Journal of Intelligent and Robotic Systems **1**: 315–342, 1989.

Towards Intelligent Autonomous Control Systems: Architecture and Fundamental Issues

P.J. ANTSAKLIS, K.M. PASSINO
Department of Electrical and Computer Engineering, University of Notre Dame, Notre Dame, IN 46556, U.S.A.

and

S.J. WANG
Jet Propulsion Laboratory, MS 198-326, 4800 Oak Grove Drive, Pasadena, CA 91109

(Received: 6 July 1988; revised: 12 December 1988)

Abstract. Autonomous control systems are designed to perform well under significant uncertainties in the system and environment for extended periods of time, and they must be able to compensate for system failures without external intervention. Intelligent autonomous control systems use techniques from the field of artificial intelligence to achieve this autonomy. Such control systems evolve from conventional control systems by adding intelligent components, and their development requires interdisciplinary research. A hierarchical functional intelligent autonomous control architecture is introduced here and its functions are described in detail. The fundamental issues in autonomous control system modelling and analysis are discussed.

Key words. Automatic control theory, autonomous control, intelligent control, intelligent systems, hierarchical systems, hybrid systems, artificial intelligence.

1. Introduction

Autonomous control systems must perform well under significant uncertainties in the plant and the environment for extended periods of time and they must be able to compensate for system failures without external intervention. Such *autonomous* behavior is a very desirable characteristic of advanced systems. An autonomous controller provides high-level adaptation to changes in the plant and environment. To achieve autonomy, the methods used for control system design should utilize both (i) algorithmic-numeric methods, based on the state-of-the-art conventional control, identification, estimation, and communication theory, and (ii) decision-making-symbolic methods, such as the ones developed in computer science and specifically in the field of artificial intelligence (AI). In addition to supervising and tuning the control algorithms, the autonomous controller must also provide a high degree of tolerance to failures. To ensure system reliability, failures must first be detected, isolated, and identified, and subsequently a new control law must be designed if it is deemed necessary. The autonomous controller must be capable of planning the necessary sequence of control actions to be taken to accomplish a complicated task. It must be able to interface to other systems as well as with the operator, and it may need learning capabilities to enhance its performance while in operation.

Advanced planning, learning, and expert systems, among others, must work together with conventional control systems in order to achieve autonomy. The need for quantitative methods to model and analyze the dynamical behavior of such autonomous systems presents significant challenges well beyond current capabilities. It is clear that the development of autonomous controllers requires significant interdisciplinary research effort as it integrates concepts and methods from areas such as control, identification, estimation, and communication theory, computer science, especially artificial intelligence, and operations research.

In this paper, an autonomous controller architecture is introduced and discussed in detail. For such controllers to become a reality, certain fundamental questions should be studied and resolved first. These fundamental problems are identified, formulated and discussed, and future research directions are outlined. Next, the focus of this paper is established and a detailed description of the results is given.

Autonomous controllers can be used in a variety of systems from manufacturing to unmanned space, atmospheric, ground, and underwater exploratory vehicles. In this paper, we develop an autonomous controller architecture for future space vehicles. Referring to a particular class of control problems has the advantage that the development addresses relatively well-defined control needs rather than abstract requirements. Furthermore, the autonomous control of space vehicles is highly demanding; consequently the developed architecture is general enough to encompass all related autonomy issues. Future space vehicles must be capable of autonomous operation to accomplish their missions. Emerging aeromaneuvering vehicles such as the Aeroassisted Orbital Transfer Vehicle and the Aerospace Plane will be required to maneuver at high altitudes and hypersonic velocities in a flight regime characterized by significant uncertainty in atmospheric density and aerodynamic characteristics. Uncertainty in these parameters may cause significant deviation from the nominal trajectory, conceivably leading to the loss of the vehicle. Significant time and communication constraints during the atmospheric flight dictate that the vehicles should perform autonomously for extended periods of time since pilot or ground support intervention may not be possible. Future space systems, such as manned space platforms, contain significant flexible structural components. Model uncertainties and system parameter variations require advanced adaptive control techniques to meet stability and performance specifications. An autonomous adaptive control system is needed to deal with gross fundamental and environmental changes in the system. For space systems these include hardware failures, docking disturbances, payload articulation, and man-motion disturbances.

It should be stressed that all the results presented here apply to any autonomous control system. In other classes of applications, the architecture, or parts of it, can be used directly and the same fundamental concepts and characteristics identified here are valid.

The architecture of autonomous controllers necessary for the operation of advanced planetary and aeromaneuvering space vehicles is developed here. The concepts and methods needed to successfully design such an autonomous controller

are introduced and discussed. A hierarchical functional autonomous controller architecture is described in detail; it is designed to ensure the autonomous operation of the control system and it allows interaction with the pilot/ground station and the systems on board the autonomous vehicle. Note that a shorter version of these results has appeared in [4].

Section 2 gives a brief history of the development of control systems to motivate the necessity for autonomous controllers. The functions, characteristics, and benefits of autonomous control are outlined. Next, it is explained that plant complexity and design requirements dictate how sophisticated a controller must be. From this it can be seen that often it is appropriate to use methods from operations research or AI to achieve autonomy. Such methods are studied in *intelligent control theory*. An overview of some relevant research literature in the field of intelligent autonomous control is given together with references that ouline research directions. In Section 3, an autonomous control functional architecture for future space vehicles is introduced. The controller is hierarchical, with three levels, the execution level (lowest level), the coordination level (middle level), and the management and organization level (highest level). The general characteristics of the overall architecture, including those of the three levels are explained, and an example to illustrate their functions is given. In Section 4, fundamental issues and attributes of intelligent autonomous system architectures are described. An approach to the quantitative, systematic modelling, analysis, and design of autonomous controllers is discussed. It is a 'hybrid' approach since it is proposed to use both conventional analysis techniques based on difference and differential equations, together with new techniques for the analysis of systems described with a symbolic formalism such as finite automata. The more global, macroscopic view of dynamical systems, taken in the development of autonomous controllers, suggests the use of a model with a hybrid or nonuniform structure, which in turn requires the use of a hybrid analysis. Finally, some concluding remarks are given in Section 5.

2. Conventional and Intelligent Autonomous Control Systems

Autonomous means having the power for self government. *Autonomous controllers* have the power and ability for self-governance in the performance of control functions. They are composed of a collection of hardware and software, which can perform the necessary control functions, without external intervention, over extended time periods. To achieve autonomy, the controller must be able to perform a number of functions in addition to the conventional control functions such as tracking and regulation. These additional functions, which include the ability to tolerate failures, are discussed later in this section.

There are several *degrees of autonomy*. A fully autonomous controller should perhaps have the ability to even perform hardware repair, if one of its components fails. Note that conventional fixed controllers can be considered to have a low degree of autonomy since they can only tolerate a restricted class of plant parameter

variations and disturbances. The autonomous controller architecture given in the next section provides the functions to attain a high level of autonomy. It can interface with both the crew, ground station and the on-board systems of the space vehicle. A command by the pilot or the ground station is executed by dividing it into appropriate subtasks which are then performed by the controller. The controller can deal with unexpected situations, new control tasks, and failures within limits. To achieve this, high-level decision-making techniques for reasoning under uncertainty and taking actions must be utilized. These techniques, if used by humans, are attributed to *intelligent* behavior. Hence, one way to achieve autonomy is to utilize high-level decision-making techniques, 'intelligent' methods, in the autonomous controller. *Autonomy is the objective, and 'intelligent' controllers are one way to achieve it.* The field of artificial intelligence [10, 56] and operations research offer some of the tools to add the higher level decision making abilities.

Autonomous controllers are evolutionary and not revolutionary. They evolve from existing controllers in a natural way fueled by actual needs, as it is now discussed.

2.1. DESIGN METHODOLOGY – HISTORY

Conventional control systems are designed using mathematical models of physical systems. A mathematical model which captures the dynamical behavior of interest is chosen and then control design techniques are applied, aided by CAD packages, to design the mathematical model of an appropriate controller. The controller is then realized via hardware or software and it is used to control the physical system. The procedure may take several iterations. The mathematical model of the system must be 'simple enough' so that it can be analyzed with available mathematical techniques, and 'accurate enough' to describe the important aspects of the relevant dynamical behavior. It approximates the behavior of a plant in the neighborhood of an operating point.

The first mathematical model to describe plant behavior for control purposes is attributed to J.C. Maxwell who in 1868 used differential equations to explain instability problems encountered with James Watt's flyball governor; the governor was introduced in 1769 to regulate the speed of steam engine vehicles. Control theory made significant strides in the past 120 years, with the use of frequency domain methods and Laplace transforms in the 30s and 40s and the introduction of the state space analysis in the 60s. Optimal control in the 50s and 60s and stochastic, robust and adaptive control methods in the 60s to today have made it possible to control more accurately significantly more complex dynamical systems than the original flyball governor.

The control methods and the underlying mathematical theory were developed to meet the ever-increasing control needs of our technology. The evolution in the control area was fueled by three major needs:

(i) The need to deal with increasingly complex dynamical systems.
(ii) The need to accomplish increasingly demanding design requirements.

