Automated Physical Modeling

Andrew Gelsey*
Computer Science Department, Yale University
P.O. Box 2158, Yale Station
New Haven, Connecticut 06520-2158

Abstract

The creation of abstract models of physical systems is an important AI research area. I describe a program which can automatically construct such models for machines like mechanical clocks or watches. The program finds an appropriate set of state variables and determines how they change as time passes. The abstract model of the mechanical device may be used to numerically simulate its behavior. My program uses short, controlled simulations to identify repet itive behavior patterns which can be used for long-term behavior prediction.

1 Introduction

Reasoning about physical systems generally involves the creation and manipulation of abstract models, normally constructed by the person studying the system. 1 will discuss the automatic generation of such models, as well as their manipulation and analysis. The physical systems I will focus on are mechanical devices.

An abstract model of a physical system is usually based on a set of *state variables.* A set of particular values for the state variables represents a particular state of the system. The model must also include a description how the state variables are related and how they change.

The values taken by state variables may be either numerical quantities or qualitative symbols. The abstract models used in engineering and the physical sciences normally use state variables which take numerical values. Much of the work by artificial intelligence researchers on reasoning about physical systems has focused on models in which state variables take only qualitative values. See, e.g., [Bobrow, 1985].

Qualitative models of physical systems are often useful for reasoning in situations where the information available about the system is limited or imprecise. However, many physical systems cannot be adequately represented by qualitative models. Geometry is especially difficult to deal with qualitatively. Geometry plays a central role in the behavior of mechanical devices, so they cannot be described by purely qualitative models, although work has been done with mixed quantitative/qualitative models for such machines. [Forbus *ct al.*, 1987, Faltings, 1987, Joskowicz, 1988, Nielsen, 1988]

In this paper I will focus on models with numerical state variables. It is often assumed that such quantitative models are not relevant to artificial intelligence because

1. We already know how to use quantitative models. Open problems are highly technical and of interest only to mathematicians.
2. The analysis of quantitative models doesn't yield results at the level needed for artificial intelligence.
3. People reason about physical situations qualitatively.

In fact, quantitative models are quite worthy of consideration by AI researchers. Let us consider the objections listed above.

1. There are many open AI questions about quantitative models. For example, "Where do they come from?", which is the question this paper addresses. While mathematicians and physical scientists have accumulated a wide variety of techniques for systematically analyzing such models, model creation has received little attention, probably because the techniques required are the very ones that AI specializes in. Model building is a very human activity, and involves questions like "What factors shall I neglect?", which don't have the sort of absolutely correct answer favored by the systematic, techniques of the mathematical sciences.
2. In Section 7 I describe the algorithm used by my program to find a concise summary of a machine's repetitive behavior patterns. These summaries can then be used to predict the machine's long-term behavier. Thus the use of quantitative models and numerical simulations does not prevent the formation of concise, qualitative output of the sort one would expect from an AI program.
3. The fact that people can talk about physical situations without using numbers does not justify the conclusion that their internal reasoning methods are purely qualitative. If there is one thing AI research has taught us, it is that there is much more to human thinking than what we perceive on the surface.

*This work was supported by the NSF through grant IRI-8610241. Some earlier portions were supported by ITT.

Spatial and geometric reasoning, in particular, unquestionably involve considerable information with very number-like qualities.

2 Input to the model generator

The only acceptable input for *fully* automated physical modeling is the actual physical system being modeled. However, dealing with this sort of input would require me to address difficult problems in robotic perception which I would rather ignore at present. Instead, my model generator takes as input a representation of the physical structure of the system to be modeled.

The representation used[1] is Constructive Solid Geometry [Requicha, 1980]. In the CSG representation, each part in a machine is represented as a closed subset of three-dimensional Euclidean space which is formed by applying the Boolean set operations of union, intersection, and difference to a small set of primitive solids. For example, a square plate with a hole in it might be represented as the difference of block and a cylinder, where the block would be appropriately sized and positioned to represent the plate, and the cylinder would have the correct diameter and position so that the difference operation would create the desired hole in the plate. The actual representation is a binary tree whose internal nodes are set operations and rigid motions and whose leaves are primitive solids.

This geometric representation is supplemented with information about non-geometric physical properties such as masses, spring constants, and coefficients of friction.

3 Identification of State Variables

There is no single correct set of state variables for a physical system. Different choices of state variables are suitable for different purposes. One choice of state variables for a mechanical device is the position and velocity of every atom of the device. Another choice is the position and velocity of every part of the mechanism, where a part is a connected solid object. In a mechanical device, geometric constraints on the relative positions of parts in the device often allow the use of very few state variables to specify the state of a device with many parts. For example, for any position of the crankshaft in Figure 1, only one position is possible for each of the other parts. Thus the state of the device can be represented with only two state variables: the position and velocity of the crankshaft. The identification of these geometric constraints on the relative positions of parts is called *kinematic analysis*.

Kinematic analysis is the first step my program does in identifying state variables for mechanical devices. The algorithms used are described in detail in [Gelsey, 1987, Gelsey and McDermott, 1988, Gelsey, 1989]. Only an outline of the methods will be given here.

Kinematic analysis starts with the identification of *kinematic pairs*, pairs of parts which mutually constrain

[1] I use the PADL-2 solid modeling system developed by the Production Automation Project at the University of Rochester.

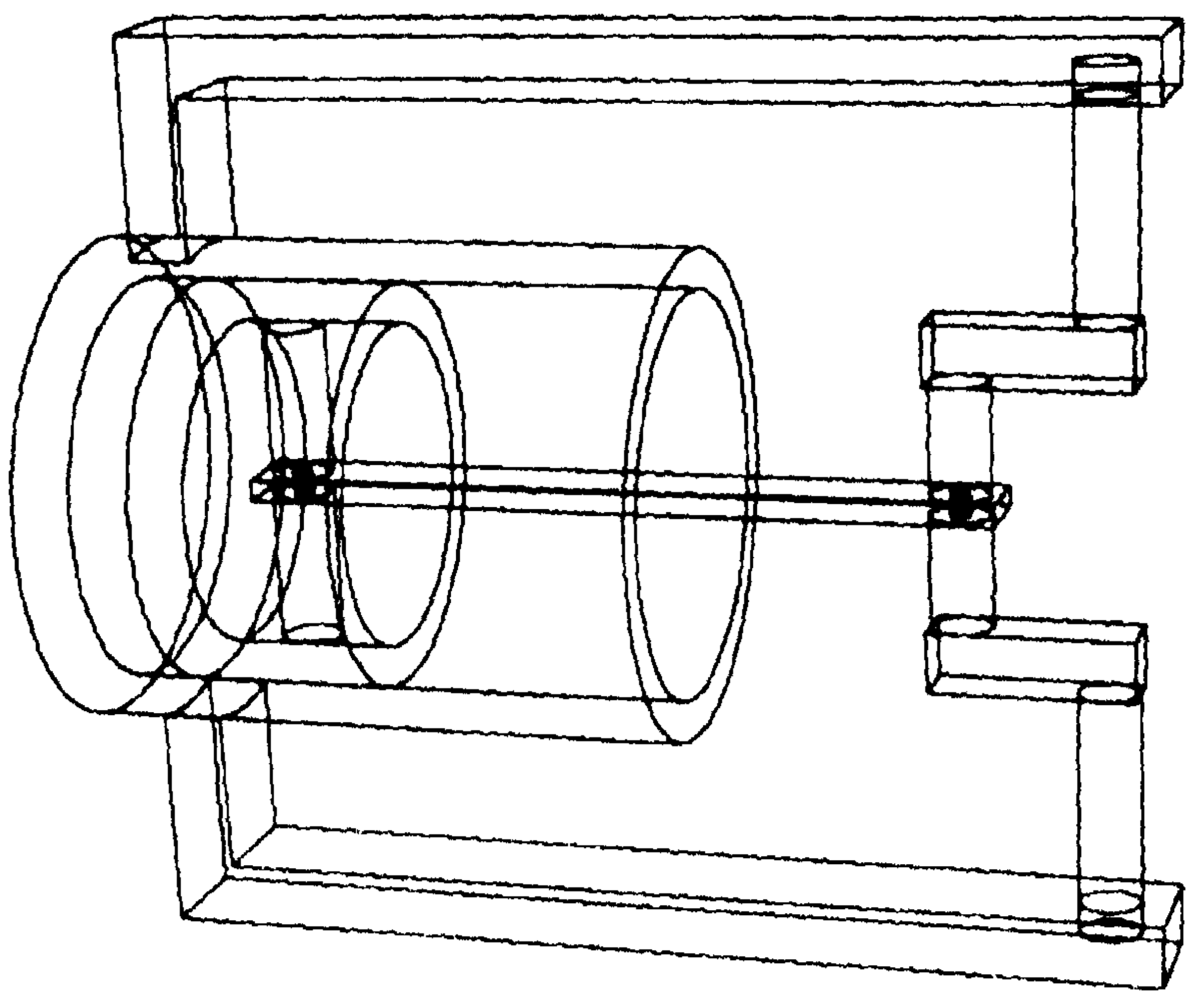


Figure 1: Piston and Crankshaft Mechanism

each other's motion. The crankshaft and frame in Figure 1 form a kinematic pair which is called a revolute pair since the only possible relative motion of the parts in the pair is rotation. Many different kinematic pairs are possible, though only a few are common.

