N90-29825

The Goddard Space Flight Center (GSFC) Robotics Technology Testbed

Rick Schnurr, Lead engineer, GSFC
Maureen O'Brien, GSFC
Sue Cofer, Lead author, Digital Equipment Corporation

ABSTRACT

Much of the technology planned for use in NASA's Flight Telerobotic Servicer (FTS) and the Demonstration Test Flight (DTF) is relatively new and untested. To provide the answers needed to design safe, reliable, and fully functional robotics for flight, NASA/GSFC is developing a robotics technology testbed for research of issues such as zero-g robot control, dual-arm teleoperation, simulations, and hierarchical control using a high-level programming language. The testbed will be used to investigate these high-risk technologies required for the FTS and DTF projects.

The robotics technology testbed is centered around the dual-arm teleoperation of a pair of 7 degree-of-freedom (DOF) manipulators, each with their own 6-DOF mini-master hand controllers. Several levels of safety are implemented using the control processor and a separate watchdog computer, as well as other low-level features. High-speed I/O ports allow the control processor to interface to a simulation workstation: all or part of the testbed hardware can be used in real-time dynamic simulation of the testbed operations, allowing a quick and safe means for testing new control strategies. The NASREM hierarchical control scheme, developed at the National Institute of Standards and Technology (NIST, formerly NBS), is being used as the reference standard for system design. All software developed for the testbed, excluding some of simulation workstation software, is being developed in Ada.

The testbed is being developed in phases. This paper describes the first phase, which is nearing completion, and highlights future developments.

1 Overview of the Robotics Technology Testbed

Much of the technology planned for use in NASA's Flight Telerobotic Servicer (FTS) and the Demonstration Test Flight (DTF) is relatively new and untested. To provide the answers needed to design safe, reliable, and fully functional robotics for flight, NASA/GSFC is developing a robotics technology testbed for research of issues such as zero-g robot control, dual-arm teleoperation, simulations, and hierarchical control using a high-level programming language.

The testbed currently is centered around the dual-arm teleoperation of a pair of 7-DOF manipulators manufactured by Robotics Research Corporation (RRC); see Figure 1. Each arm has an extension of almost 6 feet; they're mounted on one stand approximately 8 feet from the ground, 1.5 feet apart, simulating a right and left arm. Master-slave (teleoperation) control of the RRC arms is accomplished with a pair of 6-DOF mini-master hand controllers from Kraft Telerobotics (Kraft): see Figure 2.