(iii) The need to attain these design requirements with less precise advanced knowledge of the plant and its environment, that is, the need to control under increased uncertainty.

The need to achieve the demanding control specifications for increasingly complex dynamical systems has been addressed by using more complex mathematical models such as nonlinear and stochastic ones, and by developing more sophisticated design algorithms for say, optimal control. The use of highly complex mathematical models, however, can seriously inhibit our ability to develop control algorithms. Fortunately, simpler plant models, for example linear models, can be used in the control design; this is possible because of the feedback used in control which can tolerate significant model uncertainties. Controllers can then be designed to meet the specifications around an operating point, where the linear model is valid and then via a scheduler a controller emerges which can accomplish the control objectives over the whole operating range. This is, for example, the method typically used for aircraft flight control. In autonomous control, we need to significantly increase the operating range of the plant. We must be able to deal with significant uncertainties in models of increasingly complex dynamical systems in addition to increasing the validity range of our control methods. This will involve the use of intelligent decision-making processes to generate control actions so that a performance level is maintained, even though there are drastic changes in the operating conditions.

There are needs today that cannot be successfully addressed with the existing conventional control theory. They mainly pertain to the area of uncertainty. Heuristic methods may be needed to tune the parameters of an adaptive control law. New control laws to perform novel control functions should be designed while the system is in operation. Learning from past experience and planning control actions may be necessary. Failure detection and identification is needed. These functions have been performed in the past by human operators. To increase the speed of response, to relieve the pilot from mundane tasks, and to protect operators from hazards, autonomy is desired. It should be pointed out that several functions proposed in later sections, to be part of the autonomous controller, have been performed in the past by separate systems; examples include fault trees in chemical process control for failure diagnosis and hazard analysis, and control system design via expert systems.

2.2. FUNCTIONS OF AN AUTONOMOUS CONTROLLER

There are certain functions, characteristics, and behaviors that autonomous systems should possess [62, 20]. These are outlined below. Some of the important characteristics of autonomous controllers are that they relieve humans from time consuming mundane tasks thus increasing efficiency, enhance reliability since they monitor health of the system, enhance performance, protect the system from internally induced faults, and they have consistent performance in accomplishing complex tasks.

There are autonomy guidelines and goals that should be followed and sought after in the development of an autonomous system. Autonomy should reduce pilot/crew/ground station work load requirements for the performance of routine functions. The gains due to autonomy would be superficial if the maintenance and operation of the autonomous controller taxed the operators. Autonomy should enhance the functional capability of the future space vehicle. Since the autonomous controller will be performing the simpler routine tasks, men will be able to dedicate themselves to even more complex tasks.

There are certain autonomous system architectural characteristics that should be sought after in the design process. The autonomous control architecture should be amenable to evolving future space vehicle needs and updates in the state of the art. The autonomous control architecture should be functionally hierarchical. Highest authority lies nearest the pilot, crew, or ground station; for lower level subsystems to take some actions, they have to clear it with a higher level authority. The system must, however, be able to have the lowest level subsystems, that are monitoring and reconfiguring for failures, act autonomously to enhance system safety.

There are also certain operational characteristics of autonomous controllers. Ground controllers and/or the pilot or crew should have ultimate supervisory override control of future space vehicle autonomy functions. Autonomous activities should be highly visible, 'transparent', to the ground controllers and the flight crew to the maximum extent possible.

Finally, there must be certain features inherent in the autonomous system design. Autonomous design features should prevent failures that would jeopardize the overall space vehicle mission goals or safety. These features should enhance crew safety, and avoid false alarms and unnecessary hardware configuration. This implies that the controller should have self-test capability. Autonomous design features should also be tolerant of transient errors, they should not degrade the reliability or operational lifetime of future space vehicle functional elements, they should include adjustable fault detection thresholds, avoid irreversible state changes, and provide protection from erroneous or invalid external commands.

2.3. INTELLIGENT AUTONOMOUS CONTROL

The necessity for a succession of increasingly complex control systems from classical to adaptive and intelligent control, to meet the ever increasing performance requirements on the current and future complex dynamical systems, is described. The basic elements of intelligent controllers are highlighted and an outline of the relevant research on intelligent control is given.

2.3.1. *Motivation: Sophistication and Complexity in Control*

The complexity of a dynamical system model in terms of determinism, nonlinearities, etc., and the increasingly demanding closed loop system performance requirements,

necessitate the use of more complex and sophisticated controllers. For example, highly nonlinear systems normally require the use of more complex controllers than low order linear ones when goals beyond stability are to be met. The increase in uncertainty, which corresponds to the decrease in how well the problem is structured or how well the control problem is formulated, and the necessity to allow human intervention in control, also necessitate the use of increasingly sophisticated controllers. Controller complexity and sophistication is then directly proportional to both the complexities of the plant model and of the control design requirements.

Based on these ideas, [49, 24] suggest a hierarchical ranking of increasing controller sophistication on the path to *intelligent* controls. At the lowest level, deterministic feedback control based on conventional control theory is utilized for simple linear plants. As plant complexity increases, such controllers will need for instance, state estimators. When process noise is significant, Kalman filters may be needed. Also, if it is required to complete a control task in minimum time or with minimum energy, optimal control techniques are utilized. When there are many quantifiable, stochastic characteristics in the plant, stochastic control theory is used. If there are signficant variations of plant parameters, to the extent that linear robust control theory is inappropriate, adaptive control techniques are employed. For still more complex plants, self-organizing or learning control may be necessary.

At the highest level in their hierarchical ranking, plant complexity is so high, and performance specifications so demanding, that intelligent control techniques are used. The plant is so complex that it is either inappropriate or impossible to describe it with conventional system models such as differential equations. For instance, even though it might be possible to accurately describe some system with very complex nonlinear differential equations, it may be inappropriate if this description makes subsequent analysis too difficult.

The complexity of the plant model necessary for design depends on both the complexity of the physical system and on how demanding the design specifications are. There is a tradeoff between model complexity and our ability to perform analysis on the system via the model. However, if the performance specifications are not too demanding, a more abstract model can be utilized, which will make subsequent analysis simpler. This model intentionally ignores some of the system characteristics, specifically those that need not be considered in attempting to meet the particular performance specifications. Often, to obtain an abstract model, high level *symbolic* representations are utilized [60, 42, 33, 32, 46]. The choice of the modelling technique affects most aspects of analysis and design of a controller for the system; consequently, special control methodologies must be used with the abstract models. Such methodologies include advanced decision making techniques from the field of AI, which are used to reason over these representations and decide what control actions are appropriate to take. Since the AI techniques generally model the human decision-making processes, about what actions to take next, they can easily provide for human interface.

It is perhaps of interest to notice that all controllers in the hierarchy described above can be considered to be a type of *problem solving system*. (For an overview of the theory of problem solving see [56].) This is because there is a desirable goal behavior and the problem solver generates actions to change an initial undesirable behavior to the goal. It is our view that problem solving systems can be classified into two categories, conventional and AI. Several characteristics distinguish these two classes of problem solving systems. The conventional problem solving system is numeric-algorithmic, it is somewhat inflexible, it is based on the well developed theory of algorithms or differential equations, and it can thus be studied using a variety of methodical modelling, analysis, and design techniques. Classical control systems are an example of conventional problem solving systems. An AI problem solving system is a symbolic decision-maker, it is flexible with graceful performance degradation, and it is based on formalisms which are not well developed; actually there are very few systematic mathematical modelling, analysis, and design techniques for these systems. AI expert and planning systems are examples of AI problem solving systems. When comparing the characteristics of AI and non-AI systems, one can make the following observations: The decision rate in conventional systems is typically higher than that of AI systems. The abstractness and generality of the models used in AI systems is high compared with the fine granularity of models used in conventional systems. Symbolic representations, rather than numeric, are used in AI systems. High-level decision-making capabilities similar to those of humans exist in AI systems to a much greater extent than in conventional systems. The result is that a higher degree of autonomy exists in AI systems than in conventional ones.

In the hierarchical ranking of increasingly sophisticated controllers described above, the decision to choose more sophisticated control techniques is made by studying the control problem using a controller of a certain complexity belonging to a certain class. When it is determined that the class of controllers being studied (e.g., adaptive controllers) is inadequate to meet the required objectives, a more sophisticated class of controllers (e.g. intelligent controllers) is chosen. That is, if it is found that certain higher level decision-making processes are needed for the adaptive controller to meet the performance requirements, then these processes can be incorporated via the study of intelligent control theory. These intelligent autonomous controllers are the next level up in sophistication. They are *enhanced adaptive controllers*, in the sense that they can adapt to more significant global changes in the vehicle and its environment than conventional adaptive controllers, while meeting more stringent performance requirements.

One turns to more sophisticated controllers only if simpler ones cannot meet the required objectives. Below we list some of the reasons why it is necessary to use intelligent autonomous control for future space vehicles:

(i) Future space vehicles will be increasingly complex. Some characteristics that are needed in the model used to design their controller can only be described by symbolic representation techniques.

