use wearable devices such as data gloves [Cyb ] to capture hand motion separately. One potential issue of this approach is that these devices often restrict the movement of the hands for fine manipulation.

This paper introduces a new method for synthesizing human activities with both gross body motion and detailed hand manipulation. We aim to synthesize a rich variety of plausible hand motions that exhibit different manipulation strategies human hands employ, such as contact rolling, sliding, and relocation. Although it is difficult to mocap detailed finger motions, standard optical tracking systems are able to capture motions of rigid objects and the wrists with reasonable accuracy. Therefore, the input to our system is a sequence of full-body motion with accurate wrist movements, and a simultaneously acquired sequence of object motion. The output is a set of detailed, expressive, and physically plausible hand motions that seamlessly integrate with the input. The decision of our system input is crucial to the success of our algorithm because motions of the object and the wrist encode important information of possible hand motions. Movements of the object determine forces and torques that the hand must generate, as well as the locations of the fingers. In addition, motion of the wrist strictly constrains the range of finger movements. These dynamic and kinematic constraints greatly reduce the state space of our problem.

Continuous optimization techniques have been successfully applied to solve manipulation problems with similar constraints. However, they are not suitable for our problem because the distinctive manipulation strategies we aim for often lead to discontinuous constraints. Moreover, the design of an appropriate objective function

*e-mail: yuting@cc.gatech.edu
†e-mail: karenliu@cc.gatech.edu

for the desired outcome remains a difficult challenge as the criteria for optimality is not obvious. Instead of formulating a continuous optimization over joint trajectories, we develop a discrete randomized search algorithm that explores the space of possible hand-object contact positions over time. At each time step, the algorithm stochastically chooses a set of contact points on the object and determines whether they can be achieved kinematically and dynamically from the current state of the hand and the object. Because the goal of our system is to quickly generate as many complete sequences as possible while presenting rich diversity in the solutions, we introduce a few strategies to expedite the randomized search.

The key choice we made in designing our algorithm was to work in the space of contact positions instead of joint angles. The main advantage of this choice is that we can achieve desired manipulation strategies by directly controlling contact positions. In addition, given the trajectory of contact points, the hand motion can be easily reconstructed using inverse kinematics (IK). However, a naïve sampling approach in the space of contact positions is very inefficient because the probability of generating physically accurate contact points from random sampling is extremely low. The key idea of this paper is to utilize physics information of contact to improve efficiency of sampling. At each time step, we first solve for contact forces given the object motion, then use these forces to narrow down the sampling space such that only contact points that satisfy contact dynamics in the next frame will be generated as new samples. For example, if the contact force is on the boundary of the friction cone, the possible contact positions at the next frame will be limited to the opposite direction of current tangential contact force.

Our results demonstrate a rich set of manipulation strategies for various everyday tasks on objects of different shapes and properties (Figure 1). We show that detailed finger movements, such as continuous sliding and rolling as well as discrete relocation of contact points, greatly improve the believability and aesthetics of human motion. Our algorithm is able to discover sophisticated finger gaits and coordinations of both hands without any prior knowledge or data. In addition, our method allows users to control the output motion by editing object properties, such as the motion, the geometry, or friction coefficients of the object.

## 2 Related Work

Creating a natural scene with rich and close interaction between humans and the environment has been a challenging research problem. Many existing approaches combine motion capture data with motion adaptation techniques to synthesize interactions between the character and the objects in the environment. Gleicher [1998] used kinematic constraints to fix the character's hands on the manipulated objects. Yamane et al. [2004] presented a global path planner to synthesize the object's trajectory while maintaining kinematic constraints and naturalness of motion capture data. Ho et al. [2010] introduced a mesh representation to maintain implicit spatial relationship of the scene during motion editing. Jain and Liu [2009] coupled full-body mocap data with manipulated objects via physical simulation. Through the dynamic coupling, human motion adaptation can be driven by the edited motion of the object. In this work, we aim to create interactive scenes in much greater detail than what previous methods produced. The hand motion must be dynamically and kinematically consistent with the objects and the full-body motion, while exhibiting the level of complexity and diversity of real human hands during manipulation.

Many researchers have proposed different approaches to synthesizing detailed hand motions in computer animation. Hand motions can be directly captured from the real world and played back in the virtual world [Majkowska et al. 2006]. However, when the motion involves object manipulation, occlusion and imprecision become major issues. Previous work has applied kinematic approaches to create grasping motions [Koga et al. 1994; Huang et al. 1995; Aydin and Nakajima 1999] or manipulation of musical instruments [Kim et al. 2000; ElKoura and Singh 2003]. These methods add great detail to character animation, but the resulting motions usually lack physical realism and variability. Our method also applies inverse kinematics to generate joint motions for the hand. However, the contact points used to constrain the hand poses are computed in consideration of motion diversity and physical realism.

Physical simulation is another promising approach to synthesizing hand animation [Albrecht et al. 2003; Pollard and Zordan 2005; Tsang et al. 2005; Kry and Pai 2006; Sueda et al. 2008]. Previous methods developed grasp controllers using recorded hand motion [Pollard and Zordan 2005; Kry and Pai 2006] and contact forces [Kry and Pai 2006] . Because the motion is physically simulated, one can apply the same controller to different dynamic situations or objects. Although our method does not focus on realtime simulation, our results can serve as input data to exsiting dynamic controllers. Physically plausible hand motion can also be generated by optimization-based approaches. Liu [2009] formulated a layered optimization that solves for contact forces, contact positions, joint torques, and hand motion. We also use a layered framework to solve contact positions and hand motion in two separate steps, but the contact points are sampled based on contact dynamics rather than emerging as the optimal solution of a single metric. As a result, our approach generates motions with much greater details and diversity.

Detailed manipulation exploiting various manipulation strategies or finger relocation have been extensively studied in robotics. These strategies adjust hand poses and contact positions in concert to achieve more robust manipulation. Common strategies such as controlled sliding and rolling contact [Tournassoud et al. 1987; Cai and Roth 1987; Cole et al. 1992; Cherif and Gupta 1999], or finger gait [Hong et al. 1990; Han and Trinkle 1998; Xu et al. 2007] can largely improve the capability of robotic manipulators. In addition to synthesis, robotics researchers can also transfer manipulation to different situations by analyzing the quality of contact locations [Pollard and Hodgins 2002; Pollard and Wolf 2004] and the relative motion between the object and the wrist [Ciocarlie and Allen 2008; Hamer et al. 2011]. We draw many insights from the robotics literature in developing our method, but ours is fundamentally different in the underlying assumption. We do not assume the manipulation will be stable, nor do we assume any prior knowledge or data of the task are available. Rather, we employ a generic randomized algorithm to automatically and efficiently discover possible strategies consistent with the dynamics of the object and contact constraints.

