Real-time behaviour synthesis for dynamic Hand-Manipulation

Vikash Kumar, Yuval Tassa, Tom Erez, Emanuel Todorov

***Abstract*— Dexterous hand manipulation is one of the most complex types of biological movement, and has proven very difficult to replicate in robots. The usual approaches to robotic control – following pre-defined trajectories or planning online with reduced models – are both inapplicable. Dexterous manipulation is so sensitive to small variations in contact force and object location that it seems to require online planning without any simplifications. Here we demonstrate for the first time online planning (or model-predictive control) with a full physics model of a humanoid hand, with 28 degrees of freedom and 48 pneumatic actuators. We augment the actuation space with motor synergies which speed up optimization without removing dexterity. Most of our results are in simulation, showing non-prehensile object manipulation as well as typing. In both cases the input to the system is a high level task description, while all details of the hand movement emerge online from fully automated numerical optimization.**

I. INTRODUCTION

Dexterous manipulation is one of the most complex and expressive classes of biological movement. Unlike other dynamic movements such as locomotion – which have been discovered by evolution in many different forms – manipulation appears to be challenging even for biology. Indeed a large part of the primate brain is devoted to controlling the hand and its complex musculo-tendon network. Manual dexterity is perhaps the most distinguishing sensorimotor capability of humans. Therefore, if we aspire to integrate robots into our human-centered world, we must equip them with manipulators and control strategies of similar dexterity.

Dexterous manipulation not only involves effective coordination of a high degree-of-freedom (dof) manipulator, but also requires stabilizing an object that only becomes controllable through contacts. High dof, including mutually coupled dof, operate in a very compact space co-inhabited by the object being manipulated. This results in a large number of contacts and dynamic phenomena such as rolling, sliding and deformation. Movement of objects within the workspace causes contacts to appear and disappear often, which greatly complicates motion planning.

In addition to high dof, a dexterous manipulator must also exhibit a well balanced combination of capabilities such as speed, strength, low loop latency, and compliance. High number of actuated dof in a small space makes it hard to design such manipulators. Dexterous robotic hands exist [1] [2] [3], but the difficulty in controlling them has motivated many researchers to focus on more limited mechanisms. Our goal instead is to develop general-purpose control solutions that can be applied to mechanisms as complex as a human hand.

Most prior work on hand control has focused on achieving stable grasp. While a lot of progress has been made on the theoretical side, practical applications are largely limited to using task-specific manipulators in carefully controlled environments. Part of the difficulty in deploying manipulation in less constrained environments comes from challenges in sensing and state estimation – due to visual occlusions and limited tactile sensing. But control is also challenging. Even if we assume a perfect model, most existing methods for grasp synthesis are too slow to operate in real-time, making it impossible to adapt to an uncertain environment.

For applications of dexterous manipulation in unconstrained human environments, we need the capability to plan in real time. This is because no pre-computed solution or simplifying set of assumptions are likely to work for such a complex behavior, that is so exquisitely sensitive to small variations in contact force or object position. Planning needs to be error tolerant and capable of quickly generating corrective maneuvers that stabilize the object in the face of unexpected disturbances. This is particularly challenging in non-prenehsile manipulation, i.e. in the absence of a statically-stable grasp. Numerical optimization is the only general approach we are aware of that is at least in principle capable of ”inventing” complex movements fully automatically, without intervention from a human. This is why our approach is centered on optimization.

The challenges of legged locomotion bare resemblance to those of dexterous manipulation: in both cases, an effective solution must deal with the forces acting between the robot's body, over which it exerts full control, and the unactuated dynamics – the root joint for locomotion, and the object in manipulation. After decades of research in legged locomotion, we are now capable of producing careful walking with humanoid robots [4] [5]. However these developments have not yet had an impact on hand movement control, in part because they rely on domain-specific abstractions such as ZMP, capture points, inverted pendulum approximations, etc. In fact this may be why manipulation is so much harder: in manipulation, constructing a reduced model that captures the essence of the behavior seems impossible. Any such model will clearly need to include the object and all the hand surfaces that can interact with it, along with the kinematic limitations of the manipulator – so it is not really a reduction.

From a computational perspective, hand movement synthesis is challenging because high dof (both actuated and unactuated) results in high dimensionally of the search space. Inter-finger contacts and object-finger interactions produce

The authors are with the Departments of Computer Science & Engineering and Applied Mathematics, University of Washington, WA 98195, USA
E-mail: {vikash, tassa, etom, todorov }@cs.washington.edu

large number of contacts that impose discontinuities in the search space. The optimization landscape is further complicated by active involvement of dynamic phenomenon such as rolling, sliding and large number of contacts that make and break often.