(ii) Control functions normally performed by the pilot, crew, or ground station must be incorporated into the controller for autonomous operation. Therefore, expert personnel's control decisions will have to be automated.

(iii) Human intervention in the control process should be allowed. A facility to interrupt the autonomous operation of the controller in case of design objective changes or controller failures should be included.

The need to use intelligent autonomous control stems from the need for an increased level of autonomy in achieving complex control tasks. In the next section a number of intelligent control research results which have appeared in the literature are outlined.

2.3.2. *Intelligent Autonomous Control: A Literature Overview*

The field of intelligent autonomous control is new. Some of the recent research efforts have been reported in the Proceedings of the 1985 Workshop on Intelligent Control, 1986 Intelligent Autonomous Systems Conference, the Space Telerobotics Workshop, and the Proceedings of the 1987 Symposium on Intelligent Control, and a wealth of useful references can be found there. Research that had a direct influence on our work is outlined below.

Intelligent controllers are hierarchical and the theory of hierarchical systems is relevant [36]. This work sets some of the fundamental concepts in intelligent control such as the need for varying degrees of abstractness in models used at the different levels in the controller. It also presents a theory of coordination for all subsystems of the intelligent controller. Coordination issues are also examined in [12]. The work in [18] extends Mesarovic's work. Fundamentals of intelligent systems such as the principle of increasing intelligence with decreasing precision, granularity, time scale density, model abstractness are discussed in [53, 54, 37]; the need for the integration of techniques from AI, operations research and conventional control theory to perform intelligent control tasks is also discussed there. The integration of AI and control theoretic methods is discussed in [13, 22, 11].

In [20] the authors explain how a wide variety of AI techniques will be useful in enhancing space station autonomy, capability, safety, etc. This project oriented book points to relevant AI techniques, research areas, and progress in solving the posed problems. In the Space Telerobotics Workshop, the work of several researchers from NASA, the Jet Propulsion Laboratory, Ames, Johnson Space Center [55, 21, 30, 6] and others, e.g., [57], is outlined and specific details of their programs and research directions are given. In [62] a detailed study of characteristics of autonomous space systems is given and an architecture for the complete autonomous operation of the space station is presented; examples are used to illustrate the behavior of the autonomous system.

There has been much work on developing intelligent controllers for robots. A good overview is given in [51]. The work in [7] describes an initial effort towards a

hierarchical intelligent controller based on AI planning methods. Balaram developed an architecture for the planning system that incorporates intelligent control fundamentals and that is accurately structured for his control task. Other intelligent controllers that use planning techniques are given in [26, 25, 8, 15, 16, 47]. The vision problem for intelligent controllers is examined in [17].

The work by Saridis and Valvanis in [49–54], and [63–66] probably represents the most complete mathematical approach to the analysis of intelligent machines. They stress a three level hierarchy for intelligent systems with execution, coordination, and management levels, and the "principle of increasing intelligence with decreasing precision". They use entropy as an unified quantification of disorder in each of the three levels in their intelligent system. In an intelligent controller, they choose the control action that will decrease the entropy in their system.

Other important work in the field of intelligent control is given in [1, 2, 37–40, 29, 19, 67]; a nested hierarchical controller is described in [38, 68]; some similarities between planning and intelligent control are given in [28]; and an interesting blackboard architecture is studied in [9]. Supervisory control with distributed intelligence is examined in [69], and heuristic control of real time processes is reported in [48]. The intelligent restructurable controls problem for aircraft was studied in [61, 58, 43]. The fault detection and identification problem in an intelligent controller was examined in [27, 44–45].

There have been numerous studies on the use of expert systems to control various process. For instance, in [23] the authors use an expert system in the adaptive control of large space structures. In [5] expert systems have been used in chemical process control and the term 'expert control' has been coined. There are interesting relationships between the type of problems examined in intelligent autonomous control, 'fuzzy control' [71], and 'automated reasoning' [70]. There exist numerous books and articles on these topics. As reported in the above literature, several limited versions of autonomous systems have been constructed. There are many sorts of robots and vehicles, each achieving a certain level of autonomy. They have been used for manufacturing [34], underwater exploration [41], and other applications.

A detailed functional architecture for autonomous controllers is the essential first step in their development. Such an architecture is introduced in the next section. It will show how to combine intelligent functions in a controller to achieve autonomy. Based on this architecture, the fundamental concepts and methods that need to be developed are identified.

3. An Intelligent Autonomous Control Architecture for Future Space Vehicles

In this section, a fundamental architecture of an autonomous controller for future space vehicles is introduced and discussed. This hierarchical architecture has three levels, the execution level, the coordination level, and the management and organization level. The functions of each level are described in detail. The architecture exhibits

certain characteristics, as discussed below, which have been shown in the literature to be necessary and desirable in autonomous systems. Based on this architecture we identify the important fundamental issues and concepts that are needed for an autonomous control theory.

3.1. ARCHITECTURE OVERVIEW: STRUCTURE AND CHARACTERISTICS

The overall functional architecture for an autonomous controller is given by the architectural schematic of Figure 1. This is a functional architecture rather than a hardware processing one, therefore it does not specify the arrangement and duties of the hardware used to implement the functions described. Note that the processing architecture also depends on the characteristics of the current processing technology; centralized or distributed processing may be chosen for function implementation depending on available computer technology.

The architecture in Figure 1 has three levels. At the lowest level, the execution level, there is the interface to the vehicle and its environment via the sensors and actuators. At the highest level, the management and organization level, there is the interface to the pilot and crew, ground station, or onboard systems. The middle level, called the coordination level, provides the link between the execution level and the management level. Note that we follow the somewhat standard viewpoint that there are three major levels in the hierarchy. It must be stressed that the system may have more or fewer levels. For instance, see the architecture developed in [62]. Some characteristics of the system which dictate the number of levels are the extent to which the operator can intervene in the system's operations, the degree of autonomy or level of intelligence in the various subsystems, the dexterity of the subsystems, and the hierarchical characteristics of the plant.

The sensors and actuators are implemented mainly with hardware. They are the connection between the physical system and the controller. Software and perhaps hardware are used to implement the execution level. Mainly software is used for both coordination and management levels. Note that the multiple copies of the different levels reflect the distinct character of the various control functions necessary to achieve autonomy. For example, there may be one control manager which directs a number of different adaptive control algorithms to control the flexible modes of the vehicle via appropriate sensors and actuators. Another control manager is responsible for the control functions of a robot arm for satellite repair. The control executive issues commands to the managers and coordinates their actions.

Note that the autonomous controller is only one of the autonomous systems on the vehicle. It is responsible for all the functions related to the control of the physical system and allows for continuous online development of the autonomous controller and to provide for various phases of mission operations. The tier structure of the architecture allows us to build on existing advanced control theory. Development progresses, creating each time, higher-level adaptation and a new system which can