Sampling-based approach for controlling the outcome of physical simulation has been applied to rigid body [Chenney and Forsyth 2000; Twigg and James 2007] and various character animatioins [van de Panne and Fiume 1993; Ngo and Marks 1993; Sims 1994; Liu et al. 2010; Sok et al. 2007]. For methods involving actively controlled systems, determining a proper sampling space is crucial. For example, Liu et al. [2010] sampled the desired joint angles around the nominal trajectory, Sok et al. [2007] sampled the initial joint configuration of the character, and van de Panne and Fiume [1993] sampled the weights of a neural network for the sensors and actuators of a controlled system. Our method generates random samples in the domain of contact positions on the object. This choice of sampling space has the advantages of providing important constraints to hand poses, and at the same time, being highly constrained by the state of contact forces. The former simplifies the process of creating

final hand motion and the latter greatly reduces the number of required samples. A few applications in computer animation adopt randomized path planning algorithms from robotics literature [Choi et al. 2003; Yamane et al. 2004]. Our problem is different in that the dynamic constraint in contacts defines a narrow and nonsmooth feasible region. While existing path planning methods, such as Rapidly-exploring Random Trees [Lavalle and Kuffner 2000] and Probabilistic Roadmap [Kavraki et al. 1996], are designed for efficient exploration of space. Additional techniques are required to navigate such a complex constrained space.

# 3 Overview

The input to our system is a mocap sequence of an actor performing a full-body motion while physically interacting with objects in the environment. The sequence is acquired using a mocap system calibrated for wide-range full-body motions. The resolution of the system is sufficient to capture the wrist and the object movements, but not enough for fine finger movements. Our goal is to create realistic, detailed hand motions to fill the missing gap between the full-body and the manipulated object.

Our algorithm, illustrated in Algorithm 1, consists of two steps: search of feasible contact point trajectories (Section 4) and reconstruction of hand motion (Section 6). While the second step is necessary to generate smooth final animation, the main contribution of this work lies in the first step.

We formulate the problem of generating feasible contact point trajectory as a randomized depth-first tree traversal. Each level of the tree corresponds to a time instance $t$ in a contact point trajectory. Each node of the tree represents a state, $\mathbf{s} = \{\mathbf{q}, \mathbf{P}, \mathbf{F}\}$, where $\mathbf{q}$ indicates a hand pose, $\mathbf{P}$ indicates a set of contact points between the hand and the object, and $\mathbf{F}$ indicates the corresponding contact forces. Our algorithm recursively expands feasible nodes on the tree from the root to the leaf nodes at level $T$, where $T$ indicates the number of frames of the input animation. At each level $t$, we generate new nodes based on contact dynamics using information $\mathbf{s}^{(t-1)}$ from the previous level $t-1$ (Section 4.2). If a node is kinematically and dynamically feasible (Section 4.3), we continue to the next level $t+1$. Otherwise, we consider this node a dead end. When a feasible path is found, or when a dead end is encountered, we perform a randomized backtracking to explore more possibilities. In Section 5, we describe a few strategies to discover visually distinctive paths efficiently.

**Algorithm 1:** SynthesizeHandManipulation

```
1  𝕊 = {} ;
2  while isTimeLimitReached =FALSE do
3  |   S^(0) = {} ;
4  |   SearchContactPoints(𝕊, S^(0), 1) ;          // Section 4
5  foreach S^(T) ∈ 𝕊 do
6  |   q_h ← ReconstructHandMotion(S^(T)) ;       // Section 6
```

# 4 Search for Contact Point Trajectories

In this section, we introduce the basic algorithm for searching contact point trajectories using a straightforward depth-first search on a tree structure. At each level of the tree, the search algorithm completes two main tasks. First, it generates a small set of nodes as potential expansion of the tree. Second, it tests the feasibility of a node randomly chosen from this set. Before we delve into details of the algorithm, we first define a few useful notations.

## 4.1 Notations

Given the definition of a state $\mathbf{s} = \{\mathbf{q}, \mathbf{P}, \mathbf{F}\}$ from Section 3, we further define a *contact point* $\mathbf{p} \in \mathbf{P}$ as a pair of local coordinates $\mathbf{p}.\mathbf{o}$ and $\mathbf{p}.\mathbf{h}$, on the surface of the object and the surface of the hand respectively. A *contact force* $\mathbf{f} \in \mathbf{F}$ is defined by a nonnegative scalar force $f^{\perp}$ along the contact normal $\mathbf{n}$ that points to the object, and a tangential force vector $\mathbf{f}^{//}$. The force vector applied to the object in Cartesian space is computed as $\mathbf{f}_C = f^{\perp}\mathbf{n} + \mathbf{f}^{//}$.

Our definition of $\mathbf{s}$ contains sufficient information to synthesize hand motions, but the dimension of the state space is too high for the search process. We observe that when the position and orientation of the wrist is fixed, a finger pose can be approximately determined by a single contact point due to kinematic constraints. Thus, we define a more compact representation, a *guiding configuration* $\mathbf{c}$, which consists of up to one contact point for each finger. We term these contact points *guiding points*, $\tilde{\mathbf{p}}$. To recover the corresponding full state from $\mathbf{c}$, we apply inverse kinematics (IK) and collision detection to obtain $\mathbf{q}$ and $\mathbf{P}$. From $\mathbf{P}$ and the input motion of the object, we can solve for the corresponding contact forces $\mathbf{F}$. Therefore, a valid $\mathbf{c}$ sufficiently represents a full state $\mathbf{s}$. Working in the space of $\mathbf{c}$ greatly simplifies the search process illustrated in Algorithm 2.

