April 1976 Technical Memorandum ESL-TM-658

# FINITE-STATE COMPENSATORS FOR PHYSICAL SYSTEMS*

by

T. L. Johnson

## Abstract

The purpose of this memorandum is to explore conceptually some problems of direct design of optimum automata-based compensators for continuous-time physical systems. Important clues to the design of such finite-state compensators are to be found in control, information, and automata theory, numerical analysis and computer architecture; but considerable research effort will be required to unify the relevant aspects of these fields. The complexity of the theoretical development, however, is ultimately balanced by some substantial practical simplification, and potential performance improvement over existing methods.

* Investigation of this topic at the M.I.T. Electronic Systems Laboratory has been supported by the Air Force Office of Scientific Research under Grant AFOSR 72-2273. The ideas described herein were first presented on February 11, 1976, while the author was visiting the Laboratoire d'Automatique et d'Analyse des Systemes (Toulouse, France) with support provided by the Centre National de la Recherche Scientifique (CNRS) of France.

Electronic Systems Laboratory
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

## I. Introduction

The main theme of this memorandum is to suggest the advantages inherent in the direct design of finite-state compensators. To facilitate the description of a "finite-state compensator", assume that it is required to regulate a smooth continuous-time finite-dimensional dynamical system [1], $\Sigma = (U, \mathcal{U}, Y, \mathcal{Y}, X, \phi, r)$,[1] from some initial state $x_0 \in X$ at $t = 0$, to the zero state at $t = \infty$.[2] A finite-state compensator realizes a causal mapping $F: \mathcal{Y} \to \mathcal{U}$ as the composition of three maps:

(1) A coder, being a causal operation $C: \mathcal{Y} \to \mathcal{U}_c$, where $\mathcal{U}_c$ is a space of infinite sequences $\{t_k, \underline{y}_k\}$, $k = 0,1\ldots$ with $\{t_k\}$ a monotone-increasing sequence of real numbers, and $\{\underline{y}_k\}$ a sequence of (finite-length) vectors whose elements take values in finite sets,

(2) A finite-state sequential machine [2], $M = (U_c, \mathcal{U}_c, Y_c, \mathcal{Y}_c, Q, \delta, \lambda)$, where $U_c$ consists of pairs $(t_k, \underline{y}_k)$ as described above, $Y_c$ consists of pairs $(\sigma_k, \underline{u}_k)$ of the same sort, $\mathcal{Y}_c$ is the space of sequences of such objects, and $Q$, $\delta$, and $\lambda$ denote state-set, state transition, and readout mappings, respectively, which realize a causal mapping $M: \mathcal{U}_c \to \mathcal{Y}_c$[3], and

---

1. Most of the concepts to be discussed will apply equally to infinite-dimensional (e.g., time-delay, distributed) plants, and with appropriate modification, to large-scale, interconnected, and finite-state plants having finite numbers of inputs and outputs.

2. $\underline{x}_0$ is, for the present, taken to represent the effects of disturbance. Stochastic considerations will be discussed later.

3. The inclusion of "transition times" $(t_k)$ necessitates a slight modification of the usual description of FSSM's. It should be apparent how to proceed to define a "causal" mapping between $\mathcal{U}_c$ and $\mathcal{Y}_c$.

(3) A <u>decoder</u>, being a causal operation $\mathcal{D}: \mathcal{Y}_c \to \mathcal{U}$

so that formally $\mathcal{F} = \mathcal{D}\ \mathcal{M}\ \mathcal{C}$. Under appropriate assumptions the feedback interconnection of $\Sigma$ and M, as shown in Figure 1 may be shown to be a well-defined continuous-time dynamical system (not necessarily smooth and not necessarily finite-dimensional, however). In general, "design" of a finite-state compensator consists of the determination (subject to appropriate practical constraints) of the mappings $\mathcal{C}$, $\mathcal{M}$, and $\mathcal{D}$.

