A Sensor-based Dual-Arm Tele-Robotic Manipulation Platform

Daniel Kruse, Richard J. Radke, and John T. Wen
Center for Automation Technologies and Systems
Department of Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute
Troy, NY 12180
{krused2,wenj}@rpi.edu, rjradke@ecse.rpi.edu

Abstract— **This paper presents a novel telerobotic framework for human-directed dual-arm manipulation. Current telerobotic systems typically involve a single robot arm commanded by human through a joystick or a master arm. In contrast, our system involves a dual-arm robot manipulating a held object through human gestures without any mechanical coupling. Our experiment involves an industrial robot consisting of a torso, two seven degree-of-freedom arms, and two three-finger hands. We use the existing industrial robot controller, and only modify the position setpoint in the outer loop. The human interfaces to the robot using a set of gesture vocabulary. During object manipulation, the human gesture is interpreted as the desired configuration of the object. The robot performs autonomous vision-based target identification and alignment, grasp selection and force control, to ensure stable and robust object manipulation, with no demand on human for stable grasping. The heterogeneous components of the system are integrated through Robot Raconteur, a distributed communication and control software system. The system interfaces easily to powerful analysis and visualization tools, facilitating rapid algorithm development and prototyping. We envision that the integrated architecture will serve as the foundation for versatile, robust, and safe human-robot collaboration in increasingly complex sensing and manipulation tasks.**

I. INTRODUCTION

Robots have long been used to extend human reach and capability, in applications ranging from teleoperation [1], as in nuclear waste disposal, to shared control [2], as in surgical robotics, to supervised autonomy [3], as in planetary exploration. These systems typically involve a single robot arm equipped with an end effector dedicated to a specific class of tasks.

Multi-arm telerobots are less common due in part to their mechanical and system complexity. The synchronization and stability issues are much more severe in terms of force reflection, since the human operator needs to regulate both the force of interaction between the jointly held load and the environment as well as the internal force within the load. Even for autonomous operations, tasks for multi-arm robots are often pre-solved using planning algorithms with known geometry information [4]–[6] in which the motion path is computed off-line and then performed in open-loop. Sensor-based motion planning is possible but due to the computational complexity is usually reserved for relatively simple systems [7].

In this paper, we present a novel telerobotic framework for human-directed manipulation. Our approach is sensor-based, allowing flexibility in task specification and execution. We consider robots with multiple kinematically redundant arms equipped with multi-finger hands. Such robots can tackle a much broader range of tasks than a single arm, but at the same time incur increased complexity in terms of potential collision as well as force of interaction in collaborative tasks. Instead of mechanical coupling, our approach is to allow the human to direct robot action through gestures, for a more natural, less constrained, interface.

We implement our approach on a 15-degree-of-freedom (dof) dual-arm industrial robot (Motoman SDA10) and a suite of peripheral sensors and actuators (cameras, Kinect, force/torque sensors, and multi-finger hands) distributed over multiple computers. We integrate these subsystems together in a flexible, seamless manner that allows frequent modification and experimentation as well as easy code maintenance. For the middleware that coordinates the communication over this distributed system, we chose to use *Robot Raconteur* [8], the open source software system developed in our laboratory, over ROS Industrial [9] due to its true distributed implementation (no master node), and ease of use.

We use an example scenario to guide the development. A box is placed arbitrarily in the robot workspace. The robot detects the location of the box, moves both arms around it and positions the fingers for a stable grasp. The two arms then move toward each other to securely grasp the box. The human then interacts with the robot through the gestural user interface to manipulate the box and move it to a desired location without losing grip.

The successful execution of this task scenario requires the solution of a set of technical issues. Their general solutions are complex and involve detailed geometric and other model information of the system and its environment. Instead, we impose a small set of heuristics and structure on the system to allow simple but robust solutions.

- *Object identification and localization.* The robot needs to identify and locate the box, and determine good places for grasping. We draw on planar tagging schemes developed in the artificial reality community, in particular the VTT ALVAR tags [10].
- *Grasping and finger configuration.* The finger configurations of the two three-finger hands need to be chosen to properly and securely hold the box. We use finger-Jacobian heuristics to allow robust, stable, and

repeatable grasping (together with force control below).

