T.3: Intelligent control for accelerator systems

Rishi Pal Yadav
Accelerator Control Systems Division
Email: rpyadav@rrcat.gov.in

Abstract

The reliable operation of large accelerator machine requires control system with higher level of automation, flexibility, robustness, and optimisation. For fulfilling these requirements, this study aims at incorporating artificial intelligence concepts at different accelerator control system levels through intelligent objects (agents). The framework for multi-agent based control of accelerator has been developed. Using this framework different accelerator control agents are developed. Then different multi-agent based control schemes for synchrotron radiation source taking Indus-2 as case study has been analysed through simulations.

1. Introduction

Particle accelerators are machines used for increasing the energy of charged particles. They find application in many fields of fundamental physics from elementary particles, astrophysics and cosmology to solid state, nuclear and atomic physics. In the last three decades, the synchrotron radiation (produced at synchrotron radiation source (SRS) accelerator facilities) has evolved as an important tool in chemistry and biology to study molecules, atoms and nuclei structures. The operation of these accelerator machines generally requires the monitoring and control of a large number of system parameters. This is achieved with the distributed control system comprising different interconnected software and hardware control system layers spread over a large area and covering different subsystems of accelerators like injector, transport lines, storage ring and beam lines. The important control requirement is that all the sub-system operations are to be performed in a synchronized and sequential manner with interlocking and pre-qualification of actions at different machine operation states.

Intelligent objects are software programs called as agents that are situated in a given environment and are capable of acting in this dynamic environment towards achieving their goal. An agent program is modeled in terms of mentalistic notions such as beliefs, desires and intentions so that the developed software entity may possess some of the basic properties such as autonomy, social ability, reactivity, and pro-activity, generally exhibited by humans. With these attributes, agents provide a high abstraction level for developing software and thereby potentially simplify the control system design for complex systems.

Presently work is going towards developing intelligent object based accelerator control methods. These methods aim towards enhancing the system's operability, by changing the operation emphasis, from supervisory control of the subsystems to that on the role played by the subsystem in the overall broader perspective of the system. This includes the direct goal assignment to subsystem control agent in terms of accelerator physics parameters that are at much higher level of abstraction. It thus eases the operator efforts towards accelerator tuning by integrating the accelerator physics based decision making at the control system level. Another major advantage of agent based concept in accelerator control is the enhanced overall system performance as the agents being fast and flexible can incorporate the run-time learning abilities. This relaxes the accurate modelling pre-requirement as well as improves the system robustness. The agent's social ability of inter-agent interaction can be further utilised for on-line prioritised optimisation and cooperative tuning of accelerator subsystems with the constituent agents deciding their future course of actions through mutual consensus. Also the careful distribution of these intelligent agents at different hardware layers in multi-layer accelerator control system architecture increases the effective resource utilization.

In this article the work carried out for developing the intelligent object framework to control different accelerator systems is presented. This framework allows the formulation of various agents for control of different accelerator subsystems to collectively achieve the task of cooperative accelerator tuning and gain theoretical understanding of the approach. For this, first, the control system model of the accelerator facility is developed. Specifically, the basic model of the typical SRS facility comprising electron source, transport line, high energy booster and storage ring is developed. For this model, the control system interface similar to the actual machine interface is built. Depending on the subsystem type, the methods of modelling through first principle or model identification through experiments have been utilized. Then, in an incremental manner for each of the sub-systems namely electron source (pre-injector), transport line, booster and storage ring, the intelligent agents are formulated. Simulation program is developed for analysing the effectiveness of the agent based control of individual subsystems and the multi-agent based scheme for the overall system. Then the scheme is extended by formulating the distributed multi-agent based control for tuning of beam orbit in the synchrotron radiation source. Section 2 of the article

discusses the typical SRS control systems. Section 3 discusses the brief overview of agent based controls. In section 4 the results of simulation using developed control agents are presented for different scenarios.

2. SRS control systems

2.1 SRS control system evolution

The particle accelerators started appearing around 1930. The first accelerator used for accelerating protons to energy of 400 keV was built with few sub systems like high voltage generator, proton source, vacuum pump, lithium target and scintillation screen. It was fully manually controlled during experiments. With advancements in various technologies accelerators are now being made bigger and bigger with a large number of subsystems spread over large geographical areas. Much more complex instruments and machines are needed to be run in a synchronized and sequential manner for their operation. Also the number of parameters to be controlled and monitored during the experiments have increased to such an extent that the manual control and observation is beyond the scope of human capabilities. This has led to the development of accelerator control systems in accelerator facilities. In today's accelerator facilities almost all the operations are exercised remotely and centrally with operators sitting in control room and working from operator console PCs with the help of Graphical User Interfaces (GUI) running on them.