The proposed structure of Figure 1 is of course not "new": Among control engineers, the physical implementation, consisting of an analog-to-digital converter ($\mathcal{C}$), followed by a small computer ($\mathcal{M}$), followed by a digital-to-analog converter ($\mathcal{D}$) has been widely studied for more than twenty years (e.g., [3], [4]). While a complete review of this literature could be more overwhelming than informative, it is useful to identify, in a few brief sketches, how some of the current methodologies of compensator design might be implemented with such a structure.[4] The questions to be explored in the sequel elucidate many new issues in finite-state compensation which arise often from the shortcomings of existing methodology. While many of the current methods yield performance which is good or at least adequate in terms of their respective design criteria, their digital implementation may be at best inefficient in terms of memory requirements, and at worst, may achieve only a fraction of the performance that the hardware is capable of attaining.

---

4. Our classification of available methodologies is largely one of convenience. The references cited are in no way complete, but are rather illustrations of "typical" cases.

Industrial control (and automation) [5] is by almost any economic measure, the most important area of feedback control design, and yet it has been all but ignored by the control theorists. And in turn, many industrial control systems have been designed with a great deal of engineering intuition and "common sense", but very little control theory. Probably the great majority of such systems rely heavily on mechanical linkages and relay networks, and represent the most elementary example of a finite-state controller. In the last decade advances in design, it would appear, have been primarily motivated by innovations in computer and solid-state technology. Although design principles do exist, this author is not aware of any very general underlying theory which characterizes industrial controllers currently being produced. Historically, such control systems, have made very effective use of limited hardware in each particular application. In terms of hardware and software, such controllers tend to be qualitatively quite different from those produced by the following "schools".

Another group of methods might be loosely described as constrained-structure compensation methods. The structure of the feedback controller is fixed, up to the selection of a limited number of parameters, and the performance of the resulting closed-loop system is evaluated as a function of the parameter values. The design parameters are then fixed (or adaptively adjusted) to yield the most desirable over-all performance. Such methods have been widely applied for nonlinear systems; they have the advantage that if the controller structure is well-chosen, then the free parameters may be uniquely determined, and the controller is implementable. Traditional gain-setting methods [6] are based on this idea, as are more

recent approaches typified by the linear multivariable compensators of Levine, Johnson and Athans [7], Sandell and Athans [8], and the moving-average representation used by Astrom and his co-workers [9]. The latter representation is particularly attractive for finite-state implementation, provided that large inputs and outputs can be adequately treated (e.g., in the presence of saturation). A reasonable criticism of such methods is that the resulting performance of the compensator may depend more critically on the engineer's intuition in choosing the appropriate structure, than on the actual parameter determination. Where finite-state implementation is anticipated, one either uses a discrete-time model of the system itself and designs a discrete-time compensator or else approximates the continuous-time compensator directly.

The geometric methods take the dynamical structure of the systems to be controlled as the most important determinant of compensator structure, and view the determination of parameters as a secondary issue. These methods have been employed by Wonham [10], Rosenbrock [11], Brockett [12], Wolovich [13], and many others. The role of controllability, observability, and stability in decoupling and pole-placement has been fully worked out for linear systems in [10], [11], [13], and some of the essential notions have also been defined for linear, finite-state machines [1], [2], but the particular structure of Figure 1 seems not to have been fully investigated. These geometric methods are conceptually quite appealing and are very general, but they can be quite difficult to apply in practical situations. The presence of noise or nonlinearities drastically increases the complexity of the theoretical solutions, and linearized or approximate models may fail to preserve qualitative properties of the nonlinear dynamics, for instance.

A second (but not fundamental) shortcoming of much of the work in the engineering literature is that it tends to rely heavily on ordinary calculus of real valued variables without considering more general topologies or operations, with the consequence that existing designs require great care in digital implementation (viz., any approximation method should preserve the relevant geometric properties).