Figure 1: Dual Mounted Robot Research Corporation (RRC) Slave Arms

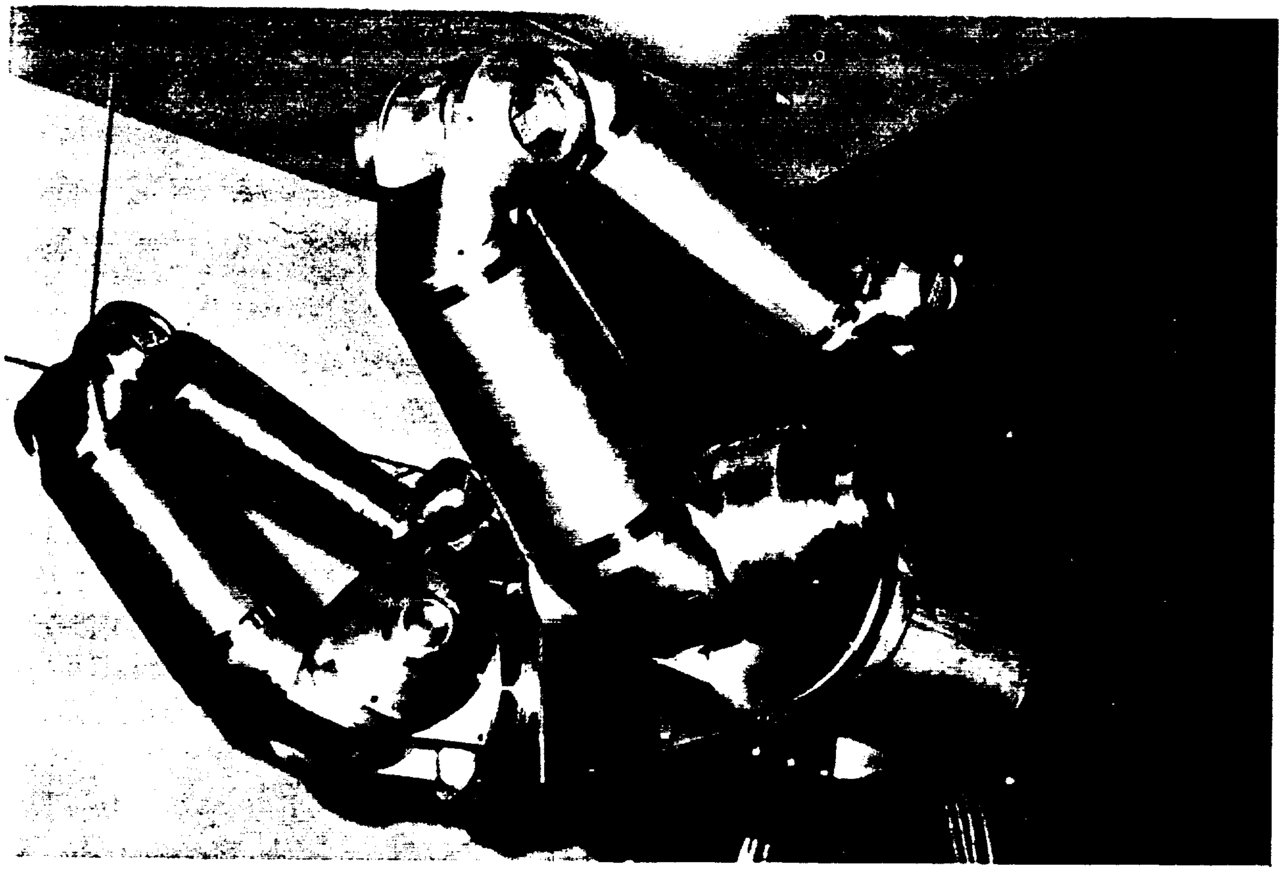
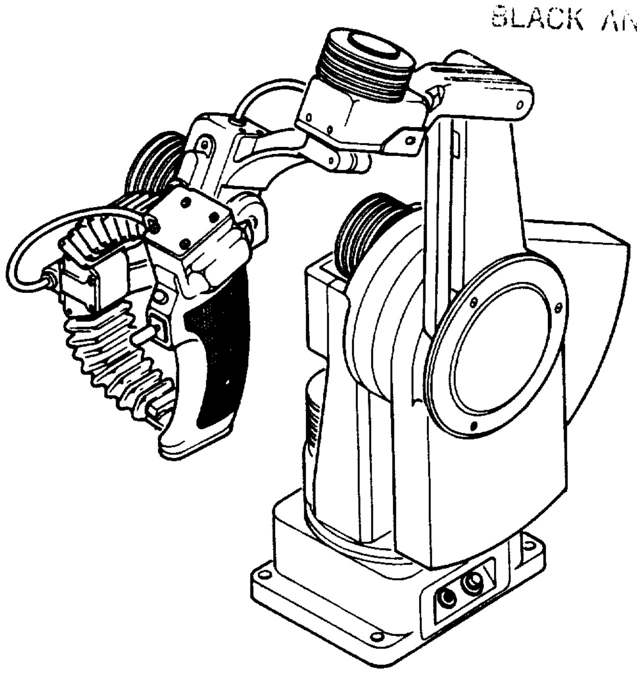


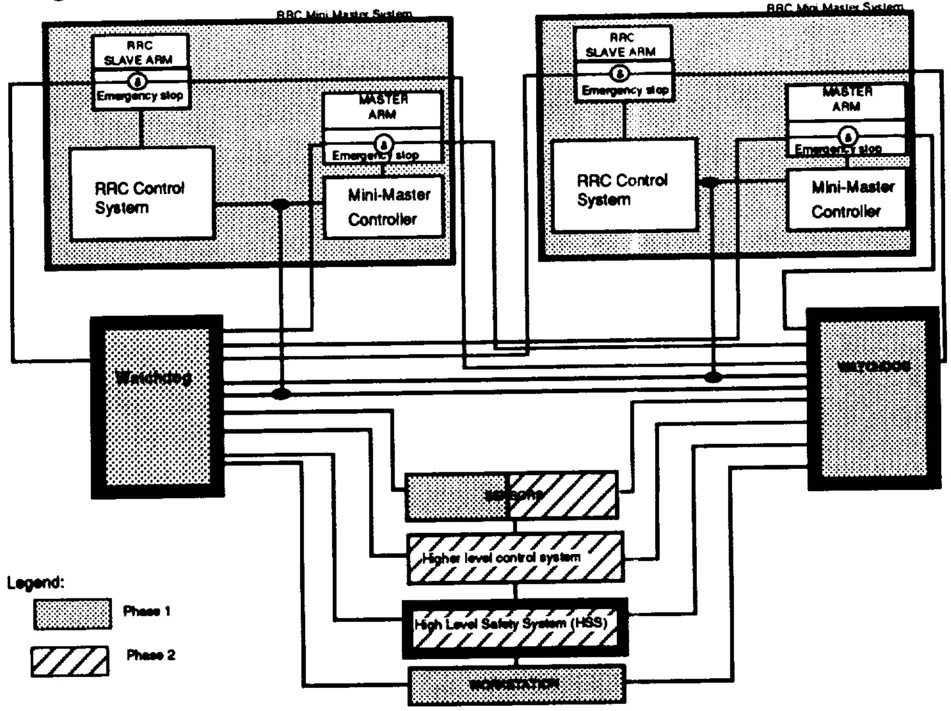
Figure 2: Kraft Telerobotics Mini-Master Arm

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH



Additional control switches on the Kraft master hand controller allow for control of end-effectors mounted of the RRC slave arms. Algorithms implemented in the control processor provide all coordinate transformations between 6-DOF and 7-DOF space as well as several different force-feedback control schemes. Several levels of safety are implemented using the control processor and a separate watchdog computer, as well as other low-level features. Figure 3 diagrams a high-level system integration of the testbed.

