A PERFORMANCE MEASURE FOR INTELLIGENT MACHINES BASED ON COMPLEXITY AND RELIABILITY

P. LIMA*, G. SARIDIS**

Dept. Engenharia Electrotécnica, Instituto Superior Técnico, Av. Rovisco Pais, 1 - 1096 Lisboa Codex, PORTUGAL. E-mail: d2116@alfa.ist.utl.pt. JNICT grant #BD/1357/91-IA.

***Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, U.S.A.. E-mail: saridis@ral.rpi.edu. NASA grant #NAGW-1333.*

Abstract. This paper introduces a formalism which combines *reliability* and *complexity* in a measure of performance for Intelligent Machines. For a given **desired accuracy**, different algorithms may be available which are reliable enough. Reliability is defined as the probability that an algorithm meets, at run-time, the accuracy specifications which guided its design. However it is important to have a means of choosing the algorithm of least cost among the reliable ones. The cost measures resources usage, such as CPU time, memory space or number of processors. The Theory of Information-Based Complexity provides a formalism to deal with different sources of information, thus with distinct algorithms at all levels of the machine. Two case studies on motion control and image processing illustrate the method.

Key Words. Hierarchical Intelligent Machines, Performance Measures, Complexity, Reliability.

1. INTRODUCTION

The analytic approach to the theory of Intelligent Machines has developed in the past few years a formalism for the architecture of an Hierarchical and Goal-Directed Intelligent Machine (HGDIM), as described by Valavanis and Saridis (1992).

Other authors (Technical Committee on Intelligent Control, 1994) have also pointed out that an analytic design based on measures of performance assures some degree of certainty about the measurability of that design. Also, an approach somewhat different from traditional control cost functionals is needed, since a myriad of different sources of information is present in such a machine. To accommodate this, the general goal of the Analytic Theory of Intelligent Machines has been to decrease of *entropy* at all levels of the HGDIM, since entropy is a measure of information, a concept manipulated by all the different algorithms. Reliability has been proposed as an equivalent measure of entropy by McInroy and Saridis (1994). Complexity was first included in the performance function by Lima and Saridis (1993).

In this paper a complementary approach to the use of reliability as a performance measure is introduced. For a specified accuracy, different algorithms may be available which solve the problem with a reliability greater than some threshold. Hence it is important to have the means to choose the algorithm of least *computational cost* among the reliable ones. Another point of view, followed here, is to combine cost and reliability in a cost function which balances the two and guides the decision process. When uncertainty is present, *Information Based Theory of Complexity*, introduced by Traub *et al* (1988), provides the background to formalize a coherent definition of reliability and cost. Also, this theory is capable of dealing with different sources of information, thus with distinct algorithms at all levels of the machine.

The paper is organized as follows: after this introduction, section 2 briefly describes the formalism of Information-Based Complexity. The core of the paper is section 3 where the performance function based on cost and reliability is introduced, as well as the equations to propagate it from the Execution level to the other 2 levels of the HGDIM: Organization and Coordination. Two applications of the theory to Intelligent Robotic Systems are described in section 4. Section 5 concludes the paper.

2. INFORMATION-BASED COMPLEXITY

Control problems associated to Intelligent Machines, whether they consist of servomechanisms, adaptive systems, robotic vision, neural net or fuzzy logic controllers, deal with information of all kinds, and this information is often partial, noisy and costly. Moreover, Intelligent Machines frequently move within strongly uncertain environments, and their goal is to reduce the degree of uncertainty in controlling these environments. Hence, in the sequel the focus will be on Information-Based Complexity, not on Combinatorial Complexity (Traub *et al* (1988)).

2.1. *Problem Formulation*

For each $f \in F$, where F is a set of *problem elements*, it is desired to compute an approximation $U(f)$ of $S(f)$, where $S : F \to G$ is called a *problem solution* and G is a normed linear space over the scalar field of real or complex numbers. To measure the distance between $S(f)$ and $U(f)$ an absolute error criterion, $\|S(f) - U(f)\|$, is used, where $\|(.)\|$ represents some norm defined in G.

$U(f)$ is an ϵ-*approximation* of $S(f)$ iff $\|S(f) - U(f)\| \leq \epsilon \geq 0$. ϵ is called the *accuracy* of the approximation.

