

Design of a Reconfigurable State Transition Algorithm for Fuzzy Automata

Janos L. Grantner, Paolo A. Tamayo, Ramakrishna
Gottipati

*Department of Electrical and Computer Engineering
Western Michigan University
Kalamazoo MI 49008-5329, USA*

janos.grantner@wmich.edu, p3tamayo@wmich.edu,
r0gottip@wmich.edu

George A. Fodor

*ABB Automation Technology Products
Vasteras, Sweden*

george.a.fodor@se.abb.com

Abstract—This paper presents a reconfigurable state transition architecture for the Hybrid Fuzzy-Boolean Finite State Machine (HFB FSM). The architecture is created using parameterized components to add versatility with respect to the number of fuzzy inputs and the number of states. This is a very attractive property to implement virtual fuzzy automata for supervisory controllers of complex systems where relevant state clusters vary a great degree. The reconfigurable architecture allows the supervisory controller to reset the fuzzy automaton in order to model a particular state cluster, as needed. The design was done using VHDL.

I. INTRODUCTION

One of the current trends in computing is the emergence of reconfigurable hardware. If reconfiguration can be performed rapidly without much affect on the operation of the system, the hardware will be able to support functions that would require many times over the size of the actual physical implementation.

Typical large-scale, complex industrial control systems have a two-level hierarchical architecture: a plant control level and a supervisory level. At the supervisory level, monitoring continuously the global state space at once while moving along a desired goal path is impractical. As it was shown in [4], if the current goal state is known along with all possible disturbances, a small section of the total state space is only relevant to make a decision on the next state of the goal path. It is like moving a small window over a large state transition graph. This approach makes a reconfigurable controller to model the ever-changing state graph configurations an attractive scheme. In order to meet real-time requirements a hardware accelerator may be needed. This paper describes the design of a reconfigurable state transition algorithm for the Hybrid Fuzzy-Boolean Finite State Machine (HFB FSM) [1].

The rest of the paper is organized as follows: as background information, first a short description of the theory of Ontological Control along with the HFB FSM for fault detection and identification is given. Section IV describes the architecture of the reconfigurable state transition algorithm and provides for details of the implementation. Section V reports on the testing and validation procedures of the design. It also discusses the limitations of the current approach and

the assumptions made. Conclusions and an outline of future plans to improve the architecture are given in Section VI.

II. BACKGROUND

Ontological Control (a contemporary supervisory control approach) concerns about automated fault detection and identification of some particular type of faults referred to as ontological de-synchronization, and the recovery from those faults in large, complex industrial control systems. It has been proposed in previous papers that the Hybrid Fuzzy-Boolean Finite State Machine (HFB FSM) can be used to address that problem [1] [2].

Ontological de-synchronization may occur at particular junctions of the goal path where the state graph exhibits certain properties. The general idea is that the Ontological Controller (OC) uses the HFB FSM to model the relevant section of the control algorithm along with plant data to make a decision on the next state of the goal path. In order to do that it needs to configure the HFB FSM for each of those particular state clusters of the control algorithm and then pass the actual plant data to the HFB FSM. If there is a fault, the HFB FSM will detect it through a state transition and advise the OC whether a recovery is possible from the fault. Since the configurations of the critical state clusters with respect to the ontological de-synchronization fault keep changing along the goal path, the OC needs to reconfigure the HFB-FSM for every state cluster of concern. In this paper we report on a significant step, i.e., the design of a reconfigurable state transition algorithm, of our research to develop a hardware accelerator for a reconfigurable HFB-FSM [3]. The design was done using VHDL and the code was mapped to a Field Programmable Gate Array (FPGA) chip.

The conditions for a fuzzy state transition can be given in the terms of fuzzy inputs (i.e., analog inputs), two-valued (digital inputs) and analog inputs with threshold (i.e., essentially, two-valued inputs). The states of the fuzzy inputs are converted to the states of a set of two-valued (Boolean) variables using the Fuzzy-to-Boolean (B) algorithm [1]. The actual mapping of any fuzzy input value to a unique set of Boolean variables is fundamental to determine the next state of the HFB FSM. Since each new instance of the HFB FSM model (as the ontological control process moves along the goal path) requires a new mapping scheme, a reconfigurable implementation of the B algorithm is a critical contribution to the design of a hardware accelerator for the HFB FSM

In the B algorithm, the Mean of Maxima (MOM) defuzzification method along with a set of non-overlapping Boolean sub-intervals covering the Universal Space for the fuzzy variable is used for the Fuzzy-to-Boolean mapping. These sub-intervals play a major role in determining the state transitions. By introducing parameters to define the Boolean sub-interval without changing the Universal Space the state transition conditions can be reconfigured.

