1 of 1

UNIVERSITY OF FLORIDA

UNIVERSITY RESEARCH PROGRAM IN ROBOTICS

ANNUAL TECHNICAL PROGRESS REPORT

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Principal Investigators

Carl D. Crane III
James S. Tulenko

College of Engineering
University of Florida
Gainesville, Florida 32611
May 1994

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

TABLE OF CONTENTS

1993 University of Florida Research Team Members	iii
1993 Publications and Presentations	iv
1993 Conferences and Other Training Activities	vi
1.0 Environmental Hardening	1
1.1 Radiation Defect Annealing	1
1.2 Full System Environmental Test Chamber	12
2.0 Database/World Modeling/Vision	12
2.1 3-D Vision System	12
2.2 Data Transfer in Existing Design/Modeling Packages	16
3.0 Man Machine Interface	18
3.1 Autostereoscopic Display Unit	18
3.2 Joystick	21
3.3 Virtual Control of the REMOTEC ANDROS Robot	22
4.0 Advanced Liquid Metal Reactor Inspection Robot	32
4.1 Design and Construction	32
4.2 Control	36
5.0 Articulated Transporter/Manipulator System (ATMS) Development	44
5.1 Introduction of a New Fabrication Material for ATMS	44
5.2 Development of a Test Environment	48
5.3 Electric Linear Actuator Study	50
5.4 Horizontal Navigation and Path Planning	51
6.0 References	52

1993 UNIVERSITY OF FLORIDA RESEARCH TEAM MEMBERS

1. PRINCIPALS	2. ASSOCIATED FACULTY	3. POST-DOCTORAL ASSOCIATES
Dr. Carl D. Crane Prof. James S. Tulenko	Prof. R. Dalton Dr. J. Duffy Dr. R. Harrell Dr. R. Fox Dr. R. Singh	P. Adsit S. Toshkov

4. PROGRAM SUPPORTED STUDENTS

H. Dai:	Ph.D. student	- Man-machine Interface
R. Jurczyk:	Ph.D. student	- 3-D Visualization
S. Ridgeway:	Ph.D. student	- ATMS Design and Fabrication
X. Wang:	Ph.D. student	- PDMS Modeling; Object Recognition
M. Allbright:	Masters student	- ATMS Electric Actuator Study
D. Gamble:	Masters student	- ALMR Inspection Robot Construction and Testing
S. Gluckman:	Masters student	- ALMR Review and ATMS Design and Fabrication
D. Haddox:	Masters student	- Robot Modeling, Control, and Man-machine Interface
D. Ioannou:	Masters student	- Navigation; Visual Data Reconciliation
Y. Jang:	Masters student	- ATMS Control and Path Planning
H. Liu:	Masters student	- Radiation Effects; Environmental Hardening
S. Mandal:	Masters student	- Radiation Effects Annealing
R. Ross:	Masters student	- ALMR Review; Graduated Fall 1993
M. Vasu:	Masters student	- Man-machine Interface; Graduated Fall 1993
C. Will:	Masters student	- Man-machine Interface

5. OTHER PROFESSIONALS

S. Clifford:	- Robot Control; Computer Integration
K. Phillips:	- Project Integration and Administration

1993 Publications and Presentations

Crane, C., M. Griffis, and J. Duffy, The Development of Intelligent Robotic Systems with Practical Applications, Second Conference on Mechatronics and Robotics, Univ. of Duisburg/Moers, Germany (1993).

Dai, H. and R. Dalton. An Implementation of Fuzzy Logic Control for KZA Cybermotion Mobile Robot. Presented at the 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

Gluckman, S., R. Ross, C. Crane, and P. Adsit. ALMR Robotic Inspection: Mobility. Presented at the 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

Haddox, D. and R. Dalton. Using Computerized 3-D Graphical Visualization and Simulation to Determine the most Efficient Maintenance Strategy for the General Electric Advanced Liquid Metal Reactor (ALMR). Presented at The 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

Ioannou, D. Remote Positioning of a Mobile Robot: Development and Testing. Presented at the 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

Jones, S.L., T. Cable, J.S. Tulenko, S. Toshkov, F.R. Sias, Jr. Environmental Hardening of a Mobile Manipulator System for Nuclear Environments, Trans. Am Nucl. Soc., 69: 487-488 (1993).

Jurczyk, R. Three Dimensional Projection Display Using Concave Mirrors. Presented at the 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

Mandal, S., R.K. Singh, and J.S. Tulenko. On-Line Annealing of MOS Devices in Nuclear Environments. Presented at the 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

Mandal, S., R.K. Singh, J.S. Tulenko, R.M. Fox. Concurrent High-Frequency Annealing of MOS Devices in Nuclear Environments, Trans. Am. Nucl. Soc., 69:489-491 (1993).

Ridgeway, S., C. Crane, and P. Adsit. The Design and Fabrication of a Four Link Articulated Robot. Presented at the 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

F.R. Jr. and J.S. Tulenko. Radiation-Hardened Microcomputers for Robotics and Operated Systems, Trans Am. Nucl. Soc., 69:488-489 (1993).

ney, F., D. Carroll, C. Chen, C. Crane, R. Dalton, J. Taylor, S. Tosunoglu, T. nouth, and K. Williams, Designs for Remote Inspection of the ALMR Reactor Vessel liary Cooling System (RVACS), ANS Fifth Topical Meeting on Robotics and Remote ms, Knoxville, TN (1993).

ko, J. and C. Crane. An Overview of Robotics Research at the University of Florida. ented at the 2nd Annual DOE/NE Robotics for Advanced Reactors Student Conference, esville, FL (1993).

g, S., G.R. Dalton, J.S. Tulenko. Utilizing Visual Reality for Design, Construction, ation, Maintenance and Decommission of Nuclear Power Facilities. Presented at the 2nd al DOE/NE Robotics for Advanced Reactors Student Conference, Gainesville, FL (1993).

Conferences and other Training Activities

1993 DOE Robotics For Advanced Reactors Student Conference, Gainesville, FL, January 1993, attended by P. Adsit, P. Cherukuri, S. Clifford, C. Crane, H. Dai, S. Gluckman, D. Haddox, D. Ioannou, R. Jurczyk, S. Mandal, S. Ridgeway, J.S. Tulenko, and S. Wang.

Embedded Systems Conference-East Exhibit, Atlanta, GA, April, 1993, attended by S. Clifford and D. Ekdahl .

SPIE Short Course on Recent Advances in Sensor Data Fusion, Orlando, FL, April 1993, attended by S. Wang.

Six'h Annual Conference on Recent Advances in Robotics, Gainesville, FL, April 1993, attended by P. Adsit, C. Crane, J. Duffy, R. Harrell, and S. Ridgeway.

Fifth Topical Meeting on Robotics and Remote Handling, Knoxville, TN, April, 1993, attended by C. Crane, P. Adsit, J. Tulenko, S. Clifford, and D. Haddox. Also demonstrated technical developments at a booth.

32-Bit Microprocessor Showcase, Las Vegas, NV, May, 1993, attended by J.S. Tulenko.

American Nuclear Society Annual Meeting, San Diego, CA, June, 1993, attended by J.S. Tulenko.

ALMR Inspection Robot technical team meetings, Oak Ridge, TN, and Gainesville, FL, June, October, and November, 1993, attended by C. D. Crane, P. Adsit, S. Ridgeway, S. Gluckman, and D. Gamble.

SBIR Site Visit to Lightwave Electronics, Mountain View, CA, November 1993, conducted by J.S. Tulenko.

American Nuclear Society Winter Meeting in San Francisco, CA, November 1993, attended by J.S. Tulenko.

Advanced Liquid Metal Reactor Inspection Robot Demonstration, Oak Ridge, TN, December 1993, participation by C. D. Crane, P. Adsit, S. Ridgeway, D. Gamble, and J. S. Tulenko.

UNIVERSITY OF FLORIDA
UNIVERSITY RESEARCH PROGRAM IN ROBOTICS
1993 ANNUAL TECHNICAL PROGRESS REPORT

1.0 ENVIRONMENTAL HARDENING

During the year 1993, the research on environmental hardening of electronic components and materials for robotic systems for advanced nuclear facilities focused on two subtasks. First, work continued toward further investigating the effect of gamma radiation and annealing on two types of metal oxide semiconductor (MOS) devices, namely MOS capacitors and CMOS inverters. Second, testing of current components of maintenance/surveillance robots continued. This task included the development of a system for testing the environmental hardness of full robotic systems.

1.1 Radiation Defect Annealing

A capacitor is the simplest form of MOS device, from which the radiation responses of gamma radiation can be determined. The annealing experiments on MOS capacitors were of two types: (1) thermal annealing and (2) AC bias annealing, in which high frequency (50MHz) was applied on the capacitors. Inverters are fabricated by the series connection of a PMOS and a NMOS transistor. CMOS inverters were chosen for our successive experiments since they are the simplest members of the CMOS family, and they have standard voltage characteristics. These characteristics can be used as a tool to study the radiation and annealing behavior of the inverters. By cross-coupling two CMOS inverters, a SRAM unit memory cell can be obtained. It is expected that the results of the CMOS inverter experiments will be applicable to SRAM chips. For the testing of CMOS inverters, a novel annealing technique was tried, based on varying the duty cycle of the high frequency annealing waveform.

Results of the MOS Capacitor Experiments

The rate of generation of oxide and interface charges was not significantly altered by low temperature thermal annealing conditions. Testing was conducted up to a total dose of 100 krad at 30 °C, 150 °C, 200 °C, 250 °C. The total annealing of interface and oxide charges was found to be less than 10% in most cases. Figure 1 shows the voltage shift due to the oxide charges at 30 °C and 150 °C.

A significant AC annealing of the capacitors was observed. For example, under worst bias conditions (+5 V), up to 25% annealing was observed when the AC bias of 7 V (peak-to-peak amplitude) was superimposed on the DC bias. These results were obtained for a total dose of 150 krad. Figure 2 and Figure 3 show the results of this experiment for oxide and interface traps.

The amount of AC annealing was found to increase significantly as the peak-to-peak voltage of the AC bias was increased. For example at 50 MHz, the total annealing increased from 25% to 50% when the peak-to-peak AC bias was increased from 7 V to 10 V.

The combination of thermal and AC annealing did not produce significant changes compared to the AC annealing case. This finding confirms our earlier results, that under these

Figure 1. Voltage shift due to oxide charges as a function of total dose at 30 °C and at 150 °C

Figure 2. Voltage shift due to oxide charges as a function of total dose for AC bias superimposed on DC bias

Figure 3. Voltage shift due to interface traps as a function of total dose for AC bias superimposed on DC bias

irradiation conditions thermal annealing does not play a significant role in the operation of MOS capacitors.

Radiation response for capacitors was a strong function of oxide thickness. As the thickness of the oxide was increased, the voltage shift due to oxide charges was found to increase significantly.

Results of the CMOS Inverter Experiments

DC bias irradiation. Results of simultaneous application of DC input bias and gamma radiation on inverter voltage characteristics show that when the input bias increases from -1 V to 5 V, lateral voltage shift of the inverter characteristics also increases. This lateral voltage shift increases in a linear fashion with respect to input bias in this range. However, below -1 V of input bias the voltage shift plateaus (see Figure 4). A total dose of 25 krad was applied to these inverters.

