An Architecture For Intelligent Task Automation

Jeffrey M. Becker and **Fred L. Garrett**
Martin Marietta Denver Aerospace
P.O. Box 179, M.S. 0428
Denver, CO 80201

Abstract

This report discusses the Martin Marietta Intelligent Task Automation Project (ITA). The purpose of the ITA project is to integrate Artificial Intelligence (AI) task planning, path planning, vision, and robotics technologies into a system designed to autonomously perform manufacturing tasks in dynamic or unstructured environments. The application domain chosen for primary demonstrations is dimensional measurement of an F-15 bulkhead. The overall goal is to be able to perform the inspection an order of magnitude faster than the current manual method, which takes about 24 hours for about 1000 inspection points. The project was conducted in two phases. Phase I, completed in December 1984, demonstrated the readiness of the technologies in each of the areas making up the ITA system. Phase II, which was mostly complete in June 1987, demonstrated that the technologies can be integrated into a working system and that the system can be transferred to other applications. The architecture of the ITA system is discussed with an emphasis on the AI components making up the system. The strengths and weaknesses of the architecture and AI techniques applied are discussed.

I. Introduction

Artificial Intelligence and Robotics technologies have advanced to the state where combining them into an intelligent system for performing industrial tasks is feasible. The purpose of this paper is to give a broad overview of the Martin Marietta Intelligent Task Automation (ITA) project so the reader can gain an understanding of its overall architecure and the AI technologies applied.

Phase I, which started in January 1983, demonstrated the readiness of component technologies of the ITA system. Sequence planning (the "traveling salesman" problem), task planning, and path planning systems were developed and demonstrated. Vision capabilities demonstrated included edge extraction and classification, planar region extraction, object recognition [Magee and Nathan, 1985], and dimensional measurement, all from laser scanner range data. Plan execution was demonstrated by performing a tool pickup and several measurement actions using a Cincinnati Milacron T3-746 arm and the 6 degree of freedom control system developed during the program. An approach to the problems of execution monitoring and exception handling was also developed and implemented [VanBaalen, 1984].

Phase II, which started in December 1985, demonstrated that the technologies developed in Phase I could be integrated into a working system. Most of the code developed for Phase I was rewritten under Phase II to incorporate lessons learned. *Figure 1* illustrates the hardware configuration for the bulkhead inspection demonstration task.

II. System Overview

The ITA Phase II system architecture is a heterogenous hierarchical planning and plan execution system consisting of a sequence level, a task level, a geometric level, and a physical level. The software consists of the thirteen components (boxes) shown in *Figure 2*. These components access the seven knowledge bases (cylinders) shown. The general sequence of operations is as follows.

Measurements to be performed are entered using the Offline Measurement Entry component. The measurement specifications are preprocessed to generate the measurement knowledge base, the sequence plan, and the Operation Planner MACROPS (generalized plans) using the ITA system in Offline Simulation mode. Though not strictly necessary, preprocessing improves the speed of online operations in a production environment. Offline simulation also provides a safe means for verifying correct system operation. When started up online via the System Monitor, Top Level executes the sequence-level plan by getting the next measurement to be performed from the Sequence Planner, getting a task plan to perform the measurement from the Operation Planner, passing the operation plan to the Plan Executive/Monitor for execution, and passing the result of plan execution back to the System Monitor for archiving. The Plan Executive/Monitor uses the Geometric Reasoner to translate qualitative parameters of the plan to quantitative values. It sends commands for robot actions and ultrasonic measurements to the Path Planner. The Path Planner

---

This work was performed at the Intelligent Task Automation Project facilities of Martin Marietta Denver Aerospace. This work was supported by the Air Force Wright Aeronautical Laboratories and the Defense Advanced Research Projects Agency under Contract F33615-82-C-5139.

uses the Collision Avoidance component to determine if a proposed path intersects with any object in the workspace, and sends commands to the Robot Controls component to execute a path plan. Commands for Scanning Laser Ranging Assembly (SLRA) measurements are sent by the Executive to the Vision component. If either the Path Planner or Vision component returns an error message for a command result, the Executive invokes the Exception Handler to diagnose the problem and generate a recovery plan. A description of each component follows.

