ARCHITECTURES

FOR

INTELLIGENT MACHINES

by

GEORGE N. SARIDIS

DIRECTOR, CIRSSE
RPI, TROY, NY, USA

ABSTRACT

The Theory of Intelligent Machines has been recently reformulated to incorporate new architectures that are using Neural and Petri nets. The analytic functions of an Intelligent Machine are implemented by Intelligent Controls, using Entropy as a measure. The resulting hierarchical control structure is based on the Principle of Increasing Precision with Decreasing Intelligence. Each of the three levels of the Intelligent Control is using different architectures, in order to satisfy the requirements of the Principle: the Organization level is modeled after a Boltzmann machine for abstract reasoning, task planning and decision making; the Coordination level is composed of a number of Petri Net Transducers supervised, for command exchange, by a dispatcher, which also serves as an interface to the Organization level; the Execution level, includes the sensory, planning for navigation and control hardware which interacts one-to-one with the appropriate Coordinators, while a VME bus provides a channel for database exchangeamong the several devices. This system is currently implemented on a robotic transporter, designed for space construction at the CIRSSE laboratories at the Rensselaer Polytechnic Institute. The progress of its development will be reported.

1. INTRODUCTION

In the last few years *Intelligent Machines*, proposed by the Saridis (1979), have reached a point of maturity to be implemented on a robotic testbed aimed for space assembly and satellite maintenance. They feature an application of the *theory of Hierarchically Intelligent Control*, which is based on the *principle of Increasing Precision with Decreasing Intelligence (IPDI)* to form an analytic methodology, using Entropy as a measure of performance. The original architecture represented a three level system, structured according to the principle, and using an information theoretic approach (Saridis and Valavanis 1988). The three levels, (Fig.1):

Organization level
Coordination level and
Execution level

representing the original architecture of the system, have not been changed, but their internal architectures have been recently modified to incorporate more efficient and effective structures dictated by experience (Fig.2).

This paper discusses these new architectures for each one of the levels separately, and justifies their effectiveness by presenting some implementation results from the robotic transporter in CIRSSE at RPI.

2. THE ORGANIZATION LEVEL

2.1 The Architecture

A *Boltzmann machine* type neural net, originally proposed for text generation, has been used for the structure that implements

the Organization level of an Intelligent Machine(Saridis and Moed 1988, Moed and Saridis 1990). This machine would connect a *finite* number of letters (nodes) into grammatically correct words (rules), by minimizing at the first layer the total entropy of connections. Replacing the letters at the nodes with words, at second layer, sentenses are created. At the third level the words are replaced by sentenses at the nodes and so on and so forth until a meaningful text is created.

The functions of the Organizer, following the model of a knowledge based system, comprise of *representation, abstract task planning* (with minimal knowledge of the current environment), *decision making, and learning* from experience. All those functions can be generated by a Boltzmann machine similar to the text gene-rating machine, by considering a finite number of primitive elements at the nodes, constituting the basic actions and actors at the *representation phase*. Strings of these primitives are generated by the Boltzmann machine at the *planning phase* with the total entropy representing the cost of connections. The selection of the string with minimum entropy is the *decision making* process, and the upgrading of the parameters of the system by rewarding the successful outcomes through feedback, is the *learning* procedure. The next to minimumentropy string may be retained as an alternate plan in case of failure of the original or errors created by the environment.

This bottom-up approach, characteristic of natural languages, is extremely simple and effective, utilizing intelligence to replace the complexity of the the top-down type task decompositions. The tasks thus generated, are practically independent of the current environment. Information about the present world should be gathered at the Coordination level. An appropriate world model is constructed from sensory and motion information available at that level. However, there the structure of the Dispatcher, designed to interpret the Organizer's strings, monitor and traffic commands

among the other Coordinators is highly dependent on the strings which represent the planned tasks.

2.2 The Analytic Model

To specify analytically the model of the organizer, it is essential to derive the domain of the operation of the machine for a particular class of problems (Saridis and Valavanis 1988). Assuming that the environment is known, one may define the following functions on the organization level:

a. Machine Representation and Abstract Reasoning, (RR) is the association of the compiled command to a number of activities and/or rules. A probability function is assigned to each activity and/or rule and the Entropy associated with it is calculated. When rules are included one has active reasoning (inference engine).

   In order to generate the required analytic model of this function the following sets are defined:

   The set of *commands* $C = \{c_1, c_2, \ldots, c_q\}$ in natural language, is received by the machine as inputs. Each command is compiled to yield an equivalent machine code explained in the next section.

   The *task domain* of the machine contains a number n of independent objects.

   The set $E = \{e_1, e_2, \ldots, e_m\}$ are *individual primitive events* stored in the long-term memory and repesenting primitive tasks to be executed. The task domain indicates the capabilities of the machine.

   The set $A = \{a_1, a_2, \ldots, a_l\}$ are *individual abstract actions* associating the above events to create sentenses by

concatenation. They are also stored in the long-term memory.

The set $S = \{s_1, s_2, \ldots, s_n\} = E \cup A$, $n=m+1$, is the group of *total objects* which combined, define actions represent complex tasks. They represent the nodes of a Neural net.

A set of *random variables* $X = \{x_1, \ldots, x_n\}$ representing the state of events is associated with each individual object s_i. If the random variable x_i is binary (either 0 or 1), it indicates whether an object s_i is *inactive* or *active*, in a particular activity and for a particular command. If the random variables x_i are continuous (or discrete but not binary) over [0,1], they reflect a membership function in a fuzzy decision making problem. In this work, the x_i's are considered to be binary.

A set of *probabilities* P associated with the random variables X is defined as follows:

$$P = \{P_i = \text{Prob}[x_i = 1];\ i=1,\ldots n\} \qquad (1)$$

The probabilities P are known at the beginning of the representation stage. In order to reduce the problem of dimensionality a subset of objects is defined for a given command c_k:

$$S_k = \{s_i;\ P_i \geq a:\ i=1\ldots n\} \subset S \qquad (2)$$

b. Machine Planning, (P), is ordering of the activities.

The ordering is obtained by properly concatenating the appropriate abstract primitive objects $s_i \in S_k$ for the particular command c_k, in order to form the right abstract activities (sentences or text).

The ordering is generated by a Boltzmann machine which measures the average flow of knowledge from node j to node i on the Neural-net

by

$$R_{ij} = -\alpha_{ij} - \tfrac{1}{2}E\{w_{ij}x_ix_j\} = -\alpha_{ij} - \tfrac{1}{2}w_{ij}P_iP_j \geq 0 \quad (18)$$

The probability due to the uncertainty of knowledge flow into node i, is calculated as in (9):

$$p(R_i) = \exp(- \alpha_i - \tfrac{1}{2}\Sigma_j w_{ij}P_iP_j) \quad (19)$$

where

$w_{ij} \geq 0$ is the interconnection weight between nodes i and j

$w_{ij} = 0$

$\alpha_i > 0$ is a probability normalizing factor.

The average Flow of Knowledge Ri into node i, is:

$$R_i = \alpha_i + \tfrac{1}{2}E\{ \Sigma_j(w_{ij}x_ix_j)\} = \alpha_i + \tfrac{1}{2}\Sigma_j(w_{ij}P_iP_j)$$

with probability $P(R_i)$, (Jaynes' Principle):

$$P(R_i) = \exp[-\alpha_i - \tfrac{1}{2}\Sigma_j(w_{ij}P_iP_j)]$$

The Entropy of Knowledge Flow in the machine is

$$H(R) = - \Sigma_i [P(R_i) \ln[P(R_i)] =$$

$$\Sigma_i[\alpha_i + \tfrac{1}{2}\Sigma_j(w_{ij}P_iP_j) \exp[-\alpha_i - \tfrac{1}{2}\Sigma_j(w_{ij}P_iP_j)] \quad (20)$$

The normalizing factor α_i is such that $\tfrac{1}{2}^n \leq P(R_i) \leq 1$.

The entropy is maximum when the associated probabilities are equal, $P(R_i i) = \tfrac{1}{2}^n$ with n the number of nodes of the network. By bounding $P(R_i)$ from below by $\tfrac{1}{2}^n$ one may obtain a unique minimization of the entropy corresponding to the most like sequence of events to be selected.

Unlike the regular Boltzmann machines, this formulation does not remove α_i when $P_i = 0$. Instead, the machine operates from a base entropy level $\alpha_i \exp(-\alpha_i)$ defined as the *Threshold Node Entropy* which it tries to reduce (Saridis and Moed, 1988).

c. Machine Decision Making, (DM) is the function of selecting the sequence with the largest probability of success.
This is accomplished through a search to connect a node ahead that will minimize the Entropy of Knowledge Flow at that node:

$$H(R_i) = (\alpha_i + \tfrac{1}{2}\Sigma_j w_{ij} P_i P_j) \exp[-\alpha_i - \tfrac{1}{2}\Sigma_j w_{ij} P_i P_j]$$

A modified genetic algorithm, involving a global random search, has been proposed by Moed and Saridis (1990) as a means of generating the best sequence of events that minimized the uncertainty of connections of the network expressed by the entropy (20).This algorthim, proven to converge globally compared favorably with other algorithms like the Simulated Annealing and the Random Search.

d. Machine Learning, (ML) (Feedback). Machine Learning is obtained by feedback devices that upgrade the probabilities P_i and the weights w_{ij} by evaluating the performance of the lower levels after a successful iteration.