Figure 3: Diagram of Robotics Technology Testbed



I/O ports allow the control processor to interface to a Silicon Graphics IRIS workstation. All or part of the testbed hardware can be used in real-time dynamic simulation of testbed operations, including real-time graphics displays of the simulated components. For instance, the system may be configured to allow the teleoperator to drive a computer model of the slave arms using the actual master arms; the operator can watch the slave arms perform the motions in graphics simulation, and force feedback based on the (simulated) dynamics model can be fed back to the control processor of the actual master arms. This approach offers a quick and safe means to test new control strategies.

The NASREM hierarchical control scheme, developed at the National Institute of Standards and Technology (NIST, formerly NBS), is being used as the reference standard for system design. All software developed for the testbed, excluding some of simulation workstation software, is being developed in Ada: this will provide information as to how useful Ada is as a robot control language. Different Ada code designs will be explored, and the performances of each tested for their suitability to robot control.

The testbed is being developed in phases. This paper describes the first phase, which is nearing completion, and outlines the remaining phases. Phase I includes set up of the physical testbed, design and manufacture of required hardware for system modifications and interfacing, design and implementation of RRC control modifications, derivation of optimal jacobians and kinematic matrices for both the master and slave arms, and design and implementation of a watchdog (servo level) safety system for testbed operations. At the end of Phase I many of the hardware components for the testbed will have been installed and tested. A basic robot control algorithm will have been implemented, and the safety system will be completely defined, the required hardware procured hardware, and the initial software integrated into the RRC robot system.

2 Kinematics

The forward kinematic matrices, referenced in this paper, are 4x4 homogeneous transformations which use joint angle data and relates the base coordinate frame to the end-effector coordinate frame; given joint angles and the base coordinate frame, then, the pose (position and orientation) of the end-effector can be determined. Similarly, inverse kinematics provide a (complex) relationship from a position in the end-effector coordinate frame to the manipulator joint angles. The jacobian matrix specifies a mapping from joint anglular velocities to resolved cartesian velocities in the end-effector coordinate frame. All kinematic and jacobian matrices used in the testbed contol system, unless otherwise noted, are generated using the following procedure:

- Generate the required input equations for MACSyma: MACSyma is a Lisp-based system written by Symbolics that performs symbolic algebra.
- MACSyma is executed; the output is actual FORTRAN code of the matrix.
- The FORTRAN code is downloaded to an IBM-PC.
- An optimizer has been written by the robotic technology testbed team to optimize the MACSyma generated code; it runs on the PC, using the machine-generated FORTRAN code as input. The optimizer extracts all occurrences of trigonometric functions and assigns their value to a variable, which will be used in the actual matrix computations: this ensures all trigonometric functions are computed only once. Next, in a recursive algorithm, common factors in the MACSyma equations are pulled out and, again, their value assign to a variable which will be used in actual matrix computations. Lastly, the optimized code is translated to Microsoft C-language; the optimizer is being extended to generate optimized Ada code.
- The optimized C code is then tested using numeric examples generated by MACSyma.

The timings presented in this paper, then, represent the execution speed of the resulting optimized optimized code in Microsoft C on a Compaq 386/v20.

2.1 The Kraft Mini-master

The Kraft mini-master forward kinematics matrix are fairly straightforward.

The Kraft jacobian transpose is a 6x6 matrix which relates the forces seen at the Kraft handle in handle coordinates to Kraft joint torques. This matrix can be used to reproduce forces/torques at the handle of the mini-master which were read by a wrist sensor on the RRC slave arm. The execution speed of the resulting optimized code is 1.7 msec.

The Kraft jacobian transpose multiplied by a 6 element force vector produces a matrix which directly relates Kraft handle forces to Kraft joint forces. This matrix was computed and optimized separate from the jacobians above in order to minimize the number of terms which needed to be computed. The execution speed of the resulting optimized code is 1.8 msec.

A dynamic model for the Kraft is in process.

2.2 The RRC Slave Arms

The RRC forward kinematics matrix is the homogeneous transformation (position/orientation) between the RRC base coordinates and the RRC end-effector coordinates.

The RRC jacobian transpose is a 6x7 matrix which relates the forces seen at the end effector in end-effector coordinates to the joint torques. This matrix can be used for impedance or compliance control. The execution speed of the resulting optimized code is 1.8 msec.