In this paper we describe our ongoing work in real-time planning (or model-precitive control, MPC) of dexterous manipulation. The most interesting results are obtained in simulation, we present these results in light of a hardware platform we are developing – a ShadowHand skeleton that we have equipped with faster and more compliant actuation [6]. This hardware platform called "ADROIT" is described in section IV. MPC control of object manipulation in simulation is presented in section VI. We also introduce an element which is often considered in hand manipulation but is novel to the MPC setting: namely hand synergies. Synergy-space planning techniques are described in section VI-E.

Our movement synthesis for dynamic hand manipulation builds upon our previous work [7] [8] which was able to synthesize full body movements. However, it was unable to scale up for motion synthesis in case of dexterous hand manipulating owing to difference in the nature of the optimization manifold. After carefully investigating the challenges, we introduce synergy-space planning and demonstrate for the first time online synthesis of dexterous object manipulation. The input to the planner is a high-level task description: we only specify the desired position of the object being manipulated, and the entire hand movement is then synthesised automatically. Similarity, in case of typing, we specify the sequence of desired key presses and the behavior emerges automatically. Instead of imposing these specification as hard constraints, we pose them as costs and mix them with other intuitive costs such as being gentle and moving at a nominal speed.

II. Related Work

Grasping has received more attention from the robotics community than dexterous manipulation. Researchers have approached grasping primarily from two different directions. One approach focuses on improving the design of the manipulator so as to improve its capabilities in terms of producing stable grasps. The other approach exploits optimization based techniques to evaluate the stability of the grasp using grasp metrics [9]. Various grasp metrics has been proposed with no clear advantage of using one over the other. These methods are less likely to generalize for dexterous manipulation but are based on important ideas. One such idea is force closure which can be treated as the ZMP equivalent in case of grasping. Another idea is position the manipulator in pre-grasp configuration and close the finger while relying on the compliance of the joints to grasp the object. Graspit [10] is a tool for simulating and evaluating grasp for different shapes which is quite popular in the community. While grasping has always been the center of hand research, one common approach towards reactive manipulation is tele-operation using data gloves. In tele-operation the hard part of planning and decision making is left to the intelligent user while the controllers blindly follows the joint trajectories. Contact invariant trajectory optimization [11] is an offline method capable of synthesising hand manipulation but its slow and trades physics for visually expressive manipulation behavior. Such approaches are hard to generalizable and are less likely to scale in real applications. Also, see review paper [12] [13]

A. Motion capture techniques

The graphics community has long utilized motion capture system and dimensionality reduction techniques (see below) to synthesize movements for animated characters. While the focus has always been on full body movements, motion capture techniques have been used for hand movement synthesis [14] [15]. These approaches have not been widely successful in the robotics research owing to the real world physics constraints (that can be violated in animation), its inability to generalize and computationally expensive post processing step. An up-to-date discussion on motion capture based hand manipulation can be found in [16]

B. Dimensionality reduction and synergy based control

Although rich and diverse, animal forms exhibit characteristic movements that can be attributed to its morphology, neural system and habitat. Researchers have ascribed these to bio-muscular and neural factors [17] [18]. Low dimensional embedding has been found at the level of kinematics [19], instantaneous muscle activity [20], spatio-temporal muscle activity [21] and feedback control law [22]. These low dimensional embeddings have long been exploited in the graphics community to reduce the dimensionality of search spaces to synthesize full body movements. It has also been demonstrated that such embeddings also exists in hand movements. Approximately 95% of the postural variance associated with hand grasping can be explained using four principal components [23]. Such low dimensional embedding and synergy spaces have been exploited by the robotics community to accelerate the pace of grasping research [10]. However, it has restricted the capabilities of present robotic devices to simple grasps. Similar to biological systems, present day robots have many dofs. While synergies and low dimensional spaces have helped us control and emulate some of the functionalities; they have restricted the behaviours to simple movements that conceal the expressiveness and dexterity of these robots.

III. ONLINE TRAJECTORY OPTIMIZATION

The hand manipulation behaviors presented here are generated autonomously via numerical optimization. At any given moment, the controller has a plan that predicts the system's state over some finite horizon into the future. This plan is constantly optimized in a receding-horizon fashion, and the initial state is set according to the estimated current state. At every iteration, our trajectory optimization algorithm employs a model of the system's dynamics to solve a finite-horizon optimal control problem, an approach known as Model Predictive Control (MPC). Here we use a first-order version of the Differential Dynamic Programming algorithm

[24]. For brevity, here we present only an outline of the algorithm; for more details on our optimization approach, see [7], [8].

A. Finite-Horizon Optimal Control

We formulate the dynamics in discrete time; given the state $\mathbf{x}_i$ and control signal $\mathbf{u}_i$ at time i, we use the model $\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)$ to compute the next state: The optimization is driven by a cost function $\ell(\mathbf{x}_i, \mathbf{u}_i)$ that depends on the instantaneous state and control. We also define a final cost function $\ell_f(\mathbf{x}_N)$ that depends only on the final state of the trajectory.