AC bias irradiation. (a) Effect of Duty cycle: For all of the experiments conducted to determine the effect of duty cycle on the lateral voltage shift of the inverter characteristics, the frequency of the input AC bias was 10 kHz. A maximum dose of 25 krad was applied at a dose rate of 280 rad/minute. Results with -5 V to 5 V amplitude square wave shows that with the gradual increase of duty cycle from 20% to 80%, inverter characteristics move to the left gradually (see Figure 5). The corresponding duty cycle versus. lateral voltage shift characteristics shows a linear increase of the voltage shift with respect to the duty cycle of the AC input bias.

Another experiment on the effect of duty cycle used an input bias of a 0 V to 5 V square wave with a frequency of 10 kHz. The results of this test showed that the voltage shift increased linearly with the average input bias (Figure 6).

(b) Effect of amplitude: Two experiments were conducted to determine the effect of the amplitude of the AC bias, one with -5 V to 5 V amplitude AC bias and one with 0 V to 5 V amplitude AC bias. In these experiments, the lateral voltage shift of the inverter characteristics obtained with 0 V to 5 V amplitude AC bias was always higher than the voltage shift obtained with -5 V to 5 V amplitude AC bias. However, the difference in the voltage shift remained constant and did not depend on the duty cycle of the AC bias. Figure 7 shows that an AC bias of -5 V to 5 V amplitude and 20% duty cycle produced minimum lateral voltage shift, whereas an AC bias of 0 V to 5 V amplitude and 80% duty cycle produced a maximum lateral voltage shift of the inverter characteristics.

(3) Effect of frequency: To determine the effect of frequency on the voltage shift of the inverter characteristics, on-line annealing experiments were conducted with AC biases of 10 kHz, 0.1 MHz, 1 MHz and 10 MHz frequencies. In each case, the biases were -5 V to 5 V square waves with 50% duty cycle. These inverters were irradiated to a total dose of 75 krad at a dose rate of 280 rad/minute (see Figure 8). A semilogarithmic plot of frequency versus voltage shift, Figure 9, shows that the voltage shift decreases linearly with the frequency of the ac bias.

Future Experimental Work

In order to further understand the effects of radiation on MOS devices, the following experiments should be conducted:

Figure 4. Inverter voltage characteristics at different input DC voltages after a total dose of 25 krad

Figure 5. Inverter voltage characteristics at different duty cycles of AC input bias after a total dose of 25 krad (-5 to 5 V, 10 kHz, square wave)

Figure 6. Inverter voltage characteristics at different duty cycles of AC input bias after a total dose of 25 krad (0 to 5 V, 10 kHz, square wave)

Figure 7. Shift of inverter voltage characteristics after total dose of 25 krad as a function of duty cycle of the input high frequency bias

Figure 8. Inverter voltage characteristics at different AC input biasing conditions after a total dose of 75 krad

Figure 9. Shift of inverter voltage characteristics after total dose of 75 krad as a function of the frequency of the input AC bias

(1) Perform A-C annealing of a single SRAM cell made from a CD 4069 Hex inverter chip.

(2) Bringing down the low level of the annealing wave forms to negative voltage has been found to accelerate the annealing process in case of the inverters. Since this negative voltage may result in loss of data from the memory cell, the low level of wave form should be kept at 0 V and at -2 V for two different experiments to ascertain the impact of negative voltage on the low level wave form.

(3) To analyze the effect of annealing, the two cross-coupled inverters forming the SRAM should be characterized individually after the annealing in order to determine the new threshold voltage of the transistors therein. The change in threshold voltage of the transistors will determine the annealing performance.

1.2 Full System Environmental Test Chamber

In 1993, an experimental setup was established to provide for full-system electronics environmental testing. The testing is being performed in the Florida Linear Accelerator, a facility located adjacent to the University of Florida campus. Figure 10 shows the layout of the test equipment in the accelerator facility. In the experimental setup, the electronics under test are placed in a test box meeting military standard specifications and are mounted on a cart that can be used to move samples past the accelerator beam. The cart is stationary for the robotics experiments. To date, black and white video cameras, color video cameras, and robotic batteries have been tested for radiation performance.

The test control electronics are mounted in a shielded housing built with lead bricks. The distance between the radiation chamber and lead housing is four to six feet, allowing the test control electronics to be outside of the direct beam. With the accelerator operating at a maximum power of 15 kW, the test control electronics will receive approximately 3.83 rad/hour. The direct beam dose (50 cm from the x-ray producing target) at this power level is 240,000 rad/hour.

A 120-foot cable transmits the signals from the device under test in the irradiation test box to a PC-462 board mounted in a PC-AT 386 computer, located in the main room of the accelerator. This computer records all parameters of interest and allows for the determination of the point of electronic failure.

The dose to the electronics is determined by the calibration of the accelerator facility and is confirmed by the use of Gafchromic dosimetry film placed on the device under test. This film changes optical density with exposure to radiation and can accurately determine the dose received.

2.0 DATABASE/WORLD MODELING/VISION

2.1 3-D Vision System

During 1993, the scheme of a CAD-based (or model-based) 3-D vision system was established. The goal of this work is to develop a real-time general vision system which may be used in both structured and unstructured environments for tasks such as inspection, repairing, and hazardous waste cleaning up. The system includes a stereo image acquiring module, an image preprocessing module, a CAD modeling module (PDMS or IGRIP), an object recognition model, a model-based segmentation module, and a 3-D surface reconstruction model.

Figure 10. **FLORIDA LINEAR ACCELERATOR**

Image Acquiring Module

The image acquiring module includes two CCD cameras, a frame grabber (DT1451) and a frame processor (DT1458), a VME bus, a Sun Sparc 2 workstation, and a Silicon Graphics Crimson VGXT workstation. The CCD cameras are the vision sensors. The function of the frame grabber is to digitize the analog image from CCD cameras into the gray scale image with values ranging from 0 to 255. The frame processor is used for some simple preprocessing, such as image adding, subtracting, and convoluting. The VME bus hosts both the frame grabber and frame processor. Its UNIX-like operating system also serves as a link to the workstations. The Sun is the major computer for image processing, while the SGI is used for graphical model generation.

Images are acquired from the two CCD cameras to obtain 3D information of a scene. An extensive study of the main parameters that influence the output of the cameras was undertaken. More complicated models than the usual linear models were considered in order to model the camera distortion. The frame processor was programmed to implement image convolution and image arithmetic operation such as adding, subtraction, multiplication and division. Incorporation of these basic functions accelerate the image preprocessing procedure.

Image Preprocessing Module--Edge detection.

A wavelet-based edge detector was chosen for this system. This approach is equivalent to that of the multiscale Canny edge detector, but uses quadratic spline 1-D wavelet to accomplish smoothing. The idea of using multiscale edge representation is to construct edges by superposing the edges of each scale. Thus, the strong edges that present at all scales will be boosted, whereas noise and fine details will be suppressed.

CAD modeling module

The model construction is accomplished using PDMS, a CAD system developed by CADCentre, Inc. The model for recognition consists of a group of 2-D gray scale images generated by REVIEW, a display package of PDMS. The model construction is done through the usual design procedure in PDMS, and then is recorded as gray scale image by a screen grabbing software. Using PDMS, the different perspective views can be obtained accurately. Separation of display and design procedures makes PDMS uniquely suitable for vision applications, because it provides high fidelity and high speed in its display, which can be recorded and compared directly with the real images. Utilization of 2-D images also solves a key problem in CAD-based vision -- the best representation for computer aided design and for machine vision are not the same

Object recognition module

Three-dimensional object recognition has attracted tremendous attention in computer vision research. This research has focused on a generic, high-performance object recognition module, analogous to the human vision system, which can recognize a 3-D object from a single 2-D gray scale image. The mathematical representation employed in this module is Fourier transformation. Fourier Transformation Shift Theorem and Conformal mapping provides this module the ability to deal with the translation, scaling and on-image-plane rotation. The salient features of this

system include: 1) unrestricted shape and complexity of the object, 2) tolerance to occlusion and noise, and 3) processing at a real-time speed. The applications of this module include object recognition, camera motion detection, image comparison, and unknown object detection.

The initial testing on a small number of objects shows that the method made a correct decision 100% of the time, even with images with up to 70% occlusion. The whole recognition procedure takes 14 seconds. However, because operations of this method are mainly the fast fourier transformation, image adding and subtraction, and image geometrical transformation, the processing time can be decreased drastically if it is implemented on hardware, such as a DSP (Digital Signal Processor) based image processor.

Model-based Segmentation Module

A post-processing module for the object recognition module, which can identify all pixels belonging to the object in the image, was also developed. The significance of this module is that it provides the accurate shape of the object at its current state and separates it from the foreground, that is, the objects that occlude it. If combined with 3-D sensory data acquired by a stereo camera or a laser range scanner, this information can be used to generate the surfaces of the object, which is very valuable for tasks such as hazardous waste cleaning up and nuclear facility decommissioning. For example, in manipulation applications, the robot controller needs to know the surface configuration to decide grasping points for the end effector.

In order to construct the segmentation module, a hybrid algorithm that combines a region-growing method and a boundary detection method was developed. In this algorithm, the template is first subtracted from the input image at the location of its centroid. Then, a threshold is applied to this resultant image to obtain an initial classification. At the region growing stage, the method compares the homogeneity of the local region of the image with the model to identify the pixels belonging to the object. At the boundary test stage, the current boundary is tested against the boundary of the model to decide if the procedure needs to be terminated. Because this operation is performed locally and under the direction of the model, it is fast and robust.

Surface reconstruction module

The surface reconstruction is a fundamental step in digital photogrammetry, in which sensed 3-D information can be extracted as a CAD model to update the existing database for autonomous robot systems. The surface reconstruction module implemented in this research uses the regularization method, which interpolates the visible surface with a thin-plate and membrane model constrained by spring forces. The distinguishing feature of this algorithm is, that by treating the nodal equations as computational molecules, the discontinuities of the surface can be handled easily by inhibiting or adding the corresponding molecules. In this implementation, both a laser range image and a video reflectance image are used. The range image provides the constraints of depth and orientation, and the reflectance image provides the discontinuity maps of depth and orientation, which are derived from its jump edge map and roof edge map, respectively. Fusion of range image and reflectance image gives an efficient and reliable way to extract the discontinuity map of surfaces for Terzopoulos' algorithm. The experiments with this implementation demonstrate the advantage of fusing different types of sensor data. They also illustrate the problems of using the spline-based surface representation for the dense range data.

Position Determination Using Straight Lines

An algorithm was developed to determine position information using straight lines in the robot's environment. Two CCD cameras are used as the sensory system, and the images that come from the cameras are fused with the use of Kalman filtering. Extraction of the straight lines is accomplished using the Hough transform. This method provides 3-D position information for autonomous robot navigation.

2.2 Data Transfer in Existing Design/Modeling Packages

Introduction

There has been a great deal of interest in importing existing engineering design and graphics data, especially when it is in electronic or magnetic form, into other design and graphics programs, allowing for the automatic reuse of data. Engineering data describing several reactor and robotic applications has been examined and transferred between various systems at the University of Florida. These programs or systems include AutoCAD, the Plant Design Management System (PDMS), and the Interactive Graphical User Interface Program (IGRIP). Extensive investigations have been completed to import and export graphic and engineering data between these programs.

General Data Exchange

In general, there are two methods of exchanging data between different programs. Using the first, one exports the data from one program, using an industry standard interface specification such as IGES or DXF. It should be noted that these IGES or DXF files are plain text files with a defined structure. Then one imports the data into the second program using the same standard interface specification. The drawback to use of such industry standard interfaces is that they tend to be of the lowest common denominator among the several packages that use such information.