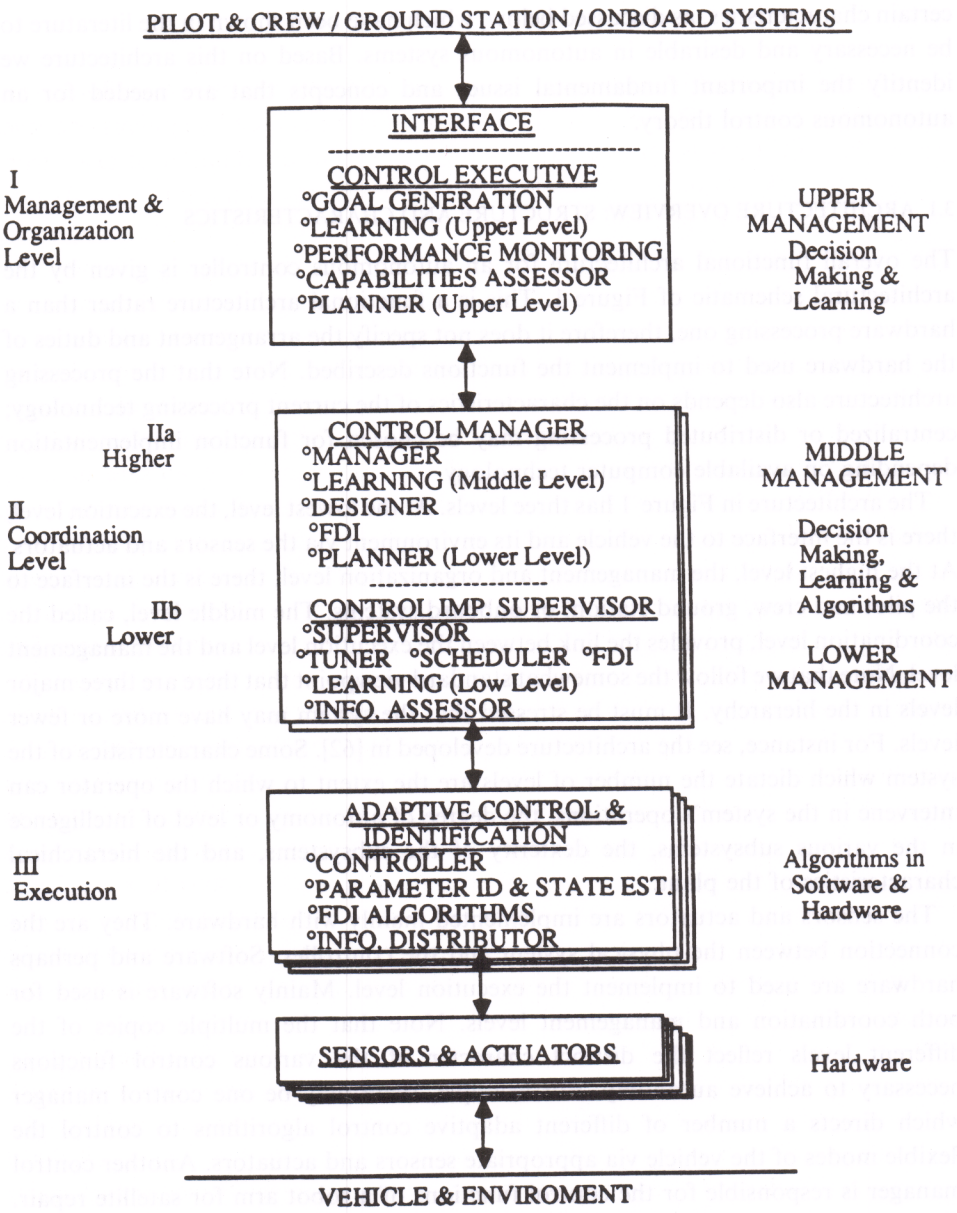

Fig. 1. Autonomous controller architectural schematic.

be operated and tested independently. The autonomous controller performs many of the functions currently performed by the pilot, crew, or ground station. The pilot and crew are thus relieved from mundane tasks and some of the ground station functions are brought aboard the vehicle. In this way the vehicle becomes more autonomous.

3.2. FUNCTIONAL OPERATION

Commands are issued by higher levels to lower levels and response data flows from lower levels upwards. Parameters of subsystems can be altered by systems one level above them in the hierarchy. There is a delegation and distribution of tasks from higher to lower levels and a layered distribution of decision-making authority. At each level, some preprocessing occurs before information is sent to higher levels. If requested, data can be passed from the lowest subsystem to the highest, e.g., for display. All subsystems provide status and health information to higher levels. Human intervention is allowed even at the control implementation supervisor level (IIb).

The specific functions at each level are described in detail in later sections. Here we present a simple illustrative example to clarify the overall operation of the autonomous controller. Suppose that the pilot desires to repair a satellite. After dialogue with the control executive via the interface, the task is refined to 'repair satellite using robot A'. This is arrived at using the capability assessing, performance monitoring, and planning functions of the control executive. The control executive decides if the repair is possible, under the current performance level of the system, and in view of near term planned functions. The control executive, using its planning capabilities, sends a sequence of subtasks sufficient to achieve the repair to the control manager. This sequence could be to order robot A to: 'go to satellite at coordinates xyz', 'open repair hatch', 'repair'. The control manager, using its planner, divides say the first subtask, 'go to satellite at coordinates xyz', into smaller subtasks: 'go from start to $x_1 y_1 z_1$', then 'maneuver around obstacle', 'move to $x_2 y_2 z_2$', . . . , 'arrive at the repair site and wait'. The other subtasks are divided in a similar manner. This information is passed to the control implementation supervisor, which recognizes the task, and uses stored control laws to accomplish the objective. The subtask 'go from start to $x_1 y_1 z_1$', can for example, be implemented using stored control algorithms to first, proceed forward 10 meters, to the right 15 degrees, etc. These control algorithms are executed in the controller at the execution level utilizing sensor information; the control actions are implemented via the actuators.

It is important at this point to discuss the *dexterity* of the controller. The execution level of a highly dexterous controller is very sophisticated and it can accomplish complex control tasks. The implementation supervisor can issue commands to the controller such as 'move 15 centimeters to the right', and 'grip standard, fixed dimension cylinder', in a dexterous controller, or it can completely dictate each mode of each joint (in a manipulator) 'move joint 1 15 degrees', then 'move joint 5 3 degrees', etc. in a less dexterous one. The simplicity, and level of abstractness of macro commands in an autonomous controller depends on its dexterity. The more sophisticated the execution level is, the simpler are the commands that the control implementation supervisor needs to issue. Notice that a very dexterous robot arm may itself have a number of autonomous functions. If two such dexterous arms were used to complete a task which required the coordination of their actions, then the arms would be considered to be two dexterous actuators and a new supervisory autonomous

controller would be placed on top for the supervision and coordination task. In general, this can happen recursively, adding more intelligent autonomous controllers as the lower level tasks, accomplished by autonomous systems, need to be supervised.

3.3. THE EXECUTION LEVEL (III)

The functional architecture for the execution level of the autonomous controller is shown in Figure 2. Its main function is to generate, via the use of numeric algorithms, low level control actions as dictated by the higher levels of the controller, and apply them to the vehicle. It senses the responses of the vehicle and environment, processes it to identify parameters, estimates states, or detects vehicle failures, and passes this information to the higher levels.

The *Sensor* and *Actuator* subsystems are depicted in Figure 2. These devices which physically accomplish the functions for the autonomous controller are at the lowest level of the architecture. The complexity of these devices depends on the dexterity of the controller. All sensors which provide information from the vehicle and environment to any component in the autonomous controller are included here. On the execution level, the controller will need feedback information about control variables. The state estimator and parameter identifier also use such outputs for their respective tasks. The failure detection and identification (FDI) algorithms need these outputs and those of special failure sensors to enable them to detect failures. To perform 'execution monitoring' for the planning systems at the higher levels, the dynamical response of the system must be sensed and passed to the planning system so that it can determine if a plan has failed. The implementation supervisor also needs sensor information so that it can, for instance, make the smooth transition in the

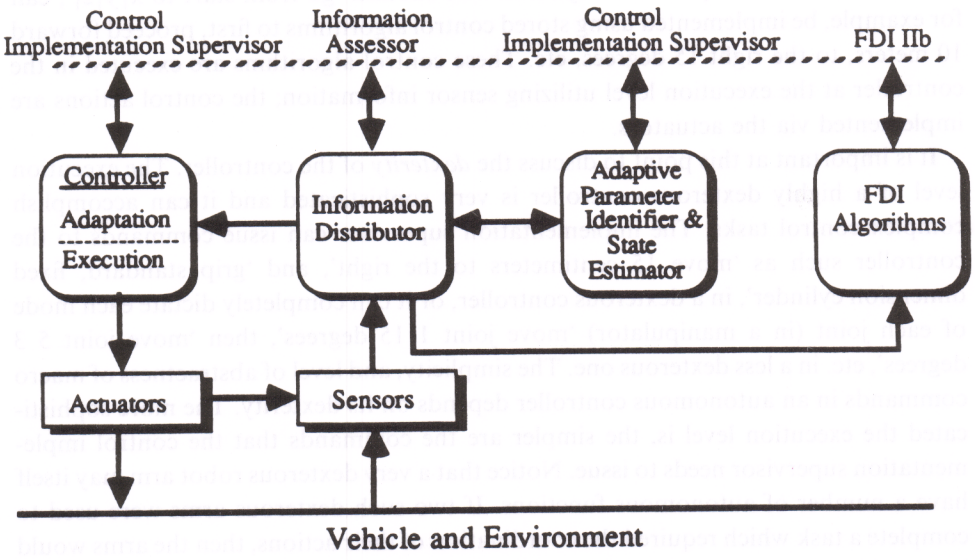

Fig. 2. Execution level.

implementation of a newly designed control law. Sensory information is also used in performance monitoring, capabilities assessing, tuning, scheduling, and display to the pilot, crew, ground station, or other onboard systems. The actuators are the usual control actuators (transducers) which translate the outputs of the controller to actions meaningful to the vehicle. For a highly dexterous controller, a whole manipulator may be considered to be an 'actuator'.

The main function of the *Controller* in Figure 2 is to execute the control algorithms and to issue commands to the actuators. It performs advanced conventional adaptive control functions. It receives, in real time, all the necessary data (from the information distributor) to execute the current control algorithm. The information consists of current output values from the sensors, model parameter estimates and state estimates, as they are generated from the identifier. The *adaptation* part of the controller algorithmically interprets the values of the measured plant variables and the estimated plant parameters and states; and it adjusts, on-line, the coefficients of the control law which runs in the *execution* part of the controller. These functions correspond to conventional adaptive control. The adaptation algorithm can contain information about the model to be followed, thus implementing 'model following' adaptive control. Since the model parameters are explicitly estimated and then used in the control law adaptation, the structure appears to suggest an 'indirect' adaptive control approach. However, notice that this is not necessarily the case since the model parameter estimates from the identifier can simply be ignored and the adaptation algorithm can directly process the information from sensors to directly estimate the control law coefficients, thus implementing 'direct' adaptive control. If a fixed control law is used, then the appropriate sensor data are simply fed back to the control law which is being executed. The sensor data are values of measured variables; these include states as well.