Since the only purpose of kinematic analysis is to simplify the abstract model of the mechanical device by reducing the number of state variables, it is not necessary to identify every kinematic pair in a mechanism. Pairs which are not identified will just result in extra state variables. For example, if the revolute pair formed by the crankshaft and the frame in Figure 1 were not recognized, then the two parts would be assigned separate state variables, and their mutual interaction would have to be computed from the basic physics of bodies in contact which I will describe later in this paper.

My program identifies only very common kinematic pairs like the revolute pair, the prismatic or sliding pair, and the gear pair. These kinematic pairs are typically identified by looking for pairs of parts having subparts with matching symmetries in complementary positions. The details of the algorithm may be found in the previously cited references.

After kinematic pairs have been identified, my program attempts to find *kinematic chains*, chains of parts such that each part in the chain forms a kinematic pair with its successor. A gear train is a kinematic chain, and it has the property that for any position of the first gear in the train, only one position is possible for each other gear in the train. Thus the state of an arbitrarily long gear train may be described by only two state variables: the position and velocity of the first gear in the train.

The mechanism in Figure 1 is also a kinematic chain, of a type known as a linkage. My program can analyze simple linkages like the one in Figure 1, but the general problem of linkage analysis is quite difficult. The problem has, however, received a great deal of attention from mechanical engineers, and a survey of software which does such analysis may be found in [Fallahi and Ragsdell, 1983].

My program analyzes the mechanism in Figure 1 by determining that the axes of the three revolute pairs in

