SRC-TR-87-49

A Microcomputer Based Expert Controller for Industrial Applications

by

L. Lebow

G.L. Blankenship

A MICROCOMPUTER BASED EXPERT CONTROLLER FOR INDUSTRIAL APPLICATIONS[1]

L. Lebow and G.L. Blankenship

Systems Research Center and Electrical Engineering Department, University of Maryland, College Park, Maryland 20742 USA

1 Introduction

Expert control as defined by Astrom, Anton and Arzen [7,6,1,2,3] involves the construction of a composite control structure for a complex process which includes supervisory functions, adaptive control algorithms and low level control laws all managed by an expert system which monitors process parameters and control system performance. In [1,2,3] a prototype expert controller was built using high level tools on a super-mini computer.[2] This work and the related work of Moore and colleagues [11] and others [10] has demonstrated the potential value of expert systems in management of the full range on-line control functions from alarms to single loop PID feedback elements.

In this paper we report on our efforts to produce a practical implementation of an expert (industrial) controller on microprocessor based systems. Expert systems are briefly discussed and the structure of an expert controller is outlined. As an initial step in the development of this implementation, an adaptive PID (proportional, integral and derivative) controller is being constructed. It is described in some detail; and the enhancements needed, and how they are to be accomplished, to transform the current implementation into an expert controller are then discussed.

Expert systems are computer programs designed to aid humans in complex tasks. By representing the knowledge about a given domain in the proper manner, and providing an "inference" control structure for access to the knowledge base, an expert system can "reason" to solve a problem

[1]This research was supported in part by a grant of equipment and funds from the Industrial Systems Division of Texas Instruments, in part by the Engineering Research Center of the College of Engineering, University of Maryland, and in part by the Systems Research Center under NSF Grant CDR-85-00108.

[2]Specifically, the forward chaining production system YAPS and the object-oriented Flavors system running on a VAX 11/780.

or perform a difficult task. Such problems may have limited, conflicting or unreliable data and more than one solution. The expert system is usually designed to emulate human behavior by employing some heuristic "rules of thumb" to reduce many solution possibilities to a few "good" ones.

In general, an expert system can be broken into three parts, a knowledge base, an inference engine, and a user interface. The knowledge base contains facts about the domain of interest as well as rules defining the relationships among the facts. The inference engine processes these rules and generates the solution or possible solutions to a problem. This inference can be data driven (forward chaining) or goal driven (backward chaining), or both. The inference engine may generate a search tree that represents possible paths to a solution. Rules are used to *prune* this search in an effort to find a best solution. The user interface allows a problem to be stated to the expert system along with any data that may apply. It usually provides a means for the expert system to request more data and display intermediate results as well as a final solution, and an "explanation" of the steps taken to reach the solution.

Expert systems may handle large amounts of data and solve problems of high complexity. Most of the systems in current use are static and time invariant, which is not the case with expert controllers [7,2,3]. In many conventional expert systems human interaction is often expected and required in the pursuit of a solution This helps the system eliminate unpromising paths in the search for a solution and reinforce promising ones.

2 Expert Control

An expert controller as a decision making element in a feedback control loop requires much the same decision making ability needed in other expert systems, but there are significant differences. One crucial requirement is the need to produce expert behavior in "real time". Not only must the expert controller respond quickly, but its operation interacts with the process in a dynamic time varying environment. Also, the expert controller must be interfaced directly to a process and be equipped with the means for applying control to the process.

Many current industrial controllers include some heuristics for *safety net* procedures [7]. These heuristics may only handle extreme "alarm" type situations, and a prescribed solution may be a plant shutdown until human intervention can solve the problem. An expert controller should have the

ability to *adapt* to changing situations and prevent most "alarms" from ever occurring.

Two examples illustrate possible roles for expert controllers. Variable controller parameters allow a single control algorithm to change as the process changes. Changes in a process may come from deterioration of valves, pumps and other mechanical devices in the plant. These changes are somewhat slow; but, if optimal performance is important at all times, it is clearly advantageous to have the controller change its parameters as the process changes. An expert controller might manage the selection and execution of different adaptive control algorithms [5,6,8] to maintain the control parameters at or near their optimal values for the specific process conditions. In emergency situations where major elements in a system break or falter, an expert controller may manage the reconfiguration of the control algorithm or switch to another more appropriate or robust control algorithm. If the bandwidth of the response of the expert controller is large enough, such capabilities would be extremely useful in aerospace applications where a loss of control may result the loss of life or valuable equipment. One might also argue that most situations requiring control over a wide range of (unpredictable) operating conditions could benefit from expert control.

An expert controller should have the capability of using several different control algorithms as well as the ability to *tune* the parameters of each algorithm to the process under control. Possible control algorithms might include Proportional, Integral and Derivative (PID), pole-placement, linear observers or algorithms designed for optimal control. The expert controller must provide control signals to the process (in "real time") in addition to "reasoning" about what control laws or algorithms are to be applied. Ultimately the expert controller should have available, an entire library of relevant algorithms for process control and identification. The job of the expert system would then be to orchestrate the application of these various algorithms [7]. The knowledge base consists of experiential knowledge about the process along with facts and rules that are used to infer which control algorithm to apply and what the current parameter settings for that algorithm should be. By periodically applying identification routines and monitoring the results, the expert system could accumulate more and more information about a given process to find the best control law. Identification algorithms might include methods for estimation of critical gain and periods and to Least-Squares algorithms for process parameter estimation [7].

