

Multi-Agent Control of Variable Geometry Axial Turbo Compressors

Gwyn Morgan*, George Rzevski* and Paul Wiese**

* Department of Information Systems and Computing, Brunel University, West London

** Design and Innovation Department, The Open University, Milton Keynes

Abstract

A novel distributed intelligent system has been designed by authors to control aerodynamic instabilities caused by stall and surge in variable geometry axial turbo compressors. The multi-agent control system autonomously adjusts angles of individual vanes and optimises compressor efficiency.

Keywords – Multi-Agent Control Systems, Distributed Intelligent Control, Intelligent Geometry Compressors, Intelligent Mechatronics

1. Introduction

Axial turbo compressors are used in many applications where large quantities of air or gas have to be moved or compressed. Typical examples are in jet engines, industrial gas turbines, gas-line pumping and in many process plants. All turbo compressors are limited in their performance by the aerodynamic phenomena of stall and surge, where the flow of the gas becomes unstable and can reverse in direction. Stall and surge, if allowed to develop, can cause significant mechanical damage to the compressor.

The present design of axial compressors use fixed geometry rotor blades. Stator vanes have a limited capability to vary angles against pre-set limits, using simple control algorithms. The operating point for the compressor is selected to give an adequate *surge margin* therefore avoiding the possibility of stall or surge in operation. This surge margin puts a limit on the work that a given size of compressor can do, although the aerodynamic *efficiency* can be optimised at any given operating point. The avoidance of surge has important safety implications in aerospace, as surge under extreme manoeuvres has resulted in the loss of aircraft, the most public of which was the loss of the Russian TU 144 Supersonic Transport at the Paris Air show in 1973, with the loss of 14 lives.

Surge margins of 20 to 25% are not uncommon in aero gas turbines, resulting in larger and heavier engines than could be used if the operating line could approach the surge line, in a technique sometimes known as "surge riding". This weight and size penalty can be of the order of 10%, and smaller engines obviously have major benefits, particularly in the military field.

Supported by a group of industrial organisations and experienced turbo-machinery designers, the authors decided to look at the problem of aerodynamic instability in compressors in a novel way. The decision was made to reconsider the fundamentals of compressor design by removing the usual assumption of fixed or partially variable aerofoil geometry and to apply concepts from the intelligent network paradigm, as reported in [Rzevski 1998] and [Rzevski 2003].

A design of an axial compressor with variable geometry has emerged, where intelligent agents individually control each movable element. Agents are then connected into a network and empowered to negotiate among themselves the relative positions of the movable elements, with a view to achieving a performance as close as feasible to the optimum under continuously changing aerodynamic conditions. The overall behaviour of the compressor thus emerges from the interaction of the agents.

The proposed Intelligent Geometry Compressor (IGC) will operate by using sensors to monitor the aerodynamic conditions around each movable element and throughout the compressor. Sensor information will then be used by local agents, which through the process of negotiation, will make control decisions and instruct actuators to incrementally vary aerofoil geometry to adopt the flow path that ensures optimum performance for the current aerodynamic conditions. The Intelligent Geometry Compressor is thus a genuine mechatronic system comprising mechanical components characterised by variable geometry (e.g. stator vanes), sensors, actuators, digital hardware, software and embedded artificial intelligence.

The implications are staggering. Utilising embedded processing power it becomes feasible to design into the compressor the capability of:

- Self-diagnostics (monitoring compressor conditions and identifying faults when they occur),
- Self-repair by reconfiguration (isolating faulty parts and thus making them harmless)
- Graceful degradation of performance (repositioning remaining healthy parts to achieve a reduced but acceptable level of performance)

Complimentary aerodynamic studies are showing the benefits of this approach to compressor design, particularly in the area of highly rated variable operating cycle compressors.

2. Fundamental Concepts

The intelligent control system for variable geometry compressors developed by authors is fundamentally novel. To understand how it works requires familiarity with a number of concepts from subjects of Artificial Intelligence and Distributed Intelligent Systems. The most important concepts underlying the authors' approach are reviewed below.

2.1. Intelligence

It is useful to start by defining the concept of Intelligence with a view to distinguishing intelligent control from conventional automation.

Intelligence is the capability to achieve goals under conditions of uncertainty.

The uncertainty here encompasses

- The frequent occurrence of unpredictable *External Events*, (changes in the system environment, eg, load surges or sudden lack of pressure) and/or *Internal Events* (failures of system components)
- Lack of complete or reliable information about the problem that has to be solved, eg, of airflow through a compressor.

Intelligence is thus a highly desirable capability whenever there is a need to solve ill defined or dynamic problems characterised by uncertainty and is particularly useful for axial compressor controllers, which must cope with a considerable unpredictability of environmental conditions.

Let us explore the differences between intelligent systems and conventional automation.

The key feature of intelligent systems is that they are *Knowledge-Driven* rather than *Data-Driven*. When faced with uncertainty intelligent systems consult internally stored *Knowledge* and *decide* how to react. They have a *choice of behaviours* and a mechanism for selecting the most appropriate behaviour, which is called *Decision Mechanism*. Alternative behaviours are described by *Scripts (Rules)* and therefore selecting behaviour amounts to selecting a script that appears to be the most appropriate under circumstances. If the selected script proves to be inadequate, an intelligent system will re-visit its knowledge base and select a different script remembering which script worked and which did not. In contrast, unintelligent (automated) systems react to input data in a prescribed manner. When faced with uncertainty they stall.

A control system is called intelligent if it is capable of rapidly restoring desirable operating point of the controlled plant whenever an input changes in an unpredictable manner or a plant component fails. More over, to be described as intelligent a controller must be able to self-reconfigure and continue functioning even if some of its software components fail. Such a controller achieves its main goal (optimal plant operation) under conditions of uncertainty (unpredictable change in plant inputs or software/hardware failures).

2.2. Distributed Intelligence

Intelligent systems can be centralised or distributed. *Centralised Intelligent Systems* make all decisions at a single node. These systems have one hub that is responsible for all choices of behaviour. The structure of information flows to and from the decision-making hub is hierarchical. In contrast

Distributed Intelligent Systems are systems with decision-making elements distributed throughout the system and interconnected into a network.

There are many advantages of distributed intelligence, the main being the phenomenon of *Emergence*. When many intelligent elements interact, the resultant behaviour is far more powerful than the sum of behaviours produced by the same elements without interaction. Thus, we are able to obtain a very intelligent behaviour from a large number of rather primitive building blocks by organising them into a network and providing them with facilities to communicate with each other. To obtain benefits of emergence it is sufficient to distribute decision elements logically. In other words, as long as decision elements are interlinked into a network they will generate emergent behaviour even if they are placed together in the same physical location (eg, in the same room or on the same server). Further benefits can be obtained, however, by geographical distribution, that is, by placing decision elements as close as practical to sources of input data and to actuators that execute decisions (eg, on each movable element of a compressor).

Due to the phenomenon of emergence and the closeness of its decision elements to input data sources, distributed intelligent systems are far more responsive to unpredictable changes than

centralised systems. They are also amenable to scaling - for solving large problems we can build networks of decision-making networks.