For y_k representing either P_{ij} or w_{ij}, corresponding to the command c_k, the upgrading algorithms are:

$$y_k(t_k+1) = y_k(t_k) + \beta_k(t_k+1)[\Gamma(t_k+1) - y_k(t_k)] \quad (21)$$

$$J_k(t_k+1) = J_k(t_k) + \sigma_k(t_k+1)[V^k{}_{obs}(t_k+1) - J_k(t_k)]$$

where $J_k(t_k)$ is the performance estimate, $V^k{}_{obs}$ is the observed value and

$$P_i \quad : \Gamma_k(t_k+1) = x(t_k)$$

$$w_{ij} \quad : \Gamma_k(t_k+1) = \begin{cases} 1 & \text{if } J = \min_e J_e \\ 0 & \text{otherwise} \end{cases} \qquad (22)$$

e. Memory Exchange (ME), is the retrieval and storage of information from the *long-term memory*, based on selected feedback data from the lower levels after the completion of the complex task.

The above functions may be implemented by a two level Neural net, of which the nodes of the upper level represent the primitive objects s_i and the lower level of primitive actions relating the objects eai of a certain task. The purpose of the organizer may be realized by a search in the Neural net to connect objects and actions in the most likely sequence for an executable task.
Since it was agreed to use Petri Net Transducers (PNT) to model the coordinators at the next level, a Petri Net generator is required to create the Dispatcher's PNT for every task planned. This can be accomplished by another Boltzmann machine or a part of the existing plan generating architecture.

A graph of the Boltzmann machine with the appropriate symbols is given in Fig.3.

3. THE COORDINATION LEVEL

3.1 The Architecture

The *Coordinatiion level* is a tree structure of *Petri Net Transducers* as coordinators, with the Dispatcher as the root(Wang and Saridis 1990). Fig.4 depicts such a structure. The Petri Net Transducer for the Dispatcher is generated by the Organizer for every specific plan and is transmitted, asynchronously, to the Coordination level along with the plan to be executed. The function of the Dispatcher is to interpret the plan and assign individual

tasks to the other coordinators, monitor their operation, and transmit messages and commands from one coordinator to another as needed. As an example, a command is sent to the vision and sensing coordinator to generate a model of the environment, the coordinates of the objects for manipulation to be tabulated, and then transmitted to the motion coordinator for navigation and motion control. This command is executed by having each transition of the associated Petri Nets to initialize a package corresponding to a specific action (Mittman 1990). These packages are stored in short memories associated with each of the coordinators.

The rest of the coordinators have a fixed structure with alternatate menues available at request. They communicate commands and messages with each other, through the Dispatcher. They also provide information about reception of a message, data memory location, and job completion.

No data is communicated at the Coordination level, since the task planning and monitoring may be located in a remote station, and such an exchange may cause a channel congestion. A preferred configution for such situations is that the coordinators with a local dispatcher may be located with the hardware at the work site, while a remote dispatcher, connected to the organizer, interacts with local one from a remote position. Fig.9 depicts this architecture. This concept simplifies the communication problem considerably, since only short messages are transmitted back and forth through a major channel between local and remote stations, requiring a narrow bandwidth. An example of the effectiveness of such an architecture may be demonstrated in space construction, where robots work in space while task planning and monitoring is done on earth.

Eventhough, there is no limitation to the number of coordinators attached to the Dispatcher, only the following ones are planned for an Intelligent Robot for space applications.

Vision and Sensory Coordinator. This device coordinates all the sensory activities of the robot, with cameras and lazers, and generates information of the world model in cartesian coordinates.

Motion Control Coordinator. This device receives control, object and obstacle information and uses it to navigate and move multiple robotic arms and other devices, for object manipulation and task execution. It also assigns the appropriate operations on the data aquired for the desired application.

Planning Coordinator. The task plans, optimal and alternate generated by the Organizer are stored in this device for proper monitoring of execution and possible error recovery in cases of failure of the system.

Grasping Coordinator.This device coordinates the grippers of the armsand interfaces the proximity sensors for effective grasping.

Entropy measures, are developed at each coordinator such that they may be used to minimize the complexity and improve the reliability of the system (McInroy and Saridis 1990). A typical PNT system for the Coordination level of an Intelligent Robot as proposed by Wang and Saridis (1988) is given in Fig. 5.

3.2 The Analytic Model

Petri nets have been proposed as devices to communicate and control complex heterogenous processes. These nets provide a communication protocol among stations of the process as well as the control sequence for each one of them (Peterson 1977).

Abstract task plans, suitable for many environments are generated at the organization level by a grammar (Wang and Saridis 1990):

$$G = (N, \Sigma_o, P, S)$$

where

$N = \{S, M, Q, H\}$ = Non-terminal symbols

$\Sigma_o = \{A_1, A_2, \ldots A_n\}$ = Terminal Symbols (activities)

P = Production rules

Petri Net Transducers (PNT) proposed first by Wang and Saridis

(1990) are Petri net realizations of the *Linguistic Decision Schemata* introduced by Saridis and Graham (1984) as linguistic decision making and sequencing devices. They are defined as 6-tuples:

$$M = (N, \Sigma, \delta, G, \mu, F)$$

where

$N = (P, T, I, O)$ = A Petri net with initial marking μ,
Σ = a finite input alphabet
δ = a finite output alphabet
σ = a translation mapping from T x (Σ U {\}) to finite sets of δ* and F C R(μ) a set of final markings.

A *Petri Net Transducer (PNT)* is depicted in Figure 6. Its input and output languages are *Petri Net Languages (PNL)*. In addition to its on-line decision making capability PNT's have the potential of generating communication protocols, learning by feedback, ideal for the communication and control of coordinators and their dispatcher in real time. Their architecture is given in Figure 7, and may follow a scenario suitable for the implementation of an autonomous intelligent robot.

Figure 8 depicts the Petri Net Structure of a typical *Coordination Structure (CS)* of an intelligent robot. This structure is a 7-tuple:

$$CS = (D, C, F, R_D, S_D, R_C, S_C)$$

where

$D = (N_d, \Sigma_o, \delta_o, G_d, \mu_d, F_d)$ = The PNT dispatcher
$C = \{C_1, \ldots C_n\}$ = The set of coordinators
$C_i = (N^i_c, \Sigma^i_c, \delta^i_c, G^i_c, F^i_c)$ = the ith PNT coordinator
$F = U^n_{i=1}\{f^i_I, f^i_{SI}, f^i_O, f^i_{SO}\}$ = A set of connection points

R_D, R_C = Receiving maps for dispatcher and coordinators
S_D, S_C = Sending maps for dispatcher and coordinators

Decision making in the coordination structure is accomplished by *Task Scheduling* and *Task Translation*, e.g., for a given task find σ an enabled t such that $\sigma(t,a)$, is defined and then select the right translation string from $\sigma(t,a)$ for the transition t.

The sequence of events transmitted from the organization level is received by the dispatcher which requests a world model with coordinates from a vision coordinator.
The vision coordinator generates appropriate database and upon the dispatcher's command communicates it to the planning coordinator which set a path for the arm manipulator. A new command from the dispatcher sends path information to the motion controller in terms of end points, constraint surface and performance criteria. It also initializes the force sensor and proximity sensor control for grasp activities. The vision coordinator is then switched to a monitoring mode for navigation control, and so on.

The PNT can be evaluated in *real-time* by testing the computational complexity of their operation which may be expressed uniformly in terms of entropy. Feedback information is communicated to the coordination level from the execution level during the execution of the applied command. Each coordinator, when accessed, issues a number of commands to its associated execution devices (at the execution level). Upon completion of the issued commands feedback information is received by the coordinator and is stored in the *short-term memory* of the coordination level.

This information is stored in the short-term memory of the coordination level. This information is used by other coordinators if necessary, and also to calculate the individual, accrued and overall accrued costs related to the coordination level. Therefore, the feedback information from the execution to the

coordination level will be called *on-line, real-time* feedback information.

The performance estimate and the associated subjective probabilities are updated after the k_{ij}-th execution of a task $[(u_t,x_t)_i,S_j]$ and the measurement of the estimate of the observed cost J_{ij}:

$$J_{ij}(k_{ij}+1) = J_{ij}(k_{ij})+\beta(k_{ij}+1)[J_{obs}(k_{ij}+1)-J_{ij}(k_{ij})] \quad (23)$$

$$P_{ij}(k_{ij}+1) = P_{ij}(k_{ij})+\mu(k_{ij}+1)[\Gamma_{ij}(k_{ij}+1)-P_{ij}(k_{ij})]$$

where

$$\Gamma_{ij} = \begin{cases} 1 & \text{if } J_{ij} = \text{Min} \\ 0 & \text{elsewhere} \end{cases}$$

and β and μ are harmonic sequences. Convergence of this algorithm is proven in (Saridis and Graham 1984).