III. HFB FSM MODEL

The model of HFB FSM was introduced in [2] and was extended in [1]. It is implemented by a Boolean automaton based upon two valued logic. The HFB FSM is defined by the formulas below.

$$Z_F = X_F \circ R^* \quad (1.1)$$

$$R^* = G(R_S) \quad (1.2)$$

$$Z_C = DF(Z_F) \quad (1.3)$$

$$U_B = f_u(X_B, W_B, X_T, y_B) \quad (1.4)$$

$$X_B = B(X_F) \quad (2.1)$$

$$Z_B = B(Z_F) \quad (2.2)$$

$$Y_B = f_u(X_B, W_B, Z_B, y_B, X_T,) \quad (2.3)$$

X_F , W_B and X_A stand for fuzzy, two-valued (Boolean), and analog inputs with associated threshold values, respectively. X_T is the result of the comparison of the analog inputs with the associated threshold values. Z_F , Z_C and U_B stand for fuzzy, defuzzified fuzzy, and two-valued (Boolean) outputs, respectively. R^* is the composite linguistic model (4), and \circ is the operator of composition. A fuzzy state is made up of a set of crisp states, the HFB FSM stays simultaneously in all of them, to a certain degree in each. There is a dominant crisp state in this state set for which the degree of the state membership function is 1. Each crisp state of the HFB FSM is characterized by an overall linguistic model R_S , or by a set of linguistic sub-models in the case of multiple-input-single-output (MISO), and multiple-input-multiple-output (MIMO) systems [1].

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

$$S_{Fk} : S_k, g_{Sk} \quad (3)$$

where S_{Fk} stands for fuzzy state k , S_k represents crisp state k and g_{Sk} is the state membership function associated with S_k . In (1.2), G stands for the matrix of state membership functions.

For each fuzzy state of the HFB FSM model, a R^* composite linguistic model is created from the finite set of R_{S_i} overall linguistic models ($i = 1, \dots, p$). Let the HFB FSM be in a fuzzy state R_{Fk} , then

$$R^*_k = \max[\min(\beta_1^k, R_{S1}), \dots, \min(\beta_p^k, R_{Sp})] \quad (4)$$

where β_1^k β_p^k stand for the degrees of state membership function g_{Sk} and R_{S1}, \dots, R_{Sp} are the overall linguistic models in crisp states S_1, \dots, S_p , respectively. By modifying the β degrees of the state membership function on-line, new R^* composite

linguistic models can be created under real time conditions. X_B , Z_B , Y_B and y_B stand for two-valued Boolean input, Boolean output and next state and present state of the variables, respectively. B stands for the Fuzzy-to-Boolean transformation algorithm to map a change in the status of fuzzy variable into state changes of a finite set of corresponding Boolean variables. The Z_C crisp values of the fuzzy outputs are obtained by evaluating a defuzzification strategy, DF .

IV. RECONFIGURABLE ARCHITECTURE DESIGN

The architecture of the proposed reconfigurable logic is shown in Figure 1. The design is based upon the computational model described in Section III. A multi-fuzzy input model is assumed in which the host system provides for the fuzzy representation of the analog inputs. The current design deals with fuzzy inputs only, the digital inputs and the analog inputs with thresholds are omitted at this point. The generation of the sets of corresponding Boolean variables for the fuzzy inputs is carried out using the B algorithm (2.1). The next state of the HFB FSM is devised using the obtained Boolean variables and the present state according to (2.1) and (2.3).

Each fuzzy input is treated as a vector because it consists of an array of fuzzified values. The number of elements for each vector is dependent upon the universal state space chosen for the system. It is assumed that a fuzzified input vector is placed onto the data bus every clock cycle. A start signal generated externally marks the beginning of sampling. The circuit counts the received elements of the fuzzy input vectors and compares the number with a pre-determined value (obtained from the most recent reconfiguration). After the expected number of samples has been obtained, the MOM value of the fuzzy input is calculated and passed onto the Interval Detection circuit. The Interval Detection circuit makes use of the defuzzified fuzzy input to determine the Boolean sub-interval location to be set. Using the interval locations obtained from the Interval Detection circuit and the present state the next state is determined from a lookup table circuit.

