AN INTELLIGENT ROBOT FOR HELPING ASTRONAUTS

N94-30581

J. D. Erickson*, K. A. Grimm†, and T. W. Pendleton‡
National Aeronautics and Space Administration Lyndon B. Johnson Space Center
Houston, Texas 77058

207678
p. 13

Abstract

This paper describes the development status 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 (2) for purposes of 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.

Candidate software architectures and their key technical issues which enable real work in real environments to be accomplished safely and robustly are addressed. Results of computer simulations of grasping floating objects are presented.

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 vision of freely rotating and translating objects.

* Chief Scientist, Automation and Robotics Division, Member AIAA
† EVAHR Project Manager
‡ Head, Robotic Intelligence Section

1. Introduction

Numerous facets contribute to achieving robotic intelligence. This paper, based on a more complete presentation in reference 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. Related work by others is also outlined in reference 1. 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 for providing an 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 we humans 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 (see Figure 1). 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].

Long-term space operations such as the Space Station have requirements for capabilities for rescue of extravehicular activity (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

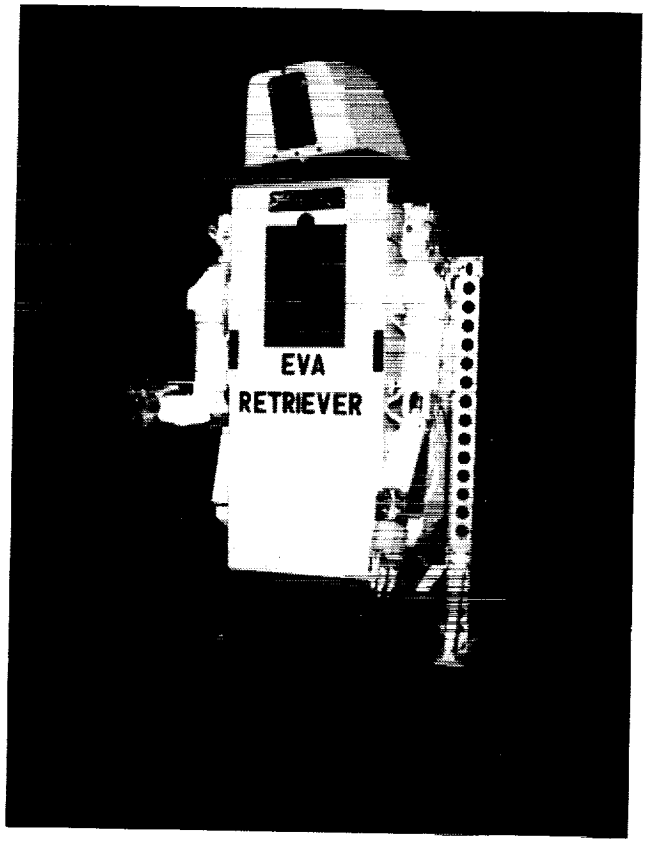

Figure 1. Phase II Retriever.

intelligent robots are enabling for human exploration missions[3,4].

The U.S. economy can reap major benefits from the development of supervised intelligent autonomous robotic systems[5,6], for such systems foster productivity improvements that raise the standard of living for everyone[7]. The solutions to the problems we will be solving to make the exploration of our solar system possible and practical will apply to the many critical problems we have on Earth which require operating in hazardous environments and to improving human productivity in many fields.

The free-flying, supervised intelligent robot called 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.

Controlling supervised intelligent robots having both mobility and dexterous manipulation is a challenge[1], as is integration of sensing and perception into planning and control in a robust way.

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[8-10] such as users do with a personal computer, yet maintaining real-time response.

2. 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 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 EVAHR 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. Table 1 summarizes some of the characteristics of the Phase II system. 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[11] with increasing integration and more relevant environments; such an approach finds design and implementation problems early when they are lowest cost to fix.

3. Hardware Design

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 7-degree of freedom (DOF) arm (Robotics Research K807i); a 5-DOF, 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 Intel i860s and transputers; and an Inertial Measurement Unit (IMU) of accelerometers and gyros.

