Contractor Report ARFSD-CR-90004

DEVELOPMENT, IMPLEMENTATION, AND VALIDATION OF CONTROL ALGORITHMS FOR THE DUAL-ARM ROBOTIC MANIPULATION OF A SINGLE OBJECT

S. J. Tricamo
F. L. Swern
Peritus, Inc.
767 Broadway
Norwood, NJ 07648

N. P. Coleman
Project Engineer
ARDEC

May 1990

US ARMY
ARMAMENT MUNITIONS
& CHEMICAL COMMAND
ARMAMENT RDE CENTER

U.S. ARMY ARMAMENT RESEARCH, DEVELOPMENT AND ENGINEERING CENTER

Fire Support Armaments Center

Picatinny Arsenal, New Jersey

The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

The citation in this report of the names of commercial firms or commercially available products or services does not constitute official endorsement by or approval of the U.S. Government.

Destroy this report when no longer needed by any method that will prevent disclosure of contents or reconstruction of the document. Do not return to the originator.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release, distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER Delivery Order 1291	5. MONITORING ORGANIZATION REPORT NUMBER Contractor Report ARFSD-CR-90004
6a. NAME OF PERFORMING ORGANIZATION Peritus, Inc. — 6b. OFFICE SYMBOL	7a. NAME OF MONITORING ORGANIZATION ARDEC, FSAC
6c. ADDRESS (CITY, STATE, AND ZIP CODE) 767 Broadway Norwood, NJ 07648	7b. ADDRESS (CITY, STATE, AND ZIP CODE) P.O. Box 12211 Research Triangle Park, NC 27709-2211 (cont)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION ARDEC, IMD STINFO Br — 8b. OFFICE SYMBOL SMCAR-IMI-I	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER DAAL03-86-D-0001
8c. ADDRESS (CITY, STATE, AND ZIP CODE) Picatinny Arsenal, NJ 07806-5000	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. — PROJECT NO. — TASK NO. — WORK UNIT ACCESSION NO.

11. TITLE (INCLUDE SECURITY CLASSIFICATION)
DEVELOPMENT, IMPLEMENTATION, AND VALIDATION OF CONTROL ALGORITHMS FOR THE DUAL-ARM ROBOTIC MANIPULATION OF A SINGLE OBJECT

12. PERSONAL AUTHOR(S)
S. J. Tricamo and F. L. Swern, Peritus, Inc., N. P. Coleman, Project Engineer, ARDEC

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (YEAR, MONTH, DAY)	15. PAGE COUNT
Final	FROM Feb 89 TO Dec 89	May 1990	73

16 SUPPLEMENTARY NOTATION
Task was performed under a Scientific Services Agreement issued by Battelle, Research Triangle Park Office, 200 Park Drive, P O Box 12297, Research Triangle Park, NC 27709

17. COSATI CODES			18. SUBJECT TERMS (CONTINUE ON REVERSE IF NECESSARY AND IDENTIFY BY BLOCK NUMBER)
FIELD	GROUP	SUB-GROUP	Robotics Automation Dual-arm manipulation

19. ABSTRACT (CONTINUE ON REVERSE IF NECESSARY AND IDENTIFY BY BLOCK NUMBER)

A control algorithm to achieve the dual-arm robotic manipulation of a single object was devised, implemented, and tested. The algorithm uses input from wrist mounted force/torque transducers to sense and correct for any trajectory inconsistencies that may develop during the cooperative manipulation of an object by two PUMA 560 robots. Test results validated the performance of the control system.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (INCLUDE AREA CODE)	22c. OFFICE SYMBOL
I HAZNEDARI	AV 880-3316	SMCAR-IMI-I

DD FORM 1473, 84 MAR

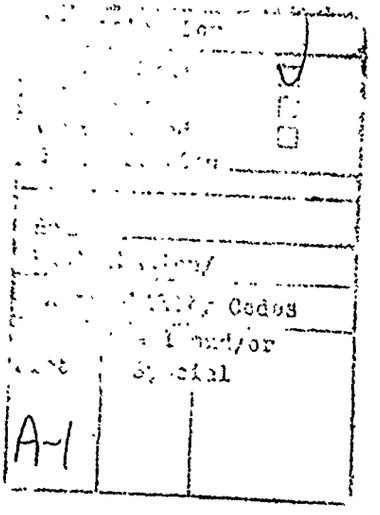
UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

7a. (cont)

ARDEC, FSAC
Fire Control Division (SMCAR-FSF-RC)
Picatinny Arsenal, NJ 07806-5000

CONTENTS

	Page
Introduction	1
Background	2
Technical Discussion	3
Robot Configuration	3
Control Approach	13
Interface to Vision System	17
Summary of Results	18
Conclusions and Recommendations for Future Work	19
References	21
Appendix - Copy of Implementation Software	23
Distribution List	67



I. INTRODUCTION

The objective of this task was to develop and validate control algorithms for the dual-arm robotic manipulation of a single object. This objective has been achieved.

The specific tasks performed to attain the objective were:

1) Prepare control boards suitable for the processing and data handling necessary for the dual-arm robotic manipulation of a single object using wrist mounted force/torque sensors as feedback devices;

2) Devise and implement a means for estimating the weight of the gripped object and eliminating that reaction from the individual force/torque transducer readings;

3) Implement an interface with an external vision system that is to be used to locate the initial position and orientation of the object to be manipulated and, using that input, individually guide each arm to an initial grip point;

4) Implement a technique for removing any initial misalignment existing between either gripper and the gripped object;

5) Implement a control technique to allow the trajectory of one of the cooperating manipulators to adapt to any misalignment error occuring during the dual-arm manipulation of the object;

6) Validate the control algorithm and develop a working demonstration of the control approach; and

7) Document all control software and test results. (KR)

II. BACKGROUND

In some military applications using automation it may be necessary to manipulate a single object using multiple robot arms. This will occur when the weight of the object exceeds the payload of a single arm or when a relatively large component of a weapons system or structure must be positioned with great accuracy.

Svern and Tricamo have published a number of works dealing with two-arm robotic manipulation and control [1]-[9]. The work detailed in this report represents an extension and development of many of the concepts embodied in these publications.

The approach offers a number of advantages, including the following:

- the capability of operating each robot autonomously in the event that loss of grip or slippage occurs at the individual robot grip points on the object,

- the capability of sensing contact between the object and the environment,

- the development of an interface with an external vision system capable of providing the location of the object to be manipulated, and

- the capability being modified to allow for the teach-mode operation of the dual-arm manipulators.

Other authors have also dealt with various aspects of the control of cooperating robots [10]-[14] and similar closed-chain configurations.

III. TECHNICAL DISCUSSION

The control approach required for multi-arm robotic manipulation differs significantly from that used in autonomous operation. When two arms grasp a single object, the resulting structure forms a closed kinematic chain. The number of actuators in this new structure is greater than the minimum needed to position the object. This redundancy can result in the creation of constraint forces within the object and manipulators. Such a condition is brought about when position or orientation errors caused by manipulator miscalibration, grip point slippage, or similar effects cause interaction between the various actuators in each arm. These forces may be of sufficient magnitude to make accurate positioning of the object difficult to achieve.

The approach implemented in the present work to achieve multi-arm cooperative robotic control is based upon the use of force/torque transducers to measure constraint forces in the object and manipulator system brought about by trajectory inconsistencies between individual robots. These constraint reaction measurements are used to determine the magnitude of position and orientation errors existing between the grippers of the cooperating arms. Evaluation of these errors provides the basis for a compensation sheme to control the coordination of the two arms.

A detailed description of the development, implementation, and testing of the control algorithm used to perform the tasks outline in section I of this report is given below.

A. Robot Configuration

1. Kinematic Model

A schematic of the two-arm configuration is shown in Figure III-1. The world coordinate reference frames of each

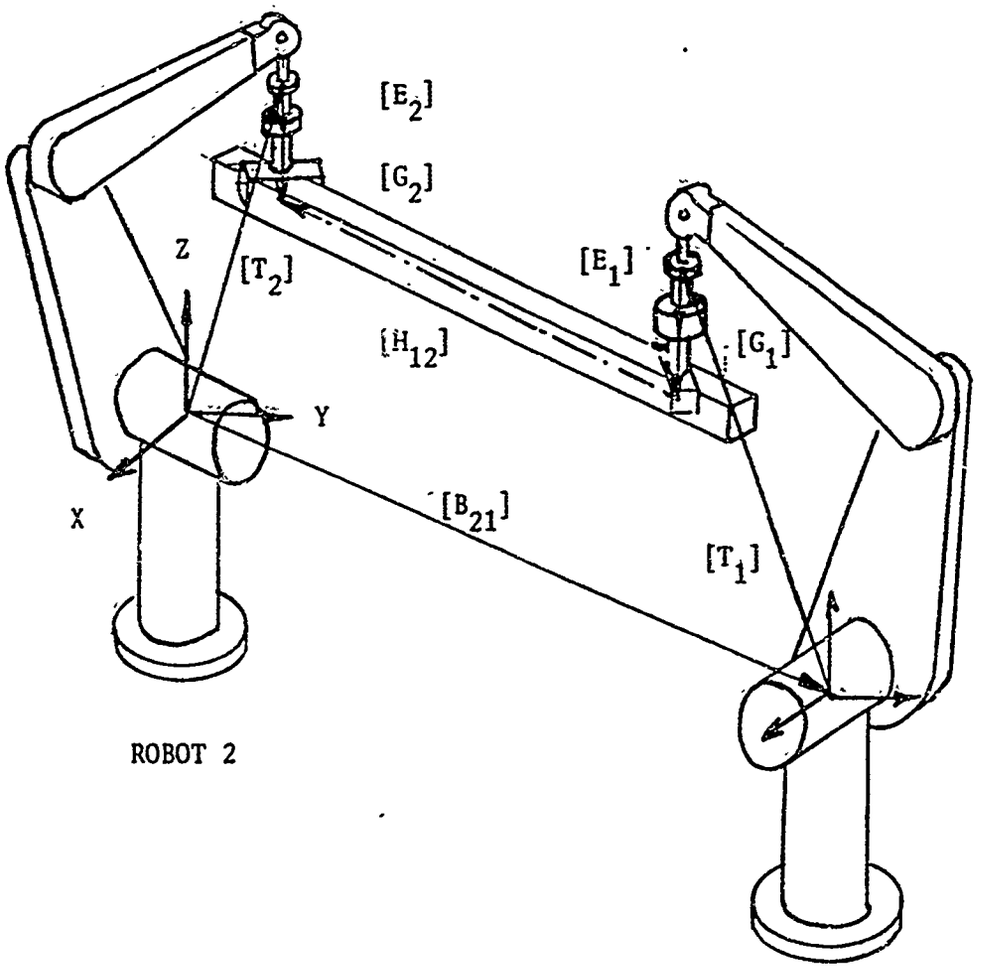


FIGURE III-1: SCHEMATIC OF TWO-ARM CONFIGURATION

robot are aligned with one another and their origins are 36.0 inches apart. The origin of the world reference frame for the muliti-robot configuration is taken as coincident with that of robot 2. The coordinate axes of the wrist mounted force/torque transducers each has a different degree of misalignment with respect to their corresponding end effector frames. This misalignment is compensated for by appropriate transformations, as shown below.

As shown in the figure, the following transformations are defined for each of robots 1 and 2 as denoted by the appropriate subscripts:

$[T_1]$,$[T_2]$ = transformation from base frame to end effector frame

$[E_1]$,$[E_2]$ = transformation from end effector frame to force/torque transducer frame

$[G_1]$,$[G_2]$ = transformation from force/torque transducer frame to gripper frame

$[H_{12}]$ = transformation between frames of grippers 1 and 2, respectively

$[B_{21}]$ = transformation between individual world frames for robots 2 and 1, respectively

Assuming perfect grip, the transformations are related as follows:

$$[B_{21}][T_1][E_1][G_1][H_{12}] = [T_2][E_2][G_2] \qquad (1)$$

where transforms $[B_{21}]$, $[E_1]$, $[G_1]$, $E_2]$, and $[G_2]$ are known and constant. Transforms $[T_1]$ and $[T_2]$ vary as each arm moves. The elements of the latter transforms are available

as output from the VAL II software in each robot controller. Determination of the remaining transform, $[H_{12}]$ is the subject of the next section.

2. Force/Torque Model

Transform $[H_{12}]$ could be found using equation (1) if there were no kinematic errors in either robot and if the object was perfectly located in each gripper. However, neither of these conditions is present and the determination of the relative positioning of each gripper must be made by other means.

This was accomplished using the force/torque sensor readings. Due to the fact that grip in the vertical direction (world z-axis) was loose, leading to inaccurate z-force and x-moment readings, it was decided to manipulate the object in a plane parallel to the X-Y world plane. This simplification in no way limits the applicablility of the devised control technique.

A schematic of the object in grip is shown in Figure III-2. Assuming, for simplification, that the object in grip and the manipulator linkages are rigid, the compliance between the grip surfaces and the object may be modeled as parallel springs, as shown in Figure III-3. The components drawn in solid lines represent the actual positions of the gripper and object while the dashed reference line represents a datum for measuring the actual spring displacements. Only the gripper and object of robot 1 are shown; that of robot 2 would be similar in appearance.