2.2. *Information*

It is assumed that the only initial knowledge about f is that it belongs to the set F, and also that more knowledge about f may be gathered using computations of the form $L(f)$, $L : F \rightarrow H$, for some set H.

H may assume several different forms. For example, it may either be the set $\{0,1\}$ of answers to a question like "what is the intensity value of pixel (i,j) in some black-and-white image?" or the set of real numbers when the information consists on the collection of a function and its derivative values at some point x, $L_i(f) = f^{(i)}(x),\ 0 \leq i \leq r$.

The information $\mathcal{I}(f)$ is then defined as $\underline{\mathcal{I}}(f) = (L_1(f), L_2(f), \ldots, L_n(f))^T,\ \forall f \in F$.

$U(f,\phi) = \phi(\underline{\mathcal{I}}(f))$ where $\phi(\underline{\mathcal{I}}(f)) \in G$ is an *algorithm* that computes an approximation of $S(f)$ given the information $\underline{\mathcal{I}}(f)$.

2.3. *Model of Computation*

The initial assumptions are:

- either a sequential or parallel model of computation is assumed;
- there is a charge for each information operation;
- all information and combinatorial operations are performed with infinite precision and finite cost.

The model postulates a constant cost c for each information operation $L(f) \in \Lambda$ and unit cost for each combinatory operation performed by ϕ over $\underline{\mathcal{I}}(f)$.

The *cost* of an algorithm ϕ has two components:

$$\text{cost}(\phi, f) = c_i(\underline{\mathcal{I}}(f), f) + c_p(\phi, \underline{\mathcal{I}}(f)) \tag{1}$$

where c_i is the cost of getting information about f needed by algorithm ϕ, and c_p is the combinatorial cost of processing that information by algorithm ϕ. Given the above, $c_i(\underline{\mathcal{I}}(f), f) = c|\underline{\mathcal{I}}(f)|$, where $|\underline{\mathcal{I}}(f)|$ denotes the cardinality of $\underline{\mathcal{I}}(f)$, that is, the number of information operations. The term c_i is inherent to information-based complexity. Information is gathered to reduce uncertainty. c_p would be the only term in the absence of uncertainty. Depending on the model used, different features are weighted (CPU time, memory space, number of processors).

2.4. *Coherent Definition of Reliability and Cost*

In order to coherently combine the definitions of cost and reliability for a given problem, the key is the desired accuracy or error specification ϵ for the problem, which must be the same in both definitions, as explained before.

ϵ-cost (*cost* for short) of a problem is defined as the minimal cost among the set Φ_{feas} of all available and feasible algorithms which solve the problem with error defined in the probabilistic sense:

$$\epsilon\text{-cost} = \inf_{\phi \in \Phi_{feas}} \{\text{cost}(\phi)\} \tag{2}$$

Suppose now that $\underline{S}(f)$ is a vector of specifications for a given problem. The problem solution $\underline{S}(f)$ is for example the desired overshoot of a control algorithm implementing a **move robot** event, and the problem element f is the output signal used to compute $\underline{U}(f,\phi)$.

Given this, *Reliability* is defined as:

$$\begin{aligned} R(\phi, f) &= \Pr\{\text{specification error} < \epsilon\} \\ &= \Pr\{\|\underline{S}(f) - \underline{U}(f,\phi)\| < \epsilon\} \end{aligned} \tag{3}$$

for some desired accuracy ϵ. Model-based computations of reliability often assume gaussian distribution (McInroy and Saridis, 1994) or no specific distribution (Musto and Saridis, 1993) for the probability of the specification error. If the reliability is learned from successes and failures along time (Lima, 1994), no distribution must be specified either.

Under the probabilistic setting the error of estimating $\underline{S}(f)$ by $\underline{U}(f,\phi)$ (the result of algorithm ϕ), is kept below ϵ, except in a subset of G with measure δ. Now, making $R_d = 1 - \delta$, where R_d is the desired reliability lower-bound, the coherent definition of cost is obtained:

$$\begin{aligned} f^* &= \arg\inf_{f \in F} \{R(\phi, f) \ni R(\phi, f) \geq R_d\} \\ C(\phi) &= \text{cost}(\phi, f^*) \end{aligned} \tag{4}$$

that is, among all $f \in F$ capable of keeping the specification error for algorithm ϕ below ϵ with reliability at least R_d, the one leading to the worst-case, i.e. the f leading to the larger probability of error, is picked. Here and henceforth, the reliability will be denoted as $R(\phi) = R(\phi, f^*)$.