SRS are built around the accelerators facilities and comprise of a pre-injector (small accelerator linac or microtron), beam transport lines, booster (bigger accelerator usually synchrotron) and storage ring. Thus they comprise of many accelerators and hence their control system has evolved out of accelerator control system. The first generation SRS machines were the electron storage rings built specifically to store continuously circulating electron beam at a fixed energy for periods up to many hours. The control system of some of these early machines were analogue type but with the availability of computers in market around the same time control systems of accelerators started using minicomputers with Central Processing Units (CPU) at the upper layer (operator interface layer) and CAMAC compliant I/O cards in CAMAC crates at lower layer. The second generation of synchrotron radiation sources started around 1980 where the accelerator and storage rings with modified lattice structure were built to attain the increased brightness by minimizing the electron beam emittance. With the use of microprocessor for most of these facilities, control system architecture, still remained two layered with much powerful computers at the upper layer and low capacity computers at lower layers. The use of real-time operating systems started appearing during this time at the lower layer controllers whereas the upper layer computers mostly used the proprietary operating systems on the concerned computers. The applications were built using the concept of distributed databases, device description tables, device tables with about 2500 parameters and data archiving rates of once per 2 minutes were developed. The development languages used were mainly C, FORTRAN and Assembly, and the databases used were of text type and relational databases. Some of the facilities developed around 1985 for the first time used the VME crates at the middle layer and were among the first to adopt the three-layer architecture. The software programs at layer one started using the new concepts like event based programming, inter-program communication facilities and were based on enhanced graphical plotting abilities (bar graphs, joined line graphs, scatter plots). The operator interfaces were equipped with Alphanumeric terminals, Color TV-raster scan with interactive cursors, trackball, computer controlled knobs with incremental encoders and the facility of multi-parameter control linked with single computer knob used for example, for producing bump on closed orbit. The third generation synchrotron radiation sources were the machines specifically built with the motto of providing higher brightness and to accommodate large number of insertion devices. These machines are the modern present day synchrotron light sources. They have the most advanced control systems and some of them also provide the facility of top-up injection. In these facilities the computer controlled system normally comprises of two parts. The first part is the hardware that directly interfaces with the sensors and actuators. Since the accelerators are spread over a large area and comprise of many subsystems, normally this hardware part is spread over one or two layers with many small computers/controllers connected over field-buses constituting the lower layer and higher computing capacity based computers constituting the upper layer. One or more upper layer computers logically combine to make the control system for one sub-system. The control systems of various sub-systems together make the overall hardware part of control system for the complete facility. The second part is the software part that comprises of the lower layer subroutines running at the lower layer controllers, the control scripts and GUI algorithm running at the operator PC consoles. With the increase in the computing powers of computer and the changing electronic market scenarios the control system of many facilities has seen upgrades at different time, Fig. T.3.1 shows the time between major upgrades in SRS control systems of different facilities. The SRS control system has passed through many technological changes from 1970 to 2010 and the present trend shows the growth around two main technologies: first

EPICS and the second TANGO, Fig. T.3.2 shows the trend in the control system technologies adopted by different sources.

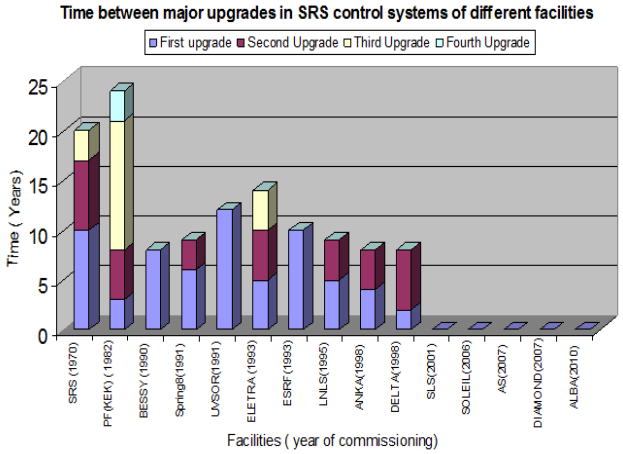


Fig. T.3.1: Time between major upgrades in SRS control systems of different facilities

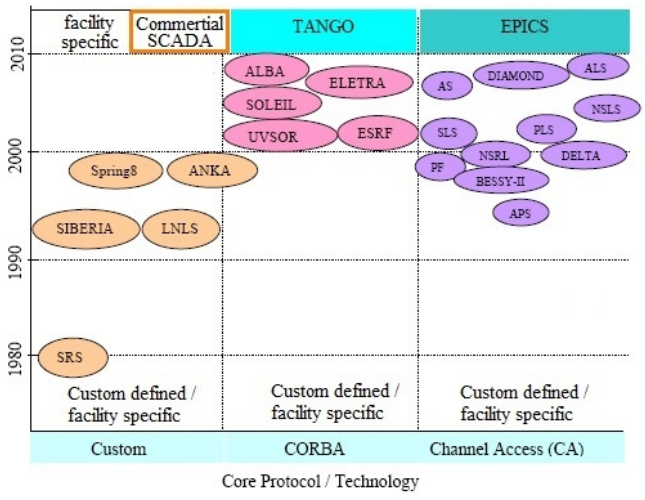


Fig. T.3.2: The control system framework adopted by different facilities

2.2 Some of the SRS control system requirements

SRS control systems have to manage the subsystems like RF, vacuum, Injector, Diagnostics, Magnet power supplies, Interlock, Timing, beam line front end and electron beam orbit control. Technical requirements for these differ from machine to machine; some of the technical requirements expected from third generation SRS control system facilities are as follows :

- Support for about 100,000 direct parameters and about 30,000 derived parameters.
- Support for about 1 or 2 Hz model-based control (used to correct steering, electron orbit, tune, chromaticity, optics etc).
- Synchronous power supply and RF ramping, with power supplies spread over a large geographical area.
- Timing system with jitter less than 100 ps and resolution down to 20 ns with facility to supply synchronization signals over large geographical areas.
- 500 MHz RF control (Amplitude, phase, synchronization)
- 5 kHz fast orbit feedback (for coherent bunch instabilities) and about 50 to 100 Hz global and local orbit feedback.
- About 20 to 200 millisecond response time equipment protection system.
- About 5 Hz updates to operators of up to 1,000 chosen parameters.
- Coherent turn-by-turn orbit data for up to 210 = 1,024 consecutive turns (for FFT).
- Archive up to 6,000 parameters at a rate of 0.5 Hz continually.
- Latch the last 10 seconds of data from all parameters in the storage ring when a fault is detected in the Machine Protection System (MPS), for postmortem analysis.
- Archive up to 1,024 consecutive turn by turn data for 1,000 parameters at a rate of 10 Hz.
- Provide pulse-to-pulse beam steering in the injectors (linac or small accelerators) at 1 Hz.

