3

PHASE II FINAL REPORT

DOE SBIR DE-FG05-94ER81897

DOE Document: DOE-OAK/SBIR

Project Title: A Framework for a General Purpose Intelligent Control System for Particle Accelerators

Project Director: Dr. Carl Stern

Small Business: Vista Control Systems, Inc.

Staffing: Dr. Robert Westervelt, Dr. William Klein, Dr. Michael Kroupa, Eric Olsson, Rick Rothrock

Abstract

Vista Control Systems, Inc. has developed a portable system for intelligent accelerator control. Our design is general in scope and is thus configurable to a wide range of accelerator facilities and control problems. The control system employs a multi-layer organization in which knowledge-based decision making is used to dynamically configure lower level optimization and control algorithms. An object-oriented physical access layer (PAL), supported by the Vsystem control database, allows abstraction from the lower level details of hardware manipulation, signal processing, and synchronization. A teleo-reactive sequencing mechanism is used to coordinate control actions. This architecture has been implemented in two deployed systems, one for automated accelerator tuning at the Brookhaven ATF and one for Argonne ATLAS. Several weeks of field testing at these two facilities have yielded very promising results. We describe the technical details of this architecture as well as our experiences in field testing in this report.

1. Historical overview.

The goal of this project was to design and prototype a portable intelligent control system for particle accelerators. The design was required to be general and to involve the assembly of general components, in order to expedite porting of the control system to new facilities. The design was intended to support robust adaptive behavior, in order to address weakly modeled systems, and to replace the role of human operators in performing the tuning and adaptations necessary to realize system objectives in everyday operations.

The initial strategy for achieving these goals was to incorporate "intelligent" and "soft computing" technologies, expert systems, fuzzy logic, neural networks, genetic algorithms, etc. The challenge was to incorporate these technologies into a principled and systematic design that combines both conventional and intelligent control methods. The design that emerged from Phase I research and that was developed in Phase II was hierarchical, distributed, and object-oriented.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Hierarchical organization supports adaptive and supervisory control and coordination between controllers. The system employs a real time data bus, Vsystem, to communicate with hardware. An object-oriented abstraction layer, the Physical Access Layer (PAL), was developed to operate on top of the data bus, handling low level details of hardware data access and hardware control. The PAL provides abstract symbolically encoded views of the hardware to controllers, allowing controllers to be programmed at a higher, more hardware independent level.

At the next level, knowledge-based controllers perform control actions either directly, by sending messages to the PAL, or indirectly, by calling the services of general purpose control modules. The general control modules encode a variety of conventional and "soft" control algorithms, including PID, hill-climbing, fuzzy logic, neural networks, genetic algorithms, etc. The algorithms at the core of these modules are implemented in as general a form as possible, to create general purpose tools.

The general control modules are then assembled and configured into larger units of functionality. Knowledge-based controllers, using rule-based decision making, apply slow but real time supervisory control and occasionally local system reconfiguration. A new sequencing technology, called *teleo-reactive control* is employed to support dynamic assembly of control actions into adaptive sequences or plans for achieving more complex control tasks.

A Phase II prototype was initially developed for tuning the ATF beamline at Brookhaven. It was quickly ported to handle tuning operations for the PII section of the ATLAS beamline at Argonne. These prototypes were gradually refined and extended over the duration of the Phase II period. In the course of repeated visits to both facilities, the prototypes matured in design, capability, and effectiveness.

We describe this evolution of design, as well as our experiences in testing and evaluation, in Sections 4 and 5 of this report. First, however, we begin with a description of the challenges and opportunities for improvement in the area of accelerator tuning and the project's technical objectives with respect to those challenges.

2. The nature of the accelerator control problem.

Tuning particle accelerators is time consuming and expensive. It involves optimization over a large parameter space, with complex interactions between parameters. Ideally models should be used to derive control parameter settings for components of a beamline. However, in most instances, models are not accurate enough to generate optimal control settings, and further human optimization is required. Automatic control methods have not generally been successful in this domain, and the result is the need for time consuming startup tuning, monitoring, and retuning of systems by human operators.

Figure 1. A typical accelerator beamline.

Figure 1 shows a typical linear accelerator beamline, with an accelerator section (a series of klystrons) and a variety of tuning and diagnostic components. These include trim magnets for steering, quadrupole magnets for focusing, Faraday cups and stripline detectors for measuring current, and profile monitors for measuring beam size and beam position. These components are placed along the beamline by design to produce particular effects at special intermediate locations and to contribute to the achievement of operational objectives at the final target locations.

Unfortunately, real systems rarely work as they are designed. Problems arise from imperfect beam production, remnant magnetic fields, poorly modeled beam behavior, misplaced or flawed control elements, and changes to the design or use of the facility after it has been built. Beamline designers consider these problems and build diagnostic components into the beamlines. Profile monitors and current detectors are used to measure beam parameters throughout the line to provide information for verifying or correcting beam characteristics. Even so, imperfect detectors, system errors, and noise due to various effects cause beamline control to be challenging.

Given these challenges, it is hard to imagine any system capable of tuning a beamline to an acceptable measure. However, physicists and technicians with expertise in beamline tuning accomplish this task, to varying degrees, every day. They do this by using a variety of tools for measuring and learning the current beamline behavior, including adjusting control elements to modify the beam, and then testing their results and re-planning their actions. Through extensive knowledge engineering we have observed a number of important characteristics of the accelerator control process:

- Beam tuners combine analytic and theoretical knowledge with heuristic search and practical experience to produce good tunes.
- Humans know when to trust system data, and when to ignore noisy or incorrect readings.
- Expert beam tuners use many of the same search algorithms applied in traditional AI systems.

- Combined, these characteristics describe many of the attributes of an automated knowledge-based system for control, and suggest that such a system could perform as well or better than a human at the beamline tuning task.

Control systems have indeed been built to perform some of the tasks which human operators perform in beamline tuning. In recent years with isolated successes, advanced technologies such as expert systems, neural networks, and genetic algorithms have been applied to the individual pieces of this problem. However, until now, no general portable system for beamline tuning has been designed and built. This is the task that our Phase II work attempted to address.

3. Previous attempts at accelerator control.

An example of conventional accelerator control is a project by Himel et al. [1993] that used analytic methods for noise canceling at the Stanford Linear Collider. This project applied MIMO (Multiple Input Multiple Output) adaptive noise cancellers to seven beam-steering feedback loops operating on the same beam. This effort was successful at providing a supervisory filtering mechanism for a set of parallel tuning controllers. Neural networks have been applied to accelerator control for actual manipulation of control parameters as well as for simulation. Nguyen et al. [1991] applied single layer neural networks for simulating effects of steering magnets over a series of 16 beam position monitors. Howell et al. [1990] used neural networks for modeling and control of a negative-ion accelerator source, but with limited success.

