Dexterous TeleManipulation with A Multi-Touch Interface

Yue Peng Toh, Shan Huang, Joy Lin, Maria Bajzek, Garth Zeglin, and Nancy S. Pollard
Carnegie Mellon University

***Abstract*—Robust manipulation with a dexterous robot hand is a grand challenge of robotics. Impressive levels of dexterity can be achieved through teleoperation. However, teleoperation devices such as a glove or force reflecting master-slave system can be expensive and can tie the robot down to a restricted workspace. We observe that inexpensive and widely available multi-touch interfaces can achieve excellent performance for a large range of telemanipulation tasks, making dexterous robot telemanipulation broadly accessible. Our key insight is that dexterous grasping and manipulation interactions frequently focus on precise control of the fingertips in a plane. Following this observation, our novel multi-touch interface focuses on reliable replication of planar fingertip trajectories, making previously difficult actions such as grasping, dragging, reorienting, rolling, and smoothing as intuitive as miming the action on a multi-touch surface. We demonstrate and evaluate these and other interactions using an iPad interface to a Shadow Hand mounted on a Motoman SDA10 robot.**

I. INTRODUCTION

Dexterous manipulation is one of the grand challenges of robotics. Precision, adaptability, and robustness are required for robots to operate successfully within natural or hazardous environments or in workspaces designed for people, such as the home or office. However, current algorithms do not begin to approach human dexterous manipulation capabilities.

The most impressive examples of robotic manipulation have been demonstrated through remote operation, typically using a glove input device [1][2][3][4] or force feedback master-slave system [5][6][7]. However, these interface devices are not without difficulties. Force feedback master-slave systems are expensive and can be highly specialized. Even after decades on the market, glove input devices suffer from a limited number of degrees of freedom and dissatisfaction with calibration, evidenced by continuing publications related to glove/hand calibration techniques [8][9][10][11].

An inexpensive, reliable input device would open up many possibilities. It would allow rapid and precise completion of challenging teleoperation tasks. It would make it possible to rapidly, intuitively, and repeatedly demonstrate tasks, facilitating learning manipulation through demonstration. By making telemanipulation accessible to many users, it would also allow remote and collaborative telemanipulation over the network.

This work was supported by the National Science Foundation awards IIS-1145640 and CNS-0855171.

Y. P. Toh is with the Department of Computer Science, Carnegie Mellon University kentoh86@gmail.com

S. Huang, J. Lin, and Maria Bajzek are with the Department of Computer Science, Carnegie Mellon University [shanhuan|joylin|mbajzek]@andrew.cmu.edu

G. Zeglin and N. S. Pollard are with the Robotics Institute, Carnegie Mellon University [garthz|nsp]@cs.cmu.edu

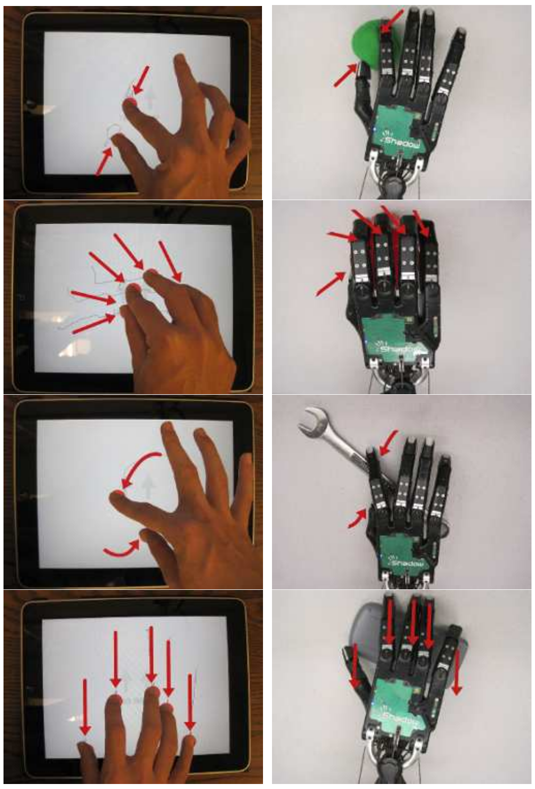

Fig. 1. Multi-touch telemanipulation: From top to bottom, two-fingered pinching, five-fingered power grasp, two-point rotation and sliding

In this paper, we demonstrate an inexpensive, intuitive and light-weight interface for multi-fingered robot telemanipulation using a multi-touch surface. Inexpensive and portable multi-touch surfaces have become ubiquitous in recent years, which makes them cost-effective potential replacements for traditional teleoperation interfaces. Perhaps the most important benefit that a multi-touch interface offers is its directness of interaction in the form of multiple finger touches.

Our key observation is that many natural manipulation actions focus on the fingertips and involve motions of the fingertips within a plane. Our environment is fully of planar surfaces on which we place, pick up, pinch, push, slide, smooth, and otherwise maneuver objects (Figure 1). As such, a planar interface device focused on the fingertips provides a natural means to express these actions. The focus of measurement is on the fingertips themselves, making extensive calibration unnecessary, as relative fingertip motions are captured with precision on a multi-touch surface. In addition to being natural, this form of direct interaction has the advantage of being unobtrusive; the operator can telemanipulate with his or her bare fingertips without having to control an additional master device such as a haptic dataglove.

Our technique of mapping multi-touch information involves treating registered finger touches as target end-effector positions for the robot to achieve. In each time step, the Jacobian pseudoinverse is used to solve for the inverse kinematics of the arm and hand joints. Secondary goals such as a default pose are enforced within the nullspace to encourage natural-looking robot postures.

