Experiments with a Dual-Armed, Cooperative, Flexible-Drivetrain Robot System

Lawrence E. Pfeffer and Robert H. Cannon, Jr.

Aerospace Robotics Laboratory
Department of Aeronautics and Astronautics, Stanford University
Stanford, California 94305

Abstract

Many tasks are difficult or impossible for "one-armed" robots, for example, assembly with long parts. Multi-armed robots can do such tasks well, but only if both robot and controls are systematically designed for cooperation. We have built such a two-armed robot. We present its design and its experimental performance. The manipulators have added drivetrain flexibility to aid studying its effects on robot cooperation. The control system is a four level hierarchy: joint, arm, object, and task levels. The joint level handles flexibility via joint-torque control. The arm level uses nonlinear end-point feedback to control tip forces and positions. The object level manages the object via object impedance control. The task level directs multi-step tasks autonomously. Experiments are shown for each level, culminating with a "two-handed" insertion of a long part into a deep hole.

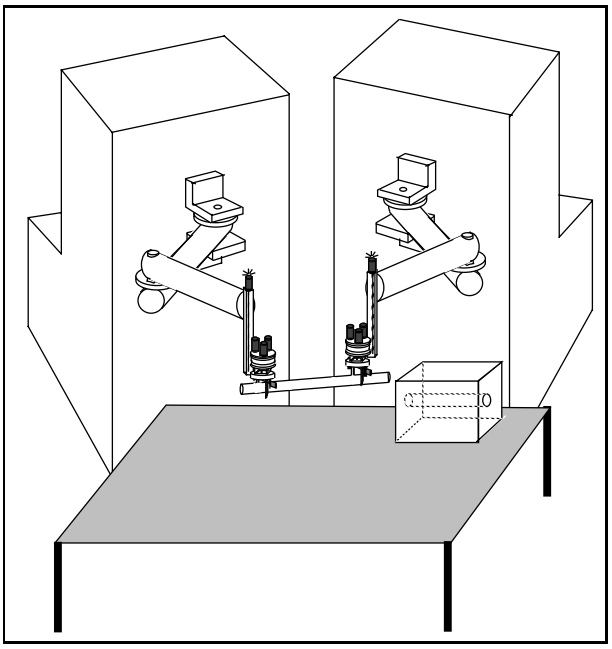

Figure 1: **Cooperative insertion task schematic**

1 Introduction

There are many tasks that are difficult or impossible to do with conventional, "one-armed" robots. Examples include manipulation of long/awkward parts or tasks requiring fine alignments (e.g. insertions) or large torques. These tasks can potentially be done well by a dual-armed robot, but only if both the robot and its control system are specifically designed to facilitate cooperation. Figure 1 exemplifies the basic goals of this research. Basically, we want to design and control a dual-armed robot to perform cooperative tasks autonomously. Additionally, we want the robot system to handle drivetrain flexibility and work effectively in an unstructured environment.

To accomplish these goals, we use several approaches: We design the robot and the control system concurrently, so they are matched well in key characteristics. We use endpoint and object sensing so the robot can work effectively in an unstructured environment. We use hierarchical control (on a multiple-processor real-time computer) to manage the data flow and computational burden. We incorporate exaggerated drivetrain flexibility, along with joint-torque sensors, to facilitate the study and control of flexibility in cooperative robots.

i	α_{i-1}	a_{i-1}	d_i	θ_i	q_i range
1	0	0	0	q_1	-1.27 – 1.27 rad.
2	0	406.4 mm.	0	q_2	-2.62 – 2.62 rad.
3	0	406.4 mm.	q_3	0	-249. – -97. mm.
4	0	0	0	q_4	$-\pi$ – π rad.

Table 1: Manipulator Kinematic Parameters

2 Experimental system

The experimental system consists of five components: the manipulators, the vision system, the objects, the real-time computer, and the workstation. The manipulators are both identical SCARA geometry arms with grippers. The basic geometry for each of the two arms is shown in figure 2 and described in detail by the kinematic parameters in table 1. The four links are named the upper-arm, fore-arm, slider, and wrist, respectively. The grippers have a grasp gap of 0 – 70 mm., and have a lifting capability of roughly 4 kg/arm. The arms sit side-by-side, separated by 615.4 mm. in the world Y direction. Both arms have deliberately exaggerated flexibility in the drivetrains of the first two links and joint-torque sensors built into the corresponding joints. The added flexibility provides a way to investigate how well control schemes work with non-ideal robot dynamics. Both arms are equipped with force/torque sensors at the the grippers as well as infra-red emitters for the vision system; together, these provide end-point sensing of position and force.