The RRC jacobian pseudo-inverse is a 6x7 matrix which relates a cartesian velocity vector measured in the end-effector coordinates of the RRC to a joint space velocity vector for the RRC. Since the RRC is a redundant manipulator (7-DOF) there are an infinite number of possible inverse jacobian matrices. The matrix used in the testbed control system is based on the following Moore-Penrose pseudo-inverse equation:

$$J^{dagger} = J^T * (J * J^T)^{-1}$$

where J^{dagger} is the jacobian psuedo-inverse. The derivation of the equation is based on minimizing the sum of the squares of the joint velocities; this maximizes the speed of the arm. The disadvantage to this is that the elbow (the redundant motion plane) is allowed to move without restraint.

The $J * J^T$ matrix has been derived using the process outlined above (optimized MACSyma code); software, based on Gaussian elimination, had to be written to numerically compute the inverse of this matrix because it is too complicated for MACSyma to invert symbolically. A breakdown of the timings for the resulting optimized code are:

1.8 msec - compute the jacobian J

2.7 msec - compute $J * J^T$

9.7 msec - invert $J * J^T$ and then multiply it by J^T

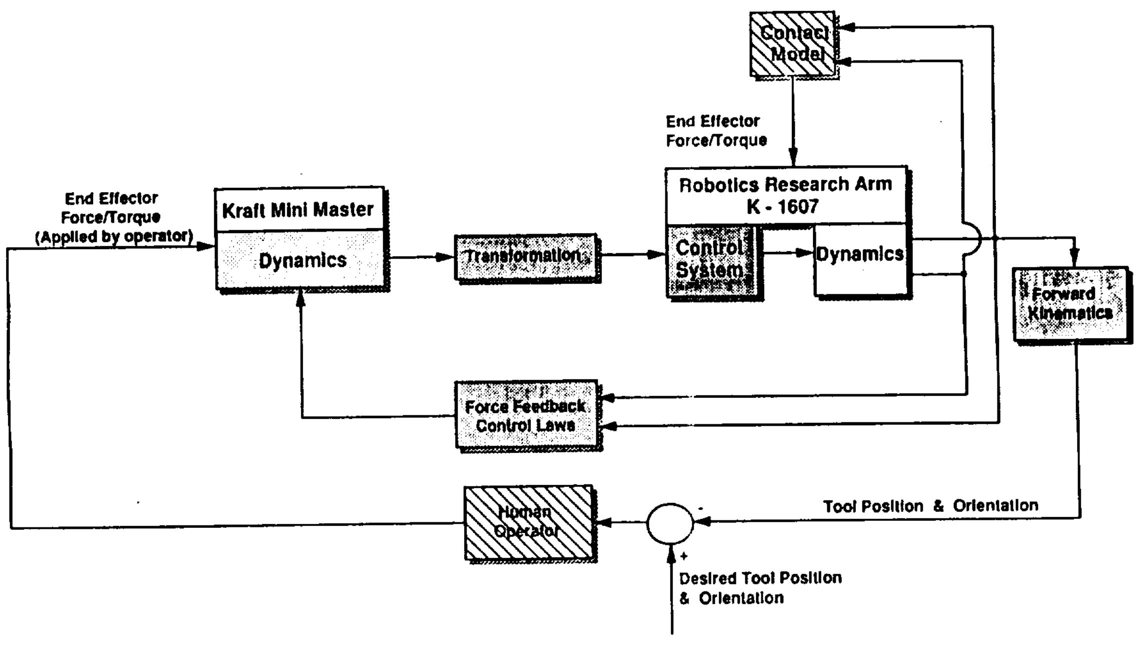
14.2 msec - total execution time to compute the jacobian psuedo-inverse.

Other inverse kinematics schemes have been explored; at least one of them will be implemented and tested, in addition to the jacobian psuedo-inverse described above, during the Phase I effort.

3 Control Algorithms

Phase I of the GSFC robotic technology testbed is concentrating on the NASREM servo level control. Trade-off analysis between using the joint torque sensors and a wrist force torque sensor are continuing; both will probably be implemented. A high-level diagram for the testbed teleoperation control loop is shown in Figure 4.

Figure 4: GSFC Robotic Technology Testbed Teleoperation Control



Contact Model
End Effector Force/Torque
End Effector Force/Torque (Applied by operator)
Kraft Mini Master
Dynamics
Transformation
Robotics Research Arm K-1607
Control System
Dynamics
Forward Kinematics
Force Feedback Control Laws
Human Operator
-
+
Tool Position & Orientation
Desired Tool Position & Orientation

3.1 Force Reflection

The first attempt at force reflection will be done in cartesian coordinates. A wrist sensor mounted on the RRC slave arm(s) is used as the source of force/torque data; these forces/torques will then be translated back to the Kraft mini-master. There are two types of force feedback which have been explored: position-position and force-rate. The position-position force reflection generates a torque on both the master and slave proportional to the differential position between the two. When the rigidity of the RRC control loop and the weight of the manipulator are considered, position-position force feedback does not look promising. However, due to the large size of the RRC slave arms, they cannot move fast enough to keep up with the smaller mini-masters; some means must be found to keep the operator of the mini-master from getting to far ahead of the slave manipulator. A variant of position-position force reflection which matches the master arm motion to the slave arm motion accomplishes this goal.

