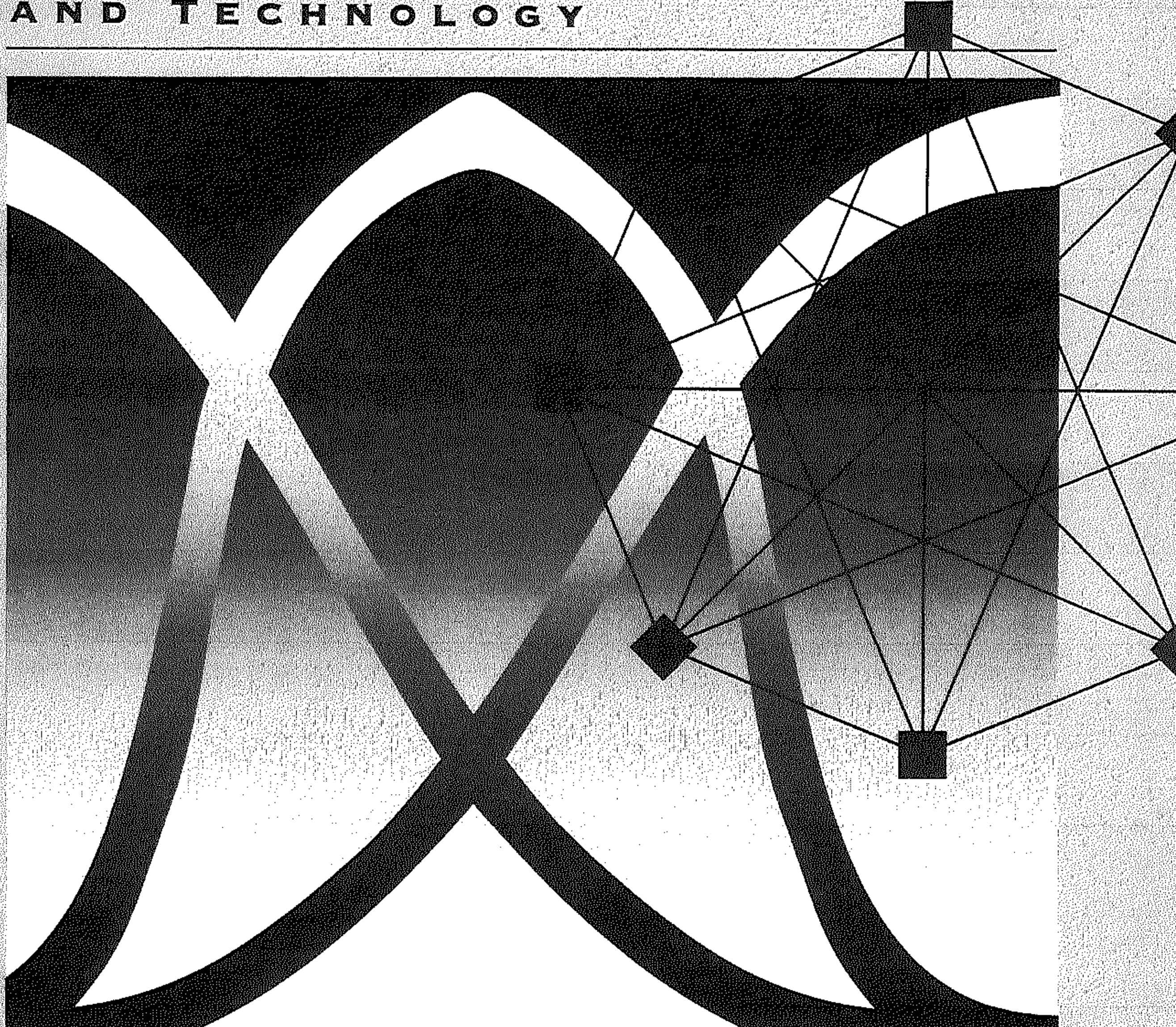
VOLUME 7 ISSUE 1 1999

JOURNAL OF INTELLIGENT & FUZZY SYSTEMS

APPLICATIONS IN ENGINEERING AND TECHNOLOGY



IOS Press
Ohmsha

Developing a general purpose intelligent control system for particle accelerators

William B. Klein and Robert T. Westervelt
Vista Control Systems Inc., Los Alamos, NM 87544, USA

George F. Luger
University of New Mexico, Albuquerque, NM 87131, USA

Tuning and controlling particle accelerators is time consuming and expensive. Inherently nonlinear, this control problem is one to which conventional methods have not satisfactorily been applied; the result is constant and expensive monitoring by human operators. In recent years, and with isolated successes, advanced information technologies such as expert systems and neural networks have been applied to the individual pieces of this problem. Most advanced information technology attempts are also very special purpose and built in a manner not at all generalizable to other accelerator installations. In this paper, we discuss preliminary results of our research combining various methodologies from the field of artificial intelligence into a general control system for accelerator tuning. We consider state space search and adaptive/learning algorithms including fuzzy logic, rule-based reasoning, neural networks, and genetic algorithms. We then propose a framework for applying these methods to a general purpose system for control. Finally, we discuss future plans for extending the system to include parallel distributed reasoning, an enhanced object structure, and additional heuristic control methods.

1. Problem overview

The goal of this project is to develop a flexible intelligent controller that can reduce the tuning time and the need for human intervention in control of a particle accelerator. We also wish to produce better and more stable tunes than those that are now achieved by human operators. This means that the controller will be able to maintain the proper tune with smaller beam deviations than are currently possible with human monitoring.