In Figure III-3, the following nomenclature is used:

x_{gi}, x_{pi} = position displacement at spring end of gripper and part (object), respectively, as measured from the undeflected position of the spring
(i=1,2)

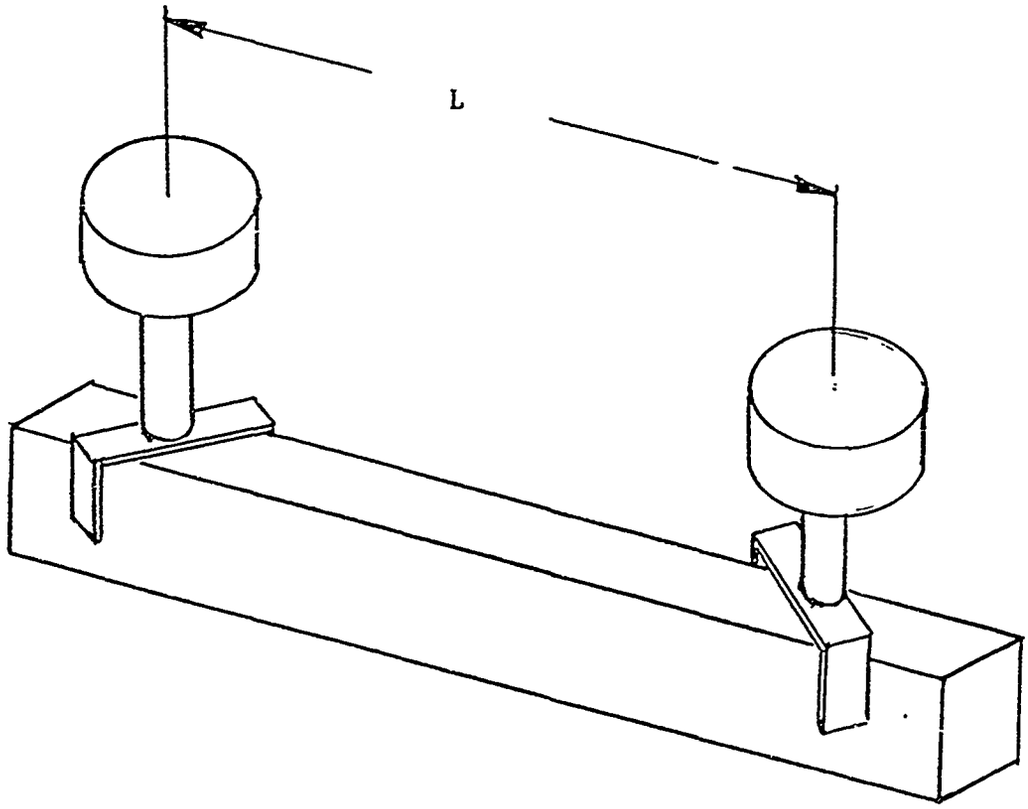


FIGURE III-2: SCHEMATIC OF OBJECT IN GRIP

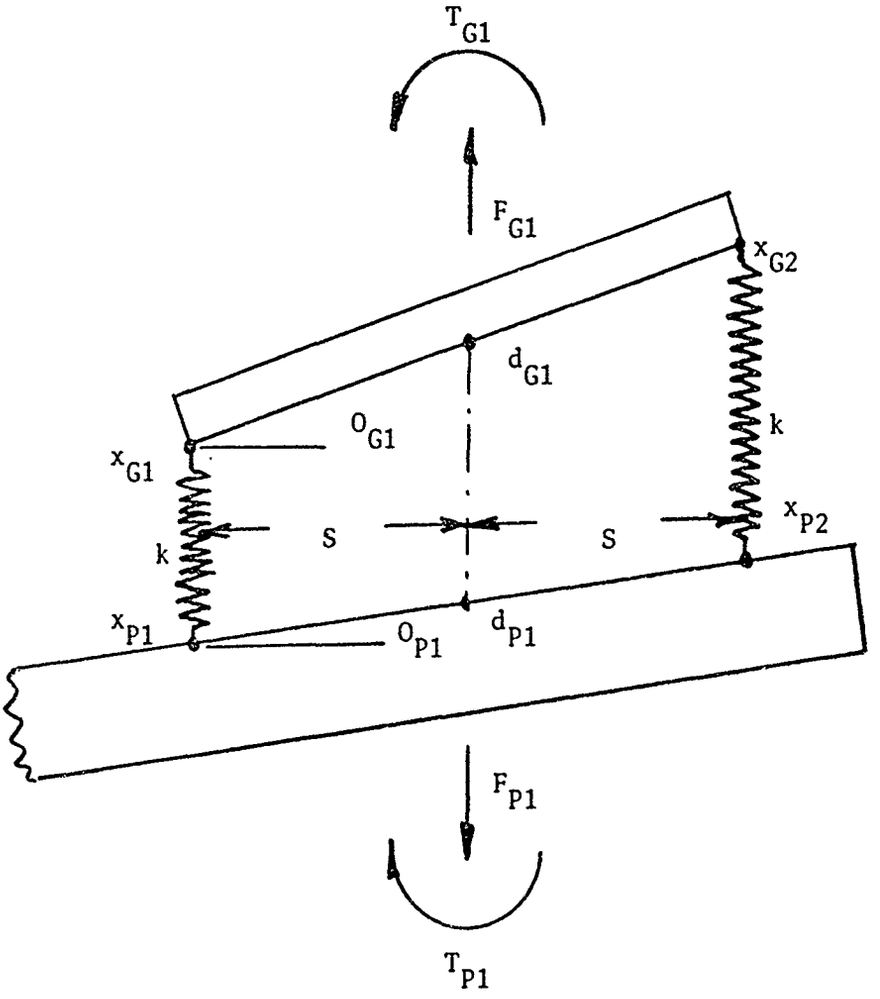


FIGURE III-3: MODELING OF GRIP REACTIONS

Assuming the same spring constant, k, for each contacting surface, the forces developed at the attachment points for each spring are given by:

$$f_{g1} = k(x_{g1} - x_{p1}) \tag{2}$$

$$f_{g2} = k(x_{g2} - x_{p2}) \tag{3}$$

$$f_{p1} = k(x_{p1} - x_{g1}) \tag{4}$$

$$f_{p2} = k(x_{p2} - x_{g2}) \tag{5}$$

The force and torque developed on the gripper are:

$$F_{g1} = f_{g1} + f_{g2} \tag{6}$$

$$T_{g1} = s(f_{g2} - f_{g1}) \tag{7}$$

The position and orientation errors for the gripper corresponding to these forces and torques are given by:

$$d_{g1} = (x_{g1} + x_{g2})/2 \tag{8}$$

$$\theta_{g1} = (x_{g2} - x_{g1})/2s \tag{9}$$

Similar expressions for the force, torque, position error, and orientation error in the object are:

$$F_{p1} = - F_{g1} \tag{10}$$

$$T_{p1} = - T_{g1} \tag{11}$$

$$\begin{aligned} d_{p1} &= (x_{p1} + x_{p2})/2 \\ &= d_{g1} - F_{g1}/2k \end{aligned} \tag{12}$$

$$\begin{aligned} \theta_{p1} &= (x_{p2} - x_{p1})/2s \\ &= \theta_{g1} - T_{g1}/2ks^2 \end{aligned} \tag{13}$$

where equations (12) and (13) were obtained with the help of equations (2),(3),(6),(7),(10, and (11).

Equations (10)-(13) may be expressed in the following matrix form:

$$\begin{bmatrix} F_{p1} \\ T_{p1} \\ d_{p1} \\ \theta_{p1} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ -1/2k & 0 & 1 & 0 \\ 0 & -1/2ks^2 & 0 & 1 \end{bmatrix} \begin{bmatrix} F_{g1} \\ T_{g1} \\ d_{g1} \\ \theta_{g1} \end{bmatrix} \tag{14}$$

In a similar manner, the following matrix expression may be obtained for the gripper of robot 2:

$$\begin{matrix} F_{g2} \\ T_{g2} \end{matrix} = \begin{matrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{matrix} \begin{matrix} F_{p2} \\ T_{p2} \end{matrix} \tag{15}$$

$$\begin{matrix} d_{g2} \\ \theta_{g2} \end{matrix} \quad \begin{matrix} -1/2k & 0 & 1 & 0 \\ 0 & -1/2ks^2 & 0 & 1 \end{matrix} \quad \begin{matrix} d_{p2} \\ \theta_{p2} \end{matrix}$$

For the rigid bar, as shown in Figure III-4, the following matrix expression relating quantities at robot 1 and 2 grip points may be determined:

$$\begin{bmatrix} F_{p2} \\ T_{p2} \\ d_{p2} \\ \theta_{p2} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ -L & -1 & 0 & 0 \\ 0 & 0 & 1 & -L \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} F_{p1} \\ T_{p1} \\ d_{p1} \\ \theta_{p1} \end{bmatrix} \qquad (16)$$

In the above, it should be noted that if a flexible object were gripped equation (16) would have to be modified to include the elastic properties of the object. The overall approach, however, would remain the same.

Combining expressions (14),(15), and (16) results in the following relationship between reactions and displacements for the two grippers:

$$\begin{bmatrix} F_{g2} \\ T_{g2} \\ d_{g2} \\ \theta_{g2} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ -L & -1 & 0 & 0 \\ -1/k & -L/2ks^2 & 1 & -L \\ -L/2ks^2 & -1/ks^2 & 0 & 1 \end{bmatrix} \begin{bmatrix} F_{g1} \\ T_{g1} \\ \alpha_{g1} \\ \theta_{g_1} \end{bmatrix} \qquad (17)$$

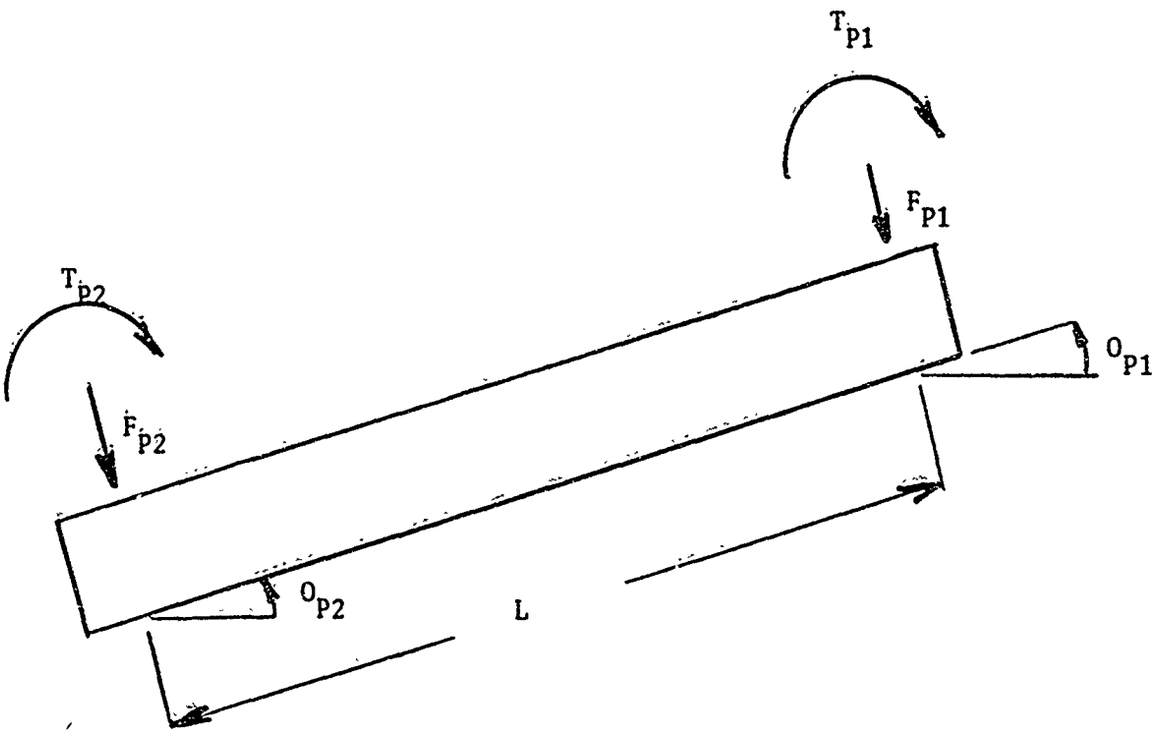


FIGURE III-4: REACTIONS ON RIGID OBJECT

To determine the relative linear and angular displacements of gripper 2 (slave) with respect to gripper 1 (master), let $d_{g1}=\theta_{g1}=0$ in expression (17). Using the first two lines of this same expression, the following is obtained:

$$\begin{bmatrix} d_{g2} \\ \theta_{g2} \end{bmatrix} = 1/2ks^2 \begin{bmatrix} -(L^2-2s^2) & L \\ -L & 2 \end{bmatrix} \begin{bmatrix} F_{g2} \\ T_{g2} \end{bmatrix} \quad (18)$$

Expression (18) then gives a relationship between the sensed force and torque reactions at the slave gripper due to the misalignment between the grippers and the corresponding linear and angular displacements required to eliminate that misalignment. As such, it represents the basis for the control approach. By repeated sampling of the force/torque reactions, corrections as defined by expression (18) are made to the location and orientation of the slave gripper. Previous work has shown (see reference [7]) that even in the presence of noise and other errors in the sensed reactions or physical parameters the control approach is stable and will result in eliminating the misalignment over relatively few iterations.

B. Control Approach

1. Hardware Configuration

The force/torque readings were obtained using two LORD Model 15/50 transducers. This device has a parallel port available that transmits three force and three torque readings as a single group at rates up to 100 groups per second. The transducer also has a language that can be used to perform simple thresholding operations on its readings, and to logically combine these discrete outputs.

Referring to Figure III-5, the central component of the control system is an INTEL 8614 single board processor containing an 8086 processor. This processor is used as a discrete controller with an iteration time of 28 msec to match that of the PUMA 560. The controlling INTEL 8614 processor is connected to a second 8614 board, serving as an I/O controller, via a Multi-Bus I. The I/O controller contains sufficient parallel and serial ports to interface both force/torque sensors as well as the VAL II ALTER interface.

In the development of the control strategy, it was noted that the iteration time of the controller is fixed by the availability of force/torque data from the transducers and the ability of the VAL processor to accept and execute movement commands. Figure III-6 shows the iteration time of the various loops involved, including the loops internal to the robot controllers.

In testing the system, it was further noted that performance was limited by the processing capability of the 8086 processor.

2. Gravity Compensation

An advantage gained in the use of two force/torque transducers was the ability to easily measure external reactions acting on the manipulated object by simply summing the corresponding readings of each platform. In the absence of external reactions, the combined readings on all axes would be zero, indicating the presence of only constraint reactions. Finite sums indicate the presence of an external force or torque.