By keeping track of all control algorithms applied and their respective effects, a history about the process could be compiled. Such a history could

be used either for user information and education, or by the expert system itself in a comparative study versus a reference process and its control history. Should the expert controller be able to recognize a given process configuration based on such a comparison, the optimal control might already be known. The expert system should continuously manage the search for the best control law and its optimal parameters. As a process goes through gradual changes, perhaps due to mechanical wear, the expert system should respond by directing adaptive changes in control parameters. When major upsets enter the system the expert controller should be capable of recognizing the situation and quickly responding with a control law that accommodates the new configuration.

A major requirement for the expert controller lies in the "real time" requirements of process sampling and control. In architecture proposed here the control algorithm implementation (discrete time and digitally computed) resides in a separate microprocessor from that employed by the knowledge base and inference engine. Current technology, both hardware and software, are only just now approaching technical abilities that would allow inference to proceed fast enough to avoid undesirable delays in the application of a control law to a process. With the development of Lisp chips and compiled Prolog inference systems, such high speed inference may not be far off. However, separation of control algorithm implementation and the inference procedures seems more efficient and flexible within the current technology. Flexibility is provided by the fact that inconsistent search times in picking a new set of parameters or a new control law do not effect the ongoing actual control of the process. In this configuration the expert system itself co-exists with the control law processor, and the union is an "expert controller."

In this realization the expert system may be viewed as having two "user" interfaces. The controller (control algorithm computer) takes intermediate control solutions and applies them to the process. This process, regarded as an expert system "user," responds to the control in some fashion providing the expert system with more information about itself and the relative success of the current solution. The second user, a human who is monitoring the expert system, may intervene with a "suggested" control law, new parameters, or some constraints on what the expert system should consider as a "good" response. This human user interface also indicates that a separation of control law implementation and the actual expert system is desirable. Communication with humans takes a great deal of time in relation to other computer tasks. This time should not interfere with the smooth ongoing application of a control algorithm.

The expert controller is then a complex system that is able to monitor its environment, make decisions based on experiential and new information as well as known facts, and enact those decisions in a feedback loop. The first level of adaptability is provided by the ability to tune the parameters of a given control law, perhaps through an auto-tuning control strategy [5,6,8]. Alternately, the expert controller could use inference to pick the proportional, integral, and derivative coefficients in a standard PID control algorithm. A strategy for this is discussed in in Section 3. A more challenging level of adaptability is the capability of changing from one control law to another. This implies the use of a combination of process identification algorithms in conjunction with a library of control algorithms. Continued monitoring and adjustment should eventually lead to optimal control within a selected class. Ultimately, the expert controller might be a learning system that could synthesize control algorithms on its own.

3 Microcomputer Based Expert Controller

The main thrust of the present study is to produce a microcomputer based expert controller using available industrial controller technology with some enhancements. The expert controller is based on hardware supplied by Texas Instruments Inc. and is essentially built about a Programmable Logic Controller (PLC) and its supporting elements. The enhancement that provides this standard controller with the potential to support expert capabilities is the addition of personal computer (PC) capabilities, including a direct communication link with the programmable controller. The personal computer technology allows the expert system to be programmed in high level languages such as C, Lisp, Prolog etc. The hardware and software capabilities of this system provide all the power necessary to build and program a expert controller for certain industrial applications.

The expert controller is being developed in stages. An adaptive "intelligent" PID controller has been chosen as a primary goal. PID control enjoys wide acceptance in industry and its properties are well understood. The secondary, or next goal would be, to develop a controller that relies on adaptive PID control as its basic mode of operation but has the ability to *learn* about the process. Using this *experience*, the enhanced version could switch to more sophisticated control algorithms such as pole placement or linear observers. The initial system should lay the foundation for this future system. Hence, the current configuration should provide the flexibility for