For example, N image frames or more need to be averaged to increase to a certain value the probability that the error of locating an object in a noisy image is below ϵ. Every image resulting from the average of a different number frames is a problem element. If the cost of processing that information is not considered, the overall cost will be equal to c_i and proportional to the number of averaged frames. Among the number of image frames which have to be averaged, N corresponds to the worst-case specification error. A greater number of averages will decrease the error probability, while a smaller number will push the corresponding approximated problem solution to the subset of G with measure δ, for which $\Pr\{\|\underline{S}(f) - \underline{U}(f,\phi)\| < \epsilon\} < 1 - \delta$.

The link between the definitions of reliability and cost is the assumption that all algorithms are designed to meet an error specification ϵ of the problem they can solve. Given some desired reliability for the problem, the cost of obtaining that reliability can be determined for each of the algorithms, according to the cost measure defined (number of operations, elapsed CPU-time, memory used) for the problem. Conversely, if the cost measure is fixed at different values for the different algorithms, this will correspond to different reliabilities for each of them.

3. A PERFORMANCE MEASURE FOR INTELLIGENT MACHINES

When dealing with very large systems, some amount of uncertainty exists in the model of the system to be

controlled. Hence there is always uncertainty about the result of a given command sent to the controlled system.
The different algorithms used at the Execution Level of an Intelligent Machine (Valavanis and Saridis (1992)) are frequently designed in order to meet a set of specifications or, without loss of generality, in order to keep the error of a set of involved variables below some desired *accuracy* ϵ.
The uncertainty involved in the design of these algorithms is mostly due to approximate or incomplete modeling, and statistical fluctuations around nominal parameters. Hence it can be modeled statistically. McInroy and Saridis (1994) and Musto and Saridis (1993) describe algorithm selection techniques based on entropy, capable of choosing the most reliable from a set of different algorithms capable of solving some specific problem. However, the most reliable algorithm may have a non feasible computational cost, in terms of the time it takes to complete, the amount of memory it uses or the number of resources (e.g. processors) required.
The coherent definition of reliability and complexity introduced in the previous section allows the definition of a cost function combining the two, assuming that each algorithm is designed to meet a set of specifications:

$$J = 1 - R + \rho C \tag{5}$$

where R is the reliability, C the cost and ρ a normalizing factor such that $\rho C \in [0, 1]$. In general, ρ will be such that the cost does not overwhelm the reliability when searching for the optimal action. Examples of ρ are $\rho = \frac{1}{\max_{a \in A} C(a)}$ or $\rho = \frac{1}{\sum_{a \in A} C(a)}$, where A is the set of algorithms capable of solving a problem.
Equation (5) applies to all levels of the HGDIM, i.e., the performance of an algorithm, primitive event or task can evaluated by (5) if the cost and reliability are appropriately propagated bottom-up through the hierarchy.
A task t is composed by several events $e_k \in E^t$, where E^t is the set of events composing task t, occurring in sequence or in parallel. For each event e_k there exist a set of alternative algorithms A^k capable of solving the problem represented by the event.
The propagation equations are:
Cost of event $e_k \in E^t$ is the minimum cost among all algorithms translating the event:

$$C(e_k) \triangleq \min_{a \in A^k} \{C(a)\} \tag{6}$$

Action probability p_a **of algorithm** $a \in A^k$ is the current probability of a being applied. A probability density function is defined over the discrete algorithm space A^k. Its purpose, not discussed here, is to help a learning algorithm converging to the algorithm which minimizes the cost function J.
Reliability of event e_k is the average reliability among all algorithms translating the event:

$$R(e_k) \triangleq \sum_{a \in A^k} p_a r_a \tag{7}$$

where r_a is the reliability of algorithm a.
The **cost of parallel execution of events** e_1, e_2 is

$$C(e_1 // e_2) \triangleq \max_{e_1, e_2 \in E^t} \{C(e_1), C(e_2)\} \tag{8}$$

while the **cost of** n **events** $e_1, \ldots, e_n \in E^t$ **executed in series** is