A. MOM Calculation

The calculation of the MOM value is done incrementally while fuzzy inputs are sampled. Figure 2 shows the algorithm where the number of samples is equal to the number of points in the universal set of discourse. The algorithm simply tests each sample for a maximum value and records this value. Each time the maximum value is received at particular input, a counter is incremented. Also the corresponding sample count is added to a running sum. When a new maximum value is sampled, this new value will be recorded as the new maximum number and the running sum will be reset back to the corresponding sample count of the new value.

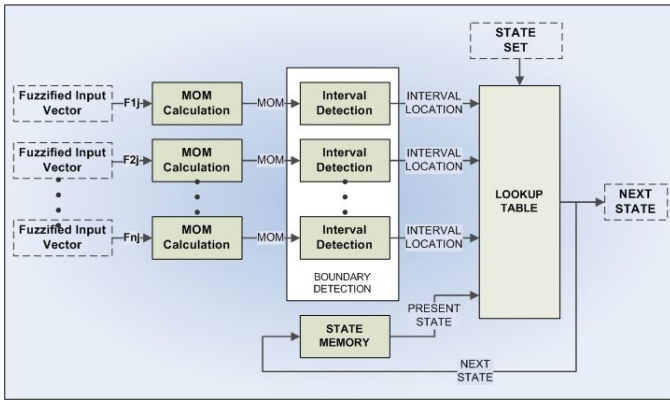


Fig. 1. Reconfigurable logic architecture.

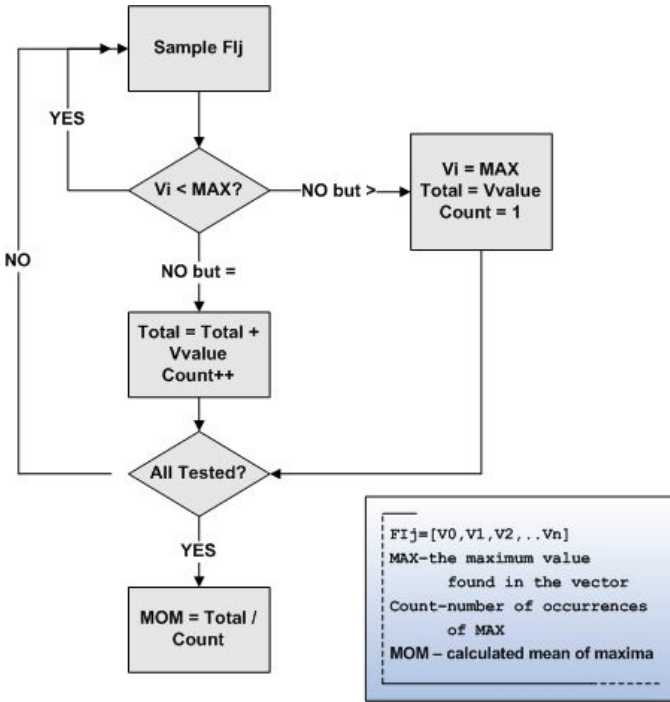


Fig. 2. Mean of Maxima calculation algorithm.

For example if the current maximum value is 0.7 and a new value of 0.8 is detected at sample number 10 of the universal set space of 25 elements, the new value will replace the old one and the running sum will be set to 10. If 0.8 is found again at sample number 20 the running sum will be updated to $10+20=30$. The final value of the counter that counts the number of occurrence of the maximum value will be used as the divisor for the running sum of the added sample counts. For the example of maximum value 0.8 the divisor will be 2 since it was detected only at sample numbers 10 and 20. Therefore, the computed MOM value will be 15. Since each fuzzy input is assumed to be available every clock, the clock signal should be generated in phase with the available data samples.

B. Interval Detection

The interval detection logic is implemented using cascaded comparators that compare the MOM value with the Boolean sub-interval limits. The Boolean sub-interval limits are pre-determined. Those values are dependent upon the choice of the state transition conditions as outlined in Section II. For example, if the boundary limits are modified in such a way that the widths of the sub-intervals are widened or narrowed, the state transition can be modified. To illustrate this feature, let us consider the boundary limits of 0-5 and 6-10, respectively, for a universal space with two Boolean subintervals. Let us further assume that a state transition from the present state to state x is launched if the MOM value of a fuzzy input falls in sub-interval 0-5, and the MOM evaluation produces a value of 4. However, we can effectively change the behavior of the system by changing the boundaries of the first Boolean sub-interval to 0-3. Hence, the state transition from the present state to state x will be modified to a transition to, say, state y.