All possible conventional control functions can be performed via the proposed achitecture. For fixed control laws, one could envision a loop containing the sensors providing feedback information, through the information distributor, to the controller; the control actions are performed via the actuators. For adaptive control this also involves the model parameter identifier. In addition to advanced adapative control functions, the controller has the following capabilities: The controller allows intervention from above. It of course allows the introduction of reference signals for set points and tracking as conventional controllers do. In addition it receives commands: (i) To alter the parameters of adaptation (as determined by the tuner in the coordination level), and (ii) To switch to different control laws altogether suggested by the scheduler or the control redesigner.

If the higher levels of the architecture are ignored, the intervention to the controller can be envisioned as being that of a human operator who adjusts certain parameters depending on performance, sets the set points, switches to different algorithms from time to time or to new control laws when failures occur. It returns status information to the higher level, such as what particular control law is currently running; also information about the health of the system (errors in implementation, etc.).

The controller has access to a variety of stored control laws. The particular location of the stored programs is not important in this functional architecture. They could be located in the controller, or in the level above (implementation supervisor). If they are located above, then one should allow for down loading these programs. Since control law switching is desirable, transition programs, for smooth control law switching are necessary. When the scheduler and control redesigner send new control laws to be implemented they should also attach a program to ensure the smooth transition from the current to the new control law. The controller consists basically of software plus hardware as is necessary.

The main function of the *information distributor* shown in Figure 2, is to distribute sensor, parameter, and state information where it is needed. Since the control models and therefore the control, identification, estimation, and FDI algorithms do change, it is essential to guarantee that the execution level subsystems receive each time the correct information. Information about the current control models and current algorithms is provided from above. Using stored information, the distributor provides the correct sensor information to the controller for control feedback purposes, to the identifier for model parameter identification and state estimation, and to the FDI for detection and isolation. After perhaps some preprocessing, it also provides this information to higher levels.

The main function of the *adaptive parameter identifier and state estimator* shown in Figure 2 is to execute parameter identification algorithms and state estimation algorithms and to continuously pass this information to the controller, to the FDI algorithms, and to higher levels. It receives all appropriate sensor information from the information distributor. The parameters and the states, the estimates of which are sought, depend on the particular control model used. Since the control model and the control law do change, the parameter identifier and state estimator should be able to switch control models and identification and estimation algorithms. This information is given from above. It provides the necessary parameter and state estimates to the controller and to the FDI algorithms via the information distributor. It returns to the higher level, parameter estimates and state estimates of the current model (via the information distributor) and information as to the status and health of the system directly.

The main function of the *FDI algorithms* shown in Figure 2, is to execute FDI algorithms for failures detected at the execution level of the autonomous controller. It receives all appropriate sensor information via the information distributor. This includes information from sensors specifically located to detect failures at the actuator level of the control system; it also includes model parameter and state estimates from the identifier. It has the ability to switch algorithms and plant models. The FDI algorithms return information to the higher level FDI subsystems.

3.4. COORDINATION LEVEL (IIb)

The functional architecture for coordination level IIb is shown in Figure 3. Coordination level IIb receives commands to perform predetermined specific control tasks from

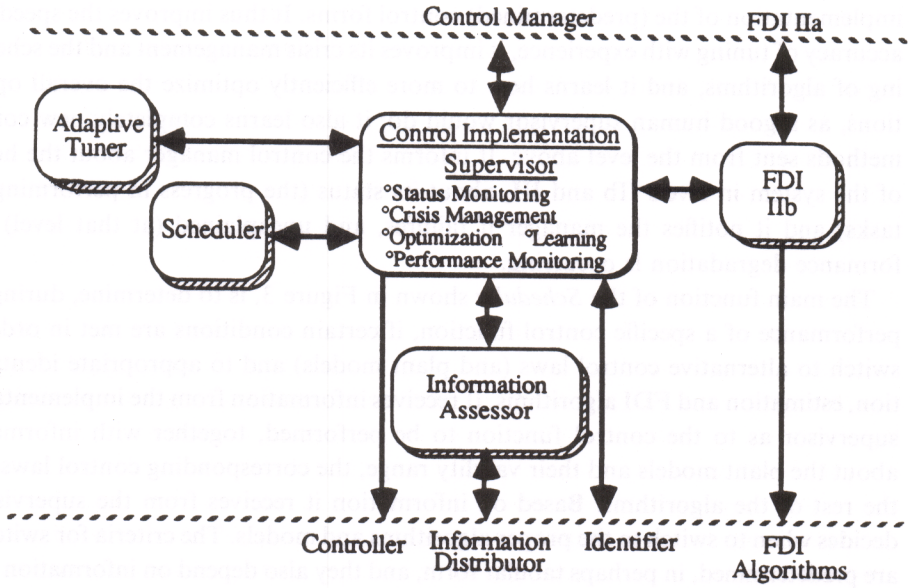

Fig. 3. Coordination level IIb.

the control manager in the level above. It provides the appropriate sequence of control and identification algorithms to the execution level below. Its ability to deal with extensive uncertainties is limited.

The main function of the *control implementation supervisor* shown in Figure 3, is to carry out the sequence of control actions dictated by the control manager. It can accomplish predetermined control actions and cope with limited predetermined crisis situations. The supervisor receives the sequence of control tasks to be accomplished from the control manager and it has access to a variety of control models, and control, identification and estimation algorithms. It selects appropriate reference signals for the controller and it *optimizes* the subsequences of actions to accomplish the tasks dictated by the above levels in the best way possible. The supervisor uses the scheduler to decide what models and algorithms to use in the controller and identifier; it uses the tuner to decide how to adapt parameters in the algorithms, which are currently used, and it sends this information to the execution level. It monitors the *status* of the system at IIb and III, i.e., what algorithms and models are currently used, and the *health* of the systems. The supervisor does *performance monitoring* on IIb and III levels using information provided by the information assessor and FDI IIb. It contains a *crisis management* facility to deal with certain failures. This includes a number of methods to maintain performance or to maintain a certain degree of safety in operations, while degrading performance gracefully. For example, if a failure in an actuator or sensor is detected, it can switch to an alternative control method using other actuators or sensors to maintain performance. If performance cannot be maintained, it should degrade gracefully, guaranteeing safety (stability). It will take the necessary steps to maintain stability after a failure is detected and it is isolated and identified. The control implementation supervisor uses *learning* to improve the

implementation of the (predetermined) control forms. It thus improves the speed and accuracy of tuning with experience, it improves its crisis management and the scheduling of algorithms, and it learns how to more efficiently optimize the overall operations, as a good human supervisor would do; it also learns completely new control methods sent from the level above. It informs the control manager about the health of the system in levels IIb and III, about its status (the progress in performing the tasks) and it notifies the manager if failures, and unexplained (at that level) performance degradation is occurring.

The main function of the *Scheduler* shown in Figure 3, is to determine, during the performance of a specific control function, if certain conditions are met in order to switch to alternative control laws (and plant models) and to appropriate identification, estimation and FDI algorithms. It receives information from the implementation supervisor as to the control function to be performed, together with information about the plant models and their validity range, the corresponding control laws, and the rest of the algorithms. Based on information it receives from the supervisor it decides when to switch to the proper algorithms and models. The criteria for switching are predetermined, in perhaps tabular form, and they also depend on information from environmental sensors. This information is transmitted from the higher level through the supervisor. Examples will be the schedulers used for docking control. Depending, for example, on approaching speed and attitude, an appropriate new control law is selected. Here, the scheduler also selects corresponding plant model when necessary. The scheduler does not deal with crisis situations.

The main function of the *adaptive tuner* shown in Figure 3 is to determine, during the execution of particular algorithms, if specific conditions are met in order to adjust, tune, certain parameters in the adaptation laws. It receives information from the implementation supervisor as to the current algorithm being executed, control and identification algorithms, and also information from the information assessor (via the supervisor) necessary to decide first if timing is appropriate. Then based on predetermined criteria, it selects the new values for the parameters in the adaptation laws. The criteria for tuning will be based on excessive output, state, and parameter errors, and the selection of the new adaptive parameter values will depend on algorithms or heuristic rules using performance measures and actual past and present inputs and outputs. In this way parameter tuning of identification and control algorithms (adaptive, robust, optimal) is accomplished.

The main function of the *information assessor* shown in Figure 3 is to process and distribute sensor, state and parameter information to the information distributor (execution level) and the implementation supervisor. It receives information from the supervisor as to the current plant model, control, estimation and identification, and FDI algorithms, and it instructs the information distributor to pass the necessary sensor information to the controller, identifier, and FDI systems. It receives, from the identifier via the distributor, information about the current model parameter and state estimates. After instruction from the supervisor it supplies to the supervisor processed information such as errors in state and parameter estimation. To do this,

it uses sensor data supplied by the distributor and models supplied by the supervisor. This processed information is used by the tuner, the scheduler, and the control implementation supervisor for performance monitoring.