The *learning process* is measured by the entropy associated to the subjective probabilities. If

$$H(M) = H(E) + H(T/E) \quad (24)$$

where H(E) is the environmental uncertainty and H(T/E) is the pure translation uncertainty. Only the latter may be reduced by learning.

4. THE EXECUTION LEVEL.

4.1 The System and the Architecture

The *Execution level* contains all the hardware required by the Intelligent Machine to execute a task. There is a one-to-one corresondence between hardware groups and coordinators. Therefore

their structure is usually fixed. This level also contains all the drivers, VME buses, short memory units, processors, actuators and special purpose devices needed for the execution of a task. After the successful completion of a job feedback information is generated at this level for evaluation and parameter updating of the whole machine. Complexity dominates the performance of this level. Since *precision* is proportional to *complexity*, it also defines the amount of effort required to execute a task. It has been shown that all the activities of this level can be measured by entropy, which may serve as a measure of complexity as well. Minimization of local complexity through feedback, may serve as local design procedure. The localization of data exchange at this level provides a means of efficient remote control of the Intelligent Machine, (see Fig.9)

Because of the diversity of the hardware in a general purpose Intelligent Machine, this work will focus on the special case of a robot designed for space constrution like the CIRSSE transporter. The following hardware groups are available:

The Vision and Sensory System. This systems consists of two cameras fixed at the ceiling of the lab., two penlight cameras on the wrist of one PUMA arm, and a lazer rangefinder. They are all controlled by a Datacube with a versatile menu of various hardwired functions and a VME bus for internal communications. The functions assigned to them, e.g. create a world model in cartesian space, find the fiducial marks on the object to be manipulated, or track a moving object are supported by software specialized for the hardware of the system. Calibration and control of the hardware is an important part of the system. Since we are dealing with information processing the system's performancecan be easily measured with entropy. Actual data for visual servoing can be generated on the VME bus and transmitted through the Dispatcher to the Motion Control system. Direct connection of the VME bus with the Motion Control System is planned in the future.

The Motion Control System. This system is a unified structure for cooperative motion and force control for multiple arm manipulation. Since mtion affects force but not vice versa, motion control is designed independent of the constraint forces, and force control by treating inertial forces as disturbance. Integral force feedback is

used with full dynamics control algorithms. The resulting system, named CTOS, was developed as a multiple-processor, VME-bus based, real time robot control system for the CIRSSE 18-degree-of-freedom transporter. It hierarchically integrates the execution algorithms in planning, interaction, and servo control. It works together with the VXWORKS software and provides most of the transformations, and other kinematics and dynamics tools needed for servoing and manipulation. In earlier work it was shown that the control activities can be measured by entropy (Saridis 1985b). Therefore the measure of performance of the Motion Control System is consistent with the rest of the architecture of the Intelligent Machine.

The Grasping System. This system is planned to be separate from the Motion Control System. It would involve the grasping operations, the information gathering from various proximity sensors, and integration of these activities with the gripper motion control. It will be driven by a special coordinator, and provide information back of proper grasping for job control purposes. However at the present time it is only a subsystem of the Motion Control System and isfollows commands issued by the its Coordinator, for purposes of expediency.

4.2 Entropy Formulation of Motion Control.

The cost of control at the hardware level can be expressed as an entropy which measures the uncertainty of selecting an appropriate control to execute a task. By selecting an optimal control, one minimizes the entropy, e.g., the uncertainty of execution. The entropy may be viewed in the respect as an energy in the original sense of Boltzmann, as in Saridis (1988).

Optimal control theory utilizes a non-negative functional of the state of the system $x(t) \in \Omega_x$ the state space, and a specific control $u(x,t) \in \Omega_u \times T$; $\Omega_u \subset \Omega_x$ the set of all admissible feedback controls, to define the performance measure for some initial conditions $x_0(t_0)$, representing a generalized energy function, of the form:

$$V(x_0,t_0) = E\{\int_{t0}^{tf} L(x,t;u(x,t))\ dt\} \quad (25)$$

where $L(x,t;u(x,t)) > 0$, subject to the differential constraints dictated by the underlying process

$$\begin{aligned} dx/dt &= f(x,u(x,t),w,t); \quad & x(t_0) &= x_0 \\ z &= g(x,v,t); & x(t_f) &\in M_f \end{aligned} \tag{26}$$

where x_0, $w(t)$, $v(t)$ are random variables with associated probability densities $p(x_0)$, $p(w(t))$, $p(v(t))$ and Mf a maninfold in Ω_x. The trajectories of the system (26) are defined for a fixed but arbitrarily selected control $u(x,t)$ from the set of admissible feedback controls Ω_u.

In order to express the control problem in terms of an entropy function, one may assume that the performance measure $V(x_0,t_0,u(x,t))$ is distributed in u according to the probability density $p(u(x,t))$ of the controls $u(x,t) \in \Omega_u$. *The differential entropy* H(u) corresponding to the density is defined as

$$H(u) = - \int_{\Omega u} p(u(x,t)) \ln p(u(x,t)) \, dx$$

and represents the uncertainty of selecting a control $u(x,t)$ from all possible admissible feedback controls Ω_u. The optimal performance should correspond to the maximum value of the associated density $p(u(x,t))$. Equivalently, the optimal control $u^*(x,t)$ should minimize the entropy function H(u).

This is satisfied if the density function is selected to satify *Jaynes' Principle of Maximum Entropy* (1956), e.g.,

$$p(u(x,t)) = \exp\{-\lambda - \mu V(x_0,t_0;u(x,t))\} \tag{27}$$

where λ and μ are normalizing constants.

It was shown by Saridis (1985b) that the expression H(u) representing the entropy for a particular control action $u(x,t)$ is

given by:

$$H(u) = \int_{\Omega u} p(x,t;u(x,t))V(x_0,t_0;u(x,t))\ dx = \\ = \lambda + \mu V(x_0,t_0;u(x,t)) \qquad (28)$$

This implies that the average performance measure of a feedback control problem corresponding to a specifically selected control, is an entropy function. The optimal control $u^*(x,t)$ that minimizes $V(x_0,t_0;u(x,t))$, maximizes $p(x,t;u(x,t))$, and consequently minimizes the entropy $H(u)$.

$$u^*(x,t) : E\{V(x_0,t_0;u^*(x,t))\} \\ = \min \int_{\Omega u} V(x_0,t_0;u(x,t))p(u(x,t))dx \qquad (29)$$

This statement is the generalization of a theorem proven in (Saridis 1988) and establishes equivalent measures between information theoretic and optimal control problem and provides the information and feedback control theories with a common measure of performance.

4.3 Entropy Measure of the Vision System.

The optimal control theory designed mainly for motion control, can be implemented for vision control, path planning and other sensory system pertinent to an Intelligent Machine by slightly modifying the system equations and cost functions. After all one is dealing with real-time dynamic systems which may be modeled by a dynamic set of equations.

A *Stereo Vision* system of a pair of cameras mounted at the end of a robot arm, may be positioned at N different view points to reduce problems with noise, considered one at a time due to time limitations. The accuracy of measuring the object's position depends upon its relative position in the camera frame. Consiquently, each viewpoint will have different measurement error

and time statistics. These statistics may be generated to define the uncertainty of the measurement of the Vision system as in McInroy and Saridis (1991).

For a point c of the object, the measurement error of its 3-D position in the camera coordinate frame e_{pc} is given by:

$$e_{pc} = M_c \, n_c \tag{30}$$

where n_c is the 3-D image position errors, and M_c an appropriate 3×3 matrix, depending on the position of the object.
The linearized orientation error is given by:

$$\delta = (M^T M)^{-1} M^T M' F n \tag{31}$$

where

δ is the orientation error in the camera frame,
M is a matrix formed from camera coordinate frame positions,
M' is a constant matrix,
F is the matrix formed from the camera pameters and measured positions,
n is the vector of the image position errors at the four points.