3. Agents as intelligent objects

3.1 Types of software agents

Software agents are software entities designed to perform a designated function on behalf of a user/ its human counterpart. For example, an accelerator controlling agent can be viewed as the virtual operator that controls the accelerator operations on behalf of the actual operator. The agent concept has been studied by many researchers from varied fields such as e-commerce, web based applications, distributed computing, medical diagnosis, supply chain distribution, production scheduling, industrial process control and many more. To distinguish the intelligent agents from classical programs different researchers have given different definitions. The following attributes distinguish intelligent agents from classical programs.

- capable of acting in an environment
- able to communicate directly with other agents
- is driven by a set of tendencies (in the form of individual objectives or of a satisfaction/survival function which it tries to optimize)
- has resources of its own
- capable of perceiving its environment (but to a limited extent)

- has only a partial representation of its environment (and perhaps none at all)
- possesses skills and can offer services
- can improve its skills through learning
- tends towards satisfying its objectives, taking account of the resources and skills available to it and depending on its perception, its representations and the communications it receives.
- capable of taking initiatives to adapt in dynamic environment

The above listed attributes are encapsulated in modules that serve as the building block for the realization of the agent structure. Starting from simple abstract agent (Algorithm 1) different types of agent architectures are in practice for agent implementation. Algorithm 2 gives the pseudo-code for a subsumption agent. Algorithm 3 gives the pseudo-code for a logic based agent. Algorithm 4 gives the standard Beliefs-Desire-Intention (BDI) interpreter.

Algorithm 1: Execution steps of an abstract agent.

```
Data: observed environment state(s : S), action set A
Result: perform action â
initialise internal state to i_0 ;
repeat
    /*observe environment for state */
    READ s ;
    /*and generate perception */
    p ← see(s) ;
    /* select action according to internal state and percept */
    â ← action(next(i_0, p));
    /*perform selected action */
    DO â;
until stop;
```

Algorithm 2: Pseudocode for action selection by a subsumption agent.

```
Data: percept (p : P), action set A
Result: selected action â
/*get the list of all the rules for the percept P */
initialise fired:P(R) ;
/*for the current percept p generate the list of fired rules */
fired:={(c, â)|(c, â) ∈ R and p ∈ c};
/*if fired rule list is not empty */
if |fired| > 0 then
    /*then select the highest priority rule */
    find minimum (c, â);
    /*and return the action sequence associated with it */
    return â;
else
    /*otherwise do nothing */
    return null;
end
```

Algorithm 3: Pseudocode for action selection by a logic based agent.

```
Data: system state Δ, logic sentences set L, L-formulae set D, action set A
Result: selected action â
/*get the deduction rules ρ from the Knowledge-Base K */
initialise get ρ, ;
/*for each action â element of set A */
for each a ∈ A do
    /*if from the internal state Δ the formula ℘(â) can be proved
      using deduction rule ρ */
    if Δ⊢_ρ℘(â) then
        /*then return action â as the valid action */
        return â ;
    end
end
/*for each action â element of set A */
for each a ∈ A do
    /*if from the internal state Δ the formula ¬℘(â) can not be
      proved using deduction rule ρ */
    if Δ⊬_ρ¬℘(â) then
        /*then return action â as the valid action */
        return â ;
    end
end
/*otherwise do nothing */
return null ;
```

Algorithm 4: Standard BDI interpreter.

```
Data: Sets beliefs B, Desires D, Intentions I
Result: perform action()
initialise internal state ;
repeat
    /*observe environment for events */
    READ event − queue ;
    /*and generate options */
    options ← OPTION-GENERATE(event − queue) ;
    /* select valid options */
    selected − options ← DELIBERATE(options);
    /* add the selected options to intentions list */
    UPDATE-INTENTIONS(selected − options);
    /* execute one action execution cycle */
    EXECUTE(intentions);
    /* get new external events */
    READ event − queue ;
    /* remove the sucessfull attitudes */
    DROP(successful-attitudes);
    /* remove the impossible attitudes */
    DROP(impossible-attitudes);
until stop;
```

3.2 Intelligent control in accelerator environment

In accelerator community the first use of agent application was demonstrated by Jennings [1]. Jennings connected the two diagnostic expert systems in the accelerator control environment namely CODES (control system diagnosis expert system) and BEDES (beam diagnosis expert system) by using the ARCHON framework. Thus converting them to two distinct intelligent agents, which can work together to

fulfil the common goal of diagnosing the faults. Figure T.3.3 gives the architecture for ARCHON framework [2]. Strengths of ARCHON architecture were: use of a top down approach to look at the overall needs of the application and a bottom up approach to look at the capabilities of the existing system so that the development efforts can be minimised. Problem solution was achieved through pre-specified recipes called as plans. Plans were represented in the form of tree with nodes as tasks and arcs as conditions. The self-model and the acquaintance model were provided for assistance. Another effort towards the intelligent control of accelerator was done by Klein *et al.* [3,4]. They proposed a control system architecture based on hierarchical distributed knowledge-based controllers, each of which is an 'expert' in controlling some section of the environment or in performing some function over that environment. These Knowledge-based controllers were provided with the capabilities for planning, diagnosis, and learning, as well as knowledge acquired from human domain experts to select, sequence, and configure control actions.