4. Software Design

During Phase IIIA we are using a subset of the reaction plan architecture while we are exploring two new approaches to the software architecture for Phase IIIB. The first is a version of the three-tiered, asynchronous, heterogeneous architecture for mobile robots[12-14] adapted to include manipulation. The second is a version of the SOAR architecture[15] applied to robots[16]. SOAR is of interest because of its capabilities in learning, including recent work in situated, interactive natural language instruction[17]. To be practical, the robot "programming" bottleneck must be avoided by using learning from experience and instruction to acquire skills and knowledge. SOAR has also been used to achieve resource-dependent behavior[18] and to learn reactive, stimulus-response rules, in addition to search control.

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. We present our evaluation results on SOAR in the section on results. We are not far enough along on prototyping the three-tiered architecture to have results yet.

Table 1. Unique and Special Aspects of Phase II EVAHR.

- Prototype supervised, intelligent, autonomous robot
- Voice commands provide goals and directions
- Clips into space-worthy Manned Maneuvering Unit (MMU) which has flown from Shuttle
- "Flies" by propelling pressurized gas from MMU thrusters it controls
- Self-locating in analogy to space use of global positioning satellites where retriever uses camera, gyroscopes, and accelerometers
- Builds its own internal dynamic knowledge of its environment based on continuous sensory perception – No preprogrammed environmental model to which the environment must conform
- Planning/replanning based on goals and internal dynamic knowledge of its environment and constraints such as flight rules
  - Path planner for obstacle avoidance and rendezvous can reason in advance about the success of the mission
  - Actions are synchronized to events in the world through sensing of preconditions of planned actions
  - Deals with unpredictability by detection/replanning if needed
- Range image obstacle location and target tracking, orientation, and grasp location
- Acts to acquire knowledge about obscured target
- Maneuvers to optimize grasp success relative to target orientation
- Chooses between one-handed grasp and two-armed grapple, depending on target size it perceives
- Uses dexterous grasping with proximity sensors, compliant grasp, and force-limited grasp
  - Right hand has 5 proximity sensors
  - Left hand has 3 proximity sensors and 9 tactile sensors (3 per finger)
- Uses pressure sensors on chest for two-armed grapple of large targets
- Uses fourteen 10-MIPS transputers, six 68020 controllers, and one 80386 processor in a hierarchical, distributed architecture

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 three-tiered architecture is needed for comparative evaluation. We are pursuing the development of one such approach[19].

The following discussion is due to Schoppers[20]. A statement of the pivotal problem in successfully coupling symbolic reasoning with the ability to guarantee production of a timely response has recently been made: "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"[21].

The AI community has responded to this dilemma in roughly three ways[22]. When building a system that must act in real time as well as reasoning, one can choose to

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 three-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.

4.1 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[23], and is a design principle underlying the three-tiered approach.

Several researchers[12-14] have developed the three-tiered architecture to enable faster, more efficient interaction with the world and to allow the planner sufficient time to make intelligent decisions. Decisions based on the details of the local world are postponed and a "sketchy" plan is passed on to the next layer. The three layers 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. This plan is somewhat sketchy since not every detail of implementation, which would be determined by the current situation, is included. The AI planner which we are evaluating for this application is the AP Planner[24]. 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[13].

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 onto the hardware. The reactive controller also monitors for success or failure of these commanded activities.

4.2 Phases IIIA and IIIB Software Architecture

The EVAHR Phase IIIA software is composed of sensing, perception, world modeling, planning, and acting. Figure 2 shows the relationship among these elements for the on-orbit retrieval problem where a free-floating target must be rendezvoused with, grasped, and returned. As tasks are added to the crew helper's repertoire in Phase IIIB, additional elements must be added to support AI planning, force feedback arm control, and voice interaction with the crew.

Sensing software provides the low-level interface to the hardware sensors, reading and time tagging sensor data and providing preprocessing to account for the effects of nonideal sensors. Sensing software also provides an interface to perception.

Visual sensing software is the primary module for acquiring information about the environment via optical sensors such as the 10 image/sec laser scanner and the 30 dual-image/sec stereo vision system. Software for voice and data reception (Phase IIIB) handles speech recognition, Global Positioning System (GPS) decoding, and design and operations knowledge support system (DOKSS) interfacing. Software for force/torque sensing, proximity sensing, and tactile sensing provides data acquisition and time tagging.