Evaluation of our interface with naive users suggests that people find the mode of interaction highly intuitive, although training time is necessary to become proficient. In our experiments, telemanipulation with the real robot appeared more intuitive and satisfying than when using a simulator, perhaps because of our powerful sense of 3D space and ability to precisely gauge force and contact in the physical world.

II. RELATED WORK

Teleoperation is the control of a robot by a human operator to perform tasks in a remote environment. Being able to carry out tasks remotely can be valuable, especially when the environment is hazardous, uncertain or inaccessible to humans. As such, teleoperation systems have found a wide range of applications in space, undersea, surgical, and military operations, as well as in virtual reality [12].

Significant research has been focused on telemanipulation of robot extremities, in particular the hand. These systems are often bilateral [7]; the operator manipulates a ”master” manipulator which structurally resembles the ”slave” manipulator to be controlled. The ”slave” hand manipulator in turn reflects haptic feedback in the form of contact forces to the ”master”, allowing the operator to gain a augmented sense of telepresence during the execution of the dexterous task.

The bulkiness associated with traditional master manipulators prompted some researchers to focus on more portable alternatives, such as the dataglove. The dataglove is a glove-like device which records the degree of bending of the operator’s finger joints. Glove data can be combined with haptic feedback to control the robot hand efficiently for telemanipulation tasks [3][2]. Associated strategies for mapping between human and robot hand workspaces have been discussed in [8] and [9]. In other unobtrusive systems, vision has also been employed to track the operator’s hand during dexterous telemanipulation [13]. However, this tracking task is extremely challenging due to the complexity of the hand and limited visibility of fingers in many poses.

Some systems have explored mapping low dimensional commands to complex manipulation trajectories, with simple input devices such as joysticks [14]. These approaches also emphasize transparency as well as sharing control with the robot [15]; the human operator focuses on issuing high-level commands to the robot, and is isolated from the low-level complexities involved in the actual trajectory generation.

In our paper, we develop techniques for telemanipulating a robot hand dexterously and intuitively using a lightweight multi-touch input device. Actions such as tilting and scrolling allow the operator to translate the hand to where it is needed using the same inexpensive device. In contrast with joysticks and other traditional interfaces, multi-touch systems allow simultaneous control of all fingers of the robot hand, allowing a variety of dexterous manipulation actions to be performed through direct user interaction. In contrast to glove based systems, multi-touch devices allow direct control of relative fingertip positions, and hence more direct control of the manipulation task. In contrast with master-slave systems, multi-touch devices are lightweight, portable, and widely accessible.

While multi-touch interfaces for robot control have been explored in recent work [16][17], these systems are typically geared towards high-level teleoperation tasks such as controlling the movement of mobile robots. To the best of our knowledge, no prior system has employed a multi-touch interface for the dexterous control of a multi-fingered robot hand. On the other hand, the use of fingertips for the direct manipulation of virtual objects such as a 3D anatomical model with a multi-touch interface has been discussed extensively [18], [19], [20]. Our work applies the essence of multi-touch interaction to the manipulation of real objects by proxy, via the fingertips of the teleoperated robot hand.

III. THE ASSUMPTION OF PLANARITY

Use of a multi-touch surface for teleoperation creates a bias towards motions of the fingertips within a plane. However, planar grasping and manipulation actions are already commonplace. Much attention in robotics has been applied to planar rotation [21], pushing [22], push grasping [23], moving objects out of the way [24], and sliding / gathering [25]. Planners must consider common constraints such as maintaining an object upright, e.g., to avoid spilling [26], and controlled contact with a planar surface has been recognized as critical for successful grasping of small objects [27].

Research into human grasp preferences has indicated a preference for alignment of the hand to principle axes of an object, which tends to lend itself to horizontal and vertical orientations [28]. In a recent video survey of interactions prior to grasping, it was noted that complex actions such as 6 degree-of-freedom tumbling were the exception, and most observed actions fell into categories of planar rotation, planar sliding, and set arrangement on a horizontal surface [29].

Observations such as these have led to the idea that it is possible and perhaps intuitive to perform common telemanipulation tasks using a multitouch device as described in this paper. The examples in this paper and the accompanying video show the wide range of actions that can be performed using the assumption that active fingertips operate in a single plane of action.

IV. SYSTEM OVERVIEW

An overview of our system is shown in Figure 3. Finger touch positions are fed from a multi-touch device to a server, which converts multi-touch coordinates into target positions for the fingertips of a robot hand. Joint angles for the robot are obtained using an inverse kinematics algorithm (described below), and trajectory targets are sent to the robot arm and hand in real-time.[1] The robot arm is fairly stiff; however, there is some compliance in the fingers, which is helpful in allowing the operator to modulate contact forces.

The robot used in our experiments system consists of a Shadow Robot Hand mounted on the right arm of a 15-DOF Motoman SDA10 robot. The Shadow Hand is controlled using antagonistic pneumatic muscles, and it has a kinematic structure that resembles that of a human right hand with 24 degrees of freedom (Figure 2). In our experience, the workspace of the Shadow Hand is sufficient to allow for natural and intuitive direct control of fingertip manipulation actions and whole hand grasps using the operator's own hand.

