Transputer control of a flexible robot link

Computer control of lightweight robot arms is complex; the mechanical flexibility of the arms makes it difficult to position the end-effector accurately. **A C J Stavenuiter, G Ter Reehorst and A W P Bakkers** show how a transputer control system can help

The applicability of transputers in control systems is investigated. This is done by implementing a controller for a flexible robot arm with one degree of freedom on a system consisting of an IBM-AT and four transputers. It is found that a control system with transputers offers a great improvement compared with conventional digital control systems. Transputers can solve the common problem in control practice, i.e. having very sophisticted controllers but not being able to implement them because they need too much computing time. However, transputers are not an optimal solution for more sophisticated control systems because of shortcomings in the scheduling mechanism.

control systems transputers robots

Today's robots, although built to huge sizes, are only able to manipulate masses that are small compared with their own weight. In some applications, as in automotive robots, light arms are preferred. A relatively low mass can be driven by small motors and higher accelerations can be achieved. However, because of the inherent flexibility of light arms it is much more difficult to position the end-effector accurately.

To study the problems in controlling light, flexible robot links, a 'robot' was constructed as a long aluminium strip (190 × 6 × 0.4 cm) driven by a DC motor. Until recently this robot was controlled by a Z80 microprocessor system with a floating-point coprocessor, but it was soon found that this system was too slow for accurate control of the robot arm. Evidently, more processing power was needed. This could be obtained by faster processors like the Motorola 68020. However, with the increasing complexity of control algorithms, the limits of these processors will soon be reached as well. Parallelism is the key to the solution of this problem: a parallel system can be expanded gradually to a high capacity, although one has to be aware that not every problem can be solved in parallel without introducing a large communication overhead.

The transputer has been designed for building concurrent systems, so we have investigated whether transputer networks can be useful in complex robot control applications.

REQUIREMENTS FOR DIGITAL CONTROL SYSTEMS

Digital control systems have a specific quality; the plant to be controlled has to be sampled at specific equidistant moments. It is this strict equidistant sampling that distinguishes digital control systems from other realtime systems. Generally speaking, two types of tasks exist in such a system, i.e. time-bounded and time-limited tasks[1]. Time-bounded tasks (the sample and control actions) have to be executed at specific moments; time-limited processes (computing processes) have to be executed before specific moments. This results in a need for scheduling, synchronization and communication.

Here we encounter an advantage and a disadvantage of the transputer. Though ideally suited for communication and synchronization, the scheduling mechanism of the transputer is not well suited to handling different priority levels[2]. One possible solution is to write a custom scheduler. However, this is a difficult solution for the transputer because one has to bypass the hardware scheduler. Furthermore, it imposes a large overhead on the system: the scheduler has to be triggered at a frequency much higher than the sampling frequency to obtain accuracy with respect to the sampling moments. It has been shown[1] that it should be possible to implement

Control Systems and Computer Engineering Laboratory, Department of Electrical Engineering, University of Twente, PO Box 217, 7500 AE Enschede, The Netherlands

Paper received: 15 August 1988. Revised: 8 December 1988

0141-9331/89/03227-06 $03.00

an extended scheduling mechanism in the transputer hardware but, given the current transputer, a system well suited to control applications will be hard to realize.

IMPLEMENTATION OF DIGITAL CONTROLLERS

There are three approaches to realizing a realtime digital controller.

Dedicated approach

It is possible to write a dedicated program for each application. The advantage of this approach is that it yields fast programs that are optimal for the given application. The disadvantages of this approach are the complexity of scheduling and monitoring of all the processes; the time needed to debug the software; and the fact that for each new application, and for each change in the application, a new program has to be written. But it is still the approach most widely used.

Realtime operating system

A realtime operating system can be used and the program can be written in a normal language. Realtime operating systems offer the necessary primitives for scheduling, communication and synchronization, thus facilitating the creation of realtime programs. Also, realtime operating systems are better protected than dedicated application programs; there is no danger of one program accessing resources granted to another program. However, this protection comes at the price of extra CPU overhead. Unfortunately, no realtime operating systems are available for transputers yet.