The SETUP program developed at CERN [Bouche 1995] is representative of efforts to apply AI techniques to small subsystems. SETUP is only used for pre-control equipment setup. The program uses an object-oriented description language for representing control actions. The reasoning system searches the oriented graph defined by an object description to make decisions about equipment setup without human assistance. The program provides a good example of using object models for control decision making. It does not attempt to perform real time control or use on-line feedback from the system.

The ZEUS project at Deutsches Elektronen Synchrotron, Germany [Behrens et al. 1996] is an effort toward automating a substantial portion of an accelerator facility. The ZEUS expert system, ZEX, is a blackboard-based architecture designed to add human experience for supervisory control. ZEUS and ZEX work together to provide slow control, data acquisition, data quality monitoring, and run control. The reasoning architecture is a forward chaining production system manipulating a complex hierarchy of control objects. Data is gathered and symbolized using syntactic pattern recognition at lower levels by clustering observed phenomena. Knowledge sources in the blackboard attempt to recognize control state over posted patterns and contribute to a global control solution. While this project constitutes a significant achievement in accelerator control automation, it is a very special purpose project requiring its own crew of experts.

The ABLE and GOLD systems developed by Clearwater and Lee [Lee & Clearwater 1987] were prototypes for real-time control of an entire beamline. ABLE used simple rule-based reasoning to perform tests and directly manipulate a beamline simulation to correct beam transport errors. ABLE was successful on a number of simple simulations, but has never been tested as a general solution or on a real accelerator.

Other attempts at intelligent control for accelerators include the ISIS tune advisor [Schultz et al. 1990], the LAMPF Beam Loss Expert [Clearwater et al. 1986], and a learning system based on RL4 [Clearwater et al. 1990]. The ISIS tune advisor and LAMPF Beam Loss Expert were both expert systems for indirect control (advising human operators) which were never implemented as general or real time control solutions. The learning system used knowledge-based induction for off-line learning of beam position monitor placement, but was not implemented as a general learning algorithm.

In summary, many attempts based on conventional control algorithms have been made to automate part or all of the accelerator control process. Other attempts that have included heuristic or "intelligent" approaches to the control problem have selected a small subset of control technologies, for instance model-based control or supervisory control using expert systems. This piecemeal approach to applying AI has been valuable for determining the usefulness of a number of approaches to accelerator control, but is not satisfactory as a solution for total automation of the process. We propose a technique that combines different methodologies and builds upon their strengths [Klein et al. 1997a].

4. Technical objectives and high level design issues.

We now describe in more detail our technical objectives for the Phase II work. These include:

- Portability and extensibility;
- A distributed object-oriented architecture;
- Adaptive knowledge-based control;
- Support for flexible human-machine interaction and multi-initiative control.

A fundamental design objective was to create a general machine-independent control automation architecture. In principle, this architecture would allow rapid implementation of automated tuning and diagnostic functionality on top of any new or existing accelerator control system. In addition, it would allow incremental development and integration, so that new functionality can be added seamlessly at later times.

A second objective was to realize this general extensible architecture in the form of a distributed framework, where control automation modules can be offloaded to one or more networked computers, so as to minimize the added computation burden on the primary control system computers. A related objective was to encapsulate control automation functionality in an object-oriented framework, where object methods are

accessed through an abstract communication protocol running on top of TCP-IP. This would allow transparent integration of software modules running on top of different hardware platforms and operating systems.

A third objective was to design a framework that provides intrinsic flexibility and adaptation. The chosen means for achieving this adaptive capability is to incorporate a knowledge-based framework at the higher levels of the control system. The intention is to design a control system capable of reasoning about system state and analyzing the cause of unsuccessful control actions in order to appropriately modify those actions or select alternative control strategies.

The final objective was to support interaction and collaboration with expert operators and physicists. This also requires use of a knowledge-based framework. The use of an explicit high level representation of goals, control strategies, and system state allows the system to share its belief state and rationale for control actions with its human collaborators. This allows human overseers to correct mistaken beliefs regarding system state, redirect misguided control actions, or provide alternative goals and strategies.

5. Design for a knowledge-based hierarchical distributed control architecture.

At the top level, our system is based upon a distributed hierarchical architecture designed to incorporate a wide variety of representations, both analytical and knowledge-based, into a single control framework. At the heart of the architecture is a group of knowledge-based *controllers*. These controllers are hierarchically organized in a structural/functional hybrid design [Klein 1997, Klein et al. 1997a, 1997b]. Controllers at the lowest levels are restricted to a local sphere of application, applying specific control functions to individual hardware components or small subsystems. At intermediate and higher levels, controllers incorporate a progressively more integrated and global view, coordinating the local control functions underneath them.

Controllers are responsible for making decisions about how control actions will be performed, what those actions will be, when they occur, and how their performance will be measured. Controllers are also responsible for reasoning about system state, diagnosing errors in control solutions, decomposing goals into tasks and actions, and initiating any necessary human interaction.

In addition to controllers that encode information about domain specific elements, algorithms, and experience, we also use *solvers* for component-based application of specific algorithms. Solvers encode algorithmic units that can be assembled (again in a hierarchical manner) for run-time construction of control algorithms.

Consider, for example, a search procedure for a search algorithm such as simple hill-climbing for optimizing beam transmission. The procedure is broken into three parts, a component for generating data points that must be measured during search, a component

Figure 2. Hierarchical control architecture.

for acquiring the actual measurements and applying an evaluation function, and a parent controller which coordinates actions of the two children in a way that produces hill-climbing. A different algorithm, such as Newton's method, can be constructed by merely substituting the parent controller for one that applies a different top-level procedure. Different child controllers may also be substituted when operating different sets of control elements or in cases where specific constraints (e.g., noise handling, speed, etc.) are important. [Klein et al. 1997b]

Supporting the control hierarchy is the Physical Access Layer (PAL). The PAL is an abstraction mechanism, linking the control hierarchy to an underlying data bus that provides data access and direct hardware control. The PAL defines a flexible interface to the data bus and to hardware that abstracts much of the low-level detail of data access and hardware manipulation. This simplifies the task of writing high level controllers, allowing the higher level controllers to be constructed in a way that is more general and portable. The full control system architecture is shown in Figure 2. We now describe each component of this architecture in more detail.

