A Prototype of Fault Diagnostic System for Robots

Yoshio Endo　Tomohiro Miyazaki
Michitaka Oshima　Shiro Hagihara

Industrial Electronics and Systems Development Laboratory,
Mitsubishi Electric Corporation, Amagasaki/JAPAN

1. INTRODUCTION

Maintenance robots which operate at nuclear power plants require extremely high reliability, because they should not harm or damage any of the equipment or systems in the plants. This rule should be kept even when the maintenance robots themselves would fail.

Such a robot will be a complex system composed of moving mechanisms, a vision sensor subsystem, a manipulation subsystem, and several kinds of sensors for inspection, etc. Hence the controller of the robot will contain so much amount of hardware components. Needless to say, every component in the robot system should have sufficient reliability. The authors remark that the reliability of the motion of the manipulator is particularly important in the system, because it treats the critical and important devices directly by its end effecter.

It is predicted that causes to bring about a fault to a robot are spread over the entire system. Above all, the controller of servo actuators can be regarded as the most critical part in the system; once a fault occurs in an actuator controller, it is most likely to cause a fatal fault in a moment to the entire system. Generally, in controlling actuators in robots, rather fast processings are needed. In industrial robots, for example, the sampling time of control is often in the order of millisecond or sometimes less than that. To respond to such fast processes, the diagnostic methods in many of the industrial robots are based on hardware limiters, software limiters or mechanical stoppers. Since those limiters and stoppers are to trap the data over a certain limit and are able to operate only in the limited area, the method cannot cover all the range of the robot motion.

Fault tree analysis has been originally developed for reliability and risk analysis and is now one of the methods to analyze faults in systems or devices systematically. It has been applied to various areas, for example, fault diagnosis of avionic subsystems on aircrafts[1], hazard analysis in chemical processes[2], safety analysis of industrial robots[3], fault analysis of control systems with control loops[4], etc. Some techniques are developed for fault tree analysis with the aid of computers[5,6,7]. Recently, artificial intelligence technology is applied to fault diagnosis using fault tree[8,9]. The fault tree analysis is intrinsically based on static analyses of the system. It reveals the strong ability in fault diagnosis, but it is difficult to apply the technique to the real-time fault detection of the dynamic systems with such fast processes as described above.

To satisfy the demand for both the real-time fault detection and the extension of the range of diagnosis, the authors divide the diagnostic process into three stages — the fault detection, the fault analysis and the fault part identification. In the method, faults in robots are detected in real-time using an mathematical model of the control system. When a fault is detected, the phenomena caused by the fault are analyzed with another model of faults in the control system, and the faulty part is identified using knowledge base and inferring mechanisms. This paper also describes the configuration of a prototype system developed to verify the effectiveness of the method. It shows experimental results on a multi-axis robot manipulator.

In the next chapter, a conceptual configuration of the hardware system of a robot controller is shown and the fault types possible in the system are classified. In chapter 3, the outline of the approach addressed in this paper is summarized. In chapter 4, the method of modeling the control system to detect faults in real-time is described. In chapter 5, the method of modeling the faults in the control system is described. In chapter 6, the method to identify the faulty part with the artificial intelligence technology is presented. In chapter 7, the configuration of a prototype of the fault diagnostic system is shown. In chapter 8, the experiments and discussions to verify the diagnostic methods on a multi-axis robot manipulator are described.

2. CONFIGURATION OF A ROBOT CONTROLLER AND ITS FAULTS

Figure 2-1 shows a conceptual configuration of the hardware system for an axis of a robot controller. This is a digital control system, in which the angular velocity (ω) of the motor axis, and the armature current (I) of the motor are fed back into the servo CPU.

The applied voltage (V_0) to the motor is calculated and is also converted into pulse-width-modulated (PWM) signals by the servo CPU. The on/off timings of four power transistors ($Tr_1 \sim Tr_4$) are controlled by these signals. The armature current of the motor is fed back into the servo CPU through an A/D converter. Channels A and B from the encoder circuit are converted into two quadrature counts — *up* counts and *down* counts, which increment the *up* counter and the *down* counter, respectively. Both counts are fed back into the servo CPU. Thus the control system is mainly composed of digital circuits except the detection part of the armature current of the motor.

There can be various types of faults in the control system described above. Generally, faults in a system can be classified into two categories — those caused by internal sources and those caused by external sources. It is rather difficult to construct a model including the external sources and environments, because they are so various and are different from site to site. In this paper the faults caused by the internal sources are treated. In digital circuits, disconnections of signal lines, fractures of semi-conductor devices are in consideration. These faults are represented as sticking at "high" or "low" of the logical signal level. In analog circuits, faults are classified into two types — one is type of *gain change* and the other is type of *large offset* in the output.

The fault types, which are possible in the control system shown in Figure 2-1, are classified in Table 2-1. The "fault modeling functions" in the table will be explained in chapter 5.

3. OUTLINE OF FAULT DIAGNOSIS

The diagnostic process is divided into 3 stages — (1) fault detection, (2) fault analysis and (3) fault part identification. The fault detection is executed in real-time. After detecting a fault, the diagnostic system issues an emergency stop command to the robot. It then analyzes the phenomena caused by the fault and identifies the faulty part. Stage (1) is based on a model of the dynamic characteristics of the control system, stage (2) is based on another model of faults in the system and stage (3) is based on the inference using the knowledge base of cause-effect relations about the faults in the robot.

