59-63

44784

A PROTOTYPE SUPERVISED INTELLIGENT ROBOT FOR HELPING ASTRONAUTS

J. D. Erickson, K. A. Grimm, and T. W. Pendleton
Automation and Robotics Division/ER
NASA-Lyndon B. Johnson Space Center

ABSTRACT

The development status is described of a prototype supervised intelligent robot for space application for purposes of (1) helping the crew of a spacecraft such as the Space Station with various tasks such as holding objects and retrieving/replacing tools and other objects from/into storage, and for purposes of (2) retrieving detached objects, such as equipment or crew, that have become separated from their spacecraft. In addition to this set of tasks in this low Earth orbiting spacecraft environment, it is argued that certain aspects of the technology can be viewed as generic in approach, thereby offering insight into intelligent robots for other tasks and environments.

Also described are characterization results on the usable reduced gravity environment in an aircraft flying parabolas (to simulate weightlessness) and results on hardware performance there. These results show it is feasible to use that environment for evaluative testing of dexterous grasping based on real-time visual sensing of freely rotating and translating objects.

INTRODUCTION

Numerous facets contribute to achieving robotic intelligence. This paper, based on a more complete presentation in [1], describes many of these facets and attempts to relate them to the central theme of a software architecture that enables a sufficient level of robotic intelligence, and thus, real work in real environments under supervision by exception.

The essence of intelligent systems is that they are capable of collecting and applying knowledge of the situation gained at execution time and correlating it with other knowledge to take effective actions in achieving goals. Intelligent systems are composed of sensors for perceiving both the external and internal environments, effectors for acting on the world, and computer hardware and software systems providing intelligent connection between the sensors and effectors.

Part of the processing by these computer systems is symbolic in a nonnumeric sense and thus enables practical reasoning, or the behavior which in humans we call intelligent. The intelligent system we will be addressing, the Extravehicular Activity Helper/Retriever (EVAHR), is a supervised, intelligent, mobile robot with arms and end effectors Intelligent robots of this nature are required for long-term operations in space and are mandatory for space exploration to improve safety, reliability, and productivity, while enabling large cost savings through minimizing logistics [2].

PROBLEM STATEMENT

Long-term space operations such as the Space Station have requirements for capabilities for rescue of EVA crew and retrieval of equipment. A space station cannot chase separated crew or equipment, and other vehicles such as the Space Shuttle will not usually be available. In addition to the retrieval of drifting objects, another need is for robotic help to EVA crewmembers in various tasks such as holding objects; retrieving and replacing tools and other items from and into storage; performing inspections; setting up and dismantling work sites; performing servicing, maintenance, and repairs; and deploying and retrieving payloads. Modeling, simulation, and analysis studies of space exploration missions have shown that supervised intelligent robots are enabling for human exploration missions [3, 4].

The free-flying, supervised intelligent robot called EVA Helper/Retriever (EVAHR) is being prototyped as a potential solution to the crew helper and detached crew and equipment retrieval need. EVAHR is a technology test-bed providing evaluation and demonstration of the technology included for the following three purposes: 1) Robotic retrieval of objects which become detached from their spacecraft; e.g., astronauts adrift from the Space Station. 2) A robotic crew helper around a spacecraft; e.g., inspector, "go-fer," holder, maintainer, servicer, tester, etc. 3) A "generic" prototype supervised, intelligent autonomous robot (for planetary surfaces with different mobility such as wheels or tracks and for terrestrial applications with appropriate adaptations).

Early supervised intelligent robotic systems with initial capabilities to meet real needs are beginning to emerge from laboratories and manufacturers. It is now possible, in our opinion, to construct robots capable of accomplishing several specific high level tasks in unstructured real world environments.

The ability to acquire and apply knowledge and skills to achieve stated goals in the face of variations, difficulties, and complexities imposed by a dynamic environment with significant unpredictability is our working definition of "robotic intelligence." This does not require a broad-based general intelligence or common sense by the robot. However, doing the work needed to accomplish goals does require, in general, both mobility and manipulation in addition to reacting, or deciding "intelligently" at each step what to do. Further, supervised intelligent robots are required for human-robot teams where supervision is most naturally provided by voice.