5.1 Knowledge-based controllers.

The primary function of controllers is goals-task decomposition. Controllers are given goals and tasks either by a human user or by task delegation from parent controllers. Given a goal or a task, a controller must select an action or control program for goal achievement or task realization. The matching of goals or tasks to actions depends on current conditions, e.g., on the tune-state of the beam, on the operational state of hardware. Thus controller knowledge consists of rules of the following form:

IF

Goal = *Goal_J* AND

Holds_In_Current_State(*Condition_K* AND *Condition_L* AND ...)

THEN

Next_Action = *Action_N*

The predicate *Holds_In_Current_State* defines the preconditions of *Action_N*.

Determining whether these preconditions hold may require new measurements, complex inference, or some combination of both.

Once an action or control program is selected, its progress is monitored with respect to the conditions it is expected to achieve. If the action is unsuccessful in achieving those conditions, then the controller that selected the action enters a diagnostic mode. It performs the measurements and reasoning necessary to verify that the action preconditions continue to hold. If they do not, then the controller attempts to determine why and if possible, to take corrective measures. These corrective measures may either consist in restoring the action's preconditions or in abandoning the action and selecting another action or control program based on different preconditions.

Knowledge-based action selection and action monitoring allow for the application of learning and replanning. When it is determined that an action or control program is unsuccessful in achieving its objective even when its preconditions hold, then the rule that triggered the action selection must be revised. The action's preconditions must be strengthened to ensure that the whenever those stronger preconditions are satisfied, the action will reliably achieve its intended objectives. Although we did not have time to implement it, a method for implementing action-selection learning is described by Benson [Benson 1995].

5.2 *Teleo-reactive sequencing.*

Sometimes a sequence of actions, called a control plan, is required to achieve a system goal. One often used method for constructing action sequences is scripting. In a script, actions are performed one after another in rigid order. The transition from one action to the next may be determined by a clock or by the monitoring of conditions that act as transition thresholds. This method, however, lacks the flexible action ordering characteristic of skilled operator tuning. We found from observing operators at Brookhaven and Argonne that the flexible alternation of tuning actions, e.g., steering and focusing actions that move back and forth between contiguous beamline sections, is essential for beamline tuning. Such iterative tune refinement is particularly necessary for beamlines with weak models and significant source variability and noise.

After a brief search for appropriate technology, we settled on an adaptive sequencing mechanism developed at Stanford for mobile robot control [Nilsson 1994]. By adding

several modifications, we adapted the teleo-reactive mechanism to accelerator control sequencing.

Teleo-reactive control occupies a region between feedback-based control and traditional discrete action planning. TR programs sequence the execution of actions that have been assembled into a certain kind of goal-related plan. Unlike traditional planning environments, no assumption is made that actions are discrete and uninterruptible and that an action's effects are completely predictable. On the contrary teleo-actions are typically durative and are executed as long as the action's preconditions hold and its goal has not yet been achieved (unless some other action closer to the final goal becomes activated.) A short sense-react cycle ensures that when the environment changes, the control action changes to fit the new state.

Teleo-reactive action sequences or programs are represented in a data structure called a TR tree. A TR tree can be described as a set of condition-action pairs:

$$\begin{array}{l} C_0 \rightarrow A_0 \\ C_1 \rightarrow A_1 \\ \dots \\ C_n \rightarrow A_n \end{array}$$

where C s are conditions and A s are the associated actions. C_0 is typically the top level goal of the tree and A_0 is the null action, i.e., do nothing if the goal is achieved. At each execution cycle the C is are evaluated from top to bottom until the first true condition is found. The associated action A_i is then performed. The evaluation cycle is then repeated at a frequency that simulates the reactivity of circuit based control.

Teleo-reactive programs are constructed so that an action performed at a lower level in the list will satisfy the preconditions for an action higher on the list, and that action will satisfy some higher condition, and so on, until some final goal state is reached. Since the same goal condition generally requires different actions from different starting conditions, teleo-reactive programs are typically tree structured (Figure 3). In this diagram a labeled circle represents some condition C_i and a labeled arrow a_i represents the knowledge that if action a_i 's precondition holds then a_i will make the condition C_i at the end of the arrow true. C_0 at the root of the tree represents the top-level goal.

Since an action is executed based on its list of valid conditions and not on the history of previous actions, a teleo-reactive program can exhibit a number of valuable properties. First, a program can begin its execution at an arbitrary point in the sequence. Actions for conditions which have already been satisfied can be skipped. If the action at some step in the program happens to satisfy multiple conditions higher up on the tree, the program can skip a series of actions. A program can adaptively backtrack when an action produces unintended side effects that impact previously achieved conditions, as is sometimes the

Figure 3. A tree-structured TR program.

case during steering and focusing operations. Finally, a program can adapt to new running conditions, such as sudden shifts in the beam source intensity or equipment failures, by dropping to the appropriate level in the tree.

We found that the teleo-reactive mechanism provides a useful framework for representing and implementing accelerator tuning algorithms for several reasons:

- i) Accelerator beams and their associated diagnostics are typically dynamic and noisy.
- ii) Achievement of tuning goals is often affected by stochastic processes such as RF breakdown or oscillations in the source.
- iii) Many of the actions used for tuning are of indefinite durations. This is often true of tweaking and optimization operations: they are typically done until specific criteria are met.
- iv) We have found TR programs to be an intuitive framework for encoding tuning plans acquired from accelerator physicists. Richard Pardo of Argonne, for example, encoded, with relatively little assistance, a TR plan for transverse tuning of the PII beam line.

5.2.1 Necessary modifications in moving teleo-reactive sequencing from robot to accelerator control.

In porting Nilsson's teleo-reactive architecture from its original robot control domain to our accelerator control architecture, a number of important modifications were necessary. Most important of these was the substitution of selective monitoring for the total environment monitoring that Nilsson's original architecture presupposes. In Nilsson's architecture there is no notion of the time and resource costs associated with observation, measurement, and data interpretation. Nilsson and Benson's Botworld implementations [Nilsson 1994, Benson 1995] assume that teleo-reactive sequencing is controlled by condition variables whose values can be directly observed without time or resource costs.

In the domain of accelerator control this assumption does not hold. In many cases, for observations to be made, monitors must be inserted or retracted. The beam must be steered or focused within certain limits for measurements returned by a beam profile or position monitor to be meaningful and accurate. Furthermore, the adaptive filters, e.g., neural networks, required to recognize certain beam properties, consume time and computing resources. For these reasons, condition evaluation in teleo-reactive sequencing must rely on selective monitoring, where the teleo-reactive execution environment includes the capability to make appropriate decisions regarding what needs to be monitored at any given point in time.