A number of approaches have been taken to automate accelerator control [2, 11, 13, 15, 17], with varying degrees of success. Most effort has been directed toward solving specific problems for a particular facility, and little effort has been directed toward developing more general solutions applicable to the diverse specifications and tasks of a number of different accelerators. Our goal is to produce a control system capable of handling common control tasks across many domains as well as a system which can abstract from specialized control code to more general methods.

Our research consists of two phases: First to build an accelerator control code capable of handling representative tasks which may be simulated through modeling codes. We view this first phase as an important proof of concept which is a prerequisite to testing against a real accelerator facility. Once a control concept has been established in simulation, the second phase of research can go forward. The second phase involves continued development of ideas and codes realized in phase one, as well as testing against an actual accelerator. This paper reports the status of our research after its first phase concluded with the production of a prototype for a general purpose accelerator controller. We end our paper with a summary of our plans for continuing the effort into its second phase, and going on-line with control at two accelerator sites.

The architectural framework for the controller consists of an expert system that guides specialized subcontrollers based on the present state of the system and current tuning goals. We have developed a number of realistic simulations to test the controller including steering and focusing tasks on a periodic line. We have examined several types of subcontrol tuning algorithms including backpropagating connectionist networks, fuzzy logic control, analytic rule systems, and genetic algorithms.

We introduce the problem of accelerator control by presenting two typical scenarios in beam transport:

Journal of Intelligent and Fuzzy Systems 7 (1999) 1–12
ISSN 1064-1246 / $8.00

steering and *focusing*. In our work to date, we have modeled a standard transport line including steering and focusing elements using the TRANSPORT beam modeling code [1]. The problems modeled included adverse tuning conditions such as noise (initial beam fluctuation) and component failure. We designed the controller using CLIPS [4], an expert system development tool developed by NASA, to analyze characteristics of transport line components and determine appropriate solution strategies depending on current beamline conditions. Because steering and focusing are complex tasks which can be partitioned into easier separate subproblems, they were initially solved independently. Because steering and focusing are not independent on a real beamline, once solutions were developed for each situation, we developed a combined steering/focusing solution. The combined solution used an iterative search to alternately readjust steering and then focusing, based upon the individual control methods determined for each.

In the remainder of this section we describe the accelerator tuning problem in more detail, including descriptions of the steering and focusing problems. We also describe the TRANSPORT simulation code for particle accelerators and introduce the Vsystem control methodology which provides the real-time supporting data for our controller. In Section 2 we present top level search and representation issues for controller design. In Sections 3 and 4 we present our solutions for individual control modules. Section 5 summarizes our current work and points to the future directions for our phase two efforts.

1.1. Accelerator simulation

We began by developing a computer model to simulate steering for prototyping of the intelligent controller. Steering is one of the initial tasks of a beamline tuner. We first considered the situation in which steering was controlled by two steering magnets (SMs) separated by some distance. Two beam position monitors (BPMs) downstream (in the sense of beamline flow) of the steerers monitored steering effects. This situation is depicted in Fig. 1.

The goal of steering is to adjust the current directed to the steerers so that the position indication from each of the BPMs corresponds to some desired offset (zero for an on-axis tune). Steering must take into account beamline alignment, electronic offset and drift, and downstream tuning requirements. In general, there is jitter in the initial beam coordinates, with some frequency caused by beam-source and mechanical and electrical variations. The accuracy of the solution is limited by this jitter as well as the resolution and noise inherent in both SMs and BPMs.

Beamline steering is a basic building block for most transport lines. Although apparently simple, with few exceptions, notably Nguyen et al. [13] and Himel et al. [6], steering control has not been successfully implemented as an automatic procedure. This lack of automation is due, in part, to the great number of parameters which can effect steering, including iterative procedural considerations, peculiarities of particular components, and imprecision in measuring devices. In practice, human operators spend a large portion of their time establishing and/or maintaining adequate steering.

The second basic element of beam transport is the periodic line for focusing. The line consists of alternating focusing and defocussing quadrupole lenses as shown in Fig. 2. These lenses produce a periodic variation in the beam envelope and provide a standard means for transporting a beam over a distance. Beam root mean square sizes are measured on profile monitors which directly measure intensity distribution. The profile monitors, often wire scanners, contain inherent inaccuracies due to beam fluctuation during measurement and component error. The relationship between quadrupole settings and the beam profile is nonlinear, making accurate real-time tuning difficult even for human operators.

1.2. TRANSPORT and the Vsystem software

To build an environment for testing our control algorithms we interfaced TRANSPORT [1], a standard accelerator modeling program, to Vsystem [3]. TRANSPORT is a well established tool used by accelerator physicists for system design. It is also used as a tool by expert operators for diagnosis and initial solution modeling. Vsystem is a commercial software product for developing control systems. Vsystem provides a distributed database and tools for accessing real-time data, as well as a graphical environment for display and control of database channels. The combination of Vsystem as the control interface and TRANSPORT as the accelerator model provided an automatic system for effectively simulating the real-time response of an