The force-rate mode of force reflection takes data from a force/torque sensor on the slave and uses this data to generate force/torque commands on the master. The difference in position between the master and slave causes a torque to be impressed on the slave. The differential rate generates torques on both the master and slave.

A fully robust force reflection algorithm will ultimately integrate both control methodologies: a design and implementation for this integration is currently in progress.

3.2 Indexing

Another major subject which must be addressed is the indexing scheme. Indexing is the act of disabling the control algorithm for the master arm, allowing the operator to move the master to a more comfortable (workable) position, and then re-initializing the control parameters and resuming teleoperation of the slave arm. There are two challenges embedded in the indexing problem: first, to re-establish the control link between the master and the slave without moving the slave. Second, to resolve the reflected forces on the indexed master so the axis of motion match the force reflection axis of the slave arm. Separate matrix formulations for indexing will be required, and are currently being calculated and optimized using the method outlined in Section 2.

Experimentation with one indexing scheme has been completed: however, after analysis of the positioning requirements, one indexing scheme is not going to satisfy all possible teleoperation tasks and a hybrid of different control schemes will ultimately be required. This issue will probably not be totally resolved until later phase developments.

3.3 Teleoperation

Two formal approaches to teleoperator control will be explored (see Section 2). Both center on developing formal control equations, performing stability analysis/simulations, simulating control algorithms using dynamic simulations, and then implementing the control code in Ada. One group will be focusing on advanced adaptive control schemes: this group is funded through a grant to Catholic University. The second is the testbed team, which will be heavily involved in development of control algorithms using the real robots and DTF and FTS tasks as a basis. These efforts are scheduled to be completed next year.

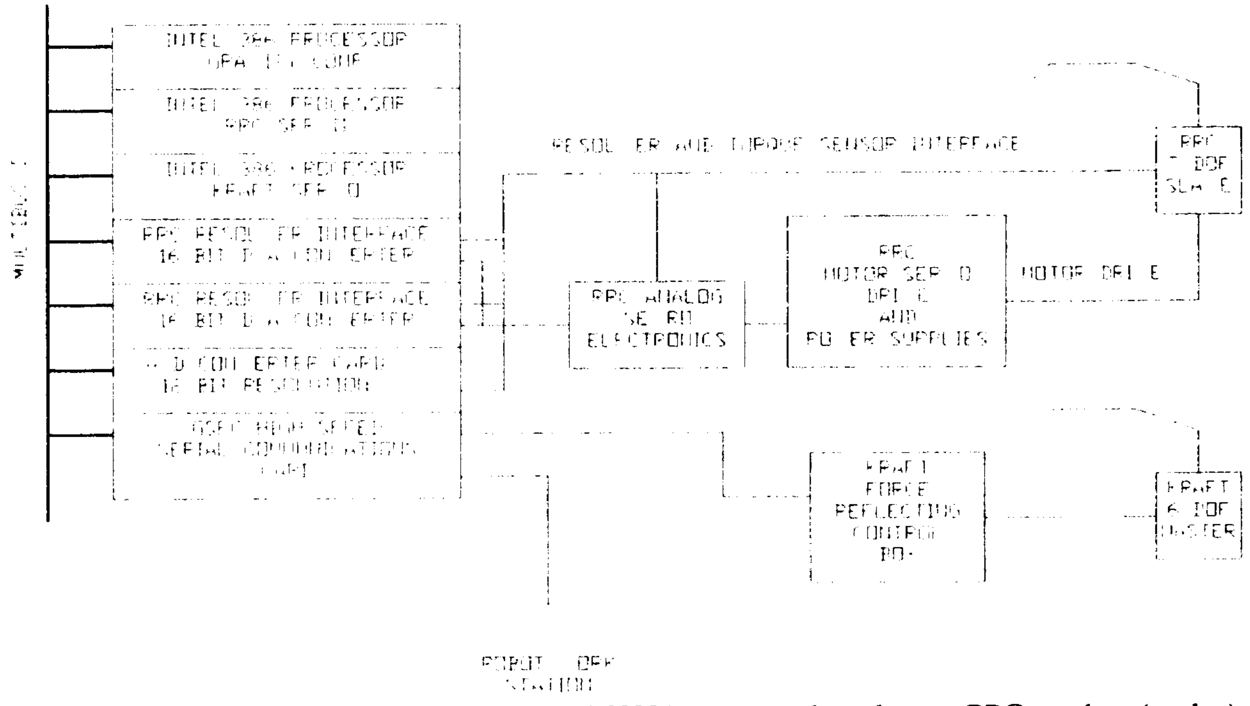
3.4 Data Collection and Testing

Data collection and testing will ultimately be required for dynamic simulation efforts. All tap points available in the RRC controller have been brought out to connectors where they can be hooked to a oscilloscope and later to a data acquisition system. One possible data acquisition system is the Safety system. The Safety system, by design, already requires most of this data to perform it's watchdog functions (see Section 6). In fact, once the Safety system is interfaced with a micro-VAX a complete self contained data collection/reduction station will exist.