The approach addressed in this paper assumes the following:

(a) the servo CPU is fault free, and

(b) multiple faults do not occur within one servo control cycle simultaneously.

The merits of the method are as follows:

(1) no additional sensors are needed, and

(2) no special hardware tests other than normal operations are needed.

The merit (2) minimizes the risk to damage the equipment in the plants. All the operation added to the robot system by the diagnostic method is to get the reference data and sensory feedback data of the control system for every control cycle.

4. METHOD OF FAULT DETECTION

The control system of a robot — including its mechanisms — is regarded as a dynamic system. Some methods have been presented for the fault detection in dynamic systems. The likelihood ratios in the statistics have been used to the problem[10,11]. Another approach is based on testing statistical hypotheses[12]. These methods need so much calculation that are difficult to be applied to the real-time fault detection.

The method of fault detection adopted in this paper is as follows. We constructed a model of the dynamic motion of each axis in mathematical formulae. The diagnostic system monitors the input and output of the robot controller in each servo cycle. It predicts the motion of the axis by the model in real-time. Faults are detected by the difference between the real motion and the predicted motion by the model.

The detail of the model is shown in Figure 4-1. In general, the dynamic characteristics of a robot have strong non-linearity and are very complicated because of the dynamical mutual interferences among axes[13]. It is difficult to calculate the dynamic motion precisely in real-time, because the amount of the calculation is too large. Thus we approximate the velocity control loop as a linear system. This approximation is valid if the effects of the mechanical resonance and the electrical dynamics in the high frequency range are eliminated by a low pass filter. Hence the angular velocity reference Ω_{ref} and the predicted angular velocity Ω_{prd} in the Laplace transformation are related by

$$\Omega_{prd} = \frac{k\ \Omega_{ref}}{s + p} \qquad (4\text{-}1)$$

where s is the operator of Laplace transformation, and k and p are constants. The approximation of equation (4-1) in the discrete time system yields

$$\omega_{prd}^{i} = \frac{k\ T_s}{1 + p\ T_s}\ \omega_{ref}^{i} + \frac{1}{1 + p\ T_s}\ \omega_{prd}^{i-1} \qquad (4\text{-}2)$$

where ω_{prd}^{i} and ω_{ref}^{i} denote the predicted angular velocity ω_{prd} and the angular velocity reference ω_{ref} at time i in the discrete time system, respectively. T_s denotes the sampling time of the discrete time system.

Let the detected value of the angular velocity be ω^{i}, the output of ω_{prd}^{i} and that of ω^{i} through the low pass filter be $\omega_{prd,F}^{i}$ and ω_{F}^{i}, respectively. The rule to judge whether a fault occurs or not is as follows:

$$\left.\begin{array}{l} |\omega_{prd,F}^{i} - \omega_{F}^{i}\,| \leq \varepsilon \ \ ==> \ \text{normal} \\ |\omega_{prd,F}^{i} - \omega_{F}^{i}\,| > \varepsilon \ \ ==> \ \text{failure} \end{array}\right\} \qquad (4\text{-}3)$$

where ε is a positive finite value.

5. METHOD TO ANALYZE FAULT PHENOMENA

To analyze faults in the robot, the authors developed a method to know what kind of phenomena has occurred in the robot. When a fault occurs in the robot, the transfer characteristic of the faulty component changes from that in the normal state. The authors model such a change in the transfer characteristic by assuming that an element should be added to the normal control system. To express this idea, a component in the control system is divided into two elements — a *function element* and a *fault element*, which are connected in series as shown in Figure 5-1. The function element is in charge of the normal (without the fault) function, while the fault element is in charge of only the effect of the fault. Thus the control system of the robot and its faults are modeled as shown in Figure 5-2. In the figure, E_i(i=1,2,3) denotes each fault element and represents the faults as follows:

E_1 : faults in the PWM control part
E_2 : faults in the detection part of the armature current
E_3 : faults in the encoder pulse processing part.

Let the input and output of the fault element E_i be x_i and y_i, respectively. When no faults are occurred in the robot, the input to each fault element should be transferred directly to the output, i.e., $y_i = x_i$. Let n_i be the number of fault modes of E_i. Let the transfer characteristic of E_i when the j-th fault mode in E_i occurs be expressed as:

$$y_i = e_i^{\,j}\,(x_i) \qquad (5\text{-}1)$$

We call the function $e_i^{\,j}$ as a *fault modeling function.*

The fault modeling functions for the fault types classified in chapter 2 are shown in the column of the right end in Table 2-1. In the table, the input and output of E_i are expressed as x_{ik} and y_{ik} (k=0,1,2,...), respectively. x_{12}, y_{12} and x_{13}, y_{13} in the table are a little different from x_{i0}, y_{i0} in the point that those are based on the model including the effects of the blind sector in PWM control circuit†, while x_{i0}, y_{i0} are based on the model without those effects. They are prepared for recognizing fault phenomena in the PWM control part.

To get the transfer characteristic of each fault element of the robot controller, the input and the output of the element are calculated independently; the input to the element is calculated using only the reference data to the robot controller, while the output of the element is calculated using only the sensory feedback data.