Our proprioceptive sensing software reads and time tags the IMU accelerometers and gyroscopes, GPS, position sensors on the manipulators and hands, thruster firing sensors, position sensors on the pan/tilt unit, fault sensors throughout the hardware, and robot resource status sensors.

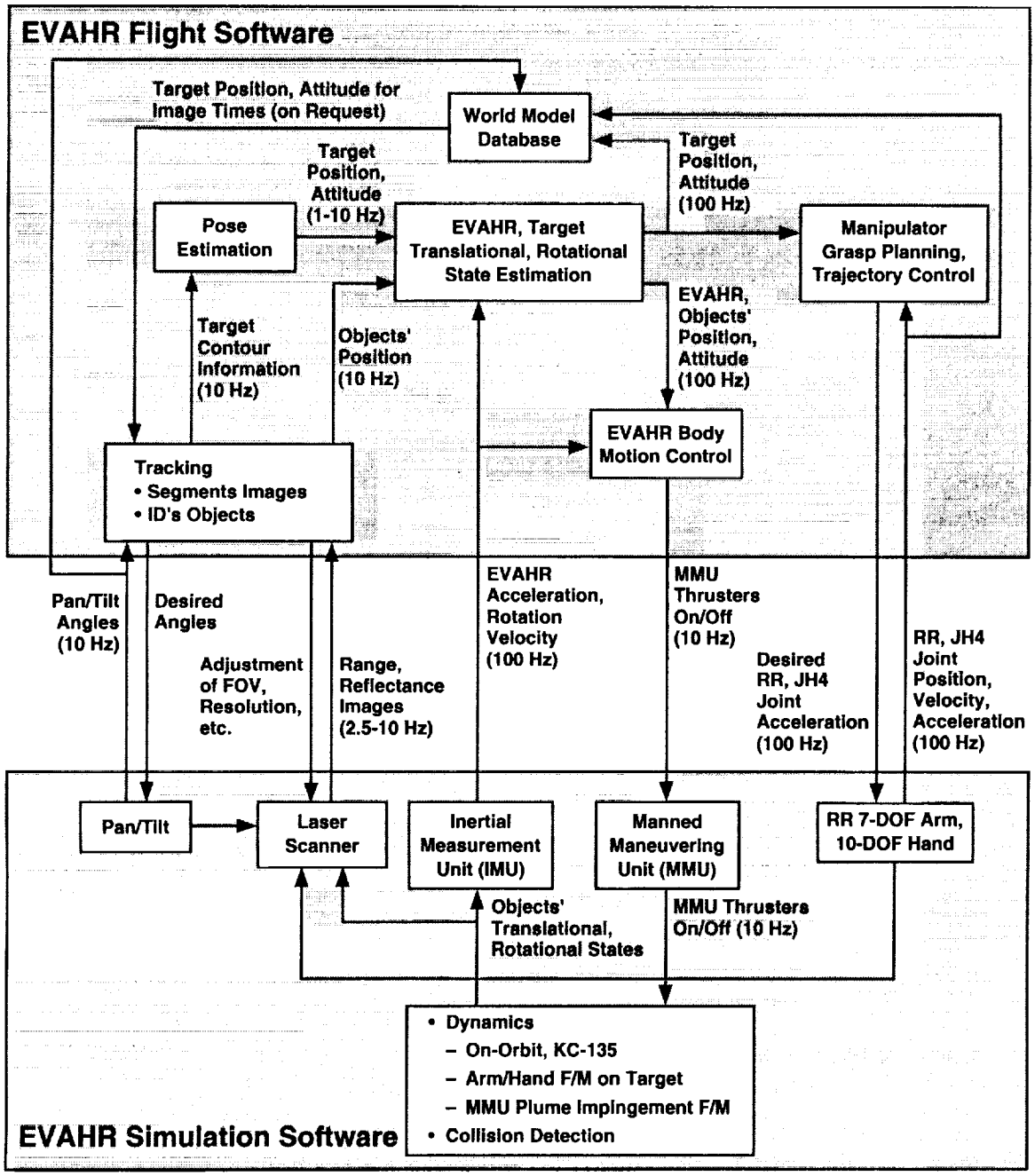

Figure 2. EVAHR Phase IIIA Flight/Simulation Software Architecture.