A final group of methods are those based on optimal control [14] and stochastic control [15], [16] with which we presume the reader to be familiar. Conceptually, the result of dynamic programming methods may be represented in the form of a feedback compensator. However, these methods require solutions of an infinite-dimensional problem which is not readily approximated. The optimal control law may be difficult to implement digitally, and the result may be sensitive to parameter variations or round-off error [3]. Considerable care must be exercised in the choice of a performance index, since the resulting controller may have to satisfy multiple objectives and/or constraints, and since the optimal controller may otherwise fail to take advantage of structural properties of the plant.

Most attempts to digitally implement dynamic compensators based on the constrained structure, geometric or optimal-control methodologies have followed a procedure of developing a discrete-time (recursion equation) version of the problem, realizing the compensator as a discrete-time dynamical system, and then dealing with problems of simplification, saturation, drift, etc. in an ad hoc manner as they occur. Typically, coding and decoding of the continuous-time inputs and outputs of the physical plant is accomplished by a A/D or D/A converter. The synchronization and sampling

(1) Curse of dimensionality: For systems of even moderate complexity, the design of compensators will require large amounts of off-line computation, and in some cases also on-line computation. This may involve, for instance, gradient algorithms, polynomial factorizations, or simply a large number of linear operations. The problem is perhaps most acute in control of stochastic and/or distributed systems. Usually, exact solutions are not computable in finite time (e.g., the space being optimized over is infinite-dimensional). This might be overlooked in the face of increasing speed and decreasing cost of computation; but the development of general computational algorithms is certainly an obstacle when the system to be designed is small, and the loss of engineering insight about the computational process itself is disturbing.

(2) Nonlinearities: When the plant experiences significant nonlinearities under normal operating conditions the design of control laws is significantly complicated. Though recent progress has been made in the design of nonlinear controllers for multivariable systems, our understanding of the qualitative features of nonlinear control laws is quite inadequate [12], and large computational requirements abound. On the other hand, there exist many practical nonlinear systems where even an ad hoc nonlinear control law is vastly superior to the best linear control law.

(3) Disturbances and Stochastic Effects: A great deal of disagreement is evident concerning the issue of how to represent disturbances and stochastic effects when designing control strategies, and also concerning the strong effect of information structure on the procedure for designing controls. In many practical applications, the information about the system which is

problems are left to the interrupt-handling software of the machine. Storage and word-length limitations are ignored unless they lead to difficulties.

To any person who has programmed a small-scale computer, it is apparent that this methodology makes ineffective use of the capabilities of most common machines. Even accepting the "architecture" of existing machines, it is clear that (for instance):

- a whole range of shifting and indirect addressing instructions is not very much used
- thoughtless programming will probably fail to utilize recursion or push-down capabilities of the machine
- the programming language is probably much more versatile than required by this sort of application
- the CPU spends most of its time handling I/O interrupts and memory access instructions
- overflow and underflow are likely to occur, but have not been anticipated
- round-off errors are likely to accumulate
- memory requirements for the program may be larger than expected.

These are merely examples of a mismatch between machine capabilities and the algorithms it is required to execute. In some cases too much, and in other cases, too little is expected of the machine.

Consider also certain common difficulties encountered in the computation which must precede application of these approaches to practical design problems:

required for "acceptable" control is far less than knowledge of say, the conditional density of the state given all past information. Often, the most important random effects occur at relatively low frequencies, and may be approximately represented as "disturbances". Unfortunately, the "best" control law structure using current methods often appears to depend crucially upon an accurate characterization of the effects of such disturbances. Linear-gaussian approximations often yield poor performance in the face of such disturbances. Performance evaluation in practice is often based on sample-path characteristics rather than the mathematical expectation used in the theoretical analysis. No natural and general way to incorporate discrete-valued random signals in compensator design has been proposed.