The vision system is composed of a CCD camera, a custom VME board, and a dedicated real-time processor. Together, they track points and objects fitted with infra-red emitters. There are two types of objects: the graspable parts are long, thin objects for two-handed manipulation; the insertion targets are rectangular boxes with through holes for part insertions/withdrawals. The real-time processor is a VME bus system with five 680x0 based processors and associated peripherals. The real-time software uses the VxWorks operating system and the ControlShell real-time framework, [1]. The real-time system communicated via Ethernet to the workstation, a SUN SPARCstation IPC.

3 Manipulator Modeling

Figure 2 shows the coordinates for one of the manipulator arms, along with other key variables. (The vertical translation and final wrist rotation are omitted for clarity.) We adopt Craig's [2] basic approach

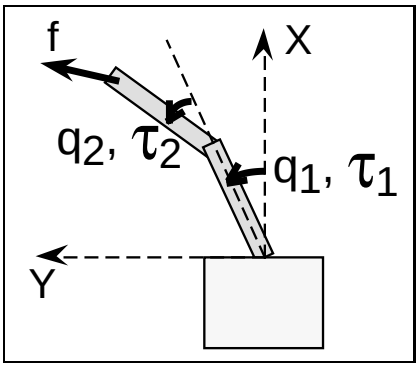

Figure 2: **Robot Coordinates**

Link	Mass (kg)	d_{cm} (mm)	I_{zz} (kg m^2)
Upper-arm	4.902	148.7	0.141
Fore-arm	7.020	241.8	0.241

Table 2: Mass properties of primary links

for kinematics and rigid-body dynamics. The mass properties for the two primary links are given in table 2. The mass center displacement for each link is in that link's x direction from the frame origin (due to link symmetry); these are denoted as d_{cm}. I_{zz} denotes the inertia of each link about a vertical axis through the mass center. Note that we have lumped the masses of the last two links (the slider and wrist) together with the fore-arm. This is a reasonable simplification of the rigid-body dynamics, given the mass and geometry of the last two links. The geometric and mass properties determine the details of the rigid body dynamics.

We treat the flexible sub-systems as effectively decoupled sub-systems, one for each degree-of-freedom (DOF) with exaggerated drivetrain flexibility. We use the rigid body model along with with the flexible subsystem models for control. Note that by this approach, we are implicitly considering the system to be composed of slow (rigid body) and fast (flexible subsystems) dynamics. This approach follows the general technique of singular-perturbation, as in [3]. [4] provides a rigorous development that supports this approach to flexible-drivetrain robot dynamics.

We denote joint coordinates by $\mathbf{q}$, and joint torques by $\boldsymbol{\tau}$, the corresponding actuator quantities by $\mathbf{q_a}$ and $\boldsymbol{\tau}_a$. For each of the two manipulators, the rigid body dynamics can then be represented as:

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{G}(\mathbf{q}) + \mathbf{J}'(\mathbf{q})\mathbf{f} \qquad (1)$$

where $\mathbf{M}$, $\mathbf{C}$, $\mathbf{G}$, and $\mathbf{J}$ are the mass, cen-

tripetal/coriolis, gravity, and Jacobian matrices respectively; $\mathbf{f}$ is the endpoint force vector. The drivetrain flexibility introduces additional dynamics to the manipulators' behavior. For each of the four flexible subsystems (two arms, two flexible drivetrains per arm), we can represent the additional coupled dynamics with the following scalar equations:

$$\tau = nk(q_a - nq) \quad (2)$$

$$\tau_a = I_a\ddot{q}_a + b\dot{q}_a + k(q_a - nq) \quad (3)$$

$$0 = I_{L_{eff}}\ddot{q} + nk(nq - q_a) + N_{nl} + C_{cpl} \quad (4)$$

where I_a denotes the actuator inertia, $I_{L_{eff}}$, denotes the effective inertia of the link (plus outboard structure) and n, k, and b are the gear ratio, spring constant, and damping, respectively. We use N_{nl} to represent the nonlinear terms for this DOF and C_{cpl} to represent the effect of joint and actuator quantities from other degrees of freedom. For the link dominant case, $\frac{n^2 I_a}{I_{L_{eff}}} \ll 1$, both of these terms can be neglected; this makes the subsystems effectively decoupled.