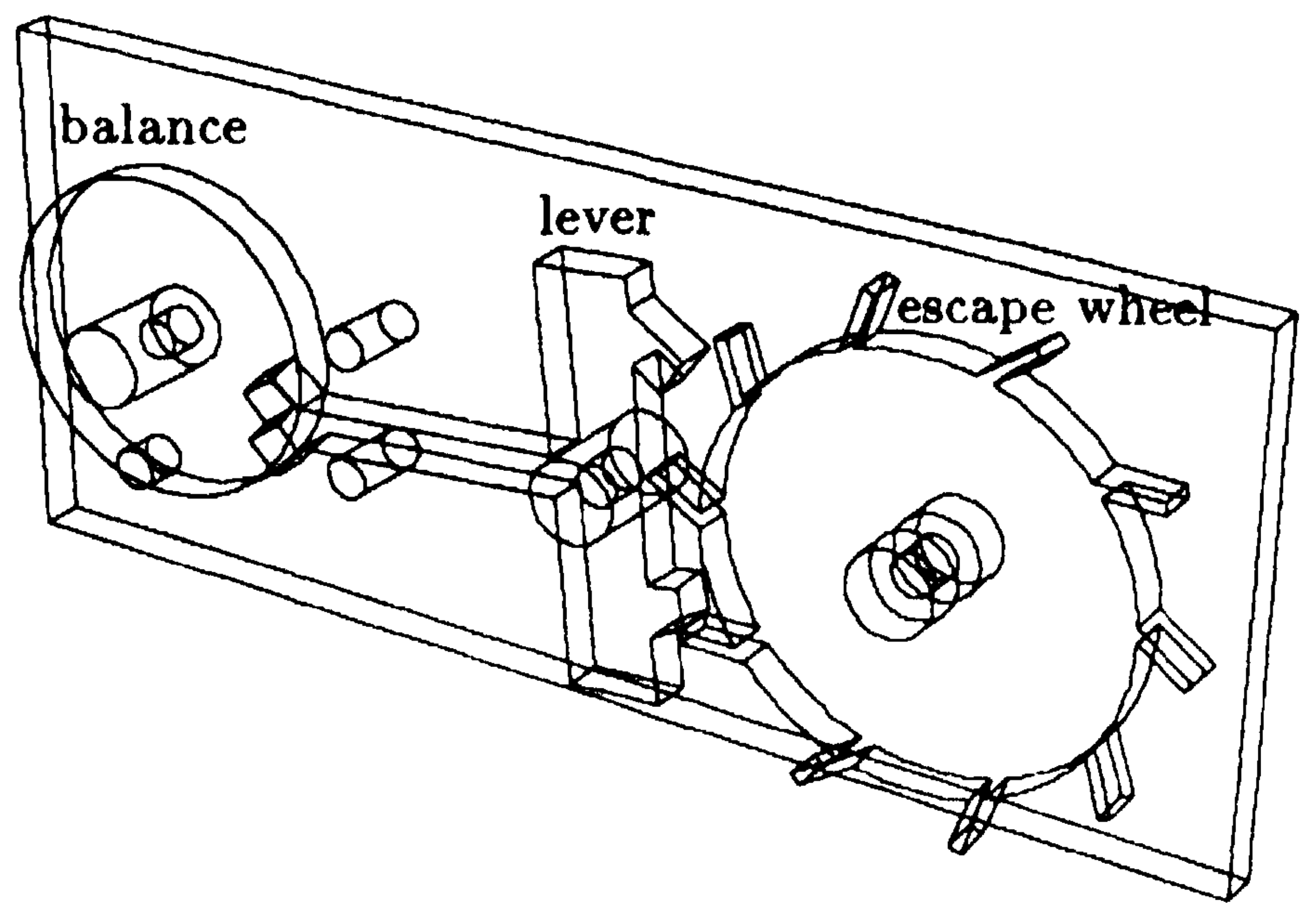


Figure 2: Clock or watch escapement mechanism

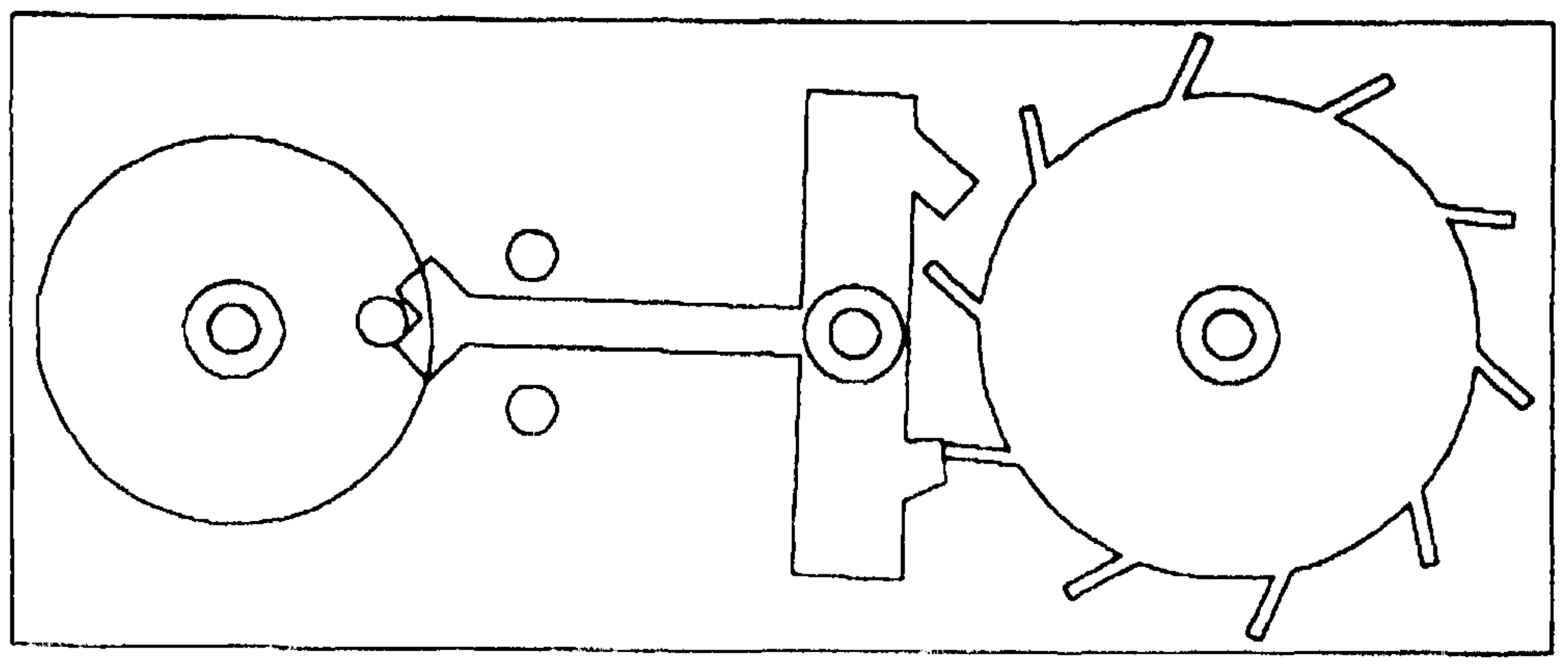


Figure 4: Escape wheel pushes lever and balance

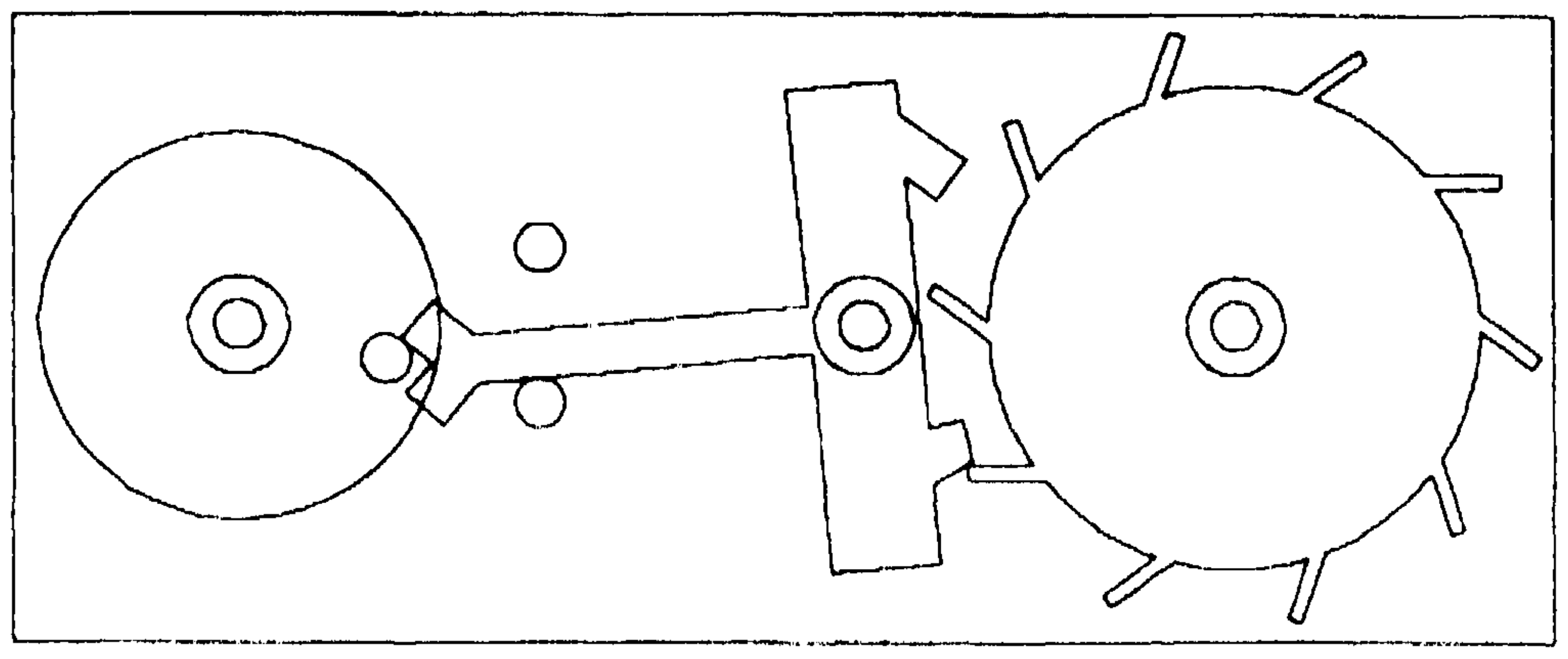


Figure 3: Balance collides with lever

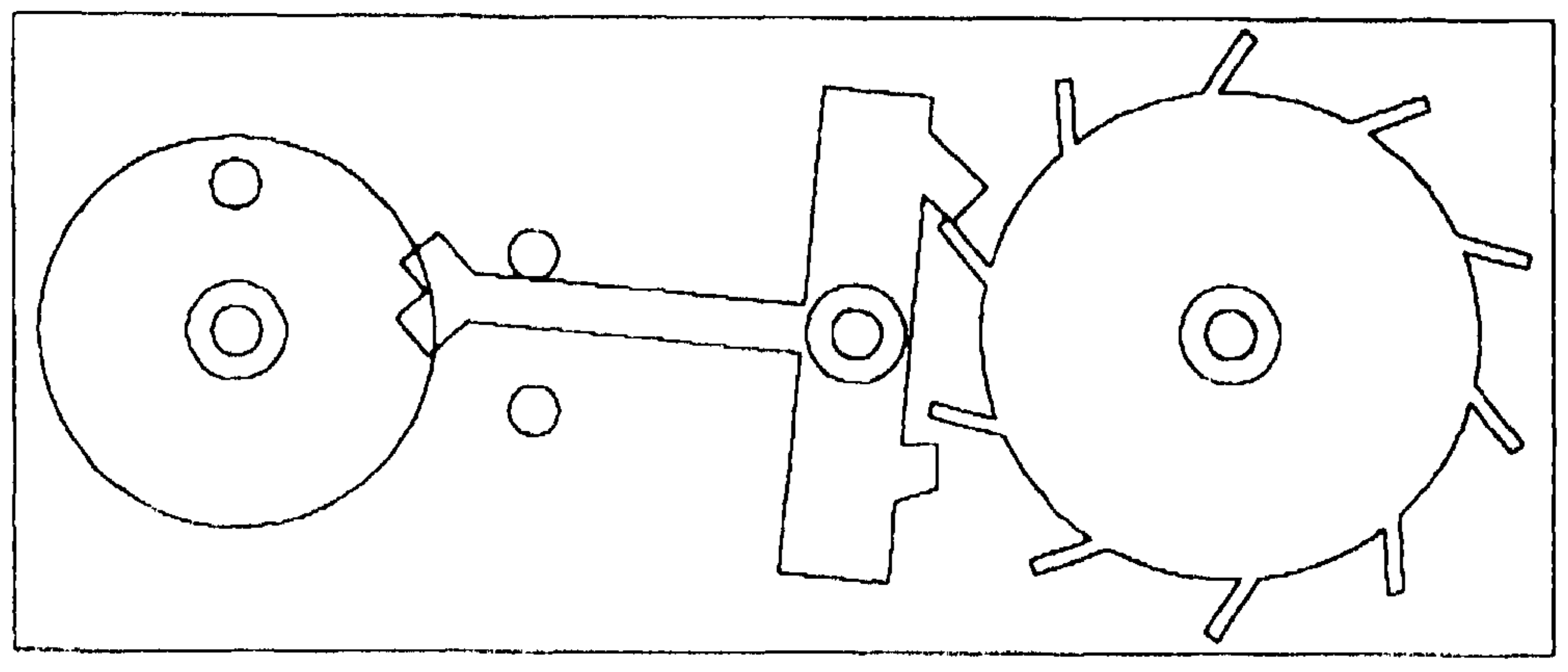


Figure 5: Halfway through a full cycle

the mechanism are parallel and therefore may be projected onto a plane. The projection consists of three points which determine a triangle which may be analyzed with basic trigonometry. This analysis shows that for any position of the crankshaft, only one position is possible for each of the other parts. Thus the state of the device can be represented with only two state variables: the position and velocity of the crankshaft.

My program used the results of kinematic analysis to partition the components of a mechanism into *kinematic subsystems* having the property that the positions of all the components of a subsystem are determined by a single position state variable. Therefore, the total number of state variables for a mechanism will be twice the number of kinematic subsystems, since there will be one position state variable and one velocity state variable for each subsystem.

4 Neglect

When a person creates an abstract model of a physical system, perhaps the most important issue is which aspects of the system should be included in the model, and which ones should be neglected. Different choices result in different abstract models. As a concrete example of such a choice, consider the clock or watch escapement mechanism displayed in Figures 2 through 5.

This mechanism forces the mainspring of the watch to unwind at a constant speed by allowing the escape wheel to advance by only one tooth for each swing of the balance, which is a harmonic oscillator [Cuss, 1952]. The cycle begins in Figure 2 with the motion of the escape wheel, which is attached to the mainspring, blocked by the lever, and the balance motionless at the top of its swing about to start moving towards the lever. In Figure 3 the balance has swung down to hit the lever. The momentum of the balance pushes the lever far enough to free the escape wheel, which then pushes both lever and balance as in Figure 4. Finally, in Figure 5, escape wheel and lever are once again locked, and the balance is at the top of its swing on the opposite side.

The initial kinematic analysis of this mechanism finds that each of the three moving parts forms a revolute pair with the frame. No kinematic constraints on the relative motions of these pairs are found. Thus by the methods of the previous section there are six state variables; three positions and three velocities. The interesting behavior of the mechanism occurs when the moving parts collide and push each other. These interactions are not predicted by the initial kinematic analysis and thus must be modeled in terms of forces between the parts due to (very small) elastic deformations of parts, which therefore can lot be treated as perfectly rigid bodies.

A different but also valid model of the mechanism results from a decision to neglect elastic properties of the mechanism and treat each part as a rigid body. In this kinematic analysis must be used to predict the behavior of the machine, but since the pairs of moving parts are only in contact part of the time, temporary kinematic pairs are formed and dissolved dynamically as the escapement moves. Therefore, the number of state variables changes as the mechanism runs.

My modeling program can automatically form either of these models from the same input (described in Section 2). They both predict the same behavior of the escapement as displayed graphically in Figure 6.

5 Computing Time Derivatives of State Variables

Identifying state variables is only the first step in creating an abstract model of a physical system. The other essential component of the model is a description of how the values of the state variables change. The abstract models my program generates represents this information in the form of a recipe for computing the time derivative of every state variable in the model. The full details of the algorithms used to find these "recipes" are given in [Gelsey, 1989]. Only an outline of the methods will be given here.

Let x_i be the position state variable for kinematic subsystem i, and let vi be its velocity state variable. The time derivative of xi is just vi, so the only non-trivial recipes we need to find are those for the time derivatives of the *Vi.*

To simplify the exposition in this paper I will assume that each kinematic subsystem consists of a single kinematic pair. This assumption is satisfied by the first model of the escapement mechanism described in Section 4. However, neither the second escapement model nor the mechanism in Figure 1 satisfies the assumption. They do satisfy the more general assumptions made in [Gelsey, 1989], which cover a wide selection of common machines. The general treatment, while more complex, is essentially similar to the one given here.