A vector containing the position and orientation errors due to image noise is given by:

$$e_c = [e^T_{pc} \delta^T]^T = Ln \tag{32}$$

where L depends on the camera parameters and the four measured camera frame positions of the points. The statistics of the image noise n, due to individual pixel errors are assumed to be uniformely distributed. Assuming that feature matching centroids is used by the vision system, its distributions tend to be independent Gaussian, due to the Central Limit Theorem.

$$n \approx N(0, C_v) \quad \text{and} \quad e_c \approx N(0, LC_vL^T) \tag{33}$$

The time which each vision algorithm consumes is also random due to the matching period. Therefore the total vision time, for the ith Algorithm that includes camera positioning time, image processing time, and transformation to the base frame, is assumed Gaussian:

$$t_{vi} \approx N(\mu_{tvi}, \sigma^2_{tvi}). \qquad (34)$$

Once the probability density functions are obtained, the resulting Entropies $H(t_{vi})$, and $H(e_c)$, are obtained in a straight forward manner for the ith Algorithm (McInroy and Saridis 1991):

$$H(t_{vi}) = \ln\sqrt{2\pi e\sigma^2_{tvi}} \qquad (35)$$
$$H(e_c) = \ln\sqrt{(2\pi e)^6 \det[C_v]} + E\{\ln[\det L_i]\}$$

The total Entropy, may be used as a measure of uncertainty of the Vision system (imprecision), and can be minimized wrt. the available system parameters:

$$H(V) = H(t_{vi}) + H(e_c). \qquad (36)$$

5. APPLICATION TO ROBOTIC SYSTEMS.

The theory of Intelligent Controls has direct application to the design of Intelligent Robots. The IPDI provides a means of structuring hierarchically the levels of the machine. Since for a passive task the flow of knowledge through the machine must be constant, it assigns the highest level with the highest machine intelligence and smallest complexity (size of data base), and the lowest level with the lowest machine intelligence and largest complexity. Such a structure agrees with the concept of most organizational structures encountered in human societies. Application to machine structures is straight forward.

Even at the present time there is a large variety of applications

for intelligent machines. Automated material handling and assembly in an automated factory, automation inspection, sentries in a nuclear containment are some of the areas where intelligent machines have and will find a great use. However, the most important application for the author's group is the application of Intelligent Machines to unmanned space exploration where, because of the distance involved, autonomous anthropomorphis tasks must be executed and only general commands and reports of executions may be communicated (see Wang Kyriakopoulos et al. 1990).

Such tasks are suitable for intelligent robots capable of executing anthropomorphic tasks in unstructured uncertain environments. They are structured uncertain environment.

They are structured usually in a human-like shape and are equipped with vision and other tactile sensors to sense the environment, two areas to execute tasks and locomotion for appropriate mobility in the unstructured environment. The controls of such a machine are performed according to the Theory of Intelligent Machines previously discussed (Saridis and Stephanou 1977), (Saridis 1983, 1985a, 1985b, 1988a), (Meystel 1985, 1986). The three levels of controls, obeying the *Principle of Increasing Precision with Decreasing Intelligence*, are presently tested on a testbed composed of two PUMA 600 robot arms with stereo vision and force sensing, with the structure of Figure 10.

Recent research has been focused in the application of the Theory of Intelligent Machines to design robots for autonomous manipulation and locomotion in space. Satellite maintenance, construction of the space station and autonomous planet exploration vehicles are typical examples. A testbed for earth simulation of such activities in space has been built in the Center for Intelligent Robotics for Space Exploration at Rensselaer and graphically depicted in Figure 11.

5. CONCLUSIONS.

The architecture described in this paper does not differ substantially from the architecture originally proposed by Saridis (1979). The details have been more elaborated and more efficient internal structures have been used. The main contribution though is that this system is been successfully implemented and that the resulting structure is extremely efficient, effective, versatile, capable for remote operation as compared to other proposed architectures. Evaluation results will be reportd in a follow-up paper.

ACKNOWLEDGEMENTS

This work was supported by NASA Grant NAGW 1333.

REFERENCES

Albus, J.S. (1975), "A New Approach to Manipulation Control: The Cerebellar Model Articulation Controller", *Transactions of ASME, J. Dynamics Systems, Measurement and Control, 97*, pp. 220-227.

Hinton, G.E. and Sejnowski, T.J. (1986), "Learning and Relearning in Boltzmann Machines", pp. 282-317, in *Parallel Distributed Processing*, eds. D.E. Rumelhart and J.L. McClellan, MIT Press.

Jaynes, E.T. (1957), "Information Theory and Statistical Mechanics", *Physical Review*, pp. 106, 4.

Kolmogorov, A.N. (1956), "On Some Asymptotic Characteristics of Completely Bounded Metric Systems", *Dokl Akad Nauk*, SSSR, 108, No. 3, pp. 385-389.

McInroy J.E., Saridis G.N.,(1991), "Reliability Based Control and Sensing Design for Intelligent Machines", in *Reliability Analysis* ed. J.H. Graham, Elsevier North Holland, N.Y.

Meystel, A. (1986), "Cognitive Controller for Autonomous Systems", *IEEE Workshop on Intelligent Control 1985*, p. 222, RPI, Troy, New York.

Mittman M.,(1990), "TOKENPASSER: A Petri Net Specification Tool" Master's Thesis, Rensselaer Poly. Inst., Dec.

Moed, M.C. and Saridis, G.N. (1990), "A Boltzmann Machine for the Organization of Intelligent Machines", *IEEE Transactions on Systems Man and Cybernetics, 20*, No. 5, Sept.

Nilsson, N.J. (1969), "A Mobile Automaton: An Application of Artificial Intelligence Techniques", *Proc. Int. Joint Conf. on AI*, Washington, D.C.

Peterson, J.L. (1977), "Petri-Nets", *Computing Survey*, 9, No. 3, pp. 223-252, September.

Saridis, G.N. (1977), *Self-Organizing Controls of Stochastic Systems*, Marcel Dekker, New York, New York.

Saridis, G.N. (1979), "Toward the Realization of Intelligent Controls", *IEEE Proceedings, 67*, No. 8.

Saridis, G. N. (1983), "Intelligent Robotic Control", *IEEE Trans. on AC, 28*, 4, pp. 547-557, April.

Saridis, G.N. (1985b), "Control Performance as an Entropy", *Control Theory and Advanced Technology, 1*, 2, pp. 125-138, Aug.

Saridis, G.N. (1985c), "Foundations of Intelligent Controls", *Proceedings of IEEE Workshop on Intelligent Controls*, p. 23, RPI, Troy, New York.

Saridis, G.N. (1988), "Entropy Formulation for Optimal and Adaptive Control", *IEEE Transactions on AC, 33*, No. 8, pp. 713-721, Aug.

Saridis, G.N. (1989), "Analytic Formulation of the IPDI for Intelligent Machines", *AUTOMATICA the IFAC Journal, 25*, No. 3, pp. 461-467.

Saridis, G.N. and Graham, J.H. (1984), "Linguistic Decision Schemata for Intelligent Robots", *AUTOMATICA the IFAC Journal, 20*, No. 1, pp. 121-126, Jan.

Saridis, G.N. and Moed, M.C. (1988), "Analytic Formulation of Intelligent Machines as Neural Nets", *Symposium on Intelligent Control*, Washington, D.C., August.

Saridis, G.N. and Stephanou, H.E. (1977), "A Hierarchical Approach to the Control of a Prosthetic Arm", *IEEE Trans. on SMC, 7*, No. 6, pp. 407-420, June.

Saridis, G.N. and Valavanis, K.P. (1988), "Analytical Design of Intelligent Machines", *AUTOMATICA the IFAC Journal, 24*, No. 2, pp. 123-133, March.

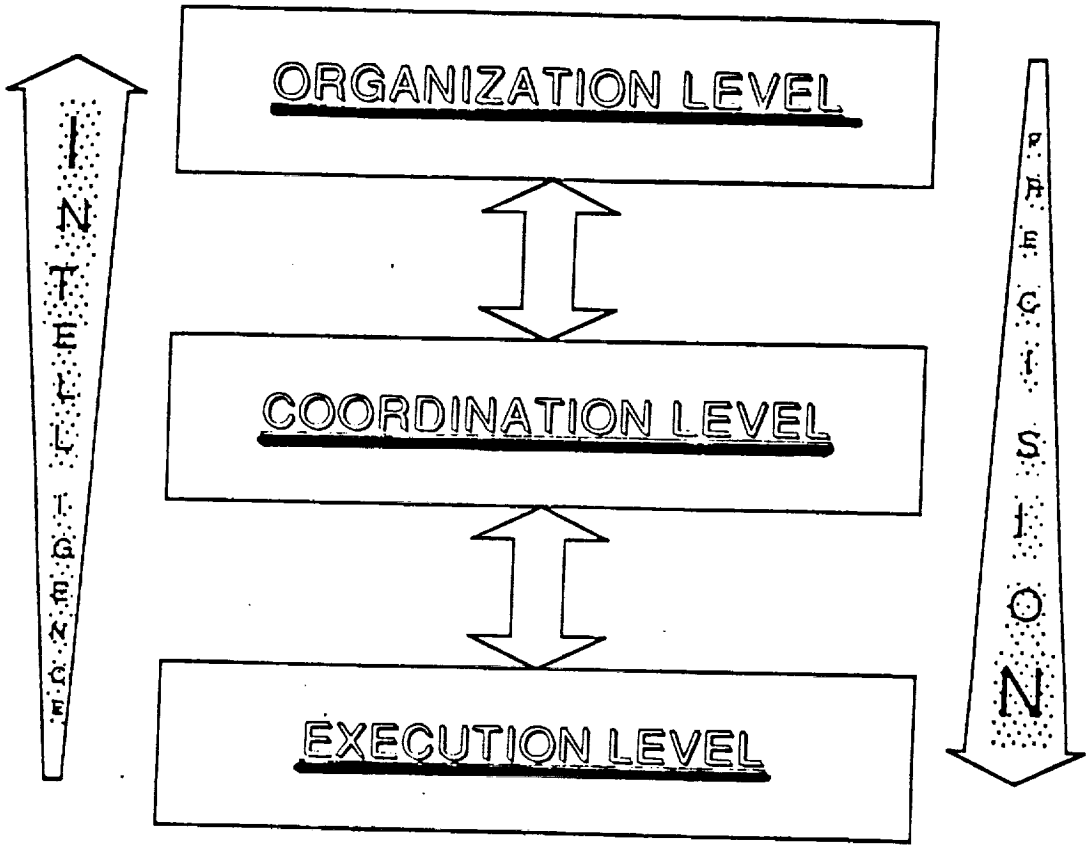
Shannon, C. and Weaver, W. (1963), *The Mathematical Theory of Communications*, Illini Books.