2.3. Intelligent Agents and Multi-Agent Technology

Distributed intelligent systems can be effectively implemented using *Multi-Agent Technology*. What follows is a description of a particular brand of multi-agent technology, based on authors' research. Other researchers and developers have pursued other approaches and readers are encouraged to make comparisons.

Intelligent Agents (also called Smart Agents or Software Agents) are computer programs that are capable of accomplishing their goals under conditions of uncertainty by means of collaborating or competing with each other

As a rule, an agent is created when needed and it achieves its goals by analysing its current task, deducing, from the available knowledge, how the task can be solved, composing messages (for other agents or humans), sending messages to selected correspondents, receiving messages (from other agents or humans), interpreting received messages, deciding how to react to received messages and implementing decisions. When an agent accomplishes its mission it may be destroyed.

Sets of interacting agents are called Multi-Agent Systems (or Agent Swarms). Multi-agent systems solve problems in a stepwise fashion. Constituent agents negotiate among themselves how to make every step towards the goal. Note that performance of these systems depends primarily on the effectiveness of the interaction among constituent agents rather than on agents themselves. Therefore, systems consisting of a very large number of very simple agents engaged in a rich interaction with each other are particularly powerful and cost-effective. Such systems are called *High Granularity Multi-Agent Systems*. The comparison is often made with a colony of ants or a swarm of bees: each bee has only a limited intelligence and yet the swarm is capable of a complex behaviour resulting from the ability of bees to pass to each other useful information.

3. The Design of a Multi-Agent Control System

The general approach to the logical design of a multi-agent control system for an intelligent geometry compressor application involves a three-step process.

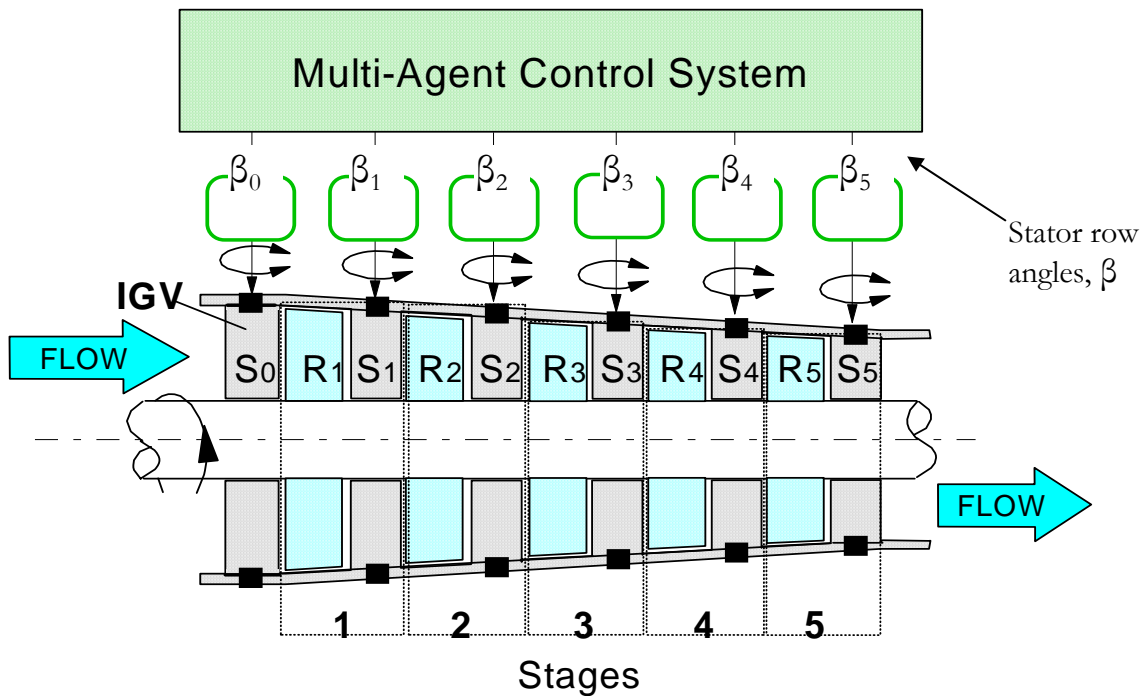
The first step is to identify the *constituent agents* of the system. This is done by analysing the goals and tasks involved in the compressor application, the related information requirements, and the physical constraints of the machine. The objective is to identify agents which can function autonomously i.e. are able to achieve goals independently. Some goals, however, may require the action of more than one agent therefore the second step is to consider the intended control strategies and thus clarify the *interaction* required between agents to achieve the respective goals. Specifically this means deciding methods of *co-operation* and the supporting modes of *communication*. The first two steps of the design process effectively determine the *architecture* of the multi-agent system. In the third step, the internal design of agents is addressed in terms of structure, modules and, ultimately, control rules and algorithms.

In practice, the above procedure is iterative and a number of potential solutions may emerge depending on the trade-offs made between each of the steps involved. This approach was applied by Morgan[2002] to the design of a multi-agent control system for a hypothetical IGC as briefly described in the following sections.

System Requirements

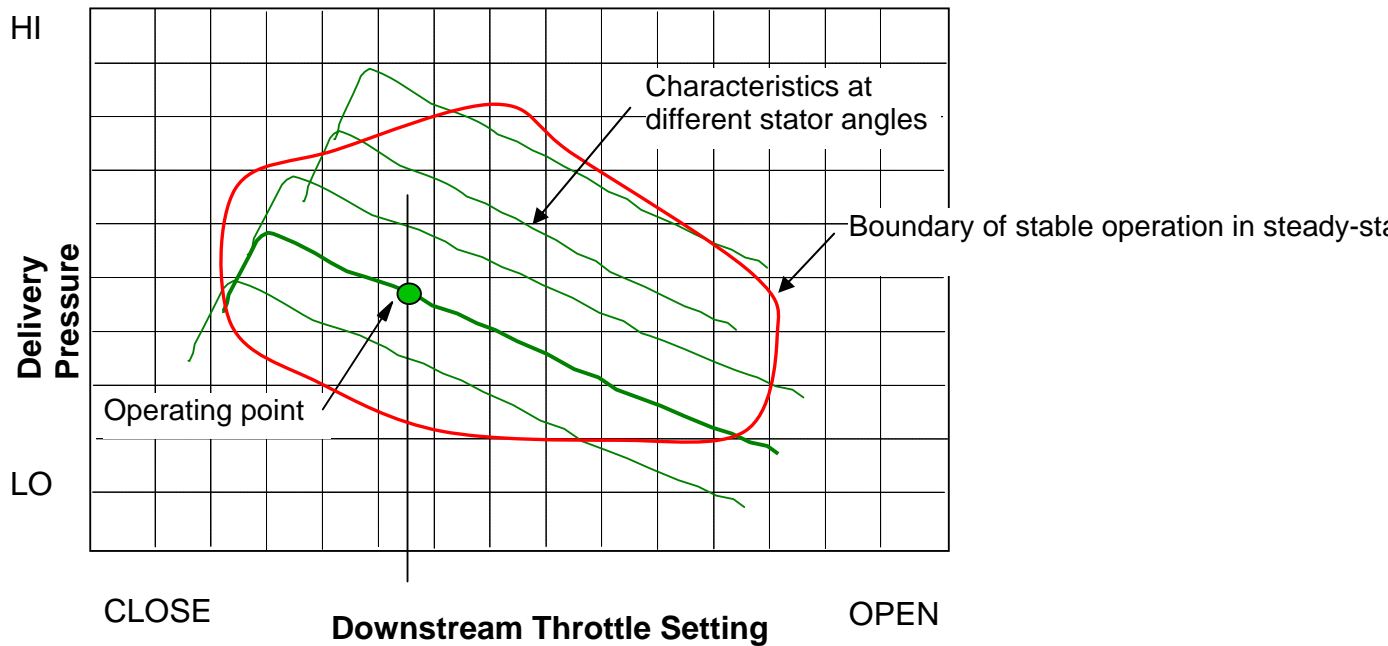
The hypothetical IGC is shown schematically in Fig 3.1. The machine arrangement is based on a 5-stage axial compressor with independently adjustable inlet guide and stator rows.