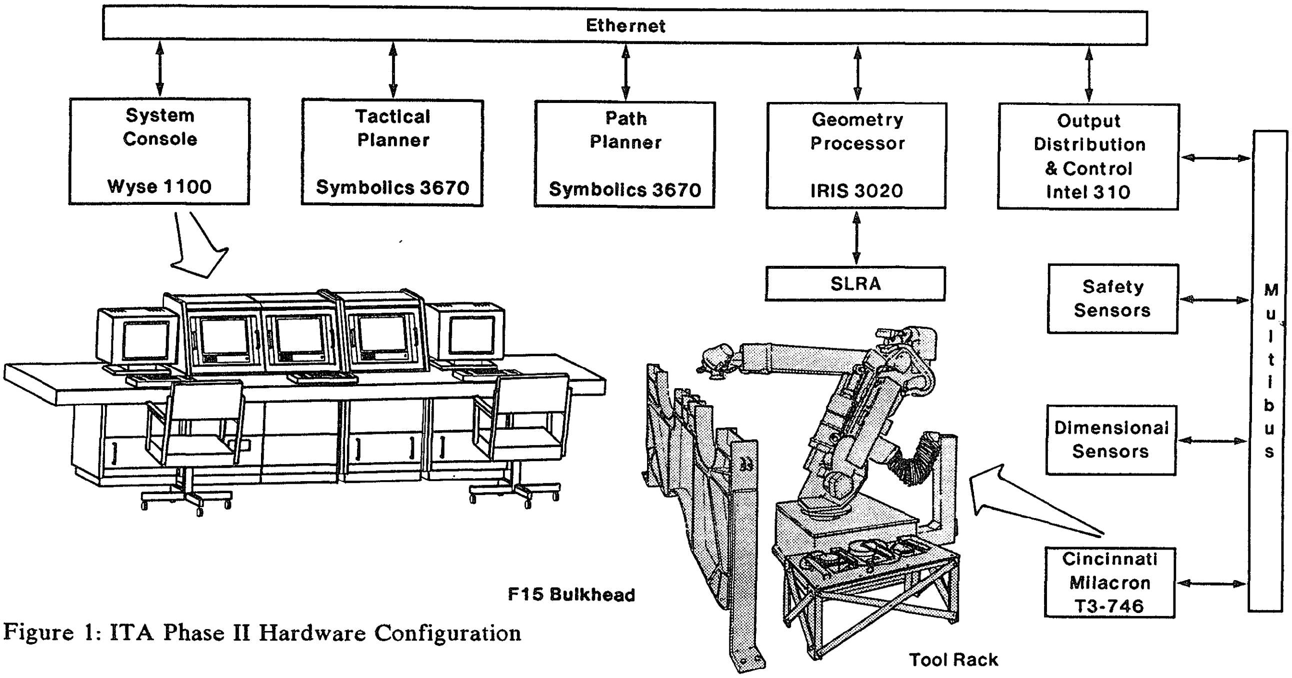

Figure 1: ITA Phase II Hardware Configuration

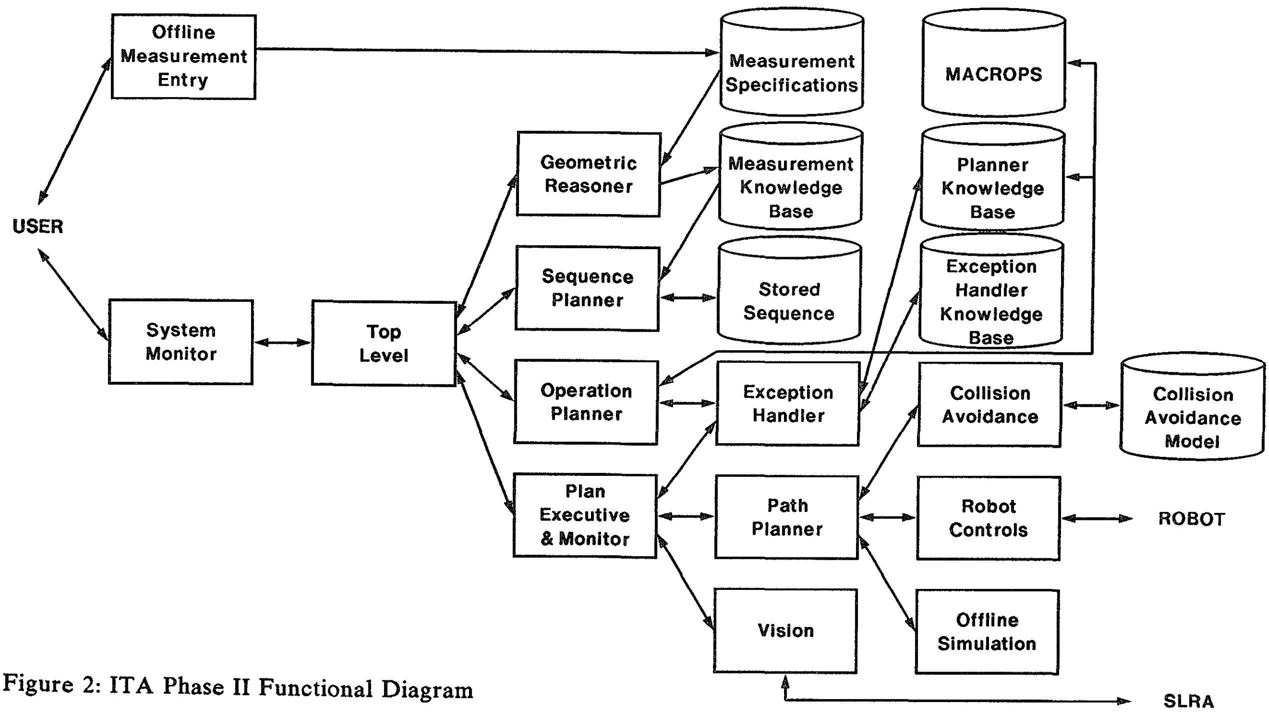

Figure 2: ITA Phase II Functional Diagram

The Offline components are used to enter and edit specifications of the measurements to be performed to inspect a part, to generate problem sets for testing the path planner, and to provide a 3-D graphics simulation capability for display of path planner output. The simulation may be driven instead of the actual robot for overall system verification. Measurement specifications are entered using a graphics display of the bulkhead to select measurement locations, and using a menu-oriented interface for entering additional parameters such as dimension and tolerance.

The System Monitor serves as the user interface to the ITA online system, and may also be used to monitor preprocessing activities. Commands are included for starting, stopping, interrupting, and continuing inspection activities, and for displaying the results of the inspection. Graphics interfaces are provided for the Sequence Planner, the Operation Planner, the Plan Executive/Monitor, the Exception Handler, and the Path Planner to allow for detailed examination of system activities.

The Top Level component executes sequence-level plans as described above. Top Level also watches for STOP and INTERRUPT commands from the System Monitor and will either halt all system activity immediately or interrupt activity after the current plan completes execution accordingly.