Realtime language approach

As a third option, a realtime language can be used. A realtime language provides a number of special constructs for realtime applications. The advantage of realtime languages lies in the greater checking of the compiler, and in the reduced overhead compared with realtime operating systems. Unfortunately, no realtime language for transputers exists at present. In our group an OCCAM-like language with limited realtime capabilities has been developed. This language is executable on an IBM AT-compatible and experience with current controller implementations has shown the value of a realtime language that relieves the implementer from the burden of handling the time-dependent tasks.

Due to a lack of available alternatives, the 'dedicated approach' was selected for the controller described in this paper.

OVERVIEW OF THE CONTROLLER

This section gives a brief description of the basic principles of the controller used in the flexible arm set-up. More details can be found in Appendix 1 and in Reference 3.

Figure 1. Robot arm under vibration

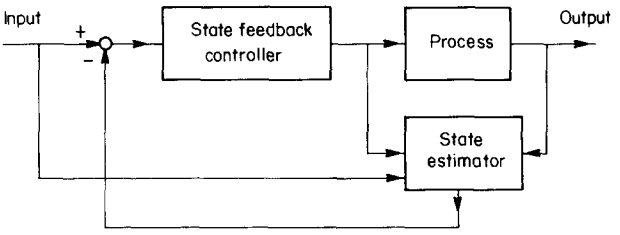


Figure 2. The controlled process

The process to be controlled consists of a long flexible bar attached to a motor. If the motor starts or stops, the bar will bend and vibrate (Figure 1). The longer and more flexible the bar, the stronger these vibrations will be and the longer they will last. These vibrations are the result of the (infinite number of) resonance frequencies of the bar. A vibration mode is associated with each one of these frequencies. By applying appropriate control, it is possible to suppress the vibration modes due to the accelerating or decelerating motor. Figure 2 gives an overview of the architecture of the controller. Each mode can be described by two state variables. The system can be given a desired response by appropriate feedback of these state variables. Unfortunately, the state variables cannot be measured directly, so it is necessary to estimate them with a state estimator. However, this increases the amount of computation considerably.

In principle, the number of vibration modes is infinite, but for the given application, it can be shown[3] that only the three modes with the lowest resonance frequencies are important and, furthermore, that control of each resonance frequency is independent of the control of the others. This is an important aspect since it enables independent computation of the control signals for the various modes. Addition of these control signals yields the control voltage for the entire bar.

APPLYING THE TRANSPUTER

As mentioned above, the control of the arm was first implemented on a 4 MHz Z80. It was found that the time needed to perform all the calculations for the control of one mode was about 40 ms, yielding a maximum sample frequency of 25 Hz. This time was halved by adding a floating-point coprocessor, but the dynamics of the system did not allow control of more than one mode. As

the resonance frequency of the third mode is about 13 Hz, and using the rule of thumb that the sampling frequency has to be ten times the highest system frequency, the minimal sampling frequency had to be 130 Hz. Therefore, the controller has been implemented on a multitransputer system.

Hardware architecture

A system with four transputers was chosen because a master-slave configuration could be set up in such a way that each slave performs the computations for one of the three modes, while the master takes care of communication with the outside world. Furthermore, the master computes the contributions of the various modes from the strain gauge signals and feeds each slave with the necessary information. Also, it computes the actual motor control signal from the results delivered by each slave.

The total control system consists of

- A 1.9 m long aluminium bar.
- Three pairs of strain gauges to measure the vibrations.
- A Whetstone bridge for the strain gauges.
- A DC motor with resolver to measure the motor angle.
- A motor amplifier.
- An IBM AT with three extra cards
  - RDC/DA-card for digital-to-analogue conversion of the control signal and with a resolver-to-digital converter for angle measurement.
  - A/D card: analogue-to-digital converter for the strain gauge measurements.
  - T4, with four T414s, or four T800s, on one card.