Fig 3.1 Hypothetical 5-stage IGC



For purposes of control system design only steady-state operation at constant rotational speed is considered. To represent the effects of a variable downstream load on the compressor an adjustable throttle valve having a linear pressure-flow characteristic is introduced into the overall flow system. The operating characteristics of the IGC in terms of delivery pressure vs throttle setting are illustrated in Fig 3.2. The location of each characteristic is determined by the angular positions of the stator rows and the operating point of the compressor is defined by the intersection of the throttle setting ordinate and the prevailing characteristic.

Fig 3.2 IGC Characteristics



The onset of stall determines the boundary of stable operation of the IGC. In the hypothetical machine, the onset of stall is represented by a stall margin variable which is assumed to be measurable for each blade row i.e. stator and rotor. The stall margin of a given row is a continuous, signed variable, which depends on the incidence of the inlet flow velocity vector and has a zero value at onset of stall and a maximum value of 1. Furthermore, the overall stall margin for the IGC is defined as the lesser value of all row stall margins. By definition, therefore, operating points on the boundary are points at which the overall stall margin is zero.

The multi-agent system is required to adjust the stator rows such that a required value of delivery pressure is achieved regardless of throttle setting but within the limit of stable operation. In this way, the operating range of the machine is maximised. In addition, it is required that stator row settings result in optimum machine performance at any sustained operating point. Optimisation typically means maximising the overall efficiency of the machine. The most important of these requirements is the avoidance of stall. The next priority is to achieve the required delivery pressure, or set-point, followed by optimisation.

Constituent Agents

A minimum set of agents is identified immediately based on a physical decomposition of the system (as advocated by Parunak et al []). This also satisfies the physical constraint that each stator row must be capable of independent adjustment. Thus a set of 6 'row agents' is introduced, one for each variable stator row, which carry out, at least, the basic task of setting the stator rows to the 'required' stagger angles. The need for other agents then depends on the extent to which the row agents are able to support the control tasks involved in achieving the system performance objectives. This is decided by the following task analysis.

a) Stator stall margin control

This objective is local to each stator row and requires adjustment of stagger in response to changes in the local flow conditions at the stator arising from changes in the setting of the throttle or of other adjustable stator rows. Stall margin control involves both ‘avoidance’ and ‘correction’. In the latter case response time is critical to minimise the extent and duration of excursion into stall conditions. For these reasons, the control action should be as direct as possible. Because both the control variable (stator stall margin) and the manipulated variable (stator row setting) are directly available then it is possible for row agents to execute this task autonomously. This would give the best performance and therefore the task of stator stall margin control is assigned to row agents.

b) Rotor stall margin control

The requirements for this objective are basically the same as for stator stall margin control and therefore it is again desirable for control action to be as direct as possible. Analysis of the flow through the compressor indicates that the stall margin for a rotor row is determined by the local axial flow velocity and the setting of the stator row immediately upstream. Therefore, given the availability of rotor stall margin information, the necessary control action can be performed, autonomously, by the row agent associated with the upstream stator row. However, should such action be in conflict with that for stator stall margin control then action by other row agents will be required to bring about a global change in axial flow velocity.

c) Delivery pressure control

This is a system-level objective which involves the collective action of all row agents in response to changes in delivery pressure set-point or throttle setting in order to achieve, or maintain, the required delivery pressure. It is desirable for control action to be fast in response to throttle changes but this is probably not important when dealing with set-point changes. During the control action it is not necessary to co-ordinate the relative adjustment of stator rows explicitly. Instead, the strategy follows that of Reiss and Blöcker [], in that changes to operating point are made as quickly as possible and then, once at the new operating point, an optimisation process is applied.

d) Optimisation

Due to the independent action of row agents it is expected that stator row settings would not necessarily be optimal following a change in operating point. The optimisation task therefore requires the determination and application of stator row settings which will maintain the current operating point but maximise the overall efficiency of the machine.

Morgan[] describes two methods for optimisation, one of which requires the availability of a direct measure of overall efficiency and introduces an *efficiency agent* into the system to facilitate the coordination of the row agents. The second method is based on the objective of maximising overall stall margin and is able to be carried out by row agents, concurrently, without the need for an additional agent.

Agent Interaction

The nature of the interaction between agents depends on the control strategies proposed for achieving the system objectives. From the task analysis of the previous section, the need for interaction arises in the context of rotor stall margin control and optimisation.

a) Agent Interaction for Rotor Stall Margin Control

The need for agent interaction arises if a row agent detects a conflict between the corrective action required for local stator stall control and that required for downstream rotor stall control. In such a case the row agent sends messages to all other row agents effectively requesting that they adjust their respective stator row settings so as to increase the stall margin of the particular rotor row in question. The row agent repeats the transmission of messages until the rotor row stall margin is once again of positive value. There is no need for the receiving agents to send any return message.

It is possible that conditions might arise in which more than one row agent detects such local conflict and thus other row agents will receive messages requesting help from more than one source. However, the action required of the receiving agent is the same regardless of the source of the request so this is of no consequence. Since the content of the message sent by a row agent in this situation is the same for all destinations then the appropriate mode of communication is *broadcast*.

b) Agent Interaction for Optimisation

When row agents have completed their control action following a change in operating point, their status is communicated to the efficiency agent which initiates a process of optimisation. Essentially this involves each row agent, in turn, making incremental adjustment of local stator setting as long as overall efficiency increases. The efficiency agent coordinates successive cycles of this process until no further increase in efficiency is detected. During this activity the modes of communication between agents is *direct point-to-point* and broadcast.

Multi-Agent System Architecture

Based on the preceding sections, the resulting system architecture for the hypothetical IGC application is an *agent network* comprising 6, similar, row agents. As indicated, an additional facilitator agent may be included depending on the strategy chosen for optimisation. The simplified system diagram below shows the agents and their communication links.