The only external reaction dealt with in the present application was that due to gravity. The algorithm calculates the weight of the object as indicated above. Care must be taken in evaluating this quantity to be certain that

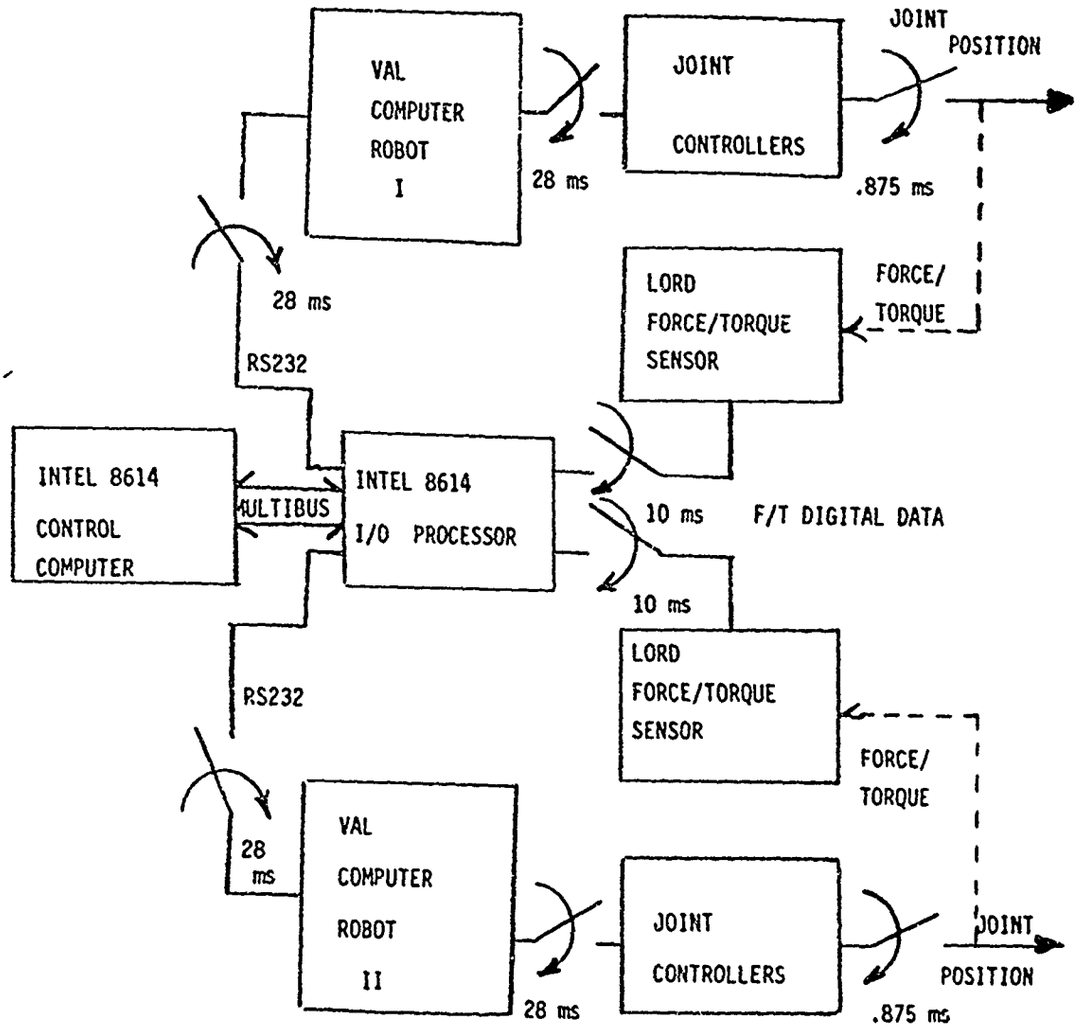


FIGURE III-5: HARDWARE INTERCONNECTION DIAGRAM

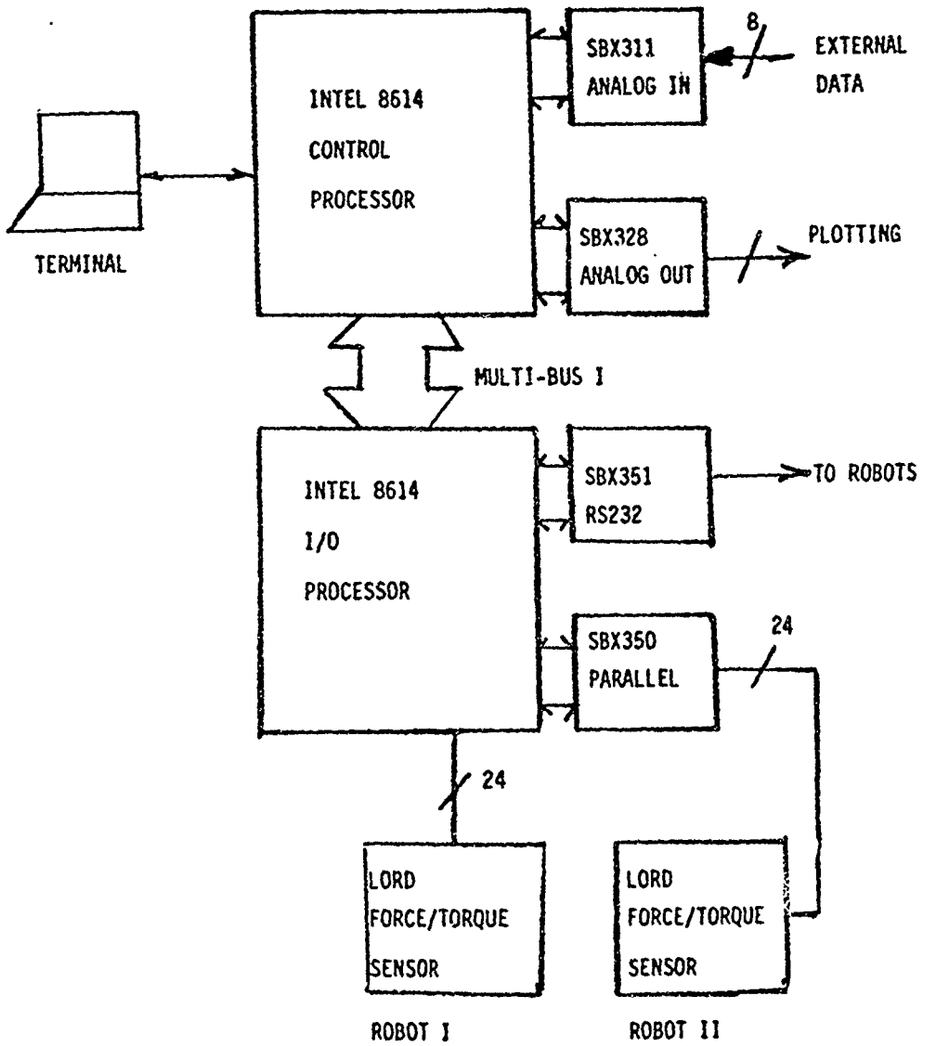


FIGURE III-6: HARDWARE TIMING DIAGRAM

all readings are transformed to a common reference frame before summation is undertaken. After the weight measurement is taken, it is assumed that the weight is equally distributed between the two grippers and an appropriate correction is made to individual sensor readings to bias off this gravity effect, thereby leaving only constraint reactions readings for use in correcting misalignments.

3. Rate Filter

To avoid sudden movements due to erroneous force/torque readings due to noise or other effects, a rate filter was employed to condition the input data. This was accomplished by averaging the sensor readings on each channel over 8 samples as the iteration scheme progressed. This technique worked well, but it did result in some degradation of performance due to the slower response inherent with this approach.

C. Interface to Vision System

The control algorithm provides a means of using an external vision system to locate and transmit the initial grip point locations and orientations for the two manipulators.

If both robots have not reached their grip point, then the controller sends back position corrections only based on vision system input. After both robots reach their grip point, both navigational and constraint based feedback may occur. Coordinate changes due to vision system input must be removed for the robot to reach the target location. After navigation is complete, force feedback is implemented.

IV. SUMMARY OF RESULTS

The control algorithms were validated using the following sequence of operations for the dual-arm configuration:

1. Each robot arm moves to a preset location under autonomous operation

2. A target grip point is computed for each manipulator using input data from the vision system

3. Each arm is commanded to move to the target grip point

4. Each gripper closes to grasp the object

5. A path is computed back to the positions taught in step 1.

6. Both arms move to target position as force/torque algorithm removes any misalignment in grippers

7. With both grippers now aligned, coordinated movement to new location is accomplished under force/torque control

The test sequence was successfully executed many times using objects of differing weight and stiffness. The control algorithms functioned as intended in each case.

V. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

A control algorithm and corresponding hardware to accomplish the dual-arm robotic manipulation of a single object was devised, implemented, and successfully tested. The algorithm uses input from wrist mounted force/torque transducers to sense and correct for any misalignment or trajectory inconsistencies that may develop during the cooperative manipulation of the test object. The control algorithm is capable of taking in data from an external vision system to guide each robot arm to its initial grip position.

The existing system represents one of the few dual-arm robotic test facilities in the nation. To further improve and extend its performance capabilities, the following suggestions are made for future work:

- upgrade the processor in the control system to a 386

- interface the control system to Matrix AR

- develop a dual-arm teach mode capability

- evaluate the possibility of attaining significant payload increases using the inherent strength of the dual-arm closed-chain configuration

- incorporate a dual-arm control technique capable of using external force signals to accomplish a given objective (assembling parts, for example)

- evaluate the advantage and feasibility of multi-arm manipulation using more than two manipulators.

REFERENCES

1. Swern, Tricamo, and Lin, "An Approach to the Control of Multirobot Manipulation", Proc. of 1988 ASME Mechanisms Conference, Orlando, Florida.

2. Tricamo and Swern, "Insertion with Two Coordinated Arms", 1988 NASA/DoD Conference on Space and Military Applications of Automation and Robotics, June 1988, Huntsville, Ala.

3. Swern and Tricamo, "Experimental Results in the Implementation of Control Algorithms for the Two-Arm Robotic Manipulation of a Single Object", AIAA Space Programs and Technologies Conf., Houston, Tex, June 1988

4. Tricamo and Swern, "Control of Multiple Robotic Arms Engaged in Cooperative Manipulation and Assembly Operation", 1988 IEEE International CIM Conf., RPI, Troy, NY, 1988

5. Coleman, Swern, and Tricamo, "Modeling and Simulation of Two-Arm Robots", Proc. of 19th Annual Pittsburgh Conf. on Modeling and Simulation, 1988

6. Swern and Tricamo, "An Approach to Controlling Multi-Arm Robotic Manipulation of a Single Body" , Proc. of 1988 IEEE Robotics and Automation Conf., Phila., PA, 1988

7. Swern and Tricamo, "Final Report: Formulation and Evaluation of Control Algorithms for Two-Arm Robotic Manipulation", for ARDEC, Jan. 30, 1989

8. Tricamo and Swern, "Final Report: Advanced Algorithm Development Modeling and Analysis for Armament/Robotics Applications", for ARDEC, March 1, 1988

9. Swern and Tricamo, "Final Report: Implementation of a Conrol Algorithm for a Robot Manipulator", for ARDEC, Jan. 1987

10: Lim and Chyung; "A Control Scheme for Two Cooperating Robot Arms"; IEEE Control Systems Magazine; Feb. 1987; pp. 65-68

11: Alford and Belyeu; "Coordinated Control of Two Robot Arms"; Proc. IEEE Int'l Conf. on Robotics; 1984, pp. 468-473

12: Hemami; "Kinematics of Two-Arm Robots"; IEEE Journal of Robotics and Automation; vol. RA-2; Dec. 1986, pp. 225-228

13: Ishida; "Force Control in Coordination of Two Arms", Proc. of 5th Int'l Joint Conf. on Artificial Intelligence, Aug. 1977; pp. 717-722

14: Hartenburg and Denavit; "Kinematic Synthesis of Linkages"; McGraw-Hill; 1964.