- *Force control.* To securely hold the box with the robot arms, the robot must apply a squeeze to result in enough friction to prevent the box from slipping out of grasp. We use the well-known integral force control, with a suitable modification to enhance robustness [11].
- *Redundancy Resolution.* We exploit kinematic redundancy in our system to avoid collision and enhance manipulability and grasp stability. We use a simple damped least square method for redundancy resolution [12] coupled with repellant potential fields for collision avoidance [13].
- *Human Interface.* Our system uses "shared control," i.e., the robot maintains a stable grasp and determines robot joint motion in response to the human's commanded object motion. Instead of the conventional mechanical coupling, we use the Microsoft Kinect sensor to interpret the user's gestures. This is versatile, robust, and more natural for the user as it is not limited by the motion range limitation or mechanical impedance of a mechanical interface.

We envision that the human-directed robotic system as described in this paper, which facilitates integration, prototyping, and human interaction, will serve as the foundation for more complex human-robot collaborative sensing and manipulation in the future.

II. PROBLEM FORMULATION

We use a box manipulation example to guide our research. The system involves a dual-arm robot equipped with cameras, force/torque sensors and grippers, and a box payload as shown in Figure 1. A coordinate frame of a rigid body consists of a reference point, $\mathcal{O}$, and a right-handed orthonormal frame, $\mathcal{E}$. The homogeneous transform between two frames, say **b** relative to **a**, is given by $H_{ab} = \begin{bmatrix} R_{ab} & p_{ab} \\ 0 & 1 \end{bmatrix}$ where R_{ab} is $\mathcal{E}_b$ represented in $\mathcal{E}_a$, and p_{ab} is the vector from $\mathcal{O}_a$ to $\mathcal{O}_b$, represented in $\mathcal{E}_a$. The world frame is denoted by the subscript O. The sensor, camera and gripper frames are denoted by the subscripts $S_{L/R}$, $C_{L/R}$, $G_{L/R}$, respectively, with L and R specifying the left or right arm.

The box localization function combines the camera-based measurement of the box, H_{CT}, and camera location in the world frame, H_{OC}, to determine the box location, H_{OT}. The visual servoing function uses H_{OT} to generate target gripper locations, and to move the robot joints to drive H_{OG} to the target. Since the arms are coupled at the torso, which can rotate, the kinematics of the entire robot must be considered for the alignment of both grippers. Our target load, the box, is relatively large and cannot be held between the fingers of the gripper. The force control function commands the two grippers to apply a squeezing force between them for a secure grasp. In our implementation, the operator provides input using the Kinect; such a non-contact gesture-based interface is particularly attractive since the user is unfettered by mechanical constraints.

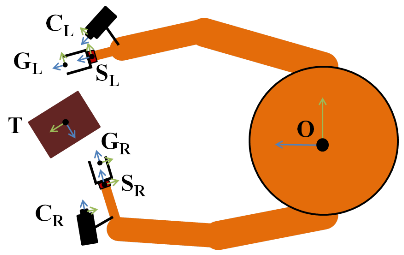


Fig. 1: The different frames of reference for the inertial frame O and task frame T along with the sensors S, grippers G, and cameras C on each arm.

III. SYSTEM DESCRIPTION

Our platform centers around the Yaskawa Motoman SDA10 robot, a dual-arm 15-dof industrial robot with two 7-dof arms and a 1-dof torso. The robot has a built-in joint controller and allows external interface through Motoman's High Speed Controller (HSC). The HSC interface provides joint angle read and incremental commanded joint angle write at a 2ms rate, with a significant delay-time (more than 100ms) from internal trajectory generation.

A Sony XCD-X710CR camera is mounted on the wrist of each arm, angled towards the gripper. Each arm is equipped with a Robotiq three-finger gripper and an ATI Mini45 force/torque transducer (between the robot wrist and the gripper). The robot with the sensors and grippers is shown in Figure 2.