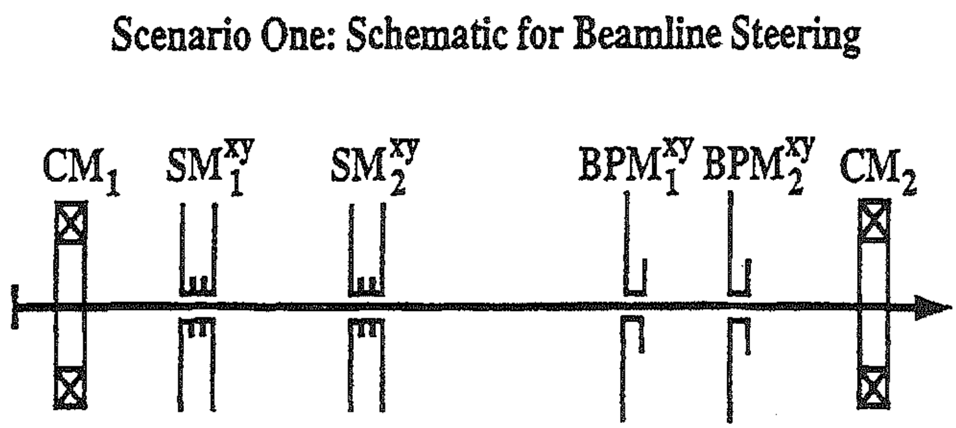


Fig. 1. Two steering magnets (SMs) direct the beam through the beam pipe. Two downstream beam position monitors (BPMs) monitor the results.

accelerator and produced a platform for generating realistic data for reasoning by the expert system.

We modified TRANSPORT by adding input types to relate TRANSPORT model elements to Vsystem database channels. We then modified TRANSPORT to automatically recalculate the simulation when input parameters change. We added noise and error effects to TRANSPORT data by filtering and varying the data as it was stored in the Vsystem database. Random gaussian noise was added to data signals for monitoring devices. Noise characteristics were configurable from the Vsystem database. Time dependent device behavior, including magnet ramp-up and delay was also included in the simulation.

2. Control design issues

We considered many control design issues during development of the control system including:

i) adaptive vs. non-adaptive control,
ii) optimal vs. "good enough" solutions,
iii) scalability,
iv) determination of failure conditions,
v) on-line and off-line learning, and
vi) stability in a heuristic control environment.

Our controller design addresses all of these issues with the possible exception of stability, which is still an open question. The design includes a top-level decision maker which determines appropriate control techniques according to the first five issues listed above. This top-level controller is able to choose from a variety of control techniques and substitute new ones as dictated by the system's states.

2.1. *Control of search*

Our design used an expert system at the top level for reasoning and control. An expert system is a computer program that can help solve complex problems using large bodies of facts and procedures gathered from human experts. These facts and rules are usually domain-specific knowledge gathered from actual problem solving experience. This knowledge is not always derived from equation-based constraints or foundational theorems and may include both theoretical and heuristic information. Expert systems use rules and facts to reason and make decisions, often based on imprecise and incomplete information. Most expert systems also have the ability to explain their reasoning and decisions [12].

Placing the expert system at the top level provided a controller capable of making top-level decisions about the control problem in a global context, without considering detailed issues; context specific subproblems were handled by lower level control modules. With direct access to the Vsystem control database, the expert system used all pertinent information to build a model for solving the system and to reason about specific components and more general tuning issues. This top-down approach reflects an expert's knowledge in

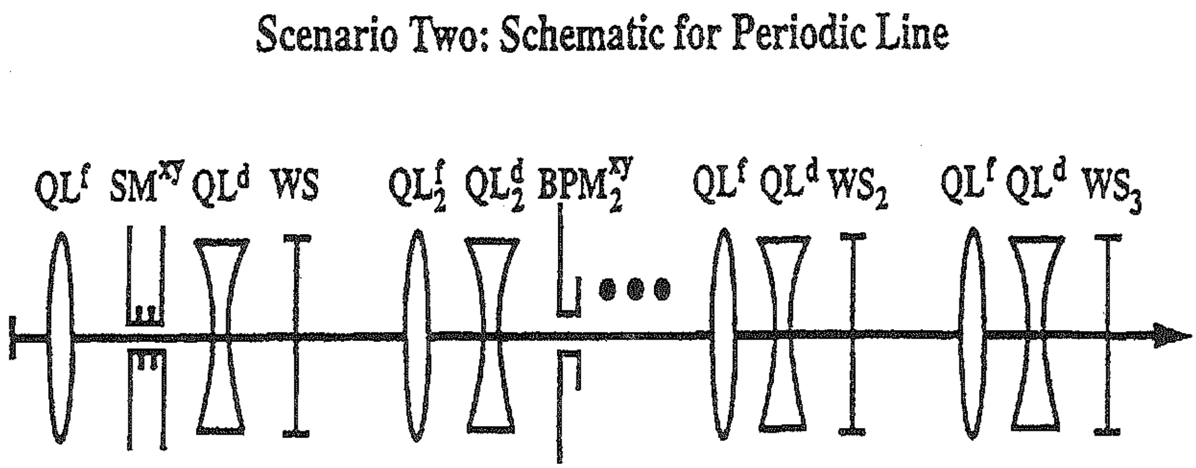


Fig. 2. Alternating focusing and defocussing quadrupole lenses (QLs) transport the beam along some distance. Wirescanners (WSs) monitor beam properties for focusing and SM and BPM components assist in steering.

a large system and provides a good framework for building proper knowledge representations.

As the first step in developing a knowledge intensive control solution, we selected an inferencing system that allowed easy prototyping and was appropriate to control problems. CLIPS, a forward chaining expert system shell developed at NASA [4], seemed appropriate because it allowed easy modification to production rules. Using CLIPS for prototyping kept the project focused on building knowledge structures rather than forcing us to design a complete inferencing system before we knew the full requirements and implications of the submodules for the final system. CLIPS was available as C source code that we could easily integrate with our own control code. CLIPS included an object system (COOL), which allowed mirroring of objects in C++ code as well as object oriented decision making. Using an object oriented design, we could make the system representation itself reflect the views of the human expert when reasoning about the control of the particle accelerator.