The Sequence Planner generates a sequence plan from an unordered set of measurement specifications. It also looks up the next measurement goal in the stored sequence plan on request from Top Level. Generating a sequence plan consists of partitioning remaining measurements into groups according to the measuring tool to be used, and then ordering the points within each group. Partitioning of measurements is performed using a set of rules coded in MRS (Meta-Level Representation System) [Genesereth, et al., 1984] for tool selection. Measurements that are already done or that cannot be done because of the unavailability of the correct tool are placed in separate groups. Ordering of the measurement points is accomplished using a near-optimal solution to the traveling salesman problem known as the "Convex Hull" algorithm followed by 2-Optimal Edge Exchange and Peephole optimizations [Golden, et al., 1980]. Looking up the next step consists of popping the next measurement specification off the stored sequence plan, and verifying the required resources are available. If not, resequencing is performed and the next measurement (if any) from the new sequence is returned.

The Operation Planner consists of a Task Planner, a Plan Generalization component that creates a macrop-operator (MACROP) from a plan, and a MACROP Lookup component that finds and instantiates a MACROP for a given initial state and goal conditions. The Operation Planner first tries MACROP Lookup. If no applicable MACROP can be found, the Task Planner is called on to generate the plan from scratch. The plan is then generalized and stored as a MACROP for future reference.

