CONF-9004125--1

~~CESAR 90/02~~

AN APPROACH TO ELEMENTAL TASK LEARNING*

Philippe Belmans
Center for Engineering Systems Advanced Research
Oak Ridge National Laboratory
P.O. Box 2008
Building 6025, MS-6364
Oak Ridge, TN 37831-6364

CONF-9004125--1

DE90 006614

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Submitted to: SPIE's Applications of Artificial Intelligence VIII, Orlando, Florida,
April 17-19, 1990

* Research sponsored by the Engineering Research Program of the Office of Basic Energy Sciences, of the U.S. Department of Energy, under contract No. DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

An approach to elemental task learning

Philippe Belmans

Center for Engineering Systems Advanced Research
Oak Ridge National Laboratory
P.O. Box 2008
Building 6025, MS-6364
Oak Ridge, TN 37831-6364

ABSTRACT

In this article we deal with the automated learning of tasks by a robotic system through observation of a human operator. Particularly, we explain what is meant by a learning ability in autonomous robots and in teleoperation systems, where several operators and several machines may work in cooperation to perform tasks. We discuss different approaches to learning in these systems and outline the features of the models they are based upon. This leads us to choose an analytical model suited for tasks analysis. We then present the software architecture for our proposed approach and show the first results obtained on sample tests.

1. INTRODUCTION

Today, a great interest is granted to learning abilities for machines. Within the context of robotics, the main purpose of learning is for the robot to add new objects and new operations in its representation of the environment to achieve an increase of its autonomous potential. Thus, a learning ability for a robot implies the capability of "understanding" the behavior of its environment. This can be done in two ways: under the supervision of a teacher, or in an autonomous way. In this last case, the robot performs operations on its environment and observes the modifications to achieve understanding. This process is called Discovery.

2. LEARNING IN A ROBOTIC SYSTEM

Learning is a feature applying to autonomous robots as well as teleoperation systems. In the first case, when the robot is unable to deal with a given situation because it occupies a non expected state according to the specification of its program, human intervention is required to allow the system recovery. As the human is solving the problem, the machine may analyze what is being done and extend its knowledge and its abilities to solve further similar situations. This point can also be adopted in teleoperation. During an application, some tasks presenting a certain degree of repetition and of determinism can be delegated to the machine, thus allowing the operator to focus on the management of unknown events.

With classical programming, task specification takes place in a different set of concepts and operations (computer language) than the one (environment) where the task has to be performed. On the contrary, learning should allow the programming of a machine with concepts that are found in its working environment, without using some ad-hoc language. Moreover, it is not required for the teacher to know the internal structure of the machine, its procedures, and specific programming languages. For instance, the learning of patterns with a neural net consists of the presentation of objects to the machine under the supervision of a teacher that indicates if recognition was correct or not. However, this supposes that the machine has models and procedures for learning on which knowledge acquisition rests. The model of the concepts the machine can handle cannot be dissociated from the control software architecture. In other

words, learning is a functionality that closely relates to the control architecture, in particular, the software architecture, i.e., the definition of entities, the nature of their interactions and the protocols that support them. If we except the hardware aspects (processors, peripheral units and instrumentation), architecture is a keypoint because it determines the processing abilities of the machine (autonomy, abstraction, rapidity, safety), and the type of functionalities that can be learned automatically by the machine.

In a further step, when the machine has acquired an accurate representation of the task, it can monitor in a discrete way the human operation taking place in a man-machine cooperation; e.g. if the operator does not perform the elemental task accurately, the machine can superimpose a correction when the severity of error is not too important.

3. MODELS FOR TASK LEARNING

What characterizes the different approaches is basically the way they model the task to be learned. To us, modelling the human operation assumes complete answering of the following questions:

- **WHAT?** This question deals with the classes and the characteristics of the model that are the most suitable to describe the task being performed.
- **HOW?** This question involves resources required for the operation.
- **WHEN?** This question deals with the significant events that must occur to activate such a task.
- **WHY?** This question considers the facts that led to the performance of this operation.

To illustrate this, let us consider a drilling operation. The constraints such as blocking the rotations of the drill and bounding the penetration speed are characteristics of the model of the task (WHAT). The use of a drilling machine, speed monitoring by the operator under force feedback and rotations blocked by the machine are concerned with the way the task is performed (HOW). Checking that the piece to drill is well tightened and that the drilling is ready constitute the set of preconditions to start the task (WHEN). Finally, this operation may take place in an assembly sequence and a hole appears to be missing, therefore it is necessary to drill it (WHY).