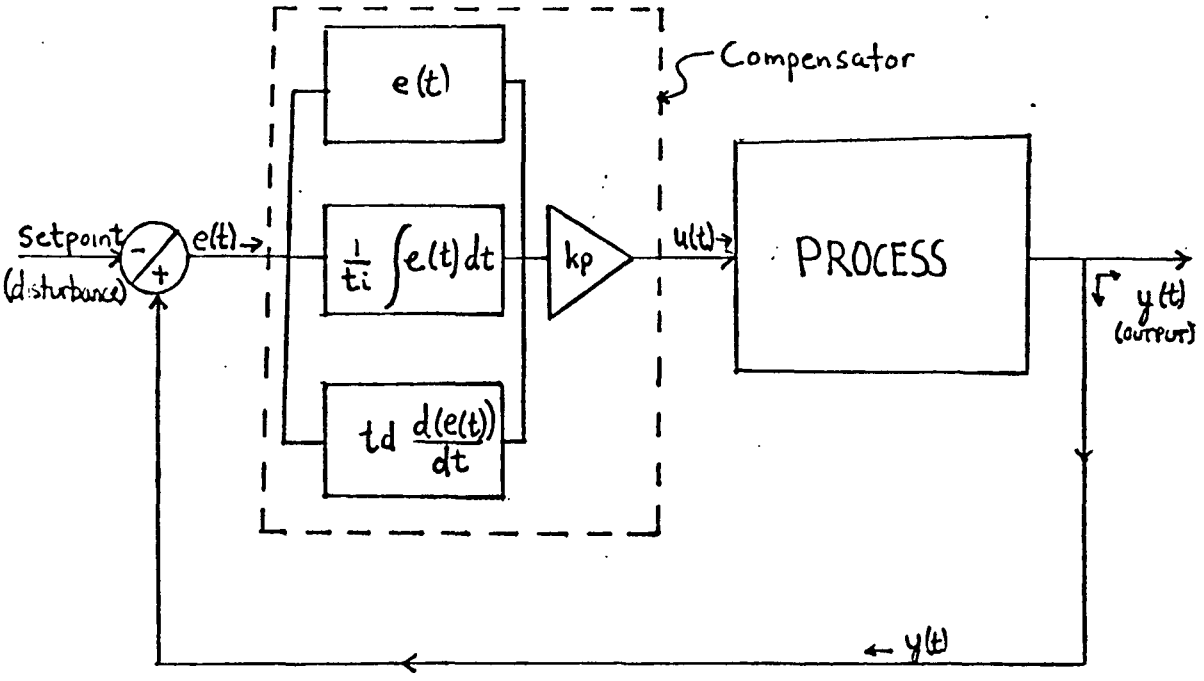


Figure 1: PID control loop.

enhancements while functioning as an adaptive PID controller.

Two kinds of tuning may be defined for PID controllers [4,8]. The first is the Zeigler-Nichols method for auto-tuning which we will be referred to as "pre-tuning". The following paragraph discusses this "pre-tuning" mode in some detail. This is followed by a description of the current implementation which employs this type of tuning. The discussion of the second type of tuning "continuous tuning," is postponed until the final section dealing with ongoing developments and future enhancements.

3.1 Self-tuning PID Control

The objective of PID control is to constrain a process response to follow input disturbances or setpoints. In general, the output is compared to the setpoint and the error signal $e(t)$ is computed and fedback to a compensator that combines it with its integral and derivative to produce the control signal

$u(t)$ as shown in equation (1) and Figure 1.

$$u(t) = k_p \left[e(t) \quad + \quad \frac{1}{t_i} \int e(t)\, dt \quad + \quad t_d \frac{de(t)}{dt} \right] \tag{1}$$

The pre-tuning mode is needed when no prior information about the process is available. That is, the process and the proper PID control parameters are completely unknown. Therefore, a basic identification routine that indicates what parameters may work, must be employed. In Zeigler-Nichols auto-tuning, the basic idea is to enter a known disturbance to the process; and, based on the system response, the three PID parameters can be estimated. These parameters are the proportional constant k_p, the integral constant t_i and the derivative constant t_d. The discrete time equivalent expression PID control used in this study is shown in equation (2). In equation (2), t is the sample period for the controller.

$$u[k] = k_p \left[e[k] \quad + \quad \frac{t}{t_i} \sum_{i=1}^{n} e[i] \quad + \quad \frac{t_d}{t} \left(e[k] - e[k-1] \right) \right] \tag{2}$$

There are actually two methods for PID parameter estimation presented by Zeigler-Nichols that may be used here. The first technique employs simple proportional control (no integral and derivative terms in equation (1)). Step functions disturbances are input to the process and responses under this proportional control are monitored. The objective is then to find the k_p, or critical gain, that just barely causes the system to become unstable. Instability is indicated when the process response grows (usually exponentially) as time increases. The limit of stability determines a critical gain, k_c and the period of oscillation for the response under these conditions determines a critical frequency, t_c. The parameters k_p, t_i and t_d can then be computed as shown in equation (3). For this implementation a binary search or bisection algorithm is used to "zero in" on the desired critical frequency.

$$k_p = 0.6k_c \qquad t_i = \frac{t_c}{2} \qquad t_d = \frac{t_c}{8} \tag{3}$$

The second Zeigler-Nichols method requires only one test with a unit step disturbance applied and the open loop response monitored. As the response signal increases toward the setpoint, the maximum slope, s, is determined. The point on the time axis where the tangent to this slope intersects is called the deadtime, d. These two values yield PID parameters according to equation (4).

$$k_p = \frac{1.2}{s\,d} \qquad t_i = 2\,d \qquad t_d = \frac{d}{2} \tag{4}$$

This second method is clearly easier to apply and much less disturbing to the plant or process to be controlled. Therefore, this method is usually used for pre-tuning. The first method may still be applied when an exact measure for the critical proportional gain is desired. Although both methods claim to supply this information, testing as outlined in method one, actually graphically illustrates its determination and is therefore more dependable. Due to delays in the actual application of the control law along with noise in the process and controller connections, the critical frequency may only be *reliably estimated* with the bisection search approach used in method one. Nonetheless, both techniques provide an effective means for finding estimates of PID parameters when no prior information is available.

A prototype self-tuning PID controller has been constructed and successfully tested. This controller employs either of the "pre-tuning" methods discussed above to tune itself to a given process. In an effort to establish feasibility first, a simple configuration was used as a starting point. The following paragraph details this configuration and its basic operation.

In the prototype system both the adaptive algorithms and the control law implementation are performed in the PC portion of the system. This was done to establish the potential for true expert system performance before proceeding to more advanced configurations.

The processes and their disturbances are simulated in the PLC. These processes are in discrete state space form and are modeled by

$$\begin{aligned} \vec{x}[k+1] &= \bar{A}\,\vec{x}[k] + \bar{B}\,(u[k] + v[k]) \\ \vec{y}[k] &= \bar{C}\,\vec{x}[k] \end{aligned} \tag{5}$$

In equation (5) $\vec{x}$ is a vector of state variables and $\vec{y}$ is the process output. $\bar{A}$, $\bar{B}$ and $\bar{C}$ are the system matrix, the input matrix and the output matrix of the process. Also, $u[k]$ is the current control signal and $v[k]$ is the current disturbance.