$$C(e_1 | \ldots | e_n) \triangleq \frac{1}{n} \sum_{i=1}^{n} C(e_i) \tag{9}$$

The mean in equation (9) intends to keep the cost in the interval $[0, 1]$.
The successive application of these rules leads to the **cost of a task**, $C(t)$.
The parallel execution of events is not logically parallel from the reliability point of view. In fact, **all** events must be successful to complete a task. Hence, the **reliability of task** t is (Lima, 1994)

$$R(t) \triangleq \prod_{e_k \in E^t} R(e_k) \tag{10}$$

4. APPLICATION TO INTELLIGENT ROBOTIC SYSTEMS

Tasks implemented by Intelligent Robotic Systems may generally be decomposed on primitive events, such as **Move Robot, Locate Object, Plan Path, Grasp Object**. In this section, the paradigm just formulated will be used to derive the relation between cost and reliability of algorithms capable of solving the problems corresponding to two of these primitive events. Emphasis was put on cost measures other than execution or computing time, to enhance the flexibility of the definition.

4.1. *Motion Control*

The dynamics of an n-degree of freedom robot manipulator can be expressed by the following compact form of Euler-Lagrange's equations of motion:

$$D(\underline{\theta})\ddot{\underline{\theta}} + \underline{NL}(\underline{\theta}, \dot{\underline{\theta}}) = \underline{u} \tag{11}$$

where $\underline{\theta} \in \Re^n$ is the joint angles vector, $\underline{u} \in \Re^n$ is the control torques vector, $D(\underline{\theta}) : \Re^n \rightarrow \Re^{n\text{X}n}$ is the inertia matrix, and $\underline{NL}(\underline{\theta}, \dot{\underline{\theta}}) : \Re^n \text{x} \Re^n \rightarrow \Re^n$ is the vector representing nonlinear coupling of Coriolis, centrifugal, gravity and friction torques. Luo and Saridis (1985) formulated the optimal control solution for the problem of making the manipulator track a desired trajectory. They identified the system state with $\underline{x}(t) = (\underline{\theta}(t)\ \dot{\underline{\theta}}(t))^T$ and suggested the performance index

$$\begin{aligned} J(\underline{u}) \quad = \quad & \frac{1}{2}\underline{e}^T(t_f) G \underline{e}(t_f) + \\ & \frac{1}{2}\int_{t_0}^{t_f} [\underline{e}^T(t) Q \underline{e}(t) + \dot{\underline{e}}^T(t) S \dot{\underline{e}}(t)] dt \end{aligned} \tag{12}$$

where $S = \begin{bmatrix} 0 & 0 \\ 0 & S_0 \end{bmatrix}$, G is a $2n$x$2n$, S_0 an nxn real symmetric, positive definite matrix, Q is a real nonnegative $2n$x$2n$ matrix, $\underline{e}(t) = \underline{x}_d(t) - \underline{x}(t)$ and $\underline{x}_d(t) = (\underline{\theta}_d(t)\dot{\underline{\theta}}_d(t))^T$ is the desired state vector. When $t_f \rightarrow \infty$, the control law reduces to

$$\begin{aligned} \underline{u}^* &= D(\underline{\theta})\{\ddot{\underline{\theta}}_d(t) + K_p[\underline{\theta}_d(t) - \underline{\theta}(t)] + \\ & K_v[\dot{\underline{\theta}}_d(t) - \dot{\underline{\theta}}(t)]\} + \underline{NL}(\underline{\theta}, \dot{\underline{\theta}}) \end{aligned} \quad (13)$$

which has the same form of the *Computed Torque Method*, with $K_p = S_0^{-1} P_{12}$ and $K_v = S_0^{-1} P_{22}$. $P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix}$ is the solution of a continuous algebraic Ricatti equation.

Given the optimal control law, and if the sampling period is T_s, the discretized closed loop state space model comes $\underline{x}((k+1)T_s) = A_{dcl}\underline{x}(kT_s) + B_{dcl}\underline{u}_d(kT_s)$ where $\underline{u}_d = (\underline{\theta}_d^T \ \dot{\underline{\theta}}_d^T \ \ddot{\underline{\theta}}_d^T)^T$.

In this development it was assumed:

1. Perfect cancellation of the non-linear terms;
2. Non-noisy measurements;
3. Complete information about the state.

However, assumption 3 may be kept while relaxing assumptions 1 and 2, by modeling the resultant perturbations as zero mean gaussian noise. A new discrete state model will be obtained: $\underline{x}((k+1)T_s) = A_{dcl}\underline{x}(kT_s) + B_{dcl}\underline{u}_d(kT_s) + D\underline{v}(kT_s)$, where $\underline{v}$ is a gaussian noise vector with $E[\underline{v}(kT_s)] = 0$, $E[\underline{v}(kT_s)\underline{v}(kT_s)^T] = C_v$.