The link dominant design further simplifies the control of the flexible sub-systems, in that it simplifies the transfer function from actuator command to joint torque and makes it effectively constant – independent of robot configuration. In this case, we can solve for and simplify the this transfer function to:

$$\lim_{\frac{n^2 I_a}{I_{L_{eff}}} \to 0} \left\{ \frac{\tau(s)}{\tau_{a\_cmd}(s)} \doteq \frac{nk}{I_a(s^2 + 2\zeta\omega s + \omega^2)} \right\} \quad (5)$$

where

$$\lim_{\frac{n^2 I_a}{I_{L_{eff}}} \to 0} \left\{ \omega = \sqrt{\frac{k}{I_a}},\ \left| \frac{\tau(s)}{\tau_{a\_cmd}(s)} \right|_{s \to 0} = n \right\} \quad . \quad (6)$$

The constant gain and resonant frequencies of the flexible subsystems, along with their decoupling, greatly simplify the joint and arm levels of control.

4 Hierarchical control

Figure 3 shows the organization of the control system into four levels: task, object, manipulator and joint. Each level handles its own part of the overall control problem and communicates with the adjacent levels. This structure facilitates breaking up the control problem according to the time scales of the dynamics. The fast but relatively simple dynamics due to flexibility are handled at the joint level. The arm, object, and task control are increasingly more complex, but less demanding of fast sampling. Thus, the hierarchy provides clear boundaries for distributing the control computations among the real-time processors. Finally, the layers permit a certain amount of abstraction, for instance, the task and object level layers are unaware of the details of the arms' geometry; so the higher levels of control could be cleanly applied to different manipulators.

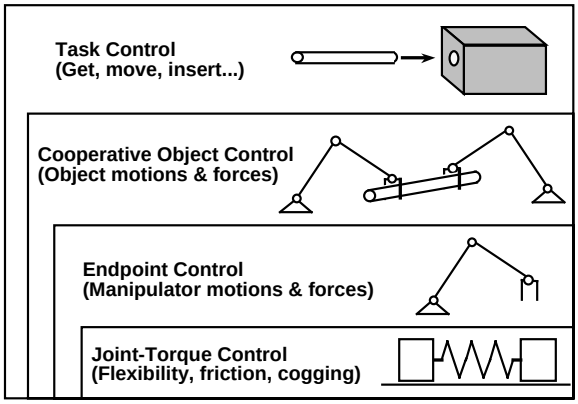

Figure 3: **Hierarchical control structure**

Joint control

The joint level is the lowest level of control for the robots. Its purpose is to make the manipulators appear to be ideal linkages with commandable joint torques, so that the manipulator level can model and control the arms via inverse dynamics, using equation 1. For the rigid DOFs, this is primarily a scaling due to gearing. However, for each flexibly-driven link, we have to control the associated dynamic subsystem as well. Following the approaches of [5] and [6], we use the joint torque sensors built into the robots and the derived transfer-function model (equations 5 and 6) as the basis for control. Models of this form were identified from frequency response data; the results are shown in table 3.

By paying attention to the control system consequences of the mechanical design, we can now take advantage of the link dominant case, (from section 3.) In this case, we can treat the fast dynamics as decoupled SISO subsystems that do not vary with robot configuration or payload. The price we have paid is that the gear ratios are limited. This limit is acceptable, since we are using stationary motors and drivetrains, anyhow. Concurrent robot/controller design has lead to a simpler control problem. For this robot, joint control needs to control only four decoupled, configuration-

Subsystem	ω (rad/sec)	ζ	nk/I_a
Rt. shoulder	$7.34 \cdot 10^1$	$7.25 \cdot 10^{-2}$	$3.07 \cdot 10^4$
Lt. shoulder	$7.32 \cdot 10^1$	$6.30 \cdot 10^{-2}$	$3.09 \cdot 10^4$
Rt. elbow	$1.04 \cdot 10^2$	$8.02 \cdot 10^{-2}$	$2.50 \cdot 10^4$
Lt. elbow	$9.60 \cdot 10^1$	$4.14 \cdot 10^{-2}$	$2.22 \cdot 10^4$

Table 3: Flexible subsystem model parameters

independent, second-order SISO systems.