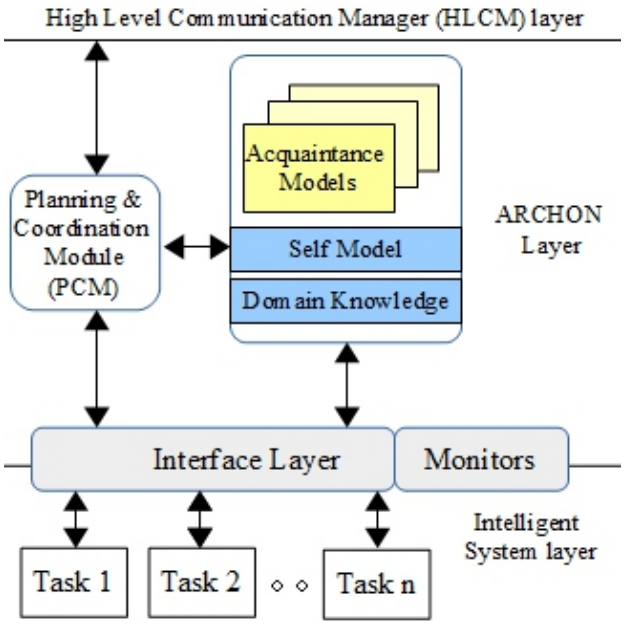


Fig. T.3.3: The ARCHON agent architecture

Controllers were also designated with the responsibility for reasoning about the system state, diagnosing errors in controller actions, decomposing goals into tasks and actions, and initiating human intervention as and when found necessary. Means for implementing the general purpose optimization or control algorithms, such as hill-climbing optimization, fuzzy logic or neural network-based feedback control, conventional control loops, etc were provided through services called as solvers. More complex procedures were formulated by coordinated assembly of individual solvers. An object oriented physical access layer (PAL) was developed as an abstraction mechanism between controllers and the underlying control system to provides a mechanism for hiding unimportant implementation details about the domain hardware and provides a uniform interface for control access. A new control approach called as teleo-reactive (TR)[5] control was utilized which is a combination of feedback-based control and traditional discrete action planning. TR programs sequence the execution of actions that have been assembled into a goal-related plan. Here unlike traditional planning environments, no assumption is made that actions are discrete and un-interruptible and that an action's effects are completely predictable. On the contrary teleo-actions are executed as long as the action's preconditions hold and its goal has not yet been achieved . The strongest point of this approach is that a short sense-react cycle ensures that when the environment changes, the control action changes to fit the new state. The intelligent system concepts has also been tried for automatic beamline alignment problem by Pugliese *et al.*[6]. They proposed a model reference based architecture by combining the hierarchical and subsumption architecture. The strength of this approach is the coupling of an artificial intelligence and a real-time sub-system as parallel, cooperating components. The neural networks and genetic algorithms based approach for electron transport line optimization has also been tried by Schiemer *et al.*[7].

4. Agent based control for SRS control systems

4.1 Agent based control for beam transport lines

Taking advantage of modular control system architecture of accelerators the model based goal based modular architecture of transport line agent is formulated. Considering Transport Line-1 (TL-1) of Indus accelerator complex as a case study the agent goals (Tune beam position and angle, cooperatively optimize injection current, auto start/stop TL-1) are formulated with associated plans. For each of these plans the preconditions and action sequences are formulated along with the required capabilities for the agent's model blocks. For implementation of this agent a three layered framework is developed in graphical programming language (Fig. T.3.4) with postman-postoffice based communication feature for communication between distributed inter-agent/ intra-agents modules. The overall model of accelerator machine comprising beam source, TL-1 and Booster ring is developed for simulation studies by combining the developed individual models of these subsystems. TL-1 and Booster models are

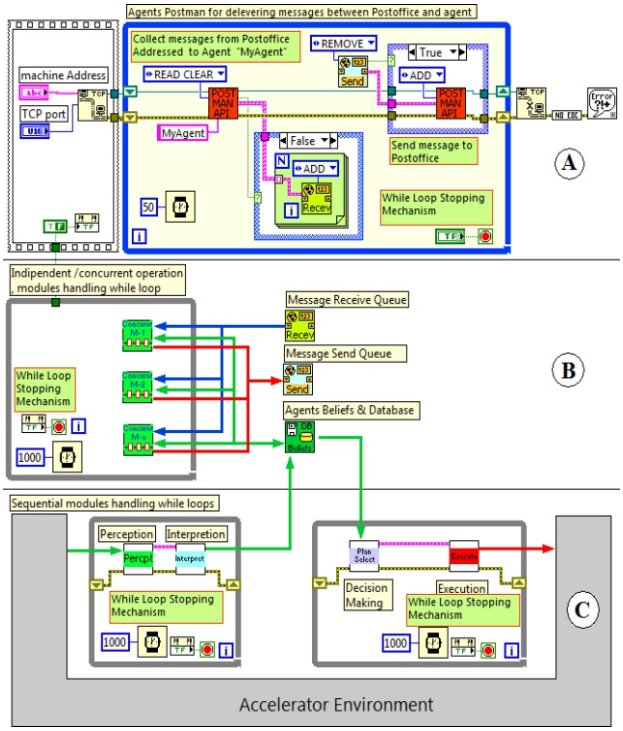


Fig. T.3.4: LabVIEW framework for Agent representation, layer (A) implements the agents communication with the postoffice, layer (B) implements the concurrent process modules, layer (C) implements the agents interaction with the accelerator environment

developed in MAD program. This developed model is validated through experimental results. The results show good agreement of the developed model with the experimentally observed parameter variations (Table 1).