To meet these requirements we have developed the concept of a "monitor" that can be turned on or off. Decision-making regarding what monitors to turn on or off is integrated into the action operator representation. We have modified action operators so that it is possible to define pre- and post-processing actions that surround the main action of a rule. These pre- and post-processing actions can be used to turn on and off monitors as well as to call functions (in the case of post-processing actions) that interpret the data returned by an optimizer and set the value of a global condition variable, i.e., a variable that helps determine which action operator will execute next.

Another important modification to Nilsson's original architecture is attachment of confidence or utility measures to action operators. These numbers are set based on the action operators's recent performance history, that is, its success or failure rate. These numbers are used by the teleo-reactive sequencer to select between alternative action operators at branches in the TR tree where two or more actions are possible. Future version versions of our teleo-reactive execution environment may add more sophisticated performance tracking, where success rate is measured relative to parameter values or conditions that were true when the action began executing. This will allow the use of historical statistics to support *inductively learning* of optimal performance regions for each action.

5.2.2 Automatic construction of TR programs.

Another characteristic of TR programs that makes them attractive is that they can be constructed automatically by application of any one of several planning algorithms. While the TR programs that we used in our field tests were hand crafted from the tuning plans suggested by domain experts, we did in fact successfully build and test two automatic TR planners. This offers a number of interesting possibilities for future development. These include the possibility of implementing automatic plan repair in situations where the cause of plan failure has been clearly identified.

TR programs can be constructed through a planning algorithm that is a modification of common AI goal reduction planning algorithms. Starting from the top-level goal, the planner searches over actions whose effect includes achievement of the goal. The preconditions of the action generate a new set of subgoals, and the procedure recurses.

Termination is achieved when the preconditions of one of the nodes of the tree are satisfied by the current state of the environment. That is, the planning algorithm regresses from the top-level goal through goal reduction to the current state. Actions, of course, generally have side effects, and the planner must be careful to verify that an action at any level does not alter conditions that are required as preconditions of actions at a higher level. Goal reduction must consequently be combined with constraint satisfaction, where a variety of action reordering strategies are employed to eliminate constraint violations.

TR tree planning algorithms typically build plans whose leaf nodes are satisfied by the current state of the environment. They do not build *complete* plans, that is, plans that can start from any possible environmental state, because such plans would generally be too large to efficiently store and execute. This is an important point because sometimes an unexpected environmental event can shift the world to a state in which no action preconditions in a TR program are satisfied. In that case, online replanning would provide an important method for addressing the current problem situation. By fleshing out an incomplete program in response to failure events, such an online planning mechanism would provide an experience-based method for incrementally improving the generality and robustness of control programs.

5.3 Solvers.

Solvers are reusable components that can be configured by controllers to apply low-level, well defined algorithms to the control process. Solvers encode procedures that can be assembled (again in a hierarchical manner) for run-time construction of control algorithms.

Figure 4. A typical solver configuration.

Solvers in our architecture encode the functionality of many of the soft computing algorithms used to achieve intelligent adaptive control. We have implemented neural network, genetic algorithm, gradient descent, and fuzzy logic solvers.

Figure 4 shows a typical solver-based procedure for applying a search algorithm, in this case simple hill-climbing. The procedure is broken into three parts, a solver for generating the data point settings that must be measured during search, a solver for acquiring and processing the measured data using appropriate readback elements in the domain, and a parent solver which coordinates actions of the two children in a way that performs a hill-climbing search. A different algorithm, Newton's method for example, can be constructed merely by substituting a different parent controller implementing the search pattern in Newton's method. Different controllers may also be substituted in cases where specific constraints (e.g., noise handling, speed, etc.) are important.

5.4 The PAL.

Because we use a symbolic representation for reasoning about the control system, raw data is rarely appropriate for direct manipulation by controllers. The same is true in reverse; a low-level interface for manipulation of control elements is usually inappropriate. For this reason we have developed an object-oriented Physical Access Layer (PAL) as an abstraction mechanism between controllers and the underlying control data bus. This provides a number of important advantages:

The PAL provides a mechanism for hiding unimportant implementation details about hardware and provides a uniform interface for control access. Resource conflicts can be initially handled at a low level and, once identified, mediated at the controller level. Controllers can pass filtering instructions to the PAL to allow pre-processing of data into a representation expected by a controller. This can happen, for example, by giving the PAL fuzzy sets for classifying data, or passing a neural network encoding to the PAL. The system is highly portable. By abstracting underlying control elements, control algorithms can be written in a generic manner. The same set of controllers can be used at multiple accelerator facilities by exchanging the PAL.

The PAL is composed of a number of Physical Layer Objects (PLOs) which are representations of a control or diagnostic element or collection of elements. These objects can be as simple as single magnets, or as complex as non-linear tuning knobs that manipulate a series of magnets. PLOs communicate with hardware indirectly through Vsystem, a high-speed software data bus [Clout 1993]. PLOs can be organized in a hierarchical fashion and operate in parallel, much like controllers. The PAL provides PLOs access to a library of tools for representing and filtering data, algorithms for noise handling, pattern recognition, and feature extraction. Figure 5 illustrates the design of the PAL.

Figure 5. The Physical Access Layer (PAL)

The PAL does more than provide a high-level interface to the underlying software control system. The PAL also performs low-level control over groups of components that together represent a control or measurement element. For instance, at Brookhaven National Laboratory's Accelerator Test Facility (ATF), beam measurements are usually taken through profile monitors which consist of phosphor screens that emit light when struck by electrons. The light is recorded by video cameras and the images from those cameras are recorded by a video frame grabber. The PAL hides the process of capturing beam characteristics from these devices and only exposes important features of the process, like conflicts in use of the frame grabber, or position information from the monitors.

6. Deployment and testing.

We carried out experiments at two accelerator facilities: Brookhaven National Laboratory's Accelerator Test Facility (ATF), and Argonne National Laboratory's Advanced Tandem Linear Accelerator Site (ATLAS). The experiments were designed to address common beam line tuning tasks that are currently performed strictly through human interaction. These experiments vary in their difficulty, but each demonstrates some feature of the control architecture. Experiments show: knowledge-based control, use of solvers for low-level control, integration of knowledge-based and analytic methods, component-based programming, hierarchical organization of a control problem, goal/task decomposition and delegation, and the overall portability of the architecture and pattern recognition components.