Typically, the first two questions, are context free; that is the robot does not have to analyze the contextual background to "understand" these aspects of the task. Learning the answers to these questions leads to a characterization of the tasks and is more related to the low level of the control.

On the contrary, the two latter questions are closely linked to the environment, since a part of history or the whole history of the operation has to be taken into account. Learning, in these cases, leads to an analysis of the planning of the tasks and is more related to the high level of the control.

To answer these questions, the process of learning requires a model. For instance, let us consider the case of HERMIES-IIB embedding the ability of discovery.¹ The robot is left in an environment of which the characteristics are unknown. The first task is to locate a control panel in this environment and to get close to it. Then by turning on the switches and the potentiometers, the robot has to reset the panel in a nominal state. The robot performs a succession of trials; after some trials, it can recognize an already known type of panel and thus executes the action to reset the panel. Otherwise, it performs other trials and inserts into the knowledge base a new type of panel, when the panel has been successfully reset. Of course, there exists a learning model

assuming that (the process connected to) the panel has a deterministic behavior and reversible evolutions. In other words, the state occupied by the system (the panel) only depends on its previous state and the inputs, and it is possible for all states to return back to the previous state by a simple transition. However, if a panel does not present these properties, it will be impossible for the robot to handle it, except if the robot embeds the ability to process the type of panel type as symbol using an explicit meta-model for classification. Such systems use rule-driven models. It is also possible to consider transition models such as state machines. Because this form of learning is related to planning concerns, these models are not adequate to characterize a task. For stationary phases of a task, such models give a condensed representation, as the task is described by a single state and no event occurs. However, if we consider transient phases, for instance, in the insertion of a peg in a hole, the major challenge is to determine which are the relevant events and the relevant states of the task. So, a-priori, each force sensor reading should be considered as an event and each motion of the arm should be considered as an action, as there is no explicit relationship between events and actions. Such a model thus leads to oversized representation of the task which is difficult to handle.

Our point is that logical models are more adequate to solve the two last questions (WHEN and WHY) but not the first two ones (WHAT and HOW). Such models (basically transition models) are best suited for describing context-dependent processes. Moreover, when executed in autonomous mode, the so modelled tasks may not achieve the goal (e.g. insertion) because they may lead to unstable control.

A second class of models includes analog models. In this class, we include Neural Nets. As far as topology is concerned, a neural net is a layered arrangement of cells connected with one another with synaptic links. This arrangement contains one input layer collecting data, one or several intermediate layers and one output layer that delivers the results. Each cell is an elementary processor which generates an output function of a linear combination of its inputs weighted by synaptic coefficients. In this type of system, the processing power is rather due to the complexity of the arrangement of processors than on their individual capabilities. Two phases can be distinguished in a neural net application. The preliminary phase is learning, which in fact consists of the tuning of the synaptic coefficients. To achieve this, data are fed in the net, and the corresponding output are used to compute a correction signal that is retropropagated in the net to modify the synaptic coefficients. However, the learning phase of a neural net usually requires a great amount of iterations, because the speed of convergence of synaptic coefficients is very low. Moreover, theoretical tools are still not available that allow one to determine the optimal structure of a net for a given application, and to analyze the behavior of the controlled system, and more particularly its stability. Neural nets are mostly used for pattern recognition, for instance with camera images or tactile images issued from an artificial skin in the gripper of the robot.

Another approach, which is under development at the CESAR Lab, deals with Random Set Theory. Like for neural nets, the processed information is elemental (usually binary patterns). To simplify, let us say that, in this approach, invariant properties between binary patterns are analyzed to gather them in sets. These sets are dynamically created during analysis each time a new property appears. Such a method ignores the a-priori relationships that may exist between inputs, and therefore, there is a combinatorial explosion when the number of object characteristics grows. The main limitations of this approach are due to memory and mass-processing of data.

Nevertheless, generality of use is an advantage of these models. Here, we investigate the use of a less general model for which, however, implications on the control are well known. In the next section an approach used to specify control of manipulators called task-function approach is recalled and the model it offers for task analysis is presented.