The performance index has to be modified when the noise is actually added to the open loop system, and it becomes $I(\underline{u}) = E[J(\underline{u})]$. For this motion control problem (event **move robot**) the algorithms cost is identified with the optimal value of I:

$$C = I(\underline{u}^*) = \underline{e}(0)^T P \underline{e}(0) + \sum_{k=1}^{N} \mathrm{tr}(PDC_vD^T) \quad (14)$$

where P is the solution of a discrete algebraic Ricatti equation (Lewis, 1986), and N the number of samples in the trajectory.

A lower bound for the Reliability can be obtained based on a method described by McInroy and Saridis (1994), when the specifications are quadratic in the tracking error $\underline{e}(kT_s)$:

$$\underline{e}(kT_s)^T Q_s \underline{e}(kT_s) \leq \epsilon, \ k = 1, \ldots, N, \ Q_s \geq 0 \quad (15)$$

where Q_s is a matrix weighting the error components. If

$$C_e^{-1}(kT_s) - Q_s(kT_s) \geq 0, \forall k = 1, \ldots, N \quad (16)$$

then $R \geq [\chi_d^2(\epsilon)]^N$, where χ_d^2 is a chi-square distribution with d degrees of freedom, $C_e(kT_s)$ is the covariance of the tracking error, N the number of points the specifications are concerned with, and d the dimension of the state vector ($d = 2n$ for a n-degree of freedom manipulator). $C_e(kT_s)$ can be determined by solving the difference equation $C_e((k+1)T_s) = A_{dcl}C_e(kT_s)A_{dcl}^T + DC_v(kT_s)D^T$.

Given Q_s and ϵ, the reliability lower bound is given by $[\chi_d^2(\epsilon)]^N$ for all different C_e which satisfy (16). The value of C_e depends on A_{dcl} which in turn is a function of the weighting matrices Q, S, G in the performance index. Hence, for different lower bound reliabilities, different Costs C will be obtained, and the performance function $J = 1 - R + \rho C$ is used to decide among different optimal algorithms resulting from different choices of Q, S, G.

In order to clarify the application of the formalism, the following definitions for this particular example should be useful:

- problem element $\underline{f} = (\underline{x} \ \underline{x}_d)$
- problem solution $\underline{S}(\underline{f}) = \underline{x}_d$
- solution approximation $\underline{U}(\underline{f}, \phi) = \underline{x}$, as obtained by algorithm ϕ
- algorithm $\phi = \phi(Q, S, G) = \underline{u}^*(Q, S, G)$

The performance function associated to the algorithms balances the penalization of error and cost of control (by penalizing joint accelerations) to track a given trajectory (joint positions, velocities and accelerations) and the reduction of uncertainty due to measurement noise.

4.2. *Image Processing*

The use of stereo vision algorithms to determine the pose (3D position + orientation) of an object in a workspace is usually prone to errors due to the camera calibration process, spot noise superimposed on pixel brightness, and pixel resolution. In the sequel, the cost-reliability analysis of a 2D object location problem using two image processing algorithms is detailed. The brightness is corrupted by superimposed spot noise only, whose influence is reduced by averaging several image frames corresponding to the same scene.

Given a rectangle inside an M x M pixels image, the problem is to estimate the position of the rectangle in the image (see Fig. 1), that is, its central pixel of coordinates (x_c, y_c). The pixels inside the rectangle are initially set to 1, while the outside pixels are set to 0. Zero mean gaussian noise is added to the initial value of each pixel in the whole image.

Several assumptions are made with the goal of simplifying the mathematical analysis and the simulation:

- The area of the rectangle is known and equal to $A = (y_e - y_b)(x_e - x_b)$;
- The whole rectangle is inside the boundaries of the image;
- No other objects are present in the image;
- Errors resulting from poor image resolution or computational roundoff are not considered, that is, without noise any algorithm should be able to determine the rectangle position with infinite accuracy.