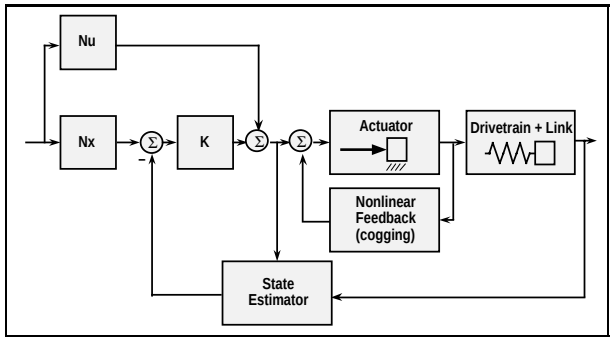

Figure 4: **Flexible subsystem block diagram**

The continuous subsytem models were put in state-space form, discretized (at 400 Hz.) and transformed to balanced-realization form. This procedure produced a model for each subsytem in the form of A, B, and C: state-transition, input, and output matricies, respectively. Figure 4 shows the structure of the joint-torque control loops used to control each of the flexible subsystems. The structure is basically the state command structure, described in [7], sect. 6.5.2. The only difference is that an inner loop is used to reduce (position dependent) cogging torques from the actuators (brushless servo-motors.) Predictive estimators were used; this helped minimize problems from computational delays. The feedback gains K were designed using LQG synthesis. Since this is a SISO system, we can treat this via symmetric root locus (SRL.) The state-weighting matrix $Q_1 = \rho C'C$, and the control weight $Q_2 = 1$.) In this technique, K becomes a function of the scalar weight ρ, and the we choose ρ for a suitable combination of performance and stability. Performance is judged primarily on transient response (ω_{cl}, ζ_{cl}) and accuracy, from D.C. loop gain. These goals must be traded off against the need for adequate stability margins to achieve robustness to nonlinearities and model imperfections. The estimator gains L were chosen in an analogous manner to SRL, where we let the process noise enter through the same matrix, B, as does the control; the process noise covariance, R_w, is fixed at unity, and the sensor noise covariance, R_v becomes the variable parameter. Table 4 shows the design parameters and the resulting performance measures.

Subsystem	Rt sh	Lt sh	Rt el	Lt el
R_v	0.05	0.05	0.02	0.02
ω_{est} (rad/sec)	371.3	372.4	421.5	397.4
ζ_{est}	0.680	0.680	0.669	0.671
ρ	0.5	0.5	0.1	0.15
ω_{cl} (rad/sec)	209.3	209.9	282.5	241.2
ζ_{cl}	0.659	0.659	0.651	0.643
D.C. loop gain	4.84	4.88	3.79	3.32
Gain margin	3.55	3.52	2.82	3.18
Phase margin (deg)	50.5	50.2	46.7	48.8

Table 4: Control design/performance parameters

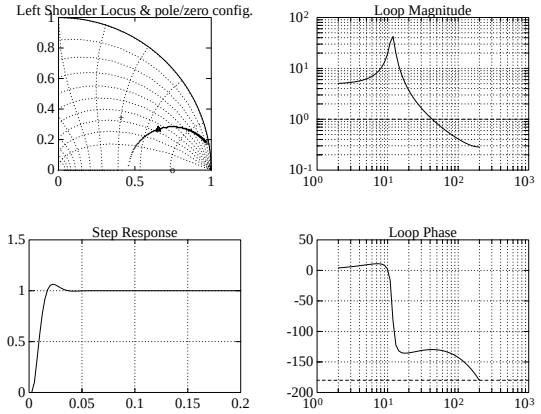

Figure 5: **Servo design for left shoulder**

Figure 5 shows some details of the design for one of the subsystems. The z-plane plot show the location of the open loop poles (x), the estimator poles (+), the closed loop poles (*), and the effective compensator zero location (o). This figure also shows a simulated step response, as well as the loop gain and phase characteristics. The closed loop quantities of table 4 may be compared to the open loop parameters in table 3; substantial improvements in frequency and damping are evident.