The high frequency components of the sensory feedback data can be eliminated by a low pass filter so that the effects of the quantification error of the

† The blind sector is used for protection of the power transistors.

encoder and the A/D converter, the delay caused by the electrical time constant of the motor and the effects of other noises can be ignored. The transfer characteristic of each fault element is determined through the statistical processing to all the input and output data calculated above and is expressed in a mathematical function.

The set of the transfer characteristics expressed in the mathematical functions indicates the phenomena occurred in the robot.

The diagnostic system recognizes the phenomena by comparing the transfer characteristics of the robot controller with the model expressed as the fault modeling functions for all the types of fault.

6. METHOD OF FAULT PART IDENTIFICATION

6.1. INTEGRATION OF KNOWLEDGE ABOUT FAULTS

The knowledge available for the fault part identification of the robot controller are classified as follows:

(1) knowledge about the structure and functions of the system, and

(2) knowledge about the relations between fault phenomena and faults, and

(3) knowledge derived through experience or heuristic knowledge.

Fault tree is adopted to express and integrate these knowledge about faults.

Let us illustrate the method of integrating knowledge with an example. The control system shown in Figure 2-1 has two outputs (the angular velocity ω and the armature current I) for one input (the applied voltage V_0). Utilizing this feature, effective information for diagnosis can be derived; the method is to calculate the input and output in two ways by changing the combination of input and output and to compare both results. With this aim, let us introduce new inputs and outputs of fault elements. Let the input and output of E_1 calculated using only the V_0 and ω be x_{11} and y_{11}, respectively. Next, let the input and output of E_2 calculated using only V_0 and I be x_{21} and y_{21}, respectively. On the contrary, x_{i0} and y_{i0} (i=1,2,3) in Table 2-1 are calculated using all of V_0, ω and I. These dependencies are summarized in Table 6-1. Utilizing the difference between the combinations, we get a set of rules for judgement of the fault component among E_1, E_2 and E_3 as follows:

$$\left.\begin{array}{l} x_{11} \neq y_{11} \text{ and } x_{21} \neq y_{21} \quad ==> \quad E_1 \text{ fault} \\ x_{11} = y_{11} \text{ and } x_{21} \neq y_{21} \quad ==> \quad E_2 \text{ fault} \\ x_{11} \neq y_{11} \text{ and } x_{21} = y_{21} \quad ==> \quad E_3 \text{ fault} \end{array}\right\}$$

This set of rules can be represented in a fault tree as shown in Figure 6-1.

The heuristic knowledge derived through the experiments can be added to the fault tree. The fault tree needs to be extended by the knowledge through experiments repeatedly. Using the systematic analyses and knowledge derived through experiments, a fault tree for the robot controller is constructed as shown in Figure 6-2.

6.2. INFERRING MECHANISM AND KNOWLEDGE REPRESENTATION

As described above, the fault tree needs to be modified repeatedly through experiments. The framework of the production system is considered to be suitable for the diagnostic system — it allows us to develop the knowledge base step by step and also gives us opportunities to add new knowledge derived heuristically in the fields.

The inferring process is split into two phases — (1) to search in the fault tree, and (2) to check over the result of the search. In the first phase, all the events in the fault tree which become true by the results of the fault analysis are collected by the forward reasoning method. Starting from the top event in the fault tree, each event is evaluated from top to bottom.

Each event can be represented by the items shown in Figure 6-3. *Event status* in the figure denotes the status of each event in the process of the inference. To contain this status in the system, each event is assigned a variable — an *event status variable*. An event status variable should indicate the status of one of "not evaluated", "true" and "false". Every event status variable is initialized to be "not evaluated" at the start of the inference and is changed if needed in the process. The format to represent each rule is shown in Figure 6-4.

In phase (2) of the inference, the events which become "true" in the first phase of inference are checked over the overlaps and the inconsistencies of events.

7. CONFIGURATION OF THE FAULT DIAGNOSTIC SYSTEM

The prototype of the diagnostic system is composed of two modules — the fault detection and analysis module (FDAM) and the fault part identification module (FPIM). The FDAM is built in the robot controller. It is in charge of the fault detection and the fault analysis. The FPIM is outside of the robot controller and is constructed on a time-sharing based operating system. It is in charge of the fault part identification.

Thus the diagnostic system is distributed into two computers — one is configured to be suitable for real-time processings and the other is configured to be suitable for AI-based, memory-demanding processings.

7.1. FAULT DETECTION AND ANALYSIS MODULE

As shown in Figure 7-1, the configuration of the FDAM is based on CPU80186 (fault detection CPU) and floating co-processor 8087. The fault detection CPU communicates with servo CPU via the dual-port RAM and with the FPIM via the serial port (RS-232C).

The outline of the process in the FDAM is shown in Figure 7-2. Before the operation of the robot, the fault detection CPU synchronizes with the servo CPU. It gets servo data via the dual-port RAM during the operation of the robot, and detects a fault. Once it detects a fault, it issues an emergency stop command to the robot after collecting the servo data for a short period (currently the period is set to about 20 milliseconds). It then analyzes the fault phenomena, converts the results into ASCII strings and transmits them to the serial port.