Top level control by an intelligent reasoning system assists partitioning both problem and solution spaces down into well defined, easy-to-reason-with subcomponents. We began by separating beamline components into groups by both functionality and control characteristics. By looking at the characteristics of each of the subcomponents, we developed partitions that imply certain types of solutions. Once the solution space has been well partitioned, an appropriate set of solution technologies is called to operate on those partitions. The top level reasoning system can focus on a particular partition and determine the best strategy for its solution with respect to the constraints of its neighbors.

2.2. The structure of the domain

With the expert system in control, representation of the accelerator was necessary for partitioning and reasoning about the components that make up the beamline. Creating an object representation of the system within CLIPS enabled us to place knowledge about a specific component within that component's representation while maintaining a separate knowledge base representing facts and rules describing the entire system. An object reasoning model allows appropriate encapsulation of knowledge with system objects, modularity of reasoning, and the possibility of distributed control.

We built objects for beamline components which represented the functionality of both physical and control characteristics. The simplest objects were readable components attached to a control database. These objects included static information about beamline placement and orientation, as well as methods for data collection during operation. BPM and wire scanner objects inherited most of their properties from readable component classes. SM and quadrupole magnet objects were derived from writeable component classes which included methods for updating magnet settings, determining component derivative relationships, and storing measured control information. Other objects included power supplies, beam sources, and current monitors.

CLIPS created beamline objects at run-time to form a model for reasoning about the beamline. The expert system queried the control database to determine what components were included in the beamline. It then built an object representation of the beamline both in CLIPS and C++. The expert system used the CLIPS object model for dynamically partitioning the control problem into solvable pieces and for reasoning about the beamline. The C++ objects were used for measuring and manipulating actual control data and for linking to the Vsystem database. Methods were included in CLIPS objects for automatically updating mirrored C++ objects.

3. A control methodology

In this section we outline a number of heuristic methods for control of the partitioned submodules of the accelerator. These control methods include neural (or connectionist) networks, fuzzy logic, genetic algorithms, as well as more traditional analytic methods. In using these heuristic methods we make certain basic assumptions about the control problem based on suggestions from Ross [14]:

i) Beamline behavior is observae and controllable. The control techniques used here rely on measurable state input and output variables.
ii) There exists a method for encapsulating knowledge about beamline control within the heuristic methods. This may come from neural network learning algorithms, a priori rule-based knowledge, or inherent knowledge encoded in the genetic algorithm population.
iii) One or more solutions exist. The set of control variables is sufficient to produce correct beamline behavior.
iv) A "good enough" solution is acceptable. We will identify a small error range within which all solutions are valid.
v) Optimality and stability may be shown through data flow analysis and empirical methods, rather than through formal proofs. Because many of these heuristics use inexact methods, formal proofs may be inappropriate if not impossible.

With the exception of v) above, it is apparent that each assumption must hold in *any* automatic control system. It is important to note that in some cases the "correct" solution may be to determine that the system is unstable or uncontrollable in its current configuration and shut down the beamline. Keeping these assumptions in mind, we adapted the following methods for controlling the accelerator system.

3.1. The use of neural network methodology

A neural (or connectionist) network is a group of individual processing elements, often divided into layers (or slabs). The neural network, having accepted sets of data through input nodes, passes the results of computations between layers and finally on to a set of output nodes. The individual processing elements, roughly analogous to biological neurons, combine data from multiple input paths and create an output using a transfer function. Neural processing elements can be combined into a variety of architectures and, along with associated training functions, can learn and recall non-linear functions and patterns. Neural networks have the additional benefits of being able to function in the presence of incomplete or noisy data. By their design they provide for the parallel processing of input values. Neural processing has been used previously for aspects of accelerator control, notably by Himel [6], Nguyen et al. [13].

We used a two layer backpropagation network to attempt to discover steering magnet (SM) and beam position monitor (BPM) relationships. We chose a backpropagation network because of the simplicity of the task (learning a linear relationship in the presence of noise) and the straightforward representation of the problem. We were able to directly map network input and output nodes to BPMs and SMs respectively. The network was then trained to recognize causal relationships between changes in beam position readings and magnet adjustments. A fully trained network was given desired beam position changes as input values. The network then produced magnet settings that would cause the appropriate adjustments. The network was used to solve steering problems by generating the beam position changes necessary to bring the beam to the desired tune (represented by the current beam position error). The network also produced SM adjustments that appropriately effected changes in the beam's direction.

There are two well known problems with backpropagation networks that affected our solutions: the development of a sufficient training set and ensuring adequate speed of convergence. Because we wanted the neural network to learn relationships on the running beamline and to adjust its weights accordingly, we attempted to train the network on a run-time data set. We allowed the system to make random adjustments to magnets on the beamline model and then record the resulting BPM changes. This process produced a real-time training instance for the network. We then passed the BPM changes through the network to generate a set of predicted magnet adjustments. The system calculated the difference between predicted and actual magnet adjustments and backpropagated this error through the system.

Although the neural network was able to learn SM/BPM relationships using limited training cycles in some basic cases, it was not able to converge in more complex cases involving many SMs and BPMs. The network failed in these cases for three important reasons. First, there was inadequate training data. Because the network gathered real-time data about the current state of the system, it could not produce training data fast enough to evaluate SM/BPM changes in the system. Second, while the network could improve its performance by continuing to take samples of the system and produce more training instances, it took too much online beam time to adequately train the network.

Even if a training set were available, the network suffered from a third problem. By generating training instances through random SM changes and attempting to learn (reverse) causal relationships, the network was not directed toward specific solution methods for adjusting the BPMs. Potentially, a large number of SM adjustments could produce the same BPM effects. The network did not converge because the random training data often included conflicting SM adjustment examples with very similar BPM results.