Figure 3 gives an overview of the architecture of the hardware of the system.

Software architecture

The software has to perform five tasks

- measuring and actuating
- terminal I/O
- estimation of the state variables

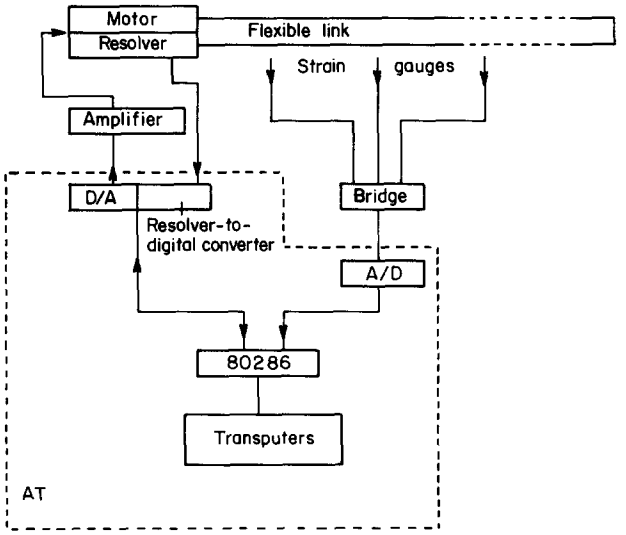


Figure 3. Hardware architecture

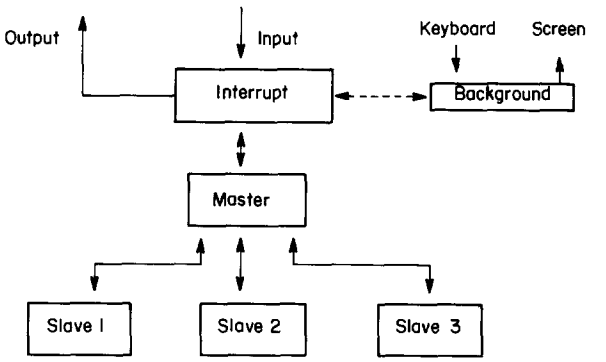


Figure 4. Software architecture

- computing the control signal for each mode
- computing the total control signal.

Figure 4 illustrates the software architecture of the system.

All computations are done by transputers. The IBM AT serves as an intelligent interface to the transputer system and provides the user with interface facilities like disc handling, graphics etc. The A/D card is programmed to generate the clock interrupt that initiates the sampling. It would be more natural to let the transputer initiate the sampling, but this solution has the disadvantage that communication between the transputer system and the IBM AT increases, and that there is some uncertainty about the actual sampling moment. Using the event line has the disadvantage that there is no guarantee that the transputer will respond immediately. For this particular system, where all the software can be viewed as one interrupt routine, this would cause no problems, because all computations and communication have to be finished before the next sampling moment. However, in a system with several control loops and with different sampling frequencies use of the event line could deteriorate the control.

RESULTS

The control algorithm was implemented in OCCAM and some performance measurements were taken of the speed of the computer system and the behaviour of the robot link.

Performance of the transputer

Table 1 compares minimum sample times on various processors. It is obvious that using transputers reduced sample times considerably compared with conventional digital systems.

Table 1. Minimum sample times in milliseconds

Processor	1 mode	3 modes	relative
Z80 + coprocessor	20	60	10
10 MHz AT	15	45	7
10 MHz AT + 80287	2	6	1
20 MHz T414 (×4)	0.71	0.71	1/8
20 MHz T800 (×4)	0.22	0.22	1/30