The Task Planner is a hierarchical, nonlinear, backward chaining planner that uses hill climbing search (backtracking is chronological). For a treatment of related planners see ABSTRIPS [Sacerdoti, 1974], Nonlin [Tate, 1977], Noah [Sacerdoti, 1977], and SIPE [Wilkins, 1984]. The Task Planner is hierarchical in the sense of ABSTRIPS - goals are weighted and only the highest level unsatisfied goals are worked on. Nonlinear plans are achieved by (1) allowing operators to be ordered in parallel with other operators in the plan if there are no interactions and (2) allowing serendipitious goal reduction. Deductive operators are used to replace explicit delete lists in the operator descriptions. Unary and n-ary constraints on operator variables are provided to generate and to test candidate bindings for operator variables, respectively. *Figure 3* shows an operator declaration and a deductive operator for the ITA domain.

```
;;; Operator for Ultrasonic Measurements:

(static-operator
  :name-and-format (us-measure $tool $arm $id)
  :preconditions
    ((couplant-applied (goal-point (meas $id)))
     (at $arm (in-contact-point (meas $id)))
     (holding $arm $tool))
  :adders
    ((measured $tool $arm (meas $id)))
  :unary-constraints
    ((type $arm arm)
     (type $tool us-tool))
  :n-ary-constraints
    ((can-lift $arm $w1)
     (weight $tool $w2)
     (<= $w2 $w1))
  :resources ($tool $arm)
  :message-pattern
    (MEASURE (min-value (meas $id))
             (max-value (meas $id)))
  :command-stream (command-stream path-planner $arm)
  :reply-pattern  (VALUE $val)
  :result-pattern (dimension $id $tool $val 0.0))


;;; Propositions denied when a tool is picked up:

(deductive-operator
  :name-and-format (holding $arm $tool)
  :denied ((location $tool $arm in-rack)
           (holding $arm (n= $tool))))
```

Figure 3: Example of Static and Deductive Operators

Plan Generalization involves replacing certain constants in a plan by variables, finding overall preconditions and adders of the plan, collecting unary and n-ary constraints, and creating additional resource constraints. *Figure 4* shows an example of a generalized plan. MACROP Lookup is a straightforward process of comparing each MACROP to the given initial state and goal conditions, and then determining if the constraints are satisfied. The MACROP is then plugged with the bindings found. MACROP Lookup is roughly two orders of magnitude faster than generating the same plan from scratch (~ 0.1 sec versus ~10.0 sec for a typical ITA domain plan). This capability is essential for meeting the production environment timing constraints of the ITA project.

```
;;; MACROP for Ultrasonic Measurement sequence:

(macrop
  :name-and-format (MEASURED $ID1 $TOOL1 $ARM1 $ID2)
  :purpose
    ((MEASURED $TOOL1 $ARM1 (MEAS $ID2)))
  :preconditions
    ((AT $ARM1 (IN-CONTACT-POINT (MEAS $ID1)))
     (HOLDING $ARM1 $TOOL1))
  :adders
    ((AT $ARM1 (IN-CONTACT-POINT (MEAS $ID2)))
     (MEASURED $TOOL1 $ARM1 (MEAS $ID2)))
  :unary-constraints
    ((TYPE $TOOL1 US-TOOL)
     (TYPE $ARM1 ARM))
  :n-ary-constraints
    ((CAN-LIFT $ARM1 $W1)
     (WEIGHT $TOOL1 $W2)
     (<= $W2 $W1))
  :resources ($TOOL1 $ARM1)
  :plan
    ((1 (MOVE-RETRACT $ARM1 $ID1) NIL)
     (2 (MOVE $ARM1 (MEAS $ID1) (MEAS $ID2)) (1))
     (3 (APPLY-COUPLANT $ARM1 $ID2) (2))
     (4 (MOVE-CONTACT $ARM1 $ID2) (3))
     (5 (US-MEASURE $TOOL1 $ARM1 $ID2) (4))))
```

Figure 4: Example of a MACROP

The Geometric Reasoner is responsible for creating, accessing, and maintaining the Measurement Knowledge Base (MKB). The MKB contains information about where the arm can be positioned to perform each measurement, the approach position in free space for ultrasonic measurements, and parameters for performing SLRA measurements such as patch sizes and locations in the field of view. This information is derived from geometric constraint and preferrence information.