7.2. FAULT PART IDENTIFICATION MODULE

As shown in Figure 7-3, the FPIM is composed of the data input and analysis part, the inference part and the display part. The inference part is constructed as a production system and is composed of the inference engine, the knowledge base (rule base), the fact base, the threshold table, the event status table and the work area. The threshold table contains the threshold values to judge whether the antecedent parts of the rules are true or not.

The outline of the process in the FPIM is shown in Figure 7-4. The FPIM gets the results from the FDAM in the format of ASCII strings via the serial port. The data input and analysis part analyzes the strings and stores the results into the fact base in the inference part. The inference part identifies the faulty part and prints out the results.

8. EXPERIMENTS AND DISCUSSIONS

The experiments to evaluate the methods are carried out on a 5-axis manipulator. An adapter to cause faults in the manipulator was added to its controller. During the operation of the manipulator, each type of fault was caused experimentally. Then the diagnostic system was examined whether it could detect them or not, how it analyzed the fault phenomena and how it could identify the faulty part.

The fault types selected to carry out experiments are fault-2,-3,-4,-6 and -7 in Table 2-1. Ten different speeds of the manipulator are selected for every fault type. The speeds are varied from -100% of the maximum speed of the manipulator to +100% of it at the interval of 20%. Five trials are carried out for every combination of a fault type and a speed.

8.1. EXPERIMENTS OF FAULT DETECTION

To judge whether a fault occurs or not, the threshold ε in equation (4-3) should be determined before the experiments. The procedure to determine it is as follows:

(1) to measure the responses of the manipulator in the normal state for a sinusoidal velocity reference input, and

(2) to calculate the value of $|\omega^{i}_{prd,F} - \omega^{i}_{F}|$, and

(3) to set the value of ε for the double of the maximum value of $|\omega^{i}_{prd,F} - \omega^{i}_{F}|$ derived in (2).

Let us define the *fault detecting rate* (FDR) and the *misreporting rate* (MRR) as follows:

$$\mathrm{FDR} = \frac{\textit{number of correct fault detection}}{\textit{number of all trials}}$$

$$\mathrm{MRR} = \frac{\textit{number of misreporting fault when no faults}}{\textit{number of all trials}}$$

As the result of the experiments, the FDR is 1.0 (100%) and the MRR is 0.0 (0%). That is to say, the diagnostic system could detect every fault type tested correctly — no failures in detection and no misreportings of faults.

Table 8-1 shows the maximum time among five trials to detect each type of fault. It shows the interval from the occurrence of a fault to its detection. As shown in the table, the diagnostic system could detect the faults quickly. But in the cases of the fault-3 at negative speeds, the results show that it took rather longer time than other cases. The causes of the delay in detection in these cases are considered as follows:

(1) the direction of the motion of the faulty axis is perpendicular, and

(2) the negative speed means the axis was moving downward, and

(3) the fault-3 is the type in which the applied voltage of the motor becomes 0V.

Because of these three causes, the axis is considered to have begun the free falling motion by the gravitational force and the inertial force when the fault occurred. At the early stage of the motion, the velocity of the free falling motion and the velocity reference does not differ so much. Thus the velocity fed back is regarded as normal by the diagnostic system at first. The diagnostic system did not recognize a fault until the difference became sensed. This is why the fault detection took a little longer time. It is regarded as a special case in which the effects by the faults did not appear quickly in the output and the robot was recognized as being "quasi-normal".

Except this, the time to detect each fault is at most 11.2 milliseconds.

8.2. EXPERIMENTS OF FAULT ANALYSIS

As shown in Table 2-1, the generic form of fault modeling functions are as follows:

$$y_{mn} = a_{mn}\, x_{mn} + b_{mn} \qquad (8\text{-}1)$$

The goal is to get the values of a_{mn} and b_{mn}. The procedure of the calculation is as follows:

(1) to calculate x_{mn} and y_{mn} from servo data (applied voltage of motor, armature current, and angular velocity),

(2) to calculate a_{mn} and b_{mn} from x_{mn} and y_{mn} using the method of least squares.

Table 8-2 shows the results for the fault types at speeds of 60% and -60% with the coefficients in the fault modeling functions. Compared with the corresponding coefficients, the parameters a_{mn} and b_{mn} are considered to be predicted precisely except in the case of the fault-2 at the speed of -60%. The cause of this difference is considered as follows. The fault type is that of no negative motor voltage, and the axis was moving in the negative direction, so that the voltage of the motor saturated quickly after the occurrence of the fault and stayed near that value. Therefore, the effects of the fault on the parameters appeared mainly in b_{12} — which denotes the offset — not in a_{12} (gain) by the method of least squares.

8.3. EXPERIMENTS OF FAULT PART IDENTIFICATION

As the result of the experiments, the diagnostic system could identify the faulty part correctly for every fault type tested. Table 8-3 shows the data of time spent in the inference. The data indicate the interval from the time when the fact base is complete to the time when the inference part finishes printing out the results on CRT. The data are measured using the system calls on a time-sharing based operating system[14] on which the FPIM is constructed. The average time and maximum time among all the fault types tested are 495 milliseconds and 835 milliseconds, respectively. Thus it is shown that the diagnostic system could identify the faulty part for the faults tested correctly and quickly.

9. CONCLUDING REMARKS

This paper presents the methods to diagnose faults in robots. The diagnostic process is divided into three stages — fault detection, fault analysis and fault part identification — and the method for each stage is described.