Certain aspects of the EVAHR technology, which provide the capability for performing specified tasks in a low-Earth-orbiting spacecraft environment, can be viewed as generic in approach, thereby offering insight into intelligent robots for other tasks and environments. This is because the design of the software architecture, which is the framework (functional decomposition) that integrates the separate functional modules into a coherent system, is dictated in large measure by the tasks and nature of the environment. And because both the goal-achieving tasks and the partially unpredictable nature of the environments are similar on Earth and in space, the software architecture can be viewed as generic - as can many of the software modules, such as the AI planner, world model, and natural language interface. Other software is bundled with certain hardware. This leads to the concept of a modular, end-user customized robot, put together from modules with standard interfaces [5-7] such as users do with a personal computer, yet maintaining real-time response.

APPROACH

The end goal for intelligent space robot development is one or more operational robots as part of human/robot teams in space. Prior to that, an evaluation of performance in space will be required.

Our approach to development of operational robots as part of human-robot teams in space is a systems engineering approach with an iterative, three-ground-phase requirements prototype development, tested in both ground and aircraft simulations of space, followed by evaluation testing of a flight test article in space. We adapt and integrate existing technology solutions.

The EVA Helper/Retriever ground-based technology demonstration was established to design, develop, and evaluate an integrated robotic hardware/software system which supports design studies of a space borne crew rescue/equipment retrieval and crew helper capability. Goals for three phases were established. The Phase I goals were to design, build, and test a retriever system test-bed by demonstrating supervised retrieval of a fixed target. Phase II goals were to enhance the test-bed subsystems with significant intelligent capability by demonstrating arbitrarily oriented target retrieval while avoiding fixed obstacles. The objectives for Phase III, which is currently in progress, are to more fully achieve supervised, intelligent, autonomous behavior by demonstrating grasp of a moving target while avoiding moving obstacles and demonstrating crew helper tasks. Phase III is divided into two parts. Phase IIIA goals are to achieve real-time complex perception and manipulator/hand control sufficient to grasp moving objects, which is a basic skill both in space retrieval and in accomplishing the transition from flying to attaching to a spacecraft. Phase IIIB goals are to achieve a software architecture for manipulation and mobility, with integrated sensing, perception, planning, and reacting, which guarantees safe, robust conduct of multiple tasks in an integrated package while successfully dealing with a dynamic environment.

Our overall testing approach is short cycle run-break-fix with increasing integration and more relevant environments; such an approach finds design and implementation problems early when they are lowest cost to fix.

The performance characteristics of the EVAHR hardware enable (or defeat) the "intelligent" behavior of the robot as "animated" by the software. We are testing only a subset of the Phase IIIB hardware in Phase IIIA.

The hardware subset includes a seven degree of freedom arm (Robotics Research K807i), a five degree of freedom, compliant, force-limited dexterous hand, a laser range imager (Perceptron), a stereo video camera system (Teleos Prism 3), a pan/tilt unit, a 700 Megaflop computational engine employing i860s and transputers, and an Inertial Measurement Unit (IMU) of accelerometers and gyros.

During Phase IIIA we are using a subset of our reaction plan architecture while we are exploring two new approaches to the software architecture for Phase IIIB. The first is a version of the 3-tiered, asynchronous, heterogeneous architecture for mobile robots [8-10] adapted to include manipulation. The second is a version of the SOAR architecture [11] applied to robots [12]. SOAR is of interest because of its capabilities in learning, including recent work in situated, interactive natural language instruction [13].

For each approach we are conducting evaluation testing of minimal prototype architecture implementations to obtain some evidence of their strengths and weaknesses for our tasks before selecting one for larger scale implementation in Phase IIIB.