Wang, F., Kyriakopoulos, K., Tsolkas T., Saridis, G.N., (1990) "A Petri-Net Coordination Model of Intelligent Mobile robots" *CIRSSE Technical Report #50*, Jan.

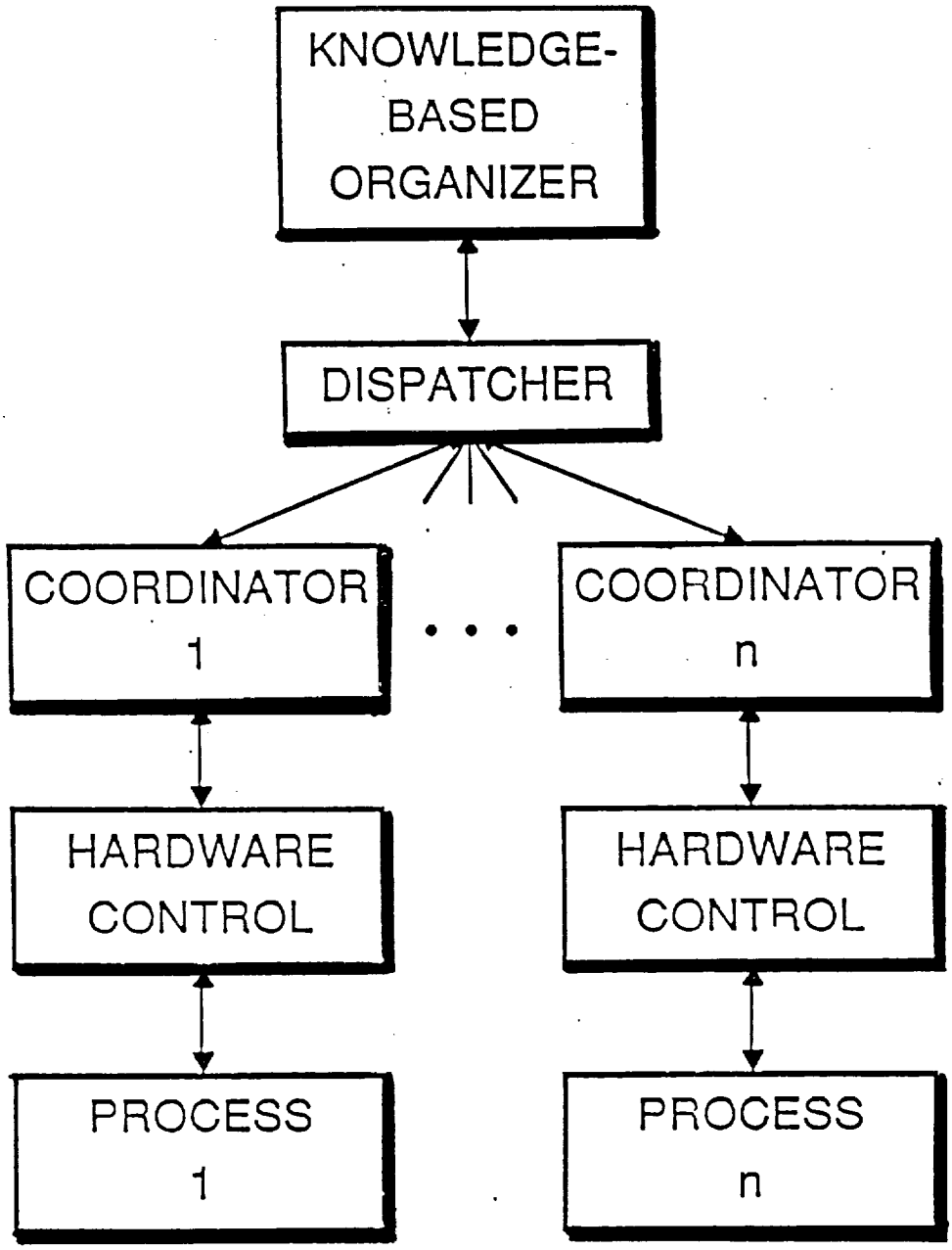
Wang, F. and Saridis, G.N. (1988), "A Model for Coordination of Intelligent Machines Using Petri Nets", *Symposium on Intelligent Control*, Washington, D.C., August.

Wang, F., Saridis, G.N. (1990) "A Coordination Theory for Intelligent Machines" *AUTOMATICA the IFAC Journal, 35*, No. 5, pp. 833-844, Sept.

Zames, G. (1979), "On the Metric Complexity of Casual Linear Systems, ϵ-entropy and ϵ-dimension for Continous Time", *IEEE Trans. Automatic Control, 24*, No. 2, pp. 220-230, April.

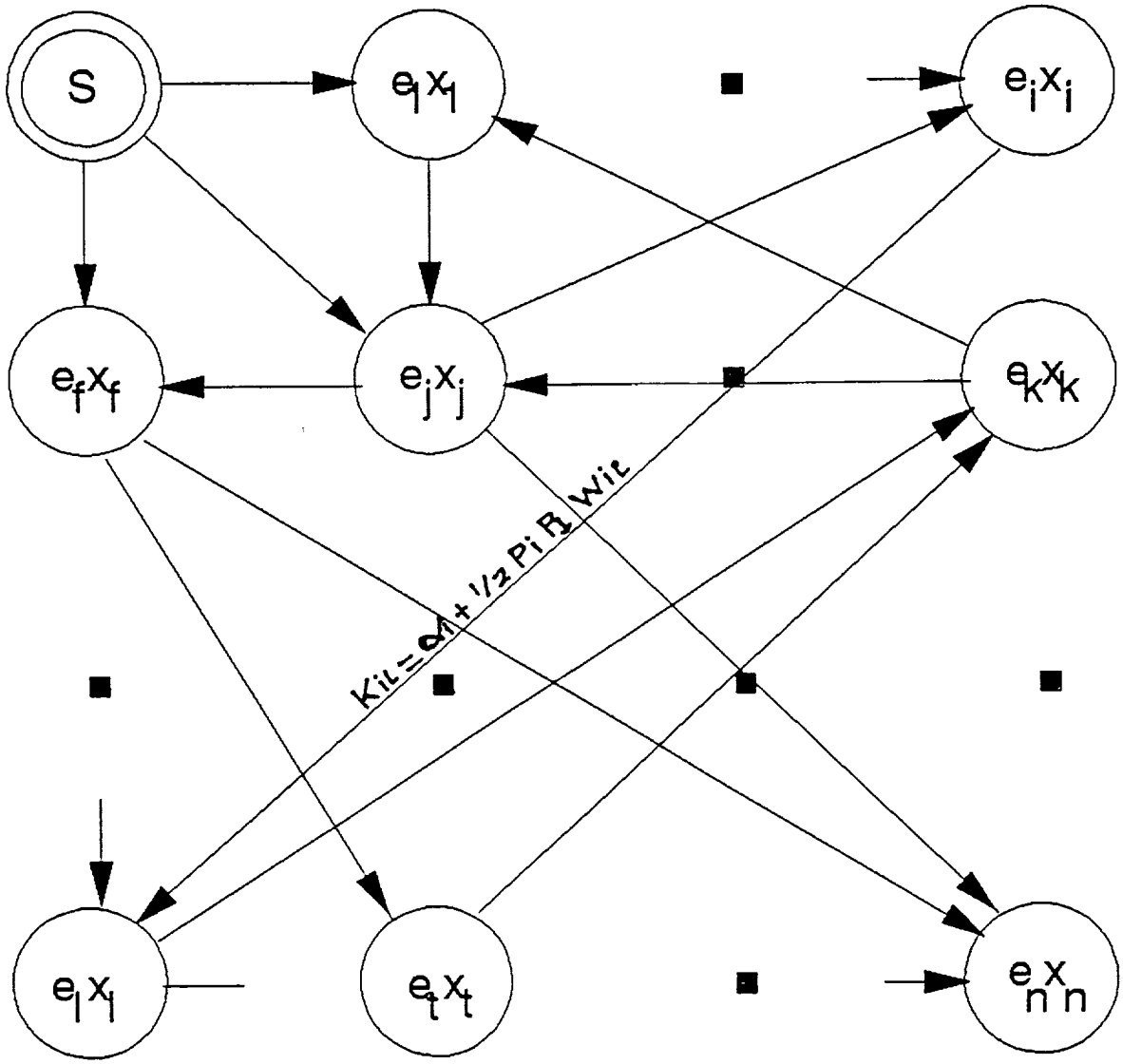


The Structure of Intelligent Machines



Hierarchical Intelligent Control System.