**Algorithm 2:** SearchContactPoints ( $\mathbb{S}, \mathbf{S}^{(t-1)}, t$ )

```
1  C^(t) ← GenerateNewNodes(s^(t-1)) ;          // Section 4.2
2  success ← FALSE ;
3  nTrials ← 0 ;
4  while nTrials < maxBranchFactor do
5  |   c^(t) ← PickOneNode(C^(t)) ;
6  |   isFeasible, s^(t) ← TestFeasibility(c^(t)) ;  // Section 4.3
7  |   if isFeasible = TRUE then
8  |   |   S^(t) ← Append(S^(t-1), s^(t)) ; // a feasible frame
9  |   |   if t=T then
10 |   |   |   𝕊.push(S^(T)) ;   // a feasible trajectory
11 |   |   |   success ← TRUE ;
12 |   |   else
13 |   |   |   success ←
   |   |   |   success OR SearchContactPoints(𝕊, S^(t), t+1) ;
14 |   |   nTrials ← nTrials + 1 ;
15 return success;
```

## 4.2 Generate New Nodes

To keep the search process tractable, our algorithm explores only states that satisfy the contact dynamics. Specifically, we generate samples of guiding points for each finger based on input motion and contact information from the previous state $\mathbf{s}^{(t-1)}$. Guiding points are then randomly grouped into a set of guiding configurations $\mathbf{C}^{(t)}$. We handle the initial case and the general case of sampling separately as follows.

**Initial case.** An initial case is a frame in the input motion at which a finger establishes contact with the object. We identify them in two preprocessing steps. First, we use the input wrist motion to compute a reachable volume for each finger tip at each frame. Next, we intersect the volumes with the surface of the moving object. We call the intersecting areas on the surface of the object *contact patches*, where kinematically feasible contact points should lie. Figure 2 illustrates this process.

Once we identify the initial frames when intersection takes place for each finger, we create uniform point samples on the contact patches at these frames. Because each contact patch is associated with a finger, we can pair each sample on the patch with the corresponding finger tip to form a guiding point $\tilde{\mathbf{p}}_i^{(t)}$, where the subscript $i$ indicates the finger index and the superscript $t$ indicates the time index. The number of samples is proportional to the area of the patches. In practice, we further prune kinematically infeasible samples in preprocessing to expedite the search process.

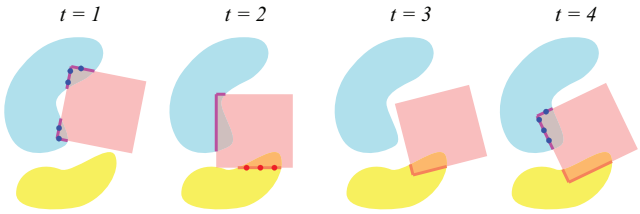



**Figure 2:** *The blue and the yellow blobs represent reachable volumes of two fingers respectively, and the pink square is the object being manipulated. The purple lines and orange lines are contact patches of each finger. We create uniform point samples (dark blue dots and red dots) on contact patches at frames when contacts are established.*

**General case.** . In the general case, we produce a set of new guiding points at level $t$ from $\mathbf{s}^{(t-1)}$ based on contact dynamics: static, sliding, and breaking. They directly correspond to three manipulation strategies: static contact, sliding contact, and relocating contact.

To generate new guiding points, we begin with selecting some *seed points* $\hat{\mathbf{P}}^{(t-1)}$ from $\mathbf{s}^{(t-1)}$. Given all the contact points from the previous level, $\mathbf{P}^{(t-1)}$, we choose the most distal contact point on each finger as a seed point $\hat{\mathbf{p}}^{(t-1)} \in \hat{\mathbf{P}}^{(t-1)}$. From a seed point $\hat{\mathbf{p}}^{(t-1)}$ and its associated contact force $\mathbf{f}^{(t-1)}$, we use the following sampling rules to create a new guiding point, $\tilde{\mathbf{p}}^{(t)}$, for the current level:

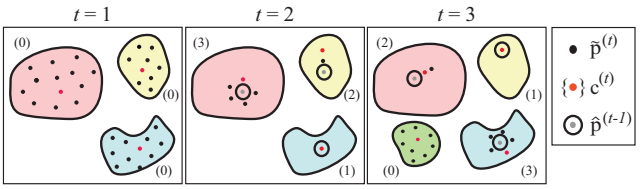
1. If $\mathbf{f}^{(t-1)}$ is within friction cone, the current contact must remain static. $\tilde{\mathbf{p}}.\mathbf{o}^{(t)}$ is at the same location as $\hat{\mathbf{p}}.\mathbf{o}^{(t-1)}$.
2. if $\mathbf{f}^{(t-1)}$ is on the boundary of friction cone, the current contact can remain static or slide. $\tilde{\mathbf{p}}.\mathbf{o}^{(t)}$ can be the same as $\hat{\mathbf{p}}.\mathbf{o}^{(t-1)}$, or be along the direction of $\mathbf{f}^{//(t-1)}$ away from $\hat{\mathbf{p}}.\mathbf{o}^{(t-1)}$.
3. If $\mathbf{f}^{(t-1)} = \mathbf{0}$, the finger can remain static, or release contact and move to a neighboring location. $\tilde{\mathbf{p}}.\mathbf{o}^{(t)}$ can be the same as $\hat{\mathbf{p}}.\mathbf{o}^{(t-1)}$, or be in the neighborhood of $\hat{\mathbf{p}}.\mathbf{o}^{(t-1)}$.

Next, $\tilde{\mathbf{p}}.\mathbf{o}^{(t)}$ is paired with $\hat{\mathbf{p}}.\mathbf{h}^{(t-1)}$ to form a new guiding point $\tilde{\mathbf{p}}^{(t)}$. Figure 3 shows examples of how contact points evolve over time on the object surface.

### 4.3 Test feasibility

After generating a new set of guiding configurations, we randomly select one guiding configuration $\mathbf{c} \in \mathbf{C}^{(t)}$ to test its feasibility. We consider $\mathbf{c}$ kinematically feasible if a penetration-free hand pose can be recovered from $\mathbf{c}$. Likewise, we consider $\mathbf{c}$ dynamically feasible if there exists a set of contact forces consistent with the dynamics of the input object motion.

**Kinematics.** To test kinematic feasibility, we attempt to reconstruct a penetration-free hand pose from $\mathbf{c}$ by solving a sequence of




(0) Establishment (1) Static (2) Sliding (3) Relocating

**Figure 3:** *Each colored region indicates a surface patch for a finger. Three initial patches appear at $t = 1$, and a fourth one appears at $t = 3$. Contact points evolve on the patches over time according to strategies (0)-(3).*

IK problems. The selected guiding configuration $\mathbf{c}$ fails the test if we cannot resolve penetration completely within the iteration limit, or if the fingers inter-penetrate each other.