The Brookhaven ATF consists of a beam source, a linear accelerator, a transport line, and three experimental beam lines. Electrons are generated from a pulsed laser striking a cathode and are initially accelerated by an RF device known as a klystron. This beam source generates short bunches of electrons at a rate of 1.5, 3, or 6 Hz. Electrons travel from the source through a solenoid magnet and then a short distance into the accelerator tube. After exiting the accelerator, the beam travels through a beam line section to an achromat, then through another short beam line, and finally into one of the beam lines in the experimental hall.

Our tuning experiments were directed to experimental beam line 3, whose purpose is to support research with a free electron laser (FEL). The FEL is a device that produces a laser beam from an interaction caused by accelerated electrons passing an undulator or wiggler. In order to cause the FEL to lase efficiently, the beam must be carefully transported from the beam source, through the accelerator, and into the entrance of the FEL. At the entrance, the beam must have a specific size and shape. The beam must also produce a waist condition inside the undulator. A waist is related to a beam's sigma matrix [Klein 1997]. As of early 1998, no human operator had successfully achieved vigorous lasing in the FEL.

Our first experiments tuned and conditioned the beam to the end of the H beamline. This involves tuning the source, tuning the accelerator, transporting the beam to a set of beam position monitors (BPMs) at the end of the H line, and then producing a minimum spot in the X and Y axis at the first BPM. The tuning process can be separated into four parts: tuning the source, tuning the accelerator, steering the beam to the final BPM, and focusing the beam to produce the correct spot.

Argonne's ATLAS is a heavy-ion facility whose main purpose is to conduct experiments using isotopes of a wide variety of single elements. Ions can be generated from a high voltage platform and accelerated by the PII accelerator or from a second platform and accelerated by a large tandem VandeGraf accelerator. A single beam line transports ions from either source down to an experimental area, and one of many experimental lines. At ATLAS, our tuning experiments were directed to the PII line between the high voltage platform and the PII accelerator. Ions leave the platform in a stream, are transported through a short section of beam line, through an achromatic bend which turns the beam 180°, and then through a section designed to chop and bunch the beam into packets and prepare it for the PII accelerator.

A standard tuning procedure for the PII linac breaks control into four sequential parts: tuning the PII0 line, tuning the PII1 achromat, tuning the PII2 beam line, and then refining the entire tune to produce proper beam conditions at the exit to the PII linac. Tuning procedures for ATLAS are similar to those at ATF. The existence of Faraday cups and wire-scanning profile monitors, as well as a shortage of tuning elements caused the full beam tuning to differ significantly. Our task was to port the control system from the ATF to ATLAS and tune the entire PII beam line.

6.1 First steps at Brookhaven.

To steer through the accelerator, it is required to put the beam on center in order to maximize beam intensity and minimize distortions of beam structure. This task is particularly challenging because of the lack of diagnostics inside the accelerator, thus correct steering was wholly dependent on metrics of the beam's quality as it exited the accelerator.

We used a variety of algorithms during steering optimization. These employed an evaluation function combining metrics of beam intensity, spot size, and beam structure based on data from a profile monitor after the accelerator. Algorithms included two knob hill climbing, gradient descent, and a genetic algorithm.

The two-knob hill-climbing algorithm achieved tuning results that were comparable to, and in some cases exceeded, the best human tuning efforts. A key to the successful application of these algorithms was the proper sequencing of tuning actions. Heuristics derived from human experts proved effective in selecting the next tuning element and action for the tuning sequence.

The second task at Brookhaven was to steer as well as possible through the center of a series of quadrupole magnets. If these quadrupole magnets are properly aligned, their magnetic centers define a true central trajectory through the surrounding beamline section. A beam that passes through a quadrupole magnet off center is steered to a degree proportional to the magnetic field strength and magnitude of the offset. The control system was generally more successful at this task than human operators. Minimization of quadrupole steering requires a tedious repetition of steps, including manipulation of quadrupole field strengths after each adjustment of steering magnets to measure the strength of quadrupole steering. Humans exhibit little patience for the number of steps required to effectively perform this task, and as a result, often do an inadequate job.

The components for automating this task were readily assembled. We were able to build a component subsystem for measuring quadrupole steering and then attach this to a hill-climbing controller that performs a steering scan through the X and Y planes to find the point of minimum steering. Our control subsystem performed minimization of quadrupole steering more effectively and in less time than that required by the human operator. However, at this time in the project we lacked a methodology for sequencing of a series of control actions, i.e., the sequence of actions required for taking the beam from the beginning to the end of the beamline section. The teleo-reactive control framework described above would later fill the role of task sequencing.

6.2 *Porting to Argonne.*

The primary effort in porting the PAL from the ATF to ATLAS was in modifying Vsystem channels [Clout 1993] encoded in various objects to point to appropriate values in the ATLAS control database. Similarities between magnet types were reflected by similar channel structures between the facilities. We added new classes to the PAL for Faraday cups and slits, and modified existing classes for profile monitors and steering, quadrupole, and dipole magnets. We were fortunate that data capture from profile monitors and Faraday cups is similar resource constraints in the frame grabber at ATF. The frame grabber class needed only slight modification to take data from these other two sources.

Overall, porting the control system was accomplished in less than six weeks by a single programmer. No firm numbers were recorded for work time, but we estimate the effort at one person-week. This short porting may not be realistic for larger and more complicated control systems, but is indicative of the great ease of portability possible when moving between similar domains.

6.3 *Noise handling.*

Noise is significant in inhibiting quality beam tuning at ATLAS, where the difference between peak intensities of two beam profiles taken in series can exceed 50%. Background noise in a single profile or pulse is high, causing simple feature detection algorithms to fail. Human operators ignore noise without use of conscious thought. Experienced tuners understand the difference between incidental beam loss due to aberration and recurring loss from source instabilities. Experts are good at identifying relevant features from noisy distributions.

The control architecture does not deal with noise in a single standard way, but divides the into various levels based on correspondence between noise frequency and knowledge necessary for its interpretation and filtering. The initial processing of information in the control architecture occurs in the PAL, where initial data filtration is necessary to respond to higher level beam feature measurement requests. The PAL allows the control system to select from a variety of tools to deal with observed noise. The PAL performs initial averaging of the beam data retrieved from both Faraday cups and profile monitors. Averaging is through software internal to the PAL and the PAL manipulation of various device drivers and hardware features. The controller selects an averaging level based on signal to noise ratio versus the time delay necessary for taking extra samples.

To compensate for pulse-to-pulse noise, we implemented a solver-level algorithm for taking measurements based on prediction. The solver worked by tracking control actions performed in succession, and comparing new values to the projected trajectory. The trajectory was defined by a vector of first and second derivatives calculated from changes in control values from changes in beam measurements. This predictive algorithm is simple

and works for well-behaved functions whose measured derivatives offer good local approximations.