Given an initial state $\mathbf{x}_0$ and a sequence of controls $\mathbf{U} \equiv \{\mathbf{u}_0, \mathbf{u}_1 \ldots, \mathbf{u}_{N-1}\}$, we compute the resulting state trajectory by repeatedly applying the dynamics $\mathbf{f}$ for N times. The solution of the optimal control problem is the control sequence that minimizes the *total cost* along a trajectory for a given initial state $\mathbf{x}_0$:

$$\mathbf{U}^*(\mathbf{x}) \equiv \underset{\mathbf{U}}{\operatorname{argmin}} \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) + \ell_f(\mathbf{x}_N).$$

B. Trajectory Optimizer

Our optimization algorithm relies on the principle of dynamic programming: the *Value* at time i indicates the minimum cost one can hope to obtain for the remaining $N - i$ steps when acting optimally. The final value is equal to the final cost: $V(\mathbf{x}, N) = \ell_N(\mathbf{x})$, and the rest is computed recursively going backwards in time along the trajectory:

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i+1)].$$

At every iteration, we compute a second-order approximation of the derivatives of V WRT $\mathbf{x}_i$, and derive from it a search direction $\Delta\mathbf{U}$. We then perform a line search, rolling out multiple trajectories with control sequences $\mathbf{U}_0 + \alpha\Delta\mathbf{U}$ for different values of α, and select the best one.

Our algorithm is the control analog of the Gauss-Newton method for nonlinear least-squares optimization: we use a first-order expansion of the dynamics and second-order expansion of the cost. For linear dynamics and quadratic cost we can compute the global optimum in a single iteration. However, the cost functions we use are not quadratic in state space (sec. VII), and the hand dynamics is highly-nonlinear, mostly due to the effects of contact making and breaking as well as due to the synergy-based actuation model. Therefore, converging to the optimal control $\mathbf{U}^*$ requires multiple iterations of local approximation. However, the system is in motion while the optimization is being computed. Therefore we find it more computationally advantageous to update x_0 after every iteration than to find the optimal control sequence for a stale initial state.

IV. ADROIT: Manipulation Platform

ADROIT is a reconfigurable manipulation platform being developed for exploring and addressing challenges in dynamic and dexterous manipulation. The physics model used in our simulations is based on the ADROIT platform. We started our search for an ideal platform from a pneumatic robot ShadowHand [1], well known for its dexterity and human-like morphology. We soon attributed its speed and compliance bottlenecks to its actuation system consisting of McKibben muscles and low flow-rate binary valves.

In [6] we developed a general purpose universal pneumatic actuation for tendons driven systems. Our actuation system allows us to move the ShadowHand skeleton faster than a human hand (70 msec limit-to-limit movement) with 30 msec overall reflex latency. The system is almost frictions less and so compliant that only 6 grams of external force at the fingertip displaces it when the system is powered. This combination of speed, force and compliance is a prerequisite for dexterous manipulation, yet it has never before been achieved with a tendon-driven system, let alone a system with 24 dof and 40 tendons.

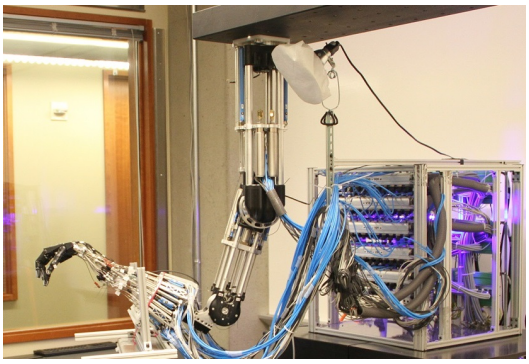

Fig. 1. The ADROIT platform (ceiling mount)

ADROIT (figure 1) is a reconfigurable platform with 24 dof hand mounted on a 4 dof arm. All the joints can be actuated independently by its two exclusive opposing cylinders; with the exception of DIP and PIP joints of the four fingers that are mutually coupled.

The skeleton is mostly dominated by the Shadowhand skeleton with modifications to accommodate our pneumatic actuation system. Other major modifications include, the reinforcement of base joint to support upside down mounting. A low level driver written in 'C' talks to the system. The system is capable of sampling all the tendons length and pressure sensors at 9000Hz. A PIC microprocessor embedded in the palm samples all the joint angle and reports data at 500Hz via a CAN channel. The pneumatic valves can be commanded at 125Hz.

The entire platform is in a stage of constant evolution with the iterative improvements between (a) the hardware requirements, as demanded by the controller while synthesising dynamic manipulation behaviors in simulation and (b) manipulation capabilities, given the development in our control strategies and hardware limitations.

V. Modelling

In order to evaluate the hardware and test our control strategies, the ADROIT simulator (figure 2) is developed using the Mujoco Physics engine. Mujoco [25] is a new physics engine that works with generalised co-ordinates and supports a number of unique features including tendons

actuation. Let $\mathbf{q}$ denote the vector of joint angles. Mujoco represents tendons $L(q)$ as path via routing points (i.e. sites), such that it does not penetrate any geometric wrapping objects (sphere and cylinders). At each time step Mujoco automatically computes the tendon's moment arms $\frac{\partial L(q)}{\partial q}$ which links tendon velocities $\dot{L}$ to the joint velocities $\dot{q}$ and joint torques τ to the scalar tension f applied on the tendon by the corresponding linear actuator.