The IK problem is formulated as a nonconvex optimization to solve for the desired hand pose.

$$\min_{\mathbf{q}} \sum_i \|f(\mathbf{q}, \tilde{\mathbf{p}}_i.\mathbf{h}) - \tilde{\mathbf{p}}_i.\mathbf{o}\|^2 \tag{1}$$

$$\text{subject to} \quad g(\mathbf{q}, \tilde{\mathbf{p}}_i.\mathbf{h})^T \mathbf{n}_i(\tilde{\mathbf{p}}_i.\mathbf{o}) \geq 0, \ \forall i \tag{2}$$

$$|\mathbf{q} - \mathbf{q}^{(t-1)}| \leq \delta\mathbf{q} \tag{3}$$

where $f(\mathbf{q}, \tilde{\mathbf{p}}.\mathbf{h})$ in Equation (1) is the forward kinematic function that computes the position of $\tilde{\mathbf{p}}.\mathbf{h}$ in the local coordinates of the object given pose $\mathbf{q}$. In Equation (2), $g(\mathbf{q}, \tilde{\mathbf{p}}.\mathbf{h})$ outputs the normal direction of the nail on the finger that contains $\tilde{\mathbf{p}}.\mathbf{h}$. By aligning the normal direction of a nail with the surface normal, we avoid many unnatural hand poses. Lastly, Equation (3) prevents large change of hand poses across frames to favor smooth motions. $\delta\mathbf{q}$ determines how fast the fingers can move in a time step.

If the solution pose penetrates the object, we formulate a new IK problem to resolve penetrations. We consider penetrations only on intermediate and distal phalanges because proximal phalanges and the palm have very limited freedom to move given the input wrist motion. For each colliding finger, we use a standard collision detection method to identify a pair of contacts that corresponds to the deepest penetration. The pair is then substituted into Equation (1)-(2) to form a new problem. The solution to the new problem can effectively reduce penetrations. In most cases, a few iterations of this process are sufficient to generate a penetration-free hand pose $\mathbf{q}$. Finally, we collect all contact points between the hand (including the palm and proximal phalanges) and the object in a set $\mathbf{P}$, which usually include the guiding points and other incidental contact points. The guiding points may not be present in $\mathbf{P}$, however, when the point on the object moves to a location that is not reachable by the finger point, or moves across discrete features such as an edge or a corner. Such change of contact is desirable because it can result in finger rolling.

**Dynamics.** Once the proposed guiding configuration $\mathbf{c}$ passes the kinematic test, we further test whether the contact points $\mathbf{P}$ can produce sufficient contact forces $\mathbf{F}$ to move the object in accordance with the input object motion. This dynamic feasibility test can be formulated as a convex conic programming. If a feasible solution cannot be found, we consider that $\mathbf{c}$ fails the dynamic feasibility test. The optimization can be formulated as follows:

$$\min_{\mathbf{F}} \sum_i w_i f_i^{\perp} \tag{4}$$

$$\text{subject to} \quad \sum_i \mathbf{J}_i^T (f_i^{\perp}\mathbf{n}_i + \mathbf{f}_i^{//}) = \mathbf{G} \tag{5}$$

$$\mathbf{n}_i^T \mathbf{f}_i^{//} = 0, \forall i \tag{6}$$

$$\mu f_i^{\perp} \geq \|\mathbf{f}_i^{//}\|, \forall i \tag{7}$$

$$f_i^{\perp} \geq 0, \forall i \tag{8}$$

Equation (5) enforces the dynamics of the object, where the $3 \times 6$ Jacobian matrix $\mathbf{J}_i$ is a transformation between Cartesian coordinates and generalized coordinates. The inertial and the gravitational forces in the generalized coordinates are included in $\mathbf{G}$. To ensure $\mathbf{f}$ is a valid contact force, we constrain the friction direction to be perpendicular to the contact normal (Equation 6) and within the Coulomb friction cone (Equation 7). If the object is also in contact with the environment, we need to solve for environment contact forces simultaneously with $\mathbf{F}$. Because the contact information between the object and the environment is a-priori, we can easily add appropriate environment contact constraints depending on whether the contacts are static or sliding. The above constraints may have many solutions. To encourage contact movements while keeping the problem simple, we minimize the sum of normal forces (Equation 4). This objective exploits friction force as much as possible to satisfy constraints because only normal forces are penalized. As a result, sliding contacts frequently occur in the resultant motion. The minimization also prevents using unnecessary forces so that fingers are more likely to relocate when they are not essential to satisfy constraints. While uniform weighting ($w_i = 1$) works well in most cases, we can fine tune the weights for better performance.

## 5 Create Diverse Solutions

Because the baseline algorithm searches the solution space exhaustively, most computation is spent on discovering very similar trajectories. To improve the baseline algorithm, we introduce four strategies to increase diversity in solutions within the same computation time.

**Sparse exploration.** We can reduce the search complexity by exploiting temporal coherence in a path. Because a static or sliding contact is not noticeable unless it persists for a few consecutive frames, we keep the same manipualtion strategy for a small window of time before branching out to take a new strategy. In other words, instead of treating a single frame as a node, we treat a short sequence of frames as a node. To generate more variations, we use a uniform random variable as the window size, whose expectation is determined by a parameter $\varepsilon$. With this treatment, the search complexity is explicitly controlled by $\varepsilon$, and independent of the length of the motion.

In addition, we choose a small branching factor to reduce computation on visually similar paths, because sibling nodes often represent similar states. When searching for a feasible path, we branch once in every two chances on average. When looking for alternative feasible paths , we choose a branching factor such that on average no more than 2.5 feasible paths can be created for each initial sample. We use these branching factors for all our examples.

**Informed backtracking.** There are many causes of an infeasible frame. If the cause of failure can be propagated backward during backtracking, we can utilize this information to expand different nodes with higher likelihood of success. In our algorithm, we identify three different infeasible situations. If a frame fails on the kinematic test for a particular finger (e.g. unsolvable penetration), we choose a new guiding point for the failing finger and keep guiding points for other fingers unchanged. If an infeasible frame is caused by inter-penetration between two fingers, we choose new guiding points that maximally separate the two fingers apart. Finally, if a frame fails on the dynamics test, we simply choose a new guiding point for every finger.

We can also diversify solutions when backtracking from feasible paths by choosing nodes that are most different from the explored ones. The metric used to measure similarity is a simple summation of Euclidean distance of guiding points.