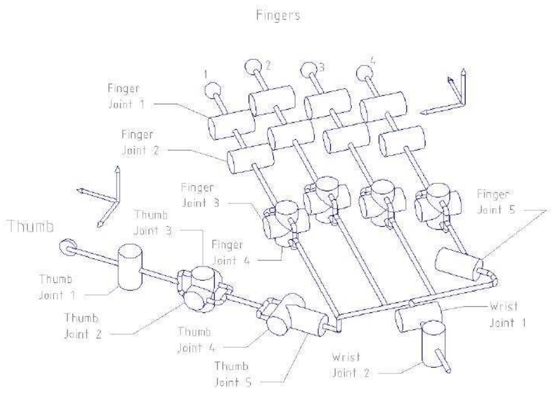

Fig. 2. Shadow Hand kinematics.

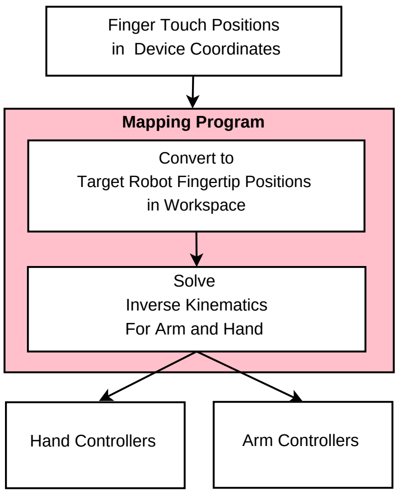
Finger Touch Positions in Device Coordinates
Mapping Program
Convert to Target Robot Fingertip Positions in Workspace
Solve Inverse Kinematics For Arm and Hand
Hand Controllers
Arm Controllers

Fig. 3. System Overview

[1]The expected lag between operator and robot motion is about 0.5 seconds, in part due to conservative velocity and acceleration bounds set for operation by naive users.

V. Multi-touch TeleManipulation

The key principle behind our multi-touch telemanipulation interface is that positions of the fingertips of the human operator's hand are used directly to set target positions for the fingertips of the robot. More specifically, the screen of a multi-touch device is directly mapped to an identical workspace area for the robot using only a rigid body transformation (i.e., no shearing or scaling), so that the mapping from the operator's fingertip movements to those of the robot is intuitive and obvious.

However, in many cases, the workspace of a multi-touch screen may be too small to carry out a desired task, and there is a need to move the fingertips out of the workspace plane.

To achieve these goals, we introduce mechanisms for horizontal and vertical scrolling, described later in this section.

A. Finger Registration

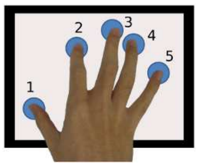

Fig. 4. Finger Registration: To avoid ambiguity, the operator is required to register finger touches sequentially, starting from the thumb.

A multi-touch device can easily track fingertip trajectories over time. However, we must avoid problems with ambiguity between fingers. To achieve this goal, we require the operator to "register" finger contacts by placing the fingers in a specific order (Figure 4). The first touch is expected to be the thumb, followed by the index finger and so on, until the desired number of fingers have been placed. This mapping order is consistent with our observation of the importance of each finger with respect to performing fingertip manipulations. We observe that very precise manipulations are commonly executed using the thumb and the index finger, and grasps that use only a subset of fingers often leave out the pinky and possibly also the ring finger. If the thumb is not needed (e.g., for some sliding tasks), the operator can simply lift it from the multi-touch surface after registration and all other fingers remain registered.

B. Horizontal Control: Edge Scrolling

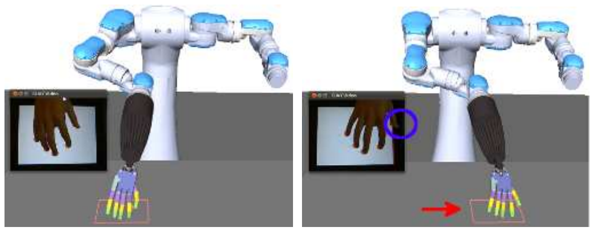

Fig. 5. Edge Scrolling: The operator moves any fingertip, in this case the thumb (circled in blue), against a particular edge of the multi-touch surface to "scroll" the hand in the direction beyond that edge.

By default, all manipulations are executed within a localized control region, a two-dimensional rectangular zone on the workspace representing the area of the physical multi-touch surface. However, the operator may want to carry out a telemanipulation task beyond the boundaries of the default control region on the workspace. An example would be to drag a piece of cloth from one end of the table to another.

To accomplish this goal, we introduce *edge-scrolling*. To move the hand laterally across the workspace plane, the operator moves any finger against an edge of the multi-touch screen. Our system interprets protracted finger touch near one of the four boundaries as a cue that the operator wants to ”scroll” the control region in the direction beyond that particular boundary. The system then proceeds to move the control region with a fixed speed in the designated direction, which causes all active finger targets and thus the hand to be translated in unison (Figure 5). Scrolling halts when the finger is removed from the touch surface or moved away from the border.

We implement this feature by checking for possible finger touches within a tiny margin on each side of the multi-touch screen in each time step. We also actively keep track of the last non-zero velocity for each active finger touch. The rationale for this is to scroll the control region diagonally beyond an edge when a finger approaches the edge at an angle; otherwise we are limited to strictly left, right, up and down moves for each scroll event. If a potential finger at a margin lingers for more than 0.5 seconds, we compute the angle from the last non-zero velocity seen for this finger and scroll the hand at a designated speed in this direction.