Two algorithms are proposed to solve the problem:

The **open loop algorithm (ol)** determines the *center of gravity* of the total image, using the equations

$$\begin{aligned} \hat{x_c} &= \frac{\sum_{i=1}^{M}\sum_{j=1}^{M} jb(i,j)}{A} \\ \hat{y_c} &= \frac{\sum_{i=1}^{M}\sum_{j=1}^{M} ib(i,j)}{A} \end{aligned} \quad (17)$$

where $b(i,j)$ is the brightness of pixel (i,j).

The **closed loop algorithm (cl)** correlates the image (*feedback*) with a pattern rectangle (*reference*) equal in size to the original noise-free rectangle of the image and with the same orientation. The rectangle is as-

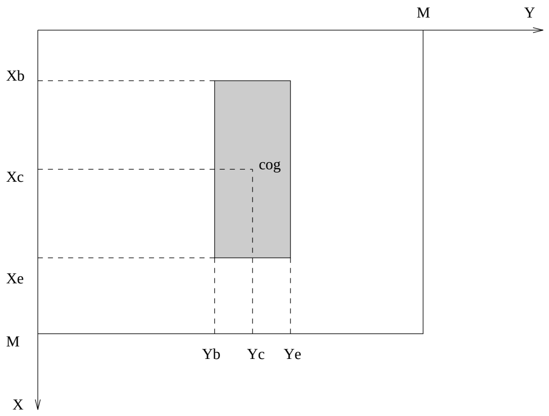

Fig. 1. Image processed by the two algorithms

sumed to be centered inside the P x P pixels pattern image. The coordinates (x_c, y_c) of the pixel with the greatest correlation coefficient are the estimates of the rectangle position in the image.
Both algorithms manipulate images resulting from the average of several image frames, in order to reduce noise.

Problem formulation. F is the set of M x M images containing rectangles with size $(x_e - x_b)$ by $(y_e - y_b)$. This includes images resulting of averaging several image frames.
The goal is to compute an ϵ-approximation $\underline{U}(f, \phi)$ of $S : F \rightarrow \Re^2$, where $\underline{S}(f) = (x_c \; y_c)^T$ and $\underline{U}(f, \phi) = (\hat{x}_c \; \hat{y}_c)^T$.
To simplify the analysis, and since what happens in one of the directions is similar to what happens in the other, only the error estimate $|\hat{x}_c - x_c|$ of x_c is checked.

Information. The information operations $L_{ij}(f)$, $L_{ij} : F \rightarrow \Re$ give the results

$$L_{ij}(f) = \begin{cases} 1 + n_{ij} & \text{if pixel}(f,i,j) \in r \\ n_{ij} & \text{if pixel}(f,i,j) \notin r \end{cases}$$

where r denotes the set of pixels inside the rectangle. Hence, $\underline{\mathcal{I}} = (L_{111}(f), \ldots, L_{kij}(f), \ldots, L_{NMM}(f))^T$ where k denotes the kth frame, N is the total number of averaged frames, M the number of pixels on each side of the image and n_{ij} is a random variable representing the noise at pixel (i,j), $n_{ij} \sim \mathcal{N}(0, {\sigma_b}^2)$, i.i.d.

Model of computation. In this simple approach, the set of algorithms is restricted to those that can solve the problem in, say polynomial time. This means that more concern is put on the cost of getting information. For example, if a mobile robot has to stop and get several frames of a scene in order to speedup posterior computations of its locations, a slower algorithm that requires less stopping time for the robot is better. The algorithm may run while the robot is performing other tasks. Sequential computation is also assumed.
Hence, $\text{cost}(\phi) = \text{cost}(\underline{\mathcal{I}}, f) = cN_{min}$, that is, cost is proportional to the minimum number of averages needed by algorithm ϕ to get the error below ϵ. Notice that here f represents an image resulting from the average of N_{min} image frames.