APPENDIX

COPY OF IMPLEMENTATION SOFTWARE

```
FORTRAN-86 COMPILER
 CONTRL.FOR

DOS 3.30 (038-N) FORTRAN-86 COMPILER V3.0
COMPILER INVOKED BY: C:\INTEL\FORT86.EXE  CONTRL.FOR

   1          BLOCK DATA
      C
      C  NAME:    Block Data
      C
      C  PURPOSE:  Initialize Common Blocks
      C
      C  DESCRIPITON:  This module contains values of constants used in other
      C       FORTRAN modules in the program:
      C
      C
      C   Main Common Variables are set to zero to insure that the controller
      C          starts from some known state.
      C
      $INCLUDE (FTORQUE.INC)
    =1 $NOLIST
    2=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
    3=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
    4=1         INTEGER MVSKED,MVBAK,MVRAK
    5=1         CHARACTER*40 PROMPT
    6=1         CHARACTER*3 DISPT
    7=1         LOGICAL*1 RESET,FTPLOT
    =1 C
    8=1         PARAMETER (R2MSG=2,R1MSG=1)
    9=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
   10=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
   11=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
   12=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
   13=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
   14=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
   15=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1       1  FTMATX(6,6),RSCALE(6)
   16=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
   17=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
   18=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
   19=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
   20=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
   21=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
    =1       1  RDMAX,RTMAX,NCALC
      C
   22           DATA FORCE,FTBIAS,FTDIF/42*0.0/
   23           DATA RSCALE/32.0,32.0,32.0,182.0444,182.0444,182.0444/
   24           DATA FTMFLAG,FTMFLAG2/2*0/
   25           DATA FTMAT2/36*0.0/
   26           DATA PROMPT/'ENTER COMMAND'/
      C
   27           END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

  1          SUBROUTINE CINIT(I,IN)
     C
     C  NAME:  Constant Initialization
     C
     C  PURPOSE:  Initialize Constants in the FORTRAN programs
     C
     C  DESCRIPTION:  This program allows certain constants to be set in memor
     C       and then moved over to be used by the FORTRAN subroutines.  It is
     C       easier to change these constants, using the debug monitor, on the
     C       fly rather than recompile the FORTRAN program.
     C
  2          INTEGER*2 IN,I
     $INCLUDE (FTORQUE.INC)
   =1 $NOLIST
  3=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
  4=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
  5=1         INTEGER MVSKED,MVBAK,MVRAK
  6=1         CHARACTER*40 PROMPT
  7=1         CHARACTER*3 DISPT
  8=1         LOGICAL*1 RESET,FTPLOT
   =1 C
  9=1         PARAMETER (R2MSG=2,R1MSG=1)
 10=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
 11=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
 12=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
 13=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
   =1 C
 14=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
 15=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
 16=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
   =1       1  FTMATX(6,6),RSCALE(6)
 17=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
   =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
 18=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
 19=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
 20=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
 21=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 22=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
   =1       1  RDMAX,RTMAX,NCALC
 23           IOUTER=0
     C
     C   This call sets the words "REINIT" on the monitor screen to verify tha
     C     the RS232 interface to that screen is operational
     C
 24           CALL DISPCH('REINIT')
     C
     C   To set the Debug print flag
 25           IF(I.EQ.3) FTPFLAG=IN
     C
     C   Scaling constants for the force/torque algorithms
     C
 26           FSCALE=1.0
 27           TSCALE=1.0
     C
     C   Set force/torque matrix gains
     C
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 28              XGAIN=.0075
 29              YGAIN=.0004
 30              ZGAIN=.0030
 31              ALENTH=11.0
 32              CALL GAINM
          C
          C   Rate Constants used for the algorithms
          C
 33              DPHIM = .02
 34              RDMAX = 5.0
 35              RTMAX = 2.0
          C
          C  Set the initialization flag and other flags
          C
 36              FTMFLAG=0
 37              FTMFLAG2=0
 38              NCALC=0
          C
          C  Clear arrays and data areas to prevent startup transients
          C
 39              DO 10 J=1,6
 40              ROBPT1(J)=0.0
 41              ROBPT2(J)=0.0
 42              ROBWDIF1(J)=0.0
 43              ROBWDIF2(J)=0.0
 44              ROBDIF1(J)=0
 45    10        ROBDIF2(J)=0
          C
          C  Reset grip angle to zero
          C
 46              PHIZ2=0.0
 47              CALL TOOLT2(0.0,0.0,-1)
          C
          C  Set base offset of two robots
          C
 48              BX=-36.0
 49              BY=0.0
 50              BZ=0.0
 51              RETURN
 52              END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

  1          SUBROUTINE CKFTF(I,IFTIN)
     C
     C  NAME:  Check Force/Torque Data
     C
     C  PURPOSE:  Preprocess data from each axis of Force/Torque Sensor
     C
     C  DESCRIPTION:  The raw Force/Torque data is corrected for sensed
     C       gravity.
     C
     C  INPUTS:  I     -  Force/Torque sensor axis identification number
     C           IFTIN -  Force/Torque sensor reading
     C
  2          INTEGER*2 I,IFTIN,FTID(14)
  3          REAL VFIL,DOBIAS
  4          DATA FTID/1,2,3,3*4,0,1,5,6,3*4,0/
     C
     $INCLUDE (FTORQUE.INC)
   =1 $NOLIST
  5=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
  6=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
  7=1         INTEGER MVSKED,MVBAK,MVRAK
  8=1         CHARACTER*40 PROMPT
  9=1         CHARACTER*3 DISPT
 10=1         LOGICAL*1 RESET,FTPLOT
   =1 C
 11=1         PARAMETER (R2MSG=2,R1MSG=1)
 12=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
 13=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
 14=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
 15=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
   =1 C
 16=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
 17=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
 18=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
   =1       1  FTMATX(6,6),RSCALE(6)
 19=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
   =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
 20=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
 21=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
 22=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
 23=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 24=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
   =1       1  RDMAX,RTMAX,NCALC
     C
 25           JFTIN=IFTIN
 26           FRIN=JFTIN
 27           GOTO (1,2,3,4,6,7),FTID(I)
     C
 28    1      FORCE(I)=FRIN
 29           GOTO 5
     C
 30    2      FORCE(I)=FRIN+TDZ*FORCE(I+4)
 31           GOTO 5
     C
 32    3      FORCE(I)=FRIN-TDZ*FORCE(I+2)
 33           GOTO 5
```

```
FORTRAN-86 COMPILER
 CCNTRL.FOR

      C
 34   4       FORCE(I)=FRIN
 35           GOTO 5
      C
 36   6       FORCE(I)=FRIN-TDZ*FORCE(I+4)
 37           GOTO 5
      C
 38   7       FORCE(I)=FRIN+TDZ*FORCE(I+2)
 39           GOTO 5
      C
      C   Calculate the Gravity compensated force (FTDIF)
      C
 40   5       FTDIF(I)=FORCE(I)-FTBIAS(I)
      C
 41           RETURN
 42           END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

1           SUBROUTINE CONTRL(DATAFLG,SYSFLAG,ROB1DAT,ROB1CMD,ROB2DAT,ROB2CMD,
          1     FTSAMP1,FTSAMP2)
      C
      C  NAME:  Control
      C
      C  PURPOSE:  Performs high rate (inner loop) Force/Torque and positioning
      C            calculations to interface the VAL alter command
      C
      C  DESCRIPTION:  A new set of Force/Torque readings is processed.  If
      C       the robots have started, but are not at their grip points,
      C       guidance is supplied to the new grip point.  After the grip point
      C       is reached, the Force/Torque matrix multiplies each force reading
      C       to obtain a new set of displacements.
      C
      C  INPUTS:  DATAFLG    -  Flags showing which data is valid on input, out
      C           SYSFLAG    -  Flags showing the state of the system
      C           ROB1DAT    -  Input data from robot 1 including positional tr
      C           ROB2DAT    -  Input data from robot 2 including positional tr
      C           FTSAMP1    -  Two Force/Torque samples from the wrist of robo
      C           FTSAMP2    -  Two Force/Torque samples from the wrist of robo
      C
      C  OUTPUTS; ROB1CMD    -  New positional delta for robot 1
      C           ROB2CMD    -  New positional delta for robot 2
      C
2           INTEGER*2 ROB1DAT(15),ROB2DAT(15),ROB1CMD(7),ROB2CMD(7),
          1     FTSAMP1(36),FTSAMP2(36),JDFLAG
3           INTEGER*1 DATAFLG,SYSFLAG,IDFLAG
4           EQUIVALENCE (IDFLAG,JDFLAG)
      C
      $INCLUDE (FTORQUE.INC)
  =1  $NOLIST
 5=1          INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
 6=1          INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
 7=1          INTEGER MVSKED,MVBAK,MVRAK
 8=1          CHARACTER*40 PROMPT
 9=1          CHARACTER*3 DISPT
10=1          LOGICAL*1 RESET,FTPLOT
  =1  C
11=1          PARAMETER (R2MSG=2,R1MSG=1)
12=1          PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
13=1          PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
14=1          PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
15=1          PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
  =1  C
16=1          COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
17=1          COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
18=1          COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
  =1        1  FTMATX(6,6),RSCALE(6)
19=1          COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
  =1        4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
20=1          COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
21=1          COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
22=1          COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
23=1          COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
24=1          COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
  =1        1  RDMAX,RTMAX,NCALC
```

```
FORTRAN-86 COMPILER
 CCNTRL.FOR

      C
      C
      C   Translate force/torque variables to floating point
      C
 25           DO 10 I=6,1,-1
 26           CALL CKFTF(I,FTSAMP1(I+1))
 27           CALL CKFTF(I+7,FTSAMP2(I+1))
 28   10      CONTINUE
      C
      C
 29           IF(IAND(IFLAG,STARTD).EQ.STARTD) THEN
 30              IDFLAG=DATAFLG
      C
      C   Translation algorithms to new rack location
      C
 31              IF (IAND(FTMFLAG,FTGRIP).EQ.0) THEN
      C
      C   If both robots have not reached the grip point, then
      C      the controller sends back position corrections only
      C      derived from camera input
      C
 32                 IF(IAND(FTMFLAG,FCALC).NE.0) THEN
 33                    NCALC=NCALC+1
      C
      C   Robot Position Commands
      C
 34                    ROB1CMD(1)=256*63
 35                    ROB2CMD(1)=256*63
 36                    DO 20 I=1,6
 37                       ROB1CMD(I+1)=ROBDIF1(I)
 38   20                  ROB2CMD(I+1)=ROBDIF2(I)
      C
 39                    JDFLAG=IOR(JDFLAG,R2MSG)
 40                    JDFLAG=IOR(JDFLAG,R1MSG)
 41                    ENDIF
      C
 42                 ELSE
      C
      C   Force/torque Control Algorithms
      C
      C   After both robots reach grip point, both navigational and force
      C      feedback may occur.  Coordinate changes due to camera input must
      C      be removed for robot to reach target location. After navigation
      C      is complete, force feedback is implemented.
      C
 43                    ROB2CMD(1)=256*63
 44                    ROB1CMD(1)=256*63
 45                    IF(IAND(FTMFLAG,FCALC).NE.0) THEN
      C        Do navigational movement
 46                       NCALC=NCALC+1
 47                       DO 30 I=1,6
 48                          ROB1CMD(I+1)=ROBDIF1(I)
 49                          ROB2CMD(I+1)=ROBDIF2(I)
 50   30                     CONTINUE
 51                       ELSE
      C        Do force/torque feedback
 52                       DO 32 I=1,6
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

  53                        XSUM=0.0
  54                        IF (IAND(FTMFLAG,FTDOIT).NE.0) THEN
  55                           DO 35 J=1,6
  56   35                         XSUM=XSUM+FTDIF(J+7)*FTMATX(I,J)
  57                           ENDIF
  58                        ROB2CMD(I+1)=XSUM*RSCALE(I)
  59                        ROB1CMD(I+1)=0
  60   32                   CONTINUE
  61                     ENDIF
       C
       C   Set robot messages are available
       C
  62                  JDFLAG=IOR(JDFLAG,R1MSG)
  63                  JDFLAG=IOR(JDFLAG,R2MSG)
  64                  ENDIF
  65               DATAFLG=IDFLAG
       C
  66               ENDIF
       C
  67            RETURN
  68            END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

  1          SUBROUTINE OUTER(FORTFLAG,SYSFLAG,ROB1DAT,ROB1CMD,ROB2DAT,ROB2CMD
           1     ,FTSAMP1,FTSAMP2)
      C
      C  NAME:  Outer Control loops
      C
      C  PURPOSE:  Perform low rate control calculations including gravity
      C         compensation and navigation
      C
      C  DESCRIPTION:  This routine updates system flags in common and clears
      C         all flags, etc. when the task is complete.  It also calls
      C         gravity compensation subroutine and navigation subroutine to