4. PRINCIPLE OF THE TASK FUNCTION APPROACH

The approach we investigate is based on the concept of task function.^{2,3} Let us briefly recall what this concept is. A task function is basically an error function $e(t)$ between the actual state of the robot $\hat{s}(t)$ and the reference state or reference trajectory $s_r(t)$ it should occupy. The error function can be a vector function to be zeroed as in path tracking:

$$e(t) = \hat{s}(t) - s_r(t) \forall t \in [0, T];$$

It also can be the norm of a vector function:

$$e(t) = \|\hat{s}(t)\| - \sigma(t) \forall t \in [0, T];$$

For instance, in obstacle avoidance the distance from an obstacle must not be less than a threshold. It can also be the combination of both expressed in a Lagrangian expression. Having expressed the task function, the main task of the robot is to zero that error function. In a learning context, our idea is to model the human elemental task as the minimization of a given task function for a given reference trajectory. If we assume that the task is fulfilled, the analysis of the sensor data will allow one to characterize the reference trajectory and, therefore, to learn the task. As an example (Fig. 1), if we consider the insertion of the peg in the hole, there exists a transient phase, followed by the insertion itself. For that phase, the difference between the sensor information issued from a force sensor and the reference interaction screw is theoretically zero. So the task will be characterized by this reference interaction screw. As this task function approach allows specification of almost all kinds of robotic tasks, we think it offers a good pattern for analysis and recognition of tasks taught by a human operator in a man-machine cooperation scenario.

Fig. 1. Example of task function.

5. PROPERTIES AND CHARACTERISTICS OF THE MODEL

The key property of this model is linearity. This means that if we consider a set of elemental causes or inputs, the action or output resulting from the whole contribution is identical to the sum of the outputs corresponding to all elemental inputs. More particularly, this means that to realize a task, it is possible to decompose it into simpler elemental concurrent tasks and to compute the control for each of them individually. The sum of these elementary controls will ensure the global behavior of the system (Fig. 2).

Fig. 2. Linear property.

This property is typically used for arm control with proximetric sensors⁴ or in mobile robots using non-hierarchical architectures.⁵ The consequence is that a task description takes place in a linear space. A generator system of this space has to be determined in order to specify any task. For the learning process, this means that the identification process can be structured in parallel filters which gives the projection of the task performed by the human operator on this generator system. This architecture of the learning process concretizes this property, by allowing parallel analysis of the different sensor outputs.

Combining the raw sensory concepts into more abstracted ones allows more abstracted processing to occur. The more sensing possibilities, the more learning possibilities. In other words, if a robot does not embed a given sensory ability, it is impossible for it to handle any concept related to it. Understanding the task means determining how it affects the environment in which it is performed. In other words, a task is only defined by the states it dictates to the environment. In robotics, the environment is described by sensor information. Typically, these are three-dimensional vectors:

- articular information issued from internal sensor: positions $q(t)$ and velocities $\frac{\partial q}{\partial t}$,
- cartesian position computed from articular information: position $X(t)$ velocity $\frac{\partial X}{\partial t}$ and acceleration $\frac{\partial^2 X}{\partial t^2}$, and
- external sensor information: position $Z(t)$ velocity $V(t)$ force $F(t)$.

In order to learn the task being performed, the program has to monitor the sensor outputs and analyze the invariance relationship verified by them, using predefined patterns to recognize the characteristics of the reference path of the task. As the operator thinks in cartesian space, this relationship will occur only in cartesian space and not in articular space. For this reason, all sensor information will be considered as three-dimensional vectors.

6. IDENTIFICATION MODELS

At this point, we assume that at least four relationships (or models) exist that can be verified by sensor data:

- quadratic relationship,
- scalar relationship,
- linear relationship,
- accumulation relationship; or
- else the relationship is undetermined.

7. QUADRATIC RELATIONSHIP OR Q-MODEL

Let us consider E^3 the set of three-dimensional measurement vectors issued from a sensor which is a subset of R^3 , a so called quadratic relationship is verified if and only if an invariance relationship can be found on the euclidian norm of vectors derived from sensor information:

$$(\exists \Omega \in R^3) \wedge (\exists \rho \in R) / (\forall M \in E^3) \|\Omega \vec{M}\| = \rho;$$

Some interpretations of such a relationship are given in Fig. 3, according to the type of information and the type of sensor it is issued from. Mainly, this relationship characterizes tasks such as constant curvilinear velocity or acceleration motion, obstacle avoidance or target tracking tasks (constant distance from the effector to an object). In a cartesian space of position, such a relationship exists when the effector is proceeding on a sphere of radius ρ and of center Ω . When the task is an arc welding operation, the operator moves the torch with a constant curvilinear velocity (this is one of the conditions related to the quality of sanding). Therefore, in a learning process, such a constraint should be evidenced by Q-filter processing cartesian velocity. When the task is a sanding operation, the operator moves the tool exerting a constant force in intensity (this is one of the conditions related to the quality of welding). The Q model should fit when applied to information issued from a force sensor. In this example, the task is performed with obstacle avoidance, when the effector is in the neighborhood of the forbidden zone. The analysis of proximity sensor information should reveal that the relative distance from the effector to the obstacle is constant, or that the relative velocity is null.

