KNOWLEDGE BASED ERROR RECOVERY IN INDUSTRIAL ROBOTS

M.H. Lee, D.P. Barnes & N.W. Hardy
Robotics Research Group
Department of Computer Science
The University College of Wales
Aberystwyth, U.K.

ABSTRACT

This paper describes the main aims of a new research project concerned with the implementation of automatic error recovery facilities in industrial robotics. An approach is discussed in which an existing manufacturing work cell is to be enhanced by the addition of a task event model contained in an error recovery knowledge base. This paper outlines the main design issues involved in this work. Reasons why conventional fault tolerance techniques are inadequate are given and the industrial application is explained. A particular approach to sensory monitoring and error diagnosis is described. The proposed system has more similarities with sensory driven expert systems than with body modelling contingency planners.

I INTRODUCTION

A. Background

This research concerns the problem of providing error recovery capabilities in industrial robot systems. The long term goal is to define software control mechanisms that will enable a robot to detect error conditions in its working environment and perform corrective actions in order to continue working unattended. The work is directed at industrial applications such as machine loading and simple assembly tasks. Our interests do not cover hardware errors in the robot or computer systems nor software errors in the control programs. Such topics are well covered in the literature on fault tolerance systems[l]. We assume that the hardware and software are relatively reliable (although we do provide some diagnostic facilities) and only deal with physical errors in the robots task environment. For example, defective components, alignment drift and jammed feeders are some of the common faults that can lead to damaged equipment, faulty products or significant plant down-time.

B. Conventional techniques

The most widely used technique for hardware recovery is that known as backward error recovery where the system is restored to an earlier error-free state by an inbuilt mechanism. Such checkpoint and rollback schemes operate quite independently of the nature of the error and so can handle a range of error types. However, despite the attractive simplicity of these methods their fixed response often proves inappropriate in robotic applications. This is because backward recovery assumes (a) that processes are reversible and objects are recoverable, and (b) that the system has full control over a well defined (i.e. hardware or software) environment. These assumptions do not hold in robotics.

C. The Assembly World

In computer systems, errors can be defined as either component error or design errors, but in the robot assembly world there exists a third type: external errors. These include external interference with processes, or components, and unexpected events such as breakages and jammed parts. Thus our systems must operate with incomplete control over the environment as well as incomplete information. When we list just a few error cases we see ample evidence of this:

incorrect or defective parts (component errors)
faulty feed process (missing part or orientation error)
faulty gripper action (pose error or dropped part)
incorrect target placement (collision or position error)

There has been very little previous work that has dealt with these issues [2,3]. We feel that progress lies in the application of knowledge engineering methods. Only when the nature of an error is understood can it be successfully treated.

D. Knowledge Based Recovery

We propose a forward recovery approach where operators are applied to transform the error state into one of the error-free states further ahead on the task cycle. This amounts to a kind of corrective feedback. In this situation the recovery agent will need all available information including:

(a) the actual state of the physical system, (b) the desired or expected state, (c) current sensory data, (d) expected sensory data and (e) details of plausible recovery actions.

We are building a software system that maintains a knowledge base containing information on the robots' task, the working environment, expected sensory Bignals and plausible contingency procedures.

II THE LABORATORY TEST-BED

In order to remain faithfull to industrial practice a laboratory rig has been built which models an actual commercial application. This consists of a work cell with a Unimation Puma 600 and a visual inspection station. Component recognition algorithms are processed by the Autoview vision package [4] running on an LST 11/23. A conveyor delivers pipe sections to the work cell which fall onto a light table and are analysed, via a TV camera, by the Autoview software. The puma then performs an appropriate action sequence for each pipe type, which normally involves transport to two machining station and ejection into an output bin. This is the normal, i.e. error free, task performance.

For error sensing, additional sensors have been added to the system. These include proximity and force feedback and special tactile sensing pads inside the gripper jaws. This distribution of multi-model sensors permits a range of errors to be deteeted:

Sensor	Errors detected
Proximity	Collisions, location errors (Arm movement monitoring)
Arm forces	Obstructions, jammed parts (Insertion/assembly monitoring)
Tactile gripper	Slip, orientation errors, lost parts (Component movement monitoring)
Vision	Part location, orientation, defects (Inspection & verification)

In the original application the Puma is controlled by a task program, written in the manipulator control language VAL, which interacts with the visual inspection algorithms. However, in the laboratory the LSI 11/23 acts as overall supervisor and is able to down-load VAL programs or even single instructions for the Puma to execute. In normal task operation the original VAL program is executed and so there is no effective difference between the two systems. However, when in recovery mode the supervisory software is able to generate and monitor additional VAL commands as determined by the prevailing sensory conditions and knowledge of the error event.