With this assumption,

$$\frac{dv_i}{dt} = \dot{v}_i = \frac{f_i}{m_i} \tag{1}$$

where fi is the net force on subsystem *i* in the direction of motion, and mi is the mass. (In the case of revolute pairs like those in the escapement, fi, is the net torque around the axis of rotation, and mi is the moment of inertia about that axis.)

The total force f_i on subsystem *i* is

$$f_i = -k_i x_i - h_i v_i - \sum_{k \in \text{contacts}} \alpha_{ik} l_k \tag{2}$$

where k_i is the spring constant of the spring, if any, attached to subsystem *i, hi* is the coefficient of linear friction of subsystem i, l_k is the contact force (described below) between the two parts in contact at contact k, and aik is the geometric force multiplier taking the contact force l_k into a force on subsystem *i.* (aik is zero if subsystem *i* is not involved in contact *k.)*

Collisions and pushing are handled in a single uniform way. Contact forces between bodies are elastic and plastic forces due to the distortion of the bodies in contact. My program uses a model of contact forces which greatly simplifies the physics involved but is still quite useful. Bodies are modeled as being rigid, but their volumes are allowed to overlap in space, and this overlap gives rise to a force. If *o* is the depth of overlap, the magnitude / of the force is defined by

$$l = \begin{cases} Eo + D\dot{o}Mo & \text{if } o \geq 0 \text{ and } E + DM\dot{o} \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The contact force is basically being modeled as a linear spring with linear damping, where *E* is the spring constant and *D* is the damping coefficient. The additional factor *Mo* in the second term on the right is there to make the force function continuous. The direction of the force is taken to be perpendicular to the surface normal at the point of contact.

A contact force contributes to vi for two different subsystems. / will be the same for both subsystems since the force acts equally though in opposite directions on each of them, but the factors *a* will be different.

The first step in computing the contact force between two bodies is to determine whether their positions overlap in space, and if so what is their depth of overlap. This computation is done by the "overlap routine", which computes the depth of overlap by applying elementary principles of geometry to primitive solids. See [Gelsey, 1989] for details. To apply Equation 3 we also need the rate of change of overlap, which has the (instantaneous) value

$$\dot{o} = \alpha_{ik} v_i + \alpha_{jk} v_j, \tag{4}$$

where vi, and v*j* are the velocities of the kinematic subsystems involved in contact *k.* The factors aik and *ajk* are the same as those that appear in Equation 2.[2] The geometric *a* factors are also computed by the overlap routine.

So the final "recipe" for finding time derivatives of velocity state variables is:

1. For each kinematic subsystem i, initialize the current value of *vi* to (—kixi—*hiui)/rrii* using the known constant values ki,hi, and mi, and the known current values of the state variables xi and vi.
2. Call the overlap routine on each pair of potentially interacting primitive solids. If they overlap then
   (a) Use Equation 4 to compute the rate of change of overlap o, using aik and *ajk* computed by the overlap routine and the known current values of state variables vi and *vj.*
   (b) Use Equation 3 to compute the contact force ln using the value of *b* just computed, the value of *o* computed by the overlap routine, and the known constant values *E, D,* and *M.*
   (c) Add *—(ocikh)/rni* to *V{* for each of the two interacting subsystems (as required by equations 2 and 1).

6 Numerical Simulation

Section 4 mentions two different abstract models of the escapement mechanism, and Section 5 gives a fairly detailed description of one of them. Both models describe

[2] That the factors are the same is basically the principle of the lever. A force at one end of a lever appears at the other end multiplied by a certain factor, and the velocity appears divided by exactly the same factor.

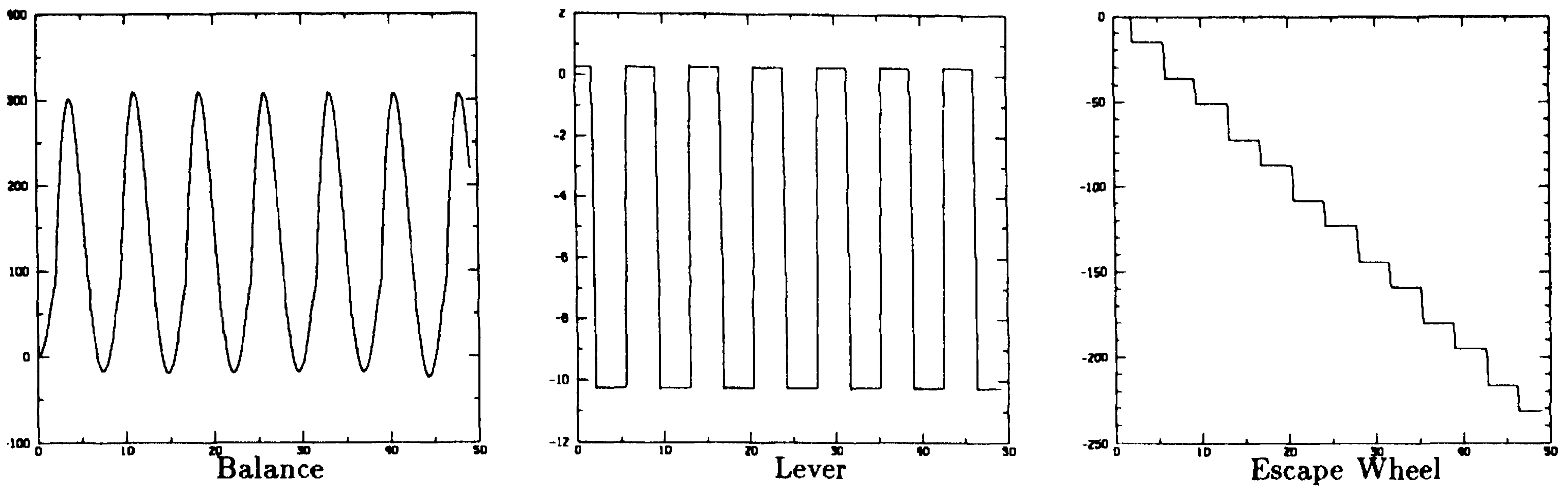


Figure 6: Motion of the escapement mechanism as a function of time

the mechanism abstractly as a set of ordinary differential equations. These equations can be used to numerically simulate the behavior of the mechanism over an arbitrary time period. Figure 6 shows a plot of the results of such a simulation for the escapement mechanism.

7 Behavior Prediction

The abstract models my program generates allow the simulation of the behavior of a physical system arbitrarily far in the future. However, straight simulation is an inefficient way to do long-term behavior prediction for the many physical systems, including most machines, which exhibit repetitive behavior, since simulation does not take advantage of this repetition. My program uses short, carefully controlled numerical simulations to attempt to identify repetitive behavior patterns which can then be used to do long-term behavior prediction.

Instead of looking for repetition directly in state variable behavior like that shown in Figure 6, my program looks at the kinetic energy of the system as a function of time. If the system has a regular behavior then so will its kinetic energy. In particular the number of global minima of the kinetic energy should be the same in each behavioral cycle, and these minima should appear in the same places.

The program generates and tests hypotheses about the system's behavioral repetition. The hypothesis data structure has the following elements:

1. The duration of the system's basic behavioral pattern (loop).
2. The number of global kinetic energy minima during each loop.
3. The displacement of each state variable from one loop to the next. For example, in the behavior of the escapement mechanism as show in Figure 6, the balance and lever have the same positions in each loop, but the escape wheel position is offset by a fixed amount from one loop to the next.

Hypotheses are generated by examining the states generated by a numerical simulation which have globally minimal kinetic energy. These are simply states whose kinetic energy is both locally minimal and near the bottom of the observed range of kinetic energy values. The full long-term behavior prediction algorithm is:

1. Numerically simulate the behavior of the system. As each new state is generated, identify global minima of the kinetic energy.
2. Ignore the first few minima, letting the system settle down. Then form a hypothesis consistent with the time and state variable values of the next two minima: the duration is the difference in their time values, the number of minima is one, and the displacements are the differences in the state variable values. When the next minima is found, test the hypothesis. If it fails, form a new hypothesis consistent with the new information. Repeat this process until a successful hypothesis is found.
3. Use the successful hypothesis to jump the system forward in time, say to a time when half its total energy has been dissipated. The number of loops to jump forward by is computed from the current total energy of the system and the total energy dissipated in each loop. The jump is accomplished by applying the hypothesized displacements to the state of the system the appropriate number of times. The numerical simulation is then continued in this new situation, and the hypothesis is tested again and modified if necessary.

This algorithm has been implemented and successfully applied to the behavior shown in Figure 6. It finds a loop with two global kinetic energy minima.

8 Related Work

While there has been considerable investigation of different types of abstract models for physical systems [Gentner and Stevens, 1983], the problem of automatically generating such models has received little attention. However, work has been done which is related to portions of my model generation process.

A great deal has been written about kinematic analysis as a human activity by people engaged in creating abstract models of mechanical systems. The classic work is [Reuleaux, 1876], and a recent example is [Suh and Radcliffe, 1978]. The problem of recognizing kinematic pairs has been generally ignored in this work, since it is a relatively easy activity for humans even though it is difficult to formalize.

Several artificial intelligence researchers have investigated kinematic analysis, including Stanfill[1983] and Gelsey[l987]. Faltings[1987] and Joskowicz[1988] have attempted to formalize the problem of kinematic pair recognition by analysis of the configuration space of a mechanism.