Agent Design

From an agent classification point of view the constituent agents of the above MAS are of the simplest type, being described by Jennings and Wooldridge[] as 'situated reactive' in that they are embedded in the system environment and react to sensed changes in that environment. The internal structure of agents follows the simple *perception, cognition, execution* model proposed by Rzevski[]. By way of example, the architecture of the row agent is described below.

Fig 3.3 Multi-Agent System Architecture

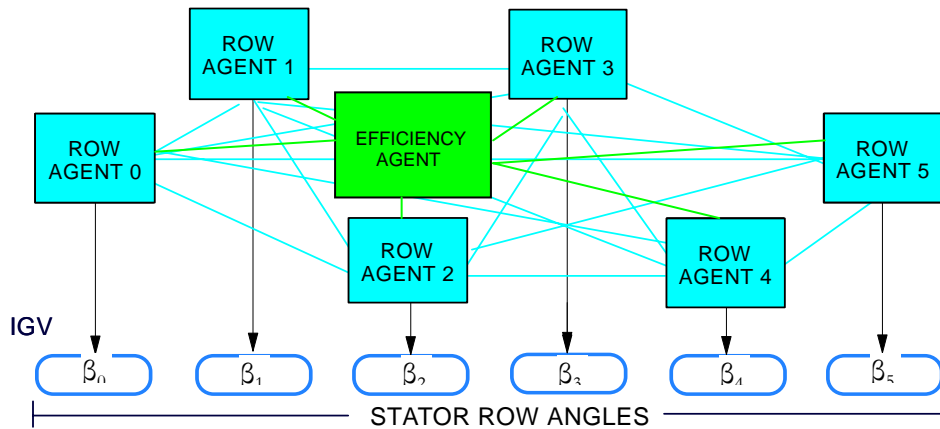
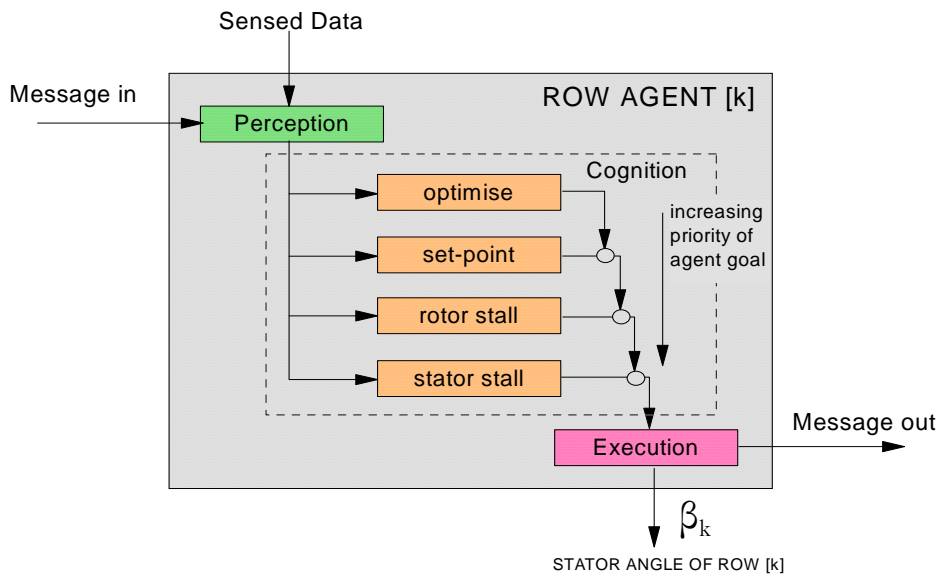


Fig 3.4 Row Agent Architecture



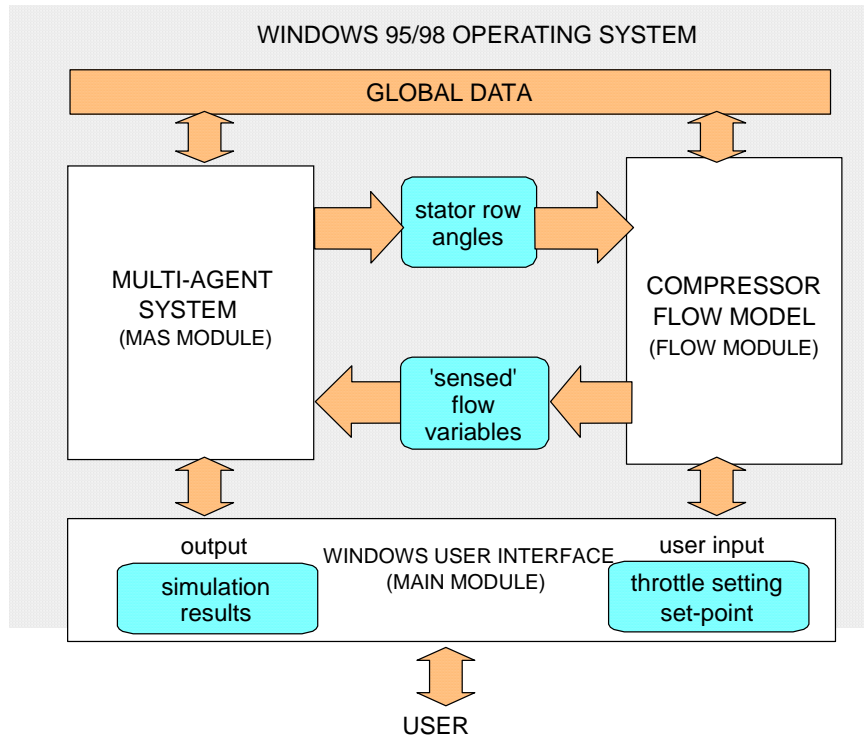
Row Agent Architecture

The internal structure of the row agent is shown below. The row agent is required to support several goals, and this, coupled with the decision to adopt a pre-determined order of priority for the goals, leads to a form of architecture similar to the *Subsumption* architecture proposed by Brooks[]. In this design the cognition module comprises a set of task modules each of which is invoked according to the assigned priority.

4. Simulation

Computer simulation is an essential part of the overall design process for the IGC to enable control strategies to be evaluated and the detail operation of the agent task modules identified above to be developed. For this purpose a Windows-based application program was developed using C++ which comprises three main modules as illustrated below.

Fig 4.1 Simulation Program Organisation



The main features of the simulation program and some of the results obtained are described in the following sections.