The upper bound on tension f is determined by the type of actuator connected to the tendons. The lower bound f_{slack} needs to be carefully maintained to avoid tendon slack. While the upper bound is strict, lower bound depends on the physical properties of the actuator (like friction, damping, stiction of the actuator and corresponding joints) and the nature of task being performed.

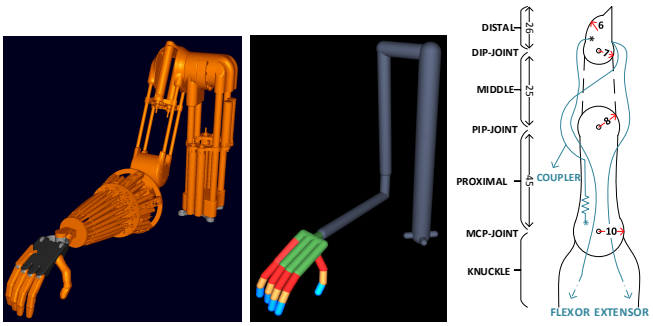

Fig. 2. ADROIT Simulator visualizations (table mount) ***Left:*** A full CAD model of our 28-DOF platform. ***Center:*** An equivalent simplified model using capsules to simplify contact detection. The kinematic tree is the same in both models. ***Right:*** Tendon schematics representing DIP-PIP joint coupling.

Mujoco's tendon length constraints are exploited to model PIP and DIP joint coupling. These joints are coupled using a tendon network that constraints the DIP flexion to be greater that PIP flexion. Figure 2 illustrates how this coupling is modelled in Mujoco using a soft tendon length constrain.

The simulator supports two different actuator mechanisms. First, a linear actuator that directly acts on a tendon to produce tension. Second, third order pneumatic actuator that uses pressure dynamics to produce tendon tension. [26] presents the modelling results of pressure dynamics of our muscle assembly (pneumatic cylinder, pipeline and valves). We present a generic parametric model of pressure dynamics that allow us to predict pressure upto 5 seconds in the future (given the current state and future voltage trajectories). The linear actuator provides a nice abstraction secluding pneumatics from the rest of the robot. Since our pressure dynamics are 3rd order, secluding pneumatics removes pressure from the list of state variables, thus reducing the dimensionality of the state space. This reduction is extremely helpful while exploring hardware capabilities independent of pneumatic actuation.

The dynamic manipulation results and the typing results mentioned in the paper were generated using linear actuators.

VI. Behavior synthesis

Our approach for synthesising dynamic hand manipulation behaviors is to use MPC, also known as online trajectory-optimization or receding-horizon control. Each MPC iteration begins with updating the state of the system using the latest estimates from the estimator. An estimator processes readings from all the sensors and maintains a coherent estimate of the state at all times. A single iteration of a trajectory-optimization algorithm is applied starting from this state, and the initial part of the resulting policy is applied to the system until the process can be repeated and the policy updated again. For MPC to succeed, large updates to the policy must be made in each step so that the optimizer can "keep-up" with the changing state. This means that the regularization parameter μ, which slows the optimizer should be small and carefully chosen.

A. State

ADROIT platform has 28 dof and is actuated using 48 pneumatics cylinders. Depending on wether linear or 3rd order pneumatic actuators are in use, our state space is either 56 (q + $\dot{q}$ for 28 joints) or 104 (q + $\dot{q}$ for 28 joints + p for 48 tendons) dimensional. Manipulation example adds another 12 dimensions for the object state and typing on keyboard adds extra 24 dimensions for the keyboard state. While Adroit is equipped with effective sensing capabilities (joint, touch, pressure and tendon length sensors) which makes its state observable, additional sensing options are required for sensing the bottle and the keyboard. We use the PhaseSpace 3D tracking system.

B. Trajectory optimization

We have demonstrated earlier [7] [8] that iLQG can be used under MPC setting to generate full body movements in real time, while the very same approach failed in producing dynamic hand manipulation behaviors.

After careful investigation we attributed its inability to the nature of optimization space to be navigated and to the difference in the morphology of hardware and actuation mechanism other than direct torque control. Unlike full body movements, large number of dof exists in a very compact workspace in hand manipulation, resulting in a compact high dimensional optimization landscape. High dof to workspace volume ratio results in large number of contacts that make and break too often embedding discontinuities in the already compact high dimensional space. High dexterity allows multiple solution to co-exists inviting optimizers to be stuck in local minima.

In following section we will address these challenges one by one and discuss ways to mitigate them.