Table. 1 Goodness of fit for TL-1 and Booster model.

Name	BR current at Inj			BR current at @3.3μs		
	SSE	R-square	RMSE	SSE	R-square	RMSE
HSC2	0.10	0.96	0.07	0.03	0.98	0.04
HSC3	0.04	0.97	0.05	0.02	0.98	0.04
HSC4	0.31	0.76	0.15	0.31	0.76	0.15
VSC2	0.05	0.98	0.05	0.07	0.97	0.06
VSC3	0.04	0.98	0.05	0.01	0.99	0.02
VSC4	0.01	0.99	0.03	0.01	0.99	0.03
VSC5	0.18	0.89	0.10	0.22	0.85	0.11

Combining the developed accelerator machine models and the TL-1 control agent the simulation program is developed with block diagram shown in Fig. T.3.5. For monitoring and control of simulation a separate GUI is developed.

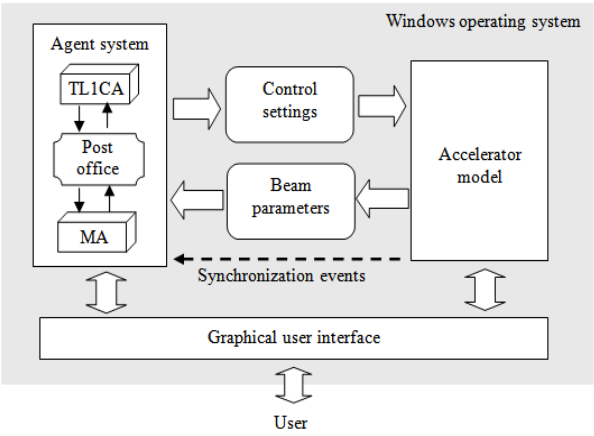


Fig. T.3.5: Simulation program organization

Simulation results (Fig. T.3.6) [8] obtained for the accelerator control with and without agent based control system towards the problem of beam position variations normally observed at the microtron output shows that this method can effectively

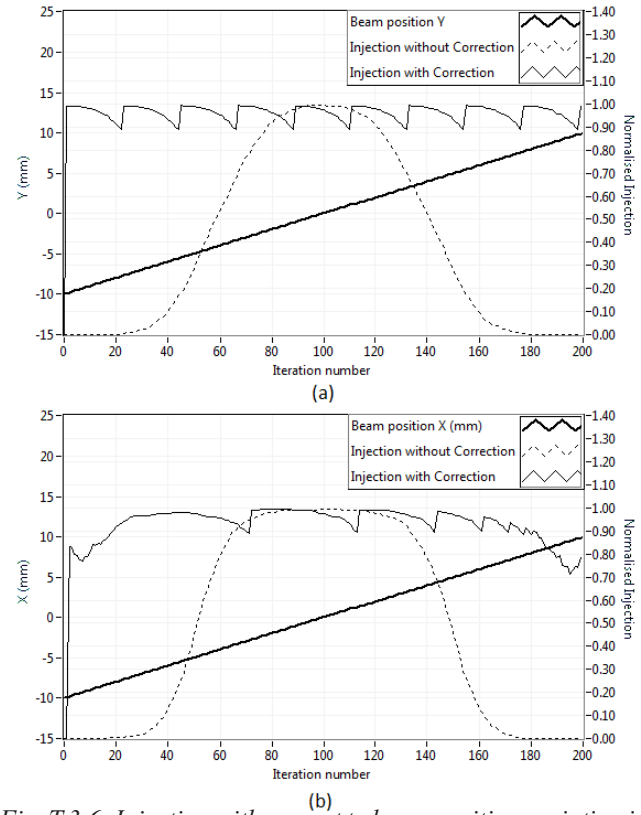


Fig. T.3.6: Injection with respect to beam position variation in (a) vertical and (b) horizontal plane with and without correction

increase the systems operating range under such conditions while maintaining the required level of injection current

4.2 TL-1 tuning using model based tracking

The agent goals and plans are added with model based tracking methods to estimate the beam position at the faulty BPI locations. This estimated data along with the measured beam position data is used by the agent for computing the system's present state, based on that it decides the next actions to be performed towards tuning of the transport lines (Fig. T.3.7). This control system scheme for different BPI failure scenarios is formulated and the condition for successful tuning of transport line by it is derived.

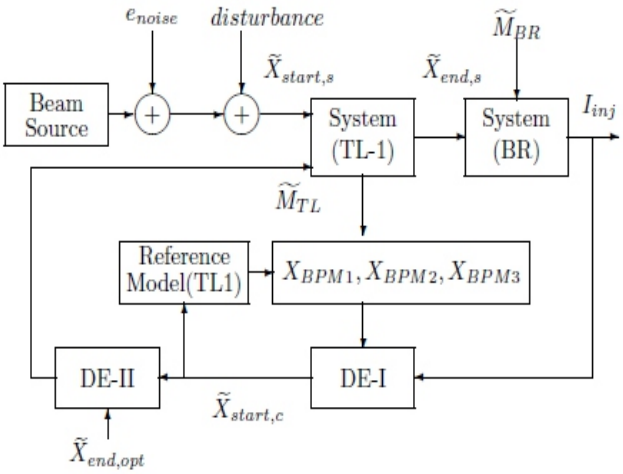


Fig. T.3.7: Block diagram of the agent based control system

The simulation results show that this scheme can successfully handle the limited BPI and corrector failure conditions while tuning the transport lines (Fig. T.3.8)[9].