Due to complications beyond the scope of this paper, processes are limited to second order and disturbances are limited to positive and negative step functions. The possible choices of second order processes that may be modeled is limited only by the discrete nature of the process simulation. Process update times less than about 0.02 seconds are not possible. Therefore processes with time constants less than 0.2 (or real poles larger than 5.0) are not considered.

The PC based expert system handles both monitoring and "pre-tuning". Upon start-up the expert system will simply monitor open loop responses.

When a human operator gives the go-ahead the expert system will execute one of the two pre-tuning methods supplying its own unit step disturbances to the process. Once the PID control parameters have been found the expert system enters a dedicated monitoring mode. By once more inputing a unit step disturbance, the expert system can monitor a "characteristic response" for the process while under PID control. Several criteria such as overshoot of the first peak beyond the setpoint and settling time are noted and stored. When monitoring is complete, the system enters the normal PID control mode.

Whenever a unit step disturbance enters the system (from some outside source), the expert system again monitors the response while PID control continues as an intertwined task. If the process has changed, the response will also be different. Very small changes in the response are ignored to avoid constant oscillation between tuning and control modes of operation. When a "significant" change occurs in the process, and hence its response, the expert system automatically invokes a new "pre-tuning" routine and recalculates the PID parameters k_p, t_i and t_d. Thus, the system performs self-tuning PID control and establishes a baseline for the creation of a true expert controller.

3.2 Continuous Tuning in the Expert Controller

Pre-tuning, while ingenious, really involves no requirement for expert behavior. While it may be true that most controllers do not have the ability to test the system and then monitor and interpret the results, the pre-tuning algorithms are still basically numerical analysis. The addition of "continuous tuning" begins to utilize concepts and facilities that qualify the controller as an expert. The objective of continuous tuning is to manipulate the PID control parameters such that the optimal response curve is achieved. It is assumed that some other means such as "pre-tuning" has been used to find the initial PID parameter settings. In this case, optimal is determined by the human operator and/or some generally accepted performance criteria. This specified criteria may be based on measurable characteristics of the process response curve, such as the risetime and settling time.

When a human operator attempts to *manually* tune a PID control for an optimal response curve, he/she trys to find a compromise in controller settings that meets the performance objectives as well as possible. For example, attempts to reduce overshoot (height above setpoint) might result in an increase to settling time. Thus, the human operator finds PID pa-

rameters that yield a response that is "good enough". Such a response is a compromise, where each performance criteria may be only partially satisfied. Heuristic rules that indicate which parameter to change, what direction and by how much, tell the human operator what adjustments to make for a desired result. Years of experience and "rules of thumb," combined with common sense, allow the operator to find a satisfactory response for his/her plant requirements.

The expert controller must mimic this human behavior and tune *itself* in an effort to find the "ideal" process response. The term "continuous tuning" is used because a cycle of monitoring and then adjusting parameters always takes place. If it is indicated that a *best* compromise has been achieved, no adjustments will occur but monitoring still continues. Any change in the process and its response will automatically invoke more tuning. Therefore, the expert controller is constantly in this tuning mode, even if no adjustments are currently required.

The transient curves that result from the application of PID control to various processes generally fall into one of five basic classes [10]: (i) Overdamped with one peak or less; (ii) Overdamped with more than one peak; (iii) Underdamped with only one peak; (iv) Underdamped with more than one peak; and (v) Underdamped with high-frequency oscillations. The fourth type, underdamped with more than one peak, is commonly accepted as the ideal form for PID control responses given that the overshoot, number of peaks, etc. are within the constraints set by plant operators. A typical curve is illustrated in Figure 2.

For the purposes of this study, it is assumed that a curve of this general type can be found by manipulating the parameters properly. That is, any of the other four response curves can be changed to this general form by proper PID parameter adjustments. (The PID parameters determined by Zeigler-Nichols "pre-tuning" are designed to produce curves with these characteristics). To efficiently adjust these parameters, some interactive self-tuning method is needed. Once the proper overall curve is achieved a more sensitive interactive tuning method is needed to find the parameters that produce the response that best meets the criteria set out by the plant operator. A methodology that allows various types to be changed to this desired type, and more importantly, provides the means to optimize the form of this classic response is outlined in the following paragraph.

This tuning method requires two skills generally unique to humans. The first is ***pattern recognition*** and the second is the ability to interpret when a response is ***"satisfactory"***. The expert controller must know what the