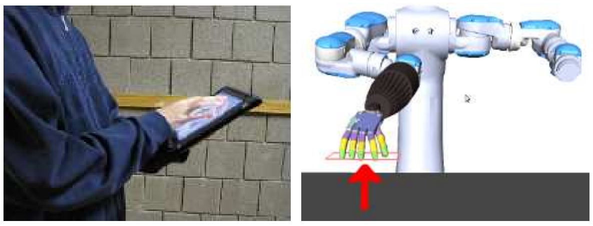

Fig. 6. Device Tilting: The operator tilts the device and holds it beyond a predefined angle to ”scroll” the hand upwards.

C. Vertical Control: Device Tilting

Edge scrolling allows the operator to maneuver over an entire horizontal workspace, such as a table top. However, actions such as cloth folding or pick and place require moving the hand vertically from the table surface. To achieve this goal, we use *device tilting*.

With our vertical scrolling interface, the operator controls the height of the workspace by altering the pitch angle of the multitouch device (Figure 6). The control region is ”scrolled” upwards at a designated constant speed as long as the device is held at a pitch angle greater than a designated angle, θ. The reverse motion occurs when the device is held at a pitch angle of $-\theta$. An angle value lying within the ”deadzone” range of $-\theta$ and θ results in no movement; the operator can hence halt any scrolling by restoring the pitch of the device to a more or less flat configuration. The ”deadzone” is also essential to prevent accidental scrolling.

To implement this feature, we leverage the accelerometer that is built into the iPad to find the pitch of the device at a particular point of time. In particular, we are interested in the x-component of the acceleration vector, which is defined as the axis that runs vertically down the face of the device when the latter is in a landscape configuration. In order to eliminate noise, we first isolate the portion of the acceleration that is attributed to gravity from the portion that is caused by any device motion, by using a basic low-pass filter, as recommended by Apple [30]:

$$\mathbf{g_x} = \mathbf{x_{raw}} * \alpha + (\mathbf{g_{last}} * (1 - \alpha)) \tag{1}$$

where $\mathbf{g_x}$ is that x-component of the gravity vector we are interested in computing, $\mathbf{x_{raw}}$ is the x-component of the raw acceleration vector and $\mathbf{g_{last}}$ is the last filtered x-component of the gravity value. α is the smoothing factor, which specifies the relative proportion of unfiltered acceleration data and the previously filtered value to be used for smoothing. We used an α value of 0.1.

The control region is then moved at a fixed velocity v, depending on the interval that this value lies:

$$v = \begin{cases} v_s, & \text{if } \mathbf{g_x} > |\mathbf{g}| \sin(\theta). \\ -v_s, & \text{if } \mathbf{g_x} < -|\mathbf{g}| \sin(\theta). \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

where v_s is the scroll velocity and $\mathbf{g}$ is the acceleration vector, filtered in the manner of Equation 1. In our implementation, we use a θ value of 15°. In addition, the minimum and maximum heights of the control region are clamped at the height of the workspace surface and a reasonably safe height above the table surface respectively, but the latter can be adjusted easily based on the nature of the teleoperation task.

VI. Hand and Arm Pose

Our multi-touch interface focuses on direct operator control of fingertip motion. Target configurations for the full arm and hand must be computed from these fingertip positions, with the goal of providing smooth and predictable arm and hand control.

Fingertip position targets for active fingers are obtained by mapping multi-touch positions to the robot workspace. We use an initial default mapping that places the hand above the table and roughly in the center of the workspace. This mapping is updated during user control to reflect results of horizontal and vertical scrolling.

We must also assign a location on each finger of the robot as the point of contact with the surface. A position that we found to work well is the middle point of the roundest part of each finger, with the exception of the thumb. For the thumb, we assigned a location on the outer side of the thumb because both the human and the robot thumb tend to be rolled outwards, away from the center of the hand by construct. We find these assignments to be consistent with the typical human fingertip regions that contact the multi-touch surface for many multi-touch gestures.

Fingers that are not active are not a source of concern. Fingertip positions of inactive fingers are not directly controlled and these fingers will move towards a rest pose that places them above the multi-touch surface due to secondary (nullspace) control as described below.

Hand and arm joint angle configurations are solved through a Jacobian pseudoinverse approach, where the robot's arm, palm, and finger links are considered as a kinematic skeleton rooted at the shoulder, with multiple branching finger chains. While an analytical inverse-kinematics solver has obvious benefits such as computational speed which would be valuable for real-time teleoperation, we chose a numerical approach to solve our inverse kinematics problem. A numerical approach was chosen because, in a teleoperation task, it makes sense to have the robot fingertips track their desired targets as well as possible, even when they are out of reach. An analytical framework results in no movement if there is no inverse kinematics solution for a particular set of fingertip targets, while the incremental nature of a numerical method ensures that the end-effectors always seek to reduce the error between the current fingertip positions and the desired targets, regardless of whether these targets are immediately reachable or not.

To solve our given problem, we first compute the active Jacobian of our system. The Jacobian J relates changes in our active joints $\dot{\mathbf{q}}$ to changes in our active fingertip positions $\dot{\mathbf{x}}$:

$$\dot{\mathbf{x}} = J\dot{\mathbf{q}} \tag{3}$$

In our case, $\dot{\mathbf{q}}$ has a dimension equal to the total number of degrees of freedoms in all the joints of the entire active kinematic skeleton, while $\dot{\mathbf{x}}$ has a dimension of 3 x a, where a is the current number of active fingers. Because the set of active fingers can change at any time, the dimensions of J can vary as well.