**Manipulation strategy preference.** Instead of picking a random strategy for each finger every time, we can give high priority to a preferred manipulation strategy. For example, by always choosing static contacts, we can obtain a trajectory with minimum contact changes. Alternatively, we can choose sliding or relocating contacts whenever possible to produce a trajectory with rich movements. More interesting behaviors can emerge if we assign different preferences to different portions of the input motion.

**Contact force preference.** Because the set of possible manipulation strategies are determined by contact forces, we can adjust Equation 4 to indirectly favor different strategies. For example, if contact relocation is preferred, we can assign larger weights to fingers that do not apply force in the previous frame. As a result, once a finger is released, it tends to remain free, provided that the dynamic constraints can be satisfied.

## 6 Reconstruct Hand Motion

After we obtain feasible trajectories of contact points, we apply a spacetime optimization to create smooth hand animations with respect to the contact points.

$$\min_{\mathbf{q}_h} \sum_{t=1}^{T} E_1^{(t)} + wE_2^{(t)}$$

$$\text{subject to} \quad f(\mathbf{q}_h^{(t)}, \mathbf{p}.\mathbf{h}^{(t)}) = \mathbf{p}.\mathbf{o}^{(t)}, \quad t = 1, \cdots, T \tag{9}$$

where the objective term $E_1$ encourages smooth change of joint angles by minimizing acceleration (Equation 10), and $E_2$ favors joints on the same finger having similar bending angles (Equation 11). The subscripts of $q$ indicate DOFs of the same rotation axis on three consecutive joints shown in Figure 4.

$$E_1^{(t)} = \|\mathbf{q}_h^{(t+1)} - 2\mathbf{q}_h^{(t)} + \mathbf{q}_h^{(t-1)}\|^2 \tag{10}$$

$$E_2^{(t)} = \sum \|q_{DIP_z}^{(t)} - 2q_{IIP_z}^{(t)} + q_{PIP_z}^{(t)}\|^2 \tag{11}$$

For fingers in contact, we initialize them with poses solved in the kinematic test. For fingers not in contact, they assume a rest pose specified by users, or a similar pose as neighboring fingers. Non-contacting cases include finger gaiting between contact locations and transition between manipulating different objects. We break down a long sequence into short segments to expedite the optimization. Because contact constraints only exist when fingers exert contact forces, the resultant motion may exhibit penetration artifacts for fingers not in contact. In such cases, we iterate between resolving collisions (as described in Section 4.3) and smoothing joint angles until penetrations are resolved.

Finally, we attach the resultant motions to the wrists of the character, and we complete the reconstruction of a full-body human motion with both locomotion and hand manipulation of objects.

## 7 Results

We applied our algorithm to a variety of hand manipulation tasks, ranging from simple lifting and turning of a box on a tabletop, to a realistic cooking scene that involves objects of different shapes, as well as two-hand manipulation tasks. Our algorithm automatically generates many possible hand motions with rich variations and details. In addition, users can modify object properties, such as geometry, material, or motion, after the data acquisition process. Please see the supplementary video to evaluate our results.

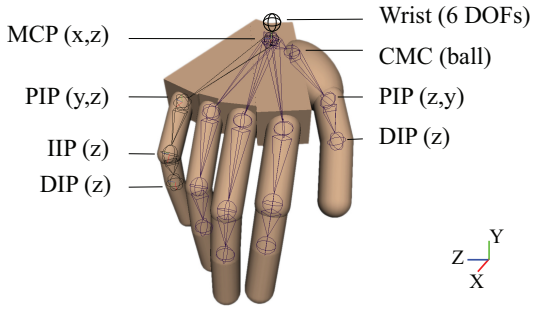



**Figure 4:** *Each finger in our hand model has* 6 *DOFs. Each of the four digits has a* 2*-DOF joint originated near the wrist to model small deformations of the palm.*

We use a hand model with 36 DOFs (Figure 4). The six DOFs on the wrist are given as input and the remaining 30 DOFs on the palm and fingers are synthesized by our algorithm. We solve the nonconvex IK optimization using SNOPT [Gill et al. 1996], and the convex conic program using MOSEK [Andersen et al. 2009]. We use Bullet [Coumans 2005] for collision detection.

**Performance.** We tested our algorithm on a 2.8GHz Intel Core 2 Duo machine running as a single thread. The runtime of optimization highly depends on the number of contacting fingers and contact points. With five contacting fingers, each frame takes 200$ms$ on average for the kinematic test and $5-10ms$ for the dynamic test. The performance of search also depends on the size of feasible regions in the problem domain. Because our method is not designed for real-time, interactive applications, we sometimes trade off performance for more variations in results. For example, we adjust the parameter $\varepsilon$ so that the search algorithm branches less frequently and spends more time to seek solutions with greater variations within the same amount of time. Table 1 shows statistics for one execution of Algorithm 1, which terminates with a desired number of solutions. Due to the stochastic nature of the algorithm, the numbers can vary across different executions. Column 6-10 show the average numbers of 5 executions.

**A cooking scene.** We tested our algorithm in a realistic cooking scene where the actor moves around in a cluttered environment to fetch and manipulate kitchenware of various shapes (Figure 5). Motions of the full-body and the objects were captured using a standard motion capture setting. We segmented the input sequence into short clips to improve the performance of the search algorithm.

Results show that the same algorithm can synthesize detailed hand motions for a variety of shapes and manipulation tasks without any prior knowledge or user intervention. They also show that our algorithm is insensitive to the initial contact location and timing. For evaluation, we manually specified the timing of the initial contact for each finger in the scene of turning a small bottle on the table, and compared the result with the one synthesized automatically (Figure 6). In both motions, the fingers exhibit qualitatively similar behaviors.

(a) motion capture setting

(b) our synthesized result

**Figure 5:** *Our algorithm synthesized detail hand motions for a realistic cooking scene.*

Objects with sharp edges such as the milk box present challenges to the dynamic test due to the discontinuity of normal directions. Grasping near an edge in a simulation could result in inconsistent forces across frames. In reality, grasping on edges tends to make manipulation easier, because the contact area captures a large range of normal directions. To model such phenomena, we approximate an area contact by computing forces on a few proxy points in the neighborhood of an actual contact point, capturing the local features of geometry. As a result, grasping on an edge becomes an available grasp style in our solutions.