Perception software extracts understanding of the environment from preprocessed sensor and voice recognition receiver data.

Visual perception is carried out through a combination of various visual functions. Visual functions that have been implemented in software include search, tracking, and pose estimation. Other visual functions, such as those for object recognition, will be integrated in the near future. Pose estimation is calculating the orientation of an observed object in a given image. Our approach to pose estimation is known as image-based (or multiview based) pose estimation[25].

Natural language understanding processing (Phase IIIB) starts with a symbolic representation that Retriever can interpret and act upon, returning an appropriate response. Such systems are practical when limited to a specific domain and a well-defined application.

In general, our world model stores internal state representations of the external world at a

high level of abstraction, which allows the implicit predictions associated with the state (that it will remain valid for some time) to more likely remain valid for the lifetime of the internal state[26]. For moving objects, however, we use world model state estimators to bring the past measurements of motion descriptors up to the present time.

Planning enables the EVAHR to take a high-level goal and decide which subtasks must be accomplished to move the system to the goal. This selection and ordering of subtasks becomes very challenging, particularly if the system is monitoring the consequences of actions, replanning, or juggling multiple goals with changing priorities.

The vision system planner has been described previously[27].

A mobility planner is responsible for determining an optimal positional and rotational trajectory for the robot's body. "Optimal" usually implies (1) obstacle avoidance between the points of departure and arrival and (2) minimization of time, distance, and/or fuel consumption. In orbital scenarios (e.g., Space Station) fuel is at a premium, although with astronaut rescue, time is more critical. For this purpose, a trajectory planner/controller was developed based on the Clohessy-Wiltshire equations. This planner provides a minimum fuel or time trajectory between two moving bodies in orbit.

All of the tasks in Phase III require moving the manipulator in the presence of obstacles. Because many Phase III problems involve moving objects, potential field methods[28], which are very fast, are employed.

In Phase IIIA, the work for grasping a moving object is divided into two basic levels. A low-level high-bandwidth controller attempts to track a virtual grasp frame on the objects – but it steers clear of joint limits, obstacles, and singularities. A higher level grasp planner continually selects (heuristically) the best virtual grasp frame on the object to track. In Phase IIIB, the controller will be expanded to include guarded moves (where contact with a fixed object is expected), force and impedance control, and position control with their hybrids.

Speech planning software starts with an unambiguous message created from the internal representation and attempts to construct a meaningful sentence in response. This is then sent to speech synthesis hardware. Several general-purpose single-sentence generators of natural language are moving toward full-scale commercial strength[29] and real-time generation[30], with the latter a candidate for EVAHR use.

Acting software provides low-level controllers of motors and other actuators. One important feature in EVAHR's acting software is visually directed sensing. Sensor parameters such as the field of view (FOV), the focus of attention (via pan/tilt devices), or the data acquisition rate can be dynamically selected in order to acquire richer information about the environment or objects of interest[31].

5. Phase IIIA Results to Date

Results from Phase II have been reported previously[32]. Some preliminary results from Phase IIIA have also been reported[25,33-38]. Results from Phase IIIA consist of evaluations of software architectures such as SOAR, along with computer simulation results of various portions of the software capabilities, including results allowing an estimate of the central processing unit (CPU) and communications requirements to achieve realtime grasp of floating objects. Results from KC-135 tests of unintegrated hardware and software subsystems are also given.

5.1 SOAR Evaluation for Phase IIIB

SOAR[16] 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. It has been applied in numerous fields, such as education and training. As a production-based system, SOAR starts with an initial state of the problem and applies operators which make changes to the problem state to reach the goal state. Finding the sequence of operators to apply to the current problem state is the major challenge in its planning.

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.

5.2 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 the orbital and KC-135 simulations[39], and these results are also described below.

5.2.1 Phase IIIA Computer Simulation Results – Uncluttered Search

The 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[40,41]. 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. If an object is found, the search is terminated and the estimate of where the object is located is iteratively refined by adjusting the sensor gimbals toward the object and reducing the FOV until the object is centered and large in the image.