For the second method one looks for a pair of computer programs that share a one-to-one interface format specification that is unique to the pair of programs. The PDMS / WaveFront interface is an example of such an interface. Here, because the specification involves only two groups, the interface can be modified quickly to accommodate new advances in technology. It thus is generally up to date, allowing transfer of the latest advances in each program.

The PDMS System

The PDMS program is actually a very large group of programs that run sequentially on a wide range of large computers, from IBMs to Vaxes, Silicon Graphics workstations, and Suns. The heart of the PDMS system is a single, very large, very secure database. This database can be used by the Draft module to construct and print isometric engineering drawings to be used for shop fabrication of the objects. Another module can be used to assemble pre-build equipment groups into large processes and even entire plants, with the ability to automatically check for spatial clashes, and compatibility of engineering specifications, such as pressure and materials. There are modules that will take data from the same database and assemble bills of materials for ordering and scheduling during constructions. Finally there is a package, Review II, that can be

used to produce attractive, artistic presentations, of the assembled product, process, or plant for video presentation. The output of this module can be sent to further video processing using products such as WaveFront to produce smooth blends of captions, music, and renderings of technical information.

Input To PDMS. Input of data into PDMS has traditionally been restricted to the normal PDMS structured computer graphics interface provided by PDMS, using keyboard and mouse. One could not take a pre-built file of input data, in say the IGES format, and import it into PDMS. Because of the very high priority placed on data integrity within PDMS, it will accept data only through the PDMS interface, where checking and verification can take place.

On a second input issue, construction of "catalogues" of engineering data within PDMS, there seems to be similar block. This is in the area of constructing catalogues of standard parts with standard sizes, and other standard engineering attributes, e.g. maximum pressure, maximum temperature. It is with these "catalogue" entries that PDMS is able to do consistency checking from one part to the next as parts are put together within PDMS to form an engineering assembly.

Discussions with the developer of PDMS, CadCentre, Inc., and other users (Chapman and Deacon at University College, London) indicate that building catalogues is best, and possibly exclusively done, by CadCentre staff. It seems possible that training in these construction techniques could be arranged. But, the techniques are not readily available, nor in the readily available documentation.

Output From PDMS. Output of data from PDMS to external programs or devices is available in at least three interfaces, IGES, DXF, and WaveFront. The output from IGES and DXF are both accessible from within the PDMS Draft module. As the name implies the main thrust of Draft is to produce 2-D drawings, in the form of top, side, front views, and isometric views. These views, along with dimensions, construction notes, and other information are designed to produce working drawings for shop construction of the hardware described.

As a result of the starting point of 2-D drawings, all of the output from PDMS to IGES and DXF formats is in the form of 2-D drawing-like objects. The data describing the Argonne vacuum furnace tank was exported from PDMS using each of the two interfaces, IGES and DXF. This data was then imported into AutoCad Release 12. After importation, the AutoCad drawing could be rotated and it was then clear that the data was all in a single plane, representing the exported top view from PDMS.

Another output interface has been developed between CadCentre and WaveFront Technologies to produce high quality video and large format hardcopy prints in color. The suite of WaveFront programs allow the importation and manipulation of 3-D PDMS images within the WaveFront programs. In general the cost of the WaveFront programs is the same order of magnitude, many 10's of thousands of dollars, as the cost of PDMS. So, although PDMS users have large design teams using the programs for design year around, the demand for WaveFront processing is much less frequent and much less intensive. PDMS users have often passed the output from PDMS to a WaveFront software service center for final processing to produce polished video and hard copy images. Unfortunately even exporting data in the WaveFront format requires an additional program be purchased from the CadCentre. Then one must gain access to a high powered workstation that has the WaveFront program available. One copy of the WaveFront program has just been brought to the University of Florida campus.

Data Exchange Between AutoCad and IGRIP

Both the IGRIP robotic modeling package and the AutoCAD engineering design package support the generic interfaces, IGES and DXF, for import and export of data. Both the packages support 3-D constructions, manipulations and import/export of 3-D data. Data can be exported from the IGRIP via the IGES interface. A primitive version of the Argonne vacuum furnace tank was constructed in IGRIP and exported through the IGES interface to a file. The wire frame version of the model arrived and was displayed within AutoCAD, and rotation confirmed the true 3-D nature of the model. The wire-frame model could then be rendered as a solid using features of AutoCAD. Note that the DXF format is also available to IGRIP, but not in the version installed at the University of Florida.

Practical Limitations On Data Interchange

There are, in fact, reasons that the exchange of graphical (and other types) of data between top of the line computer programs tends to lag far behind the individual computer program's abilities to generate and use graphical data. The companies that produce the top of the line computer graphic programs are always striving to introduce new techniques that will be at the leading edge applications of the hardware and software technology. These techniques are typically pioneered by a single company, and only a year or two later do they begin to appear in competitors' hard/software. It is only then, after a new technique is widespread, that it is incorporated into generic interchange format descriptions. Of course, by this time, the top of the line graphic features are no longer top of the line, and they have been replaced by yet newer techniques, often very useful, but not widely shared by competitors.

To put it briefly, generic information interchange specifications tend to include those features that are widely accepted and used by many competitors. These industry wide standard interfaces rarely include newest, most desirable features of a top of the line products.

3.0 MAN MACHINE INTERFACE

3.1 Autostereoscopic Display Unit

Introduction

The three dimensional viewing device described here will not require the viewer to wear any headgear, and the viewer will have the ability to move about the image during viewing. As the viewer moves around the display unit, he/she will be able to see the object being displayed from various angles, just as though he/she were viewing a 'real' object. The intention here is to offer a TRUE three-dimensional display unit, that imposes no viewing constraints. This device may be described as an "Autostereoscopic Display Unit" (ADU).

General Description

The viewing device proposed in this work is designed to utilize computer stored images and information and produce a three dimensional image. The premise upon which the theory is based is the human eye's ability to "fuse" images that are displayed at speeds greater than 17 Hz.

The computer generated images are projected onto a semi-translucent surface that is being rotated at a high speed (i.e., 1200 rpm). The images projected onto the viewing surface are slices of whole images. The slices of the image are then "fused" together by the human brain.

Figure 11. Overview of ADU system

Additionally, front and rear views of the same image may be simultaneously projected onto the viewing device from different angles. This would allow the device to not only display the front view of an object, but the rear view could be seen if the viewer were to walk to the back of the display. While this screen is rotating, a full-color projection display projects 'slices' of an image sequentially onto the projection display. Due to the high rate of speed of the display surface, the viewer's eye will fuse all of the sequential images to one continuous three-dimensional image.

Projection Surface

Various materials have been studied for use as a projection surface. The projection surface should ideally be made of a milky-white, semi-transparent material, i.e. semi-transparent plastic, like that of a milk jug. The projection surface is centrally mounted, is circular when viewed from the top (see Figure 12), and it should be inclined at a certain angle (i.e., 30 degrees). The projection surface sweeps out a volume of space as it spins around its center axis. In the case at hand, the space "swept out" by the device would be in the form of a vertical cylinder. The cylinder acts as the projection screen or medium upon which the image may be displayed.

Figure 12: Projection surface: (a) top and (b) side view.

A slight adjustment must be made to the rotating screen because, although rotation of the projection surface is continuous with time, the image displayed by the projector is non-continuous due to the nature of projectors. A rotating shutter is added to the system to correct for the non-continuous projection of image slices. The shutter is placed between the projector and the display surface, and keeps the displayed image from "washing out" or "bleeding".

Projection Display

The display used for projection may be any display that is capable of projecting multiple images per second. The image projector must be able to display images that are stored in electronic format.

For our initial tests, a color LCD projection display will be used. This display will be controlled through a computer, and is capable of displaying SVGA images at a rate of 72 frames/sec (Hz), and at a resolution of 640 x 480 pixels.

Motorized Base Unit

The projection screen needs to be rotated at a speed greater than 17 Hz (the speed at which the human eye fuses images). For this reason a speed of 20 Hz was selected, resulting in an overall rotational speed of the projection display of 1200 revolution per minute (rpm). This is achieved by using a direct-drive electric motor, synchronized with the rest of the system through a computer.

A mechanical shutter, located between the projector and the projection display, is also connected to an electrical motor and synchronized by the computer. This shutter consists of two blades attached to a shaft through its center, similar to a propeller. The motor operates at a speed of 2160 rpm (72 images / sec are displayed through the shutter with two blades).

Computer Equipment

The ADU is computer controlled. As a test platform a PC with an Intel® P5 (Pentium) processor, operating at 60 MHz and running OS/2® version. 2.1 was chosen. (OS/2 is a multi-tasking, multi-threaded, graphical operating system, allowing multiple programs to be run on the computer simultaneously.)

The images to be displayed are generated by, or downloaded to the PC. They are processed, if necessary, and a set of consecutive frames of slices of the object in 1 degree increments are generated. These frames are stored and are then displayed by the projection display.

If the image changes (i.e. if the image is a live picture, such as in the case of air traffic control), the above steps are simply repeated continuously, and new frames are continuously generated and displayed.

Size of Projection Surface / Resolution

The size of the projection surface can be customized for any use. It only depends on desired resolution, the type and quality of the projection display used, and the computing power available. For this test, a pixel size of 1 mm x 1 mm was selected as offering sufficient resolution. Since the projection display has a resolution of 640 x 480 pixels, this results in a display size of 640 mm x 480 mm (25.2 in x 18.9 in).

Applications

Applications could include such uses as monitoring of air-traffic over busy airports, or the display of a new prototype that hasn't been built yet. Military uses would include such things as battlefield simulation or overview, monitoring of satellites, etc.

3.2 Joystick

Introduction

Remote operation of manipulator systems reduces human exposure to dangerous environments. Task efficiency improves as teleoperation systems become more transparent. Force reflection is an example of controller augmentation that increases this transparency. Reactive force-feedback controllers can only exert force on the operator as a result of his effort. This yields potentially safer control than traditional actuated joysticks. It is also free from the problem of self-actuation.

General Description

The passive force feedback joystick built by Paul Goodall uses 6 electromagnetic hysteresis brakes and a force-torque sensor. Position is measured with high precision linear pots for each axis. The brakes were chosen because they can provide a constant resistance regardless of speed, even if it is stationary. They also have long service life because there are no contacting friction members. The brakes have exhibited a problem with cogging as a result of residual magnetism. This cogging occurs as an alternating light drag.

Past Efforts

Control strategies for minimizing cogging suggested by the brake manufacturer were mostly unsuccessful. The most successful strategy involved slowly deactivating the brake while it was being moved. This gave the controller a sticky feel, and it did not do much to remove the cogging. A new control strategy involving bi-directional current was tested and met with some success. The new strategy required rewiring of the joystick, new brake control circuitry, and a new power supply.

Current Goals

Control of the joystick has been implemented with an elementary graphical simulation. Simulation of a more general environment that can be interfaced with other sensor simulation and control programs is the next goal.

3.3 Virtual Control of the REMOTEC ANDROS Robot

Introduction

The University of Florida team has replaced the standard ANDROS control system has been replaced by the University of Florida with an in-house developed virtual control system. Through the use of Deneb Robotics' Interactive Graphical Robot Interface Program (IGRIP) and a Silicon Graphics Crimson Elan VGXT workstation, a 3-D computer model has been created that replicates the robot and its environment. In this "synthetic environment" IGRIP is able to quickly calculate the movements needed by the *andros* model, or "virtual" robot (*andros*), to reach an object and/or to carry out a task, where a task is any set of coordinated movements. Upon successful review and operator approval, the necessary commands are automatically conveyed to the actual ANDROS robot to execute the identical set of movements.