**Two-hand manipulations.** Our algorithm can be directly applied to two-hand manipulation tasks. These tasks require the hands to coordinate and apply contact forces collectively (Figure 1). In the example of fiddling a small box with alternating hands, our algorithm accurately estimates the timing and position of finger-object contacts, simply based on the relative motion between the wrists and the object. Another example is to transport a bigger box from a table to the floor. Before lifting, the hands casually reorient the box

| | $\varepsilon$ | T | Time (sec) | Solution | Dead end | Total frame | Success frame | Success ratio | Solution ratio |
|---|---|---|---|---|---|---|---|---|---|
| Milk box (turning) | 10 | 71 | 834 | 10 | 241 | 2710 | 661 | 24.4% | 4.0% |
| Milk box (pick up) | 20 | 101 | 917 | 10 | 2208 | 26588 | 974 | 3.7% | 0.45% |
| Spatula | 20 | 226 | 1323 | 20 | 285 | 6324 | 3735 | 59.1% | 6.6% |
| Pot (with Wine) | 20 | 120 | 636 | 10 | 340 | 3235 | 1144 | 35.4% | 2.9% |
| Two-hand (small box) | 15 | 141 | 666 | 35 | 273 | 8833 | 4901 | 55.5% | 11% |

**Table 1:** *Parameters and statistics of a few examples. "Dead end" indicates the number of infeasible frames encountered. "Total frame" indicates the number of frames explored. "Success frame" indicates the number of distinctive frames on all solution paths. "Success ratio" is the number of success frames to the total number of frames, and "Solution ratio" is the number of solutions to the total number of paths (including both solutions and dead ends) explored. As the number of success frames gets closer to the number of solutions multiplied by T, the solution paths contain more and more distinctive movements. The fact that success ratio is much higher than the solution ratio means our algorithm does not waste computation on dead ends thanks to efficient backtracking.*

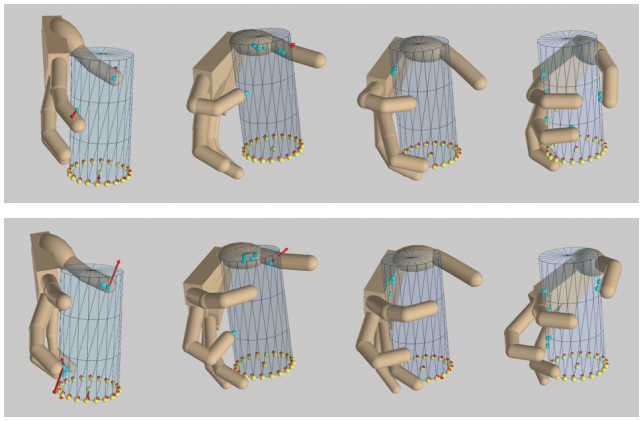


**Figure 6:** *Our algorithm is insensitive to the initial contact location and timing. Top row: automatically synthesized motion; bottom row: manually specified initial contact timing.*

by sliding it on the table. The contact establishment and release generated by our algorithm appear coordinated although no prior knowledge is used. During transportation, the fingers naturally slide and move on the surface of the box due to physical constraints and the relative motion between the wrists and the box.

**Different manipulation strategies.** Users can choose manipulation strategies to create various finger gaits. Our algorithm respects the user preference regardless of the friction condition, but friction affects the execution of the preferred strategy. We first chose the sliding strategy and tested it with different friction conditions. Results show significant finger sliding in all cases. However, the sliding direction changes from the horizontal to the vertical direction as the friction cone widens. This behavior is the result of force optimization exploiting friction force to minimize normal force. We next tested static contacts under low friction and observed minimum finger sliding. However, fingers start rolling when it becomes kinematically difficult to maintain a static contact.

We can also generate finger gaits by changing the weighting scheme in Equation (4). In the example of manipulating a paper cup in hand, the fingers have to roll and relocate contacts asynchronously to provide necessary torques within the kinematic limits. We encourage finger movements by penalizing force usage of free fingers. As a result, the free fingers start applying forces only after they find a new contact location that is more efficient than the existing ones. For the contacting fingers, rolling becomes the only feasible strategy given the dynamics of the cup and kinematic constraints of the hand. It is surprising that such a complex behavior can be synthesized efficiently via a stochastic approach without any prior knowledge or heuristics. Our experiment suggests that much of the finger information is implicitly encoded in the relative angular velocity of the object and the wrist. When we tested on a synthetic object motion with constant angular velocity, the finger gaits appear unnatural.

**Editing object properties.** The ability to adapt the hand motion to various object properties is highly desirable in a production pipeline because virtual props often need to be modified after capturing the actor's performance.

When we modified the distance between the object and the hand, different grasp styles emerged. For example, closer distance results in a power grasp while a longer distance generates a precision grasp. In the cooking scene example, the original spatula motion results in a hand motion that uses mostly finger tips to grip. By moving the spatula closer to the palm, the fingers automatically curl around the spatula to form an envelop grasp and use the palm to exert forces (Figure 7). We can also use the same captured motion on objects with different shapes. We show that replacing a box with a bunny or a mug retains the quality of the original motion as the fingers naturally adapt to the new object with the same wrist motion (Figure 9).

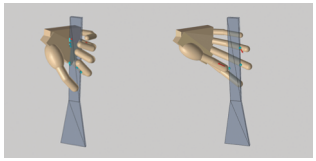


**Figure 7:** *Different grasp styles.*

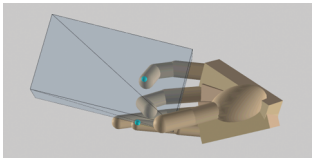


**Figure 8:** *Fiddling with a box.*

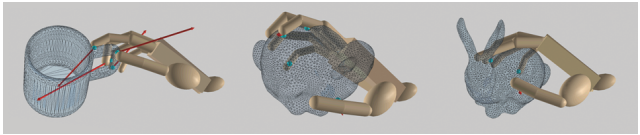


**Figure 9:** *The hand adapts to a mug and a bunny from the same input motions.*

## 8 Evaluation

**Comparison with ground truth.** To evaluate the quality of our results, we compare them side-by-side with close-range motion capture data and video footage. We used a close-range camera setting to mocap a motion of picking up a small box on the table, and used motions of the wrist and the object as input to synthesize several motion sequences of the hand. Within 10 sequences, we were

able to find one that is visually similar to the captured data, although not identical (Figure 10).