C. Contact pruning

In MuJoCo, the most intensive part of computing a single step is the handling of contacts. Unlike full body movements where the number of active contacts are small with low variance, manipulation with high dof manipulator involves large number of active contacts. Self contacts in case of full body movements are rarely active and few when active. For

dexterous hands like ours, dof to workspace volume ratio is very high which results in most of the contacts being active all the time. Number of contacts with the environment in the former case is mostly small (feet-ground contacts). However in the later case, object under manipulation interacts with almost all the links of the hand resulting in large number of active contacts. Moreover these contacts make and break too often producing huge dynamic non-linearities in the search space. Computation time is severely affected as number of contacts (inter-finger and finger-object) increases. High variability in the number of contacts introduces variability in computation timings and hence policy lags.

For real-time behaviours synthesis, mesh collisions were never an option. Even simple geometric collision models using capsule resulted in too many active contacts (40 on average) for real time behaviours. To speed up the computations, contact space was very carefully populated considering kinematic constraints and joint couplings. Furthermore, nature of these contacts were carefully picked. Mujoco supports 3 types of contacts: 1-D contacts, 3-D contacts, 6-D contacts. 1-D contacts are only capable of producing normal forces. All inter-finger contacts are 1-D contacts. 3-D contacts are capable of producing normal and surface friction. In addition to normal and surface friction, 6D contacts produces rolling and torsional friction. All object finger contacts are 3D contacts, except finger tip-object contacts which are 6D.

It should be noted that we never made compromises while pruning the contact space. If violations were ever found, we immediately add the respective contacts back.

D. Passive springs and armature inertia

Low-weight finger segments, strong and low friction actuators provide ADROIT its unique combination of speed, strength and compliance [6]. Such a combination is desirable for any robot hardware but not-so-desirable for numerical optimizers and naive controllers. Numerical optimizers enjoy accelerating the light weight segments producing behaviours that are either unsafe or look unnatural. Weak joint springs were added around the resting configuration of the hand (as shown in Figure 2) in the model that was available to the optimizers for planning. These springs acted as a passive attractors towards the resting configuration of the hand and prevented optimizers from choosing unnatural behaviours. Behavioural artifacts resulting from high acceleration were mitigated by adding armature inertia to the joints proportional to the link masses. It not only removed the artifacts but also improved convergence thereby improving the quality of the solution.

E. Synergy spaces

Animal life form exhibits complex and diverse set of behaviours. These behaviours are produced by numerous bio-mechanical muscles that transmit force using skeletal tendons. Unlike the rest of the body, hands exhibit complicated tendon network that span across multiple joints that introduce substantial coupling between joints. For full body movements, its possible to have individual control over each joints. Kinematics constraints and a regulariser around default body posture have long been exploited to synthesize movements close to animal life forms. However, these tricks were not enough for synthesis of human like hand movements. Optimization rarely produces desired behaviors and if it does, it produces a solution that is too rich to be exhibited by human hands.

To make the optimization space more tractable for numerical optimizers, we introduced five hand synergy dimension [23] on top of individual control dimensions and made them slightly cheaper for the optimizer to choose them. First four synergies individually influence the curl of the four fingers and the fifth synergy influences the spread of the four fingers. Addition of synergy dimensions increased the space of control dimensions by five but made the problem more tractable for the optimizer, without compromising the control over individual joints (hence dexterity) of the hand. Synergies help the optimizer to quickly find a pre-grasping pose, after which individual joints dominate to produce expressive behaviours.

This is for the first time that iLQG has been demonstrated to be amenable in synergy spaces. Traditionally synergy spaces have always been exploited as a mechanism for dimensionality reduction to make a high dimensional problem trackable. However, here we are adding synergy dimensions without removing existing dimensions, thus expanding the dimensionality of an already high dimensional problem. This is a little counterintuitive. We found that the optimizer exploits synergy spaces to quickly navigate through the space full of nonlinearities and discontinuities to localize itself in the correct neighbourhood, where dimensions other than synergies (i.e. individual joint actuation) dominate to produce expressive behaviours.

VII. Cost terms

Synthesis of behaviours involves specification of high level cost function that encodes the task objectives. All behaviours presented in the paper were generated using very simple high level cost functions. These cost function are composed of few simple terms with intuitive meaning associated with each term that talks about the task. Note that no cost terms outlining the movement of the hand (or grasp metric) are provided. We focused our efforts on generic behaviors like dynamic manipulation and object relocation which constitutes significant portion of our daily activities. Once the framework was in place it was easily extendible for specific tasks like typing. In addition to the regular control penalization, which was common for all behaviors, the following other cost terms were used:

Dynamic manipulation:

Dynamic manipulation behaviours include dynamic catching of a falling objects, grasping and stabilizing of unstable objects. Cost terms penalizes

- Control synergies. These bear slightly lower penalty than the rest of the independent actuators.
- Velocity of the object being manipulated.
- Distance of the object to its desired goal configuration (position and orientation)

- The last cost term, with a small coefficient penalizes the distance of the object from the center of hand's grasp envelop.