Fig. 3. Example of tasks obeying Q-model.

7.1. Scalar relationship or S-model

Considering E^3 the set of three-dimensional data issued from sensors, there exists a scalar relationship if and only if the scalar product of these vectors with a constant vector is constant, i.e.:

$$(\exists \vec{u} \in R^3) \wedge (\exists h \in R) / (\forall \vec{M} \in E^3) \vec{OM} \cdot \vec{u} = h;$$

Some interpretations of this relationship are given in Fig. 4. Typically, such a relationship expresses task constraints, such as motion taking place in a plane of perpendicular $\vec{u}$, e.g. a horizontal plane ($z = h$). In the case of a planar milling operation, the tool must stay in a plane (e.g. horizontal plane of altitude K). It is the part of the S-filter to detect such correlations between cartesian positions computed from encoders. Considering the same task, the analysis by an S-filter will show that the component of the cartesian velocity is null on the Z axis. In fact, this relationship and the previous one are equivalent, since the velocity space is always tangent to the position space. In that case, the analysis gives redundant results. If we assume that a force sensor is mounted on the tool, the analysis of data according to an S-model should show that the force exerted on the tool can be represented by a vector belonging to the horizontal plane. This is a complementary characterization of the task.

Fig. 4. Example of tasks obeying S-model.

7.2. Linear relationship or L-model

In this case, there exists a colinearity relationship between the measurement vectors issued from the sensor:

$$(\exists \vec{u} \in R^3) \wedge (\exists \Omega \in R^3) \wedge (\exists \lambda \in R) / (\forall \vec{M} \in E^3) \Omega \vec{M} = \lambda \vec{u};$$

Fig. 5. Example of tasks obeying L-model.

In the cartesian space of position or velocity, this is the characteristic of a motion along a line. In the transient phase of a drilling task, this relation is true, as the action of the drill is exerted parallel to the penetration. In the case of a drilling task, the processing of position, velocity and force with a L-filter should reveal that each of them is parallel to the Z axis.

7.3. Accumulation relationship or A-model

This relationship is true when a subspace of sensor output E^3 becomes invariant in time:

$$(\exists \Omega \in R^3) / E^3 = \{\Omega\};$$

Fig. 6. Example of tasks obeying A-model.

This is the typical case of a stopping point, when velocity and rotation are null, i.e. when the position and the orientation of the effector remain constant. If we consider a pick and place task, there are goal points indicating where to pick up objects and where to place them. An A-filter will detect such points when the sequence of position and orientation is constant. In an insertion task, the force exerted on the effector by the hole is null along the X and the Y axis, while the torque is null along Z . These characteristics of the task should be evidenced by the corresponding subset of force sensor data information. Moreover, the A-filter should detect the convergence of (X, Y) to (X_r, Y_r) . In a welding operation, usually, the operator maintains the torch with a constant orientation relative to the local curvilinear point. Processing induction sensor information with an A-filter would give constant orientations relative to the joint frame or null rotation during the task performance. A similar property would be found on the orientations when analyzing cartesian information issued from internal sensors in the case of a drilling task.

7.4. Undetermined relationship

This relationship is the default one, when none of the preceding relationships are true. If such a relationship occurs on a set of data, either no significant constraint exists on the task, or a constraint exists that cannot be identified by the system. This typically happens when the human operator uses a kind of perception that is not artificially available to the robot. This is also a problem that may appear if the filters are not well tuned and thus identification is not robust.