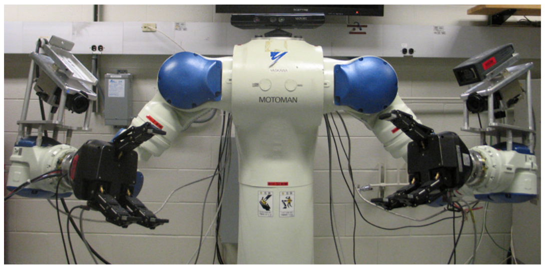


Fig. 2: The industrial robot and its peripherals

The components of the overall system are coordinated using our home-grown distributed control system, called Robot Raconteur [8]. The system architecture is illustrated in Figure 3. Separate Robot Raconteur services are written (in C#) for the Motoman HSC interface, Robotiq gripper, ATI force/torque sensor, cameras, image processing, and Kinect interface, residing on three separate computers linked together in a local area network via Ethernet and TCP/IP. The overall coordination is conducted by a MATLAB script that connects to these services as a client.

IV. CONTROL MODES

The operation of the box manipulation example is represented as a finite state machine shown in Figure 5, containing the three major components: visual servoing, stable grasping, and human-commanded motion. The transition between the states is either through motion or force control to specified locations or thresholds, or by detecting user gestures. After localizing the box, positioning the grippers, and securely

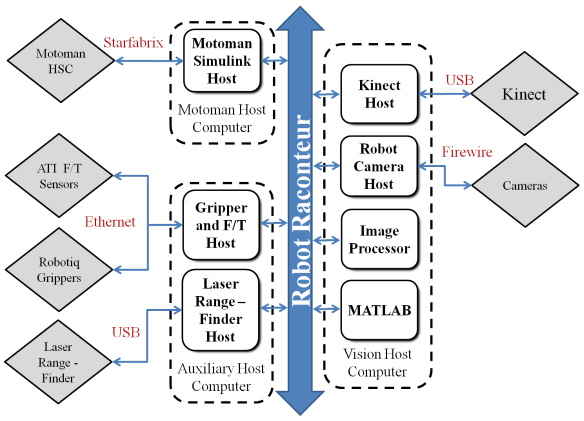


Fig. 3: Overall distributed control system architecture using the RobotRaconteur middleware

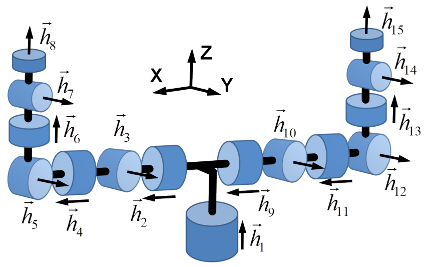


Fig. 4: The 15 axes of revolution for the Motoman robot

grasping the box, the robot moves the box to the specified home position. Then the *start* gesture (left leg kick) initiates the human-directed motion using the Kinect interface. The *pause* gesture (hands together) stops the human interface and waits in place. The *home* gesture (hands far apart) returns the system to the home configuration. The *exit* gesture (left leg kick) terminates the operation. The *release* gesture (hands far apart) releases the box.

A. Task-Space-Based Motion Control

The input to the Motoman HSC is a vector of 15 joint corrections which then passes through a trajectory generator. With tight motor servo loops, we can ignore the dynamics and reasonably approximate the robot response with a first-order-plus-dead-time plant (the dead-time is configuration dependent, varying between 100-150ms). Because the sensor provides task space information, we apply a task space control law. Let J be the 12×15 Jacobian matrix that maps the joint velocity to the spatial velocities (angular and linear velocities) of both grippers:

$$[\omega_{G_L}, v_{G_L}, \omega_{G_R}, v_{G_R}]^\top = J\dot{q} \tag{1}$$

where $(\omega_{G_{L/R}}, v_{G_{L/R}})$ denotes the angular and linear velocities of the left and right grippers, and q is the stacked vector of q_T, q_L, and q_R, the torso, left arm, and right arm joint angles. The coordinate-free form for the ith column of J, J_i, is given by

$$J_i = \begin{cases} \begin{bmatrix} \vec{h}_i & \vec{h}_i \times \vec{p}_{iG_L} & \vec{h}_i & \vec{h}_i \times \vec{p}_{iG_R} \end{bmatrix}^\top & i = 1 \\ \begin{bmatrix} \vec{h}_i & \vec{h}_i \times \vec{p}_{iG_L} & 0 & 0 \end{bmatrix}^\top & i = 2, \dots, 8 \\ \begin{bmatrix} 0 & 0 & \vec{h}_i & \vec{h}_i \times \vec{p}_{iG_R} \end{bmatrix}^\top & i = 9, \dots, 15 \end{cases}$$

where the axes of revolution, $\vec{h}_i$, are shown in Figure 4. For computation, we represent J_i in the base frame (i.e., all vectors in J_i are represented in the base frame). Since we use the industrial controller in the inner loop, we consider $\dot{q}$ as the command input (to be sent to the HSC).