Because we typically have a large number of joints in our active kinematic skeleton with respect to the degrees of freedom in our goal, our system is highly redundant. This results in the Jacobian being non-square and thus non-invertible. We use the Jacobian pseudoinverse method to minimize the residual norm $||J\dot{\mathbf{q}} - \dot{\mathbf{x}}||$ as well as $||\dot{\mathbf{q}}||$:

$$\dot{\mathbf{q}} = J^{\dagger}\dot{\mathbf{x}} \tag{4}$$

where $J^{\dagger}$ is the pseudoinverse of J. However, the pseudoinverse method returns only one possible solution, which might not always yield a good arm and hand pose given the highly redundant nature of our system. In particular, we want a pose that is reasonably natural and free from collision. A more general solution to (1) allows us to constrain our solutions in terms of well-defined secondary tasks and is given by the following equation:

$$\dot{\mathbf{q}} = J^{\dagger}\dot{\mathbf{x}} + (I - J^{\dagger}J)\mathbf{z} \tag{5}$$

The operator $(I - J^{\dagger}J)$ projects an arbitrary vector $\mathbf{z}$ in joint-velocity space onto the nullspace of J, and by choosing z carefully, one can allow the system to attempt to satisfy secondary constraints without affecting the primary task of attaining the original fingertip goal positions. Many previous authors have exploited this nullspace method to enforce secondary qualities in their solutions such as collision avoidance or joint limit avoidance [31][32][33].

We adopt the same approach for our system. For our case, we have explored a number of secondary goals. One goal that often works well is to introduce a secondary goal that constrains the roll of the robot palm to zero. This constraint is beneficial because the robot poses that our multi-touch interface supports have the palm facing generally downwards, with the finger links extending downwards from the palm onto the workspace plane. If the constraint is not enforced, the palm can roll to one side which may cause the finger links to fail to reach their desired targets. We define this secondary goal as follows:

$$\dot{\mathbf{x_2}} = J_2\dot{\mathbf{q}} \tag{6}$$

where $\dot{\mathbf{x_2}}$ is the velocity of the palm roll and J_2 is the Jacobian defined at the palm origin with respect to all arm joints only. By substituting Equation 6 into Equation 5, then solving for $\mathbf{z}$, and finally plugging z back again into Equation 5, we obtain the following equation:

$$\dot{\mathbf{q}} = J^{\dagger}\dot{\mathbf{x}} + [J_2(I - J^{\dagger}J)]^{\dagger}(\dot{\mathbf{x_2}} - J_2(J^{\dagger}\dot{\mathbf{x}})) \tag{7}$$

This equation is the same as that derived by authors in aforementioned research involving secondary goals [31][32]. $\dot{\mathbf{q}}$ now yields the joint velocities that will bring us to the desired active fingertip goal positions while at the same time ensuring that the palm roll is as upright as possible.

Alternative goals can be satisfied in a similar manner. We have also explored the secondary goal of returning to a hard-coded rest configuration, which was very effective in maintaining natural poses and was used in many of our experiments with naive users.

After obtaining joint velocities for each time step, we check for possible collisions between all robot links as well as with the environment. We do not move any joints in a particular time step if any collision is detected. In the case where joint limits are exceeded, we simply clamp the joints at the appropriate limits. As a last note, the choice of the starting hard pose also affects the quality of subsequently computed poses, since the pseudoinverse approach minimizes change in joint angles from step to step. Therefore, we hand code a desirable starting pose, one that is natural and free from nearby obstacles. This pose is identical to the rest pose that is used as a secondary goal to be approached using nullspace motions.

VII. IMPLEMENTATION DETAILS

We use the Apple iPad as our multi-touch device for teleoperation. We also rely on a TUIO-based open-source finger tracker application for iPad, TuioPad [34], which tracks finger positions on the touch-screen and sends normalized active finger coordinates as UDP packets to our host machine via wireless Ethernet. The host machine runs on an Intel Pentium 4 processor at 3.60 GHz. Our mapping program, which resides on this machine, listens on a suitable UDP port for possible touch events. During each time-frame, it converts the most currently received touch positions in touch screen coordinates to target fingertip world positions in the plane of the control region on the actual workspace, and computes the new inverse kinematics for the robot arm and hand joint positions. We use OpenRAVE [35] to model the simulated robot and the workspace environment. An open loop is run continuously at a frequency of 50Hz on the host which dispatches the latest computed joint positions to the arm and hand joint controllers during each iteration.

VIII. RESULTS AND EVALUATION

Figures 8 - 12 show various snapshots of teleoperated tasks achieved with simple finger gestures using our system, namely sliding, pivoting, and pick-and place tasks, including manipulation of a flexible object (a small towel).

We tested our teleoperation system informally on the robot with 4 naive users. All were given a demonstration of the interface and instructions for finger registration, horizontal scrolling (edge scrolling), and vertical scrolling (device tilting), and then were given some time for freeform practice with the user interface. They were then asked to perform sliding tasks with the following objects: a floppy disk, a screwdriver, a pill bottle, a small spongy ball, and a towel. Start and goal positions were randomized, and the distance between start and goal was 0.2m. Two blocks of trials were performed. In the first block of trials, most testers took more than 50 seconds to complete a sliding trial. Completion times in the second block of trials ranged from approximately 30 seconds to one minute. Completion times for an expert user averaged 19 seconds per sliding task, and performance of the same task by a person (not using our interface) required one to a few seconds.