The implementation of this control system required many sub tasks involving development in the following areas:

- Development of the 3-D model
 - a) model the ANDROS with IGRIP primitives
 - b) model the environment
 - c) program the inverse kinematics into the model

- Development of the IGRIP control program
 - a) develop GUI in IGRIP (raft popups, etc.)
 - b) program development for single joint movement
 - c) program multiple joint movement
 - d) create tag point structure for grabbing an object
 - e) create a targeting and movement scheme
 - f) develop an encoder and position feedback
- Development of ANDROS command program
 - a) keyboard control only, with serial communication
 - b) socket communication with IGRIP (via IGRIP sockets)
 - c) non-blocking I/O on both keyboard and sockets
 - d) multiple joint-movement commands performed simultaneously
- Client-server connection via Transmission Control Protocol/Internet Protocol (TCP/IP)
 - a) standard sockets
 - b) IGRIP sockets; macros of Berkeley Standard Distribution (BSD) sockets
 - c) client-server paradigm
 - d) server routes commands to the serial port
- Hardware additions/modifications for ANDROS
 - a) install an embedded computer (486SX PC)
 - b) design and install joint encoders
 - c) install sonar sensors for obstacle avoidance
 - d) design and install an infrared positioning system
 - e) replace serial connection to ANDROS with an Ethernet link

Because of the flexibility of the "synthetic environment," the University of Florida (UF) has created a system that is able to give an ANDROS operator the ability to "point-and-shoot" the robot to carry out routine movements and tasks. Likewise, the robot can be directed to target, move to, and manipulate any object that is both in the real world and the "synthetic environment" with just a touch of a button. The control program calculates the myriad commands and the forward and inverse kinematics to perform these tasks, thus automating many robot operations. This development frees the ANDROS operator from the tedium of performing repetitive tasks and lets the operator function in a supervisory mode.

Modeling the Robot

The IGRIP Software. IGRIP is a computer based software package designed for workcell layout, simulation and off-line programming. Individual parts are created within the CAD context. Multiple parts are attached to form devices with both geometric and non-geometric attributes (kinematics, dynamics, velocities, degrees of freedom, etc.). A workcell is composed of various devices positioned relative to each other forming a synthetic environment for the virtual robot. In the context of the IGRIP workcell, it is easy to visually maneuver about an object, seeing both external and internal detail of any part or device in the workcell. Dimensional and/or positional analysis of devices or between devices can be reported in a

continuous mode or upon demand. Furthermore, devices in this synthetic environment can be controlled by programming using IGRIP's Graphical Simulation Language (GSL), a PASCAL/C-like language. The software has many capabilities including: collision detection, excessive joint acceleration/extension detection, center of gravity calculation, and the ability to communicate to external processes via sockets, files, and pipes.

The ANDROS Robot and its Virtual Twin. The ANDROS Mark VA robot was designed to remove an adult from a burning building or subdue a criminal with its taser or shotgun weapons systems. It has tank-like tracks for mobility and a set of front and back appendages ("legs") for climbing stairs or raising its torso to higher elevations. A robotic arm with four degrees of freedom with a powerful gripper are attached to the torso. Cameras, variable intensity light sources, speaker and microphone give the robot two-way communication back to a command center.

The virtual robot (*andros*) was created in IGRIP with a set of "identical" geometric and non-geometric features for simulation purposes. A total of approximately 30 parts consisting of from one to twenty sub-parts form the *andros* robot in the workcell. The workcell is also appointed with several of the more obvious features (blackboard, tables, desks, chairs, etc) in the Mobile Robotics for Hazardous Environments Laboratory (MRHEL) at the Nuclear Engineering Sciences building, University of Florida. The other objects in the MRHEL provide the opportunity for programming the robot with path planning, object avoidance, and collision detection, in addition to providing some recognizable features for simulating the views from the cameras that are on-board ANDROS.

Building the inverse and forward kinematics model for the *andros* took planning and ingenuity. *Andros* is composed of seven kinematically activated parts:

- "dummy" is the first or base part to which all other parts and devices are attached. It is a 0.1 mm diameter disk that is not seen in the workcell. A simple six degree of freedom (DOF) inverse kinematic map with three translational (TTT) and three rotational (RRR) is assigned to this part giving the operator the ability to place the entire robot at any position/orientation in the workcell. The torso of the *andros* is attached to "dummy".
- front "legs" are attached to the torso with one rotational degree of freedom (R) allowing them to be rotated up or down using forward kinematics.
- since independent movement of the rear legs is required, a separate kinematic device is required for the rear legs of *andros*.
- the "arm" is a four DOF (RRRR) device and is attached to the front of the torso.
- the gripper is composed of a set of opposing "jaws" that have equal but opposite rotational DOF.
- the color pan and tilt camera (RR) is attached to the arm's shoulder joint and forward kinematics is used for camera pointing.

All devices and joints can be moved simultaneously or alone using forward kinematics. However, only "dummy" and the arm can be move using inverse kinematics.

One other "invisible" device is required so that the gripper can be made to move directly and automatically to any reachable x, y, z coordinate in any orientation in the workcell. A device called "dummy2" is placed at the center of rotation (about z) of the shoulder. An array of 8100 distinct (x, y, z, yaw, pitch, roll) "tag points" form a plane of reachable points. This improvised

technique was necessary because the inverse kinematics requires six parameters and the end effector of *andros*/ANDROS lacks the necessary DOF's to access the exact approach axis. However, for many manipulation tasks, for example, grabbing a pipe or a block off of the floor, having a unique approach vector is unnecessary.

Development of the Control Program

Philosophy Behind Virtual Robotic Control (VRC). Maneuvering through a complex work environment can be challenging for an operator of a mobile robot. Errors in operator judgement may lead to equipment or system damage. Camera perspectives often can be ambiguous, leaving the operator confused resulting in a trial-and-error operational mode. Distance and positional information via sensors alone aid in robotic movement, but do not provide quick solutions for moving around obstacles on the way to its objective position. Path planning algorithms can solve the obstacle avoidance problem, but generally do not provide quick on-the-fly movement solutions to uncharted tasks.

The VRC system provides quick easy access to any object from any point to any point in the workcell. The GSL program coupled with two external C processes, can actually place the end effector of the robot on any object at any point on the object by simply touching the object with the mouse pointer; both *andros* and ANDROS move simultaneously to a floor base position from which the object is reachable with its arm; the arm then automatically moves to the object. IGRIP calculates the kinematics necessary to make the movements; the GSL program is designed to send "small" incremental movements via sockets to a server that actually issues commands to ANDROS. However, the virtual robot (*andros*) in its synthetic environment must exactly coincide with the actual robot (ANDROS) environment, otherwise positional errors could be potentially catastrophic. The hardware and software considerations to allow for accurate positioning of the robot will be discussed later in this paper.

ANDROS Control User Interface Program (ACUIP). Development of the Graphical User Interface using IGRIP is fairly routine. There are two categories of popup menus available to the GSL programmer: input/output Popups and Procedural Popups. Input Popups are used to indicate selections from a displayed list, or for storing values to be assigned to variables. Procedural Popups automatically execute a user-defined set of GSL procedures upon selection of a displayed Popup button. Raft Popups are primarily used to modify the behavior of a running program or external process. Furthermore, Raft Popups can be displayed or suppressed to allow or disallow user intervention. All of the previously mentioned GUI features are incorporated into the ACUIP.

The first level of robot control establishes joint-by-joint control. A basic GSL program with procedural Raft Popups was designed to give the user the ability to "jog" the *andros*/ANDROS, moving one joint by a selected number of degree increments ranging from 1 to 180. A slightly more difficult problem to solve became apparent when simulated leg movement for *andros*/ANDROS was attempted, since rotating the leg downward changed the angle of the torso and the relative angle of the other leg to the floor. The solution involved solving the geometry shown in Figure 13. The two angles, a_1 and a_2 are the rotatable leg joints of ANDROS and can be rotated to plus or minus 90 degrees. When angle a_1 or angle a_2 is

rotated, all other angles change along with line segment l3. Line segments L1 and L2 are of constant length. The solution to the geometry diagramed below is as follows:

Figure 13. *andros* Leg Geometry

$$l_4 = \sqrt{l_1^2 + l_2^2 + 2l_1 l_2 \sin a_2} \quad (1)$$

$$a' = \cos^{-1} \frac{l_4^2 + l_2^2 - l_1^2}{2l_4 l_2} \quad (2)$$

$$b_1 = \cos^{-1} \frac{l_1^2 + l_3^2 - l_4^2}{2l_1 l_3} \quad (3)$$

$$b_2 = 2\pi - a_1 - a_2 - b_1 \quad (4)$$

$$l_3 = \sqrt{l_4^2 + l_1^2 - 2l_1 l_4 \cos(a')} \quad (5)$$

$$b_2 = 2\pi - a_1 - a_2 - b_1 \quad (6)$$

$$\theta = \pi - a_1 - b_1 \quad (7)$$

The joint-by-joint control method proved not to be a substantial improvement to the standard joystick control mode although its development provided a 3-D graphical interpretation of actual events and the basis for the next design phase.

Three additional areas of development were required for the robust control desired:

1. simultaneous multiple joint movement for both the actual and virtual robots.
2. a targeting and retrieval software mechanism
3. a feedback loop to correct for cumulative errors in movement caused by joint "slop" or slippage.

Simultaneous joint movement alone is not a significant GSL programming problem. The programmer could simply issue a command "packet" with the proper bits turned on and multiple joints will move a selected number of degrees. The more difficult problem was determining which joints and the total number of degrees for movement to the desired destination. A smooth, coordinated move where both *andros*/ANDROS matched "exactly" in position and orientation for each time/fraction-of-degree increment of movement was desired. The solution was an interrupt driven routine for both the GSL and the external C program that would send a command packet every second (the standard IGRIP updates graphics at 1 Hz, 30 Hz is available in its LLTI upgrade) to the ANDROS. At this frequency of transmission, "real-time" movement could be simulated. Since *andros*' movement is actually slightly ahead of ANDROS' movement, collision and joint over-extension can be prevented by continuously testing for and circumventing these conditions before they actually occur.

The next challenge was to create the "sphere" of reachable tag points (x,y,z, yaw,pitch, roll) that the *andros*/ANDROS arm could reach so that simply with a mouse button command, the arm would place its gripper at an intended point. A fine spherical mesh of points was desired; this was impossible to create manually in the IGRIP program. A separate program for placing a tag point at the end of the gripper was written so that as the arm moved to a reachable position, a tag point was placed permanently and invisibly marking that point in the workcell relative to the robot. This process of moving and tagging was carried out for a plane of movement directly in front of the robot. A plane of 8100 tag points was made with a movement resolution suitable for our purposes. However, virtually any resolution for a tag point mesh is possible. As the *andros* moves through its workcell, the plane of tags moves with it, maintaining its relative position to the center of Z rotation of the shoulder. When the shoulder is rotated about the Z axis to any angle, the plane of tags is also rotated, yielding an "infinite" number of reachable tags located spherically around ANDROS.

Targeting the selected object was somewhat of a trivial geometric solution but required dividing the workcell into quadrants to insure proper rotation of the entire robot. When targeting an object, a determination of whether or not the object is reachable in its present position and orientation must be made, or if "reachability" can be achieved by rotating the shoulder about its Z axis will suffice, or finally, if the entire robot must be rotated and translated to achieve reachability. Distances between points on the ANDROS and its robotic arm are calculate in relation to the target; this is a standard utility feature of the IGRIP software. By triangulating, the approach vector is determined. The ANDROS moves rotates and translates as necessary and "grabs" the object at the selected tag point. However, before the arm can actually move to the

object, it must first determine which of the 8100 tag points is closest (usually within 1 mm with this resolution). This determination is accomplished with another external C that does a comparison (only in the correct quadrant) or search for the best kinematic tag and returns the tag identification number. The GSL then commands *andros*/ANDROS to move to the tag, giving the operator the ability to move the *robot*/ROBOT anywhere in their respective environments with the "touch" of a mouse cursor.