Given the desired position and pose of the gripper, (p^d_{OG}, R^d_{OG}), we set the position and orientation errors as

$$e_p = p_{OG} - p^d_{OG}, \;\; e_R = \sigma(R^d_{OG} R^\top_{OG})$$

where σ is a 3-parameter representation of rotation. We choose σ to be the vector quaternion, but any other representation may be used. Let J_σ be the representation Jacobian, i.e., $\dot{\sigma} = J_\sigma \omega$. Define

$$J_a = \text{diag}(J_{\sigma_L}, I, J_{\sigma_R}, I)J. \tag{2}$$

Then the following proportional damped least-square task space feedback controller [12] would drive the position and orientation error to zero:

$$\dot{q} = -\underbrace{J_a^\top (J_a J_a^\top + \beta I)^{-1}}_{:= J_a^\dagger} Ke \tag{3}$$

where e is the stacked error vector $(e_{R_L}, e_{p_L}, e_{R_R}, e_{p_R})$, $J_a^\dagger$ is the approximate pseudo-inverse with βI added to avoid singularity, and K is the feedback gain matrix. Note that a choice of specific units for linear and angular velocities implicitly means a relative weighting has been applied between linear and rotational motion.

For collision and joint limit avoidance, we use the standard artificial potential field approach [13]. For a task space constraint $p_{OG} \in \mathcal{P}$, where $\mathcal{P}$ is the feasible region for the gripper position, we construct a smooth external penalty function, $\rho_p(p_{OG})$, that is zero in the feasible region and positive in the infeasible region with no local minima. For a joint space constraint $q \in \mathcal{Q}$, where $\mathcal{Q}$ is the feasible region for the joint angles, we similarly construct an external penalty function, $\rho_q(q)$. Let P be the projection of the spatial velocity to linear velocity; then $\dot{p}_{OG} = P J_a \dot{q}$. The task space controller may now be modified as:

$$\dot{q} = -J_a^\dagger \left(Ke + \gamma_p \rho_p P^\top (\nabla \rho_p(p_{OG}))^\top + \gamma_q \rho_q J_a^{\dagger\top} (\nabla \rho_q)^\top \right) \tag{4}$$

where γ_p and γ_q are weightings for the repelling potential functions.

B. Visual Servoing

To facilitate box detection and localization, we mark the grasping point with 2D ALVAR tags [10]. The ALVAR library determines the pose of each tag by mapping the homography between the known position of points in the tag frame to the measured pixels in the image frame. It is straightforward to use the estimated homography to recover the homogeneous transform H_{CG} given an image of a planar tag [14]. However, it is well documented [15] that a reflection ambiguity about the plane perpendicular to the optical axis creates two local minima for the orientation. To resolve this ambiguity, we include a set of "support tags" around the target tag and ascertain the plane's orientation by majority

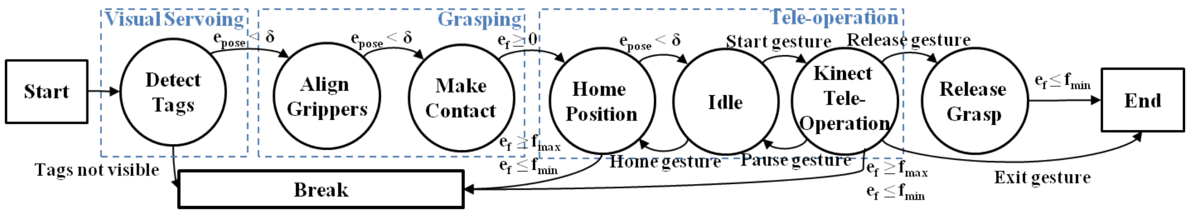


Fig. 5: Finite State Machine demonstrating modes of operation and the state changes causing one to shift to another. Loop break commands involve an inability to find the object or the measured forces detecting a slipped grasp or a crushing force.

poll. Experimentally, this has proven much more reliable than trying to resolve the pose ambiguity of a single tag.