4.3 Microtron control agent development

For microtron control agent the model of microtron is developed using the black-box identification method. The experiments are performed on the system to identify the dependence of emission, reflected power, FCT and beam position over control parameters, cathode current and RF frequency with RF level, dipole current, RHC, LHC and mid-plane current values kept constant. From the Experimentally collected data the interdependence between different parameters of microtron are computed and the four-input five-output nonlinear model of the microtron system is coded in graphical programming language as shown in Fig. T.3.9. The abstract architecture of microtron control agent is formulated with architecture shown in Fig. T.3.10.

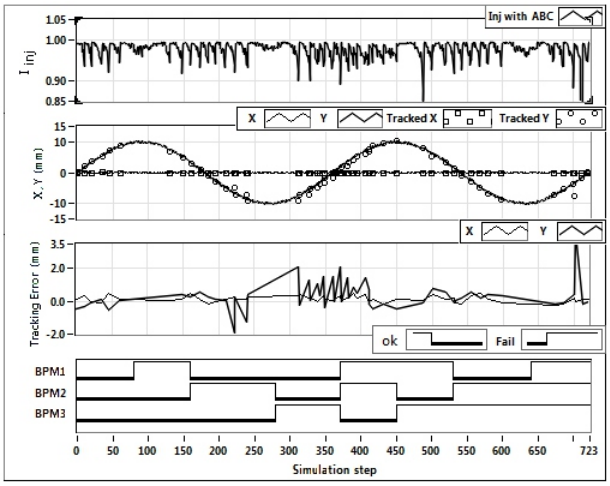


Fig. T.3.8: Injection current, tracked beam position, and tracking error for different BPI failure conditions

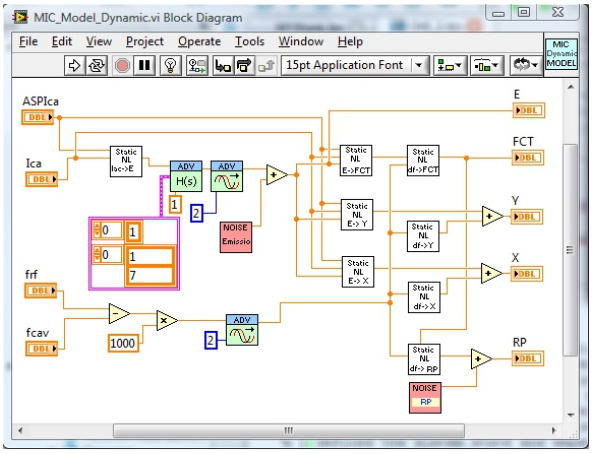


Fig. T.3.9: The microtron model VI block diagram

4.4 Multi-agent based control for accelerators

The developed accelerator models and agents are connected as shown in Fig. T.3.11 and are analysed for cooperative tuning of accelerators by TL-1 and microtron control agents. The simulation results (Fig. T.3.12, T.3.13 and T.3.14)[10] for controlling the pre-injector accelerator microtron and Transport line under the influence of disturbance on beam [for three scenarios: (1) when microtron and TL-1 agents optimizes there operations individually, (2) when both of the agents work cooperatively for optimizing the injection current as well as reducing the no of changes in TL-1 settings, (3) when both of the agents work cooperatively with dynamics

learning case.] in this scheme shows that this scheme can be used successfully for their optimal control without operator interventions.

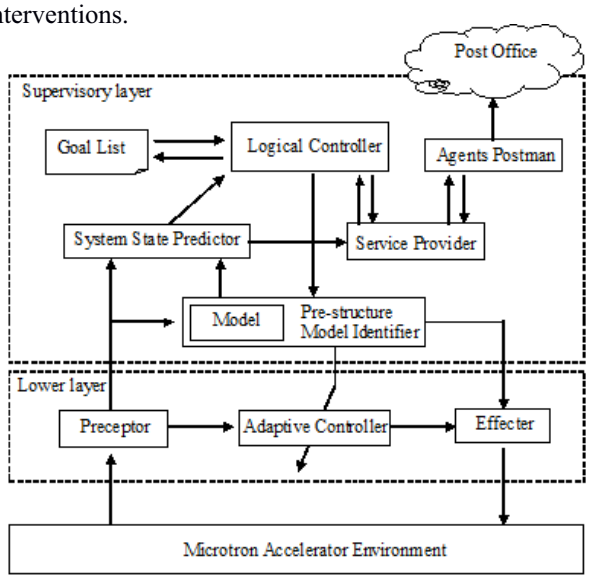


Fig. T.3.10: Microtron control agent architecture

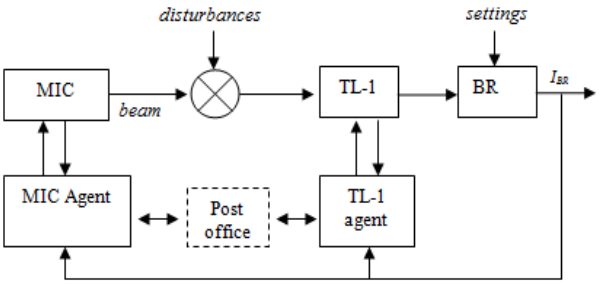


Fig. T.3.11: Block diagram for multi-agent based control of Microtron and TL-1

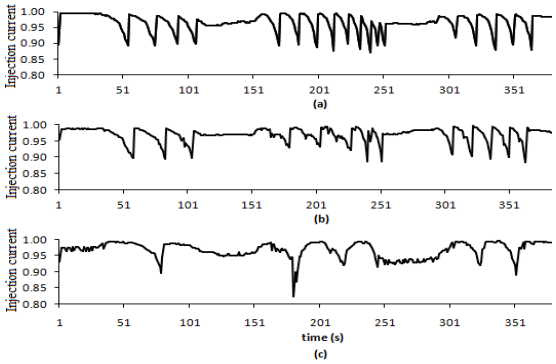


Fig. T.3.12: Normalized injection current in booster for, (a) scenario 1 (b) scenario 2 (c) scenario 3