A natural alternative to these substantial difficulties is to work directly with the digital compensator as a finite-state sequential machine. For a given machine, a given machine language and a given interface with the plant, this involves optimizing the program and data directly. More generally, it involves design of the coder, decoder, and machine architecture as well. These possibilities are discussed more fully in the succeeding sections, and will be seen to involve some quite different issues from those traditionally considered. These issues might appropriately be viewed as extensions of the traditional design considerations, though, because any of the above methodologies may be applied[6] in the context of finite-state compensation.

---

6. We return to a discussion of what might be required in the concluding section.

## II. Coding and Decoding

The celebrated example of Witsenhausen [15] illustrates the crucial role which coding and decoding may play in stochastic control problems, i.e. in a situation where useful information may be coded into bits of a word which are insignificant for numerical computations. It might be further suggested, in fact, that this represents a generic situation which might be encountered in all sorts of infinite-dimensional optimization problems where the results are not insensitive to the chosen topology. The real numbers have infinite resolution. Engineers have usually chosen to avoid such problems - many times with considerably difficulty - by reformulating design problems so that such singular situations do not occur. Quantities in an actual machine computation, of course, will be represented by the contents of binary registers, which must be drawn from a finite set of alternatives (assuming a finite register). In this context, the sorts of situations above reduce to the consideration of a finite (albeit often very large) set of possible solutions: the question now concerns how best to allocate the available bits. This is a coding problem.

Before discussing in greater detail relevant aspects of coding and decoding, one might take note of another implication of Witsenhausen's example, viz., the distinction of "data flow" from "control" or "information" signals. Loosely, "data flow" concerns numerical transformations on signals "passing through" a digital processor, whereas "control" or "information" signals (the terms being used differently than in the control literature) relate to the sequencing and types of operation which are carried out by the processor hardware (possibly in response to external commands or

interrupts). This distinction has been widely exploited in computer architecture [17], though the idea appears rather new to control theory. The control engineer is quite accustomed to thinking in terms of data flow, e.g. the "analog" operations performed on measurements to obtain return signals; but he often appears to take the "control" structure for granted (it is thought to be "enough" to specify an algorithm without concern for how it is to be implemented). However, "control" and "data" paths are often conceived separately by the computer-designer (for good reasons) - so that the coder/decoder in Figure 1 must provide "inputs" for both of these systems. The problem of designing data paths is well-studied. The less-studied "control" or "information" signals might consist of a number of bits in the output register[7] of the coder, along with the sequence of times $t_k$ at which transitions in all (or part) of the contents change, and current theories do not seem to provide an adequate treatment of the processing of such signals. It is worthwhile to remark that signals in industrial control systems are often generated directly in discrete form (e.g., triggering a limit switch) and thus signals for the control structure are generated directly. Many commercial industrial control computers in fact possess an elaborate "control structure" and very limited "data structure".

Herein, the coder is viewed as if it were a memoryless device; in fact, it might use memory in a very crucial fashion, but such capabilities may always be absorbed in the description of the FSSM for conceptual purposes. The design of the coder/decoder appears to be one of the more complex issues

---

7. For it can be viewed this way.

of finite-state compensator design, because it brings together several complicated problems:

(a) Transmitting the essential control information. Subject to the constraints on sensor choice and actuator deployment, the general operation of the compensator is to produce control signals by monitoring, identifying and processing that available information most relevant to closed-loop performance of the plant (however this is defined). The coder reduces a signal with (theoretically) infinite resolution (continuous time-function) to one with finite resolution. A theory is needed which concisely defines that finite set of numbers which in an appropriate sense, preserves the maximum freedom to control the plant behavior, using the class of operations realizable by a FSSM. This has been tentatively termed "essential control information". Some notion of entropy appears relevant here, but a precise definition is still premature.