Once we have an estimated position and orientation of the target, we can use the task space controller to drive the grippers to the corresponding targets.

The controller needs a set of world space potential functions based on virtual walls both to prevent the end effector from colliding into the object and to prevent the tag from leaving the camera's field of view. We define each virtual wall by its center point p_{OC} and inward normal n. We define the corresponding potential function by the projection of $p_{OG} - p_{OC}$ onto n. If the projection is negative, then ρ is zero, otherwise, ρ is increasing. The controller (4) may now be applied to drive the grippers to their targets while avoiding task space obstacles and joint limits.

C. Grasping

Before we can make contact, the grippers need to be in a desirable pose. The Robotiq grippers have 3 fingers with underactuated joints, such that during a grasp at the palm, the fingers will conform to the surface of the object. However, this also means that each joint can have a large amount of flexibility based on the pose of the finger.

We found experimentally that in a fully open pose, the contact plane created by the three fingers would not necessarily be perpendicular to the force/torque sensor's normal. This results in an undesirable moment applied to the box. To avoid this undesirable joint motion, we set the gripper in a singular position, i.e. all finger links are collinear. In this singular configuration, the fingers can best resist flexing and give a more predictable and reliable contact plane.

Before moving on to the tele-operation stage, a brief step is included to align the grippers with respect to each other, instead of to the target box. Since we did not guarantee the tag normals to be collinear, this gives an added assurance for a stable, moment-less grasp.

D. Load Motion Command with Squeeze Force Control

For the force control, we apply the generalized damper approach that converts the force error to a velocity. The force error is direction-dependent since a higher squeeze force is more desirable (tighter grip) than lower squeeze force (which could cause a slipped grasp). As recommended in [11] for robust force control of a single arm, we use a direction-dependent gain (damping coefficient):

$$\dot{p}_{OG} = -\beta K_f e_f$$
$$\beta = \begin{cases} 1 & e_f < 0 \text{ (push in)} \\ \beta_f & e_f \geq 0 \text{ (back away)} \end{cases} \tag{5}$$

where K_f is a diagonal matrix containing the reciprocals of damping coefficients (equivalently, integral force feedback gains, set to 0 if no force control is desired) and β_f is a constant between 0 and 1 (we chose 0.3).

The force error is projected along the vector connecting the center of the arm force/torque sensors to ensure that the motion intended to apply a squeeze force does not inadvertently apply an external moment on the box. A dead zone is also applied to the force error to prevent small errors from winding up the controller and generating unwanted oscillations or instability.

The damper equation (5) modifies the position set point, resulting in the following hybrid position/force control law:

$$\dot{q} = -J_a^\dagger \Big(Ke + \beta \int K_f e_f + \gamma_p \rho_p P^\top (\nabla \rho_p(p_{0G}))^\top + \gamma_q \rho_q {J_a^\dagger}^\top (\nabla \rho_q)^\top \Big) \tag{6}$$

The task error is now specified in terms of the box configuration:

$$e = G^\top \begin{bmatrix} -J_\sigma R^d_{OT} R^\top_{OT} \sigma \left(R^d_{OT} R^\top_{OT} \right) \\ p_{OT} - p^d_{OT} \end{bmatrix}_O . \tag{7}$$

where G is the grasp matrix mapping the spatial force on the box to the contact spatial force, and J_σ is the Jacobian mapping $\sigma \left(R^d_{OT} R^\top_{OT} \right)$ to ω_T. The position error is saturated to a maximum value to prevent excessive initial jerk and possible instability.

V. RESULTS

The control systems are implemented in MATLAB, running on an Intel Core i7 running the Windows 7 operating system. This system is not real-time, but the performance is adequate with sampling time at ~7Hz during visual servoing (due to the demand of image processing) and ~30 Hz during external command. The sampling times for these two modes of operations are shown in Figure 6.