4 Hardware

The GSFC robotic technology testbed control electronics are shown in Figure 5.

Figure 5: GSFC Robotic Technology Testbed Hardware Diagram


INTEL 386 PROCESSOR
INTEL 386 PROCESSOR
INTEL 386 PROCESSOR
RRC RESOLVER INTERFACE
RRC RESOLVER INTERFACE
GSFC HIGH SPEED SERIAL COMMUNICATIONS
RRC RESOLVER AND TORQUE SENSOR INTERFACE
RRC 7 DOF SLAVE
RRC ANALOG SERVO ELECTRONICS
RRC MOTOR SERVO DRIVE AND POWER SUPPLIES
MOTOR DRIVE
KRAFT FORCE REFLECTING CONTROLLER
KRAFT 6 DOF MASTER
ROBOT OPERATOR STATION

The testbed control computer will use three Intel 80386 processor boards, two RRC resolver (analog) to digital/digital to analog converter (DAC) cards, one Intel parallel I/O card, a 12 bit resolution analog-to-digital converter (ADC), and one high speed serial card, all on a multibus I.

The first processor, the RRC servo, will read resolver position, the analog joint velocity, and the joint torque sensor for each joint of the RRC arm. It will then calculate a motor torque and set a DAC on the resolver card.

The second 80386 processor, the Kraft servo, will be interfaced to a Kraft mini-master over a RS422 link. This processor will read position data from the mini master, implement the force reflecting control algorithm, and send joint torque data to the mini-master.

The third 80386 processor, gravity compensation, serves two functions: first, it computes the torques which will be seen at each slave joint due to gravity. To do this the computer will have to execute a recursive force transformation algorithm, requiring the positions and masses of the link centers of gravity. The expected joint torques will be available to the other processors for use in compensating for gravity. The second function performed by the processor is to provide a way to limit allowable end-effector positions and to check for run away controller behavior: if either is detected operations of the robots will be shut down.

The RRC resolver interface cards allow the testbed control system to interface directly with the RRC joint resolver and torque sensor. It receives joint resolver and torque data from the the RRC arms and makes it available to the testbed control system. There are also DACs present on the card: these convert the motor torques, calculated by the testbed control system, to analog signals and sends them to the existing RRC analog servo electronics.

The analog-to-digital converter (ADC) card will be used to digitize the velocity and torque signals with 12-bit accuracy. The signals read by the ADC are buffered by a GSFC designed buffer card. This card makes these signals available to the Safety system without allowing the Safety system to corrupt them.

Seven analog boards (not shown in Figure 5) are connected to the existing RRC motor servo drive cards. These analog boards serve a dual function: first, they process the motor voltage and motor current signals for the Safety system, and, second, they compute the motor rpm. The motor rpm data is compared to the resolver velocity; also, the joint velocity is compared to a preset velocity limit. A deviance in either will cause the RRC robot to shut down. The velocity limits will be set very low when new servo control software is being tested.

The digital I/O card (not shown in Figure 5) is the computer's interface with the RRC joint home switches, the servo/enable status indicators, and the output drivers for panel lights and the arm enable relay.

The high-speed serial interface subsystem can be broken down into two parts: the first communicates with the Kraft mini-masters over a 2-wire packet interface, RS422 asyncronous protocol at 93750 baud. The Kraft is a slave in the communications protocol; when a valid torque command packet is received by the Kraft controller, a position report packet is sent out. The second part of the serial interface communicates with the Safety system and the RRC workstation. These interfaces are still in their definition phase. Long term it would be desirable to use a standard networking protocol between these elements. Candidates would be IEEE 802.3, MIL-STD-1553B, or high-speed serial multi-drop.

4.1 Modifications to the Existing RRC Arms

Two 80386 processors were added to the 80386 processor already present in the controller. A 12-bit ADC was added to measure joint torque and velocity signals. A buffer board was added to isolate critical signals for safety system. Seven analog boards were added to buffer and filter motor voltage and current. These motor interface boards also calculate motor velocity using motor voltage, motor current, and motor circuit resistance.