Safety is a major issue in human-robot teams, especially in space. Since robotic motion control programs cannot be considered safe unless they run in hard real time, an approach which addresses this issue in a different manner from that of the 3-tiered architecture is needed for comparative evaluation. We are pursuing the development of one such approach [14].

The pivotal problem in successfully coupling symbolic reasoning with the ability to guarantee production of a timely response is that the timing of actions taken by a real-time system must have low variances, so that the effects of those actions on unfolding processes can be predicted with sufficient accuracy. But intelligent software reserves the option of extended searching, which has very high variance. Therefore, when building a system that must act in real time as well as reasoning, one can choose to either 1) Subject the AI component of the system to hard deadlines. This effectively embeds the AI reasoner within the real-time system, and under time pressure, results in loss of intelligent function. 2) Refuse to subject the AI component of the system to hard deadlines, and have the real-time subsystem "do its best" with whatever commands the AI subsystem can generate in time. This effectively embeds the real-time subsystem within the AI system, and under time pressure, results in loss of timely control. 3) Refuse to subject the AI component of the system to hard deadlines, but let the AI components "negotiate" with the real-time subsystem to obtain a feasible schedule for task execution. This does not embed either subsystem within the other, and with proper selection of the real-time executive's task schedule, has the promise of remaining functional under time pressure. The 3-tiered approach is a category three approach, whereas we interpret SOAR to be a category two approach.

We can now summarize the state of the art. Simple control systems can get away with seeming to be "fast enough," but that approach becomes potentially very dangerous in more complex systems, particularly in intelligent systems where the set of tasks being executed changes over time. In a system that may perform any subset of N possible tasks, there are 2^N possible combinations of tasks, and it becomes impossible to test the performance of each combination by hand when N is large. Therefore, it becomes imperative to have automated support for obtaining a guarantee that the system can always perform in hard real time.

THREE-TIERED SOFTWARE ARCHITECTURE

Combining all prior knowledge and knowledge sensed during a task requires that planning in advance can only be guidance, with control decisions as to what to do postponed until such time as the situation is being sensed and the task is being executed. This is the essence of Agre and Chapman's theory of plans-as-advice [15], and is a design principle underlying the 3-tiered approach. The three tiers are the planner, the sequencer, and the reactive controller. The responsibility of the planning layer is to determine which tasks would accomplish the goal, and in what approximate order. Thus, the planning layer forms a partially ordered set of tasks for the robot to perform, with temporal constraints. The AI planner which we are evaluating for this application is the AP Planner [16]. It may be possible to use SOAR for this application.

The sequencing "middle" layer is responsible for controlling sequences of primitive physical activities and deliberative computations. Operating asynchronously from the planner, yet receiving inputs from that layer, the sequencer takes the sketchy plan and expands it, based on the current situation. Thus, the hierarchical plan expansion happens at execution time rather than at the deliberative stage. To implement the sequencer, data structures called Reactive Action Packages (RAP's) are used to represent tasks and their methods for executing [9].

At the lowest level, the reactive controller accepts sensing data and action commands, sensorimotor actions that cannot be decomposed any further, from the sequencer. For example, "move," "turn," or "grasp" are all examples of action commands that are passed on to the hardware. The reactive controller also monitors for success or failure of these commanded activities.

PHASE IIIA RESULTS TO DATE

Results from Phase II have been reported previously [17]. Some preliminary results from Phase IIIA have also been reported [18-23].

SOAR Evaluation for Phase IIIB

SOAR was selected for study as a promising candidate system for the EVAHR planning system. SOAR is a symbolic AI architecture which emphasizes problem-solving, planning, and learning.

One major advantage of SOAR is its ability to learn by taking a new experience, and saving the sequence of steps to the goal as a "chunk." This chunk is in the form of a set of production rules, and if the same scenario is encountered in the future, the associated chunk will execute without having to search for the correct sequence as it did initially.

From our experience with Hero-SOAR, a subset of SOAR for a Hero robot, we know that the reactivity of SOAR is an important capability needed to respond to the environment quickly. SOAR may be seen as a system with a planner, which plans in the traditional sense yet with no actual data structure produced; a mechanism to execute the plan; and a fast replanning ability.