Figure 6 shows the open and closed-loop responses of the left shoulder subsystem to a 10 N-m step in torque. Note that the open-loop response has poor tracking at both levels, and has a slow, lightly damped

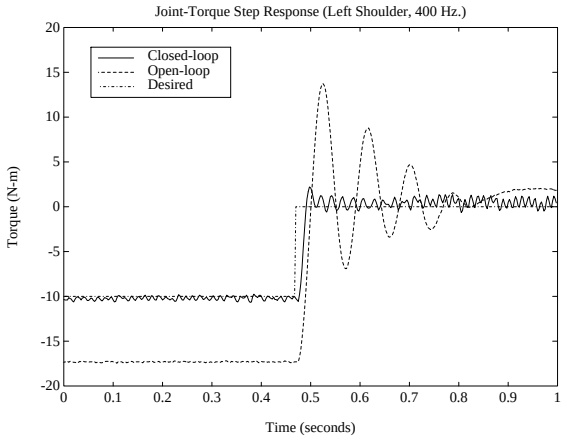

Figure 6: **Joint-torque response of left shoulder**

response. Contrast this with the closed-loop response, which shows good tracking at both torque levels, as well as a fast, well damped transient response. The closed loop response compares nicely to the simulation in figure 5 in overshoot and settling time. The noise is due to the PWM amplifiers driving the motors; it does not materially effect performance. This experiment shows that the joint level control is successful in achieving fast accurate torque tracking. Good torque tracking enables the arm control layer to control the flexible-drivetrain manipulators.

Manipulator control

The manipulator control layer's purpose is to control the end-point motions and forces of the two manipulator arms. The arms can be controlled independently, or they can work with the object layer to perform object-oriented cooperative control. The basis for this control layer is the concept of impedance control of Hogan, [8] and, specifically, the dynamic endpoint impedance formulation of Schneider, [9]. Dynamic endpoint impedance uses inverse dynamic control to make the arm endpoint behave as a programmable second order mechanical impedance – i.e. a programmable mass, spring, damper. The impedance is defined by m_{imp}, b_{imp}, k_{imp}, and the desired (zero force) trajectory, composed of the position, velocity and accelerations $\mathbf{x}_d$, $\dot{\mathbf{x}}_d$, and $\ddot{\mathbf{x}}_d$, respectively. The basic linear, second order impedance relationship is:

$$m_{imp}(\ddot{\mathbf{x}} - \ddot{\mathbf{x}}_d) + b_{imp}(\dot{\mathbf{x}} - \dot{\mathbf{x}}_d) + k_{imp}(\mathbf{x} - \mathbf{x}_d) = \mathbf{f} \quad (7)$$

where $\mathbf{x}$, $\dot{\mathbf{x}}$, and $\ddot{\mathbf{x}}$ are the endpoint position, velocity, and acceleration, respectively and $\mathbf{f}$ is the force measured at the endpoint. This impedance relationship can be solved for $\ddot{\mathbf{x}}$. This impedance-commanded acceleration is used, with the kinematics and manipulator dynamics (equation 1), to solve for the torques that will make the arm simulate the specified impedance. Additional arm forces, if desired, can be mapped through $\mathbf{J}'$.