      C         update (slow moving) quantities for the next group of iteration
      C
      C
      C  INPUTS:  FORTFLAG   -  Flags summarizing the system state
      C           SYSFLAG    -  Flags showing the state of each robot in the sy
      C           ROB1DAT    -  Input data from robot 1 including positional tr
      C           ROB2DAT    -  Input data from robot 2 including positional tr
      C           FTSAMP1    -  Two Force/Torque samples from the wrist of robo
      C           FTSAMP2    -  Two Force/Torque samples from the wrist of robo
      C
      C  OUTPUTS: ROB1CMD    -  Robot 1 commands zeroed when robot 1 is done
      C           ROB2CMD    -  Robot 2 commands zeroed when robot 2 is done
      C
      C
  2          INTEGER*2 ROB1DAT(15),ROB2DAT(15),ROB1CMD(7),ROB2CMD(7),
           1     FTSAMP1(36),FTSAMP2(36),JDFLAG,JFFLAG
  3          INTEGER SEG,MOTION,T11,T21,T31,T12,T22,T32,T13,T23,T33,T14,
           1     T24,T34
  4          PARAMETER(SEG=2,MOTION=3,T11=4,T21=5,T31=6,T12=7,T22=8,
           1     T32=9,T13=10,T23=11,T33=12,T14=13,T24=14,T34=15)
  5          INTEGER*1 DATAFLG,SYSFLAG,IDFLAG,IFFLAG,FORTFLAG(2)
  6          EQUIVALENCE (IDFLAG,JDFLAG),(IFFLAG,JFFLAG)
      C
      $INCLUDE (FTORQUE.INC)
    =1 $NOLIST
   7=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
   8=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
   9=1         INTEGER MVSKED,MVBAK,MVRAK
  10=1         CHARACTER*40 PROMPT
  11=1         CHARACTER*3 DISPT
  12=1         LOGICAL*1 RESET,FTPLOT
    =1 C
  13=1         PARAMETER (R2MSG=2,R1MSG=1)
  14=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
  15=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
  16=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
  17=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
  18=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
  19=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
  20=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1       1  FTMATX(6,6),RSCALE(6)
  21=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
  22=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 23=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
 24=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
 25=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 26=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
   =1       1  RDMAX,RTMAX,NCALC
      C
 27           IOUTER=IOUTER+1
      C
      C   Housekeeping for flags and variables
      C
 28           JDFLAG=0
 29           JFFLAG=0
 30           IDFLAG=SYSFLAG
 31           IFFLAG=FORTFLAG(1)
 32           IFLAG=JDFLAG
 33           FTMFLAG=IAND(FTMFLAG,240)+IAND(JFFLAG,15)
      C
      C   Reset everything when robot has dropped START
      C
 34           IF(IAND(FTMFLAG,FSTART).NE.FSTART) THEN
 35              ROB1CMD(1)=0
 36              ROB2CMD(1)=0
 37              DO 10 I=1,6
 38                 ROB1CMD(I+1)=0
 39                 ROB2CMD(I+1)=0
 40   10            CONTINUE
 41              NCALC=0
 42              FTMFLAG=0
 43              FTMFLAG2=0
 44              ELSE
      C
      C  Call Gravity Compensation and Make Navigational Calculations
      C
 45              CALL GRAV(ROB1DAT,ROB2DAT)
      C
      C  Rack coordinates supplied by the camera are represented as increments
      C      rack coordinates the robot is taught.  To properly compensate,
      C      navigation is done by the following steps:
      C        1)  Move to grip point robot was taught
      C        2)  Compute true grip point from camera increments
      C        3)  Move to true grip point
      C        4)  Grasp object
      C        5)  Compute path to taught grip point
      C        6)  Move back to taught grip point
      C        7)  Continue force/torque algorithms
      C
 46              IF (IAND(IFLAG,GRIPPD).EQ.GRIPPD) THEN
 47                 IF (IAND(FTMFLAG,FTGRIP).EQ.0) THEN
      C   At grip point, but not gripped
 48                    IF(IAND(FTMFLAG2,MVSKED).NE.0) THEN
      C   Then compute true grip point, set flags to do only once
 49                       CALL RAKMOV(1)
 50                       FTMFLAG2=IAND(FTMFLAG2,255-MVSKED)
 51                       FTMFLAG2=IOR(FTMFLAG2,MVBAK)
 52                       FTMFLAG=IOR(FTMFLAG,NAVSW)
 53                       ENDIF
 54                    ENDIF
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 55                  ELSE
      C   Object gripped
 56                  IF (IAND(FTMFLAG2,MVBAK).NE.0) THEN
      C   Compute path back to taught grip point, reset flags
 57                      CALL RAKMOV(2)
 58                      FTMFLAG2=IAND(FTMFLAG2,255-MVBAK)
 59                      FTMFLAG=IOR(FTMFLAG,NAVSW)
 60                      ENDIF
 61                  ENDIF
      C
      C   Compute movement increments for current path
      C
 62               CALL NAV(ROB1DAT,ROB2DAT)
 63               ENDIF
      C
      C  Return synchronization flag to main process
      C
 64            JFFLAG=IAND(FTMFLAG,240)
 65            FORTFLAG(2)=IFFLAG
      C
 66            RETURN
 67            END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


  1          SUBROUTINE GRAV(ROB1DAT,ROB2DAT)
     C
     C
     C  NAME:  Gravity Compensation
     C
     C  PURPOSE:  Compute external forces on the object and set FTBIAS to refl
     C      these external forces
     C
     C  DESCRIPTION:  The transformation matrix supplied by VAL is translated
     C      real numbers.  A vector between the grippers is also calculated. A
     C      dimensional vector representing the direction from robot 2 to robo
     C      in robot 2 tool coordinates is calculated.  Forces are then transl
     C      to the world coordinate system, and summed.  It is assumed that an
     C      forces after summation is due to external forces, and these forces
     C      load shared between the two robots by setting FTBIAS.
     C
     C  INPUTS:  ROB1DAT  -  Robot 1 transformation data supplied by VAL
     C           ROB2DAT  -  Robot 2 transfomration data supplied by VAL
     C
     C
  2          INTEGER*2 ROB1DAT(15),ROB2DAT(15)
  3          INTEGER SEG,MOTION,T11,T21,T31,T12,T22,T32,T13,T23,T33,T14,
           1     T24,T34
  4          PARAMETER(SEG=2,MOTION=3,T11=4,T21=5,T31=6,T12=7,T22=8,
           1     T32=9,T13=10,T23=11,T33=12,T14=13,T24=14,T34=15)
     C
     $INCLUDE (FTORQUE.INC)
    =1 $NOLIST
   5=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
   6=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
   7=1         INTEGER MVSKED,MVBAK,MVRAK
   8=1         CHARACTER*40 PROMPT
   9=1         CHARACTER*3 DISPT
  10=1         LOGICAL*1 RESET,FTPLOT
    =1 C
  11=1         PARAMETER (R2MSG=2,R1MSG=1)
  12=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
  13=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
  14=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
  15=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
  16=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
  17=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
  18=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1       1  FTMATX(6,6),RSCALE(6)
  19=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
  20=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
  21=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
  22=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
  23=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
  24=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
    =1       1  RDMAX,RTMAX,NCALC
     C
     C   Calculate transformations and position vectors from VAL inputs
     C
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

      C       Compute Robot positions
      C
   25         RX1=REAL(ROB1DAT(T14))/32.0
   26         RY1=REAL(ROB1DAT(T24))/32.0
   27         RZ1=REAL(ROB1DAT(T34))/32.0
   28         RX2=REAL(ROB2DAT(T14))/32.0
   29         RY2=REAL(ROB2DAT(T24))/32.0
   30         RZ2=REAL(ROB2DAT(T34))/32.0
      C
      C       Compute distances between grippers
      C
   31         DX=BX+(RX1-RX2)/25.4
   32         DY=BY+(RY1-RY2)/25.4
   33         DZ=BZ+(RZ1-RZ2)/25.4
   34         TDZ=6.0
      C
      C       Directional cosines
      C
   35         CTTOW1(1,1)=REAL(ROB1DAT(4))/16384.0
   36         CTTOW1(2,1)=REAL(ROB1DAT(5))/16384.0
   37         CTTOW1(3,1)=REAL(ROB1DAT(6))/16384.0
   38         CTTOW1(1,2)=REAL(ROB1DAT(7))/16384.0
   39         CTTOW1(2,2)=REAL(ROB1DAT(8))/16384.0
   40         CTTOW1(3,2)=REAL(ROB1DAT(9))/16384.0
   41         CTTOW1(1,3)=REAL(ROB1DAT(10))/16384.0
   42         CTTOW1(2,3)=REAL(ROB1DAT(11))/16384.0
   43         CTTOW1(3,3)=REAL(ROB1DAT(12))/16384.0
      C
   44         CTTOW2(1,1)=REAL(ROB2DAT(4))/16384.0
   45         CTTOW2(2,1)=REAL(ROB2DAT(5))/16384.0
   46         CTTOW2(3,1)=REAL(ROB2DAT(6))/16384.0
   47         CTTOW2(1,2)=REAL(ROB2DAT(7))/16384.0
   48         CTTOW2(2,2)=REAL(ROB2DAT(8))/16384.0
   49         CTTOW2(3,2)=REAL(ROB2DAT(9))/16384.0
   50         CTTOW2(1,3)=REAL(ROB2DAT(10))/16384.0
   51         CTTOW2(2,3)=REAL(ROB2DAT(11))/16384.0
   52         CTTOW2(3,3)=REAL(ROB2DAT(12))/16384.0
      C
      C       Direction to  Robot 1 in Robot 2 Coordinates
      C
   53         DX2=CTTOW2(1,1)*DX+CTTOW2(2,1)*DY+CTTOW2(3,1)*DZ
   54         DY2=CTTOW2(1,2)*DX+CTTOW2(2,2)*DY+CTTOW2(3,2)*DZ
      C
      C   Realign the control Matrix
      C
   55         CALL TOOLT2(DX2,DY2,0)
      C
      C   Force/Torque summation calculations
      C
      C
      C       Translate Forces and Torques to world coordinates
      C
   56         DO 5 I=1,3
   57         FTOOL1(I)=0.0
   58         MTOOL1(I)=0.0
   59         FTOOL2(I)=0.0
   60         MTOOL2(I)=0.0
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 61            DO 5 J=1,3
 62            FTOOL1(I)=FTOOL1(I)+CTTOW1(I,J)*FORCE(7-J)
 63            MTOOL1(I)=MTOOL1(I)+CTTOW1(I,J)*FORCE(4-J)
 64            FTOOL2(I)=FTOOL2(I)+CTTOW2(I,J)*FORCE(14-J)
 65            MTOOL2(I)=MTOOL2(I)+CTTOW2(I,J)*FORCE(11-J)
 66     5      CONTINUE
        C
        C      Sum Forces and Torques over both platforms
        C
 67            DO 15 I=1,3
 68            F1ZSUM(I)=MTOOL1(4-I)
 69            F1ZSUM(I+3)=FTOOL1(4-I)
 70            F2ZSUM(I)=MTOOL2(4-I)
 71            F2ZSUM(I+3)=FTOOL2(4-I)
 72            FZSUM(I)=F1ZSUM(I)-F2ZSUM(I)
 73     15     FZSUM(I+3)=F1ZSUM(I+3)+F2ZSUM(I+3)
        C
        C      Moment compensation for forces at the other platform
        C        (Weight component is ignored in this compensation)
        C
        C   Mz = Mz + Fx*Dy + Fy*Dx
 74            FZSUM(1)=FZSUM(1)+F1ZSUM(6)*DY-F1ZSUM(5)*DX
        C   My = My + Fx*Dz + Fz*Dx
 75            FZSUM(2)=FZSUM(2)-F1ZSUM(6)*DZ+(F1ZSUM(4)-.5*FZSUM(4))*DX
        C   Mx = Mx + Fy*Dz + Fz*Dy
 76            FZSUM(3)=FZSUM(3)+F1ZSUM(5)*DZ-(F1ZSUM(4)-.5*FZSUM(4))*DY
        C
        C      Transform the Force sum to Tool Coordinates of Robot 2
        C
 77            DO 25 I=1,3
 78            FSUM(I)=0.0
 79            FSUM(I+3)=0.0
 80            DO 25 J=1,3
 81            FSUM(I)=FSUM(I)-FZSUM(J)*CTTOW2(4-J,4-I)
 82            FSUM(I+3)=FSUM(I+3)+FZSUM(J+3)*CTTOW2(4-J,4-I)
 83     25     CONTINUE
        C
        C      Calculate load balancing biases
        C
 84            DO 20 I=1,6
        C       FTBIAS(I)=-FSUM(I)/2.0
        C       FTBIAS(I+7)=FSUM(I)/2.0
 85            FTBIAS(I+7)=.95*FTBIAS(I+7)+.05*FSUM(I)/2.0
 86     20     CONTINUE
        C
        C      Did we hit something ?
        C
        C     (  CHECK WHETHER FZSUM IS POINTING DOWN, AND HAS CHANGED SIGNIFICAN
        C
 87            RETURN
 88            END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

  1          SUBROUTINE NAV(ROB1DAT,ROB2DAT)
     C
     C
     C  NAME:  Navigation
     C
     C  PURPOSE:  Compute navigational path for robot to reach rack point
     C
     C  DESCRIPTION:  If the robot is moving toward a set rackpoint, this rout
     C      calculates the incrememtal rate of movement.  Movement is open loo
     C      subroutine CONTRL counts the number of movements it has commanded,
     C      this subroutine assumes that the robot has actually moved that num
     C      of invrements at its current incremental rate.  First, the routine
     C      updates the movement goal by subtracting the distance already move
     C      Then, it calculates the movement differential by dividing the move
     C      goal by 25 (approx no. of contrl subr executions/Nav execution).
     C      value is limited to prevent severe robot motions.  Then, these mov
     C      are transformed to tool coordinates.
     C
     C  INPUTS:  ROB1DAT  -  Robot 1 transformation data supplied by VAL
     C          ROB2DAT  -  Robot 2 transfomration data supplied by VAL
     C
     C
     C
  2          INTEGER*2 ROB1DAT(15),ROB2DAT(15)
  3          INTEGER SEG,MOTION,T11,T21,T31,T12,T22,T32,T13,T23,T33,T14,
           1     T24,T34
  4          PARAMETER(SEG=2,MOTION=3,T11=4,T21=5,T31=6,T12=7,T22=8,
           1     T32=9,T13=10,T23=11,T33=12,T14=13,T24=14,T34=15)
     C
     $INCLUDE (FTORQUE.INC)
    =1 $NOLIST
   5=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
   6=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
   7=1         INTEGER MVSKED,MVBAK,MVRAK
   8=1         CHARACTER*40 PROMPT
   9=1         CHARACTER*3 DISPT
  10=1         LOGICAL*1 RESET,FTPLOT
    =1 C
  11=1         PARAMETER (R2MSG=2,R1MSG=1)
  12=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
  13=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
  14=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
  15=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
  16=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
  17=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
  18=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1       1  FTMATX(6,6),RSCALE(6)
  19=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
  20=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
  21=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
  22=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
  23=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
  24=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
