Using Fuzzy Logic for Bounded Recovery of Autonomous Agents

Janos L. Grantner
Western Michigan University
Department of Electrical and Computer Engineering
Kalamazoo, MI 49008-5066, USA

George Fodor
ABB Industrial Systems AB, ISY/AMC
S-721 67 Vasteras, Sweden,
and
Western Michigan University
Department of Electrical and Computer Engineering
Kalamazoo, MI 49008-5066, USA

Dimiter Driankov
University of Linköping
Department of Information and Computer Science
S-581 83 Linköping, Sweden

Abstract[1]

The solution to the problem of application-independent fault recovery of autonomous agents requires a specification method for the agent's capacity to act outside of its normal operational limits. This paper presents a recovery method based upon the theory of a fuzzy finite state machine. A fuzzy specification is given for the bounds within which an autonomous agent is capable to recover after an unexpected situation has occurred in its environment. It has been shown that the three main components of the recovery problem: (i) fault detection, (ii) fault recovery, and (iii) the properties of the actuator / sensor gear of an autonomous agent are interrelated. The suggested method can be implemented either by an application-independent software algorithm, or by fuzzy logic hardware.

[1]The research is supported by the Michigan Space Grant Consortium (Grant #: 9633401), Western Michigan University, ABB Industrial Systems, and the Swedish Research Council for Engineering Sciences (Grant #: 271/96-134).

1. Introduction

Many unattended applications in the areas of robotics, marine vessels, and avionics are autonomous systems. They are required to act consistently relative to the initial goals when they encounter unexpected situations in their environment. Such systems comprise of a reactive level that interacts directly with their environment through sensors and actuators, and which can perform control tasks such as positioning, or sequencing [1].

In the case of an autonomous agent, a supervisory level of the system can detect when the reactive level cannot cope with, or does not intend to proceed with the current control task, and by using additional stored knowledge, will perform a recovery operation, or a re-planning of the reactive control level. The capacity of a system to recover after a failure is recognized in industry as a very important property. However, a recovery mechanism will be of interest only if it is made application-independent, (that is, it can be implemented, say, in the operating system of a Programmable Logic Controller rather than being re-designed with each application). That means the recovery mechanism cannot be based on a

model of the environment. Moreover, that means the error detection level is required to be application-independent as well.

The results reported here provide a solution to the problem of application-independent recovery. It is based upon modeling the bounds of the reactive capacity of an autonomous agent at the time instance when the agent detects an unexpected change, or failure. The approach exploits the correlation that exists between the theories of the Fuzzy-State-Fuzzy-Output Finite State Machine (FSFO FSM), [6], and Ontological Control [3].

When an unexpected change occurs in the environment, the discrete states involved in the erroneous situation are isolated by the Ontological Controller (OC). The OC then invokes the fuzzy specification of the discrete states involved in the erroneous situation. That includes a set of fuzzy states of the FSFO FSM, and a state transient trigger algorithm. The OC uses the fuzzy state transient specification to determine if a recovery is possible, that is, the FSFO FSM is the specification of the bounds within which the recovery is possible. If the FSFO FSM enters a suitable fuzzy state, it will return the particular control action that achieves the recovery.

The following two sections present the main theoretical tools used, and the method is then illustrated by an example.

2. FSFO FSM Model

The model of the FSFO FSM that is implemented by a Boolean automata based on two-valued logic is given by the formulas (1), where X and Z stand for a finite set of fuzzy inputs and outputs, respectively, R* is a composite linguistic model (3), and ∘ is the operator of composition. Each state of the Boolean FSM is characterized by an overall linguistic model R_s, or a set of linguistic sub-models with multiple-input-single-output (MISO), and multiple-input-multiple-output (MIMO) systems:

$$\begin{aligned} Z &= X \circ R^* \\ R^* &= G(R_S) \\ z_c &= DF(Z) \\ X_B &= B(X) \\ Y &= f_y(X_B, y) \end{aligned} \qquad (1)$$