8. SOFTWARE ARCHITECTURE

We will deal with the physical implementation of this approach, which is a typical architecture problem. Without pre-supposing of a given architecture, it is possible to distinguish two kind of processing. The numerical processing that we are dealing with and the symbolic processing we have to take into account so as to provide results that can be properly exploited. More particularly the numerical (or signal) processing in charge of model identification is the first step of the whole process of task analysis and includes at least three stages:

- **Prediction:** for each sample vector issued from a sensor, we consider a future tendency according to a model by considering the following occurring vectors according to a fixed size window. For this processing, we use a least-square method for parameters identification.
- **Discrimination:** for a given vector, some predictions are likely to occur, while some are not because they present too much noise; this discarding process is called discrimination. It compares the noise of model fitting to a given threshold.
- **Compression:** this process gathers vectors that verifies the same invariant (i.e. having the same prediction) into the same set called segment.

9. STRUCTURE OF THE IDENTIFICATION PROCESS

Fig. 7. Architecture of the system.

The stages of identification are performed by parallel computations for all the sensor data. Particularly, a general frame of computation can be duplicated for each of the components to extract parameters of different models. These outputs are called segments. The symbolic stage, with which we will deal in the future, deals with fusion and interpretation. The role of fusion is

to link among them the different analyses performed for each sensor. The interpretation searches for the invariances among the different symbolic representations obtained by the identification process for several performances of the task. The output of these processes are called tokens. So far, fusion has not yet been implemented.

We have tested different configurations of the identification system. The first configuration was a single algorithm that processed the fitting of the different models in parallel. However, this system revealed to be too complex to tune and improve because the parallel processing hinders a clear analysis of interaction between the different algorithms of the program. This led us to consider an updated multipass version, easier to tune and to debug. The system now includes seven passes.

- **Pass-1** deals with prediction on a fixed size window of samples.
- **Pass-2** is the compression of sensor data verifying the A-relationship.
- **Pass-f** is the discrimination filter that discards too noisy fittings.
- **Pass-3** is the compression for L-model.
- **Pass-4** is the compression for vectors verifying a Q and a S model simultaneously. This is due to singularities of the model.
- **Pass-5** is the compression of Q model.
- **Pass-6** is the compression of S model.

So far we have tested this system for simulated tasks. The following table (Fig. 8) illustrates a task with noiseless positions that were fed into the system and Fig. 9 represents what was recognized by the system.

for point 1	to point 50	point (0,0,0);
for point 51	to point 250	line of vector (1,0,1);
for point 251	to point 290	point (200,0,200);
for point 291	to point 400	line of vector (-1,1,0);
for point 401	to point 450	point (90,110,200);
for point 451	to point 650	line of vector (0,-1,-1);
for point 651	to point 710	point (90,-90,0);
for point 711	to point 910	plane $z=10$; $F(x,y)=0$;
for point 911	to point 1010	point (293.4,106.7,0);

Fig. 8. Example of tasks functions.

Fig. 9. Recognition by the system.

10. DISCUSSION

The results are satisfactory. From a qualitative point of view, it is possible to reconstruct the original simulated trajectory. However, there are some quantitative differences. For each identification, ten points are missing at the end of each series, which is actually due to the sampling window size. The system cuts segments up to three points apart from where they were originally, which represents a quite negligible distance. The noise level is very low (no more than 0.45), which is due to segmentation and the error of identification is practically null, as a simple truncation of the parameters give the original ones. For the sample task which contains several segments, the recognition is correct. Therefore, we consider that the system performs a good recognition for noiseless trajectory. In an on-going activity, we are considering noise input to the system and are also studying the behavior of the system with a real human operator.

For the future, we plan to implement this system in a teleoperated arm, first to test the different algorithms and more particularly to analyze their behavior towards a real human noise. Secondly, we want to design a fusion process to obtain tokens that model the multi-sensor tasks and implement task functions in the arm controller to fully validate our approach.

11. REFERENCES

1. P. F. Spelt, G. de Saussure, E. Lyness, F. G. Pin, and C. R. Weisbin, "Learning by an Autonomous Robot at a Process Control Panel," *IEEE Expert*, 1989 Winter Edition.
2. R. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation*, RA-2, No. 1, April 1986.
3. B. Espiau, M. Le Borgne, C. Samson, "Robot Control," Oxford Press, to be published.
4. C. Samson, "Une Approche pour la Synthèse et l'Analyse de la Commande des Robots Manipulateurs Rigides," Internal Report I.R.I.S.A. n.356, Rennes, France, April 1987.
5. C. Samson, "Commande de Robots Manipulateurs Rigides - Techniques de la Robotique - Architectures et Commande," Editions HERMES, Paris, France, 1988.

Received by GSTI

FEB 07 1990