Phase IIIA Computer Simulation Results

Software modules for grasping of free-floating objects in a zero-g, 6-DOF environment have been described in previous sections. Results of performance testing of these modules as subsystems are described in this section. The modules have also been integrated and tested in our orbital and KC-135 simulations[24], and these results are also described below.

Search is the first visual function to be performed when there is no knowledge about the location of an object of interest. It is carried out as follows[25,26]: EVAHR's front hemisphere is divided into concentric "rings," and each ring is further divided into sectors, each of which is enclosed by the FOV of the sensor. Each search starts from the center ring and spirals outward until an object is found.

Algorithms for image-based pose estimation have been implemented. Several objects were chosen for testing. These objects include some orbital replaceable units (ORU), a star tracker, a jettison handle, and some wrenches.

To test the robustness of the software, 500 tests were run on each test object with actual poses of the object randomly oriented using a random number generator in (simulated) images. Noise was added to the "range" component of the image to test the sensitivity of the algorithms to noise. There were two indications from the test results: 1) Most estimation errors are less than 5 degrees (with up to 3 percent noise in range); 2) The performance of the pose estimation software gradually degraded with increasing noise in range measurements.

The rotational state estimator uses intermittent delayed poses from the pose estimator software to provide the arm trajectory planner with current estimates of the target's rotational state at the rate of 100 Hz. The estimator utilizes an extended Kalman filter because of the inherent nonlinear nature of rotational dynamics. The effects of varying various parameters on the performance of the standalone rotational state estimator have been reported [19]. Testing on the integrated rotational state estimator shows it converges within 4 pose estimates (about 4 sec) and maintains error estimates of less than 3 degrees, which meets requirements.

The relative translational state estimator used for the KC-135 experiment does not use an inertial coordinate system. The equations describing the dynamics are nonlinear. Therefore, the estimator design is based on an extended Kalman filter. The results of its performance in the KC-135 simulator show an accuracy similar to that for the orbital case[27].

Integrated software testing in the orbital simulation has concentrated on and produced results in two areas: (1) determining the overall system performance against grasping different moving targets with random initial states and (2) determining the computational requirements for the pose estimation software, using rate and delay as parameters. Grasp impact dynamics calculations are made to verify that the target is not knocked away during the grasp or by a prior collision with the arm. Under these conditions, the system has achieved a >70-percent successful grasp rate for both objects tested. The state estimates have less than 1 inch and five degrees of error.

Results from the second suite of tests show that pose estimation rate and delay also have a direct effect on the time-to-grasp in successful tests. Assuming pose estimation rate and delay of 0.1 sec, we were able to estimate that six i860 processors would be sufficient to achieve these rates and delays.

AIRCRAFT REDUCED GRAVITY ENVIRONMENT

Some microgravity research can be conducted inside an aircraft simulating space by flying vertical parabolic flight paths, but only for very limited amounts of time. During Phase IIIA we are flying a subset of the EVAHR Phase IIIB hardware and software aboard the NASA Reduced Gravity Program's KC-135 aircraft. This aircraft flies a series of parabolic trajectories resulting in approximately 15 sec of near microgravity (<.01-g) in the cabin during each parabola. The robotic arm, hand, vision sensor with pan/tilt system, and IMU (Inertial Measurement Unit of accelerometers and gyroscopes) is attached to the floor of the aircraft. During microgravity, an object is released, tracked by the vision system, and grasped by the hand. All of these objects have a complex construction with multiple graspable points.

On several KC-135 preliminary flights, data characterizing the reduced gravity was collected from an IMU placed on the cabin floor. Video recordings also were made of objects floating during the reduced gravity interval. The vertical acceleration fluctuated significantly about zero-g. Fluctuations between 75 mg and -75 mg were commonplace. These fluctuations caused the released object to accelerate toward either the ceiling or floor of the airplane. Lateral accelerations were also observed and were due to air turbulence, flight path corrections, or other effects.