Study of the algorithms. The **open loop** algorithm estimates the center coordinates of the rectangle using equations (17). Since the brightness of each pixel is a gaussian distributed random variable

$$p_{B_{ij}}(b_{ij}) \sim \begin{cases} \mathcal{N}(1, \sigma_b^2) & \text{pixel}(f,i,j) \in r \\ \mathcal{N}(0, \sigma_b^2) & \text{pixel}(f,i,j) \notin r \end{cases}$$

and the B_{ij}'s are uncorrelated from pixel to pixel (given that they are independent), it can be deduced that, after N averages of distinct frames of the same image and if assuming independent noise from frame to frame, $p_{\hat{X}_c}(\hat{x}_c) \sim \mathcal{N}(\mu_{\hat{X}_c}, \sigma^2_{\hat{X}_c})$ with $\mu_{\hat{X}_c} = x_c$ and

$$\sigma^2_{\hat{X}_c} = \frac{M^2(M+1)(2M+1)\sigma_b^2}{6A^2N} \tag{18}$$

Now, given an accuracy ϵ and a desired reliability R_d, N is determined such that $\Pr\{|\hat{x}_c - x_c| \leq \epsilon\} \geq R_d$
From the table of standard normal $\eta(R_d)$ in $\Pr\{\frac{|\hat{x}_c - x_c|}{\sigma_{\hat{X}_c}} \leq \eta\} = R_d$ is obtained.
Equating $\eta(R_d) = \epsilon/\sigma_{\hat{X}_c}$ and using (18): $N \geq N_{min} = \frac{\eta^2(R_d)M^2(M+1)(2M+1)\sigma_b^2}{6A^2\epsilon^2}$. The inequality comes from the fact that the reliability is to be lower bounded by R_d.
The **closed loop** algorithm looks for the pixel where the noisy output of the correlator achieves a maximum when the pattern is displaced around the image. Due to noise, there is some probability that the wrong pixel is chosen. In order to make the problem tractable, some assumptions are made, such as working at 1D again, and considering errors of 1 pixel displacement at most.
If the correlator input $u_{corr}(x) = r(x) + n(x)$, where $r(x)$ is a rectangle of length $x_e - x_b$ and $n(x)$ is gaussian noise of zero mean and variance σ_b^2/N, and the impulse response of the correlator is $r(x_c - x)$, then $y_{corr}(x) = y_r(x) + y_n(x)$ is the correlator output, where $y_r(x)$ is an isosceles triangle of length $2(x_e - x_b)$, centered at x_c and of height $x_e - x_b$, and $y_n(x)$ is gaussian noise with mean x and variance $\sigma_n^2 = \sigma_b^2(x_e - x_b)/N$.
Now, assuming errors of 1 pixel at most and that 1 pixel corresponds to an ϵ displacement, the reliability comes $R_d = \Pr\{|\hat{x}_c - x_c| < \epsilon\} = \Pr\{(y_{corr}(x_c + \epsilon) - y_{corr}(x_c) < 0) \wedge (y_{corr}(x_c - \epsilon) - y_{corr}(x_c) < 0)\}$. It is further assumed that the output noise of the correlator is independent from pixel to pixel. This is not actually true, but since the two random variables are correlated, under this assumption a smaller probability will be obtained, hence obtaining an upper bound for N_{min}. Noticing that the sum of two random variables jointly and marginally gaussian is another gaussian distributed random variable, $R_d = \Pr^2\{z < 0\}$, $z \sim \mathcal{N}(-\epsilon, 2\sigma_n^2)$, or $\sqrt{R_d} = \Pr\{\frac{z+\epsilon}{\sqrt{2}\sigma_n} < \eta\}$, where $\eta(R_d)$ can be read from a table of standard normal and is made equal to $\frac{\epsilon}{\sqrt{2}\sigma_n}$. Hence, the upper bound for the minimum number of averaged frames is

$$N_{minup} = \frac{2\eta^2(R_d)\sigma_b^2(x_e - x_b)}{\epsilon^2} \tag{19}$$

Comparing N_{minup} and N_{min} for the open loop algo-

Table 1 **ol** algorithm with $R_d = 90$ %.

σ_b	0.1		0.3	
ϵ	N_{min}	R	N_{min}	R
0.1	73	0.8733	655	0.8933
0.2	18	0.8333	164	0.8533

Table 2 **ol** algorithm with $\epsilon = 0.3$ pixel.

σ_b	0.1		0.3	
R_d	N_{min}	R	N_{min}	R
90 %	8	0.9267	73	0.8733
95 %	11	0.9400	103	0.9467

rithm, it may be noticed that the closed loop upper bound on the number of averages does not depend on the size of the image M, while the minimum number of averages for the open loop algorithm increases with M, thus it is possible, with a reasonable ratio of image size to pattern size, to show that the closed loop algorithm upper bound N_{minup} will be below the actual value of N_{min} for the open loop case. Simulations show that (19) is a loose upper-bound and that in practice the cost of the closed loop algorithm is much smaller, for the same reliability.