ITT ERROR DETECTION

Obviously, for good quality error detection we will need many sensors. However, errors are rare occurrences and the benefits gained from extensive sensing must be balanced against the icrease in costs and computational overheads. We feel that an effective solution is to design systems with two levels of sensory activity. Our system has a low level of sensory monitoring during normal operation but can perform much more extensive processing when required to respond to error situations. We define a sensory signature as a collection of parameters which specify the limits of acceptability of a sensed signal during a task phase. These can be one dimensional, e.g. insertion force limits, or multi-dimensional, e.g. visual silhouette measurement tolerances. In either case the signatures are continually tested at set points on the task cycle by a background monitoring process. Normally the signatures will be accepted and the task Continues. When a signature goes out of range, i.e. a "signature interrupt" occurs, the robot is halted and control handed over to the error diagnosis and recovery modules. Notice that this recovery software can involve quite disproportionate sensory and computational overheads as the system is effectively in a "down-time" status.

Although the sensory signatures are relatively simple measures, they will vary considerably (in range and relevance) with task parameters such as component type, movement speed, work space complexity, etc. We use the notion of relevancy by incorporating explicit control information in the task program so that only pertinent signatures are used at each stage of the task cycle. Thus, the monitoring regime varies dramatically, in scope and quality, over the whole task. The design of signatures and the relevancy controls is very application dependent. For each task the VAL program is created first and then signatures can be selected, parameterised and entered into a signature data base.

IV FAULT DIAGNOSIS

The second stage of recovery involves an assessment of the nature of the error and the extent of its influence. By systematic analysis of robot action sequences we can build a generic fault tree for use in diagnosis. There are essentially three basic actions which our robot can perform and these can be broken down in terms of their constituent components as follows:

1. Pick up component
   - locate (find a component)
   - approach (fine movement towards target location)
   - grasp (acquire component in the gripper)
   - depart (fine movement away from target location)
2. Transfer component
   - (coarse, fast movement between locations)
3. Place component
   - approach (fine movement towards target location)
   - insert (fine movement while in physical contact)
   - ungrasp (release component)
   - depart (fine movement away from target location)

The "locate action" is not a robot function but refers to the action of either part feeders (implicit location data) or sensors (explicit location data).

Now by examining each of these atomic actions in turn we obtain a tree of error states and their relationships. For example:-

Pick up

Location error	feeder empty or jammed
	part defective
	part orientation error
Approach error	collision with part
	missed part
	collision with workspace
Grasp error	no part
	several parts
	orientation error
	part location error
	gripper slip
Depart error	obstruction
	part error
	grasp error
	workspace error

This is only a sample of the full tree: there are further levels and more detailed expansion. By using the concept of an "atomic action" we have produced a detailed analysis for each manipulator movement axis [5]. These trees illustrate the inherent structure in the error states and their causes. If we have knowledge of the current action and sensory data then, from the tree, we can infer plausible error causes. Several iterations may be required, involving requests for selective sensory data.

V KNOWLEDGE BASE DESIGN

The ideal knowledge base for a robot would contain geometric data in a body modelling system which could represent the exact physical structure of the workspace and the objects. By using computational geometry and extensive high powered sensing the system would deal with contingencies by generating its own recovery plans [6]. However, while research is active on these topics, such object level systems are not yet well developed and will be very complex and expensive.

Our aim is more modest: to develop a knowledge base of actions and events in terms of the task descriptions used in existing manipulator languages. This is more a task event model than a body modelling system. The original task specification consists only of a sequence of VAL instructions, our final specification will have several parallel levels of ancillary knowledge cross-referenced into the VAL program. This multi-layered approach is a key feature of successful knowledge bases [7]. Our current levels are as follows:

Task level - VAL program (manipulation cycle of actions and tests).

World level - Knowledge of objects in environment.

Action level - preconditions, actions, effects and results.

Diagnostic level - sensory interrogation procedures.

Recovery level - parameterised recovery schemas.

Other levels being examined include a user level, performance level and a meta strategy level.

Our current activities are directed at-- knowledge representation (frames and/or production rules), properties of objects and events (e.g. reversability), and the separation of generic diagnosis and fault data from application specific data. The next stage will be to implement and test our design in the application framework.

ACKNOWLEDGEMENTS

We are grateful for the support of British Robotic Systems Ltd. and the S.E.R.C. through Grant GR/B/94106.

REFERENCES

fl] Anderson, T. & Lee, P.A., "Fault Tolerance: principles and practice." Prentice-Hall, 1981.

12] Park, W.T., "Robotics Research Trends." Tech. Memo. 160, S.R.I. Int 1978.

[3] Gini G. et al "Program Abstractions in Intelligent Robots" In Proc. 10th I.S.I.R. Milan, 1980.

[4] Mott, D.H., "Image Processing Algorithms." Sensor Review, April 1981.

[5] Lee, M.H., "Error Diagnostics," Tech. Memo., Department of Computer Science, U.C.W. 1982.

[6J Brooks, R.A., "Symbolic Error Analysis & Robot Planning." Int. J. Robotics Research 1:4 (1983).

[7] Mylopoulons, J., "An overview of knowledge representations." Sigart Newsletter, No.71 1981.