An evaluation of 38 parabolas was performed, and the trajectory duration determined. This interval started when the target was released and continued until the target hit the inside of the airplane fuselage, or was touched by personnel, or left the FOV of both video cameras. The results indicate 2/3 of the parabolas have 4 seconds or less of usable microgravity.

These results, especially the trajectory durations, do not match well with the extrapolation to the KC-135 of time-to-grasp results from the orbital simulation presented above.

In a separate flight of the KC-135, we exercised the unintegrated hardware subsystems (except the stereo cameras) independently. All of the hardware is designed to operate in a 1-g environment and might behave differently in the KC-135 in microgravity or after the 1.8-g pullout at the bottom of the parabolas. Motions and operations representative of those that will be used in later object tracking and grasping evaluations were used in these tests. All equipment was determined to operate without measurable changes in behavior from that expected.

CONCLUSIONS

The need for crew help and retrieval of detached crew and equipment in space has been identified. Evaluation of the practical realization of a potential solution has passed several successful milestones, but is still ongoing, with many of the critical developments yet to come. The potential solution described here is an initial attempt to build and understand a prototype of a supervised intelligent robot for use in space. It is also potentially useful in terms of the software architecture for many U.S. economy-related robot applications on Earth.

Both our Phase II and Phase IIIA results demonstrate that manipulation requires greater accuracy of sensing and perception than does mobility. Integrated testing with our Phase IIIA computer simulation has not only shown that we have a workable software design, but has also afforded us systems engineering analyses supporting computer hardware design for achieving real-time complex perception processing (sensor to percept) and grasp control (percept to action) for freely moving objects.

Our future plans are first to complete the metrology of the manipulator and joint calibration of both vision-system-manipulator pairs. We are recoding the laser scanner pose estimation software to run in real time on the i860 network. The tracker and translational state estimator are currently running in real time on i860's. The manipulator trajectory controller and grasp planner are running in real time on the transputer network. Grasp testing using targets mounted on the object-motion unit are being conducted in preparation for the KC-135 vision-guided grasping flights. Then we have several moving object grasp evaluation flights to conduct. Phase IIIB developments are dependent on the selection of a final software architecture from the preliminary prototyping efforts which are underway using a set of crew helper tasks, scenarios, and computer simulation environments with human-injected unpredictable events to assess the value of the many goal-planning and real-time reaction aspects of the supervised intelligent robot design.

ACKNOWLEDGMENTS

This effort is internally supported by the Center Director's Discretionary Fund and by the Automation and Robotics Division of the NASA Lyndon B. Johnson Space Center.

REFERENCES

1. J. D. Erickson et al., "An Intelligent Space Robot for Crew Help and Crew and Equipment Retrieval," International Journal of Applied Intelligence, accepted for publication in 1994.
2. J. D. Erickson et al., "SEI Planet Surface Systems Humans/Automation/ Robotics/Telerobotics Integrated Summary," document JSC-45100, NASA Johnson Space Center, Houston, TX, Mar. 1991.
3. J. D. Erickson, "Needs for Supervised Space Robots in Space Exploration," 1992 World Space Congress, IAF-92-0800, International Astronautical Federation, Paris, France, Aug. 1992.
4. J. D. Erickson et al., "Some Results of System Effectiveness Simulations for the First Lunar Outpost," 1993 AIAA Space Programs and Technology Conf., Huntsville, AL, Sept. 1993.
5. R. P. Bonasso and M. G. Slack, "Ideas on a System Design for End-User Robots," in Proc. SPIE 1829 Cooperative Intelligent Robotics in Space III, Nov. 1992, p. 352.
6. R. O. Ambrose, M. P. Aalund, and D. Tesar, "Designing Modular Robots for a Spectrum of Space Operations," in Proc. SPIE 1829 Cooperative Intelligent Robotics in Space III, Nov. 1992, p. 371.
7. F. daCosta et al., "An Integrated Prototyping Environment for Programmable Automation," in Proc. SPIE 1829 Cooperative Intelligent Robotics in Space III, Nov. 1992, p. 382.
8. E. Gat, "Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Real-World Mobile Robots," in AAAI-92, Proc. 10th Natl. Conf. on AI, San Jose, CA, July 1992.
9. R. J. Firby, "Adaptive Execution in Dynamic Domains," Ph.D. Thesis, Yale University, 1989.
10. R. P. Bonasso, "Using Parallel Program Specifications for Reactive Control of Underwater Vehicles," Jour. of Applied Intelligence, Kluwer Academic Publishers, Norwell, MA, June 1992.
11. J. E. Laird, A. Newell, and P. S. Rosenbloom, "SOAR: An Architecture for General Intelligence," Artificial Intelligence, Vol. 33 , No. 1, pp. 1-64, 1987.
12. J. E. Laird et al. "RoboSOAR: An Integration of External Interaction, Planning, and Learning Using SOAR," Robotics and Autonomous Systems, Vol. 8, 1991.
13. S. B. Huffman and J. E. Laird, "Learning Procedures from Interactive Natural Language instructions," in Proc. 10th Intl. Conf. on Machine Learning, June 1993.
14. M. Schoppers, "Ensuring Correct Reactive Behavior in Robotic Systems," AIAA/NASA JSC Workshop on Automation and Robotics '93, NASA Johnson Space Center, Houston, TX, Mar. 24, 1993.