Agent Representation

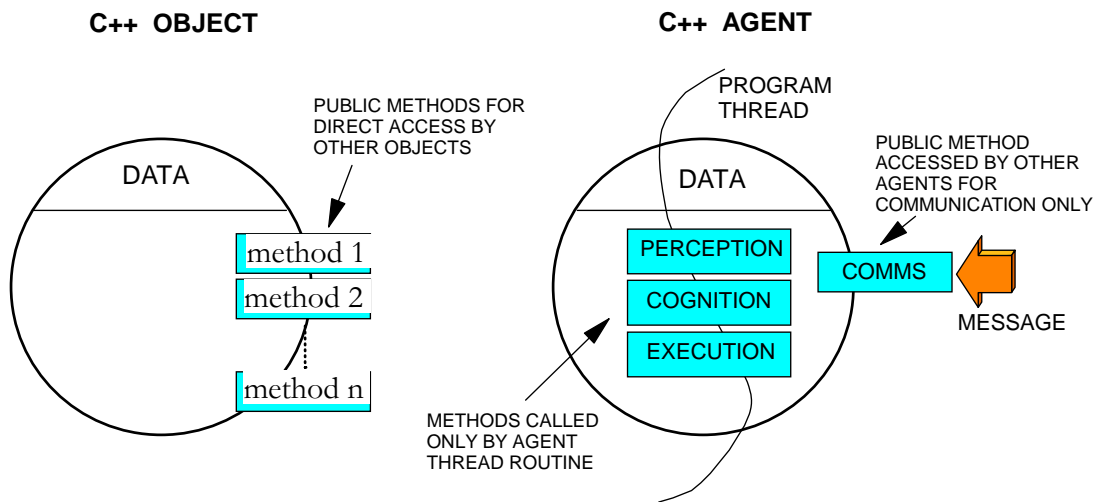
There is no facility in the C++ program language that enables an agent to be represented directly so it is necessary to define a software entity, in terms of the available program constructs, which can serve this purpose. Specifically, an entity is required which controls the choice and timing of its own actions and, in this way, appears to behave as an autonomous agent.

The highest level of abstraction available in C++ is the object and this enables a software entity to be defined that encapsulates data and methods. But the methods defined within an object class are 'public' i.e. they are available for execution by other objects or modules. In other words, an

object, unlike an autonomous agent, does not determine what particular action it takes or when this action occurs. However, if an object is instantiated in its own program thread and if the object class methods are only called by the associated thread routine then the behaviour of the object can become self-determining. For agent communication however, a public method may be used to allow agents direct access to other agents for purposes of passing messages.

The general form of agent representation described above is summarised in the following diagram which also emphasises the essential differences between an ‘agent’ and an ‘object’.

Fig 4.2 Agent Representation

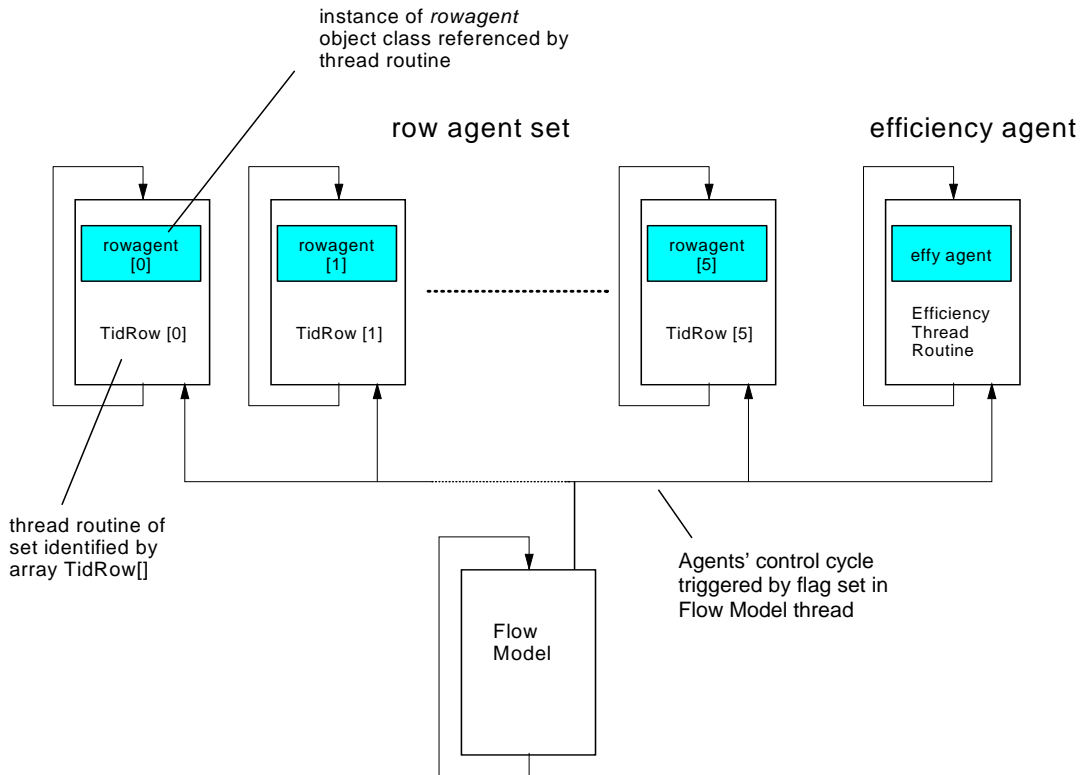


Multi-Agent System Representation

The objects associated with row agents are declared as a 6-element, 1-dimensional array of the object class *rowagent*. Similarly, the program thread identifiers are also declared as an array and this enables program threads to be created by means of a simple loop routine. On creation of each thread, a common thread routine is launched whose argument is the array index of the related thread identifier. This index is used in the thread routine to identify the particular instance of *rowagent* to be referenced when calling object methods. In this way, 6 independent row agents can be activated which run concurrently with one another and the flow model.

The overall implementation of the MAS may be visualised as shown in Fig 4.3 in which agents are depicted as instances of object classes referenced within separate concurrent program threads.