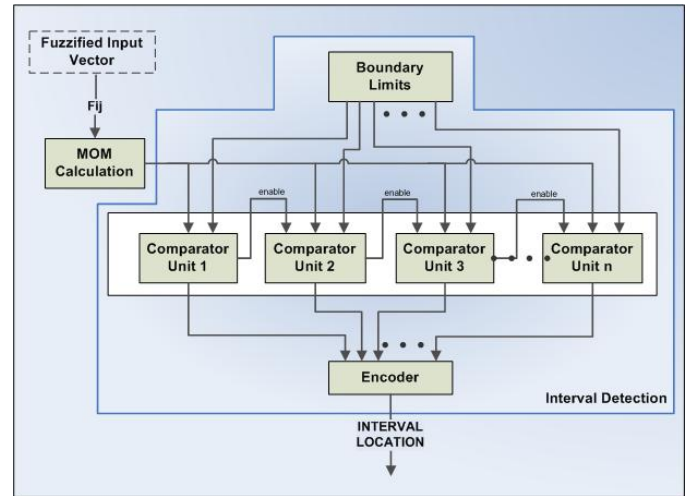


Fig. 3. Interval location detection logic.

Each comparator compares its maximum limit with the MOM value. The next comparator is enabled when the MOM value is equal, or above its maximum. A value below the maximum limit does not enable the next comparator. The enable signals are then encoded to determine the sub-interval where the MOM value falls. For example, if there are five comparators and the concatenated enable signals yields a value of "11000" in binary, then the encoded Interval Location value will be 2. The range of values of the interval location is from 0 to the number of boundaries - 1. The number of boundaries determines the number of comparators implemented to carry out the interval detection. Figure 3 shows the structure of the detection logic.

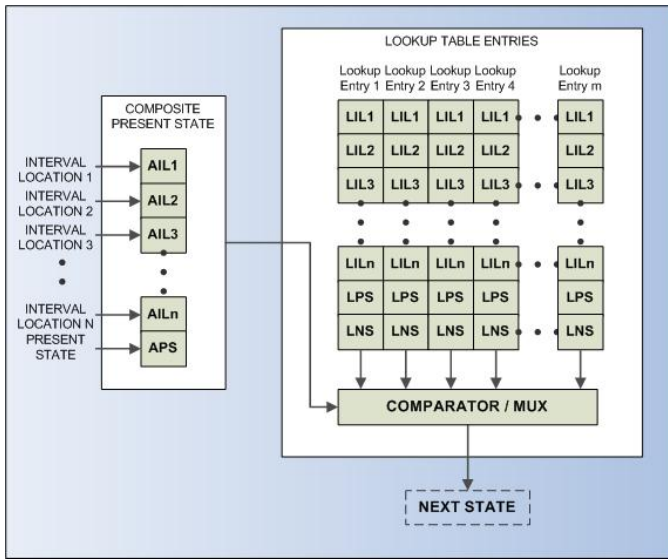


Fig. 4 Lookup Table implementation

C. Look-up Table

The calculations of the MOM values for each fuzzy input and of the interval detections are performed in parallel. The resulting interval locations are concatenated along with the present state to form a composite input vector to the system. The composite input vector is represented as follows: [AIL1:AIL2:...:AILn:PS] where AIL_{*i*} (*i*=1,..., *n*) stands for the *i*th actual interval location and PS stands for the present state of the HFB FSM. The new composite input vector is compared with all entries of a look-up table. The look-up table is a concise representation of the behavior of the HFB-FSM with respect to the defined state transitions through the dominant crisp states. The look-up table entries are given in the following vector format: [LIL1:LIL2:...:LILn:PS:NS] where LIL_{*i*} stands for the *i*th look-up interval location and PS and NS stand for the present and next states, respectively. If the composite input vector matches an entry in the look-up table then the designated next state becomes the present state. If no match is found the present state is retained. This scheme is illustrated in Figure 4. All entries in the look-up table are initialized each time the hardware is reconfigured for a new instance of the HFB-FSM.