Ideally, if the MPC horizon is long enough, we will not need the last cost term. In practice, real-time constraints restricts us from using very long horizon. In absence of this term, if the object is far away, the optimizer will not be able to find an improvement within the given time horizon. In such cases, this term acts as hint term for the hand to make an initial approach towards the object. We used a static point equidistant from - the palm base, finger's (Index, middle, ring and little) middle segments and thumb's distal segment as the center of the grasp envelop. Once the initial approach has been achieved, the first three terms dominate. Note that we used no grasp-specific costs promoting force closure or other supposedly desirable grasp properties. All the behaviors seen in the movie [goo.gl/WPTjcS]emerged from these simple cost terms.

Object relocation:

All costs for relocation were same as for dynamic manipulation except object desired configuration cost. Desired configuration was exposed as a parameter to the user. Users were allowed to change this cost by dynamically changing desired configuration in middle of the simulation, resulting in interactive grasping and relocation.

Object relocation starts as a dynamic stabilization sequence, but with a pre-specified intermediate goal configuration, called the 'hold configuration'. Any relaxed position of the hand where it can stably hold the object away from all the obstacles can be used as the hold configuration. The hand waits at the hold configuration for the user to specify the final desired relocation configuration, after which the final relocation maneuver is attempted. If the desired configuration has been preemptively specified by the user, the hand immediately exits the hold configuration and attempts to position the object at the desired configuration. Once the user is satisfied with the final configuration of the object he/she triggers the release, which switches off the grasp cost resulting in hand gracefully leaving and moving away from the object once its stable.

Typing:

This behavior include interactively typing specific numbers on a number pad. Keys on number pad are spring loaded to make the problem challenging. Key press is not detected until the key is fully pressed. Typing results exploit keyboard heat maps to pre-allocate key-finger pair. One rational behind this assumption is that although every individual has his/her own typing preferences, key-finger pairs are almost static. Downside of this approach is that if fingers get stuck in local minima, alternative options will never be explored by the optimizer. Behaviors include cost on

- Desired key pressed
- Desired key approach by the assigned finger's tip.
- Next key hover by the next assigned finger: This cost encouraged the next assigned finger, to make the initial approach towards its key if not being used thus speeding up the typing.
- Auto-correct. This terms kicks-in when accidental typing mistakes are made. A back space is pressed before moving ahead.

Our approach towards typing is event driven. Once a key press is detected, the typing sequences increments and the entire plan is recomputed for the next assignment. In other words, optimizer is not able to plan through the transitions. To allow the optimizer to plan through the transition, an additional cost term 'next key press' is appended. Smooth step-up S_{up} and step-down $S_{dn} = (1 - S_{up})$ functions are used as cost coefficients for 'desired key press' and 'next key press' respectively. When key event is detected the steps are switched which allows the cost to smoothly transition from key hover to key press.

$$S_{up}(t) = \frac{1}{2}\left(\frac{(t-a_1)}{\sqrt{(t-a_1)^2+b_1^2}} - \frac{(t-a_2)}{\sqrt{(t-a_2)^2+b_2^2}}\right)$$

where t is time, a_1 and a_2 are the step locations, and b_1 and b_2 are the step smoothness.

VIII. Results and Discussions

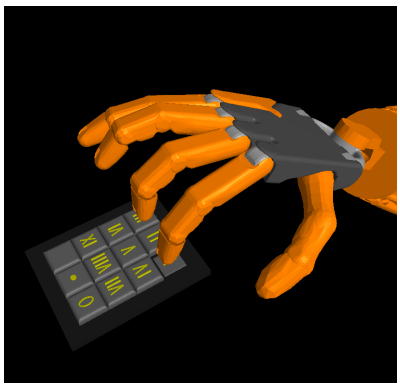

Fig. 4. ADROIT while typing the key-1 on the numeric keypad

We will now present the real time interactive behaviour synthesis results for ADROIT in general tasks like dynamic manipulation, object relocation and specific tasks like typing. All the results mentioned in this paper are generated using a Intel Xeon X5690 @3.47 Ghz processor with 12GB of memory running Windows7. Typical timings for different parts of the computation can be found in table I. Optimization parameters can be found in table II. To better appreciate our results we highly recommend watching the video attachment of our paper. Our latest results can be found at [goo.gl/WPTjcS].

A. Hand manipulation behaviors

Characteristic behaviours in the dynamic hand manipulation sequence involve catching falling objects, grasping objects from different configurations and stabilization of unstable objects. Figure 3 shows frames (150ms apart) of two agile recovery maneuvers.