As a result of our experience in using neural technology, we rejected the general use of neural networks for direct system control. Instead, we view them as useful in their more traditional supporting roles of subsystem identification, pattern classification, and feature recognition. In these roles neural networks still prove useful for diagnostic tasks, beam structure recognition, and reference model control.

3.2. Methods for analytic control

The analytic technique for steering control relies on beamline behavior consistent with a simple linear model. After the expert system determines a set of components that make up a steering section, it measures the derivative between SM power source currents and BPM readings. The expert system calculates the derivative empirically by adjusting power to the SMs in the section and recording resulting changes in BPM readings. The expert system then builds an appropriate system of equations and solves them using gaussian elimination (reduction). The analytic method makes no attempt to filter noise or eliminate component errors. In general, this method provides an accurate baseline solution given large signal-to-noise ratio and properly functioning beamline components. Once derivatives are calculated for each SM/BPM pair, subsequent adjustments due to beam drift do not require derivative re-evaluation. The system keeps track of solution accuracy and adjusts derivative measures only when they no longer match beamline behavior.

3.3 Methods based on fuzzy logic

We cannot expect a purely analytic solution to adequately tune a beamline in most cases, especially during initial startup. One of the conditions that causes difficulty for tuning is beam fluctuation (jitter). Fuzzy logic is used in the beamline controller for reasoning about real-valued data in the presence of noise or in situations where analytic methods have failed. Fuzzy logic attempts to categorize real data sets with ambiguous boundaries. An example is to classify real values into categories such as *near_zero*, *small_positive*, and *small_negative*. A single value can belong to more than one of the categories with some degree of membership.

We may consider the data we receive from beamline measuring devices as ambiguous, or imprecise, because data measurement involves some error from an unknown distribution and possibly from an unknown source. We can capture a human operator's reasoning about this imprecise data by specifying linguistic variables comparable to the fuzzy sets which match the operator's (implicit) fuzzy categories.

The fuzzy logic steering solution used fuzzy rules about BPM relationships to follow a hill-climbing algorithm towards a good solution. An example of a rule from the fuzzy steering system follows (in CLIPS format):

```
(defrule r1
      (bpm_ratio near_zero)
      (bpm1_reading large_positive)
=>
      (assert (move_sm1 large_negative))).
```

This rule instructs the system to modify the current supplied to the first SM by a large negative amount whenever the ratio between two subsequent BPM readings are near zero and when the first BPM's reading is large and positive

Not only do fuzzy rules allow expert systems to reason about real-valued data without crisp data boundaries, they also allow reasoning about how data will be measured and evaluated. In the above example, the expert system could modify the meaning of *large_positive* depending upon the context in which the term is to be interpreted. This allows the expert system to change the membership function associated with *large_positive* depending on the specific problem being solved, the accuracy required, and the current state of the system.

The fuzzy logic solver did a good job of quickly moving to an approximately correct solution, but tended to oscillate around a very accurate tune. The accuracy of the fuzzy solution depended greatly on the quality of knowledge placed in the system. For example, a pure hill climbing fuzzy system that only attempted to minimize BPM error tended to find local minima. When the knowledge base was modified to evaluate BPM ratios and isolate specific magnets for adjustment, the fuzzy solution tended to find better solutions quickly. Knowledge about how and when to adjust membership functions for fuzzy sets is equally important for dampening oscillations near good solutions. Mechanisms for determining evaluation criteria and adjusting membership functions according to context are essential parts of the controller.

3.4. Methods based on genetic algorithms

The genetic algorithm offers an appropriate heuristic for focusing control because it can search large solution spaces in non-linear domains. A genetic algorithm uses a parallel search on a number of trial solutions and uses a measure of the fitness of the resulting state of the system for each trial along with solution combination functions to generate a set of new trial solutions [12, Chapter 15]. This approach supports problem solving insituations where the system is not behaving in a manner consistent with the knowledge base. The genetic algorithm is particularly useful when the controller must function using incomplete information or when the system behaves abnormally due to component failure or other unpredictable situations.

We implemented the focusing genetic algorithm using genetic operators which modified magnet strengths according to a fuzzy pattern. Fuzzy patterns eliminate the need for a priori determination of magnet adjustment strengths and patterns. The focusing genetic algorithm was built with the realization that wire scanners cannot deliver real-time continuous feedback. Trial solutions for the genetic algorithm were evaluated by actual testing on the simulated (or real) beamline. Since typical solution patterns can be determined for focusing, we used a special genetic operator to search the solution population for unwanted solution patterns (as determined by the expert system) and replaced them with more appropriate solutions. Unwanted solution patterns may be specific patterns which have performed poorly in the past, patterns which have been explicitly prohibited through the knowledge base, or patterns which may be predicted through diagnostics or other means. For example:

Population member:	{−0.78, 1.40, −1.32}
is evaluated as:	{small_negative, small_positive, small_negative}
and replaced with:	{small_positive, small_negative, small_positive}
which defuzzifies to:	{0.68, −0.91, 1.04}

In this example, the genetic algorithm is searching for settings for three quadrupole magnets arranged as a triplet. The expert system has information in its knowledge base that triplets typically use a positive, negative, positive current pattern. The expert system has conveyed that knowledge to the genetic algorithm, along with knowledge about replacing negative instances of patterns with positive ones. The genetic algorithm is then able to replace the unwanted pattern $\{- + -\}$ with the pattern $\{+ - +\}$.