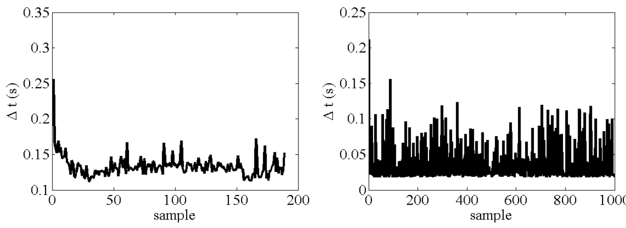


Fig. 6: (a) Visual servoing and (b) teleoperation loop times.

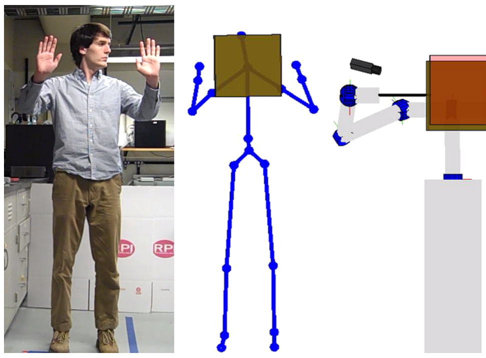
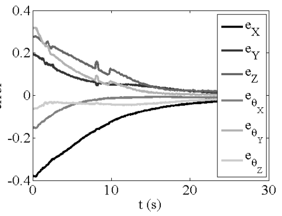
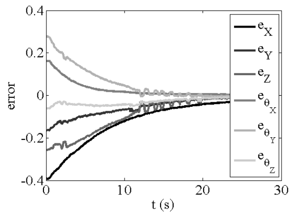


Fig. 7: A single frame from the Kinect-based external command loop. The user's pose is interpreted as a "skeleton." A virtual box is calculated between the skeleton's hands and this desired box pose is used to drive the robot.



(a) Left Arm



(b) Right Arm

Fig. 8: Convergence of the visual servoing algorithm.

During the load motion command mode, we use a Microsoft Kinect not mounted on the robot as the human control interface to provide the user with an intuitive command over the desired box pose. The desired box position is calculated as the center of the ray connecting the two detected hand positions and the orientation is aligned such that the grasped planes are orthogonal to that ray. This controller could potentially be used for long distance teleoperation. There are no stability issues, since there is no force reflection. Figure 7 illustrates the operation and the relationship between the box configuration and the operator hand pose. While this command method is more natural to the user, it has a disadvantage of having only 5 degrees of freedom for control since the Kinect cannot detect the operator's wrist orientations.

A. *Visual Servoing*

Our visual servoing algorithm is convergent as long as the cameras are able to see the tags. For our experiment, we place the arms into a configuration such that both the position and rotation displacement from desired pose is significant. Figure 8 shows how the control system is convergent even starting from a significant orientation error. There are minor jumps around $t = 12s$ on the right arm where the image-based potential functions adjust the camera's position to have the tag further from the image boundary.

B. *External Load Motion Command under Squeeze Control*

Before human interaction, we first tested the external load motion command on several test signals. This mode is very sensitive to time delay in the system, and since our plant has a significant delay ($\sim$100ms) between joint command and joint actuation, we needed to be careful during our gain tuning. There is a narrow window between our gains not impacting our controller from too low control signals and instability. Our recommendation is to first tune the force feedback gain before attempting to tune the motion gain.

We first considered a simple sinusoid with amplitude 0.1m and frequency 0.2Hz along a single direction. Figures 9a-9c show how the robot drove the box to follow the reference signal, with a phase delay of approximately 0.5s and an error of approximately 0.1 between peaks. The grasp oscillated about the target 10N squeeze grasp, but as designed by (5), the overshoot never resulted in lost grasp.

For a more challenging case, we considered a reference signal simultaneously translating along one axis and rotating about another. Figures 9d-9f present the results of this case. As before, the grasp has significant overshoot only in the positive squeeze direction, and has a similar phase delay as in the 1-D case of approximately 0.5s. However, the translational error and rotational error at their peaks differ by a significant margin, despite both receiving a 0.1 amplitude reference signal. We found experimentally that the rotational error vector needed approximately double the gain of its translational counterpart in order to get similar performance in reference tracking.