The authors developed a prototype of the diagnostic system to verify the methods and carried out experiments on a multi-axis robot manipulator. The results are as follows:

(1) the diagnostic system could detect every fault type tested quickly and correctly — no failures in detection and no misreportings of the faults,

(2) it could also identify the faulty part correctly and quickly for every fault type tested.

Thus the methods are effective and valid for the fault diagnosis of robots.

ACKNOWLEDGEMENTS

This research is done at the Nuclear Robotics Research in the Advanced Robotics Project and supported by the Agency of Industrial Science and Technology of the Ministry of International Trade and Industry in Japan.

REFERENCES

[1] M. E. Harris, "Avionics fault tree analyzer", AGARD Conf. Proc., No.343, pp.20.1-20.15, 1983.

[2] L. C. Doelp, G. K. Lee, R. E. Linney and R. W. Ormsby, "Quantitative Fault Tree Analysis: Gate by Gate Method", Plant Oper. Prog., Vol. 3, No.4, pp.227-238, 1984.
[3] K. Khodabandehloo, F. Duggan and T. M. Husband, "Reliability of Industrial Robots: A Safety Viewpoint", Proc. 7th British Robot Association Annual Conf., pp.233-242, 1984.
[4] H. Kumamoto, E. J. Henley and K. Inoue, "Signal-Flow-Based Graphs for Failure-Mode Analysis of Systems with Control Loops", IEEE Trans. on Reliability, Vol. R-30, No.2, pp.110-116, 1981.
[5] S. A. Lapp and G. J. Powers, "Computer-aided Synthesis of Fault-trees", IEEE Trans. on Reliability, Vol. R-17, No.3, pp.215-221, 1978.
[6] S. Contini and G. Squellati, "Automated fault tree construction", Eur. Rep. Comm. Eur. Commun., No. EUR-9168, 1984.
[7] P. Camarda, F. Corsi and A. Trentadue, "An Efficient Simple Algorithm for Fault Tree Automatic Synthesis from the Reliability Graph", IEEE Trans. on Reliability, Vol. R-27, No.3, 1978.
[8] N. H. Narayanan and N. Viswanadham, "A Methodology for Knowledge Acquisition and Reasoning in Failure Analysis of Systems", IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-17, No.2, pp.274-288, 1987.
[9] S. Garribba, E. Guagnini and P. Mussio, "An Expert System for Fault Tree Construction", Proc. Annual Reliability and Maintainability Symposium, pp.82-88, 1985.
[10] A. S. Wilsky and H. L. Jones, "A generalized likelihood ratio approach to the detection and estimation of jumps in linear systems", IEEE Trans. on Automatic Control, Vol. AC-21, pp.108-112, 1976.
[11] Chien, et al., "A Sequential Failure Detection Technique and its Application", IEEE Trans. on Automatic Control, Vol. AC-21, pp.750-757, 1976.
[12] A. S. Wilsky, "A Survey of Design Methods for Failure Detection in Dynamic Systems", Automatica, Vol. 12, pp.601-611, 1976.
[13] R. P. Paul, "Robot Manipulators: Mathematics, Programming, And Control", MIT Press, 1981.
[14] Tektronix, Inc., "4400 SERIES OPERATING SYSTEM MANUAL", 1986.