Initially, the control system was void of a feedback loop, and only rough estimates of movement based on motor-operation time could be achieved. Joint encoders were installed to provide the arm joints with a feedback loop. Three basic encoder types were considered:

- a. optical
- b. potentiometer
- c. magnetic

Criteria for encoder type selection are as follows:

- 1. Availability
- 2. Installation
- 3. Accuracy
- 4. Cost

Optical encoders were chosen based on cost, accuracy, reliability, ease of installation and familiarity. The installation scheme was modeled and visualized using IGRIP. A major installation consideration for the coder wheel and associated electronics was to make the encoder system insensitive to the inherent movement oscillations of the robotic arm ("slop"). The solution employed was to attach the coder wheel and optical sensor directly at the center of rotation of the specific joint. Forward and reverse movement is detected by the Computer Dynamics CDX-CTC/PIO counter-card along with the associated on-board Computer Dynamics SBC-486DX embedded PC. The encoder sensing electronics produces two "identical" wave forms that are phase-shifted by 90 degrees as shown in Figure 14.

By utilizing the programmable software gating feature of the counter card, one wave form is counted while the other is gated. The difference between the forward and backward pulse-counts is the actual joint position.

Due to alignment problems between the coder wheel and the optical sensor, the installation became difficult. Several attempts at alignment proved unsuccessful. A method was conceived for combining both components of the encoder device into one unit with a flexible coupling connection to the approximate center of joint rotation. While this method was promising, two joints of the ANDROS robot could not be fitted with this encoder system.

A third encoder system was devised and implemented on the robot's arm successfully. However, a trade-off was required. The initial system has accuracy within 20 minutes of arc compared to one degree accuracy of the latter mentioned system. But, the new system had the benefits of ease of installation and greater durability at approximately 1/10 the cost. The coder wheels were simply created in AutoCAD, printed on a transparency using a LaserJet series II, cut out by hand and easily installed on ANDROS. An optical sensor with a Schmidt trigger and amplifier was manufactured and mated to the coder wheel without the alignment problems of the stock HP encoders. Figure 15 shows the schematic of the optical sensor. Additionally, this optical system can be installed on any of the robots various joints.

Figure 14. Optical Encoder Wave Forms

Figure 15. Optical Encoder Schematic

Figure 16 shows the ACUIP control system. In the photograph, ANDROS/*andros* are seen in their respective environments. In the synthetic environment, "zooming in", "flying around", top view, front view, etcetera are easily done even as the robots are moving in their environments.

Figure 16. *andros*/ANDROS in their respective environments.

Development of the Command Program

In order to replace REMOTEC's control of the ANDROS, it was necessary to write software that could generate and send commands in the same packetized format as the original console. As a first step, a C program that "spoke" to the ANDROS through one of the workstation's serial ports, taking input from the keyboard, was written. The keyboard was mapped such that each key represented individual ANDROS movements (eg. 'd' => 'Shoulder up'). A standard C package of routines used for screen and keyboard management called *curses* was used to implement key processing and screen output.

Next, it was necessary to communicate with IGRIP to get the movement commands. This was accomplished by adding socket communications to the command program, making the command program a "server" in a client-server topology. IGRIP became the client; initiating a connection to the command program at startup, and sending commands in the form of characters (ie. like the keyboard had been mapped), followed by a number representing an integral multiple of executions of that command. For instance, sending 'd 5' through the socket would mean send five packets in a row to the ANDROS to raise its shoulder.

Once socket communications were in place it became necessary to make the program "asynchronous" or "event driven." Both sockets and *curses* allow for non-blocking, or asynchronous input/output (i/o), meaning that the program will interrupt whatever it is doing to take care of incoming information. Often the action is simply queuing by the system for sequential processing of data or signals, but the queuing insures that the information will not be lost due to a busy program. Simple "polling" of input is susceptible to loss of information that arrives out of synch with read operations.

Finally, the program was augmented to allow multiple joint-movement commands to be performed simultaneously. If the IGRIP model needs to move both the shoulder and the elbow, for example, then the command program can tell the ANDROS to move those joints together. However, keeping track of when a joint movement begins and ends puts a burden on the ACUIP, but it enhances the performance of the robot for carrying out its task.

Client-Server Connection via TCP/IP

Because the ACUIP and the command program needed to communicate with each other, it was necessary to either develop or find a reliable method for this communication. The "Internet protocol suite," (Stevens, 1990) or TCP/IP, was chosen for the following reasons:

- portability: TCP/IP is available on most platforms, including DOS, Unix, and VMS
- reliability: the Transmission Control Protocol (TCP) (Comer, 1991) provides reliable delivery of data
- programmability: TCP/IP has a standard application program interface (API) that is well documented
- commonality: IGRIP "speaks" sockets (as well as pipes)

The API used by IGRIP and the command program is the BSD socket. Sockets provide the interface to the underlying communications protocols. The IGRIP socket is a macro of the BSD socket, including length of packet information along with the data being transmitted within the packet.

The layout of the IGRIP-to-ANDROS communication path is based on a client-server paradigm. In this case, the command program acts as a server, accepting connections from the client (the ACUIP), and acting on the information transmitted through the socket. The server processes the information coming from the ACUIP by converting the character commands into the eight-byte serial command packet that ANDROS uses. Within the eight-byte command packet, each bit represents a command or condition. While it is not necessary for the server to provide both socket management and information routing, these functions can be handled by a single process here since only one connection is maintained at any given time (ie. only one ACUIP is ever running).

Hardware Additions and Modifications

The ANDROS robot is driven by an 8052 micro-controller and is not designed to perform bidirectional communication with the operator. In order to implement sensor feedback, a 486SX 33 Mhz embedded PC is being installed within ANDROS. The embedded PC has a 10 counter daughter card which collects counts from the optical encoders. This information is then to be reported back to IGRIP via the external C program. Initially the embedded PC will be connected to the workstation by a serial link.

Because IGRIP is capable of communicating via TCP sockets over Internet, a proposed idea is to replace the serial link between the embedded PC and the workstation with a wireless Ethernet connection, thus eliminating the need to route encoder information (hardware modifications previously discussed) through an intermediate C engine. This new configuration

will expedite communications by eliminating a relatively slow serial link and external C process. Instead, IGRIP could receive the data directly. Also, ethernet provides ~10 Mhz of bandwidth which is necessary when we decide to communicate vision data between the ANDROS and the workstation.

Since mobile robots readily tend to slip or slide without detection, several techniques are presently under consideration for regular orientation and position updates. One such system currently propose for use outside is an infrared (IR) tracking and positional updating scheme. The system would provide several unique infrared sources that would be attached to ANDROS' torso and end effector or gripper area. In addition, three (or more, depending on terrain) fixed infrared scanner towers would scan and identify the IR sources. By triangulation, the position and orientation of the ANDROS would be known. The positional updates would then be transmitted via wireless Ethernet, back to the workstation where the ANDROS' position and orientation would be updated for "re-targeting" the objective position.

Furthermore, for safety purposes, a redundant obstacle avoidance system using sonar sensor would be desirable. If the "real" and virtual worlds did not exactly coincide, another method for determining a safe path or system shutdown would be essential.

4.0 ADVANCED LIQUID METAL REACTOR INSPECTION ROBOT

4.1 Design and Construction

Background

General Electric's proposed Advanced Liquid Metal Reactor (ALMR) would require an ultrasonic reactor-vessel weld inspection process in order to comply with federal regulations. Standard weld inspection measures are not suitable for use with the ALMR because of temperature, contamination, and access limitations.

The ALMR utilizes liquid sodium as a heat transfer agent. To prevent the sodium from solidifying, the reactor vessel wall temperature is maintained at 400°F when the reactor is shut down. Standard reactor-vessel weld inspection techniques operate at room temperature. Liquid sodium is highly reactive, and therefore can be allowed to come into contact with only the most inert materials. Introduction of foreign material into the interior of the reactor vessel is forbidden. Miniature flaws in the welds to be inspected are most likely to form on this restricted surface. The flaws must therefore be detected from outside the reactor vessel -- through three inches of austenitic weld. The reactor vessel is encompassed by a containment vessel. The current design of the ALMR specifies an eight-inch gap ($\pm$ one inch) between the exterior of the reactor vessel and the interior of the containment vessel. The ultrasonic inspection device must operate in this eight-inch gap. This enclosed volume must also be kept free of contaminants in the event of a reactor vessel breach. To prevent oxygen from entering this volume, it is filled with inert argon gas at slightly higher than atmospheric pressure. Therefore, the ultrasonic inspection must be done without a couplant, which would contaminate this environment. Furthermore, this argon-filled chamber can only be accessed through four 15 inch x 5 inch airlocks located at the top of the reactor.

The University of Florida was given the responsibility for developing a platform that could that could operate in the space between the reactor vessel and the containment vessel of the

ALMR. The platform would carry ultrasonic sensors specially designed by the Department of Energy's Oak Ridge National Laboratory.

Mobility Concepts

Several sensor mobility concepts were considered. One concept proposed installation of a horizontal rail ringing the reactor vessel, from which a sensor suite could be lowered via a cable. The sensor suite could travel around the reactor on the rail. Such a system would be unable to adequately inspect the underside of the reactor vessel. Another idea was to use a mobile rover that maintained its position by pushing against the reactor vessel wall and the containment vessel wall simultaneously with spring-loaded arms or wheels. Concerns over slipping were later strengthened when it was discovered that such a device had been built in the 1960's, and had failed because of slippage. Another proposed rover would have used magnets to maintain vertical position. The reactor vessel walls, however, were not ferromagnetic. The prototype concept finally used was a wall-crawling rover that used vacuum-cups to maintain vertical position (see Figure 17).

Figure 17: Prototype Schematic

Implementation

Vacuum Subsystem. The vacuum cups used are constructed of a silicone-based rubber rated for continuous exposure to 480°F. Each 4.4-inch diameter cup can theoretically counter a 120-lb axial or 66-lb shear load, when supplied with 27-in Hg vacuum, and can adhere to a nonporous surface with a radius of curvature greater than 28 inches.

The prototype uses two sets of these vacuum cups for locomotion. One set enables rotary motion and also functions as a placeholder during linear motion. The other set of vacuum cups enable linear motion.

On-board vacuum generators supplied the cups with the necessary 27-in Hg vacuum required for peak performance. The generators used venturi stages to develop the vacuum when supplied with 100-psi air. While the prototype used compressed air for economic considerations, a final version would be supplied with compressed argon to prevent contamination of the work environment. The vacuum generators were fitted with off-the-shelf optional viton seals in order to meet the 400°F temperature requirement. Two vacuum generators are used on the prototype--one for each set of vacuum cups.

Two vacuum sensors were also mounted on the rover to confirm that the cup sets had established strong vacuum seals on the walls. These sensors function as safety checks during normal locomotion of the rover. Off-the-shelf sensors that could withstand the reactor temperature were obtained but were not used due to size and time constraints. Smaller custom sensors could be made if required.