Fig 4.3 Representation of MAS in Simulation Program



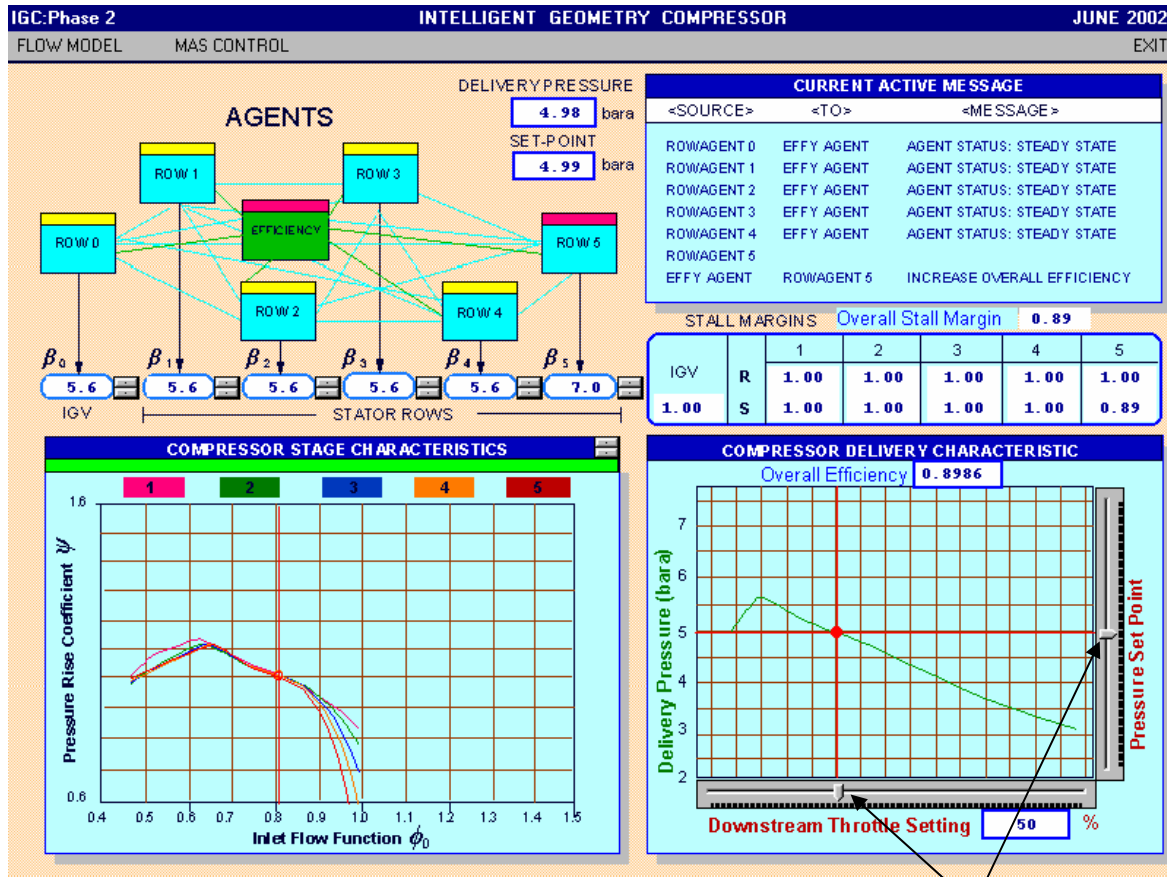
Compressor Flow Model

The flow model is based on a calculation routine for predicting steady-state compressor stage characteristics described by Howell and Bonham[]. This method was extended to take account of variable stator angle and application to a 5-stage machine. Full details of the analysis for the flow model are given by Morgan[]. The flow model runs cyclically, adopting the prevailing values for throttle setting and delivery pressure set-point as set by the user and the prevailing values of stator angles as set by the agents.

User Interface

The simulation program presents a single window 'work screen' which shows results of simulation in graphical and numerical form and allows the user to enter input data conveniently by means of trackbars and up-down controls.

Fig 4.4 Simulation Window



Main user controls

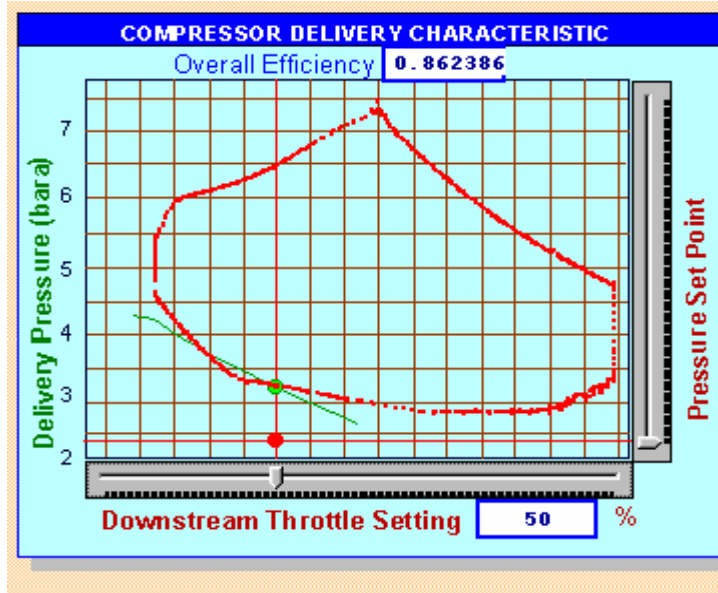
Simulation Results

Simulation trials demonstrated the effectiveness of the multi-agent control system in achieving the two main performance objectives of the intelligent geometry compressor.

a) Maximising operating range of IGC

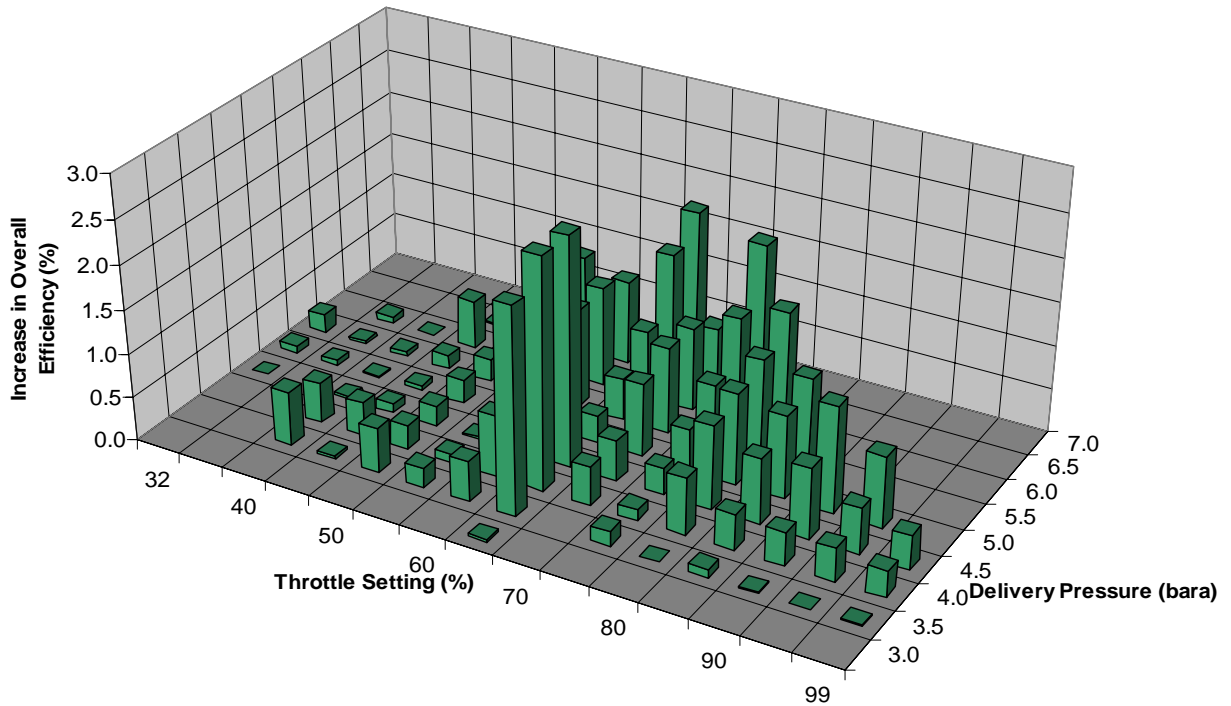
By incremental adjustment of the throttle settings at extreme values of delivery pressure set-point it is possible to determine the maximum operating envelope for the machine. The simulation program highlights any operating point at which the overall stall margin is approximately zero and so the operating envelope appears as a closed contour on the graph of delivery pressure vs throttle setting. The result of such a trial is shown in the screen image below.