The testbed RRC joint control loop, which incorporates the new hardware, is as follows: the ADC card will be used to digitize the velocity and torque signals read from the RRC robot joints with 12-bit accuracy. The signals read by the ADC are buffered by a GSFC designed buffer card. This data is used by the first 80386 processor to compute the desired joint motor torques, which is sent to a DAC on the resolver card. The voltage from this DAC chip is connected to the analog torque loop electronics. The existing motor servo amps for the RRC arms can then be driven by the output of this motor torque loop.

The high-speed serial card has been installed and is working in RS232 mode. The PC board is currently being modified to interface with the RRC high-speed serial card in RS422 mode at 288K baud.

5 RRC Arm End-effectors

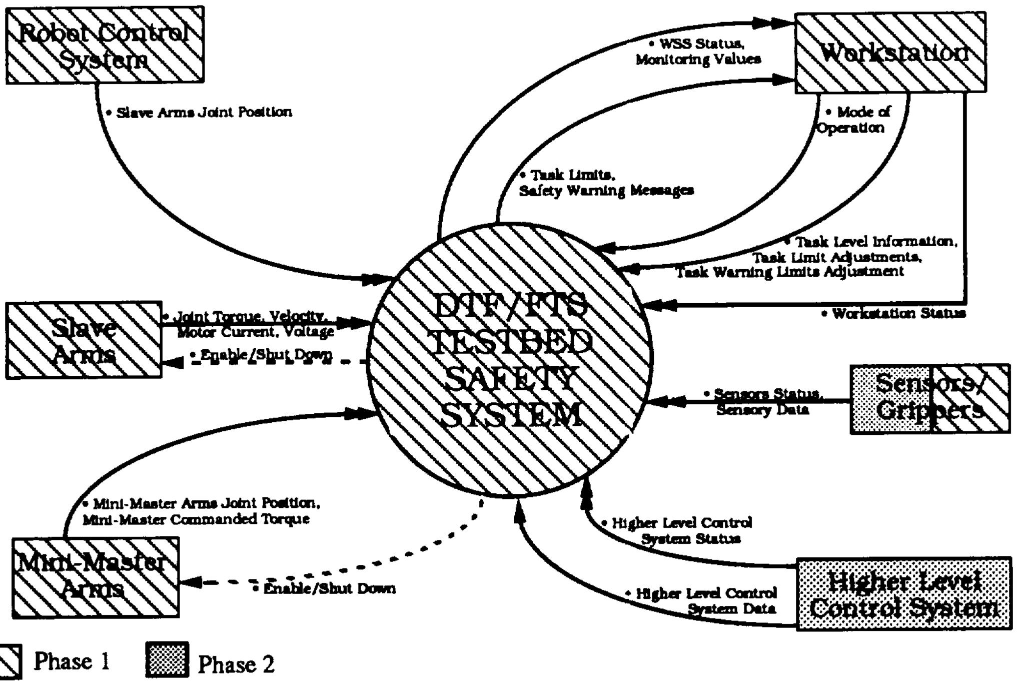
Several end-effectors, and a tool change and storage concept are currently being designed for the RRC arms. There will be interface requirements between the end-effector controller and the robot control at two levels. The first allows the end-effector to operate. Examples of this are using tool plate roll to unscrew a hex bolt through a ratchet, using tool plate roll along with a special fixture to install and stow end-effectors, and using the robot's wrist and joint force/torque sensors to provide active compliance. The second level allows the robot and the operator to verify proper execution of robot related tasks. Examples of actions which need verification are gripping something and checking that gripper distance is correct, verifying the proper tool is attached to the robot, verifying status and actions of tool change out apparatus, and verifying and monitoring proper gripper operation.

A final design concept for the interactions between the RRC controller and the controller for the end-effector(s) will exist at the end of Phase I. Also, several prototype controllers will have been tested on PUMA robots. The construction of the computer based end-effector controller(s) and the software required to run it will be completed. Mechanisms to give the capability of having a runoff between several different ORU change out tools will have been built, and a tool auto-change and storage unit will have been designed and be near the end of it's fabrication cycle by the end of Phase I.

6 The Safety System

Figure 6 diagrams the data flow between the safety system and the other components of the technology testbed telerobot control system.

Figure 6: Safety System Diagram


Robot Control System
• Slave Arms Joint Position
• WSS Status, Monitoring Values
Workstation
• Mode of Operation
• Task Limits, Safety Warning Messages
• Task Level Information, Task Limit Adjustments, Task Warning Limits Adjustment
• Workstation Status
Slave Arms
• Joint Torque, Velocity, Motor Current, Voltage
• Enable/Shut Down
DTF/FTS TESTBED SAFETY SYSTEM
• Sensors Status, Sensory Data
Sensors/ Grippers
• Mini-Master Arms Joint Position, Mini-Master Commanded Torque
Mini-Master Arms
• Enable/Shut Down
• Higher Level Control System Status
• Higher Level Control System Data
Higher Level Control System
Phase 1
Phase 2

The Safety System is designed to ensure the safe operations of the RRC and Kraft mini-master arms. Safe operations, as defined by the testbed team, requires that operation of the robot does no damage or harm to an operator or bystander, to itself, to any objects in the robots workspace. Ironically, this is a very unexplored area of robotics; as a result, the robotic technology testbed has turned out to be a testbed for exploring robot safety technology as well.