15. P. Agre and D. Chapman, "Pengi: An Implementation of a Theory of Activity," in Proc. AAAI, 1987, pp. 268-272.
16. C. Elsaesser and T. R. MacMillan, "Representation and Algorithms for Multiagent Adversarial Planning," MTR-91W000207, MITRE Corp, McLean, VA, Dec. 1991.
17. J. D. Erickson et al., Technology Test Results from an Intelligent, Free-Flying Robot for Crew and Equipment Retrieval in Space," Proc. SPIE 1612, Cooperative Intelligent Robotics in Space II, Boston, MA, November 1991.
18. M. L. Littlefield, "Adaptive Tracking of Objects for a Mobile Robot Using Range Images," in Proc. SPIE 1829, Cooperative Intelligent Robotics in Space III, Boston, MA, Nov. 1992, pp. 433-443.
19. L. Hewgill, "Motion Estimation of a Freely Rotating Body in Earth Orbit," in Proc. SPIE 1829, Cooperative Intelligent Robotics in Space III, Boston, MA, Nov. 1992, pp. 444-457.
20. K. A. Grimm et al., "Experiment in Vision-Based Autonomous Grasping Within a Reduced Gravity Environment," in Proc. SPIE 1829, Cooperative Intelligent Robotics in Space III, Boston, MA, Nov. 1992, pp. 410-420.
21. C. H. Chien, "A Computer Vision System for Extravehicular Activity Helper/Retriever, in Proc. SPIE Conference on Sensor Fusion and Aerospace Applications, Orlando, FL, Apr. 1993.
22. R. S. Norsworthy, "Performance Measurement of Autonomous Grasping Software in a Simulated Orbital Environment," in Proc. SPIE 2057, Telemanipulator Technology and Space Robotics, Boston, MA, Sept. 1993.
23. G. Anderson, "Grasping a Rigid Object on Zero-G," in Proc. SPIE 2057, Telemanipulator Technology and Space Robotics, Boston, MA, Sept. 1993.
24. R. S. Norsworthy, "Simulation of the EVAHR/KC-135 for Software Integration/Testing," Proc. AIAA/NASA Conf. on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, March 1994.
25. M. L. Littlefield, "Real-Time Tracking of Objects in a KC-135 Microgravity Experiment," Proc. AIAA/NASA Conf. on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, March 1994.
26. E. L. Huber and K. Baker, "Object Tracking with Stereo Vision," Proc. AIAA/NASA Conf. on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, March 1994.
27. L. Hewgill, "Motion Estimation of Objects in KC-135 Microgravity," Proc. AIAA/NASA Conf. on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, March 1994.