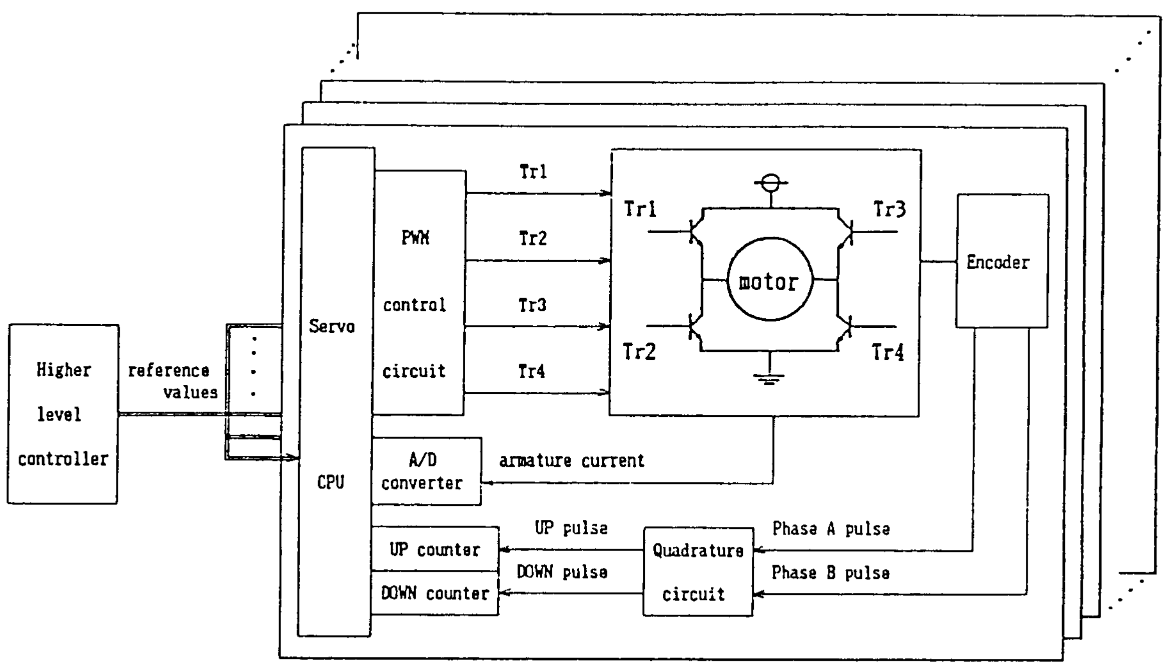

Figure 2-1. Conceptual configuration of hardware system

Table 2-1. Classification of faults

Fault type	Cause of fault	Phenomena by fault	Fault modeling function
fault-1	Tr1 stuck at OFF & Tr2 at ON , or Tr4 stuck at OFF & Tr3 at ON	Motor voltage value unable to be positive	$y_{13}=0.5x_{13}-0.5V_{max}$ (*)
fault-2	Tr1 stuck at ON & Tr2 at OFF, or Tr4 stuck at ON & Tr3 at OFF	Motor voltage value unable to be negative	$y_{12}=0.5x_{12}+0.5V_{max}$ (*)
fault-3	Tr1 stuck at OFF & Tr2 at OFF, or Tr4 stuck at OFF & Tr3 at OFF, or Tr1 stuck at ON & Tr2 at ON , or Tr4 stuck at ON & Tr3 at ON	Motor voltage is always 0 V	$y_{1e}=0$
fault-4	Tr1 stuck at OFF, or Tr4 stuck at OFF, or Tr2 stuck at ON , or Tr3 stuck at ON	Motor voltage is always 0 V when voltage reference is positive, but axis moves normaly when voltage reference is negative	$y_{1e}=0$ $(x_{1e}\geqq 0)$ $y_{1e}=x_{1e}$ $(x_{1e}<0)$
fault-5	Tr1 stuck at ON , or Tr4 stuck at ON , or Tr2 stuck at OFF, or Tr3 stuck at OFF	Axis moves normaly when voltage reference is positive, but motor voltage is always 0 V when voltage reference is negative	$y_{1e}=x_{1e}$ $(x_{1e}\geqq 0)$ $y_{1e}=0$ $(x_{1e}<0)$
fault-6	Phase A stuck at LOW , or Phase A stuck at HIGH, or Phase B stuck at LOW , or Phase B stuck at HIGH	Axis shows unexpected movement	$y_{3e}=0$
fault-7	Up pulse stuck at LOW, or Up pulse stuck at HIGH	Axis shows unexpected movement when velocity reference is positive, but moves normally when negative	$y_{3e}=0$ $(x_{3e}\geqq 0)$ $y_{3e}=x_{3e}$ $(x_{3e}<0)$
fault-8	Down pulse stuck at LOW, or Down pulse stuck at HIGH	Axis moves normally when velocity reference is positive, but shows unexpected movement when negative	$y_{3e}=x_{3e}$ $(x_{3e}\geqq 0)$ $y_{3e}=0$ $(x_{3e}<0)$
fault-9	gain change in output of current detecter	Control characteristics change	$y_{2e}=a x_{2e}$ (**)
fault-10	offset appearance in output of current detecter	Control characteristics change	$y_{2e}=x_{2e}+b$ (**)

(*) V_{max} is the maximum value of voltage that the motor can hold.
(**) a and b are constants.

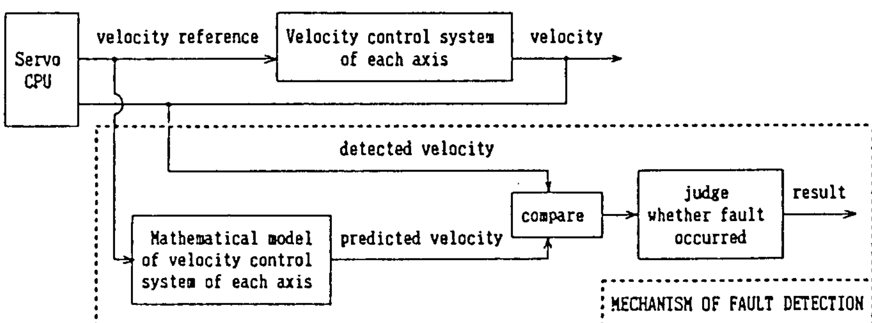

Figure 4-1. Method of fault detection

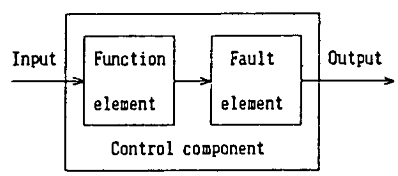

Figure 5-1. Expression of each control component

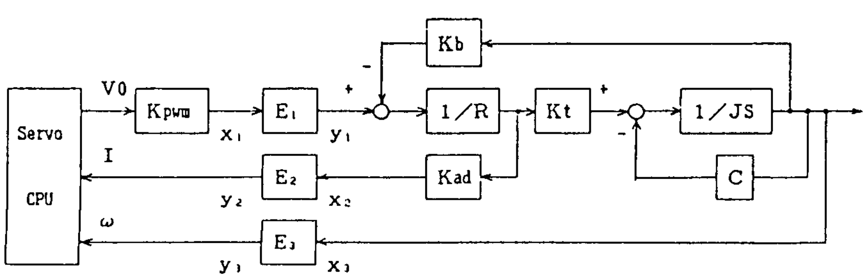

Figure 5-2. Fault modeling of control system

Nomenclature:
- V0 : Applied voltage
- I : Armature current
- ω : Angular velocity
- Kpwm : Gain of PWM circuit
- Kb : Voltage constant
- Kt : Torque constant
- Kad : Gain of A/D converter
- R : Armature resistance
- J : Inertia moment of motor and arm
- C : Viscous friction constant of axis