The main function of the *FDI IIb* subsystem shown in Figure 3 is to supervise the FDI algorithms (execution level) and to detect and identify, using algorithms and heuristic methods, failures that occurred at the execution level. It passes the information about the current models used from the supervisor to the FDI algorithms. It sends appropriate FDI algorithms to be executed to the lower level. It receives the outputs of those algorithms. It compares them with additional information from the supervisor, and it proceeds, after detecting a failure, to isolate and identify it. It informs the FDI IIa subsystem about the status of the failure and it also informs the supervisor so that predetermined crisis measures can be taken if necessary and possible. If the crisis cannot be dealt with at that level, the information is passed to the FDI IIa, and the designer via the control manager.

3.5. COORDINATION LEVEL IIa

The functional architecture for coordination level IIa is shown in Figure 4. Coordination level IIa receives commands from the management level which it must determine how to perform using the designer and planner and considering information from FDI IIa and the control implementation supervisor. It generates a sequence of control actions that the control implementation supervisor can recognize and passes them to it. This coordination level has abilities to deal with significant uncertainties.

The main function of the *control manager* shown in Figure 4 is to accomplish the control tasks given by the control executive. It can accomplish predetermined control actions, using the lower levels, but also it can cope with failures to a large degree. In general it is equipped to successfully carry out the control tasks under a wide variety of unanticipated vehicle and environmental conditions. It can also be directed to prepare for future requirements, building new control laws and contingency methods

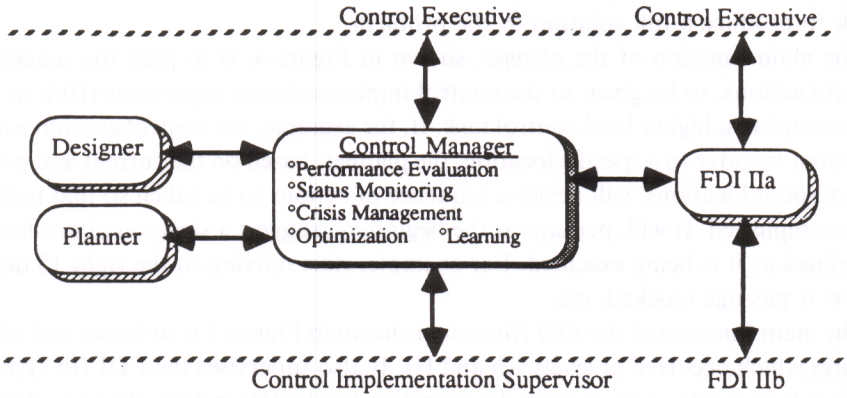

Fig. 4. Coordination level IIa.

using the designer. When a control task is given, it breaks it down into a sequence of control actions, using the planner, and it passes it to the implementation supervisor. It receives processed sensor information from the supervisor about the current positions and information from the above about the goals so that it can plan its actions. It also passes to the supervisor newly designed algorithms and contingency plans. It receives, from the implementation supervisor, *status and health* information and it passes its status and health information to the executive. It does *performance evaluation and monitoring* on II and III levels. For example, it evaluates the performance of a sequence of tasks so that, changes in the next sequence are implemented if necessary. It also contains a *crisis management facility* to deal with failures. It is similar to the one in the implementation supervisor but it deals with higher level contingencies. It has significant *learning abilities* to improve its performance. It does *optimization* of the system below it using the planned actions, and it suggests new strategies in algorithm selection to the implementation supervisor.

The main function of the *designer* shown in Figure 4 is to develop methodologies to deal with novel situations for which no prior designs have been made. These include detected failures via the FDI system, in which case new control laws must be designed on-line, using the models and specifications provided by the control manager. They also include dealing with new control tasks suggested by the manager or the higher level. When there are no requirements to develop new methods in real time, the designer, under direction of the manager, works on developing new methods to build up the crisis management algorithm inventory, and the inventory of algorithms to deal with new control tasks needed some time in the future. These algorithms are passed to the control implementation supervisor at the direction of the control manager. In this way, the system is enriched and improved, it becomes more experienced, as it can deal with a greater number of contingencies and tasks. The designer uses decision-making under uncertainty (symbolic based methods) to select design algorithms. When the designer must react in a very short time to deal with, say a major failure, it may decide to initially suggest a method which will preserve the system integrity without meeting all the performance specifications. In the mean time it can work to produce a full solution to the problem.

The main function of the *planner*, shown in Figure 4, is to plan the sequence of control actions, to be given to the control implementation supervisor (IIb), in order to accomplish a higher level control task. If, for example, the control executive orders the robot to move to a specific location, the planner, based on the current and possible future robot locations, will devise a sequence of actions to be taken so that tasks will be accomplished. It will, perhaps at the beginning, suggest a skeleton plan which will be refined as it is being executed. For example, start moving to the right 15 degrees, report if passage blocked, etc.

The main function of the *FDI IIa* system shown in Figure 4 is to detect and identify failures which occurred at levels IIb and III. It also supervises the FDI IIb system. It receives failure information from the execution level (III) via FDI IIb and additional information from the control manager. It informs the manager about the failure

location and its severity, so that measures can be taken, using perhaps the services of the designer. It directly informs the control executive about the status of the failures detected at any level since they are very important in capability assessment. It uses high level decision-making involving heuristics and few algorithms.

3.6. MANAGEMENT AND ORGANIZATION LEVEL (I)

The functional architecture for the management and organization level (I) is shown in Figure 5. It interfaces to the pilot, crew, ground station, and other onboard systems and performs the highest level control functions. It oversees and directs all the activities at both the coordination and execution levels. It is the most 'intelligent' of the three levels.

The main function of the *control executive* shown in Figure 5 is to accomplish high level control tasks given by the pilot, crew, ground station, or other onboard systems. Such a task could be: Change orbit to . . . , deploy satellite (open door, turn, etc, then deploy), repair satellite via robot A (send robot to satellite, open hatch, repair), retrieve satellite, etc. It performs high level *planning*. It *optimally* breaks down the 'macro commands' into simpler commands for the control manager (IIa). It performs *capability assessing* of the control system. It receives information about faults from FDI and about *status* and *health* from the control manager and performs high level *performance monitoring*. It evaluates the current situation and it predicts what can be reasonably expected to be accomplished in a certain time. For example, 'docking procedures under way, estimated docking in 30 seconds'. It provides this information to the *goal generation* facility, which by exchanging information, having a dialogue, with other onboard systems, pilot, or ground station via the interface, generates attainable realistic goals to be accomplished by the autonomous controller. For example, in view of the current situation, 'docking can be achieved in 30 seconds but not in 20 seconds as requested'. These goals are then used by the planner in the control executive, to plan the necessary steps which lead to their accomplishment. The control

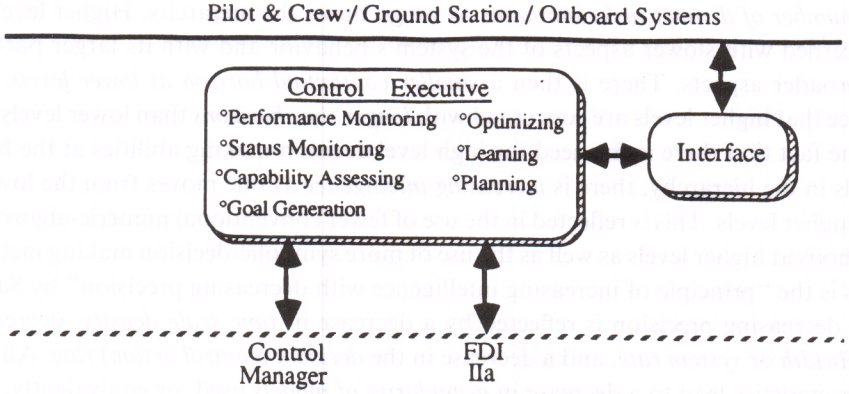

Fig. 5. Management and organization level.

executive has significant *learning abilities*. It uses past experience to increase its efficiency and to improve its capability assessment. It is informed by the control manager about new capabilities possible, by newly generated control methods. It suggests preparation for future control tasks. It uses decision making exclusively. It interprets reports from below and execution commands from above. It can request, through interface, additional information from the pilot, crew, ground station, or other onboard systems which may be useful in the control system. This includes navigation information, future uses of the autonomous controller, etc.

Learning is essential to the development of a true autonomous system. High-level learning will occur at the management and organization level. At each level of learning, beginning at coordination level IIb, information is for instance, successively generalized via induction. The controller may need to learn the model of the plant, the problem solving strategy, the goals to obtain, and the required performance level.

The main function of the *interface*, shown in Figure 5, is to provide the liaison, interface, between the autonomous control system and the pilot and crew/ground station/other onboard systems. It is an *intelligent interface* as it allows user friendly dialogue. It is a *language translator*, translating language of other systems or the crew or ground station into a language familiar to the autonomous controller. It *displays data* from the control subsystems if requested. It passes the *control status* to the crew etc., and desired behavior, and goals, to the control executive.

4. Fundamental Characteristics and Issues

Based on this architecture we identify the important fundamental concepts and characteristics that are needed for an autonomous control theory. Note that several of these have been discussed in the literature as outlined in Section 2.3.2. Here, these characteristics are brought together for completeness. Furthermore, the fundamental issues which must be addressed for a quantitative theory of intelligent autonomous control are introduced and discussed.