6.4 Goal reduction, coordination, and reactivity.

The beam tuning experiments at ATLAS illustrate a control architecture designed to respond to and implement user goals. Goals are given to a high-level controller that decomposes them and delegates action to other controllers. As these units meet conditions outside their locus of responsibility, they request assistance from their parent. Parents coordinate the activity of child controllers based on an understanding of the tasks that each can accomplish [Klein 1997]. The controllers coordinating beam sections implement plans adopted from human experts. Plans are carried out in a reactive manner using the teleo-reactive (TR) paradigm [Stern et al. 1997]. To illustrate this, we now describe in more detail, the task organization and decomposition needed to tune the PII beamline section at ATLAS.

The process of tuning the PII line can be broken down into two problems with three parts each. Transverse tuning can be broken down into tuning three separate sections named PII-0, PII-1, and PII-2. The purpose of tuning the PII-0 line is to initially steer and focus the beam for transmission through an achromat. PII-1 consists of an achromatic bend that turns the beam 180°. Generally model-derived quadrupole settings through the achromat are sufficient to minimize dispersion. Finally, PII-2 is tuned to generate specific beam conditions and maximize transmission through the PII accelerator.

Tuning the PII section requires a process of iterative refinement, an alternation of steering and focusing actions that shifts forward and back between various beamline locations. The final goal in tuning PII-0 is to produce a beam spot of a predetermined size at the profile monitor beyond the PII-1 achromat. Tuning also maximizes beam transmission through the achromat, as measured by charge readings at the Faraday cup after PII-1. Tuning begins with a nominal tune and then steering followed by focusing to maximize beam transmission at the first Faraday cup. If quad focusing causes excessive steering and beam loss, a tuner should re-steer the beam. Once transmission reaches a minimum level, the beam is focused to the first profile monitor to achieve a specific beam spot.

The final goal of PII-0 tuning is to produce a matching spot before and after the achromat. To complete this task PII-1 must first be tuned. Once PII-1 is tuned correctly, the beam is again focused (and resteeered to compensate for quadrupole steering). The tuner should verify that the spot size at the profile monitors before and after the achromat are identical. If they are not, refocusing (and resteeering) using quads and steerers in PII-0 is required.

Once PII-0 and PII-1 are tuned correctly, the beam is focused and steered down the PII-2 beam section. When target spot size at the end of PII-2 is achieved and transmission is optimized, then the beam is finally ready to enter the PII accelerator.

We constructed a control subsystem to implement the tuning strategy described above. It consisted of four teleo-reactive controllers, one each for PII-0, PII-1, and PII-2, and a parent controller called PII that sequences and coordinates the tuning actions between subsections. The final system emulated the reactive task performance of the best human operators at ATLAS. We characterize and compare the performance of our control system vis-à-vis that of human operators in the next section.

Two events in our experiments at Argonne were particularly noteworthy. We interviewed the resident tuning [Klein et al. 1997c] expert to learn the proper sequence of actions necessary to tune the PII beamline. This physicist found the TR condition-action rule representation particularly intuitive and was able to encode appropriate tuning algorithms with relatively little assistance. The second was the observation of the control system exhibiting spontaneous adaptive behavior that had not been specifically preprogrammed. In performing focusing actions, the control system would regularly backtrack to steering whenever quadrupole steering would cause excessive movement of the beam centroid and a consequent loss of beam transmission.

6.5 Comparisons of performance in tuning tasks at Brookhaven and Argonne.

The two knob hill climbing algorithm achieved tuning results through the accelerator at the Brookhaven ATF that were comparable to, and in some cases exceeded, expert human tuning efforts. A key to the successful application of these algorithms was the proper sequencing of tuning actions. Heuristics derived from human experts proved effective in controlling the selection of the next tuning element and tuning action in the sequence.

Figure 6 shows three snapshots of the beam profile at the exit of the ATF accelerator on a profile monitor after the accelerator. The first snapshot is the initial beam prior to optimization, the second is the profile after two minutes of steering optimization by our control system, and the final snapshot is the beam profile after five minutes of steering

Figure 6. Beam spots after accelerator steering at the Brookhaven ATF.

optimization. The quality of the beam profile at the exit of the accelerator reflects, to a great extent, the degree through which the beam is steered on-center through the accelerator. Off-center steering causes *tails*, loss of luminosity, increased spot size, and other asymmetry-inducing effects. Note that the profile produced after five minutes, although still asymmetric, has smaller spot size and superior luminosity at its center compared to the previous two profiles. Our control system was repeatedly able to produce smaller spots with a core luminosity equal to the best produced by human steering.

During our visit to Argonne in late March 1997 we demonstrated the feasibility of a larger scale integration of control components to perform an extended tuning task. The control system used a total of 18 controllers and solvers to tune a sequence of three transport sections. It used four teleo-reactive controllers [Stern et al. 1997], corresponding to the three sections of the beamline and a supervisory controller, to orchestrate the sequence of control actions and to alternate back and forth between control and diagnostic elements in different sections.

On two separate days of testing, and under somewhat different beam conditions, this system achieved beam transmission levels equaling or exceeding the best tunes of human operators in roughly comparable amounts of time (Table 1). In every case, human operators were able to start from the tune achieved by the control system and improve it, indicating that the control system had found each time at least a local optimum.

Control Element	Our Tune	Operator Tune
STP001_X	-0.65	-0.69
STP001_Y	-0.20	-0.14
QDP001_X	4.09	3.91
QDP001_Y	4.29	4.02
STP002_X	0.05	0.07
STP002_Y	-0.10	-0.12
Transmission	3.55 mA	3.65 mA
PMP001 Sigma_X	3.0513	
PMP001 Sigma_Y	5.3918	

Table 1. A Comparison of Control System and Operator Tunes at Argonne's ATLAS Facility.

6.6 Application to beamline alignment analysis.

In the summer of 1997 we conducted one final field test at the Brookhaven ATF. We were asked to investigate the alignment of quadrupole magnets along the beamline prior to a survey that was about to be performed there. With only about two weeks notice, we adapted our automatic tuning system for the ATF into a diagnostic system that identified quadrupole misalignments. This diagnostic functionality was achieved by reconfiguring the

algorithm for steering through the center of individual quadrupole magnets, described in Section 6.1 above, into a system for finding the optimal steer through a series of quadrupole magnets. Taking this optimal steer as a reference trajectory, our algorithm then returns the X and Y displacements of this reference trajectory with respect to the magnetic centers of each of the quadrupole magnets through which the beam passes.