The desired trajectory can come from two different places: in independent mode, the arm control level generates independent trajectories (fifth order splines) for the two arms; in cooperative mode, the desired trajectories for the two arms are specified by the object layer of control. The task level can also set m_{imp}, b_{imp}, and k_{imp} in cooperative mode. This programmability facilitates smooth control mode changes to and from cooperative operation. Figure 7 shows the

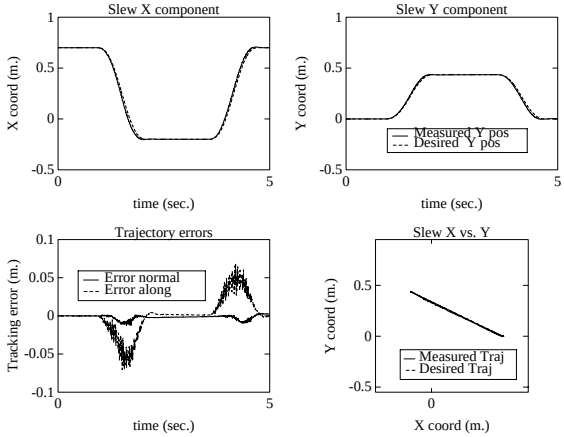

Figure 7: **One meter arm slew in 0.9 sec.**

left arm performing a rapid back-and-forth slew over one meter in 0.9 seconds, with a pause between the moves. The arm is using endpoint feedback and has impedance gains of $m_{imp} = 5$kg, $b_{imp} = 120$ N·sec/m, $k_{imp} = 1000$ N/m. These correspond to position control with $\omega = 14.14$ rad/sec and $\zeta = 0.85$, (2.25 Hz, slightly overdamped.) The slew time is limited by the available torques. The close tracking in Cartesian space shows that the nonlinear dynamics (quite significant for this arm and slew) are properly compensated. This experiment demonstrates the use of non-linear endpoint feedback in controlling a manipulator with significant drivetrain flexibility.

Object control

The object control level's purpose is to make the two manipulator arms work deftly together to manipulate objects. The object layer implements both coordinated and true cooperative control. Cooperative control is superior for our applications and is the technique described here. Our cooperative control is basically an implementation of object impedance control developed by Schneider, [9] (abbreviated as OIC.) The OIC approach is attractive — it is explicitly object based and is symmetric, as opposed to master-slave approaches. OIC provides the same unification of free-space motions and contact-force control for a manipulated *object* that impedance control provides for an individual manipulator. OIC permits explicit control of the interaction forces – the internal forces the object is subjected to. It also provides a convenient means to achieve remote center of compliance (RCC) behavior; this capability is useful in assembly tasks.

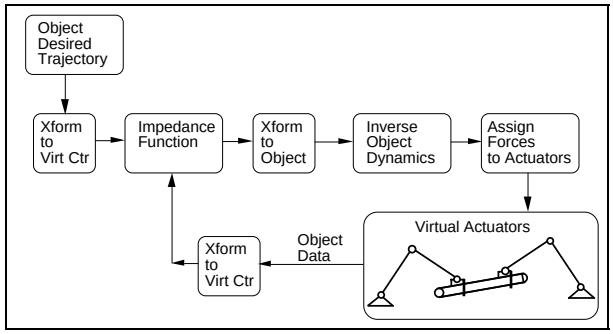

Figure 8: **Object impedance block diagram**

OIC is based on impedance control, with a few modifications. First, it uses the manipulators as "virtual actuators", accordingly, it requires tracking of endpoint forces (and accelerations) analogous to the endpoint impedance controller's need for torque tracking at the joints. The second change is the use of rigid-body kinematic transformations. OIC uses rigid-body kinematics to transform positions, velocities, and accelerations between the object's mass center (where its dynamics are easily calculated) and the virtual center (part of the object impedance specification.) The use of transformations and virtual actuators is shown schematically in figure 8.

Figure 9 shows a slew of a long object being manipulated by the two-armed robot in cooperative control mode. The slew moves the object in all DOFs possible: x, y, z and yaw (rotation about z.) This motion is a round trip maneuver, with each move taking 2 seconds. The tracking is good, and the lack of significant lag between desired and measured positions shows that the dynamic compensation is being handled well. The small errors present are primarily due to the limits on the achievable object impedance gains (due to sensor noise and sample rate) and the disturbance forces on the arms (principally from cabling.)