The Plan Executive/Monitor executes a plan by sending commands to the Path Planner, which controls robot motion and ultrasonic measurements, and to Vision, which controls SLRA measurements. The Executive splits a plan into separate command streams, one for each independently controllable sensor or effector. The Path Planning component uses a lookahead queue to do smoothing where continuous motion is possible over several commands, so it receives all of its commands from a plan at once. To synchronize a commanded process that uses a lookahead queue with other processes, the Executive inserts WAIT commands before any command that has a predecessor belonging to another command stream. The Executive sends a CONTINUE command for a WAIT command when the appropriate predecessor commands have been completed. The reply to a command can be either a normal reply or an exception reply. A command may also "time out" if a reply is not sent within a reasonable period of time. When an exception reply or timeout occurs, execution of the plan is stopped, and all relevant information about the exception is passed to the Exception Handler.

The Exception Handler is responsible for diagnosing the cause of the exception, updating the world model to correspond to the current state of the world, and generating a recovery plan. For diagnosis, the Exception Handler is given a knowledge base (MRS rules) containing information about possible causes for each fault, the number of times each exception has occured, the assertions that each available test can verify, preconditions of each test, and an estimated cost for each test. When an exception message is received, the certainty of assertions associated with possible causes is reduced. Tests are selected, executed, and the results interpreted until a single cause is isolated. The next test to execute is selected by dynamically generating a near-minimal decision tree according to fault frequency, test cost, and test precondition information. Replanning is done by the Operation Planner using the current state for the initial state and the original goals of the failed plan for the goal conditions.

The Path Planner functions as the interface between the task plan Executive/Monitor and the real-time robot controller. The Path Planner first verifies that the goal position is reachable. It then generates collision-free paths for the robot using a dual-level algorithm. First, a potential collision-free path for the end effector (modeled as a point) is found using the "visibility lines" method [Lozano-Perez, 1979] with goal optimization for producing graph nodes, and A* search for selecting the node sequence. The prospective path is then checked at incremental positions to see if any collisions involving intermediate links of the arm will occur. If a collision could occur, new intermediate subgoals are proposed and evaluated until a collision free path is found. A third trajectory-planning phase, involving profile smoothing and velocity selection, is handled in the Robot Controller.

The Collision Avoidance Model is the geometric representation of the workcell (objects, tools, robot parts) used by the Path Planner. The Collision Avoidance Model provides for determining if a point or line segment intersects any workcell object, if a robot in a particular position intersects its own links or a workcell object, and for updating the model to reflect changes in the real world. The basic representation structure is a region tree. A region tree (actually a directed graph) is a hierarchical structuring of part of space into arbitrarily oriented regions. A region can be a sphere, tube (cylinder with spheres of the same radius at both ends), or a rectangular parallelepiped. At the leaves are solid regions representing actual workcell objects. Regions need not completely contain their children, but all regions except for roots must be completely contained in some set of ancestors. Region tree nodes contain shape, size, position, orientation, and solidity information.

The Vision component is responsible for processing SLRA images to obtain dimensions for the observed parts of the bulkhead. The SLRA was developed by the Environmental Research Institute of Michigan (ERIM) under subcontract to Martin Marietta Corporation during Phase I of the ITA contract. It uses a modulated laser light source to determine the range to the target. The range is computed by determining the phase change that results when the light travels from the sensor to the target and back. The resulting 3-D range information can be used for dimensional measurement and object classification. Each measurement involves positioning rectangular patches in the image to correspond to critical areas of the part being measured. Measurements are obtained by a variety of techniques, depending on the type of measurement to be performed. These techniques include edge detection, computing surface normals, and curve fitting.

III. Results and Analysis

Phase II demonstrations have shown the ITA system works as an integrated whole. Several runs of the measurement process were performed, both simulated and with the actual robot arm and measuring tools. The system was shown to be able to handle bad measurement and broken measurement tool exceptions properly. In a separate research task funded under the ITA program, coordinated dual-arm control algorithms were demonstrated. (Further details, not available at the time of writing, will be given at the conference.)