3.4.1. Fuzzy pattern matching

Fuzzy pattern matching and replacement guides the genetic algorithm toward certain solutions and away from others according to knowledge about typical solutions contained in the expert system. The algorithm can still perform global search over the solution space and converge on a solution differing from suggested "good" patterns. It does this by only replacing the least desirable patterns from the population during any one generation. The least desirable patterns may be determined a priori, based on system constraints, or at run-time, based on current system state. We found that the fuzzy pattern matching solution focused the simulated periodic line in fewer than 100 trials and to a greater than expected accuracy.

Fuzzy pattern matching in the genetic algorithm greatly enhances search speed by promoting certain types of solutions that are known to be good solutions in a specific domain. Knowledge about good solutions can come from the expert system evaluating trends in the population or from a priori expert knowledge provided by an accelerator physicist. As with a human expert, this pattern directed solver uses knowledge about typical solutions to conduct a search within the known solution space to fine tune a beam. The fuzzy genetic algorithm is well suited for finding a specific solution instance, once the general form of a solution has been determined.

4. A synthesis of control modules

By incorporating the solution methods of Section 3 into a single system, we have developed a phase one prototype for an integrated problem solver and used it to correct many basic beamline tuning problems. By modifying existing solution algorithms or adding new solution strategies, we can enhance the quality and speed of the system within the current framework and, indeed, generalize our solution to other particle beam accelerators.

By placing the expert system as the top-level decision maker for the controller, we use expert knowledge to break the control problem into solvable units and then determine an appropriate solution strategy for various beamline problems. Not only can the expert system separate and distribute control tasks, it can also determine which problems to solve, how to address each problem, and the best order for handling the system constraints. This top-down approach allows the system to develop solutions based on general knowledge of the entire beamline, as well as to make individual adjustments as isolated units, when necessary, by applying methods appropriate to each unit.

This modular decomposition of complex problems into multiple interacting subcomponents is central to our approach. The object oriented methodology provides data structures that "wrap" the submodules in a module hierarchy, where each module contains knowledge (procedures) describing its functionality, as well as sets of methods for cooperating with other modules. Together the interacting modules make up the larger system. This approach to problem solving is called by the artificial intelligence research community a *solution strategy based on the interactions of autonomous intelligent agents* and has been used in a number of different application areas including electricity transport management [8] and building environment control [7].

4.1. The results and planned extensions of our phase one effort

During the evaluation of our prototype system, we tested it against steering and focusing models using the TRANSPORT, Vsystem, and noise filtering codes. Through a series of informal tests we also evaluated the tune time and accuracy against the skills of an expert beam physicist. Our controller repeatedly out performed the human expert in both the speed and accuracy of the tune. Our controller was able to find solutions to difficult focusing problems which were not solvable by the beam physicist without the use of a modeling code. Our tests indicated that many of the techniques described in this paper are indeed valuable for accelerator control, and the success of our controller running in simulation has led to the offer of beam time for testing against real accelerator facilities at Brookhaven National Laboratory.

Using knowledge gained during the first implementation of our research, we have prepared a plan for a complete intelligent control system for accelerators. In this planned system, an expert system coordinates the activities of a set of independent processes, which in turn control the smaller subsystems of the accelerator. The expert system manages the tuning process by identifying and configuring subgoals based on an overall goal for the accelerator. These subgoals are then either subdivided further or assigned a suitable solution strategy based on the goal and the current operational state of the accelerator. An expert system equipped with a "toolbox" of control methods can overcome limitations of any one control method by substituting a specific control strategy based on a particular subsystem goal. Figure 3 illustrates the architecture of this design.

Although the expert system with general knowledge of the control domain exists at the top level for coordination and control of beamline subcontrollers, smaller domain-specific rule sets exist throughout the object oriented component module hierarchy. Distributing knowledge throughout the system has a number of advantages:

i) Rule sets are typically smaller; large rule sets indicate the need for breaking down the problem into smaller components.
ii) Knowledge resides at the appropriate level in the system, so control objects can make domain specific decisions which are then coordinated by higher level control objects.
iii) Reasoning is faster; the conflict set for any one rule base is smaller, and independent activations may be fired simultaneously in the distributed environment.

We are developing a new expert system shell with enhanced capabilities not provided by CLIPS. The current CLIPS system will be replaced by an expert system shell designed to include four major features:

i) the ability to switch dynamically between data and goal driven reasoning,
ii) structures supporting the parallel and distributed reasoning of modules,
iii) an object system with knowledge representation capability, and
iv) implicit linkage between reasoning objects, beamline components, and Vsystem database channels.

During initial startup of the beamline when little is known about system state, goal-driven (backward chaining) reasoning is inappropriate. Data-driven (forward chaining) reasoning is then used to reason about system behavior and to determine a specific goal which is used to tune the beamline. Once a particular system goal is identified, the control switches dynamically to goal-driven reasoning and searches that part of the solution space pertaining directly to those beamline components useful in accomplishing the current goal. When all system goals are satisfied, the system again returns to data-driven reasoning to check for possible component failures and to determine new tuning goals.

The concept of object models for accelerator systems is a methodology gaining a large following in the accelerator control community. Work has been done at numerous sites to develop an object framework for describing accelerator control applications [16]. Our object system is designed specifically to provide a control abstraction linkage between the intelligent controller and the actual physical system.

To build a framework capable of operation at multiple accelerator installations, we are enhancing the object system with the capability for greater problem decomposition. This object model codifies the behavior and effects of each component type, allowing imbedded low-level reasoning within objects. Component specific knowledge is contained within the object model itself rather than requiring that rule/fact relations be described explicitly in the expert system rule base. Parallel distributed reasoning is possible in the system through distributed rule sets in low-level objects along with communication mechanisms which allow for their coordination by higher-level objects.