Rotary Subsystem. Three vacuum cups were mounted on a triangular rotor located at the geometric center of the rover's underbelly. The rotor was turned by a pneumatic rotary actuator (see Figure 18). The rotary actuator consisted of a pinion wedged between two racks. When the racks were moved linearly in opposite directions by pneumatic pressure, the pinion and attached shaft rotated. The rotor was threaded onto the pinion shaft, and was prevented from backing out by a left-handed bolt and washer secured into the pinion shaft. The rotary actuator used was capable of rotating through 180°, and could deliver 498 in-lbs of torque when supplied with 100-psi air.

Air flow to the rotary actuator was controlled by a pneumatic servo-valve (not shown). A closed control loop was obtained by using a custom fiber-optic optical encoder for position feedback. The 12-bit encoder used a fiber-optic cable to pass signals from the temperature-resistant encoder wheel to the electronics, which would be located outside the reactor. The encoder used had a 16-ft fiber-optic cable. Longer cables can be obtained if necessary.

The entire rotary subsystem could be extended and retracted by two pancake pneumatic cylinders. The pneumatic cylinders and the rotary actuator were off-the-shelf components fitted with optional viton seals. The only rotary subsystem component that does not meet the temperature limitations is the pneumatic servo-valve. If a heat-resistant version could not be found after further search, it would have to be placed in the cooling box developed by the University of Texas that was designed to cool navigation equipment contributed by the University of Tennessee.

Linear Subsystem. Four vacuum cups were used in the linear motion subsystem. The cups were positioned near the four corners of the wall-crawler for extra stability when holding onto the wall. Each cup was attached to a rigid frame that could be moved linearly along the

main axis of the crawler by a threaded drive shaft with a two-mm pitch (see Figure 19). The drive shaft was turned by a small DC motor producing 60 oz-in of torque at 250 rpm. Geometric considerations limited the linear subsystem to a four-inch stroke. The sliding frame consisted of a pair of linear rails, with two sliding blocks (with attached vacuum cups) moving along each rail. The entire linear subsystem could be extended or retracted by four pancake pneumatic cylinders positioned over the vacuum cups. The position of the linear subsystem was reported by an optical encoder identical to the one used in the rotary subsystem. The custom encoder was attached to the drive shaft, and linear position was calculated using the known angular position and the drive shaft pitch. Limit switches were used to help keep the subsystem in calibration. After moving to its four-inch limit, the linear subsystem would reset as follows:

1. Rotary subsystem extended toward wall.
2. Attachment of rotary subsystem vacuum cups to wall verified by vacuum sensors
3. Linear subsystem retracted from wall
4. Linear subsystem reset to beginning of stroke cycle (at other movement extreme).
5. Linear subsystem extended toward wall
6. Attachment of linear subsystem vacuum cups to wall verified by vacuum sensors
7. Rotary subsystem retracted from wall

The linear subsystem could then move the platform forward another four inches. By cycling through the above steps, the wall-crawler could be made to "walk" along the wall. A change of direction could be made when the rotary subsystem is attached to the wall and the linear system is retracted.

Figure 18: Components of rotary subsystem

Figure 19: Components of linear subsystem

Test and Performance

The prototype sensor platform was first demonstrated at Oak Ridge National Laboratory in December 1993. The platform crawled vertically for several minutes on a full-scale mockup of the ALMR containment vessel. No slippage of the platform was discernible, even when an approximately 100-lb downward force was exerted on the attached rover (with all seven vacuum cups attached).

There were, however, several difficulties. The rotor was found to be off-axis, introducing a wobble to the platform's movements. As this wobble could precipitate a loss of contact with the wall, rotary testing was postponed until the rotor can be rebuilt. The pneumatic servo-valve controlling the rotary subsystem has also failed, probably due to clogging, and needs to be cleaned. Finally, the fiber-optic encoders do not deliver output, and have been temporarily replaced by potentiometers.

Conclusions

The validity of the mobility concept has been established with the successful prototype demonstration. Future wall-crawlers based on the demonstrated prototype should be able to perform the ALMR inspection successfully. While there are relatively minor technical flaws in the prototype, they arise from the implementation of the mobility concept, and not from the concept itself. Work is currently underway to repair these flaws.

4.2 Control

Along with designing and constructing a new robot comes the need to develop control hardware and software and a programming method. Two robots meeting different environmental and size requirements designed by the University of Florida are the Articulated Transporter/Manipulator System (ATMS) and the Advanced Liquid Metal Reactor (ALMR) inspection robot. This section will describe the programming language developed to control these robots.

Robot Hardware and Software

The ALMR inspection robot was controlled using a Force SYS68K CPU-30ZBE (MC68030, 25 MHz) mounted in a VME chassis running the OS/9 operating system. Along with a central processing unit, several input/output boards were utilized to interface the computer to the environment: a VME Microsystems International Corporation VMIVME-3116 16-bit analog to digital (A/D) converter board, a VMIVME-4100 12-bit digital to analog (D/A) converter board, and a Force SYS68K PIO-1 parallel input/output (PIO) board. A schematic of the hardware can be found in Figure 20. Two additional computers were also used—a Macintosh SE/30 and a Silicon Graphics (SG) computer. The Macintosh computer was utilized as an interface terminal to the Force computer and provided a platform for developing robot programs and data graphing capabilities. The SG computer was utilized for path planning, kinematic modeling, and displayed an animated view of the robot. Other hardware included servo-amplifiers, solid state relays for switching and sensing the status of AC and DC devices, angular encoders, linear transducers, and actuators.

Figure 20. Computer hardware

Figure 21. Operating environment

Intelligent robot operation required the ability to sense the environment, make intelligent decisions, and output information to actuation devices. The operating environment (Figure 21) was divided into an operating system, debugging facilities, programming language, path planning, a data base, and the work environment. Central to the operating environment was a data base. The data base provided a method of data exchange between various elements in the operating environment. The data base represented a section of computer memory reserved for storage of global information.

The operating system included an input and output interface to the work environment (input and output drivers), watchdog routines, and a scheduler. Input drivers included the hardware and software necessary to quantify the work environment. An output driver included the hardware and software necessary to produce either a change in or change in perception of the work environment. The watchdog software monitored system activity for faults and took appropriate actions if fault conditions were detected. The scheduler was a real-time kernel used to synchronized execution of the software components in the operating environment.

An example of an input driver was a collection of the components required to quantify a robot joint position. In the simplest case where a potentiometer was used on a robot joint, the components of an input driver included the power supply to the potentiometer, the potentiometer itself, A/D converter hardware, and software required to obtain and convert the A/D data to a meaningful result.

An example of an output driver included the components required to actuate a robot joint. For a hydraulic cylinder, the output driver would include the software necessary to calculate a signal to send to the joint actuator, D/A converter hardware to convert a digital number to an analog voltage for a servo-amplifier, a servo-amplifier, and other servo-hydraulics components such as a hydraulic power unit, a servo-valve, and the actuator itself.

The watchdog software monitored the scheduler routine and removed hydraulic system pressure in case the scheduler software was not operating at the real-time rate of 100 Hz. The watchdog routine was only implemented for the ATMS robot and provided a safety feature.

The debugging facilities were used to monitor activity during execution of a robot program and included a user interface and a data logger. Through user interface facilities, data base information could be observed and changes could be made to data base parameters during robot operation. A data logger utility recorded robot data at a real-time rate and was used to document robot operation. Data were transferred to the Macintosh computer for graphing purposes.

The SG computer was used for path planning and display purposes. In the case where the SG computer was used to generate a path for the robot, the desired joint data were transferred to the data base over an ethernet line using the Helix software (Jones, 1992). Actual joint data were transferred at a rate of ≈ 10 Hz. from the data base to the SG for remote display of the actual robot motions during operation.

Programming Language Overview

This section provides an overview of the programming language used to develop programs for the ALMR robot. This language was also used to develop programs for the ATMS robot, described in section 5.0. A detailed description of the programming language can be found in Adsit (1989). The programming language consisted of three elements: states, models, and parameter definitions. Robot actions were specified using the state element of the programming

language. The states were linked together into a state network using logical combinations of modeling results and global parameters. Each state specified actions to be performed, determined when actions had been completed or a problem existed, and specified the next state to activate. The model language structure provided a method to detect important events by combining sensor and system information. Models reduced input and output driver and system data to binary values for use in state links. Global parameters were used for communication between the user interface and a robot program and were also used for storage of information between control cycles.

Programming a robot task is shown graphically in Figure 22. Action specifications for a state were shown as rounded rectangles and represented modifications to the data base. Lines (or links) between the states represented decisions and indicated when and which state should next be activated. A single state consisted of one rectangle and all links emanating from that state to other states. Only one state was active at any given time. The other remaining states were inactive. A state was placed in the active mode by performing modifications to the data base according to the actions specified by the state. Examples of data base modifications included specification of a particular feedback control mode or establishing control set points. After modifications were made to the data base, the state was placed in the active mode. While the state was active, links to other states were evaluated to determine when the next state should be activated. If a link evaluated true, the current state was deactivated and the state pointed to by the true link was activated.

State links evaluated the status of global parameters and modeling results and directed program flow. In Figure 22, the links to other states are numerically labeled with a state designation letter prefixing a number. Thus state A was linked to states B and C with the links A1 and A2 respectively. The number associated with the link indicated the link evaluation order. While state A was active, the actions specified by state A were performed while the links A1 and A2 were sequentially evaluated. State A remained active if both links A1 and A2 evaluated false. If one of the links evaluated true, state A was deactivated and the state pointed to by that link was activated. The numerical ordering of the links was important since as soon as a link evaluated true, the state pointed to by that link was scheduled for activation.

Changes to a task program were accomplished by the addition of new states and linking existing states to the new states. In this manner, problems recognized during execution of a robot program could be solved. This concept is shown in Figure 23 with the addition of state E and links A3, E1, and E2. The link A3 determined if a problem occurred while state A was active and, if so, activated state E. Links from state E to existing states C and D continued program execution after the problem had been solved by state E. Note that only state A in the pre-existing network of states had to be altered. This feature allowed alterations to be made with minimal modification to an existing robot program.

Models characterized events occurring in the environment by using modeling criteria to map robot data to binary results. Although input drivers mapped infinitely variable environmental quantities to finite numerical results, the data sets were still too large to be handled efficiently in state links. It was easier and more uniform to program state links by using models results. All models had two elements: model logic and model parameters. A natural language version of a model to determine if a joint position (p) was outside a defined minimum (p_{min}) or maximum (p_{max}) position limits was:

Figure 22. Programming concept

Figure 23. Addition of a state to program

IF (p is greater than pmax) OR (p is less than pmin)
 THEN joint position is out of position limits
 ELSE joint position is within position limits.

In C language syntax, the model logic would be: $fpool = (p > pmax) \parallel (p < pmin)$;
 The joint position (p) was a result from an input driver that quantified joint position. The minimum (pmin) and maximum (pmax) position limits were model parameters. The parameter fpool was a model result flag and, if true, indicated that the joint position was out of positional limits. If fpool was false, the joint position was within the minimum/maximum limits.

The state network was programmed using a spread sheet that was divided into two parts—the initialization section and the state link section. An example of the state network developed for the ALMR robot is given in Figures 24a and 24b. Note that due to paper size limitations, the state network was printed in two parts. The initialization section occupied the upper portion of the spread sheet indicated by the *Initialization* row heading. The first column in the initialization section lists modeling parameters, global parameters, or output driver set points. Table 1 contains a summary of these parameters. Each column (except the first) in the state network represented a state and the first cell in each column contained the state name. The state link portion occupied the lower portion of the spread sheet and started with the *State Links* row heading. The state name is repeated in this row for convenience. A link from one state to another was represented by an entry at the intersection of a state column and a state row.