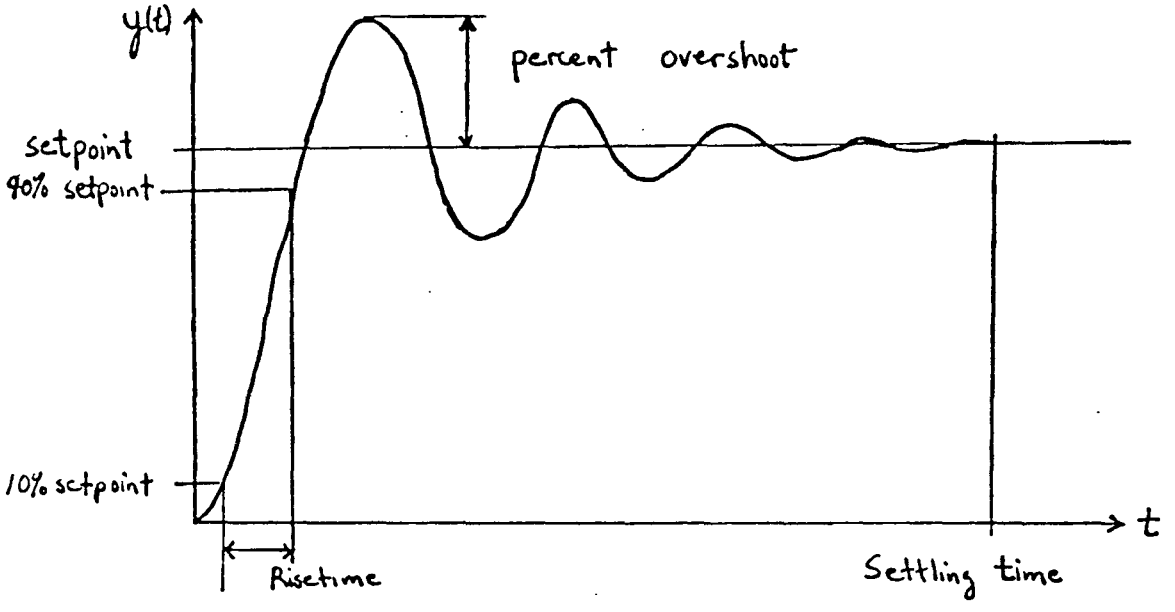


Figure 2: Typical response trajectory.

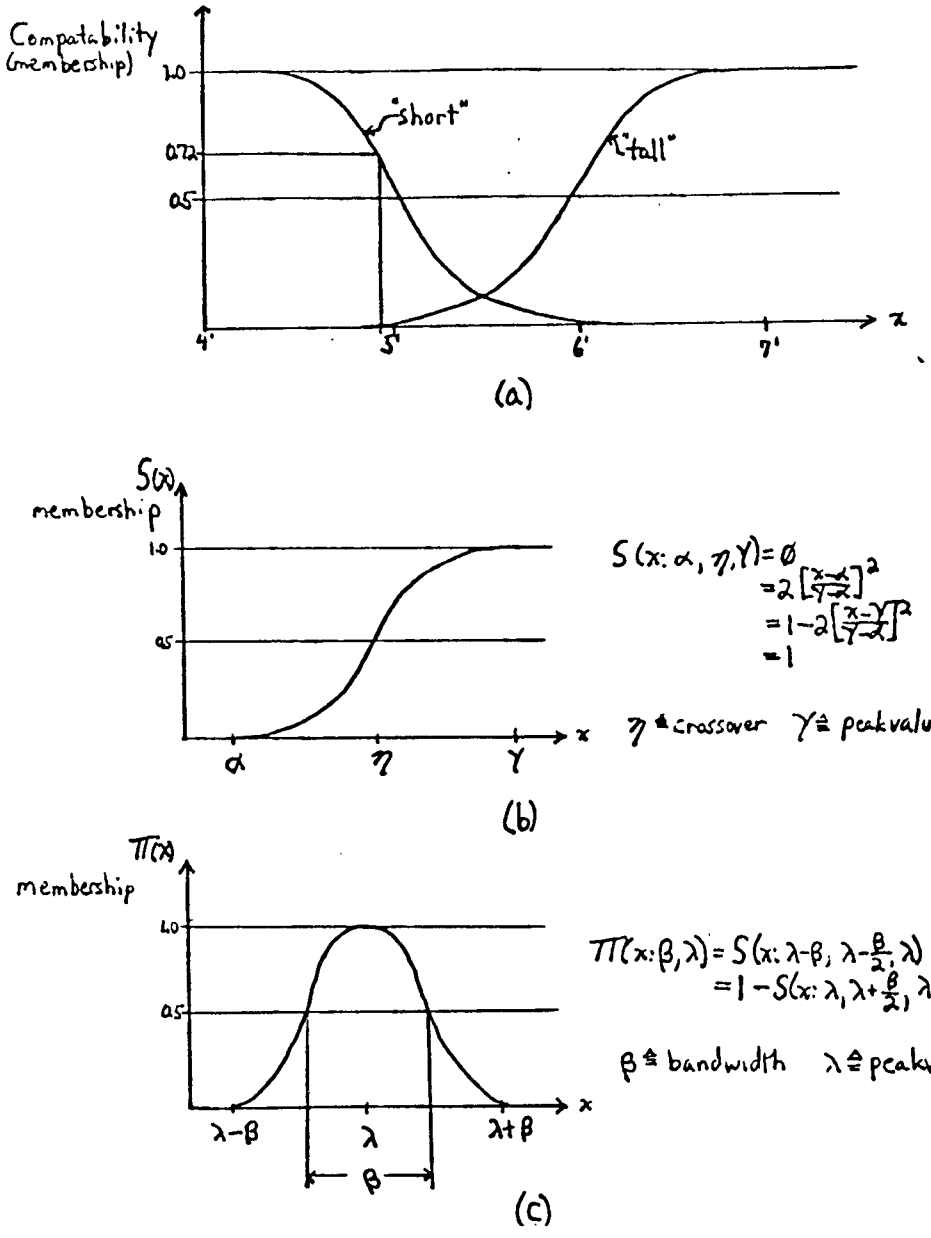
current response curve looks like before it can proceed to make tuning decisions. Fortunately, this rather complex business of pattern recognition, can be simplified to a definition of features for this application. The separate curves can be distinguished in general by the absence or presence of peaks relative to the setpoint. Crucial tuning criteria such as overshoot, risetime and settling time are also available by basic monitoring of the response. By recognizing features and making useful measurements, the expert controller can know enough about a given response to proceed with the business of making decisions based upon this information.