Although major strides were made in building an integrated intelligent robot system, the system is still not as flexible nor as powerful as we would like for truly general-purpose manipulator automation. For example, to make the system more flexible, the Top Level component, which is currently hard coded for the inspection domain, should be replaced by a high-level planner that can call on special- purpose functions such as the current sequence planner as tools.

Because of the heterogenous hierarchical architecture used, the task planner only has to plan for a single measurement at a time. This makes the task planner's job much easier. In fact, we have found the branching factor of the ITA measurement domain to be less than that of the standard blocks world domain for task planning. Even so, ITA task plans share many subsequences. We would like to add the capabilities of selectively generalizing interesting subsequences of a plan as in Morris [Minton, 1985], and of using MACROPs in addition to primitive operators for constructing a task plan. We are also looking into incremental task plan revision techniques [Simmons, 1985] as an alternate means of replanning following an exception under a research task associated with the ITA project. Overall, richer representations of domain objects and robot actions are needed to allow more powerful, knowledge-based task planning for more difficult domains.

We have found that truly robust exception handling in robotics domains requires powerful sensory capabilities, especially vision. Reasoning can do little to replace perception when it comes to determining the state of an environment subject to external influences. Our choice of a break-and-resume approach to exception handling was based on the (correct) assumption that high-level sensing operations could not generally be done in real time. Given a fast vision system for real-time hand-eye control, many problems that are now treated as exceptions (e.g., bumping into something because of positioning inaccuracy) could be easily avoided. We hope that a second arm and a more general vision component can be added back to the system in follow-on work. Object recognition research conducted during Phase I could be applied to such an effort.

IV. Conclusions

We have shown that current artificial intelligence technology can be applied to provide a powerful system for controlling an industrial robot in a real-world domain. Being able to integrate such a system is very much a team effort and requires organizational commitment as well as technological expertise. Martin Marietta is currently assessing the possibility of making the Intelligent Task Automation system available as a test bed for outside research in the areas of planning, compliant and multi-arm controls, and integrating vision with robotics.

Acknowledgements

The Martin Marietta Intelligent Task Automation Project is a team effort, and this paper is based upon the efforts of and has received input from many people. Our thanks to the entire ITA team. Special thanks to Dennis Haley (project manager), Don Mathis, and Mark Thomas.

References

Brooks, R. A., Lozano-Perez, T., 1983. "A Subdivision Algorithm in Configuration Space for Findpath with Rotation", *Proceedings IJCAI-83*, Karlsruhe, West Germany, 1983, pp 799-806.

Genesereth, M., R. Greiner, M. Grinberg, and D. Smith, 1984. *The MRS Dictionary*, Heuristic Programming Project Report No. HPP-80-24, Stanford University, Stanford, CA, January 1984.

Golden, B., L. Bodin, T. Doyle, and W. Stewart Jr., 1980. "Approximate Traveling Salesman Algorithms", *Operations Research*, Vol 28, No. 3, Part II, May-June 1980, pp 694-711.

Lozano-Perez, T., 1979. "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles", *Communications of the ACM*, Vol 22, No. 10, October 1979, pp 560-570.

Magee, M., and M. Nathan, 1985. "A Rule Based System for Pattern Recognition that Exploits Topological Constraints", *Proceedings of IEEE CVPR85*, June 1985.

Minton, S., 1985. "Selectively Generalizing Plans for Problem Solving", *Proceedings IJCAI-85*, Los Angeles, CA, 1985, pp 596-599.

Sacerdoti, E., 1977. *A Structure for Plans and Behavior*, North-Holland, New York, 1977.

Simmons, R., 1985. *Knowledge Intensive Plan Debugging*, Internal Report, MIT AI Laboratory, January 3, 1985.

Tate, A., 1977. "Generating Project Networks", *Proceedings IJCAI-77*, Cambridge, MA, 1977, pp 888-893.

Wilkins, D., 1984. "Domain-Independent Planning: Representation and Plan Generation", *Artificial Intelligence 22*, 1984, pp 269-301.

Van Baalen, J., 1984. "Exception Handling in a Robot Planning System", *IEEE Workshop on Principles of Knowledge-Based Systems*, Denver, CO, December 1984. Not Published - late submission.