Simulation results. The **open loop** algorithm was tested with different sets of parameters as follows:

- $R_d = 90\%$ and $\epsilon = 0.1$ and 0.2.
- $\epsilon = 0.3$ and $R_d = 90\%$ and 95%.

Each of the setups was tested with standard deviation of pixel noise $\sigma_b = 0.1$ and $\sigma_b = 0.3$. Each side of the image had 32 pixels and the rectangle had 13 pixels in the x direction, 9 in the y direction.
The results are presented in Tables 1 and 2, showing for each setup the reliability obtained after 150 runs with a number of averages slightly greater than the minimum theoretically required. In general, the outcomes agree quite well with the expected results.
The **closed loop** algorithm was simulated under the same setup. The standard deviation of the superimposed noise needed to obtain reliabilities close to those of the open loop case with the same number of averages was determined by trial and error. Table 3 shows these results. ϵ was made equal to 1 pixel, because the closed loop algorithm can not achieve sub-pixel resolution. The same N_{min} is sufficient to obtain accurate estimates with the same reliability, but with a noise standard deviation approximately 10 times larger.
Both results show that, given a desired accuracy ϵ and different environmental conditions (symbolized by different pixel noise variances), the cost of the object location algorithms increases with increasing demand on the reliability.

Table 3 **ol** and **cl** algorithms compared.

N_{min}	R_{ol}/σ_b	R_{cl}/σ_b
4	98%/0.2	99%/2.0
9	97%/0.3	98%/3.0

Constraining the number of averages to some value, the open loop algorithm can only attain the reliability of the closed loop algorithm under a much more favorable environment. Hence, for the same cost and under the same environment, reliability would distinguish the two. Also, if different algorithms of both types, distinguished by the choice of different N at design time, were available, a combination of the cost and reliability associated to each of them would help in the selection of the most reliable algorithm among those constrained by some cost. If cost of processing information were considered, an algorithm compromising the speed of the **open loop** algorithm and the reliability of the **closed loop algorithm** would minimize the cost function J.

5. CONCLUSIONS

In this paper, the formalism of Information-Based Theory of Complexity was used to state a coherent definition of Cost and Reliability for the different algorithms composing a feasible set for a problem. Two examples of application to subtasks of an Intelligent Robotic System were presented.
The combined measure of reliability and cost presented in this paper may be used in the off-line selection of plans, if viewed as an extension of the work by McInroy and Saridis (1994). It may also be used to build a cost function which is updated recursively on-line and used to learn the action that minimizes the cost function, as described by Lima (1994). In the last case, the cost is determined off-line and the reliability estimated from experience. No assumptions about the distribution of the specification error are required. Adaptation to changes in the environment is possible in some situations. In both cases, the cost function introduced here can be used as a tool to design Intelligent Machines and as a measure of their performance.

6. REFERENCES

Lewis, F. L. (1986). *Optimal Estimation.* John Wiley and Sons.

Lima, P. U. (1994). Intelligent Machines as Hierarchical Stochastic Automata. PhD thesis. Rensselaer Polytechnic Institute. Troy, NY 12180-3590.

Lima, P. U. and G. N. Saridis (1993). Measuring Complexity of Intelligent Machines. In: *Proceedings of 1993 IEEE Int. Conf. Robotics and Automat.*

Luo, G. L. and G. N. Saridis (1985). Optimal/PID formulation for control of robotic manipulators. *IEEE Journal of Robotics and Automation* **1**(3).

McInroy, J. and G. Saridis (1994). Techniques for selecting pose algorithms. *to be published in Automatica* .

Musto, J. and G. N. Saridis (1993). An entropy-based reliability assessment technique for intelligent machines. In: *Proceedings of 8th International Symposium on Intelligent Control.*

Technical Committee on Intelligent Control (1994). Report of task force on Intelligent Control, IEEE Control Systems Society. *IEEE Control Systems Magazine* **14**(3). P. Antsaklis, editor.

Traub, J., G. Wasilkowsky and H. Woźniakowsky (1988). *Information-Based Complexity.* Academic Press, Inc.

Valavanis, K. P. and G. N. Saridis (1992). *Intelligent Robotic Systems.* Kluwier Publishers.