Fig 4.5 Operating Range of IGC



b) Optimisation

The simulation results shown below indicate the effectiveness of agent optimisation. The chart shows the increase in overall efficiency that was achieved at operating points at which the stator settings were initially sub-optimal.



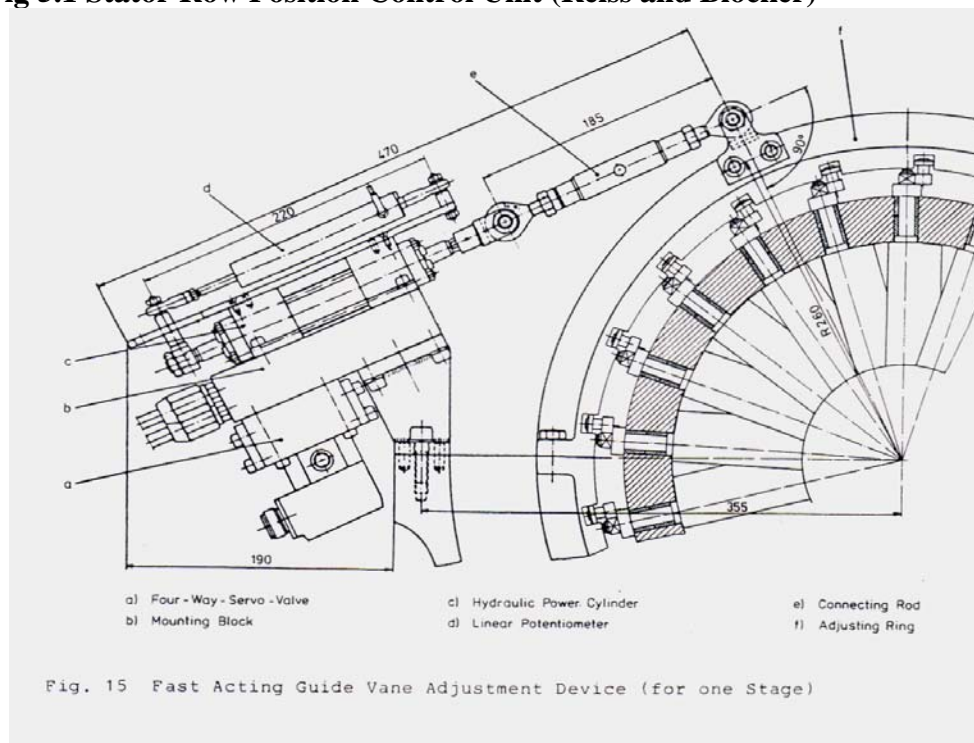
5. Implementation

The defining characteristic of a multi-agent control system is the concurrent, autonomous and cooperative behaviour of the constituent agents. Thus the choice of implementation must provide, at least, a dedicated computing resource for each agent and a means of communication between agents. For the IGC application considered in the previous sections two basic approaches to the implementation of the multi-agent control system are possible.

Single MAS Controller

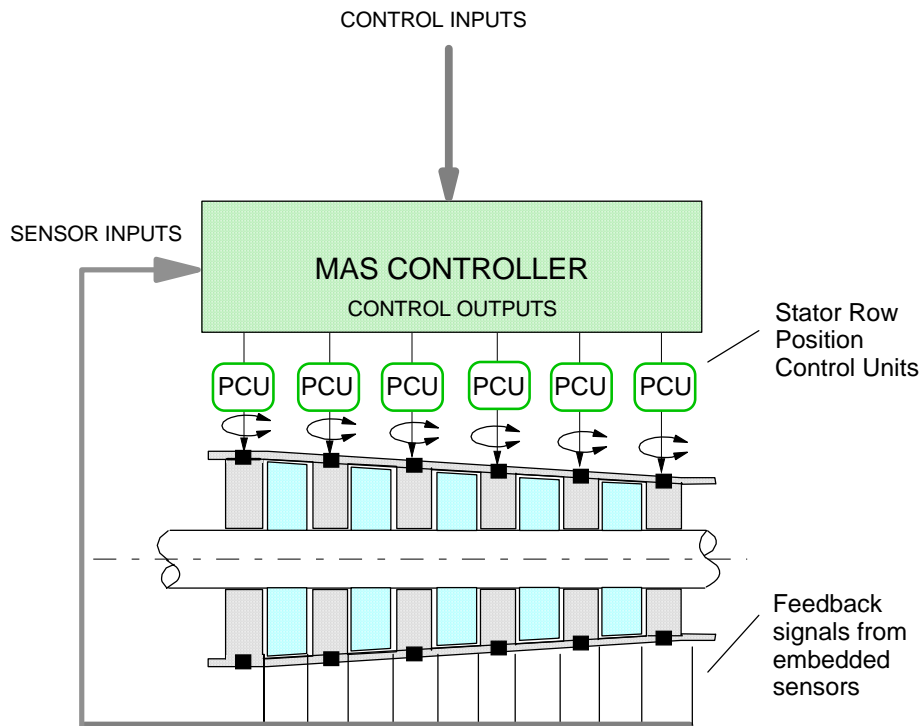
Firstly, the system could be implemented as a single control unit comprising a single processor supporting a suitable multi-tasking operating system. Each agent has its own program thread in much the same way as in the simulation program described above and also has dedicated I/O channels. Output signals from each agent channel connect to position control units located at each adjustable stator row. An example of an electro-hydraulic position control unit, as described by Reiss and Blöcker [], is shown in Fig 5.1.

Fig 5.1 Stator Row Position Control Unit (Reiss and Blöcker)



Sensors located in the compressor connect directly to the MAS controller and provide the necessary feedback of both 'local' and 'global' control variables to the agents. In this implementation agents are essentially *software* entities and may communicate through shared memory or other inter-thread communication modes supported by the operating system (Cohen and Woodring []). The overall system arrangement is shown schematically below.

Fig 5.2 Single MAS Control Unit Implementation

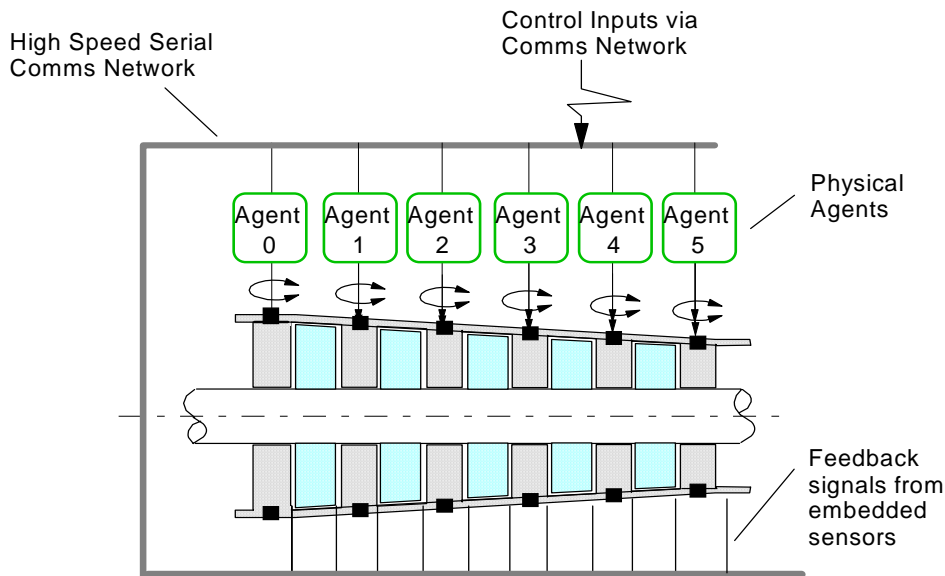


The main disadvantages of this implementation are the inherent limitations of multi-tasking and the physical remoteness of the agents from their respective application environments. The impact of these factors on system performance will depend on the particular application. However, there may be some cost advantage in applications involving existing variable geometry compressors.