Although they were not instructed how to manipulate the objects, testers naturally used a combination of strategies, including pinning, gathering, and pinch grasping to slide the objects. Successful pinning entailed landing a number of fingers on the center of the object and then maintaining an appropriate amount of force to slide the object along the table (a “sticky fingers” approach), and was commonly used with flat objects such as the floppy disk. Successful gathering required expanding the fingers to encompass the object in order to inhibit object rotation during sliding. Subjects were most successful in manipulating the screwdriver, pill bottle, and towel, because they were able to get a good grip on these objects. Almost all testers used a pinch gesture with 2 or more fingers to create a pinch grasp of these objects before sliding them to the target location.

We performed a second study with three additional users to compare telemanipulation in a simulated environment vs. telemanipulation of the robot using our interface. Subjects were instructed on the interface as before and then asked to perform four tasks.

- Task A: With blocks placed on table as obstacles, move the hand from left to right, going around the blocks.
- Task B: With one block placed in the air, draw a circle centered at the block and try not to collide with the block (either clockwise or counterclockwise).
- Task C: Three test blocks are placed on the table. Push the test blocks to any point on the table so that they are touching each other.
- Task D: After Task C, separate the blocks by at least a hand.

Figure 7 shows timing results from these tests. Although subjects could perform the first two tasks more quickly in the simulator, the last two tasks, which required actually manipulating the objects, were faster on the real robot.

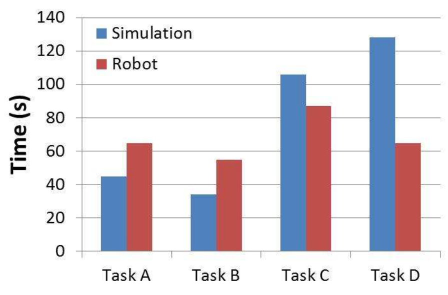

Fig. 7. Average execution times for tasks performed by subjects on the robot vs. the simulated environment.

Subjectively, it appeared that testers experienced greater satisfaction and less frustration when using the robot vs. the simulator. We believe this can be attributed to the limitations of the simulator in representing true real-world physical interactions. We believe that testers using the robot may have experienced a greater sense of control because they were better able to anticipate, evaluate, and predict the effects of their actions in the world.

A. User Feedback

Comments from users indicated that they found the interface intuitive, but they did have to learn to “work around” the latency (approximately 0.5seconds) between their motions and the robot motions. Testers compensated by moving more slowly and by stopping scrolling slightly before the desired goal position. A problem that we observed across some failure cases was the occurrence of accidental touches, which could cause the hand to attempt to achieve incorrect and awkward poses. Testers learned to lift their hands from the multi-touch device and re-register fingers when such events occurred. A more sophisticated approach for identifying and tracking fingers would avoid this difficulty. Our subjects also commented that for a long sequence of tasks, scrolling could be tiring. Multiple subjects mentioned that they wished to be able to more easily lift and replace the hand on the surface (e.g., to replace the current scrolling modality). Overall, however, testers reflected that the controls were easy to learn and natural because the mapping from their finger motions to the corresponding robot finger motions felt direct. Most testers were confident of localizing the hand at a desired location fairly accurately.

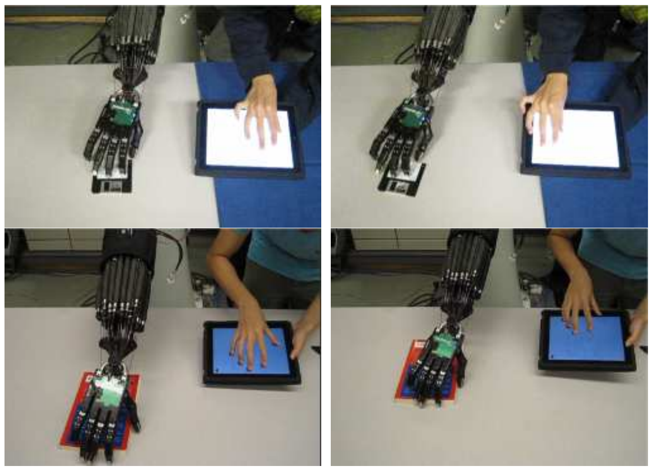
Fig. 8. Sliding: pinning (top row) versus gathering (bottom row)

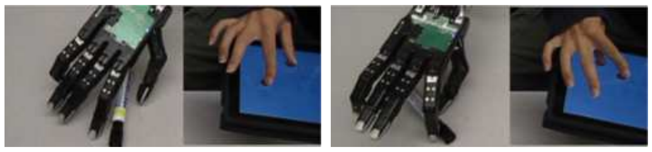
Fig. 9. 2-finger rotation of a pen

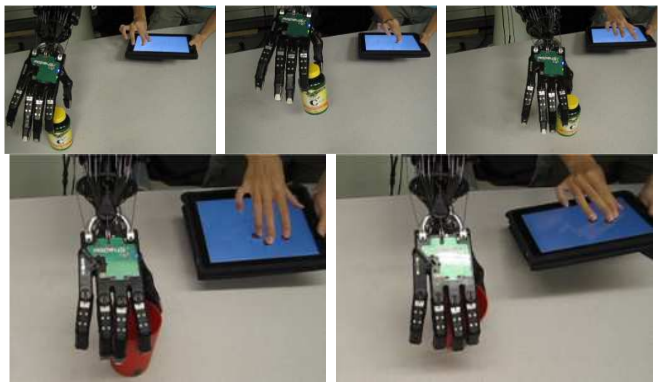
Fig. 10. Pick-and-place: a 2-fingered pinch on a pill bottle (top) and a 4-fingered pinch to execute a power grasp over the opening of a cup (bottom)