We used this automated system for defining reference trajectories and off-center displacements repeatedly, choosing different sets of quadrupoles on each run to define different reference trajectories. We then selected runs that gave similar reference trajectories, and chose an average of those similar trajectories as the optimal reference trajectory. Finally, we used this optimal reference trajectory to identify the most probable quadrupole misalignments.

Even with automation, this procedure proved to be quite time consuming. Initial setup and bug fixes consumed a significant amount of the two days of beam time that was available, and data collected within the remaining beam time was not as consistent as we would have hoped. However, even with partial inconsistencies in our data, we were able to conjecture misalignments among series of quadrupoles in two beamline sections that were similar to those conjectured by the supervising physicist at the ATF, Xijie Wang.

7 *Recent project history.*

William Klein's departure from Vista in the fall of 1997 was a major development for our project. Dr. Klein had been a key figure on this research project since early in Phase I and had played a major role in developing the hierarchical distributed design of the control system. This work had been a part of Dr. Klein's dissertation research, and he decided to move on shortly after completion of his Ph.D.

Dr. Klein's departure marked a major reduction in the level of available effort. Work in late 1997 focused on investigation of potential alternatives to our homegrown distributed object architecture. We examined CORBA, ACE, CDEV, and other frameworks for distributed object communication and execution. We found them unsatisfactory for the purposes of our knowledge-based control system due to limitations in speed, maturity, and expressive power.

Eric Olssen, a more recent addition to our team, left Vista at the end of 1997. After Olssen's departure and for much of 1998 our continuing work focused on development of various aspects of the teleo-reactive framework. We constructed two planners capable of automatically writing teleo-reactive programs. These planners were tested repeatedly in a variety of simulated environments. They were also modified to perform plan repair, and were tested on a number of defective plans adapted from our earlier ATLAS control system.

We also continued design and implementation work on integrating online simulation and modeling services into the control system. We also continued to refine the simulations / models for Brookhaven and Argonne. This integration of online simulations / models is intended to extend the decision-making capabilities of our control architecture. By advance testing of alternative actions or control programs in a simulated version of the current environment, the control system is provided with an additional method for determining which control action or control program to run next.

A final thrust of Phase II work was the experimentation and development of two new neural network components. These new neural network templates, a flexible version of the perceptron network, and a multi-layer backpropagation network, constitute powerful extensions to our library of general purpose *solvers*.

8 *Phase III prospects.*

Continued development of this control architecture has been delayed due to lack of funding. Vista had intended to commercialize promising technologies, leveraging Phase II research in neural network control, fuzzy logic control, genetic algorithm control, teleo-reactive control, etc. The distributed object framework and the model of knowledge-based adaptive and supervisory control developed in Phase II had also been considered for application to general industrial control.

Vista had hoped to secure partial Phase III funding for commercialization via R&D partnerships with one or more of its large corporate customers. However, this plan has been undermined by the Asian economic crisis. Many of Vista's largest customers are in the metals processing industry and this industry has been strongly impacted by the recent flood of inexpensive Asian metal imports. Vista's ability to fund further R&D has also been impacted by this development. In addition, Vista's recent economic conditions have made it hard for Vista to retain key personnel from the Phase II research effort, further increasing the difficulty of continuing into Phase III.

9. *Assessing the current status of the intelligent control architecture.*

This project constitutes a highly ambitious and unique effort to use AI technologies to develop a large-scale control architecture. AI technologies play roles in our architecture at two key levels. At the highest levels, knowledge-based decision making and inference is used for goal-to-task decomposition, and for diagnosis and reconfiguration in the case of control plan failures. In the context of our hierarchical distributed architecture, high level knowledge-based controllers adaptively execute control plans that coordinate independently acting local controllers and integrate local control actions into larger more global control strategies.

The second role of AI technology is at the intermediate level of the control system. Adaptive and search-based technologies, including hill-climbing as well as fuzzy logic,

genetic algorithm and neural network control, have been used to implement various forms of intelligent adaptive behavior. These technologies have been selected because of their ability to emulate the performance of skilled human operators and their potential for performance improvement through learning. Our fundamental goal in using these technologies has been to extend the envelope of effective control beyond conventional closed loop control, automating tasks for which human skills and human technical expertise had previously been required.

The large scale and ambitious scope of this design has impacted our development trajectory and the prospects for commercialization in several ways. First, the time and resources required for developing and maturing such an ambitious architecture exceeded what could be achieved in a Phase II project. This has meant that there is much in our architecture that remains immature and requires further refinement. Second, although we have demonstrated the power and utility of the control architecture on a small number of problems in accelerator tuning, a far larger range of applications will have to be developed both to generalize the technology and to establish the range of applications for which it is appropriate and cost effective.

The issue of appropriate and cost effective use of this technology is a critical issue that remains to be resolved. Vista's industrial control customers are wisely very conservative in their choice of control technologies. They insist on proven technology for which stability and reliability can be established both mathematically and experientially. The high potential cost of control system failures makes this choice prudent. On the other hand, the behavior of "intelligent technologies" can be regarded as fundamentally more difficult to verify precisely because of the very characteristics that give them their power, namely adaptation and learning. This would seem to imply that the range of applications for intelligent technologies should be those involving a high degree of difficulty and relatively low risk.

Another possibility is that these technologies be used to support cooperative multi-initiative control alongside human operators. This is generally the approach that we have taken in the development of automated accelerator control. We have developed our control automation software as a set of tools that speed up certain tedious or repetitive operations and also expedite critical knowledge and information access, thereby extending the capabilities and awareness of the human operator. In this vein, Vista may contemplate the addition of some of these intelligent technologies to its operator-based control tools, including its logging and alarm utilities and its scripting-based automation tool.

10. Summary.

The purpose of this project has been to develop a portable intelligent system for accelerator control. The design for this system is based on the concept of an extensible architecture that integrates a variety of conventional and "intelligent" control components. This knowledge intensive approach is intended to provide the potential for real-time adaptive control and performance improvement through learning.

Phase I work developed and tested this design concept. Adaptive configuration of hill-climbing optimization components, neural networks, genetic algorithms, and fuzzy controllers was tested in a simulated setting on a variety of small tuning problems. The results of these early tests established the potential utility of this approach in accelerator tuning.

This Phase II goal was to incorporate the methodology of adaptive configuration into control system design in a principled and systematic fashion. This led to the development of a knowledge-based hierarchical distributed architecture. At the lowest level, a physical access layer provides an object-oriented abstraction of the target system. A series of intermediate layers implement general algorithms for control, optimization, data interpretation, and diagnosis. Decision making and planning is organized by knowledge-based components that utilize knowledge acquired from human experts to appropriately direct and configure lower level services. The general nature of the representations and algorithms at lower levels gives this architecture a high degree of potential portability. The knowledge-based decision-making and planning at higher levels gives this system an adaptive capability as well as making it readily configurable to new environments.