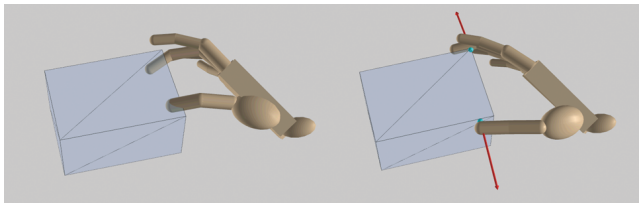


**Figure 10:** *Left: motion capture data. Right: our synthesized result.*

We also recorded a video of an actor fiddling with a box using one hand, and captured motions of the wrist and the object at the same time (Figure 8). The motion exhibits frequent contact movements and complex contact relations among the hand, the object, and the table. Although our solution is different from the actual performance, partly due to the discrepancy in hand modeling, the overall features of finger gaiting and contact sliding were reconstructed in the synthesized motion. However, this challenging example also revealed some drawbacks of our method. For instance, the fingers sometimes penetrated the table during manipulation because we did not consider collisions between the hand and the environment.

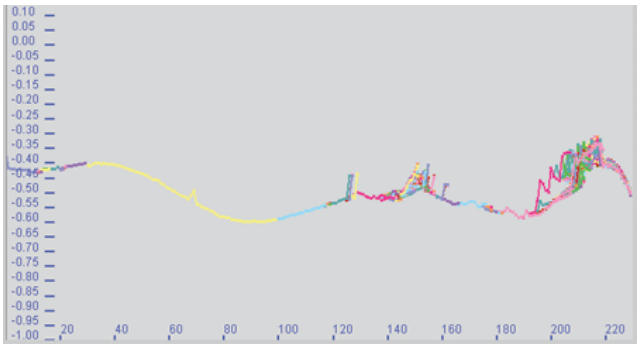
**Path variability.** Our improved search algorithm is more efficient in discovering variabilities in the solution space compared to the baseline. We visualized the search results after exploring the first 3000 nodes on a motion sequence of 230 frames (Figure 11). While the baseline algorithm spent most computation time searching in a narrow space, our algorithm covered a significantly larger space and provided more variations in the solutions. Figure 12 shows some distinctive solutions among the first 50.

**Limitations.** Many of the current limitations are due to the rudimentary hand modeling. A hand made of rigid bodies cannot model continuous area contacts, which are essential to many manipulation tasks. It is also difficult to handle simultaneous contacts with multiple objects using a rigid hand. For example, scooping up a spatula lying on the table from the bottom cannot be handled using the current collision resolution algorithm. In addition, our hand model does not simulate interdependency among fingers. Consequently, the fingers appear to move independently and sometimes unnaturally. Incorporating a more accurate hand model with anatomically correct structure [Deshpande et al. 2009] and deformable skins [Ciocarlie et al. 2007; Jain and Liu 2011] is a fruitful future direction to pursue.

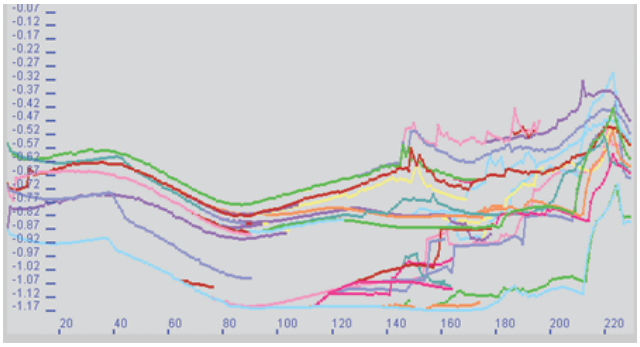
One drawback of using only guiding points for sampling is that the sliding friction constraints might not be satisfied by all the contact points. Although we did not remove such samples during search, we excluded the solutions with this artifact at the motion reconstruction stage. With a reasonable number of solutions, we are always able to find ones that respect constraints for all contact points.

The bottleneck of our current framework is the kinematic test which resolves penetration by solving a few nonconvex optimizations. While the method is straightforward, it fails at some difficult cases such as manipulating a tiny marble or a piece of paper. A more robust and efficient collision resolution routine can greatly improve the performance and capability of our method.

Our search algorithm currently has limited planning capability because it does not anticipate future movements in choosing manipulation strategies. Consequently, it is not efficient in discovering




(a) baseline algorithm




(b) our improved search strategy

**Figure 11:** *Trajectories of a joint on the index finger after searching* 3000 *frames.*

finger gaits that requires longer term planning. We can improve the algorithm in challenging cases by incorporating short horizon or receding horizon planning in search and sampling, and using grasp quality metrics [Miller and Allen 1999; Pollard 2004] for early pruning. Planning is also important for free fingers to avoid penetrations when transitioning between contact points. A more sophisticated pre-grasp and re-grasp planning algorithm will be more suitable for general cases than our current simple treatment.

## 9 Conclusion

We introduce a new method for synthesizing human activities involving both gross body motion and fine manipulation. Given a full-body motion and object motion simultaneously acquired by standard mocap system, our method automatically synthesizes detailed, expressive, and physically plausible hand motions. Our randomized algorithm searches for a set of hand motions that exploit a wide variety of manipulation strategies, such as static, sliding, rolling contact, as well as finger gaits with discrete relocation of contact points.

In this work, we make a deliberate choice not to use any example motions of grasps to test the capability of sampling techniques. Incorporating a database of natural hand poses and a hand model that captures the inter-dependencies of fingers may result in more natural grasps and finger gaits.

Our technique opens up a few interesting future research directions. Our current algorithm allows some limited user control, such as choosing manipulation strategies (more steady or slippery), modulating physical properties of the hands and objects, and modifying the geometry of the objects. In practice, artists often demand direct forward kinematics and inverse kinematics control. We would like

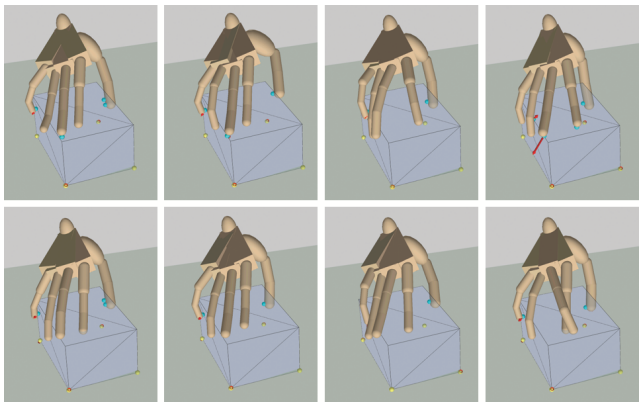


**Figure 12:** *Our algorithm explores a variety of solutions.*

to include these features in the future such that users can directly specify angles for a particular joint or set a contact point at a particular location. Further, an intuitive user interface for visualizing and selecting desired motions from a large pool of solutions can be immensely useful. Similar to the Many-Worlds-Browsing technique [Twigg and James 2007], one possible direction is to create an interactive interface that allows users to browse and adjust parts of the scene with ease. To provide a seamless interaction experience, we can solve many solutions in parallel thanks to the stochastic nature of our algorithm.