A fuzzy state is defined by a crisp (Boolean) state and a state membership function

$$S_{F_k} : S_k, g_{S_k} \qquad (2)$$

where S_{F_k} stands for fuzzy state k, S_k represents crisp state k, and g_{S_k} is the state membership function associated with S_k. G stands for the matrix of state membership functions, X_B, Y, and y are two-valued Boolean input and state variables, respectively. B stands for a Fuzzy-to-Boolean transformation algorithm to map a change in the status of a fuzzy variable into state changes of a finite set of Boolean variables. The z_c crisp values of the fuzzy outputs are obtained by evaluating a defuzzification algorithm, DF. On the basis of the concept of a fuzzy state, the FSM stays in a number of crisp states simultaneously, to a certain degree in each. One of these states is referred to as a dominant state for which the state membership function is a 1 (full membership). The concept of the fuzzy FSM based on a Boolean FSM and the State Membership Functions, in this context, have been introduced in [6] and [8], respectively. A formal representation was given in [10].

For each fuzzy state of the FSFO FSM model, a R_i^* composite linguistic model is created from the finite set of R_{S_i} overall linguistic models (i=1,..,p). Let the FSFO FSM be in fuzzy state S_{F_k}, then

$$\begin{aligned} R_k^* = \max[&\min(\beta_1^k, R_{S_1}), \min(\beta_2^k, R_{S_2}), \ldots, \\ &\ldots, \min(\beta_k^k, R_{S_k}), \ldots, \min(\beta_p^k, R_{S_p})] \end{aligned} \qquad (3)$$

where $\beta_1^k, \beta_2^k, \ldots, \beta_p^k$ stand for the degrees of state membership function g_{S_k}, and $R_{S_1}, R_{S_2}, \ldots, R_{S_p}$ are the overall rules in crisp states $S_1, S_2, \ldots, S_p$, respectively. With (3), a single-input-single-output (SISO) system is assumed. In adaptive systems R_k^* is not stored in memory, it is dynamically created by computing (3), instead. By modifying the β degrees of the state membership functions on-line, new R^* composite linguistic models can be created under real-time conditions. The switch over between composite linguistic models is determined by the state transients of the FSFO FSM.

The state transients of the FSFO FSM are specified by means of a sequence of changes at the fuzzy inputs and outputs. Those changes are mapped into the corresponding sequence of changes of the Boolean input and output variable sets, respectively, using the B algorithm. The Boolean input/output sequence specification is passed to an algorithm, and the Boolean FSM will then be automatically synthesized [2], [9]. A fuzzy logic controller hardware accelerator of pipeline architecture has been presented in [7].

The theory of the FSFO FSM will be used in the sequel to model the changes in the control algorithm of a low level reactive system recovering from a violation of the ontological assumptions (VOA).

3. Ontological Desynchronization

When a control system such as an autonomous agent is designed, the modeling assumptions for its control algorithm are inherently extended by additional assumptions about the complex environment in which the control takes place. These assumptions are not represented by formal means, hence, an agent cannot verify whether they are true. Ontological control investigates the case when these assumptions are violated, situations in which a controller acts under violations of the ontological assumptions.

The architecture of an Ontological Controller (OC) capable to detect violations of ontological assumptions (VOA) in the context of a desynchronization from the execution of a goal path was shown in [5]. However, the OC is unable to recover from a VOA by itself [5]. The concept of a state is defined in ontological control by (4):

$$S_i = (y_i, u_{i,j}) \ (i,j=1,...,n) \qquad (4)$$

where y_i stands for a Boolean formula (referred to as a "plant formula") showing what condition is true at a given time in the controlled plant. Each relevant plant situation has a corresponding plant formula in some state. A control action denoted as $u_{i,j}$ is executed when y_i is true. The expected outcome of this action is that the plant changes such that at the next time instance y_j will be true. However, if some external action (disturbance) occurs, the expected change in the plant does not take place but a new plant state, y_k, will materialize instead. The disturbance is considered as an external action and, if known in advance, is denoted as $u_{i,k}^{ext}$ where the indices point to the respective two plant formulas before and after the external action. The new state can be denoted as some $S_k = (y_k, u_{k,l})$. The control will then proceed in a succession of states. The states that can materialize from an arbitrary state S_i by external actions (disturbances) are referred to as "collateral states to S_i" and the set of such states is denoted as $K_{\perp}(S_i)$. An example is given below to illustrate the problem of the ontological desynchronization.