Over the past three and half years we have used more than five weeks of beam time to test this architecture at Brookhaven and Argonne. We tested individual control components capable of a single tuning task such as producing a waist in the beam or minimizing steering through a sequence of quad magnets. In almost every case, performance achieved in these individual tuning tasks was comparable to the best tunes achieved by human operators.

A good portion of our more recent effort has been spent coordinating the activity of these control subsystems to adaptively execute sustained sequences of control actions of the type required to tune the entire accelerator from source to target. Experiments using a teleo-reactive mode of execution have been very successful, again achieving tunes comparable to the best human tunes. These larger scale results suggest that the use of an hierarchical distributed control architecture in conjunction with a teleo-reactive control mechanism for sequencing may be adequate to provide the degree of adaptive control coordination required by the accelerator tuning task.

Finally, our most recent effort involved applying this control system framework to the diagnosis of beamline misalignments at the Brookhaven ATF. Once again the utility of our component-based approach was demonstrated in the speed with which this new, albeit somewhat rough, application could be assembled, i.e., less than two weeks. Brief testing at Brookhaven of this alignment analysis tool showed early but promising results.

References

- Acar, L. and Ozgunner, U. 1993. Design of Structure-Based Hierarchies for Distributed Intelligent Control. In Antsaklis et al., *An Introduction to Intelligent and Autonomous Control*, Boston: Kluwer.
- Antsaklis, Panos J., and Passino, Kevin M., eds. 1993. *An Introduction to Intelligent and Autonomous Control*. Boston: Kluwer Academic Publishers.
- Antsaklis, P. J., Passino, K. M., and Wang, S. J. 1989. Towards intelligent control systems: Architectures and fundamental issues. *Journal of Intelligent and Robotic Systems*, 1: 315-342, 1989.
- Astrom, K. J., Anton, J. J., and Arzen, K.E. 1986. Expert Control *Automatica*, Volume 22 Number 3.
- Behrens, Ulf, Flasiński, Mariusz, Hagge, Lars, Jurek, Janusz, and Ohrenberg, Kars. 1996. Recent Developments of the ZEUS Expert System ZEX. *IEEE Transactions on Nuclear Science*, Vol 43.
- Benson, S. 1995. Inductive Learning of Reactive Action Models in *Machine Learning: Proceedings of the Twelve International Conference*, San Francisco CA: Morgan Kaufmann.
- Bouche, J. M., Daems, G., Vilimonov, V., Khomoutnikov, V., Perriollat, F., and Ryabov, Yu. 1995. Representation and Usage of Knowledge for Initialization of Accelerator Equipment. CERN Rpt.
- Clearwater, Scott, Papcun, George, and Clark, David. 1986. Developing an Expert System to Control a Beamline at the Los Alamos Meson Physics Facility. *Nuclear Instruments and Methods in Physics Research*, A247.
- Clearwater, Scott H. and Stern, Eric G. 1990. Beyond Expert Systems: Learning Programs in Large Physics Control Systems. *Nuclear Instruments and Methods in Physics Research*, A293.
- Clout, P. 1993. The status of Vsystem. *Proceedings of the Third International Conference on Accelerator and Large Experimental Physics Control Systems*, Germany.
- Fikes, R. E. and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2, 189-208.
- Himel, T., Allison, S., Grossberg, P., Hendrickson, L., Sass, R., Shoaee, H. 1993. Adaptive Cascaded Beam-Based Feedback at the SLC. Proceedings of the 1993 IEEE Particle Accelerator Conference. Washington, D.C.
- Howell, J. A., Barnes, C. W., Brown, S. K., Flake, G. W., Jones, R. D., Lee, Y. C. Qian, S., and Wright, R. M. 1990. Control of a Negative-Ion Accelerator Source Using Neural Networks. *Nuclear Instruments and Methods in Physics Research*, A293.
- Klein, W. 1997. *A Software Architecture for Intelligent Control*. Doctoral Dissertation, Department of Computer Science, University of New Mexico, Albuquerque, NM.

- Klein, W., Stern, C., Luger, G., and Olsson E. 1997a. An Intelligent Control Architecture for Accelerator Beamline Tuning. *Proceedings of the Innovative Applications of Artificial Intelligence Conference*, Cambridge MA: MIT Press.
- Klein, W., Stern, C., Luger G., and Olsson, E. 1997b. Designing a Portable Architecture for Intelligent Particle Accelerator Control, *Proceedings of the Particle Accelerator Conference*, American Physical Society.
- Klein, W., Stern, C., Kroupa, M., Westervelt, R., Luger G., and Olsson, E. 1997c. Tuning and Optimization at Brookhaven and Argonne: Results of Recent Experiments, *Proceedings of the Particle Accelerator Conference*, American Physical Society.
- Klein, W.B., Westervelt, R. T., and Luger, G. F. 1997c, Developing a General Purpose Intelligent Control System for Particle Accelerators. *Journal of Intelligent and Fuzzy Systems*, in press.
- Lee, Martin and Clearwater, Scott. 1987. Generic orbit and lattice debugger. *Proceedings of the Model Based Accelerator Controls Workshop*, Brookhaven National Laboratory.
- Muggleton, S. and Feng, C. 1990. Efficient induction of logic programs in *Proceedings of the First Conference on Algorithm Learning Theory*, pp. 368-381.
- Nilsson, N. J. 1994. Teleo-Reactive Programs for Agent Control.. *Journal of Artificial Intelligence Research*, 1, pp. 139-158, January 1994.
- Nguyen, D., Lee, M., Sass, R., Shoaee, H. 1991. Accelerator and Feedback Control Simulation Using Neural Networks. *SLAC-PUB-5503*.
- Shultz, David E. and Brown, Pamela A. 1990. The Development of an Expert System to Tune a Beamline. *Nuclear Instruments and Methods in Physics Research*, A293.
- Stern, C., Olsson, E., Kroupa, M., Westervelt, R., Luger, G., Klein, W., A. 1998. Control System for Accelerator Tuning Combining Adaptive Plan Execution with Online Learning, Talk at ICALEPCS'97.
- Stern, C., W. Klein, G. Luger, M. Kroupa. 1997. Abductive Model Refinement for Accelerator Control. Read at: Computational Accelerator Physics Conference 1996. Published in Computational Accelerator Physics: AIP Conference Proceedings 391. Woodbury, NY: AIP Press.