3.3 Logical Specifications of Performance

The determination of whether a given response is "satisfactory" or not, requires a bit more sophistication. One approach may be based on the use of compatibility or membership functions defined in fuzzy logic and set theory [12]. The curves shown in Figure 3 are examples of such membership functions along with their defining equations. In fuzzy set theory, the curve determines the degree of membership that individuals in some universe have to the fuzzy subset in question. For example, if Tim is five feet tall he has a membership value of .72 in the fuzzy subset called "short". Curve (a) in Figure 3 might represent the relative memberships that a given height has in the fuzzy subset "short". For heights less than say, four feet and six inches, the membership value might be unity.

A similar curve may be interpreted to signify the proximity of a given value for a response's settling time to some ideal settling time. Again values below some point receive a rating of 1.0 on this continuous compatibility scale ranging from 0.0 to 1.0. Thus, settling times less than a given value all get optimal ratings. Curve (b) is simply a generalization of curve (a), since each side is based upon the same curve shown in (a). This curve is useful to describe membership to properties that have an optimal value, and membership values must be available for both above and below this value. The other defining parameter of this curve can be thought of as an allowable bandwidth.

A property of PID response curves that may be defined in this manner is the ratio of the second peak overshoot to the first. For a "nice" curve this ratio should be 0.25. However, values both below and above are acceptable within a certain range. Fuzzy membership functions provide an analytical method for assigning a measure just how "good" a given response may be. Each specific criteria such as overshoot, risetime and settling time has its



$$S(x;\alpha,\eta,\gamma) = \begin{cases} 0 & x \le \alpha \\ 2\left[\frac{x-\alpha}{\gamma-\alpha}\right]^2 & \alpha \le x \le \eta \\ 1-2\left[\frac{x-\gamma}{\gamma-\alpha}\right]^2 & \eta \le x \le \gamma \\ 1 & x \ge \gamma \end{cases}$$

$\eta \triangleq$ crossover $\gamma \triangleq$ peak value

$$\pi(x;\beta,\lambda) = \begin{cases} S(x;\lambda-\beta, \lambda-\frac{\beta}{2}, \lambda) & x \le \lambda \\ 1-S(x;\lambda,\lambda+\frac{\beta}{2},\lambda+\beta) & x \ge \lambda \end{cases}$$

$\beta \triangleq$ bandwidth $\lambda \triangleq$ peak value

Figure 3: Compatibility and membership functions.

own membership function with appropriate peak and bandwidth values. A "score" can be defined for a given response as the sum of all the criteria's membership values.

The objective of the adaptive PID control algorithm and the expert controller is to maximize this score. The raising or lowering of different PID parameters will have effects on each of these basic criteria. For example, lowering k_p while raising t_i and t_d might increase overshoot for a given type of response. Such "rules of thumb" use the membership values as an indication about how much to change a particular PID parameter. Some rules will indicate an increase in a given parameter while some will indicate a decrease. The varying degrees of membership and the opposing directions of adjustment combine to yield a compromise adjustment that will hopefully increase the "score" of the next response. These adjustments have the tendency to converge and as the best possible curve, according to the specified criteria, is reached, indicated increases and decreases offset and the resulting adjustments approach zero.

The potential of this technique lies in the fact that membership values for one or more of the criteria may be not be very high but the system has done the "best" it can. That is, it has determined that this response is "satisfactory" even though only some or none of the criteria are very close to their respective optimal values. This emulates the actions of a human operator who manually tunes a PID control loop. A refinement is the use of weighting factors for the different criteria. In a given plant the overshoot may need to be highly restricted, and its excursions should be considered as more important relative to other response characteristics. This weight factor can be achieved by applying exponential factors to the defining membership functions. Squaring a typical membership curve will cause a steeper slope and faster drop-off. Hence, for more important criteria the defining functions are raised to powers greater than one. For less important criteria fractional powers (less than one) are used. Zero weight factor will always yield a unity membership indicating that this criteria is so unimportant that its value is always more than "satisfactory".

Occasionally, basic heuristic rules may come into play also. For instance, when severe conditions such as instability or major process changes are in effect, heuristic rules may take over. Somewhat different tuning rules will apply for each of the five basic response types and there are possible variations within one type. Rules must be included that know how to pick the right tuning rule subset for a given response type. This technique for adaptively tuning PID loops is in its infancy at this time. However, preliminary

simulation studies indicate great promise. There is a great deal of flexibility in this methodology and many extensions are possible. For example, future implementations might use similar fuzzy procedures to even modify the tuning rules themselves.

3.4 Current Implementation

At the time of this writing, various portions of the expert controller are in different stages of completion and the configuration described below is well within reach. For a full diagram of the expert controller see Figure 4. Processes are simulated using a separate personal computer. These processes are in discrete state space form and are modeled by equation (5).

Using a dedicated personal computer provides a very versatile means for process simulation. There is no constraint on the order of the process, and options such as non-linear behavior and noise can be easily added to the simulation. Signals for state space variables $x_1 \ldots\ldots x_n$ are converted to analog signals and made available to the expert controller. The control and disturbance signals, both generated by the PLC, are received through an analog to digital converter and applied to the process. The Texas Instruments PLC is used to implement the control algorithm. For the case of PID control, it samples an analog signal (usually state variable x_1) and computes a control value using equation (2). This control signal is converted back to analog in a zero-order hold fashion and subsequently applied to the process.

Ultimately, the PLC could implement a variety of control algorithms in similar fashion. The ideal arrangement would allow all possible choices of control algorithms to reside in the PLC memory at one time. Then the expert system must simply indicate which is to employed. Also, for the purposes of simulation, the PLC provides a convenient means of simulating disturbances to the process. This disturbance simulation does take time and will effect the overall performance of the controller, but this is not a significant concern at this stage of development.