A position controller has the goal to move a radar antenna into a desired position 'p'. The controller has two inputs: (1) the angle 'x' of the radar antenna given by a mechanical sensor and (2) the angle 'y' of the radar antenna given by an electronic sensor. The measurement y is much more accurate than x, however the range ε_2 of y is much smaller than the range ε_1 of x. Thus the positioning algorithm has two steps: first a regulator R_1 brings the radar in the range ε_2 of the electronic sensor, then a regulator R_2 brings the radar in the desired position p. The position error is ε=q-p, where q can be either x or y, depending on the current range. By using (4), state S_1 can be given as S_1=(y_1=$|\varepsilon<\varepsilon_1|$, $u_{1,2}$='R_1 acting'). The other states are shown in Fig. 1. A disturbance can bring the radar outside range ε_1, when a state transition from S_1 to S_6 occurs, i.e. $K_{\perp}(S_1)=\{S_6\}$, or a disturbance outside ε_2 makes the transition from S_3 to S_7, i.e. $K_{\perp}(S_3)=\{S_7\}$. A violation of the ontological assumptions (VOA) occurs at state S_j = (y_j, $u_{j,k}$) if $u_{i,j}$ is executed in state S_i, but the expected y_j does not materialize in the plant, although no external action has occurred. It has been shown in [5] that a VOA manifests always as a state transition from S_i to a state in $K_{\perp}(S_j)$ where S_j is the consecutive (next expected) state to S_i. This type of transition is referred to as an ontological desynchronization.

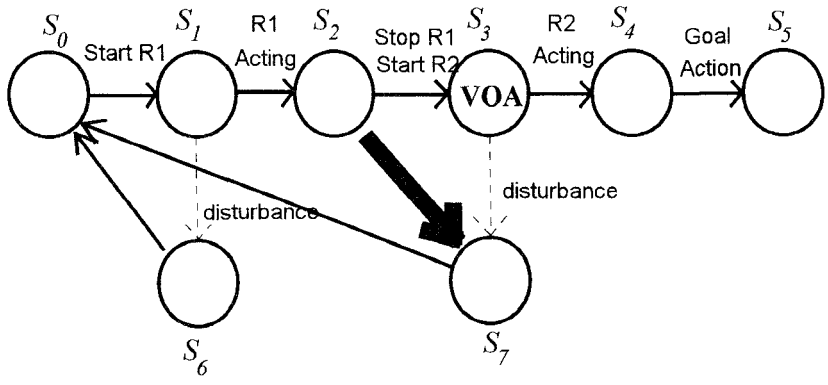

Figure 1: The state set of a position controller.

It has also been shown in [5] that for certain state sets, a VOA has the effect such that the controller enters a cycle. This cycle repeats indefinitely such that the ontological desynchronization will repeat as well.

For the system above, let us consider the following ontological violation: the mechanical part has developed an error, in fact, an angle offset between the two sensors such that when x=r then y=r+offset. This kind of error is a violation of the initial assumption that the two readings are always aligned when the operating ranges of the two sensors overlap. No measurements or other means are provided to verify this assumption. This error has the effect that after the plant formula for state S_2 materializes, (i.e., $|x-p|<\varepsilon_1$), state S_3 will not materialize since now $|y+offset-p|>\varepsilon_1$ and thus the next state will be S_7 instead. However, no disturbance has occurred that would correspond to the transition to S_7. The transition from S_2 to $S_7=K\_(S_3)$ is the ontological de-synchronization. After S_7, state S_0 will materialize next, then S_1, and then S_2. At S_2, the relation $|y+offset-p|>\varepsilon_1$ is still satisfied so the controller will jump again to S_7 and the cycle repeats as predicted by the theory.