    =1       1  RDMAX,RTMAX,NCALC
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

      C
 25           IF (IAND(FTMFLAG,NAVSW).NE.0) THEN
 26              FTMFLAG=IAND(FTMFLAG,255-FCALC)
 27              ROBMOV=0.0
      C
      C    Update for Robot 1 position for increments traveled so far
      C
      C       (ONLY WHEN POINTS ARE INCREMENTAL)
      C            ROBPT1(1)=ROBPT1(1)-NCALC*ROBWDIF1(1)
 28              CALL SUBZ(ROBPT1(1),NCALC*ROBWDIF1(1),ROBPT1(1))
      C            ROBPT1(2)=ROBPT1(2)-NCALC*ROBWDIF1(2)
 29              CALL SUBZ(ROBPT1(2),NCALC*ROBWDIF1(2),ROBPT1(2))
      C            ROBPT1(6)=ROBPT1(6)-NCALC*ROBWDIF1(6)
 30              CALL SUBZ(ROBPT1(6),NCALC*ROBWDIF1(6),ROBPT1(6))
      C
      C    Compute Robot One Desired Differential Motion in world coordinates
      C
 31              ROBWDIF1(1)=ROBPT1(1)/25.0
      C    For Absolute points, use
      C            ROBWDIF1(1)=(ROBPT1(1)-RX1)/25.0
 32              CALL LIMZ(ROBWDIF1(1),RDMAX)
 33              ROBWDIF1(2)=ROBPT1(2)/25.0
      C    For Absolute points, use
      C            ROBWDIF1(2)=(ROBPT1(2)-RY1)/25.0
 34              CALL LIMZ(ROBWDIF1(2),RDMAX)
 35              ROBWDIF1(3)=0.0
 36              ROBWDIF1(4)=0.0
 37              ROBWDIF1(5)=0.0
 38              ROBWDIF1(6)=ROBPT1(6)/25.0
 39              CALL LIMZ(ROBWDIF1(6),RTMAX)
      C
      C     Translate to tool coordinates for Robot 1
      C
 40              DO 5 I=1,3
 41              ROBDIFX=0.0
 42              DO 6 J=1,3
 43              ROBDIFX=ROBDIFX+CTTOW1(I,J)*ROBWDIF1(J)
 44    6         CONTINUE
 45              ROBDIF1(I)=ROBDIFX*RSCALE(I)
 46              ROBMOV=ROBMOV+ROBDIF1(I)
 47    5         CONTINUE
      C
      C     Need angular translation here for gripping angle
      C
 48              DO 7 I=1,3
 49              ROBDIF1(I+3)=ROBWDIF1(I+3)*RSCALE(I+3)
 50              ROBMOV=ROBMOV+ROBDIF1(I+3)
 51    7         CONTINUE
      C
      C     Update for Robot 2 position for increments traveled so far
      C
      C       (ONLY WHEN POINTS ARE INCREMENTAL)
      C            ROBPT2(1)=ROBPT2(1)-NCALC*ROBWDIF2(1)
 52              CALL SUBZ(ROBPT2(1),NCALC*ROBWDIF2(1),ROBPT2(1))
      C            ROBPT2(2)=ROBPT2(2)-NCALC*ROBWDIF2(2)
 53              CALL SUBZ(ROBPT2(2),NCALC*ROBWDIF2(2),ROBPT2(2))
      C            ROBPT2(6)=ROBPT2(6)-NCALC*ROBWDIF2(6)
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 54                CALL SUBZ(ROBPT2(6),NCALC*ROBWDIF2(6),ROBPT2(6))
      C
      C    Compute Robot Two Desired Differential Motion in world coordinates
      C
 55                ROBWDIF2(1)=ROBPT2(1)/25.0
      C    For Absolute points, use
      C             ROBWDIF2(1)=(ROBPT2(1)-RX2)/25.0
 56                CALL LIMZ(ROBWDIF2(1),RDMAX)
 57                ROBWDIF2(2)=ROBPT2(2)/25.0
      C    For Absolute points, use
      C             ROBWDIF2(2)=(ROBPT2(2)-RY2)/25.0
 58                CALL LIMZ(ROBWDIF2(2),RDMAX)
 59                ROBWDIF2(3)=0.0
 60                ROBWDIF2(4)=0.0
 61                ROBWDIF2(5)=0.0
 62                ROBWDIF2(6)=ROBPT2(6)/25.0
 63                CALL LIMZ(ROBWDIF2(6),RTMAX)
      C
      C     Translate to tool coordinates for Robot 2
      C
 64                DO 9 I=1,3
 65                ROBDIFX=0.0
 66                DO 10 J=1,3
 67                ROBDIFX=ROBDIFX+CTTOW2(I,J)*ROBWDIF2(J)
 68    10          CONTINUE
 69                ROBDIF2(I)=ROBDIFX*RSCALE(I)
 70                ROBMOV=ROBMOV+ROBDIF2(I)
 71    9           CONTINUE
      C
      C     Need angular translation here for gripping angle
      C
 72                DO 11 I=1,3
 73                ROBDIF2(I+3)=ROBWDIF2(I+3)*RSCALE(I+3)
 74                ROBMOV=ROBMOV+ROBDIF2(I+3)
 75    11          CONTINUE
      C
      C     Update count variable and flag
      C
 76                NCALC=0
 77                FTMFLAG=IOR(FTMFLAG,FCALC)
 78                IF (ROBMOV.EQ.0.0) THEN
 79                   FTMFLAG=IAND(FTMFLAG,255-NAVSW)
 80                   FTMFLAG=IOR(FTMFLAG,MOVEDONE)
 81                   ENDIF
 82                ELSE
      C    No navigation
 83                FTMFLAG=IAND(FTMFLAG,255-FCALC)
 84                ENDIF
      C
 85              RETURN
 86              END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


  1           SUBROUTINE RAKMOV(N)
              C
              C  NAME:  Rack Move
              C
              C  PURPOSE:  Compute required movement of rack for coordinates received
              C
              C  DESCRIPTION:  The offset of the rack from the current coordinates is
              C      computed, including the required rotational angle at each gripper.
              C
              C  INPUTS:  N - Flag
              C                1 - Compute move to rack from current position
              C                2 - Compute move back to position from rack
              C
              $INCLUDE (FTORQUE.INC)
   =1         $NOLIST
   2=1            INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
   3=1            INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
   4=1            INTEGER MVSKED,MVBAK,MVRAK
   5=1            CHARACTER*40 PROMPT
   6=1            CHARACTER*3 DISPT
   7=1            LOGICAL*1 RESET,FTPLOT
    =1 C
   8=1            PARAMETER (R2MSG=2,R1MSG=1)
   9=1            PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
  10=1            PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
  11=1            PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
  12=1            PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
  13=1            COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
  14=1            COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
  15=1            COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1         1  FTMATX(6,6),RSCALE(6)
  16=1            COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1         4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
  17=1            COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
  18=1            COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
  19=1            COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
  20=1            COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
  21=1            COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
    =1         1  RDMAX,RTMAX,NCALC
              C
              C  Compute angle to move robots
              C
  22              IF (N.EQ.1) THEN
  23                 ANGLE=57.3*(ATAN2(RACKPT(1,1)+25.4*DX-RACKPT(2,1),
                1          RACKPT(1,2)+25.4*DY-RACKPT(2,2))-ATAN2(DX,DY))
  24                 ENDIF
              C
              C  Robot One
              C
  25              ROBPT1(1)=RACKPT(1,1)
  26              ROBPT1(2)=RACKPT(1,2)
  27              ROBPT1(3)=0.0
  28              ROBPT1(4)=0.0
  29              ROBPT1(5)=0.0
  30              ROBPT1(6)=ANGLE
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

      C
      C  Robot Two
      C
 31           ROBPT2(1)=RACKPT(4,1)
 32           ROBPT2(2)=RACKPT(4,2)
 33           ROBPT2(3)=0.0
 34           ROBPT2(4)=0.0
 35           ROBPT2(5)=0.0
 36           ROBPT2(6)=ANGLE
      C
      C  To move back, negate all values
      C
 37           IF(N.EQ.2) THEN
 38              DO 10 I=1,6
 39              ROBPT1(I)=-ROBPT1(I)
 40   10         ROBPT2(I)=-ROBPT2(I)
 41              ENDIF
      C
 42           RETURN
 43           END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

  1          SUBROUTINE PANEL(SHFLAG,DATAFLG,FTSAMPL1,FTSAMPL2,SYSFLAG,TIME)
     C
     C  NAME:  Panel
     C
     C  PURPOSE:  Print data on a display screen for monitoring
     C
     C  DESCRIPTION:  A new screen of data is formated approximately every sev
     C       seconds.  The top of the screen is status information on the syst
     C       while the bottom of the screen can be changed by command from thi
     C       console.  The available screens are:
     C           NAV - Display navigation data, positions, rates
     C           SUM - Display force summation data, base, gripper offsets
     C           XFR - Display transformation matrices supplied by VAL
     C           MAT - Display current force/feedback matrix
     C           FOR - Display raw force/torque readings
     C
     C
     C  INPUTS:  SHFLAG     -  Shared Common Area Flags
     C           DATAFLG    -  Flags showing which data is valid on input, out
     C           SYSFLAG    -  Flags showing the state of the system
     C           ROB1DAT    -  Input data from robot 1 including positional tr
     C           ROB2DAT    -  Input data from robot 2 including positional tr
     C           FTSAMP1    -  Two Force/Torque samples from the wrist of robo
     C           FTSAMP2    -  Two Force/Torque samples from the wrist of robo
     C
  2          INTEGER*1 SHFLAG(10),DATAFLG,K,SYSFLAG
  3          INTEGER*2 FTSAMPL1(36),FTSAMPL2(36),J,TIME(2),IPC
  4          CHARACTER*1 HEXDIG(16)
  5          DATA HEXDIG/'0','1','2','3','4','5','6','7','8','9','A',
           1    'B','C','D','E','F'/
  6          EQUIVALENCE (J,K)
     C
     $INCLUDE (FTORQUE.INC)
    =1 $NOLIST
   7=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
   8=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
   9=1         INTEGER MVSKED,MVBAK,MVRAK
  10=1         CHARACTER*40 PROMPT
  11=1         CHARACTER*3 DISPT
  12=1         LOGICAL*1 RESET,FTPLOT
    =1 C
  13=1         PARAMETER (R2MSG=2,R1MSG=1)
  14=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
  15=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
  16=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
  17=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
  18=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
  19=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
  20=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1       1  FTMATX(6,6),RSCALE(6)
  21=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
  22=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
  23=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
  24=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 25=1          COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 26=1          COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
   =1        1  RDMAX,RTMAX,NCALC
       C
       C   Erase the screen,  new page
       C
 27            CALL DISPCH(CHAR(27))
 28            CALL DISPCH(CHAR(42))
       C
       C   Buffer and Flag status
       C
 29            J=0
 30            CALL DISPCH('SHARE BUFFER FLAG -     ')
 31            K=SHFLAG(1)
 32            I=J/16
 33            CALL DISPCH(HEXDIG(I+1))
 34            I=J-16*I
 35            CALL DISPCH(HEXDIG(I+1))
 36            CALL DISPCH('   FTMFLAG -     ')
 37            K=FTMFLAG
 38            I=J/16
 39            CALL DISPCH(HEXDIG(I+1))
 40            I=J-16*I
 41            CALL DISPCH(HEXDIG(I+1))
 42            CALL DISPCH('   FTMFLAG2 -     ')
 43            K=FTMFLAG2
 44            I=J/16
 45            CALL DISPCH(HEXDIG(I+1))
 46            I=J-16*I
 47            CALL DISPCH(HEXDIG(I+1))
 48            CALL DISPCH(CHAR(13))
 49            CALL DISPCH(CHAR(10))
 50            CALL DISPCH(CHAR(10))
       C
       C   Device status
       C
 51            CALL DISPCH('DEVICE FLAGS - ')
 52            I=SHFLAG(2)
 53            CALL DISPCH('ROB 1: ')
 54            IF (IAND(SHFLAG(2),1).NE.0) THEN
 55               CALL DISPCH('ON ')
 56               ELSE
 57               CALL DISPCH('   ')
 58               ENDIF
 59            CALL DISPCH('ROB 2: ')
 60            IF (IAND(SHFLAG(2),2).NE.0) THEN
 61               CALL DISPCH('ON ')
 62               ELSE
 63               CALL DISPCH('   ')
 64               ENDIF
 65            CALL DISPCH('F/T 1: ')
 66            IF (IAND(SHFLAG(2),4).NE.0) THEN
 67               CALL DISPCH('ON ')
 68               ELSE
 69               CALL DISPCH('   ')
 70               ENDIF
 71            CALL DISPCH('F/T 2: ')
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 72           IF (IAND(SHFLAG(2),8).NE.0) THEN
 73              CALL DISPCH('ON ')
 74              ELSE
 75              CALL DISPCH('   ')
 76              ENDIF
 77           CALL DISPCH('BRD 1: ')
 78           IF (IAND(SHFLAG(2),16).NE.0) THEN
 79              CALL DISPCH('ON ')
 80              ELSE
 81              CALL DISPCH('   ')
 82              ENDIF
 83           CALL DISPCH('BRD 2: ')
 84           IF (IAND(SHFLAG(2),32).NE.0) THEN
 85              CALL DISPCH('ON ')
 86              ELSE
 87              CALL DISPCH('   ')
 88              ENDIF
 89           CALL DISPCH(CHAR(13))
 90           CALL DISPCH(CHAR(10))
 91           IF (IAND(SHFLAG(2),64).NE.0) THEN
 92              CALL DISPCH('                          ')
 93              CALL DISPCH('ALGORITHM ACTIVE')
 94              CALL DISPCH(CHAR(13))
 95              ENDIF
 96           CALL DISPCH(CHAR(10))
 97           CALL DISPCH(CHAR(10))
      C
      C   Error Flags
      C
 98           CALL DISPCH('ERROR FLAG ROBOT 1 -    ')
 99           K=SHFLAG(3)
100           I=J/16
101           CALL DISPCH(HEXDIG(I+1))
102           I=J-16*I
103           CALL DISPCH(HEXDIG(I+1))
104           CALL DISPCH('  ')
105           K=SHFLAG(5)
106           I=J/16
107           CALL DISPCH(HEXDIG(I+1))
108           I=J-16*I
109           CALL DISPCH(HEXDIG(I+1))
110           CALL DISPCH('          NAV   LOOPS = ')
111           CALL DISPNO(NCALC,0)
112           CALL DISPCH(CHAR(13))
113           CALL DISPCH(CHAR(10))
114           CALL DISPCH(CHAR(10))
      C
115           CALL DISPCH('ERROR FLAG ROBOT 2 -    ')
116           K=SHFLAG(4)
117           I=J/16
118           CALL DISPCH(HEXDIG(I+1))
119           I=J-16*I
120           CALL DISPCH(HEXDIG(I+1))
121           CALL DISPCH('  ')
122           K=SHFLAG(6)
123           I=J/16
124           CALL DISPCH(HEXDIG(I+1))
```

```
125           I=J-16*I
126           CALL DISPCH(HEXDIG(I+1))
127           CALL DISPCH('           OUTER LOOPS = ')
128           CALL DISPNO(IOUTER,0)
129           IOUTER=0
130           CALL DISPCH(CHAR(13))
131           CALL DISPCH(CHAR(10))
132           CALL DISPCH(CHAR(10))
      C