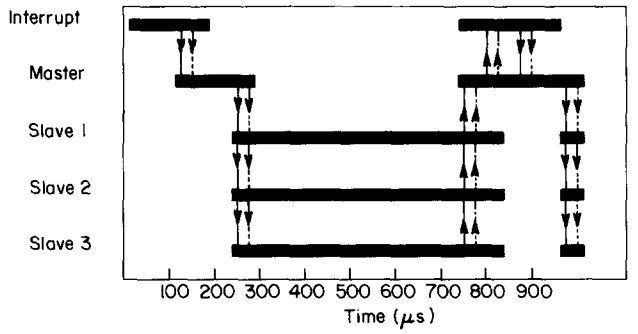


Figure 5. Division of workload

Figure 5 shows the workload on the T414s and the IBM AT. Notice that, after the master transputer resumes its work, the slaves perform some preprocessing of data which can be used in computations on the next samples. The interrupt routine on the IBM AT varies from 160 to 180 μs because the A/D conversion time is data dependent. Time measurements on the transputers were made using the transputer timer. The master transputer was idle for a relatively long time, but in this application it was not possible to achieve a better division of workload.

The entire transputer code was written as a sequential program. The software on the slaves was also implemented in a way which allows a more general approach; various standard blocks (integrator, gain, summator etc.) were executed in parallel, but it was found that the minimum sampling time increased considerably due to the increased process scheduling and communication.

Control using T800s at a sampling frequency of 4.5 kHz showed an inferior behaviour of the robot link compared with control at a sampling frequency of 1 kHz. This is caused by the fact that 160–180 of the 220 μs were used by the IBM AT for measuring and actuation; this implies that the sampling interval cannot now be considered to be instantaneous, an assumption that was made in the design of the controller. Of course, a 4.5 kHz sample frequency for this robot link is a little overdone, but this test clearly shows that we have now passed the computing bottleneck, and that, to make full use of the transputer's computing power, a faster communication between the transputer system and the outside world is necessary.

Performance of the controller

Figure 6 shows the output angle and the motor voltage when the motor is forced to make a step and no attempt is made to control the vibrations; only the motor angle is controlled. The output angle is the angle between the arm at rest and a straight line between the base and the end of the link.

Figure 7 shows the three lowest resonance frequencies when no attempt is made to suppress them. In both figures it can be seen that the arm exhibits strong vibrations which decay only slowly. Figure 8 shows a step response when the first mode is controlled. The behaviour of the arm approaches the behaviour of a rigid arm, but still the contributions of the second and third mode are visible. Figure 9 shows a step response with three modes controlled. Except for a short time immediately after the start of the step, the vibrations are completely suppressed.