(b) Coding alphabet. A code is likely to place a constraint on the language used by the processor. If finite-length codes are used, great economies might in general be achieved by having the processor language operate directly with words of the code, as opposed, say, to using standard arithmetic operations. A control law might in the simplest case be generated by table look-up, which given an input codeword (generated from a sensor signal) would search for a corresponding output codeword and transmit it to the decoder. Evidently in a control scheme of any complexity the tradeoffs between accurate coding and economical machine language become extremely difficult to evaluate. But a considerable theoretical literature on programming languages exists, and certain fundamental principles have been identified. The issues of redundancy, reliability, fault tolerance, and resolution are also significant. These problems have been extensively studied in coding theory, but apparently open questions remain even in the deterministic case. Longer prefix codes may purchase noise immunity at the expense of wasting processor time and memory in the decoding operations themselves, etc.

(c) Choice of "levels". The "front end" of a coder might perform various sorts of signal conditioning, but at some stage the contents of the coder output register will change abruptly as an analog variable crosses some level - continuous-time inputs being assumed. It is evident, in general, that such transitions will occur asynchronously in time, and depending on the value of the analog signal. Apparently, rather general devices can be characterized in this manner, e.g., the transition times of the coder input stage depend on the state value of the continuous-time plant. Traditionally the control engineer may have tended to regard this type of elementary nonlinearity as a nuisance; in fact, it could turn out to be one of the key features of coder design for such compensation problems. As noted below, finite-state simulations of dynamical systems are (from a theoretical viewpoint) constrained to be very poor, and the class of functions computable by a finite-state compensator are also rather limited in a sense. There is, however, one

significant theoretical freedom available to the designer of finite-state compensators: the creative use of the variable time-delay. A feedback system with time-delay in the loop is infinite-dimensional (even if the feedback operation is finite-state), and hence the potential class of closed-loop behaviors obtainable with finite-state compensation is much more broad than one might expect. How to design finite-state control laws which realize this potential is presently an open question.

## III. Finite-State Machines for Compensation

In the long term one can foresee a clear need for the study of architecture and language of control computers. Certain structural features of these small machines would be common to many applications (e.g., hardware for matrix computations is frequently noded), while languages might be more tailored to particular applications; these requirements appear favorable for the manufacture of medium-size lots of microprocessors for control applications, insofar as "languages" of dedicated machines can be effectively hardwired. Control applications do seem to require certain machine features which are not yet well-understood, however:

(a) Real-time requirements: One has the impression that computer scientists are exasperated by the necessity for input-output devices. The spectre of a high-speed machine with efficient software being eternally interrupted by some plodding mechanical gadget is admittedly disturbing, and it is probably most repulsive in an on-line control application (when some control engineer is trying to make the machine "simulate a discrete-time dynamical system"). Rather than having to tailor its operations to some external clock, an asynchronous control law implementation might be more efficient. In this scheme, the coder signals its transition times and/or transitions to the CPU, but the data is actually processed more or less at the convenience of the CPU, rather than having interrupts generated by the I/O device (this is a sort of "check flag" situation). The CPU processes various inputs according to a priority scheme, and changes the outputs immediately on completion of a computation (or after a programmed delay). In real-time applications, the constraint of using a synchronous control law is primarily an analytical convenience to the designer; there is otherwise no reason why control laws should have synchronous implementations and there are

good reasons for the opposite scheme to be adopted. After all, many industrial controllers operate asynchronously, and human operators certainly do. There is also the previously noted potential advantage that the control law can be made "infinite-dimensional" in this way.

(b) Internal Model Principle: Whether finite-state compensation should obey any "internal model principle" is not clear. One can envision that an internal model would be useful in asynchronous control, viz., internally, the CPU might compute controls based on the model performance (which might even run in advance of real-time), relying on the actual plant outputs to correct the model predictions only at "convenient" times. If any internal finite-state model is to be constructed, its performance can only represent the continuous-time plant to a very restricted extent. This aggregation problem was suggested many years ago by Bellman [18], and subsequently studied by Wang [19], but only in the case of a synchronous discrete-time model.