The results from running the control algorithm with a human operator and a Microsoft Kinect are illustrated in Figures 9g-9i (please also refer to the supplementary video). The error reduces linearly during points of large displacement, due to our clamping of the error magnitude. By saturating the position error to a maximum value, the error velocity becomes a constant equal to $K\bar{e}$. We found it was very important to clamp the position error magnitude to a maximum value in order to prevent excessive jerk and force control instability; however, this causes a linear convergence rate during large displacements. We believe this to be a fair tradeoff to maintain stability over performance, since the grasp will often be lost or too excessive during the initial motion jerk otherwise.

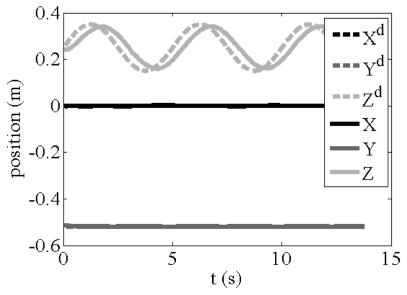
VI. FUTURE WORK

One limitation of the current process is the lack of contact detection by the gripper fingertips. We plan to detect contacts based on the measured the finger joint torques. There is also a significant and dynamic delay between joint command and joint action due to internal trajectory generation by the HSC. Future work will include methods to model this delay and compensate for it in order to improve force-feedback control.

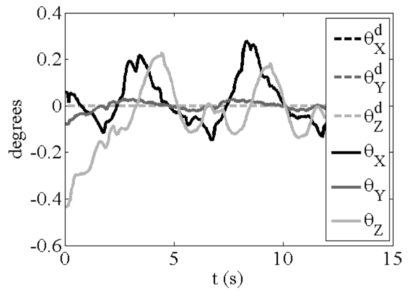
Longer-term goals for this project include manipulation of awkwardly loaded objects and flexible materials, extending to a teach-by-demonstration robotic learning system, and long-distance teleoperation of robotic platforms.

ACKNOWLEDGMENTS

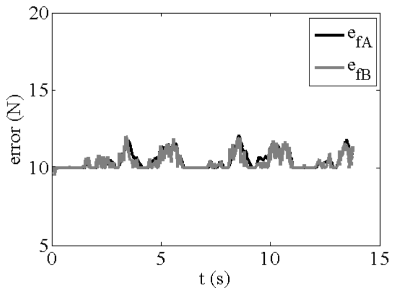
This work was supported in part by the Center for Automation Technologies and Systems (CATS) under a block grant from the New York State Empire State Development Division of Science, Technology and Innovation (NYSTAR)