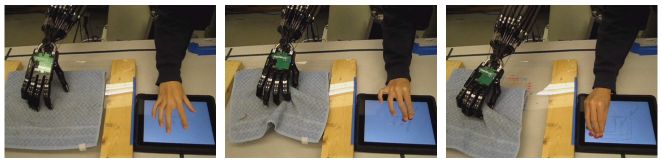
Fig. 11. Pinching and dragging a towel using a pinch-and-slide gesture

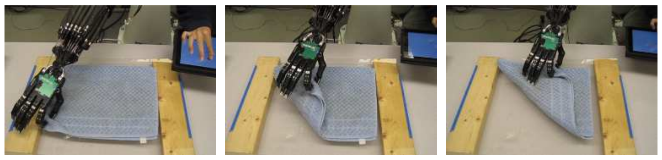
Fig. 12. Controlled folding of a towel using: a pinch, move-up, translate, move-down and release sequence.

IX. DISCUSSION AND FUTURE WORK

We have presented a novel multi-touch interface for the teleoperation of a multi-fingered robot manipulator. Our multi-touch interaction technique offers the operator a physically familiar sense of interaction, allowing him or her to directly telemanipulate objects in a natural and intuitive fashion using multiple fingertips. In addition, our system has the benefit of being portable, unobtrusive and relatively inexpensive as compared to traditional bilateral teleoperation systems. We believe that our interface will be especially valuable for the teleoperation of common tabletop manipulations, such as pinching, sliding, twisting and pick-and-place tasks.

During our evaluation, we found that task execution times were slower than would be desired. Careful attention to removing latency and better management of robot behavior near workspace limits should improve performance. In addition, we believe that there are three directions along which substantial improvements may be made. First, a larger workspace such as a tabletop multi-touch device would make it possible for users to make large, confident sweeping movements of their hand in a manner similar to how we perform such actions when manipulating objects on an actual desktop. We believe our interface is limited by screen space, and task completion will be correspondingly faster as workspace area is increased. Testing this assumption is one avenue of future work.

Second, even with smaller screen devices, the interface for moving the arm could be improved. Many options for horizontal and vertical control of the robot hand have been considered and suggested. One user reported a desire to tip the iPad like a joystick to create horizontal motion of the robot hand. Others have suggested using the accelerometer of the iPad to create vertical velocity in the robot through brief bursts of vertical iPad acceleration. Another suggestion was to use tilt of the iPad to control tilt of the robot fingertip workspace, allowing grasps from an angle or from the side of an object (changing the plane of fingertip action). One area we believe may be most productive is to make it possible for users to naturally lift their entire hand and replace it down on the surface in the way we use a mouse to move large distances in a confined space. To make this modality possible, it is necessary to eliminate the need for sequenced finger registration and to be more clever about maintaining smooth variation in hand and robot pose through discontinuities in the input signal. Further investigation of alternatives is needed.

Third, we believe that the robot must have built-in mechanisms for achieving and maintaining contact with surfaces, in a manner similar to that found in grasping algorithms such as that described in Kazemi et al. [27]. If users have confidence in the robot's ability to avoid large contact forces and maintain appropriate contact with a surface during sliding, they could pay more attention to the motion and less attention to manual and indirect force regulation through vertical scrolling. Force sensing on the multi-touch device would be an obvious extension that would give the user more direct control of fingertip pressures exerted by the robot. Exploring both automatic and manual force regulation is an additional direction of future research.

Finally, our framework can be extended naturally to bi-manual manipulators. With two sets of fingers, interesting planar telemanipulations such as tearing of paper and folding of cloth can be attained. We are also working on extending our multi-touch telemanipulation framework to robot hands that are less human-like, but may be more widely available, such as the three-fingered Barrett Hand.

X. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation awards IIS-1145640 and CNS-0855171 and the Defense Advanced Research Projects Agency through the Autonomous Robotic Manipulation SoftwareTrack (ARM-S). The authors would like to thank: Gurdayal Koonjul for the constructive discussions, Rosen Diankov for the OpenRAVE support, the testers, and the reviewers for their comments.

REFERENCES

[1] S. B. Kang and K. Ikeuchi, “A robot system that observes and replicates grasping tasks,” in *Proc. ICCV*, 1995, pp. 1093–1099.

[2] H. Hu, J. Li, Z. Xie, B. Wang, H. Liu, and G. Hirzinger, “A robot arm/hand teleoperation system with telepresence and shared control,” in *Proc. IEEE/ASME Advanced Intelligent Mechatronics*, 2005, pp. 1312–1317.

[3] M. L. Turner, R. P. Findley, W. B. Griffin, M. R. Cutkosky, and D. H. Gomez, “Development and testing of a telemanipulation system with arm and hand motion,” in *Proc. ASME Symposium on Haptic Interfaces for Virtual Environments and Teleoperators*, vol. DSC-Vol. 69-2, 2000, pp. 1057–1063.

[4] M. Honda, T. Miyoshi, T. Imamura, M. Okabe, F. M. Yazadi, and K. Terashima, “Tele-operation between usa and japan using humanoid robot hand/arm,” in *Proc. HRI*, 2011, pp. 151 –152.

[5] da Vinci Surgery. (2012) da vinci surgical system. [Online]. Available: http://www.davincisurgery.com/davinci-surgery/davinci-surgical-system

[6] T. Koyama, I. Yamano, K. Takemura, and T. Maeno, “Multi-fingered exoskeleton haptic device using passive force feedback for dexterous teleoperation,” in *Proc. IROS*, 2002, pp. 2905 –2910.