## Acknowledgements

We are thankful to Sehoon Ha for the help with motion capture and to Yuting Gu for her effort in Maya modeling and rendering. We would also like to thank Victor Zordan for his valuable suggestions. This work was supported by NSF CAREER award CCF 0742303 and Alfred P. Sloan Fellowship.

## 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.

ANDERSEN, E. D., JENSEN, B., JENSEN, J., AND SANDVIK, R. 2009. Mosek version 6. Tech. Rep. TR-2009-3, Ulf Worsøe, October.

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.

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.

CHENNEY, S., AND FORSYTH, D. A. 2000. Sampling plausible solutions to multi-body constraint problems. In *SISGGRAPH*, 219–228.

CHERIF, M., AND GUPTA, K. K. 1999. Planning quasi-static fingertip manipulations for reconfiguring objects. *IEEE Trans. on Robotics and Automation 15*, 5, 837–848.

CHOI, M. G., LEE, J., AND SHIN, S. Y. 2003. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Transactions on Graphics 22*, 182–203.

CIOCARLIE, M. T., AND ALLEN, P. K. 2008. On-line interactive dexterous grasping. *Eurohaptics* (June).

CIOCARLIE, M. T., LACKNER, C., AND ALLEN, P. K. 2007. Soft finger model with adaptive contact geometry for grasping and manipulation tasks. *IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (March).

COLE, A., HSU, P., AND SASTRY, S. 1992. Dynamic control of sliding by robot hands for regrasping. *IEEE Trans. on Robotics and Automation 8*, 1, 42–52.

COUMANS, E., 2005. Bullet physics engine. http://bulletphysics.org.

*Cyberglove Systems, http://www.cyberglovesystems.com/.*

DESHPANDE, A., KO, J., FOX, D., AND MATSUOKA, Y. 2009. Anatomically correct testbed hand control: Muscle and joint control strategies. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 4416 –4422.

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

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.

GLEICHER, M. 1998. Retargeting motion to new characters. In *SIGGRAPH*, 33–42.

HAMER, H., GALL, J., URTASUN, R., AND GOOL, L. V. 2011. Data-driven animation of hand-object interactions. In *IEEE Conference on Automatic Face and Gesture Recognition*, 360–367.

HAN, L., AND TRINKLE, J. 1998. Dextrous manipulation by rolling and finger gaiting. In *ICRA*, IEEE, 730–735.

HO, E. S., KOMURA, T., AND TAI, C.-L. 2010. Spatial relationship preserving character motion adaptation. *ACM Trans. Graph 29*, 3.

HONG, J., LAFFERRIERE, G., MISHRA, B., AND TANG, X. 1990. Fine manipulation with multifinger hand. In *ICRA*, IEEE, 1568–1573.

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

JAIN, S., AND LIU, C. K. 2009. Interactive synthesis of human-object interaction. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 173–182.

JAIN, S., AND LIU, C. K. 2011. Controlling physics-based characters using soft contacts. *ACM Trans. Graph. (SIGGRAPH Asia) 30* (Dec.), 163:1–163:10.

JOERG, S., HODGINS, J., AND SULLIVAN, C. 2010. The perception of finger motions. *Applied Perception in Graphics and Visualization (APGV)*.

KAVRAKI, L., SVESTKA, P., LATOMBE, J.-C., AND OVERMARS, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. on Robotics and Automation 12*, 4, 566–580.

KIM, J., CORDIER, F., AND MAGNENAT-THALMANN, N. 2000. Neural network-based violinistâs 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.

LAVALLE, S. M., AND KUFFNER, J. J. 2000. Rapidly-exploring random trees: Progress and prospects.

LIU, L., YIN, K., VAN DE PANNE, M., SHAO, T., AND XU, W. 2010. Sampling-based contact-rich motion control. *ACM Trans. Graph.(SIGGRAPH) 29*, 4, 1–10.

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

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

MILLER, A., AND ALLEN, P. 1999. Examples of 3d grasp quality computations. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 2, 1240 – 1246 vol.2.

NGO, J. T., AND MARKS, J. 1993. Spacetime constraints revisited. In *SIGGRAPH*, vol. 27, 343–350.

POLLARD, N. S., AND HODGINS, J. K. 2002. Generalizing demonstrated manipualtion tasks. In *Workshop on the Algorithmic Foundations of Robotics (WAFR '02)*.

POLLARD, N. S., AND WOLF, A. 2004. Grasp synthesis from example: tuning the example to a task or object. In *Workshop on Multi-point Interaction in Robotics and Virtual Reality*, IEEE International Conference on Robotics and Automation, 77–90.

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

POLLARD, N. S. 2004. Closure and quality equivalence for efficient synthesis of grasps from examples. *International Journal of Robotics Research 23*, 6 (June), 595–614.

SIMS, K. 1994. Evolving virtual creatures. In *SIGGRAPH*.

SOK, K. W., KIM, M., AND LEE, J. 2007. Simulating biped behaviors from human motion data. *ACM Trans. Graph. (SIGGRAPH) 26*, 3, 107.

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.

TWIGG, C. D., AND JAMES, D. L. 2007. Many-worlds browsing for control of multibody dynamics. *ACM Trans. Graph 26*, 3.

VAN DE PANNE, M., AND FIUME, E. 1993. Sensor-actuator networks. In *SIGGRAPH*, vol. 27, 335–342.

XU, J., KOO, T.-K. J., AND LI, Z. 2007. Finger gaits planning for multifingered manipulation. In *IROS*, IEEE, 2932–2937.

YAMANE, K., KUFFNER, J. J., AND HODGINS, J. K. 2004. Synthesizing animations of human manipulation tasks. *ACM Trans. Graph 23*, 3, 532–539.