Table 6-1. Specification of x_{ik}, y_{ik}

Fault element	Input & output of fault modeling function		Servo data		
			V_e	I	ω
E_1	x_{10}	y_{10}	○	○	○
	x_{11}	y_{11}	○	×	○
E_2	x_{20}	y_{20}	○	○	○
	x_{21}	y_{21}	○	○	×

○ : used × : not used

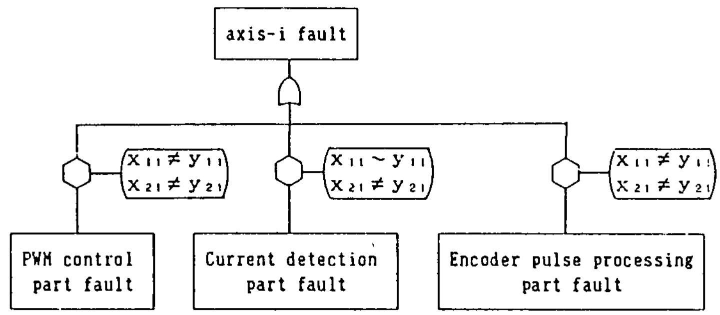

Figure 6-1. Basic fault tree

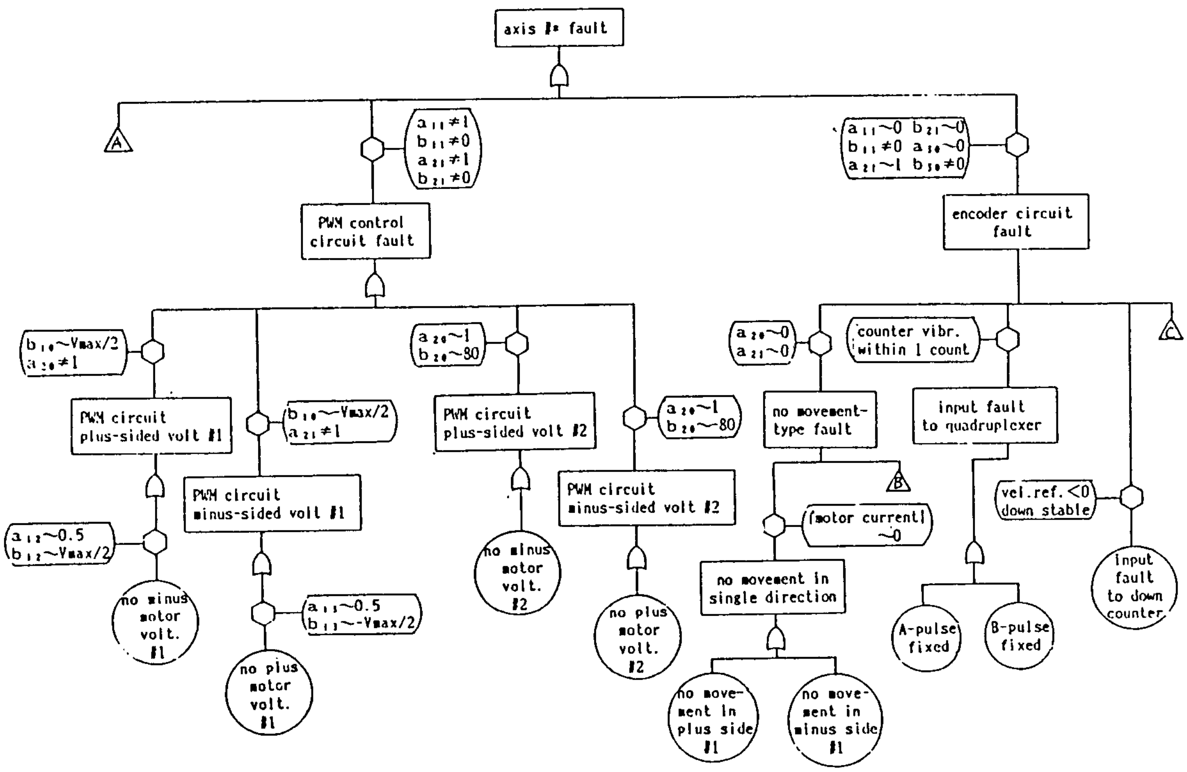

Figure 6-2. Fault tree (part)

Event id : #**
Event status : DOWN
Fault component : current detection part
Cause : fault in axis i control system
Antecedent : $a_{11} \neq 1$ && $b_{11} \neq 0$ && ……
Consequent : armature current detection part fault #1