(c) Limitations on Compensator Performance: Since the early deliberations of Turing, it has been recognized that the class of functions which automata can compute is limited. Under appropriate assumptions [2], one can establish that the input-output mapping for a Turing machine (possessing infinite memory, however) is described in terms of the class of partially recursive functions. This result of computability theory gives insight as to the nature of the limitations to be expected, but would have to be modified in the present context to account for (i) machine language and architecture, (ii) finite memory size, and (iii) real-time (causality) requirements. The problem of precisely defining this class of realizable compensation maps ($\mathcal{M}$) in any general way appears very difficult, and any progress would be significant. Approximation methods ([19], [20]) might yield some insights.

Another limitation concerns the handling of variables which go "out of range". If one considers any monotone function from a real variable into a finite ordered set, whose range contains the maximum and minimum elements of the set $(z_a, z_b)$, there will be maximal intervals of the form $(-\infty, a)$, $(b, \infty)$ which map identically into $z_a$ and $z_b$, respectively. If the coder, then, is of the A/D conversion type, there will be such ranges of motion of the system where the coder output remains constant. Unless the subsequent (open loop) action of the finite-state compensator under these conditions is such as to return the variable in question to the range [a, b], then the hypothesized regulation objective may not be readily achieved. Clearly, the consideration of such overflow (or possibly failure) modes must constitute an integral part of the design of a finite-state compensator - in the present example this might involve in part a knowledge of global stability properties of the plant itself. The behavior of more general coders and decoders can be reduced to that in the present example.

Evidently, the linear machine and linear control law do not have a natural role in finite-state compensation. The algebraic structure of control laws could be fundamentally different, as could the operations.

(d) Memory Requirements: The complexity of performance of which a finite-state sequential machine is capable is a consequence of its structure and the available memory. Economical use of memory is often achieved by having core shared between one or more programs (determining a "control" sequence for the machine) and the data set on which they operate. For a fixed memory size, this introduces in the finite-state compensator a fundamental trade-off between the complexity of the control algorithm and the amount of input data stored in order to characterize plant performance, decode information, or represent performance requirements, etc. From this perspective the significance of coding alphabet and programming language is readily comprehended.

## IV. Discussion

A probable reason that the results of control theory have not found wider acceptance to industrial control applications is that any theoretical design of sufficient complexity as to require digital implementation faces a myriad of deployment problems. Control theorists too often vaguely dismiss such difficulties as practical hardware problems peculiar to the application in question. They have, with few exceptions, clung to the methods of ordinary calculus and continuous variables in an era when the resulting controller designs are usually impossible to implement exactly, or are uneconomical. The rigor invested in results of this sort is not wasted, but in today's technology it is often misdirected.

The theme of this "white paper" concerns merely digital compensation; the inescapable conclusion (to this author) is that the vast bulk of literature on the topic is currently devoted to issues which are of secondary importance. The direct design of finite state controllers has many potential advantages including a natural resolution

of most implementation problems:

- model aggregation standards
- interface design
- saturation, fault-handling
- inclusion of finite state inputs and outputs of the plant
- alphabet and programming language
- storage allocation
- real-time constraints, interrupt-handling
- guaranteed existence of solutions to common optimization problems, and inclusion of more general constraints
- natural representations of nonlinear control laws which are computable.

In addition, constraints of information structure or decentralized control, or interconnected systems, are naturally incorporated in this framework.

These objectives indeed appear worthwhile, but a concise and intuitive resolution of the design problem, if it exists, will almost surely - in view of the preceding discussion - require some basic results from several subject areas. Hopefully, the major function of this informal presentation has been to indicate which topics require further investigation, and what might be the general objectives of such research.

One might well protest that the problems appear too complex, or too general, or too inter-related to produce a usuable result in the end. This judgement may be premature: certainly results of

admirable clarity and generality have emerged during the last decade in many of the fields cited.