5.2.2 Phase IIIA Computer Simulation Results – Pose Estimation

Algorithms for image-based pose estimation have been implemented. Several objects were chosen for testing. These objects include some orbital replaceable units (ORU's), 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.

5.2.3 Phase IIIA Computer Simulation Results – State Estimation

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[34]. Testing on the integrated rotational state estimator shows it converges within four 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[42].

5.2.4 Phase IIIA Orbital Computer Simulation Results – Grasping Moving Objects

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 targets with random initial states and (2) determining the computational requirements for the pose estimation software, using rate and delay as parameters. In those tests, the following constraints hold: The target remains stationary in an optimal location for grasping; a grasp must be achieved in 15 sec. 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. The EVAHR inertial state is assumed known. In the random initial state test suite, the target rotates in 3 DOF starting from a random initial orientation and velocity. 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 5 degrees of error. An average time line of events in a typical successful grasp test is given in Table 2.

Table 2. Grasp Test Time Line.

Event	Time from start, sec
Translational state estimation initialized	0.21
Rotational state estimation initialized	4.67
Grasp command issued	4.78
Pose estimator feedback initiated	5.73
Grasp successful	10.91

The command to grasp is issued when the task sequencer sees that the rotational state has been initialized. The "pose estimator feedback" refers to predictions made by the state estimators which are used by the pose estimators to calculate the poses faster.

In the second suite of tests, the pose estimation rate and delay were varied. Figure 3 shows a snapshot from one of these tests. Results from this same set 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.

5.3 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 of accelerometers and gyroscopes are attached to the floor of the aircraft. During microgravity, an object is released, tracked by the vision system, and grasped by the hand.

The objects to be used for grasping onboard the KC-135 aircraft range from simple to highly complex, but are limited to spheres or polyhedral surfaces. Some are lightweight mockups of actual objects used on orbit. Two of the objects are basically dumbbell-shaped objects with polyhedron-shaped masses at the ends. The more complex objects represent a battery from an Extravehicular Mobility Unit (EMU), a star tracker, and an ORU. All of these objects have a complex construction with multiple graspable points.

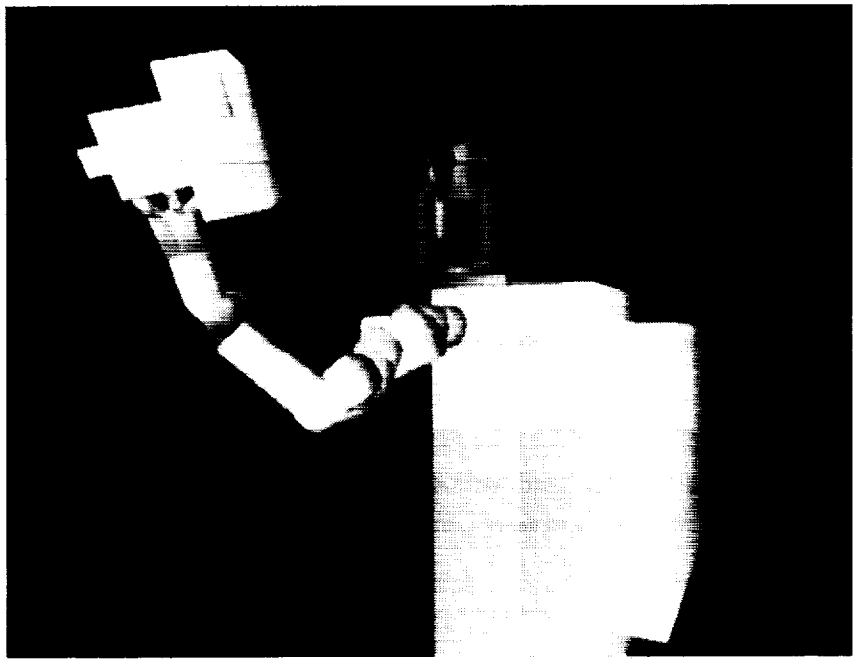

Figure 3. Orbital Simulation of EVAHR Grasping the "Backside" Handhold of Object.