Table 1. Initialization parameter list.

<u>Parameter</u>	<u>Comment</u>
f_walk	flag—initiate walking sequence.
f_robot_secured	flag—robot is secured by lab personal.
f_ok_to_walk	flag—robot is on wall and ready to walk.
f_move_direction	flag—direction of robot travel.
fc_linear_vacuum	control flag—status of linear vacuum.
fc_rotary_vacuum	control flag—status of rotary vacuum.
fc_linear_sc	control flag—linear suction cups position.
fc_rotary_sc	control flag—rotary suction cups position.
fc_linear_to_max	control flag—linear slider to maximum.
fc_linear_to_min	control flag—linear slider to minimum.
mp_state_time	model parameter—length of time (1/100 s) before model result flag is set true.

For example, when the command state was activated, the parameters in the initialization column were set to the value found in the adjoining cell in the command column. Thus, the *f_walk*, *f_robot_secured*, and *f_ok_to_walk* parameters were set to FALSE. The *f_move_direction* flag was set to TRUE. Both control flags for linear and rotary vacuum were set to OFF and both linear and rotary suction cup position were set to IN. An output driver would use the control parameters and send a command to the appropriate actuator.

The state links connected various states in the network and provided program flow. For example, the command state only had one link to another state. This was indicated by the *1. f_walk* entry at the intersection of the command column and the *walk_start* row. This link evaluated the status of the *f_walk* flag and, if true, indicated that the *walk_start* state should be activated. To start a walking sequence, the *f_walk* flag was set to true through the user interface. Initialization of the *walk_start* state set the control flags for the linear suction cups (*fc_linear_sc*) and the rotary suction cups (*fc_rotary_sc*) to the OUT value. This caused the output driver to

Initialization	command	walk_start	walk_end	linear_move_max	linear_move_min	linear_vacuum_off	rotary_vacuum_off
linear.setpoint	linear.home	linear.home	linear.home	linear.max	linear.min		
rotary.setpoint	rotary.home	rotary.home	rotary.home				
f_walk	FALSE		FALSE				
f_robot_secured	FALSE	FALSE	FALSE				
f_ok_to_walk	FALSE	FALSE	FALSE				
f_move_direction	TRUE						
fc_linear_vacuum	OFF	ON	ON			OFF	
fc_rotary_vacuum	OFF	ON	ON				OFF
fc_linear_sc	IN	OUT	OUT				
fc_rotary_sc	IN	OUT	OUT				
mp_state_time				15	15	100	50
State Links	command	walk_start	walk_end	linear_move_max	linear_move_min	linear_vacuum_off	rotary_vacuum_off
command			1. f_robot_secured				
walk_start	1. f_walk		2. f_walk				
walk_end		1. !f_walk		1. !f_walk	1. !f_walk	1. !f_walk	1. !f_walk
linear_move_max							
linear_move_min							
linear_vacuum_off							
rotary_vacuum_off							
linear_sc_out		2. f_ok_to_walk && f_move_direction && fs_vacuum		3. linear.m_slop fs_linear_max && !f_move_direction	2. linear.m_slop !fs_linear_min && f_move_direction		
linear_sc_in						2. mf_state_time && !fs_linear_vacuum	
lin_sc_in_for_rot_vac							
rotary_sc_out		3. f_ok_to_walk && !f_move_direction && fs_vacuum		2. linear.m_slop fs_linear_max && f_move_direction	3. linear.m_slop !fs_linear_min && !f_move_direction		
rotary_sc_in							2. mf_state_time && !fs_rotary_vacuum
rot_sc_in_for_lin_vac							

Figure 24a. Partial state network for the ALMR robot

Initialization	linear_sc_out	linear_sc_in	lin_sc_in_for_rot_vac	rotary_sc_out	rotary_sc_in	rot_sc_in_for_lin_vac
linear.setpoint						
rotary.setpoint						
f_walk						
f_robot_secured						
f_ok_to_walk						
f_move_direction						
fc_linear_vacuum	ON					
fc_rotary_vacuum				ON		
fc_linear_sc	OUT	IN	IN			
fc_rotary_sc				OUT	IN	IN
mp_state_time	100	100	100	50	100	50
State Links	linear_sc_out	linear_sc_in	lin_sc_in_for_rot_vac	rotary_sc_out	rotary_sc_in	rot_sc_in_for_lin_vac
command						
walk_start						
walk_end	1. !f_walk	1. !f_walk	1. !f_walk	1. !f_walk	1. !f_walk	1. !f_walk
linear_move_max		3. mf_state_time && !f_move_direction			2. mf_state_time && f_move_direction	
linear_move_min		2. mf_state_time && f_move_direction			3. mf_state_time && !f_move_direction	
linear_vacuum_off			2. fs_rotary_vacuum	2. mf_state_time && fs_rotary_vacuum		
rotary_vacuum_off	2. mf_state_time && fs_linear_vacuum					2. fs_linear_vacuum
linear_sc_out						
linear_sc_in						
lin_sc_in_for_rot_vac				3. mf_state_time && !fs_rotary_vacuum		
rotary_sc_out						
rotary_sc_in						
rot_sc_in_for_lin_vac	3. mf_state_time && !fs_linear_vacuum					

Figure 24b. Continuation of ALMR robot state network

extend both the linear and rotary suction cups away from the robot chassis. Vacuum was applied to both sets of suction cups as indicated by the ON parameter assigned to both the *fc_linear_vacuum* and *fc_rotary_vacuum* control flags. This allowed lab personal time to mount the robot on the wall.

The *walk_start* state had links to three other states. The first link was to the *walk_end* state. If the *f_walk* flag evaluated false, (!*f_walk* evaluated true), then the *walk_end* state was activated. Notice that every state except the command and *walk_end* states tested the status of the *f_walk* flag. At any time during the walking sequence, the user could stop the robot by setting the *f_walk* flag to false. If the *f_walk* flag was still true, then the second link was evaluated. This link pointed to the *linear_sc_out* state. Actually, links 2 and 3 were very similar and evaluated the status of three flags in the data base. The first flag, (*f_ok_to_walk*), was used to signal that the robot had been fixed to the wall and was ready to climb. The *f_move_direction* flag indicated the direction of the desired robot travel. The last flag was a result from an input driver. This input driver determined when a sufficient vacuum had been achieved on the linear and rotary suction cups. Link three was different from link 2 because of the logical not (!) symbol in front of the *f_move_direction* flag. After a vacuum was sensed on both the linear and rotary suction cups and the *f_ok_to_move* flag had been set, the *linear_sc_out* state or the *rotary_sc_out* state was activated depending on the status of the *f_move_direction* flag. Assuming that the *f_move_direction* flag was true, the *linear_sc_out* state was activated.

When the *linear_sc_out* state (Figure 24b, second column) was activated, the initialization for this state altered the linear vacuum control flag, (*fc_linear_vacuum*), to ON and the linear suction cup control flag, (*fc_linear_sc*), to OUT. The model parameter *mp_state_time* was changed to 100. After the state was initialized, the links were evaluated. The second and third links from this state evaluated the result from the state time model, (*mf_state_time*), and the result from an input driver that sensed the status of the linear vacuum, (*fs_linear_vacuum*). After the *linear_sc_out* state had been active for one second, (100 milliseconds), the state time model set the *mf_state_time* parameter to true. If there was a vacuum in the linear suction cups, the *rotary_vacuum_off* state was activated. If a linear vacuum was not sensed, the *rot_sc_in_for_lin_vac* state was activated. This state was added to correct a problem with the original robot walking program. It was noticed that at times, the linear suction cups were not close enough to the wall to allow a vacuum to be pulled in the linear suction cups. This was due to a manufacturing error on the rotary piece. The *rot_sc_in_for_lin_vac* state pulled the rotary piece into the robot chassis to assist in setting the linear suction cups on the wall. However, assuming that a vacuum had been obtained in the linear suction cups during the *linear_sc_out* state, the *rotary_vacuum_off* state was activated. The initialization of this state turned off the vacuum supply to the rotary suction cups. After 50 milliseconds and there was not a vacuum on the rotary suction cups, the *rotary_sc_in* state was activated. This state moved the rotary suction cups in and activated the *linear_move_max* state after one second. The *linear_move_max* state changed the control flag for the linear motor driver and the linear slider moved toward the maximum position limit. Once the maximum limit was reached, the *rotary_sc_out* state was activated. The sequence for linear walking can be mapped out by following the links in the state network. Utilizing the program given in Figure 24, the ALMR robot could walk up and down a wall.

Conclusions

The programming language was well suited for robot program development. The programming language provided a consistent framework for creating, modifying, and debugging a program. The consistent interface between the language and input and output drivers provided the flexibility to port the language to other robotic systems. The modularity of the programming language enabled problem solutions to be quickly added without major modifications to the remaining program.

Definition of models provided a method to assimilate environmental and system information. Use of model results in decision making provided a consistent method to program new state links. When additional intelligence was programmed to solve a problem, results from existing, working models could be used in state links. If the modeling logic was later changed, the logic only had to be changed in one location; every state link that used the modeling logic did not have to be altered.

5.0 ARTICULATED TRANSPORTER/MANIPULATOR SYSTEM (ATMS) DEVELOPMENT

This section describes the development that has been done on the Articulated Transporter/Manipulator System, or ATMS. The following work was accomplished in 1993.

- A new fabrication material was chosen to reduce the overall weight of the ATMS while still retaining the required strength and stiffness.
- A partial redesign of certain ATMS components was necessary to resolve manufacturing difficulties introduced with the new fabrication material.
- A test stand was designed and constructed to support multiple ATMS segments for testing.
- A graphical user interface was developed to control the new test apparatus.
- A study was performed to determine the feasibility of using electric actuators in place of the current hydraulic actuators.
- A study was performed to develop path planning algorithms for use in the horizontal navigation of the ATMS.

Each of these topics is briefly discussed in the following paragraphs.

5.1 Introduction of a New Fabrication Material for ATMS

Development of Composite Hardware

The preliminary design modifications generated by the evaluation of the 1991 prototype ATMS segment and developed in 1992 were implemented in 1993. Fabrication of major carbon fiber composite components continued through 1993, with modifications to tooling as manufacturing techniques developed. One final major component remains to be fabricated, the ring, but all other major components are built. A test stand that allows four articulated segments to be evaluated undergoing vertical navigation was designed and fabricated. The 1991 prototype segment has been mounted in the vertical navigation test stand.

1993 Part Fabrication

Layout. Figure 25 illustrates two linked segments of the ATMS 1992 design revision. Figure 26 illustrates a sectional view of one body. Four body tubes have been fabricated. The slide configuration is shown, along with actuator mounting details. Several parts have been broken into assemblies of smaller parts to facilitate fabrication. The slide is the most notable of these. It has been separated into four components that are assembled with threaded fasteners. Actuator mounts have been designed and fabricated. These are complex carbon fiber composites with bronze bearing surface inserts and aluminum mounting bosses. The ring is not shown and is still undergoing mold design revisions.

Figure 25. Segment assembly.

Figure 26. ATMS section layout.

Materials. The principal mechanical components of the ATMS 1992 design revision are fabricated from a carbon fiber - vinyl ester composite. The carbon fiber utilized is Hercules IM7, an intermediate modulus high strength fiber. The fiber is woven into a 5 harness satin weave fabric by the supplier. The manufacturing process impregnates the dry carbon fabric with catalyzed vinyl ester resin and molds the resulting composite to the desired shape. The ultimate tensile strength of the IM7 fiber is approximately 720,000 psi. The tensile modulus is between 38 and 42 million psi. The strain at failure is 1.7%. The strain at failure is of importance because it determines impact resistance, a weakness of carbon fiber composites. IM7's strain at failure is relatively high when compared with other commercially available carbon fibers. The resulting composite fabricated has an estimated ultimate tensile strength of 75,000 psi and an estimated tensile modulus of 8 million psi.