Clearly, for the position controller, the situation appears as 'normal" control. The recovery from this situation requires a supervisor level which devises a state that corresponds to the current plant situation, and which has a control action that can "break" the cycle. For fuzzy controllers, there can be found recovery solutions by adding extra rules to the rule base [4]. However, for controllers such as PLC's that have discrete states, the problem of recovery becomes more difficult since there is no state with acceptable properties in the state space of the controller that can materialize at a VOA. Thus the recovery algorithm suggested in this paper relies on techniques that can accommodate fuzzy states and yet produce Boolean outputs.

4. Recovery Using FSFO FSM

For the sake of clarity, the use of the FSFO FSM will be outlined for a single state transition at a VOA. At design time, each plant formula is associated with linguistic intervals. After an ontological desynchronization is detected, the reactive controller gives to the FSFO FSM the following information:

(i) the relevant fuzzy input X to the FSFO FSM is the fuzzified plant formulas of y_i, (the last state materialized as expected), y_j, (the state with the VOA), and the fuzzified plant formulas of the states in $K\_(S_j)$,

(ii) the fuzzy output set Z consists of the fuzzified control actions of the expected next state, and in addition, the fuzzified plant formulas of the collateral states of the expected next state,

(iii) the state membership function degrees are set such that the expected next state (dominant state) is assigned the degree of 1,

(iv) the actual mapping scheme between fuzzy and Boolean subintervals of the B algorithm in order to devise the next state of the FSFO FSM.

Due to a VOA, a fuzzified plant formula *in-between* the expected y_j (leading to a state of the FSFO FSM that corresponds to state S_3) and the fuzzified plant formulas of the states in $K\_(S_j)$ will

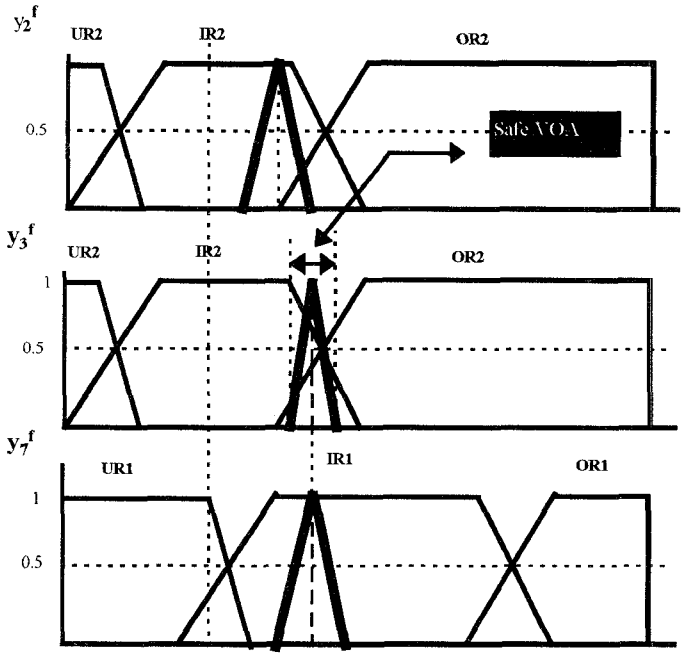

Figure 2. Fuzzified plant formulas

be received. Fuzzified plant formulas are shown in Fig. 2 for a case when recovery from a VOA is possible. On the horizontal axis, the position error is given, the linguistic labels refer to under-range, in-range, and over-range for Regulators R_1, and R_2, respectively. The membership functions for the fuzzified input readings reflect the noise and measurement imprecision with respect to the mechanical and electronic sensor, respectively. Recovery from the VOA can be achieved by turning on Regulator R_2, since the position error is still within its operational limits.