[7] P. F. Hokayem and M. W. Spong, “Bilateral teleoperation: An historical survey,” *Automatica*, vol. 42, no. 12, pp. 2035 – 2057, 2006.

[8] R. Rohling and J. Hollerbach, “Optimized fingertip mapping for teleoperation of dextrous robot hands,” in *Proc. ICRA*, 1993, pp. 769 –775 vol.3.

[9] W. B. Griffin, R. P. Findley, M. L. Turner, and M. R. Cutkosky, “Calibration and mapping of a human hand for dexterous telemanipulation,” in *Proc. ASME IMECE Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, 2000, pp. 1145–1152.

[10] H. Hu, X. Gao, J. Li, J. Wang, and H. Liu, “Calibrating human hand for teleoperating the hit/dlr hand,” in *Proc. ICRA*, 2004, pp. 4571 – 4576.

[11] B. Wang and S. Dai, “Dataglove calibration with constructed grasping gesture database,” in *Proc. VECIMS*, 2009, pp. 6 –11.

[12] T. B. Sheridan, *Telerobotics, Automation, and Human Supervisory Control*. Cambridge, MA: The MIT Press, 1992.

[13] J. Kofman, V. Siddharth, W. Xianghai, and L. Timothy, “Teleoperation of a robot manipulator from 3d human hand-arm motion,” in *Proceedings of SPIE*, vol. 5264, 2003, pp. 257–265.

[14] M. Stilman, K. Nishiwaki, and S. Kagami, “Humanoid teleoperation for whole body manipulation,” in *Proc. ICRA*, 2008, pp. 3175 –3180.

[15] W. B. Griffin, W. R. Provancher, and M. R. Cutkosky, “Feedback strategies for telemanipulation with shared control of object handling forces,” *Presence: Teleoperators and Virtual Environments*, vol. 14, no. 6, pp. 720–731, 2005.

[16] M. Micire, M. Desai, J. L. Drury, E. McCann, A. Norton, K. M. Tsui, and H. A. Yanco, “Design and validation of two-handed multi-touch tabletop controllers for robot teleoperation,” in *Proc. Intelligent user interfaces*, 2011, pp. 145–154.

[17] J. Kato, D. Sakamoto, M. Inami, and T. Igarashi, “Multi-touch interface for controlling multiple mobile robots,” in *Proc. CHI Extended Abstracts*, 2009, pp. 3443–3448.

[18] M. Hancock, T. ten Cate, and S. Carpendale, “Sticky tools: full 6dof force-based interaction for multi-touch tables,” in *Proc. ACM Interactive Tabletops and Surfaces*, 2009, pp. 133–140.

[19] J. L. Reisman, P. L. Davidson, and J. Y. Han, “A screen-space formulation for 2d and 3d direct manipulation,” in *Proc. UIST*, 2009, pp. 69–78.

[20] A. D. Wilson, S. Izadi, O. Hilliges, A. Garcia-Mendoza, and D. Kirk, “Bringing physics to the surface,” in *Proc. UIST*, 2008, pp. 67–76.

[21] L. Y. Chang, G. J. Zeglin, and N. S. Pollard, “On preparatory object rotation to adjust handle orientation for grasping,” in *Proc. Humanoids*, 2008.

[22] M. T. Mason, “Mechanics and planning of manipulator pushing operations,” *International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.

[23] M. Dogar and S. Srinivasa, “Push-grasping with dexterous hands: Mechanics and a method,” in *Proc. IROS*, 2010.

[24] M. Stilman and J. Kuffner, “Navigation among movable obstacles: Real-time reasoning in complex environments,” *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 479–503, 2005.

[25] D. Kappler, L. Chang, M. Przybylski, N. Pollard, T. Asfour, and R. Dillmann, “Templates for pre-grasp sliding interactions,” *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 411–423, 2012.

[26] D. Berenson, S. Srinivasa, D. Ferguson, and J. Kuffner, “Manipulation planning on constraint manifolds,” in *Proc. ICRA*, 2009.

[27] M. Kazemi, J.-S. Valois, J. A. Bagnell, and N. S. Pollard, “Robust object grasping using force compliant motion primitives,” in *Proc. RSS*, 2012.

[28] R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith, and Y. Matsuoka, “Human-guided grasp measures improve grasp robustness on physical robot,” in *Proc. ICRA*, 2010.

[29] L. Y. Chang and N. S. Pollard, “Video survey of pre-grasp interactions in natural hand activities,” in *Proc. RSS Workshop on Understanding the Human Hand for Advancing Robotic Manipulation*, 2009.

[30] Apple. (2011, Mar.) Event handling guide for ios. [Online]. Available: http://developer.apple.com/library/iOS/

[31] A. A. Maciejewski and C. A. Klein, “Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments,” *The International Journal of Robotics Research*, vol. 4, no. 3, pp. 109–117, 1985.

[32] P. Baerlocher and R. Boulic, “Task-priority formulations for the kinematic control of highly redundant articulated structures,” in *Proc. IROS*, vol. 1, oct 1998, pp. 323 –329 vol.1.

[33] A. Liegeois, “Automatic supervisory control of the configuration and behavior of multibody mechanisms,” *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 7, no. 12, pp. 868 –871, dec. 1977.

[34] M. Akten and M. Kaltenbrunner. Tuiopad, open source tuio app for ios based on openframeworks. [Online]. Available: https://code.google.com/p/tuiopad/

[35] R. Diankov. (2011, Aug.) Openrave. [Online]. Available: http://openrave.programmingvision.com/