Many of the control problems we encounter require gradual refinement of fuzzy membership functions related to linguistic variables in the rule set. This refinement relates to the context dependent nature of fuzzy membership functions and the ability to reuse fuzzy rules in both coarse and fine grain solutions. Our current solution relies on the a priori description of a number of membership functions with increasingly smaller values. The expert system rule set guides the proper adjustment of fuzzy membership functions depending on the current state of the solution. For instance, during an initial startup tune where the beam may be many millimeters off axis, the value *large_positive* includes values from 5 to 10 mm. During a steady-state tune, where only minute adjustments are necessary, the value of *large_positive* includes values from 0.1 to 0.05 mm. The same rule set for steering adjustment is used in both cases.

5. Summary and conclusions

In this paper, we have stressed the importance of a knowledge intensive approach to complex problem solving. By placing an expert reasoner as the top-level

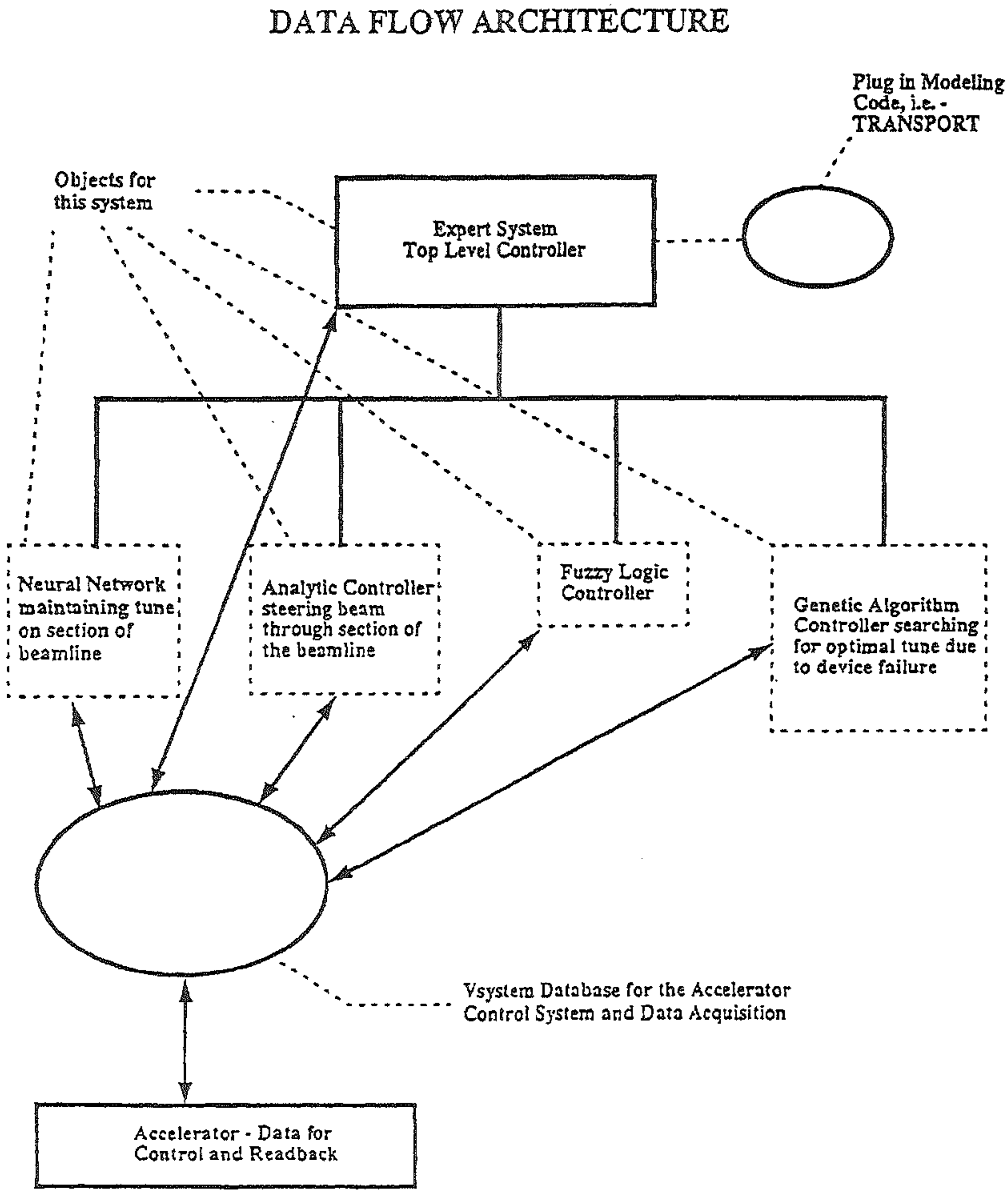


Fig. 3. This data flow diagram for the proposed architecture depicts the expert system at the top level directing different heuristic algorithms for beamline tuning. The controller represents the beamline using structured objects and accesses control parameters through the Vsystem database.

controller and decision maker, the system can develop representations for problem solving and make expert level decisions about how solutions should be developed. The expert system is able to use general knowledge about the particle accelerator to produce appropriate strategies for either an entire problem domain, such as steering, or for specific subsystems, such as for a single periodic cell. This knowledge intensive approach allows breakdown of the control problem into solvable subcomponents. Such problem deconstruction works well with object-oriented solutions where expert knowledge is encapsulated within the objects. The expert system uses global knowledge to make sequential top-level decisions, such as "steer the upstream section before proceeding to any downstream sections." The system uses the distributed intelligence resident in multiple objects to make parallel decisions, such as "match all beam position monitor objects with readings outside error tolerance, then solve."