A number of software tools exist to help mechanical engineer build abstract models of machines. A typical example is ADAMS [Dawson, 1985]. A survey of such software may be found in [Haug, 1984].

9 Conclusion

1 have described a method for the automated construction of abstract models of mechanical devices. The state-variable based models that are generated are used extensively in the physical sciences, and apply to a much wider variety of physical systems than those that my program currently handles. The particular algorithms my program uses to find recipes for time derivatives of state variables are specific to mechanics, but the central idea, the application of basic laws of physics relating state variables to their time derivatives, should make possible similar algorithms covering a considerably more varied range of physical phenomena.

Automated modeling of physical systems is an important contribution to artificial intelligence research concerning reasoning about physical systems, because any reasoning about a physical system involves an abstract model, and as long as these models are "handmade", we can't be sure how much of the reasoning which should be done by the program that manipulates the model is actually being done by the person who builds the model.

References

[Bobrow, 1985] Daniel G. Bobrow, editor. *Qualitative Reasoning about Physical Systems.* MIT Press, 1985.

[Cuss, 1952] T. P. Camerer Cuss. *The Story of Watches.* MacGibbon & Kee Ltd., London, 1952.

[Dawson, 1985] Gary Dawson. The dynamic duo: Dram and Adams. *Computers in Mechanical Engineering,* March 1985.

[Fallahi and Ragsdell, 1983] B. Fallahi and K. M. Ragsdell. A compact approach to planar kinematic analysis. *Transactions of the ASME Journal of Mechanisms, Transmissions, and Automation in Design,* 105:434-440, 1983.

[Faltings, 1987] Boi Faltings. *Qualitative Place Vocabularies For Mechanisms in Configuration Space.* PhD thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, July 1987.

[Forbus *et al,* 1987] Kenneth D. Forbus, Paul Nielsen, and Boi Faltings. Qualitative kinematics: a framework. In *Proceedings of 1987 IJCA1,* Milan, Italy, 1987.

[Gelsey and McDermott, 1988] Andrew Gelsey and Drew McDermott. Spatial reasoning about mechanisms. Computer Science Department Research Report YALEU/DCS/RR-641, Yale University, August 1988. To appear in *Advances in Spatial Reasoning,* edited by Su-shing Chen, Ablex.

[Gelsey, 1987] Andrew Gelsey. Automated reasoning about machine geometry and kinematics. In *Proceedings of the Third IEEE Conference on Artificial Intelligence Applications,* Orlando, Florida, 1987.

[Gelsey, 1989] Andrew Gelsey. From CAD/CAM to simulation: Automatic model generation for mechanical devices. In Drs. Paul Fishwick and Richard Modjeski, editors, *Knowledge-Based Simulation: Methodology and Application.* Springer-Verlag, 1989. To appear.

[Gentner and Stevens, 1983] Dedre Gentner and Albert L. Stevens. *Mental Models.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983.

[Haug, 1984] Edward J. Haug. A survey of dynamics software. In Edward J. Haug, editor, *Computer Aided Analysis and Optimization of Mechanical System Dynamics,* pages 24-31. Springer-Verlag, Berlin, 1984.

[Joskowicz, 1988] Leo Joskowicz. *Reasoning about Shape and Kinematic Function in Mechanical Devices.* PhD thesis, New York University Dept. of Computer Science, September 1988.

[Nielsen, 1988] Paul E. Nielsen. *A Qualitative Approach to Rigid Body Mechanics.* PhD thesis, Dept. of Computer Science, University of Illinois at Urbana-Champaign, November 1988.

[Requicha, 1980] Aristides A. G. Requicha. Representations for rigid solids: Theory, methods, and systems. *ACM Computing Surveys,* 12:437-464, 1980.

[Reuleaux, 1876] Franz Reuleaux. *The Kinematics of Machinery.* Macmillan and Co., London, 1876.

[Stanfill, 1983] Craig Stanfill. *Form and Function: The Representation of Machines.* PhD thesis, Dept. of Computer Science, University of Maryland, November 1983.

[Suh and Radcliffe, 1978] Chung Ha Suh and Charles W. Radcliffe. *Kinematics and Mechanisms Design.* John Wiley & Sons, New York, 1978.

A FOCUSED, CONTEXT-SENSITIVE APPROACH TO MONITORING

Richard J. Doyle
Suzanne M. Sellers
David J. Atkinson
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109

Abstract

We address two issues which arise in the task of detecting anomalous behavior in complex systems with numerous sensor channels: how to adjust alarm thresholds dynamically, within the changing operating context of the system, and how to utilize sensors selectively, so that nominal operation can be verified reliably without processing a prohibitive amount of sensor data. Our approach involves simulation of a causal model of the system, which provides information on expected sensor values, and on dependencies between predicted events, useful in assessing the relative importance of events so that sensor resources can be allocated effectively.

1. The Monitoring Problem

Timely detection of anomalous behavior is essential for the continuous safe operation and longevity of aerospace systems. The pilot of a jet aircraft must be aware of any conditions which may affect thrust during the critical moments of takeoff. The thermal environment onboard Space Station Freedom must be carefully controlled to provide uninterrupted life support for the crew. The Mars Rover must react quickly to an unpredictable environment or the mission may come to an abrupt conclusion.

Monitoring a physical system involves a number of problem-solving tasks. Dvorak, in his survey of work on expert systems for monitoring and control [Dvorak 87], lists among these tasks recognizing abnormal conditions, combining sensory information into a picture of the global state of a system, isolating faults, predicting both normal and faulted behavior, and maintaining safe operation in the presence of faults. In addition, decisions must often be made in limited time, and with partial information.

The monitoring problem becomes more difficult when the behavior of a physical system involves interactions among components or interaction with an environment. Under these conditions, correct operation becomes context-dependent; it is not possible to determine *a priori* a set of sensor values which always imply nominal operation. Moreover, when the number of sensors in a physical system becomes very large, the ability to combine sensor data into a picture of the global state of a system becomes compromised. Studies of plant catastrophes have revealed that information which might have been useful in preventing disaster was typically available but was not prominent enough within the overwhelming morass of data presented to operators.

In this paper, we concentrate on the initial step in the monitoring process—detecting anomalous behavior quickly and reliably. We do not address here the equally important steps of tracking faulted behavior and determining control actions to continue operation in the presence of faults. Within this focus, we address two important issues: (1) how to adjust nominal sensor value expectations dynamically, taking into account the changing operating context of the system, and (2) how to utilize sensors selectively, determining which subset of the available sensors to use at any given time to verify nominal operation efficiently, without processing a prohibitive amount of data.

2. Two Issues

The traditional approach to verifying the correct operation of a system being monitored involves associating alarm thresholds with sensors. Fixed threshold values for each sensor are determined ahead of time by analyzing the designed nominal behavior for the system. Whenever a sensor value crosses a threshold during operation, an alarm is raised.

The problem with this approach is that the nominal behavior of even moderately complex systems often depends on context. For example, an earth-orbiting spacecraft periodically enters and emerges from the Earth's shadow. Impingent solar radiation changes the thermal profile of the spacecraft, as does the configuration of currently active and consequently, heat-generating subsystems on board. Thresholds on temperature sensors should be adjusted accordingly. A particular temperature value may be indicative of a problem when the spacecraft is in shadow or mostly inactive, but may be within acceptable limits when the spacecraft is in sunlight or many on-board systems are operating.

Fixed alarm thresholds are useful for defining the operating limits of a physical system, such as the point of overbalance of a rover, or the temperature at which, say, the onboard computer of a spacecraft is at risk. Nonetheless, they are woefully inadequate for verifying the nominal operation of a system with many operating modes, or one which interacts with an environment The problem is that fixed alarm thresholds are derived from an over-summarized model of the behavior of a system. If the thresholds are chosen conservatively, then false alarms occur. If they are chosen boldly, then undetected anomalies occur. What is needed is a capability for adjusting alarm thresholds dynamically. Alarm thresholds should be chosen according to expectations about the nominal behavior of a system as it changes in different operating contexts. Later on in this paper, we present our approach to dynamic alarm threshold adjustment based on causal simulation of the device.

Another issue which arises in monitoring concerns how to best utilize available sensors to efficiently and reliably, but not necessarily comprehensively, verify the nominal operation of a physical system. Just as the nominal values in a system being monitored depend on context, so do the subset of sensors,which can most directly verify those values depend on context. The familiar activity of driving an automobile helps to illustrate this idea. A variety of sensors are provided to the operator of an automobile: fuel gauge, temperature gauge, speedometer, several mirrors, etc. However, the driver does not use all of these diverse sensors all of the time. The speedometer may be checked periodically, or when a speed limit sign is passed; the right-side mirror is probably only used during lane changes. There are two points to be made: one concerns relevance, the other concerns resources.

Individual sensors are appropriate for verifying only some small, localized subset of the possible behavior of a system. The choice of which sensors to sample and interpret at any particular time should be based on expectations of what is to happen in the system and, perhaps, how it is to interact with an environment. However, even after a suitable subset of the available sensors is identified, there may not be the resources available, whether human or machine, to sample all the selected sensors and interpret the data within a required response frame. What is needed is a capability for assessing the importance of predicted events, so that while it may not be possible to comprehensively verify the expected behavior of a system, still the most reliable verification within available resources can be performed.