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, was touched by personnel, or left the FOV of both video cameras. The results are presented in Table 3.

Table 3.– Duration of KC-135 Parabolas.

Duration of parabola, sec	Number of parabolas
7-8	2
6-7	5
5-6	6
4-5	2
3-4	2
≤ 3	21

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.

5.3.1 Phase IIIA Results – Hardware Evaluation From a KC-135 Flight

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.

6. 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.

From our Phase II experience with both the interleaved sense-perceive-plan-act software architecture in a stationary environment and the reaction plans architecture in a dynamic, unpredictable simulated environment, we have concluded that (1) the success of the reaction plans approach argues for such a mechanism in an intelligent robot architecture to provide the capability for an appropriate quick reaction whenever perception understands the situation to provide an index into the correct reaction plan; (2) robot control architectures should be heterogeneous (different computational structures for planning and control); and (3) putting the AI planner at a high level of abstraction, which provides plans as goal-seeking guidance rather than direct control, and into an asynchronous mode are steps toward an intelligent robot architecture that can deliver safe behavior as well as goal-achieving behavior in a supervised intelligent robot. Our Phase IIIA experience to date in simulated real-time complex perception and grasping supports the reaction plan view. A way to appropriately integrate the two elements, AI planner and reaction plans, is needed which controls both. The three-tiered architecture may offer such an approach. Both the three-tiered architecture and SOAR are practical implementations of the mathematical theory of intelligent robots[43].

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 it 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[44]. 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.

7. 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.

8. 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 Conference, Huntsville, AL, Sept. 1993.

5. Bureau of Export Administration, "National Security Assessment of the U.S. Robotics Industry." Department of Commerce, Washington, D.C., Apr. 1991.

6. J. F. Engelberger, "Robotics in Service." Cambridge: The MIT Press, Jan. 1991.

7. Office of Technology Assessment, "Exploring the Moon and Mars: Choices for the Nation." Congress of the U.S., Washington, D.C., Aug. 1991.

8. R. P. Bonasso and M. G. Slack, "Ideas on a System Design for End-User Robots." Proceedings of SPIE 1829 Cooperative Intelligent Robotics in Space III, Nov. 1992, p. 352.

9. R. O. Ambrose, M. P. Aalund, and D. Tesar, "Designing Modular Robots for a Spectrum of Space Operations." Proceedings of SPIE 1829 Cooperative Intelligent Robotics in Space III, Nov. 1992, p. 371.

10. F. daCosta et al., "An Integrated Prototyping Environment for Programmable Automation." Proceedings of SPIE 1829 Cooperative Intelligent Robotics in Space III, Nov. 1992, p. 382.

11. W. L. Livingston, "TQM, Total Quality Management." Conference of the Quality Assurance Institute, Washington, DC, May 22, 1991.

12. E. Gat, "Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Real-World Mobile Robots." AAAI-92, Proceedings of the 10th Natl. Conf. on AI, San Jose, CA, July 1992.

13. R. J. Firby, "Adaptive Execution in Dynamic Domains." Ph.D. Thesis, Yale University, 1989.

14. R. P. Bonasso, "Using Parallel Program Specifications for Reactive Control of Underwater Vehicles." Journal of Applied Intelligence, Kluwer Academic Publishers, Norwell, MA, June 1992.

15. 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.

16. J. E. Laird et al. "RoboSOAR: An Integration of External Interaction, Planning, and Learning Using SOAR." Robotics and Autonomous Systems, Vol. 8, 1991.

17. S. B. Huffman and J. E. Laird, "Learning Procedures from Interactive Natural Language Instructions." Proceedings of the 10th Intl. Conf. on Machine Learning, June 1993.

18. E. S. Yager, "Resource-Dependent Behavior Through Adaptation." Ph.D. dissertation in computer science, University of Michigan, Ann Arbor, MI, Sept. 1992.

19. 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.

20. M. Schoppers, personal communication, July 1992.

21. C. Paul et al., "Reducing Problem Solving Variance to Improve Predictability." Communications of the ACM, Vol. 34, No. 8, 1991.

22. E. Durfee, "A Cooperative Approach to Planning for Real-Time Control." Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control, 1990, pp. 277-283.