133           CALL DISPCH('OVERFLOW FLAG BOARD 1 - ')
134           K=SHFLAG(7)
135           I=J/16
136           CALL DISPCH(HEXDIG(I+1))
137           I=J-16*I
138           CALL DISPCH(HEXDIG(I+1))
139           CALL DISPCH('  ')
140           K=SHFLAG(9)
141           I=J/16
142           CALL DISPCH(HEXDIG(I+1))
143           I=J-16*I
144           CALL DISPCH(HEXDIG(I+1))
145           CALL DISPCH(CHAR(13))
146           CALL DISPCH(CHAR(10))
147           CALL DISPCH(CHAR(10))
      C
148           CALL DISPCH('OVERFLOW FLAG BOARD 2 - ')
149           K=SHFLAG(8)
150           I=J/16
151           CALL DISPCH(HEXDIG(I+1))
152           I=J-16*I
153           CALL DISPCH(HEXDIG(I+1))
154           CALL DISPCH('  ')
155           K=SHFLAG(10)
156           I=J/16
157           CALL DISPCH(HEXDIG(I+1))
158           I=J-16*I
159           CALL DISPCH(HEXDIG(I+1))
160           CALL DISPCH(CHAR(13))
161           CALL DISPCH(CHAR(10))
162           CALL DISPCH(CHAR(10))
      C
      C   Branch here to individual display screens
      C
163           IF(DISPT.EQ.'NAV') GOTO 3010
164           IF(DISPT.EQ.'SUM') GOTO 3030
165           IF(DISPT.EQ.'XFR') GOTO 3050
166           IF(DISPT.EQ.'MAT') GOTO 3060
      C
      C   FOR - Display raw force/torque readings
      C
167           CALL DISPCH('  FORCE/TORQUE SENSOR 1        READINGS - ')
168           CALL DISPNO(FTSAMPL1(1),0)
169           CALL DISPCH(CHAR(13))
170           CALL DISPCH(CHAR(10))
171           CALL DISPCH(CHAR(10))
172           DO 10 I=1,7
173           CALL DISPNO(FTSAMPL1(I+1),0)
```

```
174             CALL DISPCH('    ')
175    10       CONTINUE
176             CALL DISPCH(CHAR(13))
177             CALL DISPCH(CHAR(10))
178             CALL DISPCH(CHAR(10))
       C
179             CALL DISPCH('  FORCE/TORQUE SENSOR 2          READINGS - ')
180             CALL DISPNO(FTSAMPL2(1),0)
181             CALL DISPCH(CHAR(13))
182             CALL DISPCH(CHAR(10))
183             CALL DISPCH(CHAR(10))
184             DO 20 I=1,7
185             CALL DISPNO(FTSAMPL2(I+1),0)
186             CALL DISPCH('    ')
187    20       CONTINUE
188             CALL DISPCH(CHAR(13))
189             CALL DISPCH(CHAR(10))
190             CALL DISPCH(CHAR(10))
191             GO TO 5000
       C
       C   NAV - Display Navigational data
       C
192    3010     CALL DISPCH('  ROBOT 1 ALTER - RACK  X1=')
193             I=10*RACKPT(1,1)
194             CALL DISPNO(I,2)
195             CALL DISPCH(' Y1=')
196             I=10*RACKPT(1,2)
197             CALL DISPNO(I,2)
198             CALL DISPCH(' X2=')
199             I=10*RACKPT(2,1)
200             CALL DISPNO(I,2)
201             CALL DISPCH(' Y2=')
202             I=10*RACKPT(2,2)
203             CALL DISPNO(I,2)
204             CALL DISPCH(CHAR(13))
205             CALL DISPCH(CHAR(10))
206             DO 11 I=1,6
207             J=10*ROBPT1(I)
208             CALL DISPNO(J,2)
209             CALL DISPCH('    ')
210    11       CONTINUE
211             CALL DISPCH(CHAR(13))
212             CALL DISPCH(CHAR(10))
213             DO 12 I=1,6
214             J=10*ROBWDIF1(I)
215             CALL DISPNO(J,2)
216             CALL DISPCH('    ')
217    12       CONTINUE
218             CALL DISPCH(CHAR(13))
219             CALL DISPCH(CHAR(10))
220             DO 13 I=1,6
221             J=10*(ROBDIF1(I)/RSCALE(I))
222             CALL DISPNO(J,2)
223             CALL DISPCH('    ')
224    13       CONTINUE
225             CALL DISPCH(CHAR(13))
226             CALL DISPCH(CHAR(10))
```

```
      C
227           CALL DISPCH('  ROBOT 2 ALTER - RACK  X1=')
228           I=10*RACKPT(3,1)
229           CALL DISPNO(I,2)
230           CALL DISPCH(' Y1=')
231           I=10*RACKPT(3,2)
232           CALL DISPNO(I,2)
233           CALL DISPCH(' X2=')
234           I=10*RACKPT(4,1)
235           CALL DISPNO(I,2)
236           CALL DISPCH(' Y2=')
237           I=10*RACKPT(4,2)
238           CALL DISPNO(I,2)
239           CALL DISPCH(CHAR(13))
240           CALL DISPCH(CHAR(10))
241           DO 14 I=1,6
242           J=10*ROBPT2(I)
243           CALL DISPNO(J,2)
244           CALL DISPCH('   ')
245    14     CONTINUE
246           CALL DISPCH(CHAR(13))
247           CALL DISPCH(CHAR(10))
248           DO 15 I=1,6
249           J=10*ROBWDIF2(I)
250           CALL DISPNO(J,2)
251           CALL DISPCH('   ')
252    15     CONTINUE
253           CALL DISPCH(CHAR(13))
254           CALL DISPCH(CHAR(10))
255           DO 16 I=1,6
256           J=10*(ROBDIF2(I)/RSCALE(I))
257           CALL DISPNO(J,2)
258           CALL DISPCH('   ')
259    16     CONTINUE
260           CALL DISPCH(CHAR(13))
261           CALL DISPCH(CHAR(10))
      C
262           GO TO 5000
      C
      C  SUM - Display force/torque summation data
      C
263    3030   CALL DISPCH(' FORCE Z 1  ')
264           DO 21 I=1,6
265           J=10*F1ZSUM(I)
266           CALL DISPNO(J,2)
267           CALL DISPCH('   ')
268    21     CONTINUE
269           CALL DISPCH(CHAR(13))
270           CALL DISPCH(CHAR(10))
271           CALL DISPCH(' FORCE Z 2  ')
272           DO 22 I=1,6
273           J=10*F2ZSUM(I)
274           CALL DISPNO(J,2)
275           CALL DISPCH('   ')
276    22     CONTINUE
277           CALL DISPCH(CHAR(13))
278           CALL DISPCH(CHAR(10))
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 279           CALL DISPCH(' DOWN FORCE ')
 280           DO 23 I=1,6
 281           J=10*FZSUM(I)
 282           CALL DISPNO(J,2)
 283           CALL DISPCH('   ')
 284    23     CONTINUE
 285           CALL DISPCH(CHAR(13))
 286           CALL DISPCH(CHAR(10))
 287           CALL DISPCH(' FORCE DIF  ')
 288           DO 24 I=1,6
 289           J=10*FTDIF(I+7)
 290           CALL DISPNO(J,2)
 291           CALL DISPCH('   ')
 292    24     CONTINUE
 293           CALL DISPCH(CHAR(13))
 294           CALL DISPCH(CHAR(10))
 295           CALL DISPCH(CHAR(10))
 296           CALL DISPCH(' BAR SIZE  ')
 297           J=10*DX
 298           CALL DISPNO(J,2)
 299           CALL DISPCH('   ')
 300           J=10*DY
 301           CALL DISPNO(J,2)
 302           CALL DISPCH('   ')
 303           J=10*DZ
 304           CALL DISPNO(J,2)
 305           CALL DISPCH('   ')
 306           CALL DISPCH(CHAR(13))
 307           CALL DISPCH(CHAR(10))
 308           CALL DISPCH(' BASE DIF  ')
 309           J=10*BX
 310           CALL DISPNO(J,2)
 311           CALL DISPCH('   ')
 312           J=10*BY
 313           CALL DISPNO(J,2)
 314           CALL DISPCH('   ')
 315           J=10*BZ
 316           CALL DISPNO(J,2)
 317           CALL DISPCH('   ')
 318           CALL DISPCH(CHAR(13))
 319           CALL DISPCH(CHAR(10))
 320           CALL DISPCH(CHAR(10))
 321           GOTO 5000
               C
               C   XFR - Display current robot transformation
               C
 322    3050   CALL DISPCH('         ROBOT 1 TRANSFORM          ')
 323           CALL DISPCH('         ROBOT 2 TRANSFROM ')
 324           CALL DISPCH(CHAR(13))
 325           CALL DISPCH(CHAR(10))
 326           CALL DISPCH(CHAR(10))
 327           DO 34 L=1,3
 328           DO 32 I=1,3
 329           J=1000*CTTOW1(L,I)
 330           CALL DISPNO(J,4)
 331           CALL DISPCH('   ')
 332    32     CONTINUE
```

```
333           CALL DISPCH('      ')
334           DO 33 I=1,3
335           J=1000*CTTOW2(L,I)
336           CALL DISPNO(J,4)
337           CALL DISPCH('   ')
338     33    CONTINUE
339           CALL DISPCH(CHAR(13))
340           CALL DISPCH(CHAR(10))
341           CALL DISPCH(CHAR(10))
342     34    CONTINUE
343           GOTO 5000
        C
        C   MAT - Display force feedback matrix
        C
344     3060  CALL DISPCH('  PHIZ -  ')
345           J = 100*PHIZ2
346           CALL DISPNO(J,3)
347           CALL DISPCH('    DX2 -  ')
348           J = 100*DX2
349           CALL DISPNO(J,3)
350           CALL DISPCH('     DY2 -  ')
351           CALL DISPCH('  ')
352           J = 100*DY2
353           CALL DISPNO(J,3)
354           CALL DISPCH(CHAR(13))
355           CALL DISPCH(CHAR(10))
356           DO 38 L=1,6
357           DO 36 I=1,6
358           J=100000*FTMATX(L,I)
359           CALL DISPNO(J,6)
360           CALL DISPCH('   ')
361     36    CONTINUE
362           CALL DISPCH(CHAR(13))
363           CALL DISPCH(CHAR(10))
364     38    CONTINUE
        C
        C  Complete display, print prompt
        C
365     5000  J=0
366           CALL DISPCH('SYSTEM FLAG       -     ')
367           K=SYSFLAG
368           I=J/16
369           CALL DISPCH(HEXDIG(I+1))
370           I=J-16*I
371           CALL DISPCH(HEXDIG(I+1))
372           CALL DISPCH('     ')
373           CALL DISPCH('IDLE TIME - ')
374           FPC=100.0*TIME(1)/TIME(2)
375           IPC=FPC
376           CALL DISPNO(IPC,0)
377           CALL DISPCH(' PERCENT ')
        C
378           CALL DISPCH(CHAR(13))
379           CALL DISPCH(CHAR(10))
380           CALL DISPCH(CHAR(10))
        C
381           CALL DISPCH(PROMPT)
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

      C
      C  If reset flag was set, reset cumulative error flags
      C
382           IF (RESET) THEN
383              SHFLAG(5)=0
384              SHFLAG(6)=0
385              SHFLAG(9)=0
386              SHFLAG(10)=0
387              ENDIF
388           RESET=.FALSE.
389           RETURN
390           END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

1            SUBROUTINE DACDO(FTSAMP1,FTSAMP2)
     C
     C  NAME:  DAC Routime
     C
     C  PURPOSE:  Output data to DAC for plotting
     C
     C  DESCRIPTION:  Calls assembler DAC output routine seven times
     C
     C  INPUTS:  FTSAMP1 - Seven force/torque readings from robot 1
     C           FTSAMP2 - Seven force/torque readings from robot 2
     C
2            INTEGER*2 FTSAMP1(7),FTSAMP2(7),I
     C
3            DO 10 I=1,7
4            CALL DAC(FTSAMP2(I),I)
5    10      CONTINUE
     C
6            RETURN
7            END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


  1           SUBROUTINE COMMD(LINE,NSW)
  2           CHARACTER*(*) LINE
      C
      C  NAME:  Command
      C
      C  PURPOSE:  Process input commands and take proper action
      C
      C  DESCRIPTION:
      C
      C  INPUTS:  LINE  -  Command Line
      C           NSW   -  Switch for communicating with assembler driver
      C
      C
  3           CHARACTER*1 CMDS(10),NOS(6),CHARA,DIRS(5)
  4           DIMENSION DEL(6)
      C
      $INCLUDE (FTORQUE.INC)
   =1 $NOLIST
  5=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
  6=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
  7=1         INTEGER MVSKED,MVBAK,MVRAK
  8=1         CHARACTER*40 PROMPT
  9=1         CHARACTER*3 DISPT
 10=1         LOGICAL*1 RESET,FTPLOT
   =1 C
 11=1         PARAMETER (R2MSG=2,R1MSG=1)
 12=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
 13=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
 14=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
 15=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
   =1 C
 16=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
 17=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
 18=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
   =1       1  FTMATX(6,6),RSCALE(6)
 19=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
   =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
 20=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
 21=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
 22=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
 23=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 24=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
   =1       1  RDMAX,RTMAX,NCALC
      C
 25           DATA CMDS/'R','M','P','S','G','K','D','C','Q','F'/
 26           DATA NOS/'1','2','3','4','5','6'/
 27           DATA DEL/10.0,10.0,10.0,10.0,10.0,10.0/
 28           DATA DIRS/'X','Y','Z','A','P'/
      C
 29           IF (NSW.EQ.2) GOTO 100
      C
 30           DO 1 I=1,10
 31           IF (LINE(1:1).EQ.CMDS(I)) GOTO 5
 32    1      CONTINUE
 33           PROMPT='COMMAND NOT KNOWN'
 34           RETURN
```

```
      C
 35   5       PROMPT='ENTER COMMAND'
 36           GOTO (10,20,30,40,50,60,70,80,90,99),I
      C
      C   R - Reset command
      C
 37   10      RESET=.TRUE.
 38           NSW=1
 39           FTMFLAG=IAND(FTMFLAG,255-NAVSW-FTDOIT-MOVEDONE-FCALC)
 40           DO 15 I1=1,6
 41           ROBDIF1(I1)=0
 42           ROBDIF2(I1)=0
 43           ROBWDIF1(I1)=0.0
 44           ROBWDIF2(I1)=0.0
 45           ROBPT1(I1)=0.0
 46   15      ROBPT2(I1)=0.0
 47           RETURN
      C
      C   M - Modify command
      C
 48   20      CHARA=LINE(3:3)
 49           DO 21 I1=1,5
 50           IF(CHARA.EQ.DIRS(I1)) GOTO 24
 51   21      CONTINUE
 52           PROMPT='BAD MODIFY DIRECTION'
 53           RETURN
 54   24      I=5
 55           CALL GETNO(LINE,I,FNO,CHARA)
 56           GO TO (22,23,25,27,28),I1
 57           RETURN
 58   22      XGAIN = FNO
 59           GOTO 26
 60   23      YGAIN = FNO
 61           GOTO 26
 62   25      ZGAIN = FNO
 63           GOTO 26
 64   27      ALENTH=FNO
 65   26      CALL GAINM
 66           CALL TOOLT2(0.0,0.0,1)
 67           RETURN
 68   28      DPHIM=FNO
 69           RETURN
      C
      C  P - Print command - no longer implemented
      C
 70   30      RETURN
      C
      C  S - Enter New Co-ordinates for robotic movement
      C
 71   40      I=3
 72           DO 42 I1=1,4
 73           CALL GETNO(LINE,I,RACKPT(I1,1),CHARA)
 74           IF((CHARA.NE.',').OR.(I.LT.0)) GOTO 43
 75           CALL GETNO(LINE,I,RACKPT(I1,2),CHARA)
 76           IF((I1.EQ.4).AND.(I.LT.0)) GOTO 45
 77           IF((CHARA.NE.' ').OR.(I.LT.0)) GOTO 43
 78   42      CONTINUE
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 79    43      PROMPT='RACKPOINT ERROR'
 80    45      FTMFLAG2=IOR(FTMFLAG2,MVSKED)
 81            RETURN
       C
       C   G - Go Command
 82    50      NSW=4
 83            IF(IAND(FTMFLAG2,MVSKED).EQ.0) FTMFLAG=IOR(FTMFLAG,MOVEDONE)
 84            RETURN
       C
       C   K - Keyboard Guidance Command
       C
 85    60      NSW=2
 86            PROMPT = 'ENTER DIRECTION '
 87            RETURN
       C
       C   D - Display Command
       C
 88    70      DISPT = LINE(3:5)
 89            RETURN
       C
       C   C - Calibrate command
       C