A major benefit of our design is the ability to use expert level reasoning to implement multiple heuristic control algorithms. The reasoner makes knowledge-based decisions about how to separate the tuning problem into solvable subproblems. Each subproblem is then examined by another knowledge-based problem solver or assigned to a control algorithm. The heuristic algorithms we employ in our system include neural networks, fuzzy logic methods, and genetic algorithms. We also plan to integrate case-based reasoning capabilities [10] for development of a case library for reuse of successful control solutions. The expert reasoner determines how a problem may best be solved by these algorithms and creates objects for representing, computing, and reasoning about the relevant beamline components.

The continuation of our efforts will result in a robust system designed with many complementary control technologies. First pass solutions come from model-based methods using TRANSPORT, or from knowledge based rule modules adapted from beam control experts. Multiple solution methods are available for use in cases where model-based control strategies fail. The controller will examine beamline component characteristics, evaluate system parameters, build object representations of subsystems, and dispatch control strategies based on goals developed for each subsystem. These subsystems will be solved either sequentially or in parallel depending on over all system state as determined by the expert system.

Work is continuing in the design of a new reasoning paradigm to allow switching between data and goal driven problem solving. Neural network solution methods are being reevaluated to determine whether more advanced methods, including multi-network modeling codes and fixed input learning algorithms, are appropriate for on-line adaptation and control. We are enhancing the knowledge base to include more detailed knowledge related to specific control scenarios including both detailed accelerator theory as well as site-specific information. We are enhancing the object system to include embedded knowledge representation for beam physics and linkage to Vsystem. This Physical Access Layer (PAL) is an object-oriented construct to handle low level data gathering, filtering, communication, and control information.

Finally, with the demonstrated successes of our controller running in the simulated TRANSPORT environment, as described earlier in this paper, we have been given beam time at Brookhaven National Laboratory's Accelerator Test Facility, where our system was tested in late 1996. Our system had further tests at the Argonne accelerator facility during 1997. The results of these tests are reported in [9].

Acknowledgement

This work was supported by a grant from the Department of Energy under the SBIR program (Grant #DE FG05-94ER81897). We would like to thank Andy Jason of AOT-1 at Los Alamos National Laboratory for his invaluable assistance in developing control scenarios and for his careful explanations of the underlying physics concepts involved.

References

[1] K. Brown, F. Rothacker, D.C. Carey and C. Iselin, *TRANSPORT: A Computer Program for Designing Charge Particle Beam Transport Systems*, SLAC-PUB-91, Rev. 2, 1977.

[2] S. Clearwater and W. Cleland, A Real-time expert system for trigger logic-logic monitoring, in: *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, Vancouver, B.C. 1989.

[3] P. Clout, The status of Vsystem, in: *Proceedings of the Third International Conference on Accelerator and Large Experimental Physics Control Systems*, Germany, 1993.

[4] J. Giarratano and G. Riley, *Expert Systems: Principles and Programming*, Ch. 7, PWS-KENT Publishing Company, Boston, MA, 1989.

[5] C.J. Harris, C.G. Moore and M. Brown, *INTELLIGENT CONTROL, Aspects of Fuzzy Logic and Neural Nets*, Scientific Series in Robotics and Automated Systems, Vol. 6, World Scientific Publishing Co. Ltd., New Jersey, 1993.

[6] T. Himel, S. Allison, P. Grossberg, L. Hendrickson, R. Sass and H. Shoaee, Adaptive cascaded beam-based feedback at the SLC, in: *Proceedings of the 1993 IEEE Particle Accelerator Conference*, Washington, DC, 1993.

[7] B. Huberman and S. Clearwater, A multi-agent system for controlling building environments, in: *Proceedings of the First International Conference on Multi-Agent Systems*, MIT Press, San Francisco, CA, 1995.

[8] N.R. Jennings, J.M. Corera and I. Laresgoiti, Developing industrial multi-agent systems, in: *Proceedings of the First International Conference on Multi-Agent Systems*, MIT Press, San Francisco, CA, 1995.

[9] W.B. Klein, C.R. Stern, G.F. Luger and E.T. Olsson, An intelligent control architecture for accelerator beamline tuning, in: *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, AAAI/MIT Press, Cambridge, MA, 1997.

[10] J.L. Kolodner, *Case Based Reasoning*, Morgan Kaufmann, San Mateo, CA, 1993.

[11] M. Lee and S. Clearwater, Generic orbit and lattice debugger, in: *Proceedings of the Model Based Accelerator Controls Workshop*, Brookhaven National Laboratory, New York, 1987.

[12] G.F. Luger and W.A. Stubblefield, *Artificial Intelligence: Structures and Strateggies for Complex Problem Solving*, Addison Wesley, Reading, MA, 1998.

[13] D. Nguyen, M. Lee, R. Sass and H. Shoaee, *Accelerator and Feedback Control Simulation Using Neural Networks*, SLAC-PUB-5503, 1991.

[14] T. Ross, *Fuzzy Logic with Engineering Applications*, McGraw-Hill, Inc., New York, NY, 1995.

[15] D. Schultz, The development of an expert system to tune a beam line, in: *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, Vancouver, BC, 1977.

[16] J.F. Skelly, Object-oriented programming techniques for the AGS booster, in: *Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems*, Tsukuba, Japan, 1991.

[17] R.T. Westervelt, W.B. Klein and G.F. Luger, Framework for a general purpose control system for particle accelerators, in: *Proceedings of the IEEE Particle Accelerator Conference*, Dallas, TX, 1995.