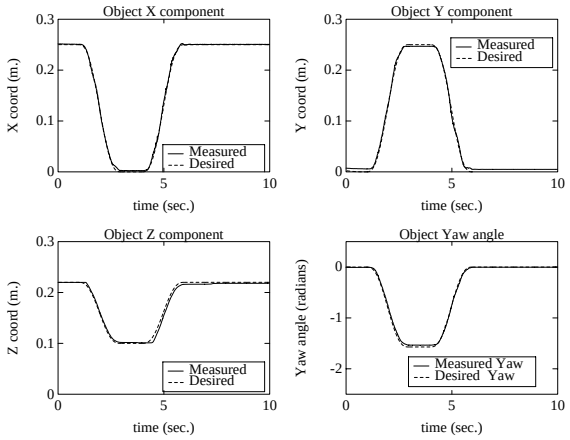

Figure 9: **Cooperative Slew of a long object**

Task Control

The task control level's purpose is to enable a user to direct the robot to perform useful manipulation tasks, and generally perform them autonomously i.e. without close (or any) supervision. Such tasks are typically multi-step operations, rather than a single maneuver of either an individual arm or object. Such tasks range from simple ones, such as part acquisition, to more difficult ones, such as long part insertion/withdrawal, which involve cooperative control during contact. In order to direct such multi-step operations, the task control layer requires a few basic primitive operations that can be put together to perform such multi-step tasks. Our primitives are mode/parameter switching, trajectory generation, and sequencing criteria. Mode switching takes the robot safely from independent control to and from cooperative control; parameter switching changes gains and impedance characteristics. Trajectory generation provides smooth trajectories for individual arms or for objects. Sequencing criteria include tests for successful completion of sub-tasks and monitoring/waiting

for a sub-task to complete. The task control layer is therefore not simply a servo-mechanism; instead it has some control-like elements (trajectory generation) and some discrete-event elements (mode changes and sequencing criteria.)

Figure 10: **Insertion task, leading edge in hole**

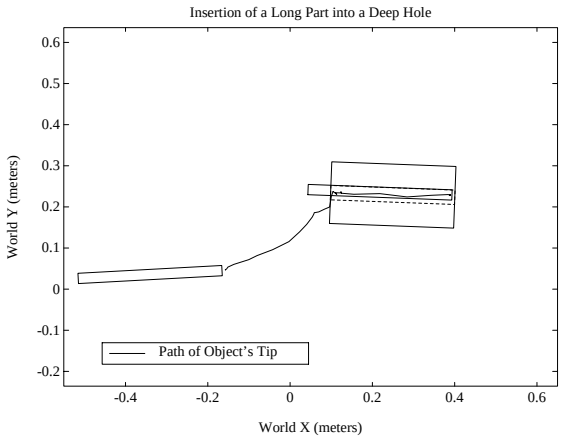

Figure 11: **Part motion during insertion task**

The primitives are used to program autonomous operations for the robot. The robot is currently capable of performing four different multiple-step, autonomous tasks. The first three are the ones already mentioned: part acquisition, part insertion, and part withdrawal. The most challenging task the robot can do is the assembly of a (specially modified) fluorescent lamp. This task demonstrates the same part insertion, while handling a delicate object – the fluorescent tube. The lamp assembly is not, however, fully autonomous, because the fluorescent tube cannot currently be tracked by the vision system. These experimental tasks have been documented on videotape, [10].

We now describe the insertion task, which is the most challenging fully autonomous task, (and is the most difficult part in the lamp assembly.) The task of inserting a long object into a deep hole is accomplished in a six step sequence. The steps use the task primitives to maneuver and insert the object into the deep (horizontal) hole in the target. The six steps are as follows:

Begin: The object is grasped cooperatively, with nominal impedance parameters, with the virtual center at the object's center.

Hover: The object moves to "hover" in front of the target, slightly tilted with respect to the hole's axis.

Contact: The object's commanded position is moved "inside" the target. This achieves a contact force of $F = K_{imp}/\Delta_{overlap}$.

Overshoot: The tip is commanded to slide along the target's face, toward and past the hole. This move, along with the tilt of the object, causes the object's leading edge to fall into the hole (the tip is not intended to reach its overshoot goal.)

Tip Inserted: The virtual center is moved outward, beyond the object's tip so that contact forces will result in alignment (RCC); the (rotational) stiffness is lowered, and the commanded position is aligned with the hole. This combination causes the object to align itself with the hole's axis, while maintaining contact force. These operations insert the tip into the hole.