 90    80      IF(LINE(3:3).EQ.'S') GOTO 86
 91            I=3
 92            CALL GETNO(LINE,I,X1,CHARA)
 93            IF((CHARA.NE.',').OR.(I.LT.0)) GOTO 83
 94            CALL GETNO(LINE,I,Y1,CHARA)
 95            IF((CHARA.NE.',').OR.(I.LT.0)) GOTO 83
 96            CALL GETNO(LINE,I,Z1,CHARA)
 97            IF(I.GE.0) GOTO 83
 98            BX = X1
 99            BY = Y1
100            BZ = Z1
101            RETURN
102    83      PROMPT = 'CALIBRATE ERROR'
103            RETURN
104    86      BX = -DX
105            BY = -DY
106            BZ = -DZ
107            RETURN
       C
       C    Q - Quit command
       C
108    90      CALL DISPCH(CHAR(27))
109            CALL DISPCH(CHAR(42))
110            DO 92 I=1,10
111            CALL DISPCH(CHAR(10))
112    92      CONTINUE
113            CALL DISPCH(CHAR(13))
114            CALL DISPCH('       PROCESSOR TWO IN TERMINATION LOOP')
115            CALL DISPCH(CHAR(10))
116            CALL DISPCH(CHAR(10))
117            CALL DISPCH(CHAR(13))
118            CALL DISPCH('       ')
119            NSW = 3
120            RETURN
       C
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

      C   F - Force/Torque feedback control
      C
 121  99      IF(LINE(3:3).EQ.'O') THEN
 122                FTMFLAG=IOR(FTMFLAG,FTDOIT)
 123                ELSE
 124                FTMFLAG=IAND(FTMFLAG,255-FTDOIT)
 125                ENDIF
 126             RETURN
      C
      C   Directional Control
      C
 127  100     CALL GETDIR(LINE,NDIR,IEND)
 128          IF(IEND.NE.0) THEN
 129             NSW=1
 130             PROMPT = 'ENTER COMMAND'
 131             RETURN
 132             ENDIF
 133          IF (NDIR) 130,120,110
 134  120     RETURN
 135  130     ROBPT1(-NDIR)=ROBPT1(-NDIR)+DEL(-NDIR)
 136          ROBPT2(-NDIR)=ROBPT2(-NDIR)+DEL(-NDIR)
 137          GOTO 140
 138  110     ROBPT1(NDIR)=ROBPT1(NDIR)-DEL(NDIR)
 139          ROBPT2(NDIR)=ROBPT2(NDIR)-DEL(NDIR)
 140  140     FTMFLAG=IOR(FTMFLAG,NAVSW)
 141          RETURN
      C
 142          END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

  1          SUBROUTINE GETDIR(LINE,NDIR,LEAVE)
     C
     C   Gets the motion direction from keyboard input line
     C
  2          CHARACTER*(*) LINE
  3          CHARACTER*1 DIRKEY(18),CHAR
  4          DATA DIRKEY/'1','2','3','4','5','6','Q','W','E','R',
           1  'T','Y','q','w','e','r','t','y'/
     C
     $INCLUDE (FTORQUE.INC)
   =1 $NOLIST
  5=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
  6=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
  7=1         INTEGER MVSKED,MVBAK,MVRAK
  8=1         CHARACTER*40 PROMPT
  9=1         CHARACTER*3 DISPT
 10=1         LOGICAL*1 RESET,FTPLOT
   =1 C
 11=1         PARAMETER (R2MSG=2,R1MSG=1)
 12=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
 13=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
 14=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
 15=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
   =1 C
 16=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
 17=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
 18=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
   =1       1  FTMATX(6,6),RSCALE(6)
 19=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
   =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
 20=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
 21=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
 22=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
 23=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 24=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
   =1       1  RDMAX,RTMAX,NCALC
     C
 25           CHAR=LINE(1:1)
 26           LEAVE=0
 27           IF((CHAR.EQ.'Z').OR.(CHAR.EQ.'z')) THEN
 28              LEAVE=1
 29              RETURN
 30              ENDIF
 31           DO 10 I=1,18
 32           IF (CHAR.EQ.DIRKEY(I)) GOTO 20
 33       10  CONTINUE
 34           PROMPT = 'BAD DIRECTION KEY!
 35           NDIR=0
 36           RETURN
      C
 37       20  IF(I.GT.12) I=I-6
 38           IF(I.GT.6) THEN
 39              NDIR=I-6
 40              ELSE
 41              NDIR=-I
 42              ENDIF
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 43             PROMPT = 'ENTER DIRECTION '
 44             RETURN
 45             END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


  1          SUBROUTINE GETNO(LINE,N,A,ECHAR)
     C
     C    Translates Number from input line
     C
     C    Inputs
     C
     C      LINE  (Character)   Input line to parse
     C      N     (Integer)     Starting position in input line
     C
     C    Outputs
     C
     C     A      (Real)        Number returned
     C     ECHAR  (Character)   Delimiting character at end of line
     C
  2          CHARACTER*(*) LINE
  3          CHARACTER*35 TEMP
  4          CHARACTER*1 NOS(13),ECHAR
  5          DATA NOS/'1','2','3','4','5','6','7','8','9','0',
           1     '.','+','-'/
     C
     $INCLUDE (FTORQUE.INC)
    =1 $NOLIST
  6=1          INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
  7=1          INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
  8=1          INTEGER MVSKED,MVBAK,MVRAK
  9=1          CHARACTER*40 PROMPT
 10=1          CHARACTER*3 DISPT
 11=1          LOGICAL*1 RESET,FTPLOT
    =1 C
 12=1          PARAMETER (R2MSG=2,R1MSG=1)
 13=1          PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
 14=1          PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
 15=1          PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
 16=1          PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
 17=1          COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
 18=1          COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
 19=1          COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1       1  FTMATX(6,6),RSCALE(6)
 20=1          COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
 21=1          COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
 22=1          COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
 23=1          COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
 24=1          COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 25=1          COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
    =1       1  RDMAX,RTMAX,NCALC
     C
 26            A=0.0
 27            IF(N.LT.0) RETURN
     C
 28            ANO=0.0
 29            APLUS=1.0
 30            ADOT=0.0
 31            DO 10 I=N,50
 32            ECHAR=LINE(I:I)
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

 33           IF(ICHAR(ECHAR).EQ.13) GOTO 110
 34           IF((ECHAR.EQ.',').OR.(ECHAR.EQ.' ')) GOTO 100
 35           DO 11 J=1,14
 36           IF (ECHAR.EQ.NOS(J)) GOTO 12
 37   11      CONTINUE
 38           GOTO 1000
 39   12      IF(J.LE.10) THEN
 40              IF(J.EQ.10) J=0
 41              IF(ADOT.EQ.0.0) THEN
 42                 ANO=10.0*ANO+J
 43                 ELSE
 44                 ANO=ANO+ADOT*J
 45                 ADOT=ADOT/10.0
 46                 ENDIF
 47              ELSE
 48              IF (J.EQ.11) ADOT=.1
 49              IF (J.EQ.13) APLUS=-1.0
 50              ENDIF
 51   10      CONTINUE
 52           GO TO 1000
      C
      C  Return from the middle of the line
      C
 53   100     A=ANO*APLUS
 54           N=I+1
 55           IF(ECHAR.NE.' ') RETURN
 56           DO 105 I=N,50
 57           IF(ICHAR(LINE(I:I)).EQ.13) GOTO 110
 58           IF(LINE(I:I).NE.' ') GOTO 103
 59   105     CONTINUE
 60   103     N=I
 61           RETURN
      C
      C  Return after carriage return
      C
 62   110     A=ANO*APLUS
 63           N=-1
 64           RETURN
      C
      C   Error return
      C
 65   1000    PROMPT='BAD NUMBER'
 66           N=-1
 67           RETURN
 68           END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


   1          SUBROUTINE TOOLT2(RX,RY,N)
   2          DIMENSION FTMATB(6,6)
      C
      $INCLUDE (FTORQUE.INC)
    =1 $NOLIST
   3=1         INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
   4=1         INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
   5=1         INTEGER MVSKED,MVBAK,MVRAK
   6=1         CHARACTER*40 PROMPT
   7=1         CHARACTER*3 DISPT
   8=1         LOGICAL*1 RESET,FTPLOT
    =1 C
   9=1         PARAMETER (R2MSG=2,R1MSG=1)
  10=1         PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
  11=1         PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
  12=1         PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
  13=1         PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
    =1 C
  14=1         COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
  15=1         COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
  16=1         COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
    =1       1  FTMATX(6,6),RSCALE(6)
  17=1         COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
    =1       4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
  18=1         COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
  19=1         COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
  20=1         COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
  21=1         COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
  22=1         COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
    =1       1  RDMAX,RTMAX,NCALC
      C
  23          IF (N.LE.0) THEN
  24             IF (N.LT.0) THEN
  25                PHIZ2=0.0
  26                ELSE
  27                PHIZX=ATAN2(RX,RY)
  28                DPHI=PHIZX-PHIZ2
  29                CALL LIMZ(DPHI,DPHIM)
  30                PHIZ2=PHIZ2+DPHI
  31                ENDIF
  32             ENDIF
      C
  33          CZ=COS(PHIZ2)
  34          SZ=SIN(PHIZ2)
      C
  35          DO 20 I=1,6
  36          DO 20 J=1,6
  37    20    GTOOL2(I,J)=0.0
      C
  38          DO 30 I=1,2
  39          GTOOL2(3*I-2,3*I-2)=CZ
  40          GTOOL2(3*I-2,3*I-1)=SZ
  41          GTOOL2(3*I-1,3*I-2)=-SZ
  42          GTOOL2(3*I-1,3*I-1)=CZ
  43          GTOOL2(3*I,3*I)=1.0
  44    30    CONTINUE
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR

      C
 45           DO 35 I=1,6
 46           DO 35 J=1,6
 47           FTMATB(I,J)=0.0
 48           DO 35 K=1,6
 49           FTMATB(I,J)=FTMATB(I,J)+FTMAT2(I,K)*GTOOL2(7-J,7-K)
 50    35     CONTINUE
      C
      C
 51           DO 45 I=1,6
 52           DO 45 J=1,6
 53           FTMSUM=0.0
 54           DO 40 K=1,6
 55           FTMSUM=FTMSUM+GTOOL2(I,K)*FTMATB(K,J)
 56    40     CONTINUE
 57    45     FTMATX(I,J)=FTMSUM
      C
      C
 58           RETURN
 59           END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


  1          SUBROUTINE GAINM
             C
             $INCLUDE (FTORQUE.INC)
   =1        $NOLIST
  2=1              INTEGER FTMFLAG,FTDOIT,FTDONE,FSTART,FTMFLAG2,ROBDIF1,ROBDIF2
  3=1              INTEGER FTGRIP,R2MSG,GRIPPD,STARTD,R1MSG,FCALC,AUTO,FTAUTO
  4=1              INTEGER MVSKED,MVBAK,MVRAK
  5=1              CHARACTER*40 PROMPT
  6=1              CHARACTER*3 DISPT
  7=1              LOGICAL*1 RESET,FTPLOT
   =1        C
  8=1              PARAMETER (R2MSG=2,R1MSG=1)
  9=1              PARAMETER (AUTO=34,GRIPPD=20,STARTD=65)
 10=1              PARAMETER (FSTART=1,FTAUTO=2,FTGRIP=4,FTDONE=8)
 11=1              PARAMETER (FTDOIT=16,NAVSW=32,MOVEDONE=64,FCALC=128)
 12=1              PARAMETER (MVSKED=1,MVBAK=2,MVRAK=4)
   =1        C
 13=1              COMMON/FTCOM/FORCE(14),FTBIAS(14),FTDIF(14),FSCALE,TSCALE,TDZ
 14=1              COMMON/FTSUM/F1ZSUM(6),F2ZSUM(6),FZSUM(6),FSUM(6)
 15=1              COMMON/FTMAT/XGAIN,YGAIN,ZGAIN,ALENTH,FTMAT2(6,6),
   =1            1  FTMATX(6,6),RSCALE(6)
 16=1              COMMON/XFORM/CTTOW1(3,3),CTTOW2(3,3),GTOOL2(6,6),
   =1            4  FTOOL1(3),FTOOL2(3),MTOOL1(3),MTOOL2(3),GTOOL1(3,3)
 17=1              COMMON/VECT/DX,DY,DZ,DX2,DY2,PHIZ2,DPHIM
 18=1              COMMON/MSGCOM/PROMPT,RESET,FTPLOT(28)
 19=1              COMMON/MCNTL/IOUTER,IFLAG,DISPT,FTMFLAG,FTMFLAG2
 20=1              COMMON/NAVECT/BX,BY,BZ,RACKPT(4,2),ROBPT1(6),ROBPT2(6)
 21=1              COMMON/NAVEPT/ROBDIF1(6),ROBDIF2(6),ROBWDIF1(6),ROBWDIF2(6),
   =1            1  RDMAX,RTMAX,NCALC
             C
             C
             C   Set the gain matrix for robot 2
             C
 22                FTMAT2(2,5)=YGAIN/20.0
 23                FTMAT2(5,2)=-YGAIN/80.0
 24                FTMAT2(1,6)=XGAIN
 25                FTMAT2(1,1)=XGAIN/ALENTH
 26                FTMAT2(6,6)=XGAIN/ALENTH
 27                FTMAT2(6,1)=2.0*XGAIN/(ALENTH*ALENTH)
 28                FTMAT2(3,4)=ZGAIN
             C      FTMAT2(4,3)=2.0*ZGAIN/(ALENTH*ALENTH)
 29                RETURN
 30                END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


  1            SUBROUTINE LIMZ(AVAL,RATE)
       C
  2            IF(AVAL.GT.0.0) THEN
  3               IF(AVAL.GT.RATE) AVAL=RATE
  4               IF(AVAL.LT..0001) AVAL=0.0
  5               ELSE
  6               IF(AVAL.LT.-RATE) AVAL=-RATE
  7               IF(AVAL.GT..0001) AVAL=0.0
  8               ENDIF
  9            RETURN
 10            END
```

```
FORTRAN-86 COMPILER
 CONTRL.FOR


   1          SUBROUTINE SUBZ(A,B,C)
   2          X=ABS(A-B)
   3          Y=A+.0005
   4          Y=ABS(Y-B)
   5          Z=A-.0005
   6          Z=ABS(Z-B)
   7          IF((X.LT..0001).OR.(Y.LT..0001).OR.(Z.LT..0001)) THEN
   8             C=0.0
   9             ELSE
  10             C=A-B
  11             ENDIF
  12          RETURN
  13          END
```

FORTRAN-86 COMPILER
CONTRL.FOR

```
 1          SUBROUTINE SUBZ(A,B,C)
 2          X=ABS(A-B)
 3          Y=A+.0005
 4          Y=ABS(Y-B)
 5          Z=A-.0005
 6          Z=ABS(Z-B)
 7          IF((X.LT..0001).OR.(Y.LT..0001).OR.(Z.LT..0001)) THEN
 8             C=0.0
 9             ELSE
10             C=A-B
11             ENDIF
12          RETURN
13          END
```

Chief
Benet Weapons Laboratory, CCAC
Armament Research, Development and Engineering Center
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCAR-CCB-TL
Watervliet, NY 12189-5000

Commander
U.S. Army Armament, Munitions and Chemical Command
ATTN: SMCAR-ESP-L
Rock Island, IL 61299-6000

Director
U.S. Army TRADOC Systems Analysis Activity
ATTN: ATAA-SL
White Sands Missile Range, NM 88002