DIRECTED GRAPH



THE BOLTZMANN MACHINE FOR THE ORGANIZATION LEVEL

e_i = primitive event
x_i = state of $e_i c\{1,0\}$; with prob. p_i
K_i = energy at node i, $K_i = \alpha_i + \frac{1}{2}\sum_j p_i p_j w_{ij}$.
w_{ij} = learned weights
$p(K_i)$ = probability of connection i-j.

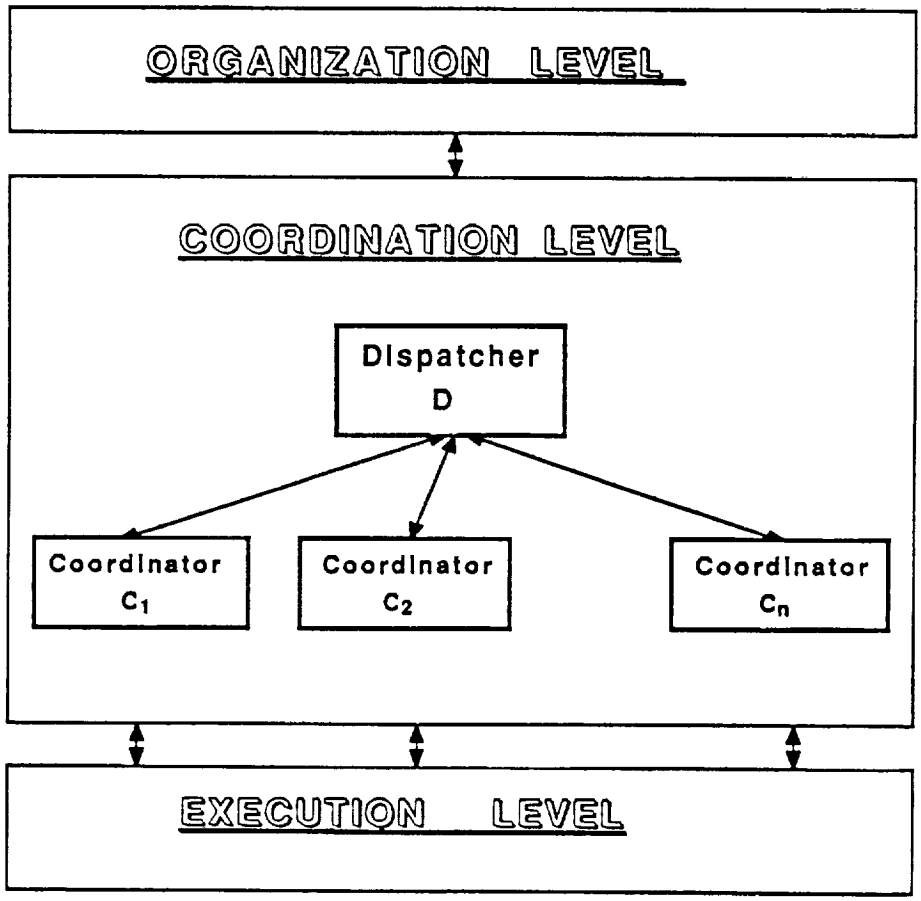
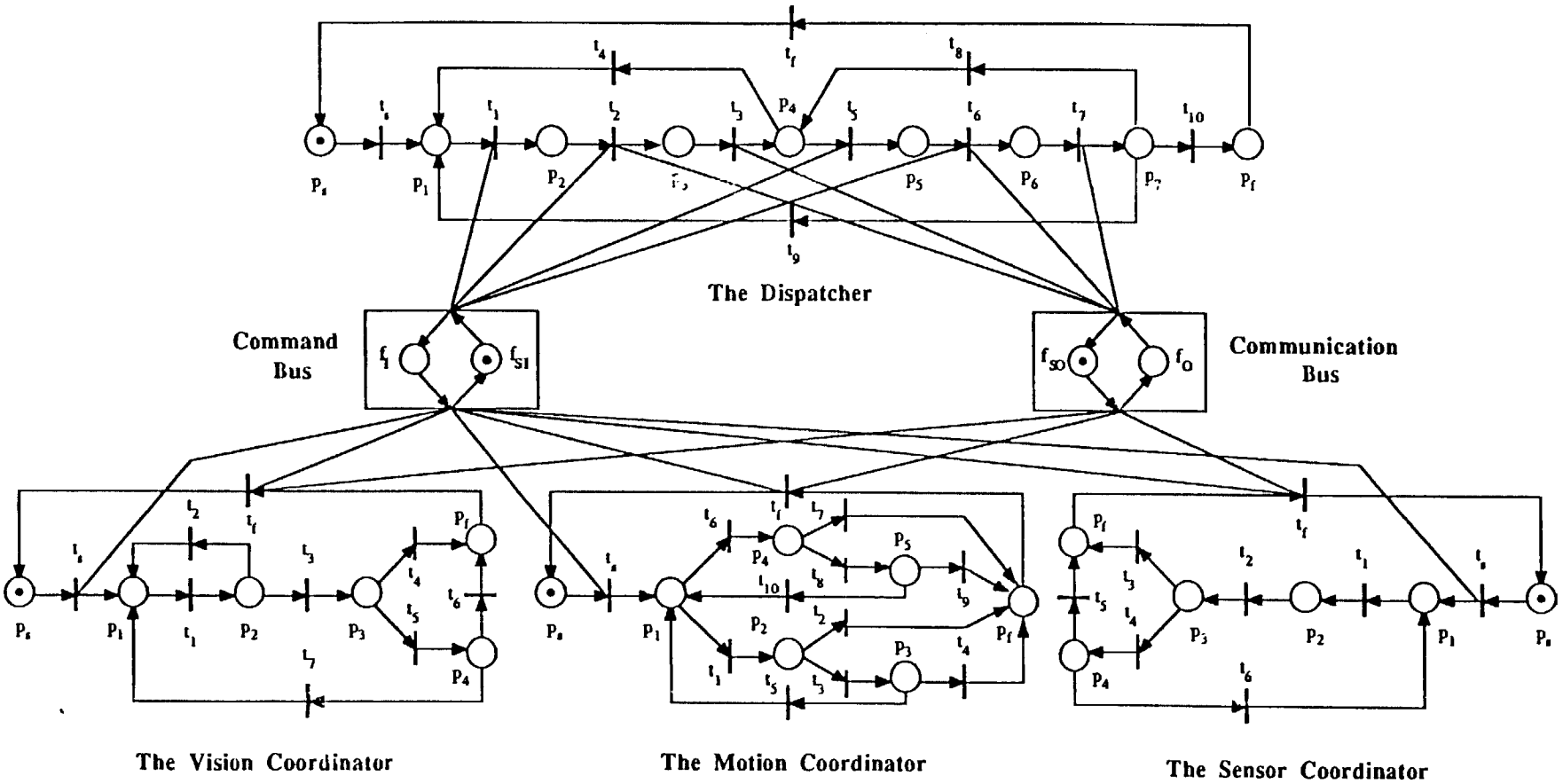
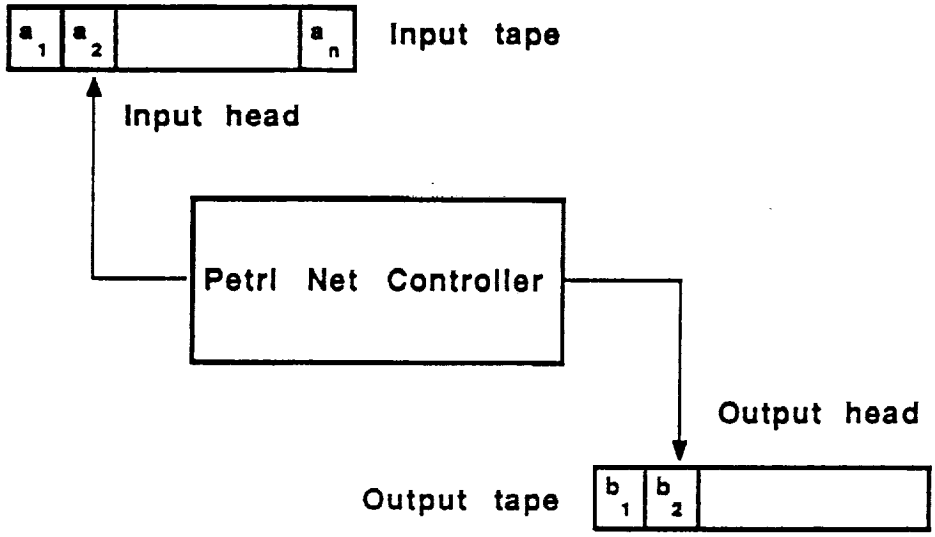