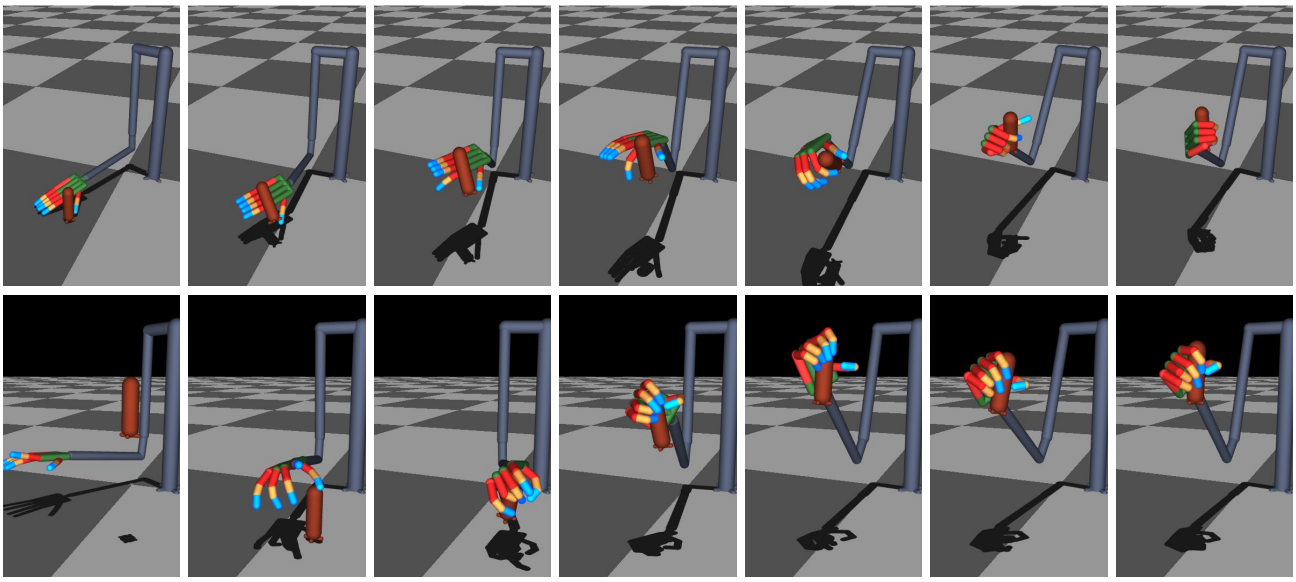

Fig. 3. Two sequences from the accompanying movie. The time between consecutive frames is 150ms. Both sequences show an agile recovery manoeuver. The top sequence begins with a back-handed fumble of the object, which is then caught and brought into the desired position. In the second sequence the initial grip on the object is not robust and the object begins to slip. The controller releases the slipping object and re-grips it in mid-flight.

Use of carefully picked contacts pairs of different nature helped reduce the average number of active contacts from 40 to 20 which provided a major boost in simulation timings. For dynamic grasping, we observed that our optimizers enjoy slightly smoother contacts over the hard ones. Large number of frequently changing contacts introduce discontinuities in the search space. Smoother contacts makes these discontinuities better behaved for optimisers to plan through them. Use of armature inertia to tame large accelerations worked much better than trying to force it using velocity cost. The success rate for dynamic manipulation was around 80%. For unsuccessful cases, either the hand gets stuck in some local minima and never emerges from it or is unable to make a good grasp due to faulty initial approach.

B. Typing on a numeric keypad

Figure 4 shows a snapshot from the typing sequence while ADROIT is pressing the first key. Average number of active contacts for typing stayed relatively low (5 on average, mostly inter finger). Our typing worked fairly well. Though rare, we did observe the fingers getting stuck in local minima for some sequences. Due to preassigned key-finger pairs, optimizers don't even explore alternative approaches to make progress, when stuck. Naive approach to get out of the local minima will be to momentarily pause typing and relax all fingers before moving ahead. We leave key-finger assignment problem as future work. Mistyping due to accidental contact of finger with the edge of the adjacent key is also observed.

TABLE I

MPC TIMINGS (SEC) FOR INTEL XEON X5690 @3.47 GHZ, 12GB MEMORY, WINDOWS7.

Timings	Dynamic Manipulation	Typing
Total	0.059	0.047
Policy lag	0.050	0.038
Rollout	0.006	0.004
Derivatives	0.036	0.025
Cost computations	0.001	0.001
Backpass	0.010	0.010
Line search	0.007	0.006

TABLE II

OPTIMIZATION PARAMETERS

Specifications	Dynamic Manipulation	Typing
Simulation dt (sec)	0.005	0.001
Optimizer dt (sec)	0.005	0.020
Horizon (sec)	0.300	0.640
Mindist(m)	0.010	0.001
Contact softness(m)	0.005	0.001

IX. FUTURE DIRECTIONS

Our motivations and goals are to implement dynamic hand manipulation on our Adroit hardware platform. To the best of our capabilities, utmost measures have being taken to not make any assumptions in our simulation results that will not be valid on the real hardware. We are in process of iteratively adapting our simulation results to the capabilities of the hardware and work on the hardware bottlenecks identified from the simulation results. Strength of the shoulder joint was one of the major hardware bottleneck that was identified and we are currently addressing it.