An illustration of the need to focus attention in monitoring comes from the jet aircraft domain. Some of the recent commercial aircraft catastrophes have been attributed to insufficient thrust during the critical moments of takeoff. There are many possible indicators of low thrust available to a flight crew. For example, a low exhaust gas temperature in an engine may produce reduced thrust Also, a low turbine fan rotation speed in an engine may imply reduced thrust, because fuel input is based partly on this parameter. The challenge is to direct the attention of the flight crew towards information useful for planning actions in real time without overwhelming them.

A monitoring strategy must take into account the reality that not all sensors should or can be checked all of the time. As the operating context of the physical system being monitored changes, the collection of sensors which provide the most immediate information on the state of the system also changes. Further on in this paper, we present our approach to sensor planning in monitoring. We describe a method for assessing the importance of predicted events in a system, based on reasoning about causal dependencies among events, and about how events relate to intended goals of the designers or operators of a system.

3. Other Work

Within NASA, there are other projects underway in which the goal is to develop a monitoring and a diagnosis capability for aerospace systems. Among these is the KATE project at the Kennedy Space Center, whose domain is the Shuttle Liquid Oxygen Loading system [Scarl *et al* 88]. In this project, causal models are used to support sensor validation, fault diagnosis, and the planning of control actions.

The goal of the FAULTFINDER project at Langley Research Center [AbboU 88] is to develop an inflight monitoring and diagnosis capability for jet aircraft These investigators have explored the use of multiple representations and multiple levels of abstraction to be able to reason about diverse faults, to focus attention during reasoning, and to provide accessible information to a flight crew.

Outside of NASA, there have been a number of efforts aimed at developing knowledge-based expert systems for monitoring and control. ESCORT [Sachs *et al* 86] is a shell for developing real-time expert systems to filter and focus information during plant emergency situations, REACTOR [Nelson 82] is an expert system for monitoring nuclear power plants which detects anomalous behavior, assesses the seriousness of the situation, and recommends appropriate actions. REALM [Touchton and Casella 86] is an advisory system which detects and classifies emergencies in nuclear power plants and is able to predict further consequences of those emergencies.

Numerous other examples exist of efforts to develop monitoring and control systems. The reader is referred to Dvorak's excellent survey of the area [Dvorak 87] and to the survey of real-time knowledge-based systems in [Laffey e t a l 88].

The causal reasoning paradigm, which is at the core of our approach to the monitoring problem, is now a well-established area of investigation within Artificial Intelligence. The advantages of the causal approach, which involves modeling a system at the level of components and mechanisms, include the ability to reason about unforeseen interactions, the ability to reason about dependencies among events, and the ability to generate accessible explanations. The seminal efforts in this area include Forbus' process-centered approach [Forbus 85], de Kleer and Brown's device-centered approach [de Kleer and Brown 85], and Kuipers' qualitative mathematics approach [Kuipers 86].

In the specific area of monitoring, Dvorak's MIMIC project stands out as the most comprehensive current research effort [Dvorak and Kuipers 89]. Dvorak creates a component-connection model of a system and employs the QSIM qualitative simulator [Kuipers 86] to generate expectations about the system's nominal behavior. An inductive learning method is used to create a set of symptom-fault rules for known faults, and these rules support the formation of fault hypotheses whenever sensor data does not match predictions from the causal model. When anomalous behavior exists, several fault models can be tracked in parallel until one emerges as the hypothesis with the most explanatory power. The ability to continue tracking a faulted system is important because large, complex systems almost always contain faults and the challenge is to maintain safe operation in the presence of faults.

4. The Approach

At the center of our approach to addressing the two issues of dynamic alarm thresholds and sensor selection is a causal model of the system being monitored and possibly, its environment. Simulation of this model directly solves the problem of alarm threshold adjustment. Predicted values and their time tags indicate how and when to alter the alarm thresholds associated with sensors so that they reflect expectations about the nominal operation of the system in changing contexts.

Another result of simulation is information about causal dependencies among predicted events of a system. This information is used to assess the importance of individual events. Briefly, the most important events are taken to be those which either cause or are caused by the greatest number of other events. An ordering on predicted events reflecting this causal notion of importance serves as the basis for allocating sensor resources to selectively verify the expected behavior of a system [Doyle *et al* 87].

In the remainder of this section, we describe (1) the architecture of our predictive monitoring system, called PREMON, (2) what our causal models of physical systems look like, and how they are simulated, and finally, (3) our approach to sensor planning, based on analyzing causal dependencies.

4.1 Architecture

There arc three modules in the PREMON system: a causal simulator, a sensor planner, and a sensor interpreter. Sec Figure 1.

The causal simulator takes as input a causal model of the system to be monitored, and a set of events describing the initial state of the system and perhaps some future scheduled events. The causal simulator produces as output a set of predicted events, and a graph of causal dependencies among those events.

The sensor planner takes as input the causal dependency graph generated by the causal simulator and determines which subset of the predicted events should be verified.

These events are passed on to the sensor interpreter.

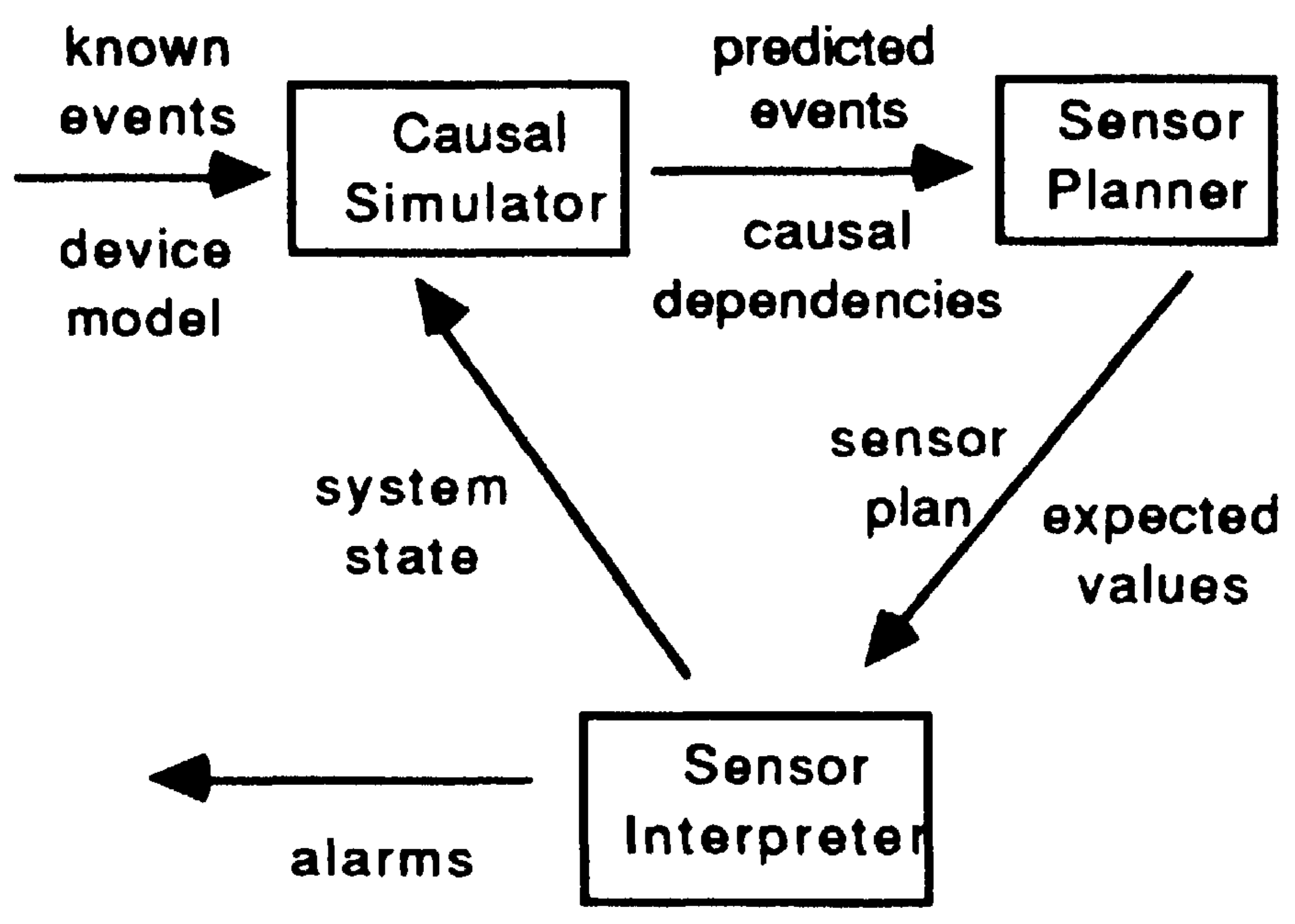


Figure 1. Architecture of PREMON.

The sensor interpreter compares expected values as predicted by the causal simulator with actual values from sensors. Alarms are raised here when there are discrepancies. Finally, the most recent sensed data is passed back to the causal simulator to contribute to another predict-plan-sense cycle of monitoring.

4.2 Causal Models and Causal Simulation

We represent physical systems as a collection of *quantities* and *mechanisms.* Quantities are continuous parameters such as temperature, position, and amount-of-stuff. Quantities are specified by a *physical object,* a *type,* and an *order.* Examples of quantities are {HEATER TEMPERATURE RATE} and {SWITCH POSITION AMOUNT}.