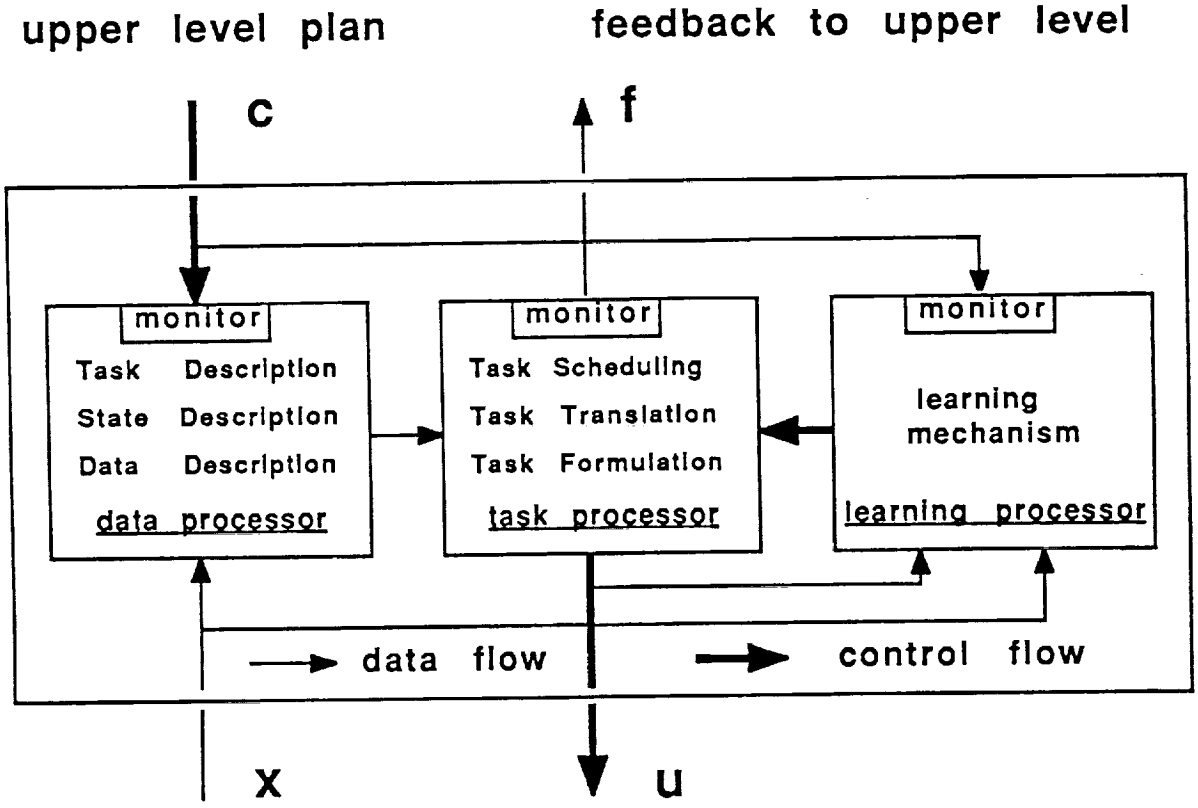
Fig.4 Topology of the Coordination Level



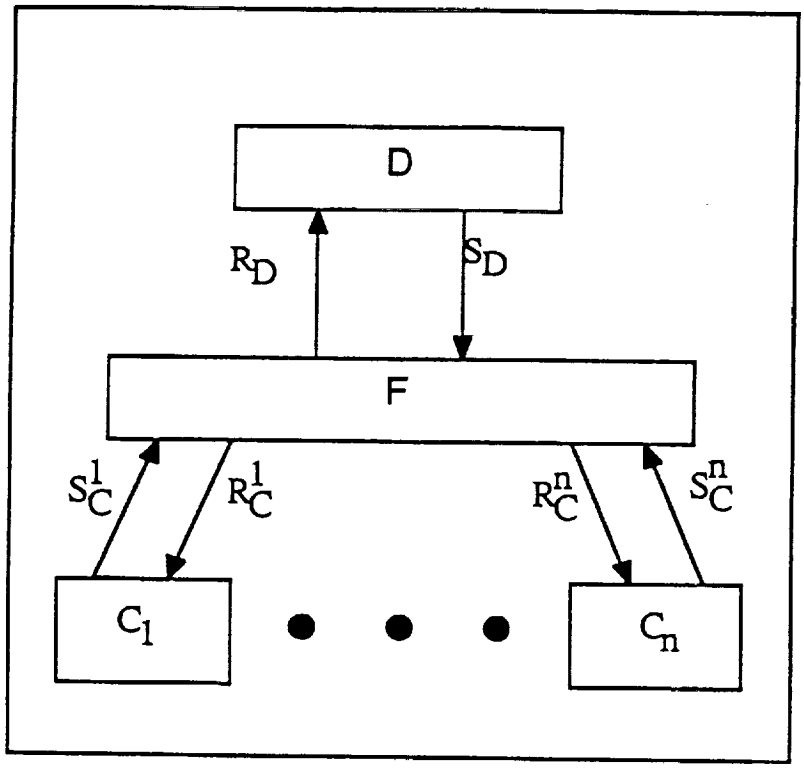
The Coordination Structure of An Intelligent Manipulator System



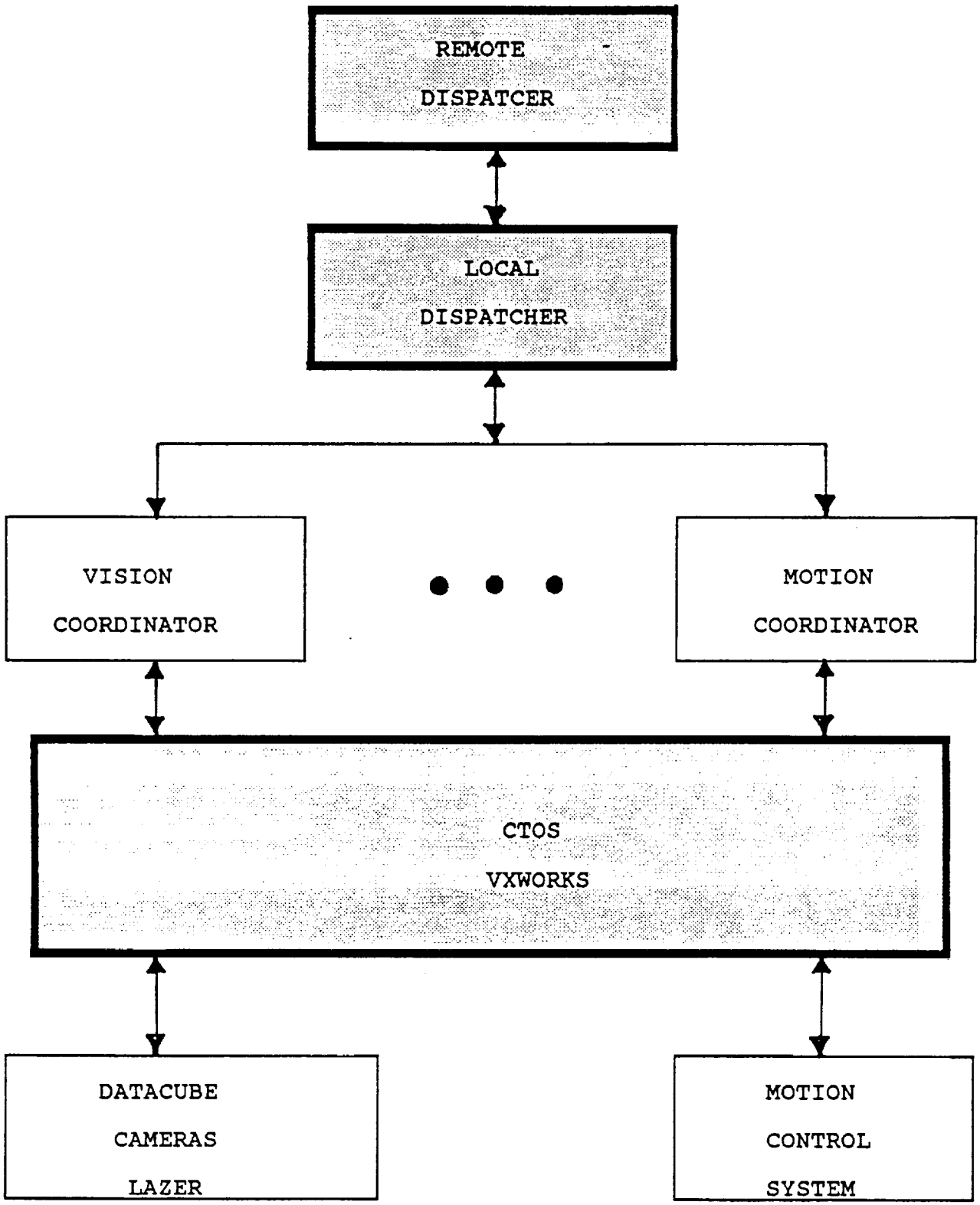
Petri Net Transducer (PNT)