The current testbed Safety System is composed of several subsystems. The High-level Safety System (HSS) has knowledge about the task that is being performed and determines the safe operational and warning limits for the task. These limits are sent to the Watchdog Safety System (WSS). The WSS, which exists at the servo level of telerobot control, monitors robot and sensor data to ensure that the data is within the safe limits determined by the HSS. It is also responsible for monitoring the health of other computers in the testbed robot control system such as the workstation computer and the robot controller.

6.1 The Watchdog Safety System (WSS)

The Watchdog Safety System monitors the health status of the testbed control system, monitors robot and sensor data to ensure that the robots are operating within safe limits, and safely shuts down the robot(s) when an unsafe condition exists.

There are many functional requirements for the WSS. It must be two fault tolerant, or redundant to fail to safe when certain hardware errors are detected. The WSS monitors the health status of different subsystems of the testbed telerobotic control system; if any of the components fails, WSS shall safely shut down the robot. The WSS also monitors the robot operational data to ensure it is operating within safe limits. Operational data includes motor current, joint position, joint velocity, joint acceleration, joint torque, sensor data, and positions entering forbidden volume. The WSS also monitors the state of the robot during operations, ensuring, for example, that end-of-arm tooling and workpieces are not inadvertantly released (in zero-g, this condition could be disastrous).

The WSS, Figure 6, was designed to meet these functional requirements. The WSS receives the required safe operational limits, listed above, from the HSS; the workstation operator can override these limits to a more constrained boundary, but cannot increase them. All subsystems must transmit health status data at regular (to be defined) intervals. The WSS also transmits system status to the workstation and HSS subsystems.

There are many safety issues which are not clearly defined; for example, what is meant by "safe shut down of the robot". This is usually defined as immediate cessation of robot motion; however, nothing is said about the state in which the robot is left. Shutting down the robot could involve simply applying **brakes** immediately to all robot joints, backing off (reflex withdrawal) and then applying **brakes** instantly applying **brakes** and then putting the robot to a compliant mode, or simply stop sending signals to the robot, leaving the robot in the last known safe state. There are safe and unsafe aspects to all of these approaches, usually dependent on the particular robot task in which the anomaly occurred. One of the things the testbed safety team will be looking into is analyzing the impact of the different safe shut down schemes on both the system and the operating environment.

Directly related to the issue of a safe shut down is the definition of a safe return to operation after a shut down has occurred. A specific restart procedure is not clearly defined: it could involve recalibrating the robot, returning to home, resetting operational parameters, or specific operator action(s) could be required, to name a few. Again, this is one area to be researched by the testbed safety team.

6.2 The High-level Safety System (HSS)

This system uses task level information to determine operational and warning limits for the WSS. It is suspected that some level of colision avoidance will be performed at this level. The HSS will be completely defined at the completion of Phase I.

7 Future Development of the Robotics Technology Testbed

The following goals are proposed for the Phase II effort of the robotics technology testbed implementation:

- Implement improved force reflecting algorithms.
- Incorporate the higher levels of the NASREM model, written in Ada, into the RRC control system for autonomous operations. This system, the Hierarchical Ada-language Robot Programming System (HARPS), is currently under development at the GSFC robotic technology testbed: Stephen Leake of NIST is the lead engineer. A paper describing HARPS is being presented and published at the 1989 IEEE International Conference on Robotics and Automation in May, 1989.
- Integrate end-effector controller(s) into existing RRC control, generate tasks to exercise it, and perform tasks. Data will be collected from these tests and presented as a deliverable item. The task set will include ORU latching and unlatching, truss node assembly, and robot to robot hand off of object(s).
- Support the dynamic simulation parameter characterization effort.
- Demonstrate the validity of the University of Iowa's dynamic model using actual RRC robot data. The University of Iowa will prepare a plan involving RRC robot tests and perform these tests under the direction of testbed team members. The reduction of data from these tests may lead to a better model of the RRC robots.
- Demonstrate validity of the University of Iowa's IRIS-IRIS model using actual RRC robot data. Two independent IRIS systems, interfaced over an ethernet, are used for the model: one to generate data and the other for graphics. The model validation will be done at the same time the other Iowa model is verified.
- Investigate sensor technologies, specifically how beneficial different sensor data would be to the Safety System.

At the end of Phase II the entire telerobot system will be fully integrated and tasks will have been performed to demonstrate its capabilities. Phase II is scheduled to be completed in August of 1989. A detailed definition of future phases will be completed during the Phase I effort.