There is a *successive delegation of duties* from the higher to lower levels; consequently the *number of distinct tasks* increases as we go down the hierarchy. Higher levels are concerned with slower aspects of the system's behavior and with its larger portions, or broader aspects. There is then a *smaller contextual horizon at lower levels*. Also notice that higher levels are concerned with *longer time horizons* than lower levels. Due to the fact that there is the need for high level decision making abilities at the higher levels in the hierarchy, there is *increasing intelligence* as one moves from the lower to the higher levels. This is reflected in the use of fewer conventional numeric-algorithmic methods at higher levels as well as the use of more symbolic-decision making methods. This is the "principle of increasing intelligence with decreasing precision" by Saridis. The decreasing precision is reflected by a decrease in *time scale density*, decrease in *bandwidth* or *system rate*, and a decrease in the *decision* (*control action*) *rate*. All these characteristics lead to a decrease in *granularity of models* used, or equivalently, to an *increase in model abstractness*. Model granularity also depends on the *dexterity* of the

autonomous controller as discussed in Sections 3.2 and 3.3. Next we discuss an approach which, in our opinion, is especially suitable for an analytical study of intelligent autonomous control systems.

The quantitative, systematic techniques for modelling, analysis, and design of control systems are of central and utmost practical importance in conventional control theory. Similar techniques for intelligent autonomous controllers do not exist. This is of course because of their novelty, but for the most part, it is due to the '*hybrid*' structure (nonuniform, nonhomogeneous nature) of the dynamical systems under consideration. The systems are hybrid since in order to examine autonomy issues, a more global, macroscopic view of a dynamical system must be taken than in conventional control theory. Modelling techniques for intelligent autonomous systems must be able to support this macroscopic view of the dynamical system, hence it is necessary to represent both numeric and symbolic information (see discussion in Section 2). We need modelling methods that can gather all information necessary for analysis and design. For example, we need to model the dynamical system to be controlled (e.g., a space platform), failures that might occur in the system, the conventional adaptive controller, and the high level decision making processes at the management and organization level of the intelligent autonomous controller (e.g., an AI planning system performing actions that were once the responsibility of the ground station). The nonuniform components of the intelligent controller all take part in the generation of the low level control inputs to the dynamical system, therefore they all must be considered in a complete analysis. For an extended discussion on the modelling of hybrid systems consult [72].

It is our viewpoint that conventional modelling, analysis, and design methods should be used whenevever they are applicable. For instance, they should be used at the execution level of many autonomous controllers. We propose to augment and enhance existing theories rather than develop a completely new theory for the hybrid systems described above; we wish to build upon existing, well understood and proven conventional methods. The symbolic/numeric interface is a very important issue; consequently it should be included in any analysis. There is a need for systematically generating less detailed, more abstract models from differential/difference equation models to be used in higher levels of the autonomous controller (coordination level). There is also a need for systematically extracting the necessary information from lower level symbolic models to generate higher level symbolic models to be used in the hierarchy where appropriate. Tools for the analysis of this information extraction also need to be developed. Research in this area is underway. In this way conventional analysis can be used in conjunction with the developed analysis methods to obtain an overall quantitative, systematic analysis paradigm for intelligent autonomous control systems. In short, we propose to use hybrid modelling, analysis, and design techniques for nonuniform systems. This approach is not unlike the approaches used in the study of any complex phenomena by the scientific and engineering communities.

A practical but very important issue is the simulation of hybrid systems. This requires simulation of both conventional differential equations and symbolic decision

making processes. Normally, numeric-algorithmic processing is done with languages like FORTRAN and symbolic decision making can be implemented with LISP or PROLOG. Sometimes the two types of processing are done on computers with quite different architectures. There is then the problem of combining symbolic and numeric processing on one computer. If the computing is done on separate computers, the communication link normally presents a serious bottleneck. Combining AI and conventional numeric processing is currently being addressed by many researchers and some promising results have been reported in [31] and [14].

It was pointed out in Section 2 that complex control problems required a controller sophistication that involved the use of AI methodologies. It is interesting to observe the following [35]: Although there are characteristics which separate intelligent from nonintelligent systems, as intelligent systems evolve, the distinction becomes less clear. Systems which were originally considered intelligent evolve to gain more character of what are considered to be non-intelligent, numeric-algorithmic systems. An example is a route planner. Although there are AI route planning systems, as problems like route planning become better understood, more conventional numeric-algorithmic solutions are developed. The AI methods which are used in intelligent systems, help us to understand complex problems so we can organize and synthesize new approaches to problem solving, in addition to being problem solving techniques themselves. AI techniques can be viewed as research vehicles for solving very complex problems. As the problem solution develops, purely algorithmic approaches, which have desirable implementation characteristics, substitute AI techniques and play a greater role in the solution of the problem. It is for this reason that we concentrate on achieving autonomy and not on whether the underlying system can be considered 'intelligent'.

5. Concluding Remarks

A hierarchical functional autonomous controller architecture was introduced. In particular, the architecture for the control of future space vehicles was described in detail; it was designed to ensure the autonomous operation of the control system and it allowed interaction with the pilot and crew/ground station, and the systems on board the autonomous vehicle. The fundamental issues in autonomous control system modelling and analysis were discussed. It was proposed to utilize a hybrid approach to modelling and analysis of autonomous systems. This will incorporate conventional control methods based on differential equations and new techniques for the analysis of systems described with a symbolic formalism. In this way, the well developed theory of conventional control can be fully utilized. It should be stressed that autonomy is the design requirement and intelligent control methods appear, at present, to offer some of the necessary tools to achieve autonomy. A conventional approach may evolve and replace some or all of the 'intelligent' functions. Note that this paper is based on the results presented in [3].

It was shown that in addition to conventional controllers, the autonomous control system incorporates planning, learning, and FDI. An initial study of the FDI problem

incorporating both conventional and AI FDI techniques was reported in [45]. Furthermore, AI planning systems were modelled and analyzed in a Petri net framework in [46].

It must be stressed that the results presented here apply to any autonomous control system. For other applications, the architecture, or parts of it, and the ideas discussed here are valid. For instance, to achieve a certain level of autonomy for a particular application one may modify the functional architecture by removing the management and organization level. In this case, the limited version of the autonomous controller would not provide for a user interface, goal generation, high level learning, etc. In general, modifying the controller for certain applications entails the removal of portions of the functional architecture which limits the attainable degree of autonomy. Hence, to use the above results for a different application one must decide what level of autonomy is needed and then include in the autonomous controller architecture those components necessary to achieve it.

Acknowledgement

This work was partially supported by the Jet Propulsion Laboratory, Pasadena, California.

References

1. Albus, J., *et al.*, Theory and practice of intelligent control, *Proc. 23rd IEEE COMPCON*, pp. 19–39 (1981).
2. Albus, J.S., *et al.*, Hierarchical control of intelligent machines applied to space station telerobotics, *Proc. Space Telerobotics Workshop*, pp. 155–166 (1988).
3. Antsaklis, P.J. and Passino, K.M., Autonomous control systems: Architecture and concepts for future space vehicles, Final report, Jet Propulsion Laboratory Contract, Oct. 1987.
4. Antsaklis, P.J., Passino, K.M., and Wang, S.J., Autonomous control systems: Architecture and fundamental issues, *Proc. Amer. Control Conference, Atlanta*, pp. 602–607 (1988).
5. Astrom, K.J., *et al.*, Expert control, *Automatica* **22**, 277–286 (1986).
6. Atkinson, D.J., Telerobot task planning and reasoning: Introduction to JPL AI research, *Proc. Space Telerobotics Workshop*, pp. 339–350 (1988).
7. Balaram, J., *et al.*, Run time control architecture for the JPL telerobot, *Proc. Space Telerobotics Workshop*, pp. 211–222, (1987).
8. Bhatt, R., *et al.*, A real time pilot for an autonomous robot, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 135–139 (1987).
9. Blank, G., Responsive system control using register vector grammar, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 461–466 (1987).
10. Charniak, E. and McDermott, D., *Introduction to Artificial Intelligence*, Addison Wesley, Reading, Mass. (1985).
11. Crosscope, J. and Bonnell, R., An integrated intelligent controller employing both conceptual and procedural knowledge, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 416–422 (1987).
12. Cruz, J.B. and Stubberud, A.R., Knowledge based approach to multiple control coordination in complex systems, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 50–53 (1987).
13. DeJong, K., Intelligent control: Integrating AI and control theory, *Proc. IEEE Trends and Applications 1983*, pp. 158–161 (1983).
14. Despain, A.M. and Patt, Y.N., Aquarius – a high performance computing system for sysmbolic/numeric applications, *Proc. COMPCON S'85*, pp. 376–382 (1985).