Figure 6-3. Expression of an event in fault tree

```
if ((condition about event status variable 1) $$
    (condition about event status variable 2) $$
    .....................
    (antecedent 1) $$
    (antecedent 2) $$
    .....................
   )
then {
    activate consequent 1;
    activate consequent 2;
    .....................
}
```

$$: logical operator(AND or OR)

Figure 6-4. Format of a rule

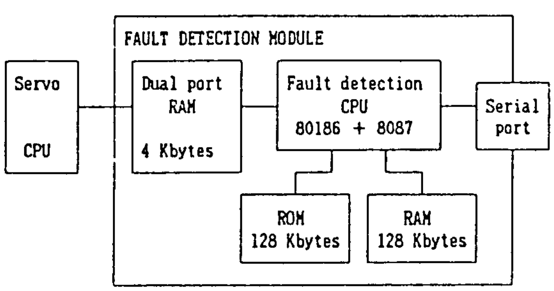

Figure 7-1. Configuration of FDAM

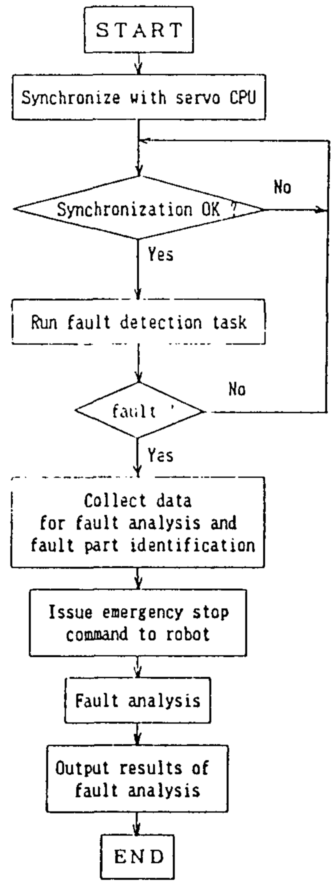

Figure 7-2. Flow in FDAM

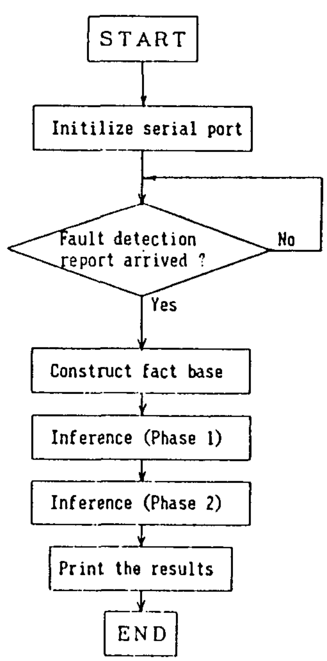

Figure 7-4. Flow in FPIM

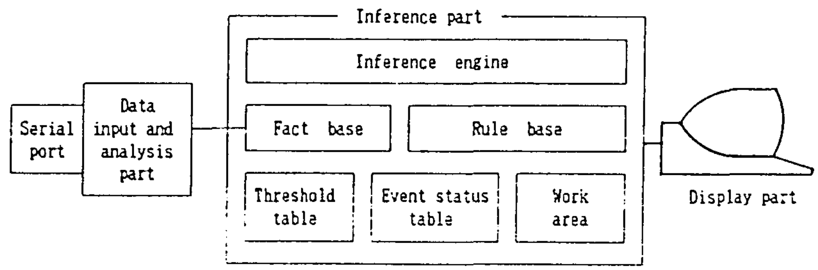

Figure 7-3. Configuration of FPIM

Table 8-1. Maximum time for fault detection (milliseconds)

speed	-100%	-80%	-60%	-40%	-20%	20%	40%	60%	80%	100%
Fault-2	4.8	4.8	4.8	4.8	4.8	7.2	8.0	7.2	8.0	8.0
Fault-3	24.8	24.0	22.4	25.6	23.2	9.6	9.6	9.6	10.4	8.8
Fault-4	–	–	–	–	–	11.2	9.6	8.8	8.8	9.6
Fault-6	1.6	1.6	0.8	0.8	0.0	1.6	0.8	0.8	0.8	0.0
Fault-7	–	–	–	–	–	1.6	0.8	0.8	1.6	0.0

Table 8-2. Results of parameter identification in fault analysis

upper row : predicted value
lower row : detected value

Fault type	speed	a_{12}	b_{12}
Fault-2	+60%	0.5 0.384	50.0 43.7
	-60%	0.5 1.172	50.0 112.9

Fault type	speed	a_{13}	b_{13}
Fault-3	+60%	0.0 0.055	0.0 11.7
	-60%	0.0 -0.007	0.0 -21.1
Fault-4	+60%	0.0 -0.221	0.0 34.0
	-60%	0.0 0.113	0.0 -12.9

Fault type	speed	a_{33}	b_{33}
Fault-6	+60%	0.0 -0.010	0.0 0.1
	-60%	0.0 0.005	0.0 0.0
Fault-7	+60%	0.0 0.000	0.0 0.0
	-60%	0.0 0.000	0.0 0.0

Table 8-3. Time for fault part identification

Fault type	Inference time (milliseconds)	
	maximum	average
Fault-2	772	515
Fault-3	749	529
Fault-4	762	386
Fault-6	759	497
Fault-7	835	550