Events describe discontinuous changes in the value of a quantity. Events are specified by a quantity, a *value,* and a *moment.* Examples of events are {HEATER TEMPERATURE RATE POSITIVE 61} and {VALVE-17 POSITION AMOUNT OPEN0}.

Mechanisms capture causal relations between quantities. More specifically, they describe how a change in one quantity results in a change in another quantity. Examples of mechanisms are HEAT FLOW, THERMAL EXPANSION, LATCH, and GRAVITY. A mechanism is specified by a *time constant,* a *distance,* a *sign,* an *efficiency,* a *bias,* an *alignment,* and a *medium.* Figure 2 shows the representation of a HEAT FLOW mechanism.

A *causal model* then, consists of a set of quantities and a set of mechanisms between those quantities. A causal model can be represented by a graph where the nodes are quantities and the arcs are mechanisms. Simulation of a causal model involves predicting new events, via mechanisms, from known or previously predicted events. The simulation method outlined in the next few paragraphs is described more fully in [Doyle 88, 89].

When the quantity named in an event appears as the cause quantity in a mechanism, a new event is predicted as follows: (1) the quantity of the new event is the effect quantity of the mechanism, (2) the value of the new event is computed from the value of the given event and the sign and efficiency of the mechanism, (3) the moment of the new event is computed from the moment of the given event and the time constant and distance of the mechanism, and (4) the new event occurs only when constraints specified in the bias, alignment, and medium of the mechanism are satisfied. The bias of a mechanism specifies constraints on directions of change. For example, current through a wire can cause it to heat up, but not to cool down. The alignment of a mechanism specifies constraints expressed as inequalities. For example, heat flow is from the warmer to the cooler site. The medium of a mechanism is a physical connection such as a wire, a pipe, a linkage, etc. The predicted effect occurs only when the specified physical connection is in place.

In Figure 2, a typical event is shown, this one describing a temperature change. The HEAT FLOW mechanism is used to predict another temperature change event.

QUANTITY	Chiller Temperature Rate
VALUE	Negative
MOMENT	60

TIME CONSTANT	1.0
DISTANCE	10.0
SIGN	+
EFFICIENCY	0.95
BIAS	nil
ALIGNMENT	<
MEDIUM	{Chiller Pipe-4 Mirror}

QUANTITY	Mirror Temperature Rate
VALUE	Negative
MOMENT	70

Figure 2. A cause event, a mechanism, and an effect event.

Simulation would be straightforward if physical systems could be modeled exclusively as simple mechanism chains between input and output quantities. However, some mechanisms serve to enable or disable other mechanisms, such as a valve controlling a fluid flow, or a latch inhibiting the transmission of motion through a mechanical coupling. In these cases, the contributions of the separate mechanisms combine multiplicatively. The contributions of separate mechanisms also can combine additively, as when two fluid lines empty into the same container, or two opposed forces produce an equilibrium state. The details of our simulation method for interacting mechanisms are described in [Doyle 88,89].

4.3 Sensor Planning

The output of the causal simulator is a trace of predicted events and the dependencies among them. The dependencies are derived from the mechanism structure of the system. A dependency between two events is a record that there is a mechanism in the system which causally relates the events.

Analysis of the causal dependencies in a simulation trace supports decisions about which events to monitor. In our approach, the importance of events is assessed by determining how many other events are effects or causes of a given event. In other words, the importance of an event is related to the amount of subsequent activity it supports and the amount of activity which supports its occurrence. Critical events which lie on several causal paths between inputs and outputs should be verified with care, perhaps with a battery of sensors. On the other hand, events which are side effects and do not support further activity in the system may be ignored completely. See Figure 3.

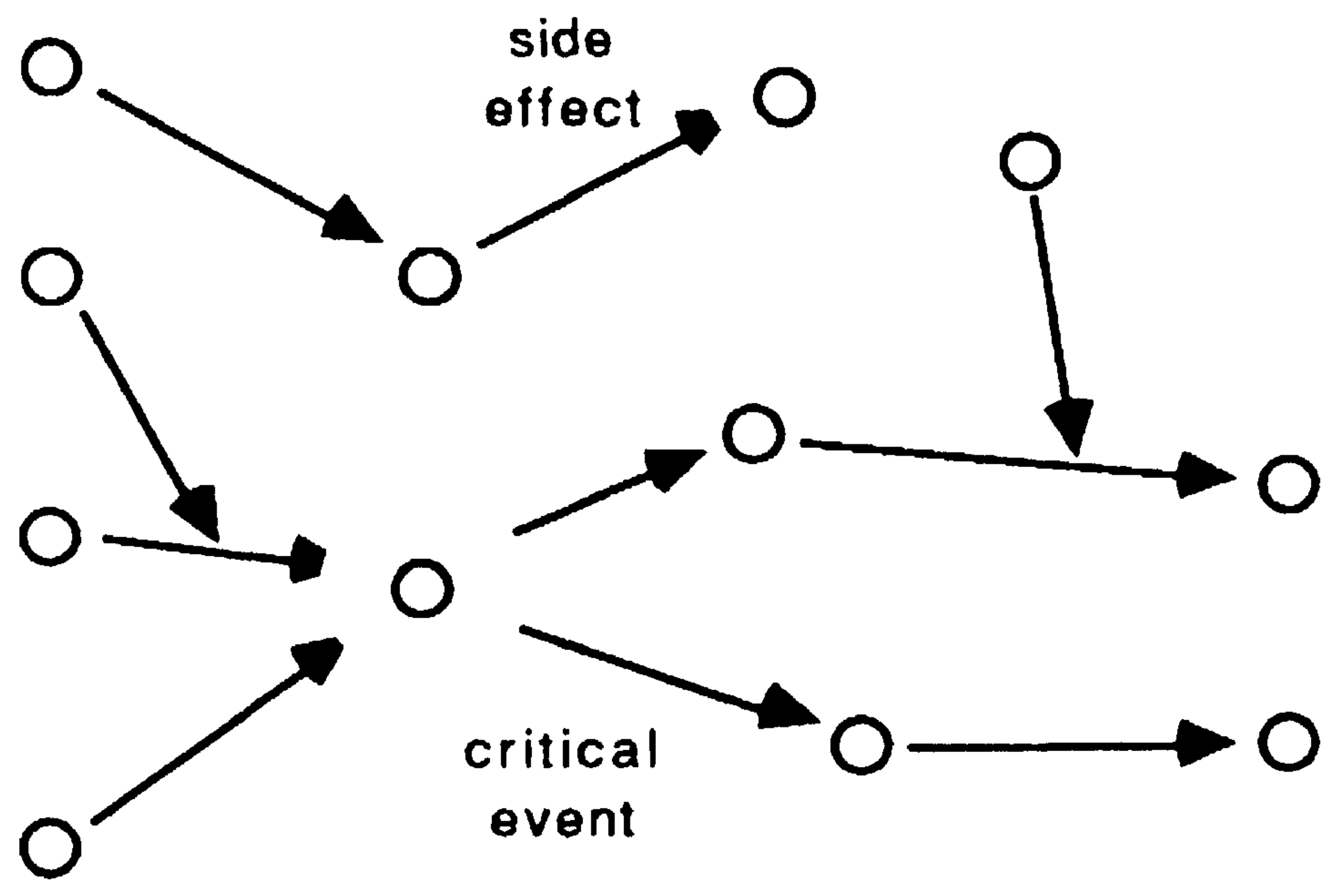


Figure 3. Assessing the importance of events.

This analysis method weights all dependencies in a causal graph equally. Several criteria might form the basis of a non-uniform weighting scheme. For example, *a priori* or empirical knowledge about probabilities of failure might bias the allocation of sensor resources towards those components in a system known to be unreliable. Similarly, parts of a system where redundancy has been built in might be given less careful attention than other parts.