Physically Distributed MAS

In the second approach, agents are conceived as separate *physical* entities and are integrated with the position control units of each adjustable stator row. In this case the agents are spatially, as well as logically, distributed. Each agent may be designed as a *mechatronic* unit incorporating all hardware, software and mechanisms to enable independent control of a stator row. Sensors in the local environment connect directly to the agent and may also be included as integral parts of the agent. A high speed serial bus provides a means of inter-agent communication. The simulation studies described earlier confirmed that it is not necessary to synchronise agent operation but it is necessary for “global” data to be consistent for all agents. A communications protocol such as the Bosch Controller Area Network (CAN) enables the required broadcast and point-to-point communication modes and ensures consistency of data at each node on the network. Smart sensors monitoring global variables could also connect to such a network to distribute data to the constituent agents. The general arrangement for this approach is shown below.

Fig 5.3 Distributed MAS Control



More sophisticated communications protocols could be implemented if required which adopt the CORBA standard (Weiss []) and enable inter-agent “conversations”.

This approach does not have the disadvantages of the single processor solution and may be expected to offer system performance advantages. Again, it would depend on the particular application as to any relative cost advantage. In addition, the physically distributed MAS is readily extended to systems involving large numbers of agents. For example, Morgan [] suggests this approach for implementing an IGC in which the vanes of all stator rows are individually adjustable resulting in a multi-agent control system of over 100 agents, one for each vane. In this case, of course, individual vane actuators are necessary.

6. Sensors and Actuators

The type and specification of the sensors to be used in an IGC is dependent on the type of geometry variation chosen. It is also heavily dependent on the type of aerodynamic disturbance to be sensed and the time constants involved. In the example described here, the angle of incidence of the leading edge of the stator blades can be measured by rotary potentiometers to the required accuracies. The aerodynamic influences are likely to necessitate measurement of small variations in static and total pressures, but again the technology required is largely available [Kurtz et al 2000]. In particular, the frequency response required is in the range 10 to 400 Hz, which is attainable with existing technologies. At this stage of the project, existing sensor technology is seen as largely adequate, but further detailed studies are currently being undertaken elsewhere and will be reported later.

The position on actuators is somewhat similar. Studies have shown that the change in incidence angle typified by change in stator vane angle can best be achieved by rotary hydraulic actuators, provided the required stiffness can be achieved. Unpublished work by [Miller 1994], based on both the Cranfield Low Speed and High Speed Compressors, has shown that the torque required to move an individual stator blade is small. In the case of the Low Speed Stage this is negligible (below 1 lb.in./0.1 Newton metres) and for the High Speed Compressor, in the region of 16 lb.in.(2 Newton metres) maximum. It has also been calculated that the frequency response required is again in the range 10 to 400 Hz. These figures are well within the range available from existing actuators, and there are numerous alternative means of actuation available, largely stemming from research work done within the UK on the EPSRC/DTI LINK research programme on The Design of High Speed Machinery, which has been adequately reported elsewhere [DTI 1997].

7. Conclusions

A novel intelligent controller has been proposed and shown, by simulation, that it is capable of improving the performance range as well as efficiency of axial compressors with variable geometry. The proposed intelligent-agent based control system is able to rapidly react to variations in the operating conditions of turbo machinery and move angles of groups of vanes, or even individual vanes, to compensate for the detected changes. The reported research is of course only an early stage of a long and intense study which will require resources beyond those that were available to authors of this paper. The results are sufficiently promising to justify further investments into research with a view to building a physical prototype of an intelligent geometry compressor.

It is important to note that the technology used for the design of the controller represents state of the art in artificial intelligence providing opportunities for further refinement and for a variety of implementations, including a physical distribution of agents throughout the compressor.

8. References

- Brooks, R.A. 1986. A Robust Layered Control System for a Mobile Robot. In *IEEE Journal of Robotics and Automation*, 2(1):14-23.
- Cohen, A. and Woodring, M. 1998. Win32 Multithreaded Programming. *O'Reilly & Associates Inc. CA, USA*.
- DTI 1997. *The Design of High Speed Machinery*. Department of Trade and Industry Publication 2671/6k/3/97/NP (Crown Copyright).
- Holland, JH 1998. *Emergence*. Oxford University Press.
- Howell, A. R. and Bonham, R. P. 1950. Overall and Stage Characteristics of Axial-flow Compressors. In: *Proceedings of I.Mech.E.*, volume 163, page 235. London.
- Jennings, N. R. and Wooldridge, M. 1998. *Agent Technology: Foundations, Applications and Markets*. Springer.
- Kurtz, A.D., Chivers, J.W., Ned, A.A. and Epstein, A.H. 2000. *Sensor Requirements for Active Gas Turbine Engine Control*: Unknown Conference paper.
- Miller, D 1994. *Vane Aerodynamic Loads at High and Low Speeds*. Unpublished internal paper for Ricardo Aerospace.
- Morgan, G 2002. *Intelligent Geometry Compressor: The Application of Multi-Agent System Technology to the Design of Intelligent Machines*. PhD Thesis, Brunel University.
- Paranuk, V.D., Baker, A. and Clark, S. 1998. The AARIA Agent Architecture: From Manufacturing Requirements to Agent-Based System Design. In: *Working Notes of the Agent-Based Manufacturing Workshop*, Minneapolis, MN.
- Riess, W. and Blöcker, U. 1987. Possibilities for On-line Surge Suppression by Fast Guide Vane Adjustment in Axial Compressors. In: *AGARD Conference Proceedings 421*, 69th symposium of PEP, pp. 31-1 to 31-13.
- Rzevski, G. 1995. *Designing Intelligent Machines Volume 1, Perception, Cognition and Execution*. Butterworth-Heinemann.
- Rzevski, G 1998. *Engineering Design for the Next Millennium: The Challenge of Artificial Intelligence*. The 86th Thomas Hawksley Memorial Lecture, IMechE.
- Rzevski, G 2003. *On Conceptual Design of Mechatronic Systems*. Mechatronics, Elsevier Science.
- Weiss, G. 1999. *Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence*. The MIT Press.