In future, we intend to carefully investigate application of MPC and other control strategy on ADROIT. Initial testing of our behavior synthesis framework on the actual hardware indicates sensitivity to modelling errors. We are performing careful system identification to obtain a good model of the system. Such a model will never be perfect and we might have to explore options with better tolerance to modeling errors.

ADROIT has the capability to emulate biological muscles.

We want to experiment with muscles models from the biomechanics literate to check if they have something to offer to the robotics community.

With respect to individual behaviors, our approach towards typing is shortsighted as the optimizer only sees and plans for present and the next key in the typing sequence, even if the entire sequence is specified in advance. One option is to incorporate a phase variable (encoding the development made by the trajectory) in the optimization and let the optimizer choose the number of key presses and plan through it. Piano playing is another problem with temporal objective that will require specific attention to key-finger assignment problem.

X. Acknowledgement

This work was supported by the NSF and the NIH.

References

[1] Shadow Robot Company, www.shadowrobot.com.

[2] S. Haidacher, J. Butterfass, M. Fischer, M. Grebenstein, K. Joehl, K. Kunze, M. Nickl, N. Seitz, and G. Hirzinger, "DLR hand ii: hard- and software architecture for information processing," *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 1, pp. 684–689 vol.1, Sept. 2003.

[3] A. Deshpande, Z. Xu, M. Weghe, L. Chang, B. Brown, D. Wilkinson, S. Bidic, and Y. Matsuoka, "Mechanisms of the anatomically correct testbed (act) hand," *IEEE/ASME Trasactions on Mechatronics*, 2011.

[4] C. Joel, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade, "Footstep planning for the honda asimo humanoid," in *IEEE International Conference on Robotics and Automation*, 2005.

[5] "Simulation and control of biped walking robots," http://mediatum.ub.tum.de/doc/997204/997204.pdf.

[6] V. Kumar, Z. Xu, and E. Todorov, "Fast, strong and compliant pneumatic actuation for dexterous tendon-driven hands," in *IEEE International Conference on Robotics and Automation*, 2013.

[7] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Conference on Intelligent Robots and Systems*, 2012, pp. 4906–4913.

[8] T. Erez, K. Lowrey, Y. Tassa, V. Kumar, S. Kolev, and E. Todorov, "An integrated system for real-time model predictive control of humanoid robots," in *International Conference on Humanoid Robots*, 2013.

[9] "Grasp quality measures," http://personalrobotics.ri.cmu.edu/courses/papers/SuarezEtal06.pdf.

[10] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *Robotics & Automation Magazine, IEEE*, vol. 11, no. 4, pp. 110–122, 2004.

[11] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2012.

[12] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 1. IEEE, 2000, pp. 348–353.

[13] A. M. Okamura, N. Smaby, and M. R. Cutkosky, "An overview of dexterous manipulation," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 1. IEEE, 2000, pp. 255–262.

[14] K. Yamane, J. J. Kuffner, and J. K. Hodgins, "Synthesizing animations of human manipulation tasks," in *ACM Transactions on Graphics (TOG)*.

[15] A. Safonova and J. K. Hodgins, "Construction and optimal search of interpolated motion graphs," in *ACM Transactions on Graphics (TOG)*.

[16] Y. Ye and C. K. Liu, "Synthesis of detailed hand manipulations using contact sampling," *ACM Transactions on Graphics (TOG)*.

[17] J. J. Kutch and F. J. Valero-Cuevas, "Challenges and new approaches to proving the existence of muscle synergies of neural origin," *PLoS computational biology*, vol. 8, no. 5, p. e1002434, 2012.

[18] M. C. Tresch and A. Jarc, "The case for and against muscle synergies," *Current opinion in neurobiology*, 2009.

[19] M. Santello, M. Flanders, and J. F. Soechting, "Postural hand synergies for tool use," *The Journal of Neuroscience*, 1998.

[20] M. C. Tresch, P. Saltiel, and E. Bizzi, "The construction of movement by the spinal cord," *Nature neuroscience*, 1999.

[21] A. d'Avella, P. Saltiel, and E. Bizzi, "Combinations of muscle synergies in the construction of a natural motor behavior," *Nature neuroscience*, 2003.

[22] D. B. Lockhart and L. H. Ting, "Optimal sensorimotor transformations for balance," *Nature neuroscience*, 2007.

[23] E. Todorov and Z. Ghahramani, "Analysis of the synergies underlying complex hand manipulation," in *Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE*. IEEE, 2004.

[24] D. H. Jacobson and D. Q. Mayne, *Differential Dynamic Programming*. Elsevier, 1970.

[25] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012.

[26] T. Yuval, T. Wu, J. Movellan, and E. Todorov, "Modeling and identification of pneumatic actuators," in *International Conference on Mechatronics anf Automation (ICMA)*, 2013.