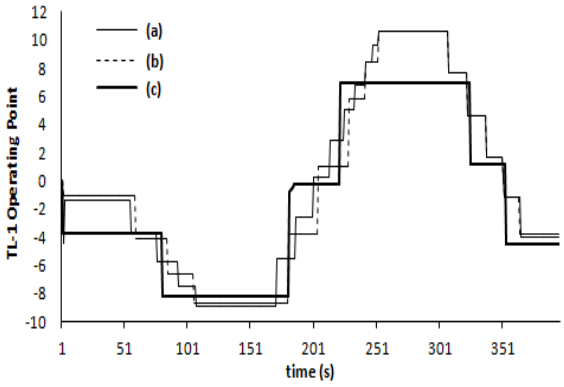


Fig. T.3.13: TL-1 operating points for which TL-1 was adjusted by TL-1 agent for, (a) scenario 1, (b) scenario 2, (c) scenario 3

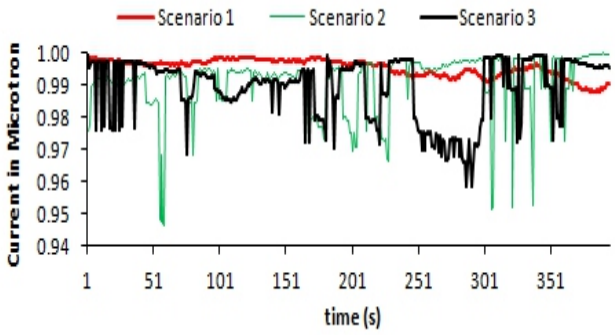
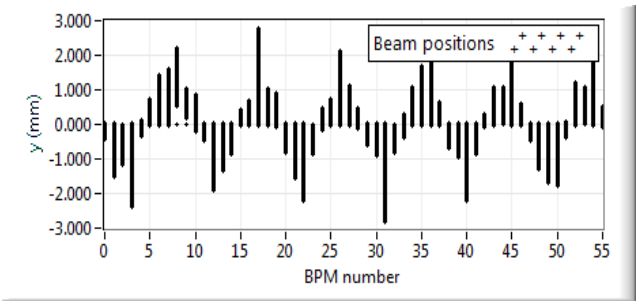


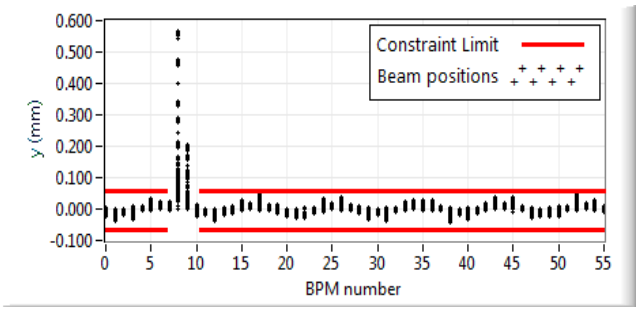
Fig. T.3.14: Beam current at microtron output when microtron is operated at different operating points by microtron agent under different scenarios

4.5 Multi-agent based control of beam orbit in SRS

To control beam orbit in synchrotron radiation sources, the distributed multi-agent based accelerator tuning framework is developed. Based on the commonly employed multilayer control system architectures for synchrotron radiation sources, the orbit control job is distributed to multiple, low complexity reactive agents that work simultaneously and control the local orbit for individual beam lines (BL) and insertion devices (ID) in an optimized manner. For each type of agent in the proposed multi-agent framework, agent architecture and design are formulated. This framework is then extended by adding the constraint gradient based reinforcement learning capability to the agents. Simulation results (Fig. T.3.15 and T.3.16) [11] show that with this type of scheme the accelerator on-line learning can be achieved with minimum disturbance to the nearby beam lines thereby improving the overall system performance.



(a)



(b)

Fig. T.3.15: Beam positions on different BPMs during the skill learning using (a) general gradient based reinforcement learning (b) constraint gradient based reinforcement learning

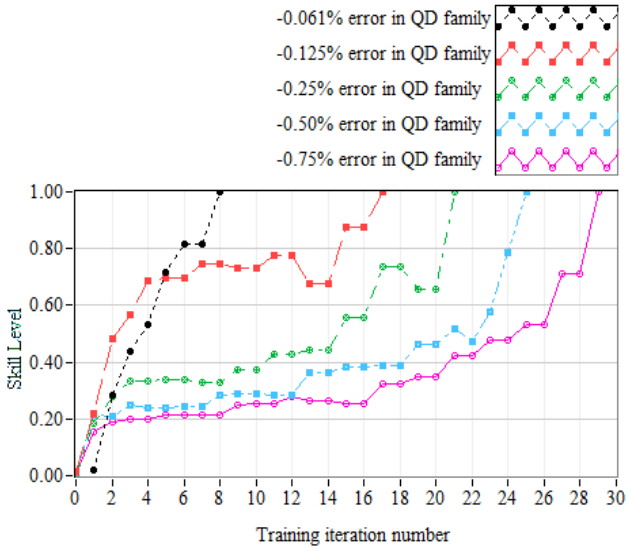


Fig. T.3.16: Skill learning by the agent (DP3 agent) for the case of machine operating point changes due to different error introduced in the defocusing quadrupole (QD) family