D. Reconfigurable Components

The main objective of the reconfiguration is to provide the system the ability to adapt to the varying clusters of states along the goal path and to model the plant. The circuit that carries out the comparison of the present composite input vector with the entries of a reconfigurable look-up table accomplishes this objective. The proposed architecture introduces parameterized components into the design that add versatility to the implementation. The architecture can be implemented as a module in a system that performs the major functions of the HFB FSM.

The parameterized components of the proposed architecture are summarized in Table I. A typical system has got multiple inputs. The number of fuzzy inputs will differ from system to system. The width of the discrete

representation of the degree of membership of each element of the universe of discourse determines the resolution of the fuzzified data. If the degree of membership is given by 4-bits then the resolution is 1/16. On the other hand, the number of elements in the universe of discourse determines the granularity of the fuzzy input. This number also represents a constraint on the number of possible Boolean sub-intervals.

TABLE I
PARAMETERIZED COMPONENTS

Component	Description
Number of Fuzzy Inputs	Defines the number of fuzzy inputs of the system. This value also determines the number of parallel MOM and Interval Detection circuits.
Resolution	The resolution of the degree of membership is resizable. This determines the number of bits needed to represent the degree of membership.
Granularity	This resizable property depends upon the number of elements in the universal set.
Boundary Count	This is the number of Boolean sub-intervals. It can be reset from problem to problem.
Boundary Limits	The right-most element of each Boolean sub-interval. The total number of limits is equal to the Boundary Count.
Number of Fuzzy States	The number of fuzzy states in the particular state cluster to be implemented.

V. TESTING AND VALIDATION

A software model that implements part of the HFB FSM is being used to develop an Intelligent Decision Support System (IDSS) for Eye-Hand Coordination Assessment. The initial results of the research were reported in [5]. The goal is to develop an automated assessment and training procedure for children with eye-hand coordination problems. This software model was used to verify the operation of the hardware implementation. By specifying the same parameters that are being used by the software model it is possible to confirm the hardware implementation works properly. The state transitions made by the hardware implementation are compared with the ones exhibited by the software model.

The universal set used in both the software model and the reconfigurable hardware implementation is shown in Figure 5. The Boolean sub-intervals used for the fuzzy inputs Accuracy (given in percentile) and Time (given in seconds) are normalized to just one uniform set to simplify the hardware implementation. Accuracy is defined with four sub-intervals ranging from Below Average to Excellent while the Time is also defined with four sub-intervals ranging from Excellent to Below Average. The upper bounds for the sub-intervals are also shown in Figure 5. The defuzzified values for the inputs Time and Accuracy are shown in Figures 7 and 8, respectively. The state transition graph to be implemented by the HFB FSM is shown in Figure 6. This state transition graph has been developed for children of age 5. It is based on experimental knowledge provided by Occupational Therapy experts. However, the research will be extended to include state transition graphs for children of age 6, 7, and 8. Supporting of the whole research project requires a reconfigurable software/hardware model which can adapt the

state transition graph to the age group that is being assessed. The state transition graph depicts a path to improved hand-eye coordination. Starting at state1 and reaching state12 without any repetitions indicates a significant improvement in the subject's eye-hand coordination skills. The state transition graph is traversed such that the next state depends upon the present state and the accuracy and time inputs received in a trial. Each pair of inputs represents one trial.

The hardware version of the reconfigurable fuzzy automaton was simulated using ModelSim by Mentor Graphics. The recorded state transitions were tabulated into a file and compared with the results of the software model. It has been found that the results of both the software simulations and hardware simulations are identical as shown in Figure 9. Figure 9 depicts a set of trials when state 12 is reached.

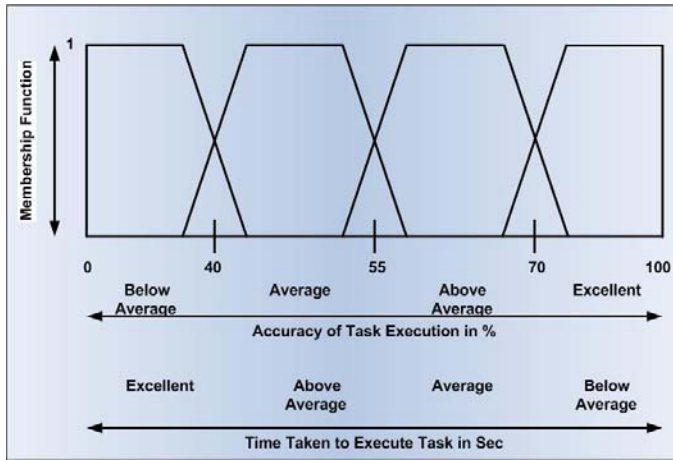


Fig. 5. Normalized universal set for Time and Accuracy inputs.