There are two main approaches to tackle this problem by using the FSFO FSM model. One of them is to deal with it as a trigger condition for the next fuzzy state of the FSFO FSM. The conditions for state transients are specified in terms of membership functions. If the received fuzzified plant formula is *not too far* from the one for y_j (that is, despite the offset the radar is still in the work range of regulator R_2) then the FSFO FSM will enter to a state corresponding to S_3 and the control action inferred from the model assigned to this state will cause the system to recover from the VOA. The FSFO FSM will move to a state corresponding to one of the states in $K\_(S_j)$, otherwise. That means the controller cannot recover from the VOA, hence, it should be stopped. The solution to the recovery in the state transition domain is based upon the actual mapping scheme specified for the B algorithm between the fuzzy and Boolean subintervals along the horizontal axis. In the example at hand, the boundaries can be specified such that the FSFO FSM enters a state in which Regulator R_2 will be turned on as long as the support of the fuzzified readings of the electronic sensor are included in the support of the IR2 label.

With the other approach, the FSFO FSM will enter to a state other than the one corresponding to S_3. The expected dominant model has a less than 1 degree in that state. The Ontological Controller has detected the ontological desynchronization, and it will attempt to dynamically tune the β degrees of state membership functions in order to find such control action that will result in a recovery from the VOA. In this case, the solution is sought primarily in the domain of the composite linguistic model.

The intervals used for the plant formulas of the state sets of the PLC's are, in fact, Boolean. Fuzzification makes the boundaries between states

continuous such that a state preserves its properties beyond its Boolean limits. Thus fuzzification "balances" the fuzzy boundaries of the states against the degree of the ontological violation and, if it is possible, helps to select a control action to recover from the VOA.

5. Conclusions

Current research has shown that by using a fuzzy state machine together with an Ontological Controller in the frame of a suitable architecture, the error detection and recovery of a system can be substantially improved. The novelty being introduced here is the combination of the two theories, and the use of fuzzy set theory as a specification tool in the context of recovery. The FSFO FSM hardware accelerator can be implemented by using reconfigurable Field Programmable Gate Arrays, hence, it can be added to high-performance systems working under real-time conditions.

References

[1] P.J. Antsaklis, K.M. Passino, An Introduction to Intelligent and Autonomous Control, Kluwer Academic Publishers, Boston / Dordrecht / London. 1994.

[2] P. Arato, Specification and realization of logic control procedures on the basis of prescribed input-output changes, Per. Pol. El. Eng., Budapest, Hungary, Vol. 31, No. 3-4, 59-153. (1987).

[3] D. Driankov, G. Fodor, Ontological Real-Time Control, Proc. and Plenary Talk, First European Congress on Fuzzy and Intelligent Technologies, Aachen, Germany, Sept. 1993.

[4] D. Driankov, G. Fodor, Fuzzy control under violations of ontological assumptions, invited plenary talk, *FLAMOC'96 Proceedings*, Sydney, Australia, January 15-18, 109-115. (1996).

[5] G. Fodor, Ontological Control: Description, Identification and Recovery from Problematic Control Situations, PhD Thesis, Dept. of Computer Science, University of Linkoeping, Sweden (1995).

[6] J. Granter, M. Patyra, VLSI implementation of fuzzy logic finite state machines, *IFSA '93 World Congress, Proceedings*, Seoul, Korea, July 4-9, Vol. II, 781-784 (1993).

[7] J. Grantner, M. Patyra, Synthesis and analysis of fuzzy logic finite state machine models, WCCI'94, Proceedings of the FUZZ-IEEE'94, Orlando, FL, June 26-29, Vol. I, 205-210, (1994).

[8] J. Grantner, M. Patyra, M. Stachowicz, Intelligent fuzzy controller for event-driven real-time systems and its VLSI implementation, in the book *Fuzzy Control Systems* (Eds. A. Kandel, G. Langholz), CRC Press, Boca Raton, FL, USA, 161-179, (1994).

[9] M. Klein, Automatisierung des Schaltwerkentwurfs auf der Basis einer zustandsfreien Spezification", MS Thesis, University of Karlsruhe, Germany, (1993).

[10] J. Grantner, Design of Event-Driven Real-Time Linguistic Models Based on Fuzzy Logic Finite State Machines for High-Speed Intelligent Fuzzy Logic Controllers, Thesis for the Degree Candidate of Technical Science, Hungarian Academy of Sciences, Hungary (1994).