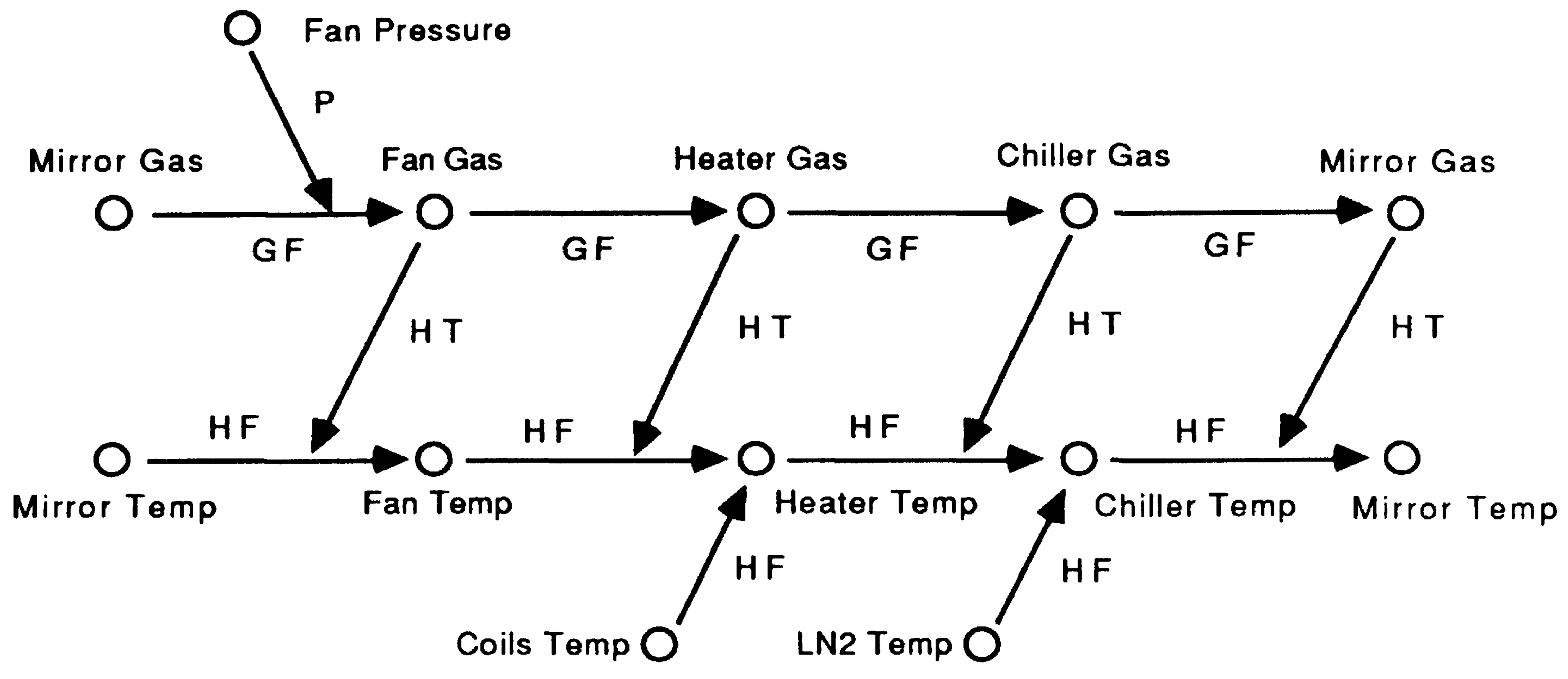
Our causal analysis method for determining what subset of predicted events to monitor is similar to the minimum entropy method of [de Kleer and Williams 87) for determining the site of the most useful next measurement in troubleshooting. Their technique involves propagating observed values and failure probabilities along a causal dependency graph for a circuit.

5. An Example: The JPL Space Simulator

The JPL Space Simulator is an environmental chamber in which spacecraft and instruments can be subjected to some of the aspects of the space environment: intense cold, near vacuum, and solar radiation.

A mirror is used to direct simulated solar radiation onto the spacecraft or instrument inside the chamber. This mirror must be cooled separately from the shroud which surrounds the chamber to compensate for the additional radiation falling on it. Cold gaseous nitrogen is used as the cooling medium and is circulated by a fan. Chilling is achieved by injecting liquid nitrogen into the gaseous nitrogen. Warming is achieved through an electrical heater. A causal simulation of this cooling circuit is shown in Figure 4.

Using the causal analysis technique outlined above, the flow of gaseous nitrogen at the fan is identified as the single most critical event in the predicted nominal behavior of the circuit. This event affects gas flow around the entire circuit and indirectly, heat flow around the entire circuit. The only events unaffected by this event are the source temperature changes at the chiller and heater. This result of causal analysis captures the intuitive notion that nothing at all happens in the cooling circuit if the fan stops operating. Other important events in the predicted operation of the circuit arc the temperature changes at the chiller and heater. Measurements made at these sites also provide informative feedback about the nominal operation of the circuit.



P: Pump GF: Gas Flow HT: Heat Transport HF: Heat Flow

Figure 4. The mirror cooling circuit.

This example has been implemented and illustrates our current causal simulation and sensor planning capabilities. We are beginning to target our developing predictive monitoring capability to other aerospace systems existing or being designed within NASA. Potential domains include power and thermal distribution systems for planetary rovers, ground antenna control systems in the Deep Space Network, and subsystems onboard earth-orbiting spacecraft.

6. Future Work

Our investigations so far have clarified and uncovered other research issues which also need to be addressed. In this section, we briefly outline these issues.

6.1 Generation of Fault Hypotheses

Because complex systems rarely operate completely fault-free, a monitoring system should include also the capability to generate fault hypotheses and to incorporate fault models into the current model of a system so that behavior can continue to be predicted and tracked in the presence of faults. During the diagnostic process, it may be necessary to track several models in parallel, each corresponding to a different fault hypothesis.

6.2 Selective Si

We have already made a case for selective use of sensors. In this vein, just as there may be insufficient resources to interpret profusive sensor data, there may be also insufficient resources to conduct a comprehensive simulation of a system. Decisions about which part of a system to simulate also are likely to be context-dependent. Selective simulation interacts with sensor planning in the following way: When simulation cycles arc shifted from one part of a system to another, an overhead of sensor reads is needed to determine current state, in order to bootstrap the new focus of simulation.

6.3 How Far Ahead to Predict

Another way of dealing with the problem of limited resources for simulation is to constrain the number of events predicted in a pass through the causal simulator. The tradeoff is between generating enough of a causal dependency graph to drive sensor planning and maintaining a real-time predict-plan-sense monitoring cycle. Another potential factor is ambiguity in simulation. Particularly when a system model is qualitative, there may be insufficient *a priori* information to determine which of several alternate states a system may enter. A direct way to counteract such branching is to suspend simulation and utilize explicit sensor reads to pin down ambiguous states.

7. Conclusions

Detecting anomalies in the operation of a system is a difficult problem when the behavior of the system is complex or involves interaction with an environment, and when the number of sensor channels is large. Under these conditions, nominal values and the most informative sensor data change according to context. We have addressed two specific issues in monitoring: how to adjust alarm thresholds dynamically, and how to verify behavior selectively but reliably. At the center of our approach to solving both problems is the use of a causal model of the system being monitored. Simulation of a causal model serves both to generate expectations about nominal sensor values, and to provide dependency information useful in assessing the importance of predicted events and in allocating sensor resources accordingly.

The key idea in this paper is letting go of the notion of comprehensive monitoring. More likely than not, there will be insufficient resources for predicting behavior and interpreting sensor data. In the face of this limitation, our emphasis is on verifying the operation of a system efficiently and reliably, by carefully focusing computational resources to gather the most informative, if incomplete, feedback on nominal operation within changing contexts.

Acknowledgements

The content of this paper has been enhanced through discussions with Kathy Abbott, Dan Dvorak, Lorraine Fesq, David Franke, Ben Kuipers, and Ethan Scarl. The expertise of Bob Lane was crucial in developing the model of the mirror cooling circuit

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[Abbott 88] Kathy H. Abbott, "Robust Operative Diagnosis as Problem Solving in a Hypothesis Space," *7th National Conference on Artificial Intelligence,* St. Paul, Minnesota,1988.

[de Kleer and Brown 85] Johan de Kleer and John Seely Brown, "A Qualitative Physics Based on Confluences," in *Qualitative Reasoning about Physical Systems,* D. Bobrow, ed., MIT Press, 1985.

fde Kleer and Williams 87] Johan de Kleer and Brian C. Williams, "Diagnosing Multiple Faults," *Artificial Intelligence,* 32, 1987.

[Doyle *et al* 87] Richard J. Doyle, Suzanne M. Sellers, and David J. Atkinson, "Predictive Monitoring Based on Causal Simulation," *2nd NASA AI Forum,* 1987.

[Doyle 88] Richard J. Doyle, "Hypothesizing Device Mechanisms: Opening Up the Black Box," Report TR-1047, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1988.

[Doyle 89] Richard J. Doyle, "Reasoning About Hidden Mechanisms," I *lth International Joint Conference on Artificial Intelligence,* Detroit, 1989.

[Dvorak 87] Daniel L. Dvorak, *Expert Systems for Monitoring and Control,* Report AI 87-55, Artificial Intelligence Laboratory, The University of Texas at Austin, 1987.

[Dvorak and Kuipers 89] Daniel L. Dvorak and Benjamin J. Kuipers, "Model-Based Monitoring of Dynamic Systems," *11th International Joint Conference on Artificial Intelligence,* Detroit, 1989.

[Forbus 851 Kenneth D. Forbus, "Qualitative Process Theory," in *Qualitative Reasoning about Physical Systems,* D. Bobrow, ed., MIT Press, 1985.

[Kuipers 86] Benjamin J. Kuipers, "Qualitative Simulation," *Artificial Intelligence,* 29, 1986.

[Laffey *et al* 88] Thomas J. Laffcy, Preston A. Cox, James L. Schmidt, Simon M. Kao, and Jackson Y. Read, "Real-Time Knowledge-Based Systems," *AI Magazine,* 9, no. 1,1988.

[Nelson 82] William R. Nelson, "Reactor: an Expert System for Diagnosis and Treatment of Nuclear Reactor Accidents," *2nd National Conference on Artificial Intelligence, Pittsburgh,* 1982.

[Sachs *et al* 86] Paul A. Sachs, Andy M. Patcrson, and Michael H. M. Turner, "Escort - an Expert System for Complex Operations in Real Time," *Expert Systems,* 3, 1986.

[Scarl *et al* 88] Ethan A. Scarl, John R. Jamicson, and Edwin New, "Deriving Fault Location and Control from a Functional Model," *3rd IEEE Symposium on Intelligent Control,* Arlington, Virginia, 1988.

[Touchton and Casclla 86] Robert A. Touchton and Mike Casella, "Reactor Emergency Action Level Monitor: A Real Time Expert System," *Instrument Society of America Convention,* 1986.