The analog to digital and digital to analog conversion is required, even though both the process and the control algorithm are digital in nature. Analog signals are used as the common signal for two reasons. First, a direct digital link is more difficult than it seems, since internal number representation and formats may vary. Second, analog signals are more likely to be the mode of communication in "real world" applications; and, to establish credibility and practicality, it makes sense to model the "real world" whenever possible.

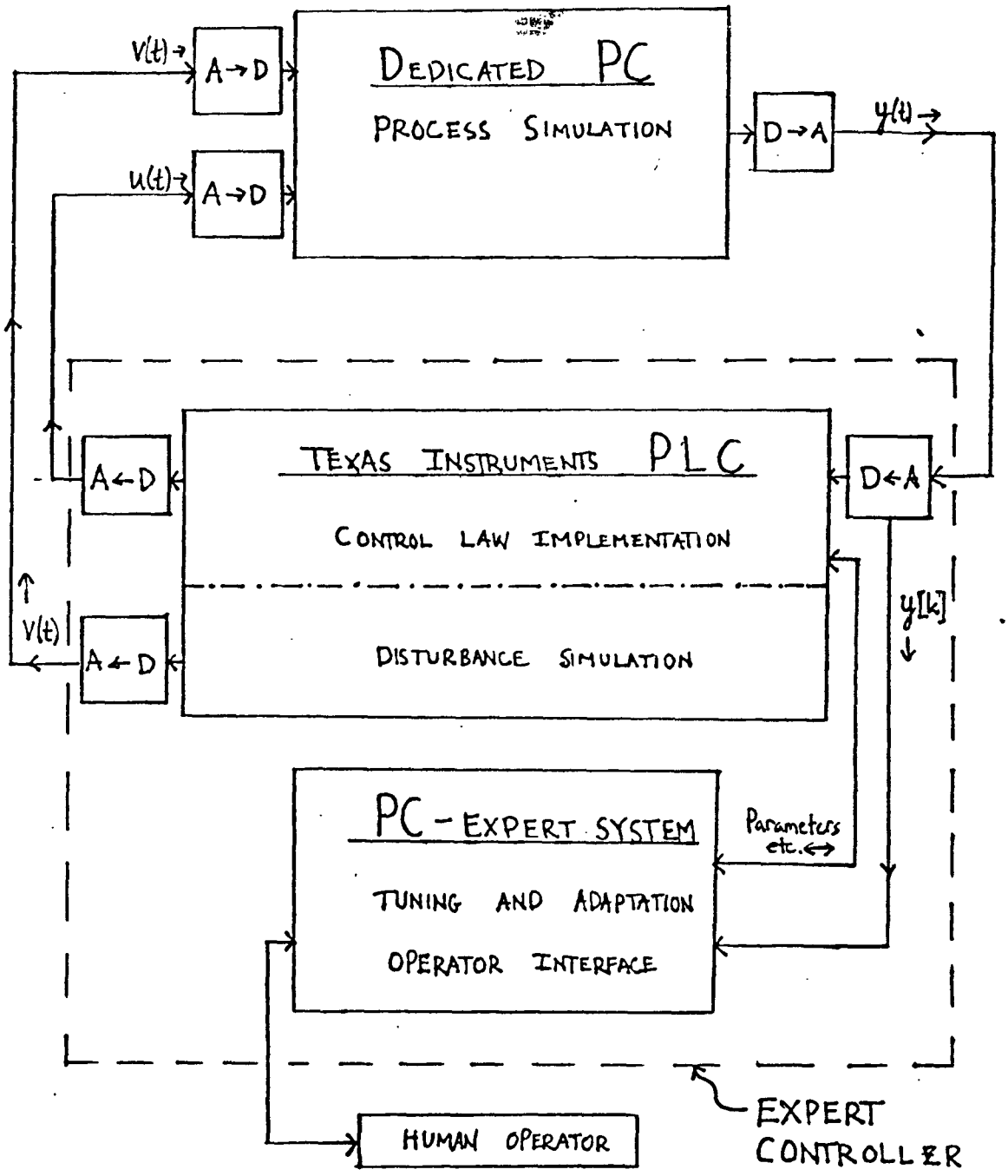


Figure 4: Current configuration of the microprocessor expert controller.

The expert system resides in a personal computer technology designed to interface directly with the PLC. This same PC technology also provides the human user interface. The operation of the expert system can be divided into several modes in the PID adaptive control configuration. In a most trivial mode the expert system waits for a disturbance to effect the system. Upon detection of the disturbance, the expert system enters a dedicated monitoring mode. During this response, all the necessary information for subsequent tuning is noted and saved. Once a complete response has been monitored the expert system leaves the PLC to control the process while it does some "thinking". Using the tuning method outlined above, this "thinking" may not take long, but the times required may not be consistent. Once a new set of parameters have been determined and the new PID constants are stored in the memory of the PLC, the expert system can return to the duty of waiting. Human interaction, unless urgent, would be taken care of during this waiting period.

The programming of the expert system is accomplished with two languages. C is used for monitoring the process responses and all other communications with the PLC. This is primarily due to the fact that the lower level interfaces, provided by Texas Instruments Inc., providing PC/PLC interaction are written in C. The decision support processes are being coded in a dialect of Lisp, called TI-Scheme, also provided by Texas Instruments. The outer shell program is the Lisp portion, which essentially uses subprograms written in C for PLC communications etc. Communication between the languages is handled through sequential disk files. This is a little slow, but it serves well enough for this implementation at its current stage of development.