23. P. Agre and D. Chapman, "Pengi: An Implementation of a Theory of Activity." Proceedings of the AAAI, 1987, pp. 268-272.

24. C. Elsaesser and T. R. MacMillan, "Representation and Algorithms for Multiagent Adversarial Planning." MTR-91W000207, MITRE Corp, McLean, VA, Dec. 1991.

25. C. H. Chien, "Multiview-Based Pose Estimation from Range Images." Proceedings of SPIE 1829, Cooperative Intelligent Robotics in Space III, Boston, MA, November 1992, pp. 421-32.

26. E. Gat, "On the Role of Stored Internal State in the Control of Autonomous Mobile Robots." AI Magazine, Spring 1993, pp. 64-73.

27. M. Magee, C. H. Chien, and T. W. Pendleton, "A Vision System Planner for the Extravehicular Activity Retriever." 1992.

28. O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots." Intl. Journal of Robotics Research, Vol. 5, No. 1, Spring 1986.

29. E. H. Hovy, "A New Level of Language Generation Technology: Capabilities and Possibilities." IEEE Expert, p. 12, Apr. 1992.

30. T. Patten and D. S. Stoops, "Realtime Generation of Natural Language." IEEE Expert, p. 15, Oct. 1991.

31. D. Ballard, "Animate Vision." AI Journal, Vol. 48, No. 1, Elsevier, pp. 57-86, Feb. 1991.

32. J. D. Erickson et al., Technology Test Results from an Intelligent, Free-Flying Robot for Crew and Equipment Retrieval in Space." Proceedings of SPIE 1612, Cooperative Intelligent Robotics in Space II, Boston, MA, Nov. 1991.

33. M. L. Littlefield, "Adaptive Tracking of Objects for a Mobile Robot Using Range Images." Proceedings of SPIE 1829, Cooperative Intelligent Robotics in Space III, Boston, MA, Nov. 1992, pp. 433-443.

34. L. Hewgill, "Motion Estimation of a Freely Rotating Body in Earth Orbit." Proceedings of SPIE 1829, Cooperative Intelligent Robotics in Space III, Boston, MA, Nov. 1992, pp. 444-457.

35. K. A. Grimm et al., "Experiment in Vision-Based Autonomous Grasping Within a Reduced Gravity Environment." Proceedings of SPIE 1829, Cooperative Intelligent Robotics in Space III, Boston, MA, Nov. 1992, pp. 410-420.

36. C. H. Chien, "A Computer Vision System for Extravehicular Activity Helper/Retriever." Proceedings of SPIE Conference on Sensor Fusion and Aerospace Applications, Orlando, FL, Apr. 1993.

37. R. S. Norsworthy, "Performance Measurement of Autonomous Grasping Software in a Simulated Orbital Environment." Proceedings of SPIE 2057, Telemanipulator

Technology and Space Robotics, Boston, MA, Sept. 1993.

38. G. Anderson, "Grasping a Rigid Object on Zero-G." Proceedings of SPIE 2057, Telemanipulator Technology and Space Robotics, Boston, MA, Sept. 1993.

39. R. S. Norsworthy, "Simulation of the EVAHR/KC-135 for Software Integration/Testing." Proceedings of AIAA/NASA Conf. on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, Mar. 1994.

40. M. L. Littlefield, "Real-Time Tracking of Objects in a KC-135 Microgravity Experiment." Proceedings of AIAA/NASA Conf. on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, Mar. 1994.

41. E. L. Huber and K. Baker, "Object Tracking with Stereo Vision." Proceedings of AIAA/NASA Conference on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, Mar. 1994.

42. L. Hewgill, "Motion Estimation of Objects in KC-135 Microgravity." Proceedings of AIAA/NASA Conference on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, Mar. 1994.

43. G. N. Saradis, "Analytic Formulation of the Principle of increasing Precision with Decreasing Intelligence for Intelligent Machines." Automatica, Vol. 25, No. 3, pp. 461-467, 1989.

44. C. H. Chien, "An Architecture for Real-Time Vision Processing." Proceedings of AIAA/NASA Conference on Intelligent Robotics in Field, Factory, Service, and Space, Houston, TX, Mar. 1994.