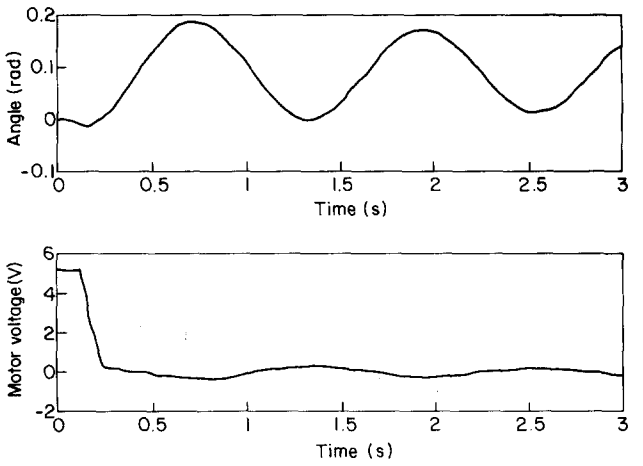


Figure 6. Motor controlled step response

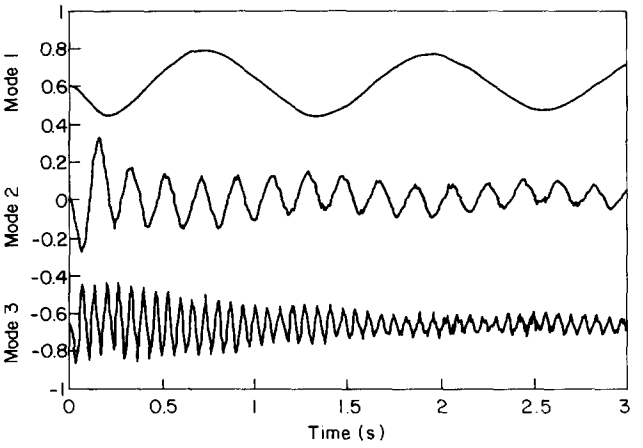


Figure 7. Motor controlled resonances

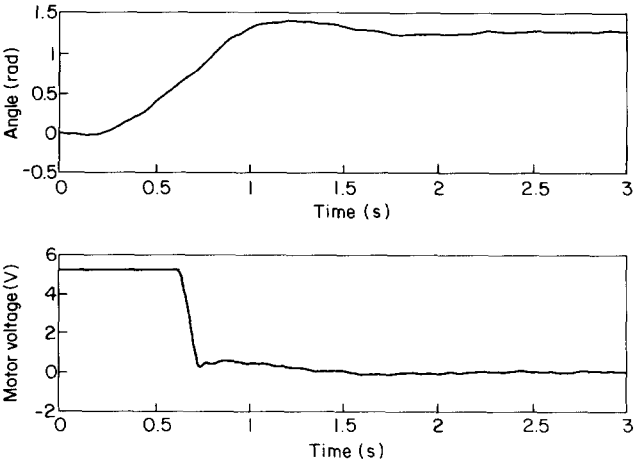


Figure 8. One mode controlled step response

FUTURE PLANS

The current robot set-up has no practical use; it has only one degree of freedom and the flexible link is not strong enough to lift more than 0.5 kg. Currently, a new robot, consisting of two flexible links and a gripper, is being developed.

A possible control scheme for this robot is a two-level controller: a high-level controller for Cartesian control of the end-effector, and a low-level controller taking care of