The subject of performance criteria for finite-state controllers concerns the over-all system and hence has not been previously discussed. Conventional criteria such as control energy expenditure and output tracking error are still perfectly valid performance measures, but would need perhaps to be appropriately augmented in cases where the plant itself had discrete-valued inputs or outputs. Compensator structure, it seems, is best viewed as a hard constraint on the closed-loop input-output behavior which can be realized, i.e., one should not apply any sort of "soft" penalties to the FSSM unless essential.

## Acknowledgement

The ideas put forward in this brief memorandum inevitably have occurred to me, in part, through association with other members of the Decision, Control, and Communications groups at M.I.T., and in particular Professors Willsky, Sandell, Gallager and Athans. Though the purpose of the note is primarily to associate some related ideas concerning compensator design in an informal way, my apology is offered in advance for lack of rigor, or clarity, and for abuse of technical jargon.

[1] Kalman, R.E., P.L. Falb and M. Arbib, Topics in Mathematical System Theory, McGraw-Hill, New York, 1969.

[2] Bobrow, L.S. and M. Arbib, Discrete Mathematics, W.B. Saunders Co., Philadèlphia (1974).

[3] Curry, R.E., Estimation and Control with Quantized Measurements, M.I.T. Press, Cambridge, Mass. (1970).

[4] Tou, J.T., Optimum Design of Digital Control Systems, Academic Press, New York, 1963 (and subsequent editions).

[5] Automation Research Council Reports, 1974.

[6] Shiners, S., Control System Design, J. Wiley and Sons, New York, 1964.

[7] Levine, W.S., Johnson, T.L. and Athans, M., "Optimal Limited State Variable Feedback Controllers for Linear Systems", IEEE Trans. Auto. Control, Vol. AC-16, No. 6, pp. 785-793 (1971).

[8] Sandell, N.R. and M. Athans, "On 'Type-L' Multivariable Linear Systems", Automatica, Vol. 9, pp. 131-136 (1973).

[9] Astrom, K.J., Introduction to Stochastic Control Theory, Academic Press, New York.

[10] Wonham, W.M. and Morse, A.S., "Decoupling and Pole Assignment in Linear Multivariable Systems: A Geometric Approach", SIAM J. Control, Vol. 8, No. 1 (1970).

[11] Rosenbrock, H.H., State Space and Multivariable Theory, Nelson & Sons, Ltd., London, 1970.

[12] Brockett, R.W., "Volterra Series and Geometric Control Theory", Proc. IFAC'75, Boston, Mass. Part IB, Paper 36.1.

[13] Wolovitch, W.A., "Multivariable System Synthesis with Step Disturbance Rejection", IEEE Trans. Auto Control, Vol. AC-19, No. 2, pp. 127-130 (1974).

[14] Athans, M., "The Role and Use of the Stochastic Linear-Quadratic Guassian Problem in Control System Design", IEEE Trans. Auto Contr., Vol. AC-16, No. 6, pp. 529-551 (1971).

[15] Witsenhausen, H., "A Counterexample in Stochastic Optimal Control", SIAM J. Control, Vol. 6, 1968.

[16] Duncan, T.E. and Varaiya, P.P., "On the Solutions of Stochastic Control Problems", SIAM J. Control Vol. 9, No. 3, pp. 354-371 (1971).

[17] Allen, J. and R. Gallager, Computation Structures, M.I.T. Course Notes (6.032), (to be published).

[18] Bellman, R.E., Proc. Symp. Appl. Math., 1960, p. 217.

[19] Wang, P.K.C., "A Method for Approxiamting Dynamical Processes by Finite-State Systems", Int. J. Control, Vol. 8, No. 3, p. 285-296 (1968).

[20] Kaliski, M.E. and Klein, Q.L., "Sequence Generation by Autonomous Discrete-Time Systems: Further Results", Proc. 13th Allerton Conference on Circuit and System Theory, 1975 (Monticello, Ill.)