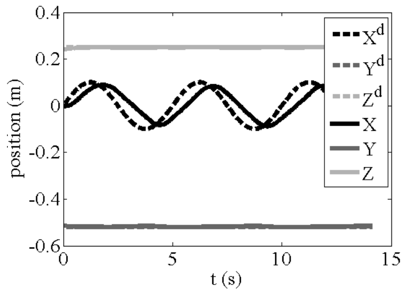
(a) Position



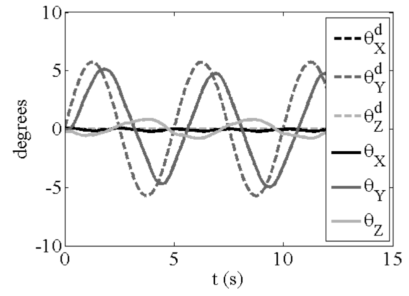
(b) Orientation



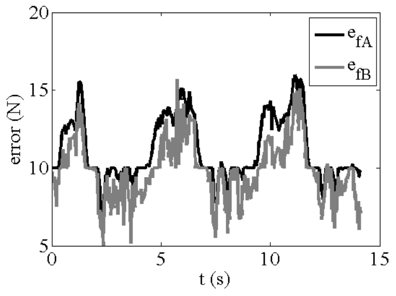
(c) Squeeze Force



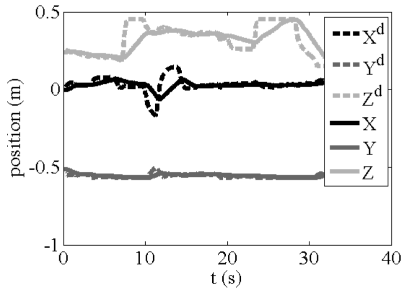
(d) Position



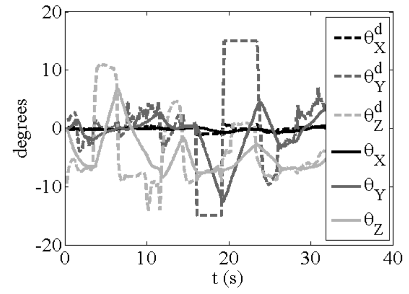
(e) Orientation



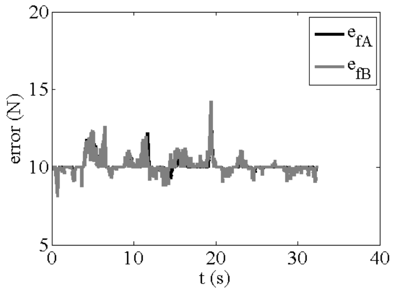
(f) Squeeze Force



(g) Position



(h) Orientation



(i) Squeeze Force Error

Fig. 9: Experimental results from different external command cases (see text). (a)-(c) Translation along Z. (d)-(f) Translation along X with rotation about Y. (g)-(i) Human command via Kinect. The target squeeze force for all cases was 10N.

and in part by the National Science Foundation (NSF) Smart Lighting Engineering Research Center (EEC-0812056).

References

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

[2] W. S. Kim, B. Hannaford, and A. Fejczy, "Force-reflection and shared compliant control in operating telemanipulators with time delay," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 2, pp. 176–185, 1992.

[3] G. Cheng and A. Zelinsky, "Supervised autonomy: A framework for human-robot systems development," *Autonomous Robots*, vol. 10, no. 3, pp. 251–266, 2001.

[4] F. Basile, F. Caccavale, P. Chiacchio, J. Coppola, and C. Curatella, "Task-oriented motion planning for multi-arm robotic systems," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 5, pp. 569 – 582, 2012.

[5] K. Hauser and V. Ng-Thow-Hing, "Randomized multi-modal motion planning for a humanoid robot manipulation task," *The International Journal of Robotics Research*, vol. 30, pp. 678–698, Dec. 2010.

[6] J.-S. You, D.-H. Kim, S.-J. Lim, S.-P. Kang, J. Y. Lee, and C.-S. Han, "Development of manipulation planning algorithm for a dual-arm robot assembly task," *IEEE International Conference on Automation Science and Engineering*, 2012.

[7] S. Hayati, T. S. Lee, K. S. Tso, P. G. Backes, and J. Lloyd, "A unified teleoperated-autonomous dual-arm robotic system," *IEEE Control Systems*, vol. 11, no. 2, pp. 3–8, 1991.

[8] J. Wason and J. Wen, "Robot Raconteur: A communication architecture and library for robotic and automation systems," in *IEEE Conference on Automation Science and Engineering*, (Trieste, Italy), Aug. 2011.

[9] S. Edwards, "ROS for industrial applications." `www.rosindustrial.org`, 2012. Accessed: 9/12/2012.

[10] VTT Technical Research Centre of Finland, "ALVAR-a library for virtual and augmented reality." `http://virtual.vtt.fi/virtual/proj2/multimedia/alvar/`, 2013. Accessed: 02/20/2013.

[11] L. Wilfinger, J. Wen, and S. Murphy, "Integral force control with robustness enhancement," *IEEE Control System Magazine*, vol. 14, pp. 31–40, Feb. 1994.

[12] C. W. Wampler, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 16, no. 1, pp. 93–101, 1986.

[13] O. Khatib, "Real time obstacle avoidance for manipulators and mobile robots," in *IEEE Robotics and Automation Conference*, 1985.

[14] Z. Zhang, "A flexible new technique for camera calibration," vol. 22, pp. 1330–1334, Nov. 2000.

[15] G. Schweighofer and A. Pinz, "Robust pose estimation from a planar target," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 2024–2030, Dec. 2006.