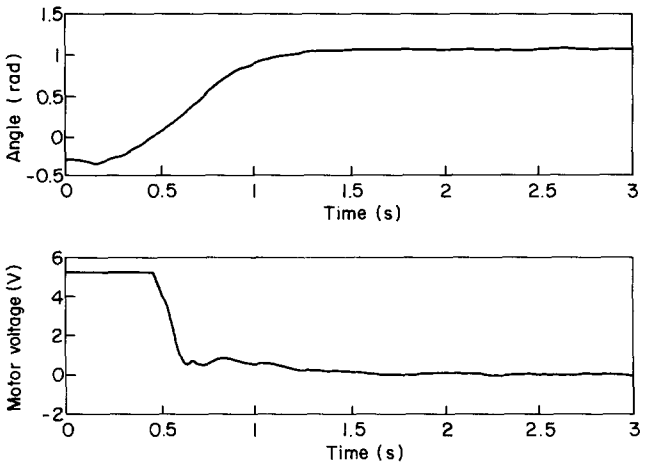


Figure 9. Three modes controlled step response

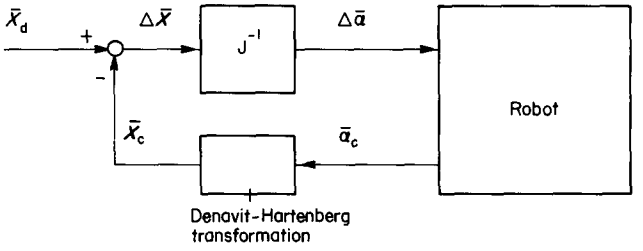


Figure 10. Cartesian control

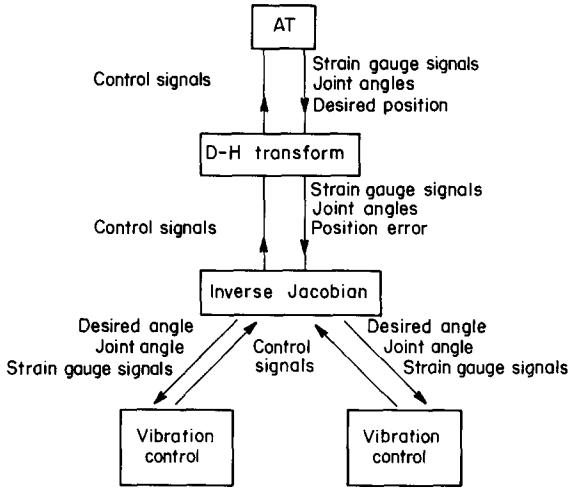


Figure 11. Multi-link control architecture

the suppression of the vibrations in the links. Figure 10 illustrates a Cartesian controller based on the Denavit-Hartenberg (D-H) transformation[4].

The robot's joint angles $\bar{\alpha}_c$ are transformed into Cartesian coordinates $\bar{X}_c$ with the D-H transformation, the inverse Jacobian transforms the Cartesian $\Delta\bar{X}$ into joint angle errors $\Delta\bar{\alpha}$. If the dynamics of the robot links are neglected, this joint angle error can be used directly as the control signal for the various motors. In the case of a flexible robot, this Cartesian controller has to be extended with a low-level controller.

The architecture of the control system for the single flexible link could be used as a control subsystem for the new robot. Figure 11 illustrates a possible architecture for this control system.

CONCLUSIONS

Until recently it was not possible to put the latest developments in control theory into practice; the algorithms needed too much computing time, and the necessary sampling frequencies could not be achieved. With the development of the transputer, this problem has disappeared. Transputers are not yet suited for complex realtime systems, however; when accuracies in the millisecond range are required another scheduling mechanism is necessary. Attention should be paid to interfacing a transputer system to the outside world; it is advisable to use more than one link for the interface with other systems to prevent this interface from becoming the bottleneck of the transputer system.

REFERENCES

1 **Wijbrans, K C J** 'Design of a real-time scheduler for a new transputer generation' *MSc thesis, University of Twente, Enschede, the Netherlands*

2 **Inmos** *Transputer reference manual* Prentice-Hall, Hemel Hempstead, UK (1988)

3 **Kruise *et al.*** 'Modelling and control of a flexible robot link' *Proc. 1988 IUTAM/IFAC Symp. dynamics of controlled mechanical systems*, to be published

4 **Craig, J J** *Introduction to robotics* Addison-Wesley, Reading, MA, USA (1986) pp 245–249

5 **Fukuda, T and Kuribayashi, Y** 'Precise control of flexible arms with reliable systems' *Proc. Int. Conf. advanced robotics, Japan Ind. Robot Assoc., Tokyo, Japan* Vol 1 (1983) pp 237–244

APPENDIX 1. PRINCIPLES OF THE CONTROLLER

It can be shown[5] that the bending of an accelerating link can be described by

$$W(r,t) = \sum_{n=1}^{n=\infty} y_n(r) M_n(t) \qquad (1)$$

where $y_n(r)$, called the 'mode shape' function, describes the shape of the arm at its nth resonance frequency, where $M_n(t)$, the modal function, is a time dependent function which can be described by

$$\omega_i^2 M_i + 2z_i\omega_i \dot{M}_i + \ddot{M}_i = -A_i \ddot{\Theta} \qquad (2)$$

where ω_i is the ith resonance frequency, $\ddot{\Theta}$ is the acceleration of the arm, and z_i and A_i are arm dependent parameters.