4 Plans for Further Development

Although the adaptive controller outlined above is *an* expert controller, much still needs to be accomplished to produce *the* expert controller described in Section 2. The current implementation does however, lay the foundation for future enhancements.

The use of a Lisp shell supporting object oriented programming provides the expert controller with an adequate AI tool to produce the kind of complex decision making that will be needed. TI-Scheme also has several features, such as graphics capabilities and processes called "engines" that can be programmed to run for a specified amount of time. These facilities

greatly enhance the interface for the human operator. Other AI languages, such as Prolog, could be used for inference if that becomes desirable. This opportunity is facilitated by the separation of lower level communications by C subprograms from the decision making processes.

The concept of creating a "history of control actions" and its usefulness was discussed in Section 2. The addition of a history file is fairly simple. The expert system simply creates a disk file that is updated after every decision making sequence. The information contained would include: the current control law; the control law parameters; all typical response criteria; membership scores; and other useful information. It might be left to a human operator to decide when a given history file begins and ends (in time), based on some knowledge about changes in the plant. Use of the history file to advantage is more subtle. Some method of determining if the current history file and some previous one match close enough to assume they are products of the same process is required. One systematic method, suggested in Section 3, would be the use of fuzzy membership functions. Also, there is the need to spend a good deal of time creating some standard history files for comparisons to be possible.

The concept of switching control algorithms mentioned in Section 3 is realized to a limited extent in the current implementation of the expert controller. The PLC being used at this point in time does not have enough memory to house several different control algorithms. However, a larger memory is readily available with other Texas Instruments PLCs and this limitation presents no problem up to a point. With enough memory, all the necessary control and identification algorithms could be available at any time. The PC technology involved here also has the ability to actually download a new algorithm from disk to the PLC. (This capability is possible but not actually available at this time.) An intriguing use for this facility might be to download modified or new control algorithms.

Of course, the expert controller is far from the kind of advanced operation required to modify existing algorithms or invent new ones. One potential concern in achieving such a powerful facility is that it may prove too slow for "real world" plants and processes. As for the decision making process that determines when and which control or identification algorithm is applied, much is still undeveloped. However, the potential for the necessary inference is provided by the the ability to program in high level languages such as Lisp and Prolog.

The possible enhancements are unlimited. It would certainly be desirable to extend the operation of the PLC to multiple input and multiple output

processes. Several PLCs might be employed and overseen by the expert system each maintaining one loop of a multi-loop process. There may be special heuristics that could be incorporated to handle unwieldy or non-linear processes. The purpose of this research is to lay the foundation for the production of such powerful systems.

In closing, there is one final point to be made about the topic of expert control. The ultimate key to success lies in identification. The more about a process that is ***known***, the more likely it becomes that optimal control can be achieved. Many techniques exist for process identification but most involve repeated testing of the process. Such testing takes time and often such repeated *disturbances* to the plant or process may not be practical or even feasible. Such "off line" testing becomes completely unreasonable when the process is prone to change. The answer is to build an expert controller that can learn about the process "on line" while maintaining some level of control. From what it learns, an internal model or configuration of the current process can be built. Based on this model and its completeness and accuracy the best possible control can employed. The more the expert controller ***knows*** the better the control will be. Therefore, an expert controller requires the ability to *learn* about a process; and so, process identification is the crucial factor.

Acknowledgements: We would like to thank Don Candy and Fred Weidemeier of TI for their support and cooperation in this project. We would also like to thank Dave Barbe, Pam Harris, and Margi Berbari for their efforts on behalf of the project.

References

[1] K.-E. Arzen, "Experiments with expert control," preprint, 1985.

[2] K.-E. Arzen, "Expert systems for process control," in Applications of Artificial Intelligence in Engineering Problems, Proc. 1st Internat. Conf., Southampton Univ., April 1986, D. Sriram and R. Adey, eds., Springer-Verlag, New York, 1986, pp. 1127-1138.

[3] K.-E. Arzen, "Use of expert systems in closed loop feedback control," Proc. Amer. Control Conf., Seattle, 1986, pp. 140-145.

[4] K.J. Astrom, "A Ziegler-Nicols auto-tuner," Report TFRT-3167, Department of Automatic Control, Lund Institute of Technology, 1982.

[5] K.J. Astrom, "Theory and applications of adaptive control," Automatica, vol. 19(1984), pp. 471-486.

[6] K.J. Astrom, "Auto-tuning, adaptation, and expert control," Proc. Amer. Control Conf., Boston, 1985, pp. 1514-1519.

[7] K.J. Astrom, J.J. Anton, and K.-E. Arzen, "Expert control," Automatica, vol. 22(1986). (See also Proc. IFAC World Congress, Budapest, 1984.)

[8] K.J. Astrom and T. Hagglund, "Automatic tuning of simple regulators with specifications on phase and amplitude margins," Automatica, vol. 20(1984), pp. 645-651.

[9] T. Fortmann and K. Hitz, An Introduction to Linear Control Systems, M. Dekker, New York, 1977.

[10] J. Litt, "An expert system for adaptive PID tuning based on pattern recognition techniques," in Instrumentation in the Chemical and Petroleum Industries, vol. 18, Proc. 1986 Conf., Secaucus, NJ, 1986, pp. 87-104.

[11] R.L. Moore, et al., "Expert control," Proc. Amer Control Conf., Boston, 1985, pp. 885-887.

[12] L. Zadeh, "Calculus of fuzzy restrictions," in Proc. US - Japan Seminar, Berkeley, Academic Press, New York, 1974, pp. 1-39.