4.6 Related developments in the Indus-2 context

Based on the findings and using the developed accelerator components models and agent codes resulting from this work, two of the Indus-2 systems: Model Predictive Control (MPC) based SOFB and Agent Based Orbit Control System have been developed. The MPC controller calculates the corrector currents considering the instantaneous system states, future system states, corrector current and rate limits, future corrector saturation. The feature of predicting the feasible orbit against given desired orbit and the associated orbit moves are also provided. The customised version of multi-agent based control of beam in SRS (section 4.5) is also implemented and is in the process of testing at Indus-2. (Fig. T.3.17)

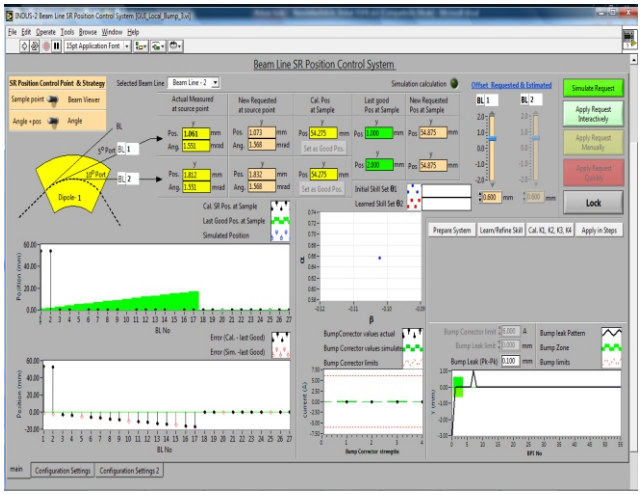


Fig. T.3.17: GUI of Agent based control system

5. Conclusions

The control system model of the accelerator facility is developed. For each of the subsystems, namely, electron source (pre-injector), transport line, booster and storage ring, the intelligent agents are formulated as per the specific subsystem requirements. The simulation results show that the agent based control can be successfully utilised for automatic tuning of individual accelerator subsystems to improve the overall injection efficiency under the dynamic behaviour of beam at source. With model based tracking concepts the agents based control improves the overall system availability for different sensor/actuator failure conditions. The multi-agent based cooperative tuning successfully optimise the injection current under dynamic system conditions. The distributed multi-agent based orbit control in synchrotron radiation sources can improve the beam reliability and can

significantly reduce the operator efforts and accelerator tuning time for providing beam to new beam lines. Based on the findings and using the developments resulting from this work, two of the Indus-2 systems, MPC based Slow Orbit Feedback System and Agent Based Orbit Control System for Indus-2 have been developed.

Acknowledgment

This work is a part of the Ph. D. thesis of the author, which was carried out under the guidance of Dr. P. V. Varde, BARC, Mumbai, Prof. P. S. V. Nataraj, IIT, Bombay and Shri P. Fatnani, RRCAT, Indore. The author is grateful for their valuable guidance and support. Author is also thankful to all the colleagues of ACSD and APS, who have helped in one way or the other during this work. Author would like to express gratitude to Shri C. P. Navathe, Shri P. R. Hannurkar, Shri G. Singh, Shri P. Shrivastava, Dr. S. B. Roy, Dr. A. Banerjee and Dr. P. D. Gupta for their continuous encouragement.

References

[1] N. R. Jennings *et al.*, Transforming Standalone Expert Systems into a Community of Cooperating Agents, *Int. Journal of Engineering Applications of Artificial Intelligence* Vol.6(4), pp.317-331,(1993)

[2] N. R. Jennings *et al.*, Using, ARCHON to develop real-world DAI applications for electricity transportation management and particle accelerator control, *IEEE Expert*, Vol.11, pp.64-70,(1996).

[3] C. Stern *et al.*, A Control System for Accelerator Tuning Combining Adaptive Plan Execution with Online Learning, *ICALEPCS*, 3-7 Nov 1997, Beijing, China, (1997)

[4] W. B. Klein *et al.*, A general purpose intelligent control system for particle accelerators. In Journal of *Intelligent and Fuzzy Systems*, New York: John Wiley (1999).

[5] N. J. Nilsson *et al.*, Teleo-Reactive Programs for Agent Control, *Journal of Artificial Intelligence Research* (1), pp.139-158,(1994)

[6] R. Pugliese *et al.*, Applying Intelligent System Concepts to Automatic Beamline Alignment, *ICALEPCS,* 3-7 Nov 1997, Beijing, China,(1997)

[7] D. Schirmer *et al.*, Electron Transport Line Optimization using Neuronal Networks and Genetic Algorithms, Proceedings of 9th European Particle Accelerator Conference *EPAC*, Edinburgh, Scotland, 26-30 June 2006,(2006)

[8] R.P. Yadav, *et al.*, Intelligent agent based control of INDUS TL-1, *InPAC*, February 15-18, 2011, Delhi, India, 2011.

[9] R. P. Yadav *et al.*, A multi-agent based control scheme for accelerator pre-injector and transport line for enhancement of accelerator operations, *Online journal Elixir Comp. Sci. & Engg.* Vol.44, pp.7405-7410,(2012).

[10] R. P. Yadav *et al.*, Model-based Tracking for Agent-based Control Systems in the Case of Sensor Failures, *International Journal of Automation and Computing,* Vol.9, No.6, pp.561-569,(2012).

[11] R.P. Yadav *et al.*, Intelligent agent based operator support and beam orbit control scheme for synchrotron radiation sources, *International Journal of Advanced Science and Technology,* Vol.52, pp.11-34,(2013).