Several of the parts manufactured include metal inserts to allow mechanical fasteners to be used for assembly and to facilitate joint motion between components. Bearing surfaces are of bronze and are simply inserted in the wet composite during fabrication. Mechanical fastening bosses are aluminum and are inserted in the wet composite during fabrication and then pinned into the composite with shear pins. The shear pins carry the majority of the interfacial loading between the aluminum and the composite so that the adhesive bond is not critical.

Body. The external structure of the segment is called the body. Four of these tubes have been fabricated from vinyl ester - carbon fiber composite. The mounting holes for the slide

bearing blocks, the hydraulic manifolds, the actuators, and the proposed wheels are molded into the body. The body is a 16 sided tube. Figure 27 shows the body. The molded-in holes have random E glass chopped mat placed in the laminate to improve elastic response to stress concentrations. This, along with carbon fibers displaced by the hole molding procedure, result in local increases in thicknesses near holes.

Actuator mounts. The actuator mounts fix the hydraulic actuators to the body. They provide a bearing surface for the trunnion of the actuator and a threaded aluminum insert that allows the actuator mounts to be fixed to the body by threaded fasteners. The actuator mounts are illustrated in Figure 27. Incorporation of the aluminum insert required the development of a fabrication method that results in both an adhesive and mechanical joint between the carbon fiber composite and the aluminum insert. This was accomplished by incorporating pins fixed in both the insert and the composite oriented in such a way as to transmit a shear load between the insert and the composite. The pin's size is held small to reduce fabrication difficulties with fiber bunching.

Figure 27. Body and actuator mounts.

Slide. The primary change in the slide is related to fabrication difficulties perceived in molding a single piece slide. The slide was broken into two pairs of symmetric parts. This allows two simple molds to be used to fabricate the slide parts, as opposed to one very complex mold for a single piece slide. The resulting assembly gives some flexibility to the assembly of the segment by allowing each slide to be adjusted to fit its corresponding recirculating ball bearing blocks. The slide assembly has bronze bearing surfaces incorporated during fabrication for assembly of the slide to ring joint. Figure 28 shows the slide assembly and related parts.

ATMS Slide Assembly

Figure 28. ATMS slide assembly.

Ring/Cross. This part only existed conceptually in the 1991 segment. Substantial work has been done in designing a rigid connection that does not interfere with the slides or bodies to which it is attached. The ring incorporates connection points for the actuators and slides. It also includes mounting for the encoders that determine joint angle. The absolute encoders procured for the ATMS are of rather large geometry, leading to a significant packaging effort.

5.2 Development Of A Test Environment

A new test apparatus was designed and constructed. This new apparatus was intended to support multiple ATMS segments in a segment chain so that new motion and sensor algorithms could be designed and tested. In addition, a new user interface was designed to give the operator direct, real-time control of the segment chain as well as provide a graphical representation of the real-time configuration of the chain.

The test apparatus is pictured in Figure 29. It was constructed entirely out of steel and was designed to support a total of four ATMS segments. The apparatus is quite large and very heavy - a necessary design feature to withstand the relatively large forces and moments created as the ATMS segments perform their commanded movements.

Figure 29. Test stand for ATMS

The user interface and graphical animation were developed on a Silicon Graphics workstation. Both features were incorporated into one environment. The purpose of the environment was to allow the user to remotely operate the test apparatus while observing the commanded motions as real-time animations on the computer screen. In addition to the animations, the operator also has control of a video camera that provides color video images of the apparatus at all times.

The user interface was developed using a library of C++ objects in a package called Motif. The library consists of graphical objects that can be placed on the screen to give the user an easier and more intuitive method of interaction. These objects include buttons, slide switches,

thumb wheels, and other facilities that the user can activate with mouse movements and button presses. Using these objects allows programmers to concentrate less on coding the user interface and concentrate more on coding the tasks the application was designed to implement. The objects also provide a professional looking user environment with little programming effort.

The graphical animation was developed using Inventor. Inventor is a library of C++ objects that were created by Silicon Graphics to facilitate graphics programming. Inventor provides the programmer with a convenient method of representing graphical elements such as cubes, spheres, cones, and lines. Inventor also provides a simple means of combining basic graphical elements into relatively complex images and animations.

The software developed for the ATMS project uses both Motif and Inventor to create an intuitive environment from which the ATMS test apparatus can be observed and controlled. The apparatus itself was designed to consist of the test stand as well as four ATMS segments connected in a serial chain. The actual apparatus hardware is controlled by a Force computer running the OS/9 real-time operating system. The hydraulic cylinders are controlled via servo-amplifiers and servo-valves while their actual positions are determined by linear transducers and analog to digital converters. The Silicon Graphics and the Force are linked through an ethernet connection. Communication between the two computers was accomplished using a library of routines called Helix. Helix allows both computers to share a portion of their physical memory space. Using Helix the computers can communicate by reading and writing from this shared memory. The Silicon Graphics communicates the desired positions of the cylinders while the Force communicates the actual cylinder positions.

Control of the apparatus from the Silicon Graphics is done completely on-screen. The user may control individual segments by simply pointing the cursor at those segments and pressing the mouse button. At this point a control window is created that displays the current segment length as well as the segment angle relative to it's preceding segment. These parameters are changed by using the slide bars to command a new length and/or angle. These new lengths and/or angles are sent to the Force to be translated into motions on the apparatus. At the same time, the Force is taking the actual lengths and/or angles and communicating them to the Silicon Graphics for display on the operator's console.

5.3 Electric Linear Actuator Study

A study was performed to determine the feasibility of using electric linear actuators in place of the current hydraulic actuators. The results and observations of the study are reviewed here. A brief comparison of the two actuation methods is presented in Table 2.

Table 2. Comparison of Actuation Methods

Attributes	Hydraulic Actuators	Electrical Actuators
Power	Inherent high power to actuator volume ratio.	Difficult to produce high power to actuator volume ratio.
Efficiency	Low efficiency due to pump and valve losses.	Inherently good efficiency.
Centralized Equipment	Requires a heavy, centralized on-board pump.	No heavy, centralized equipment required.
Umbilical Requirements	Requires electric power supplied by umbilical.	May use on-board battery or umbilical electric power.
Cooling Incorporation	Cooling may be incorporated with pump.	Additional cooling apparatus may be required.
Throw To Actuator Length Ratio	High throw to actuator length ratio.	Low throw to length ratio associated with large loads.
Overall Weight	Heavy.	Relatively light.

It is possible that linear electric actuators could perform the same task in the ATMS as the current hydraulic actuators with a smaller associated cost in terms of weight and overall system performance. It is likely that the successful actuator design will employ a ball screw drive mechanism or a variation thereof. However, the requirements for the ATMS actuator far exceed the capabilities of most off-the-shelf linear electric actuators. Therefore, the ATMS linear electric actuator will have to exhibit near cutting-edge performance in order to satisfy its requirements.

The design of the actuator will depend heavily upon the specification of an effective average actuator load which will require dynamic modeling of the ATMS as well as the specification of the average tour of duty for the vehicle. Most notably the requirement that the horizontal and vertical axis actuators should both be capable of very high loads will probably have to be abandoned if the actuator designs are to remain feasible. This is because horizontal navigation will require high-speed, low-thrust actuators while vertical navigation will require low-speed, high-thrust actuators--combining the two requirements would likely necessitate a variable reduction drive in the horizontal actuator despite already tight space allowances.

It is entirely possible that the actuator required by these specifications is outside the envelope of current technology. The ATMS specifications require six times the thrust be packaged into a volume 5.6% shorter than the Warner Electric actuator, a popular off-the-shelf component. However, if the specifications for the actuators were relaxed to allow for a slow, high-thrust vertical axis actuator and a fast, low-thrust horizontal axis actuator a successful design would be more likely.

5.4 Horizontal Navigation and Path Planning

The new design of the Articulated Transporter/Manipulator system, or ATMS robot, was developed. It is comprised of twelve individual segments which provide both locomotion and maneuverability. In this new version of the design, the ATMS robot can stretch its cylinders to change its length and orientation. This will give it more flexibility in terms of navigation, but will also increase its motion control complexity. In order to control all the movements of this new ATMS robot, a control algorithm has been developed and a computer simulation was created.

By using this simulation, all the motion analysis of vertical and horizontal navigation of the ATMS robot can be performed. The simulation is presented in two separate modules. The first is a motion demonstration module, illustrating the motion capabilities of the ATMS robot. These capabilities include navigation in the horizontal plane, crossing horizontal gaps of up to 11 feet, and vertical ascension/descension of up to 10 feet. The second module is a path planning simulation. In this module, an early path-planning algorithm which had been developed for a previous ATMS design was modified to accommodate the robot design changes. The simulation animates the 12-segment ATMS robot and moves it along a path which is planned in a virtual environment constructed by the simulation. The path is based on information gathered from the virtual environment. The simulation was developed on a Silicon Graphics IRIS R4000-50 workstation.

6.0 REFERENCES

- Adsit, P. D. 1989. Real-Time Intelligent Control of a Vision-Servoed Fruit-Picking Robot. Unpublished Ph. D. dissertation. Agricultural Engineering Department, University of Florida, Gainesville.
- Blaha, J. R., J. P. Lamoureux, and K. E. McKee. 1986. Higher order languages for robots. Manufacturing Technology Information Analysis Center, contract DLA-900-84-C-1508 for the Department of Defense. Reproduced by National Technical Information Service, U. S. Department of Commerce, Springfield, VA 22161. 129 pages.
- Comer, D. E. 1991. *Internetworking with TCP/IP Volumes I and II*. Prentice-Hall, Englewood Cliffs, New Jersey.
- Gamble, D., S. C. Ridgeway, P. D. Adsit, and C. D. Crane. 1994. Inspection Robot for an Advanced Liquid Metal Reactor," in *Proceedings of the Seventh Annual Conference on Recent Advances in Robotics*, University of Florida, Gainesville, FL, April 18-19.
- Jones, J. P., P. L. Butler, S. E. Johnston, and T. E. Heywood. 1992. "Hetero Helix: Synchronous and asynchronous control systems in heterogeneous distributed networks," *Robotics & Autonomous Systems*, (10) 85-99.
- Ridgeway, S. C., C. D. Crane, P. D. Adsit, and R. C. Harrell. 1992. "The Mechanical Design of a Parallel Actuated Joint for an Articulated Mobile Robot," in *Proceedings of the 1992 ASME Design Technical Conferences-22nd Biennial Mechanisms Conference*, Scottsdale, AZ, Sept. 13-16.
- Ridgeway, S. C., P. D. Adsit, and C. D. Crane. 1994. "Design and Construction of an Articulated Transporter/Manipulator System using Composite Materials," in *Proceedings of the Seventh Annual Conference on Recent Advances in Robotics*, University of Florida, Gainesville, FL, April 18-19.
- Stevens, W. R. 1990. *UNIX Network Programming*. Prentice-Hall, Englewood Cliffs, New Jersey.

AIM

Association for Information and Image Management

1100 Wayne Avenue, Suite 1100
Silver Spring, Maryland 20910
301/587-8202

Centimeter

Inches

MANUFACTURED TO AIM STANDARDS
BY APPLIED IMAGE, INC.

DATE

FILMED

12 / 19 / 94

END