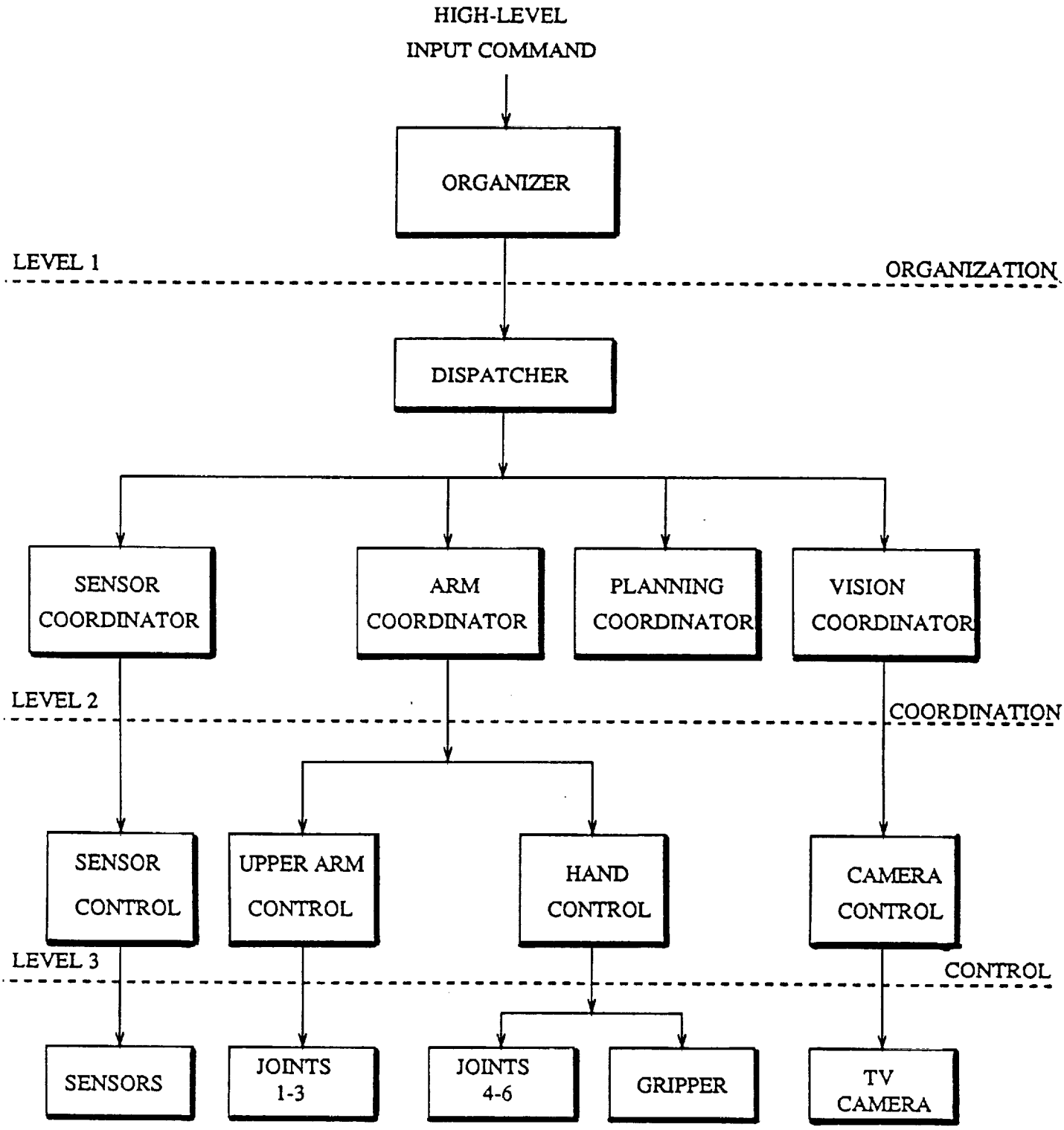
A UNIFORM ARCHITECTURE FOR THE DISPATCHER AND COORDINATORS



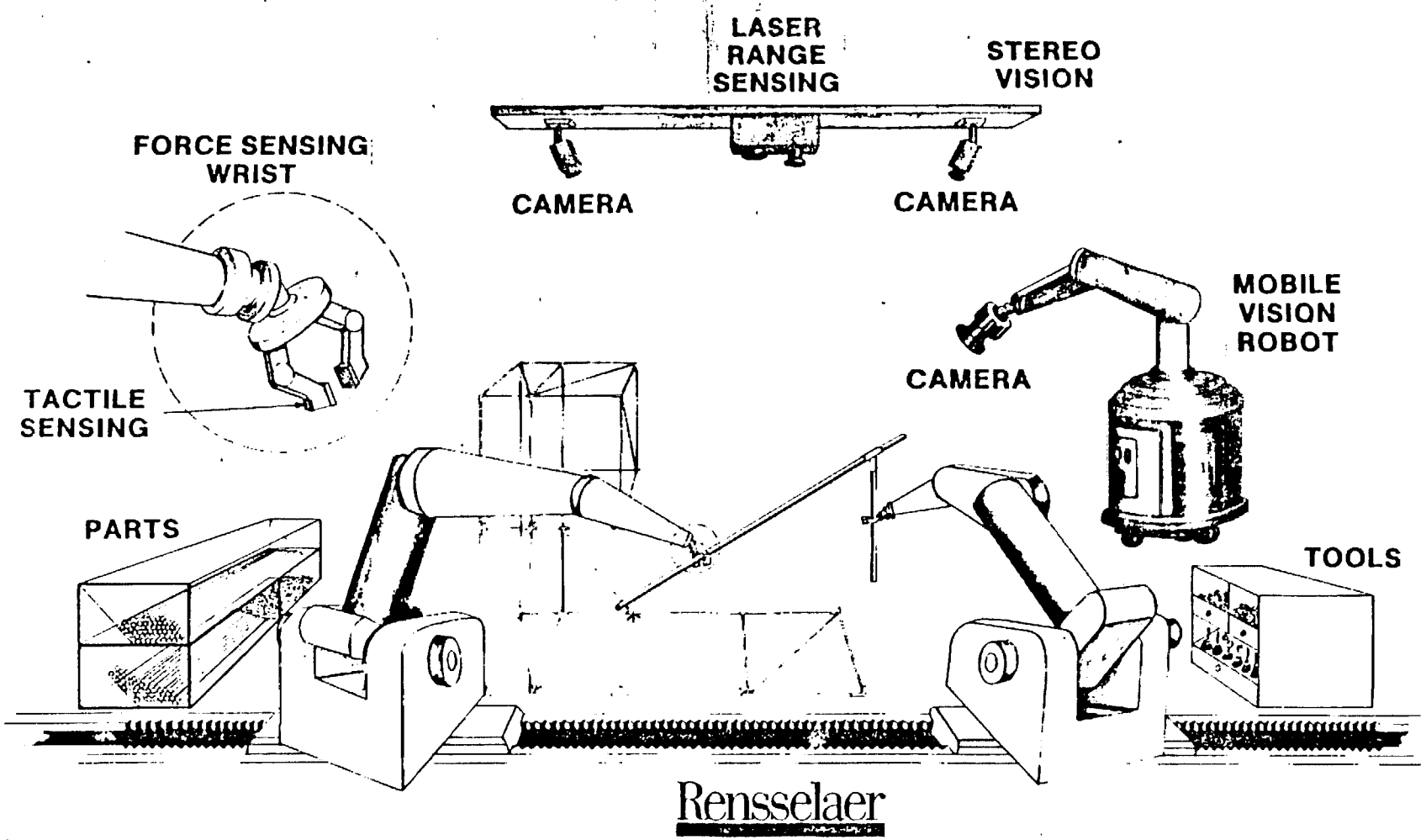
The Coordination Structure



TELEROBOTICS TESTBED CONFIGURATION



Hierarchically Intelligent Control for a Manipulator with Sensory Feedback



Rensselaer

CIRSSE LABORATORY

ROBOTIC ASSEMBLY OF IN-SPACE STRUCTURES