The combination $y_i(r)\, M_i(t)$ is called a mode. Obviously the mode shape function is something we cannot influence. However, the flexible arm can be made to rotate like a rigid one by controlling the acceleration in such a way that the modal functions decay rapidly. In practice, it is impossible to control an infinite number of modes, but the model of the arm can be simplified by assuming that

- The motor bandwidth is limited to ω_{mo}, which implies that excitation of frequencies higher than ω_{mo} may be disregarded.
- The bandwidth ω_c of the system is less than ω_{mo}, so the motor may be considered an integrator. This implies

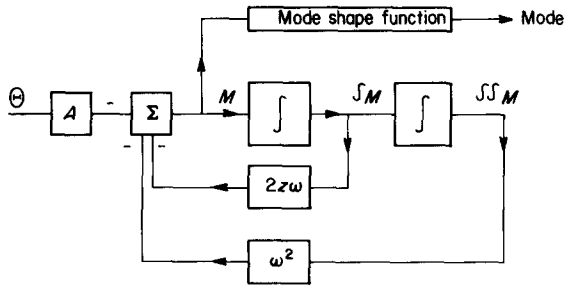


Figure 12. Model for one mode

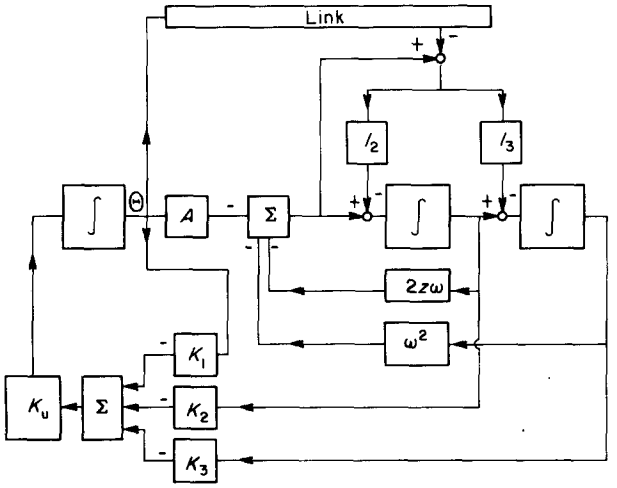


Figure 13. Controller and estimator for one mode

that modes with a resonance frequency higher than ω_c may be disregarded.

The number of modes, n, in the reduced model can thus be determined by $\omega_{n+1} > \omega_c$. Simulations indicated that control of the first three modes should yield a satisfactory response[3]. Furthermore, as these modes are independent the control signal for the entire system can be computed easily by adding the control signals for the three modes.

To obtain a model of the system the Laplace transform of Equation (2) is taken, yielding

$$s^2M_i + 2z_i\omega_iM_i + \omega_i^2M_i = -A_is^2\Theta \qquad (3)$$

which, omitting the subscripts, can be written as

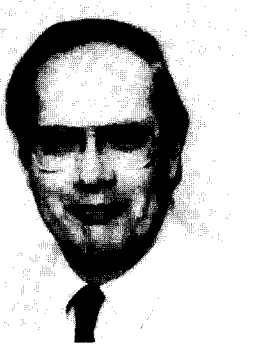
$$M + \frac{2z\omega M}{s} + \frac{\omega^2 M}{s^2} = -A\Theta \qquad (4)$$

and can be translated to the model in Figure 12.

The modal function M is controlled using state feedback, where the states are defined to be Θ, $\int M$ and $\int\int M$. Θ can be measured, $\int M$ and $\int\int M$ have to be estimated. The estimation is corrected by measuring the vibrations of the arm. As there are three nodes which contribute to the vibrations of the arm, three pairs of strain gauges were used in order to be able to compute the contribution of each mode. Figure 13 shows the controller and estimator for one mode. A mathematical equivalent of the controller and estimator is implemented on each slave.

Tonny Stavenuiter received an MSc in electrical engineering in 1987. The subject of his MSc thesis was robot control with transputer emulators. Since then he has been doing research in transputer applications for robot control.

Gerrit ter Reehorst received an MSc in electrical engineering in 1988. The subject of his MSc project was the control of the flexible arm described in this paper. He is now doing research in industrial transputer applications.



André Bakkers is currently senior scientist in the control group of the University of Twente, The Netherlands. He received an MSc at the RIT in Rochester, NY, USA. He has worked as research scientist at Saclancen in La Spezia, Italy, and as senior development engineer at General Dynamics in Rochester, NY, USA.