15. Dudziak, M.J., *et al.*, IVC: An intelligent vehicle controller with real-time strategic replanning, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 145–152 (1987).
16. Dudziak, M.J., SOLON: An autonomous vehicle mission planner, *Proc. Space Telerobotics Workshop*, pp. 289–302 (1987).
17. Farsaie, A., *et al.*, Intelligent controllers for an autonomous system, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 154–158 (1987).
18. Findeisen, W., *et al., Control and Coordination in Hierarchical Systems*, Wiley, New York (1980).
19. Fiorio, G., Integration of multi-hierarchy control architectures for complex systems, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 71–79 (1987).
20. Firschein, O., *et al., Artificial Intelligence for Space Station Automation*, Noyes, New Jersey (1986).
21. Freidland, P. and Lum, H., Building intelligent systems: Artificial intelligence research at NASA Ames Research Center, *Proc. Space Telerobotics Workshop*, pp. 19–26 (1988).
22. Fu, K.S., Learning control systems and intelligent control systems: An intersection of artificial intelligence and automatic control, *IEEE Trans. Automatic Control*, pp. 70–72 (1971).
23. Gartrell, C.F., *et al.*, The use of expert systems for adaptive control of large space stuctures, *Proc. AIAA Guidance Navigation and Control Conf.*, pp. 376–385 (1985).
24. Gevarter, W.B., *Artificial Intelligence*, Noyes, NJ (1984).
25. Graglia, P. and Meystel, A., Planning minimum time trajectory in the traversability space of a robot, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 82–87 (1987).
26. Guha, A. and Dudziak, M., Knowledge based controllers for autonomous system, *Proc. IEEE Workshop Intelligent Control*, pp. 134–138 (1985).
27. Handelman, D.A. and Stengel, R.F., Combining qualitative and quantitative reasoning in aircraft failure diagnosis, *AIAA Guidance Navigation and Control Conf.*, pp. 366–375 (1985).
28. Hawker, J. and Nagel, R., World models in intelligent control systems, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 482–488 (1987).
29. Hodgson, J., Structures for intelligent systems, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 348–353 (1987).
30. Jenkins, L., Space telerobotic systems: Applications and concepts, *Proc. Space Telerobotics Workshop*, pp. 29–34 (1988).
31. Kitzmiller, C.T. and Kowalik, J.S., Coupling symbolic and numeric computing in KB systems, *AI Magazine* **18**, 85–90 (1987).
32. Knight, J.F. and Passino, K.M., Decidability for Temporal Logic in Control Theory, *Proc. 25th Allerton Conf.*, pp. 335–344 (1987).
33. Krogh, B., Controlled Petri nets and maximally permissive feedback logic, *Proc. 25th Allerton Conf.*, pp. 317–326 (1987).
34. Lutz, P., Autonomous mobile robots in industrial production environment, in L.O. Hertzgerger and F.C.A. Groen (eds.) *Intelligent Autonomous Systems*, North-Holland, NY (1987) (Proc. of an International Conference in Amsterdam, Dec. 1986.)
35. Mendel, J. and Zapalac, J., The application of techniques of artificial intelligence to control system design, in *Advances in Control Systems*, C.T. Leondes (ed.), Academic Press, NY (1968).
36. Mesarovic, M., Macko, D. and Takahara, Y., *Theory of Hierarchical, Multilevel, Systems*, Academic Press, NY (1970).
37. Meystel, A., Intelligent control: Issues and perspectives, *Proc. IEEE Workshop Intelligent Control*, pp. 1–15 (1985).
38. Meystel, A., Nested hierarchical controller for intelligent mobile autonomous system, in L.O. Hertzgerger and F.C.A. Groen (eds.), *Intelligent Autonomous Systems*, North Holland, NY (1987) (Proc. of an International Conference in Amsterdam, Dec. 1986.)
39. Meystel, A., Planning/control architectures for master dependent autonomous systems with non-homogeneous knowledge representation, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 31–41 (1987).
40. Meystel, A., Nested hierarchical controller with partial autonomy, *Proc. Space Telerobotics Workshop*, pp. 251–270 (1988).
41. Moizer, A. and Pagurek, B., An onboard navigation system for autonomous underwater vehicles, in L.O. Hertzgerger and F.C.A. Groen (eds.), *Intelligent Autonomous Systems*, North Holland, NY (1987) (Proc. of an International Conference in Amsterdam, Dec. 1986.)

42. Ostroff, J.S., Real time computer control of discrete systems modelled by extended state machines: A temporal logic approach, PhD dissertation, Report No. 8618, Dept. of Elect. Eng., University of Toronto, Jan. 1987.
43. Passino, K.M., Restructurable controls and artificial intelligence, McDonnell Aircraft Internal Report, IR-0392, April 1986.
44. Passino, K.M. and Antsaklis, P.J., Restructurable controls study: An artificial intelligence approach to the fault detection and identification problem, Final Report, McDonnell Douglas Contract, Oct. 1986.
45. Passino, K.M. and Antsaklis, P.J., Fault detection and identification in an intelligent restructurable controller, *Journal of Intelligent and Robotic Systems* **1**, 145–161 (1988).
46. Passino, K.M. and Antsaklis, P.J., Artificial intelligence planning problems in a Petri net framework, *Proc. Amer. Control Conference*, pp. 626–631 (1988).
47. Pearson, G., Mission planning for autonomous systems, *Proc. Space Telerobotics Workshop*, pp. 303–306 (1988).
48. Raulefs, P. and Thorndyke, P.W., An architecture for heuristic control of real time processes, *Proc. Space Telerobotics Workshop*, pp. 139–148 (1988).
49. Saridis, G.N., Toward the realization of intelligent controls, *Proc. IEEE* **67**, 1115–1133 (1979).
50. Saridis, G., Intelligent controls for advanced automated processes, *Proc. Automated Decision Making and Problem Solving Conf., NASA CP-2180*, May 1980.
51. Saridis, G.N., Intelligent robot control, *IEEE Trans. Automatic Control,* **AC-28**, 547–556 (1983).
52. Saridis, G.N., Foundations of the theory of intelligent controls, *Proc. IEEE Workshop Intelligent Control*, pp. 23–28 (1985).
53. Saridis, G.N., Knowledge implementation: structures of intelligent control systems, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 9–17 (1987).
54. Saridis, G.N. and Valavanis, K.P., Software and hardware for intelligent robots, *Proc. Space Telerobotics Workshop*, pp. 241–250 (1988).
55. Schenker, P.S., Program objectives and technology outreach, *Proc. Space Telerobotics Workshop*, pp. 3–18 (1988).
56. Shapiro, S.C. (ed.), *Encyclopedia of Artificial Intelligence*, Wiley, NY (1987).
57. Stark, L., Telerobotics: Research needs for evolving space stations, *Proc. Space Telerobotics Workshop*, pp. 91–94 (1988).
58. Stengel, R.F., AI theory and reconfigurable flight control systems, Princeton University Report 1664-MAE, June 1984 (see also 1665 by D. Handelman).
59. Stephanou, H.E., An evidential framework for intelligent control, *Proc. IEEE Workshop Intelligent Control*, pp. 118–123 (1985).
60. Thistle, J.G. and Wonham, W.M., Control problems in a temporal logic framework, *Int. J. Control* **44**, 943–976 (1986).
61. Trankle, T.L., Sheu, P. and Rabin, U.H., Expert system architecture for control system design, *Proc. Amer. Control Conference*, pp. 1163–1169 (1986).
62. Turner, P.R., *et al.*, Autonomous systems: Architecture and implementation, Jet Propulsion Laboratories, Report No. JPL D-1656, August 1984.
63. Valavanis, K.P., A mathematical formulation for the analytical design of intelligent machines. PhD Dissertation, Electrical and Computer Engineering Dept., Rensselaer Polytechnic Institute, Troy NY, Nov. 1986.
64. Valvanis, K.P. and Saridis, G.N., Architectural models for intelligent machines, *Proc. 25th Conf. Decision and Control, Athens, Greece* (1986).
65. Valavanis, K.P. and Saridis, G.N., Information theoretic modelling of intelligent robotic systems, Part I: The organization level, *Proc. 26th Conf. Decision and Control, Los Angeles*, pp. 619–626 (1987).
66. Valavanis, K.P. and Saridis, G.N., Information theoretic modelling of intelligent robotic systems, Part II: The coordination and execution levels, *Proc. 26th Conf. Decision and Control*, Los Angeles, pp. 627–633 (1987).
67. Villa, A., Hybrid knowledge based/analytical control of uncertain systems, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 59–70 (1987).
68. Waldon, S., *et al.*, Updating and organizing world knowledge for an autonomous control system, *Proc. IEEE Internat. Symp. Intelligent Control*, pp. 423–430 (1987).

69. Wolfe, W.J. and Raney, S.D., Distributed intelligence for supervisory control, *Proc. Space Telerobotics Workshop*, pp. 139–148 (1988).
70. Wos, L., *Automated Reasoning: 33 Basic Research Problems*, Prentice Hall, NJ (1988).
71. Zadeh, L.A., Fuzzy logic, *Computer*, April 1988, 83–93.
72. Zeigler, B.P., Knowledge representation from Newton to Minsky and beyond, *Journal of Applied Artificial Intelligence* **1**, 87–107 (1987).