Fully Inserted: The lead arm un-grasps the part; the control mode for both arms is switched back to independent control; the lead arm moves out of the way. The rear arm finishes the insertion single-handedly and then un-grasps the object and moves away.

Figure 10 shows the robot performing the cooperative long-part insertion task; the photograph was taken at the end of step 4 (overshoot), but before the force based alignment (via RCC) of step 5 has occurred. The leading edge of the part is securely in the hole. Figure 11 shows an overhead view of the actual path of the object's tip during the cooperative long-part insertion. The path shown is the path of the object's tip, as measured by the vision system; the initial and final *desired* positions of the object are indicated, as is the

outline of the insertion target. The small initial error is primarily due to cable forces on the manipulators. (As in the object slew experiment, sampling limits and delays due to the vision system limit the achievable impedance gains.) This experiment demonstrates that the robot can perform multi-step cooperative tasks, and do so autonomously. The task control layer can direct tasks involving both independent and cooperative control, including control in both free-space and contact-constrained conditions.

5 Conclusions

We have presented an overview of a dual-armed, cooperating robot with flexible drivetrains. We have described key details of the robot/control system's design and presented experimental results from each of the four levels in the control hierarchy. The system can autonomously perform multi-step tasks that include cooperative manipulation, e.g. long part acquisition, insertion, and withdrawal. The main ideas are the following: Concurrent design of robot and controller as an integrated system is a key to good performance. Hierarchical control is a highly effective structure for the control of a dual-armed, cooperative robot. Inner joint-torque loops have been successfully used together with non-linear endpoint feedback to control flexible-drivetrain manipulator arms. Cooperative control has been implemented on these manipulators, extending the object impedance control technique to the flexible-drivetrain case. The whole system has been experimentally demonstrated via cooperative tasks, including the insertion of a long part into a deep hole.

Acknowledgements

The authors gratefully acknowledge the support of DARPA (under contracts MDA903-86-K-0037, DAAA21-89-C-0002, and N00014-92-J-1809) and General Motors (via a G.M. Research Fellowship.) The authors would also like to thank Gerardo Pardo-Castellote for his careful review of this manuscript and the other members of the Aerospace Robotics Laboratory for their invaluable help and suggestions.

References

[1] S. A. Schneider, M. A. Ullman, and V. W. Chen. Controlshell: A real-time software framework. In *Proceedings of the 1991 IEEE International Conference on Systems Engineering*, Dayton, OH, August 1991.

[2] John J. Craig. *Introduction to Robotics Mechanics and Control*. Addison-Wesley, Reading, MA, 1986.

[3] Petar V. Kokotovic, R. E. O'Malley, Jr., and P. Sannuti. Singular perturbations and order reduction in control theory—an overview. *Automatica*, 12:123–132, 1976.

[4] Khashayar Khorasani and M. W. Spong. Invariant manifolds and their application to robot manipulators with elastic joints. In *Proceedings of the International Conference on Robotics and Automation*, pages 978–983, St. Louis, MO, March 1985. IEEE, IEEE Computer Society.

[5] J. Y. S. Luh, W. B. Fisher, and R.P. Paul. Joint torque control by direct feedback for industrial robots. *IEEE Transactions on Automatic Control*, AC-28(2), Febuary 1978.

[6] Lawrence E. Pfeffer, Oussama Khatib, and John Hake. Joint torque sensory feedback in the control of a PUMA manipulator. *IEEE Transactions on Robotics and Automation*, 5(4):418–425, August 1989.

[7] Gene F. Franklin, J. David Powell, and Michael Workman. *Digital Control of Dynamic Systems*. Series in Electrical and Computer Engineering: Control Engineering. Addison-Wesley, Reading, MA, second edition, 1990.

[8] Nevill Hogan. Impedance control: An approach to manipulation. *Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control*, 107:1–24, March 1985.

[9] S. Schneider. *Experiments in the Dynamic and Strategic Control of Cooperating Manipulators*. PhD thesis, Stanford University, Stanford, CA 94305, September 1989. Also published as SUDAAR 586.

[10] Lawrence E. Pfeffer. Two-armed, cooperative, flexible-drivetrain robot experiments videotape. Stanford University Aerospace Robotics Lab Videotape #B-50, May 1992. half inch VHS.