The look-up table implemented in both the software model and the hardware implementation uses the AND compression of the states of the two fuzzy inputs to trigger a state change. This is one limitation of the current hardware implementation of the fuzzy automaton. Another current limitation of the hardware is that the fuzzy inputs must be normalized to the same universal set. Future plans to improve the design include the implementation of don't cares and the OR compression to link the states of the inputs. This new feature will add more versatility to controllers with multiple inputs. In addition, different universal sets for each input will also be implemented.

VI. CONCLUSIONS

The paper presents the results of the hardware implementation of a reconfigurable state transition architecture for the Hybrid Fuzzy-Boolean Finite State Machine (HFB FSM). By using parameterized components a great degree of versatility with respect to the number of fuzzy inputs, as well as the granularity and resolution of each fuzzy input is achieved. This represents a very attractive property to implement virtual fuzzy automata for supervisory controllers of complex systems. The architecture presented allows the

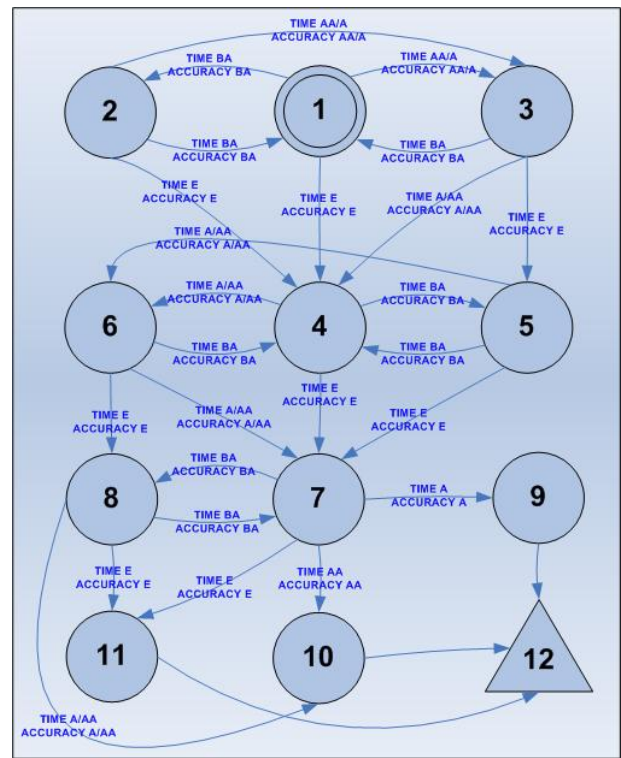


Fig. 6. State Transition Graph.

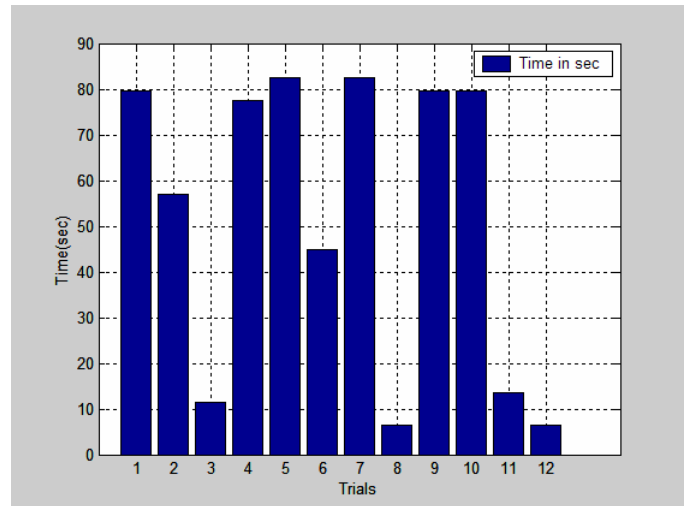


Fig. 7. Defuzzified Time input in seconds.

supervisory controller to reset the fuzzy automaton in order to model a particular state cluster as needed. Using an independent software model, the operation of the hardware implementation has been verified.

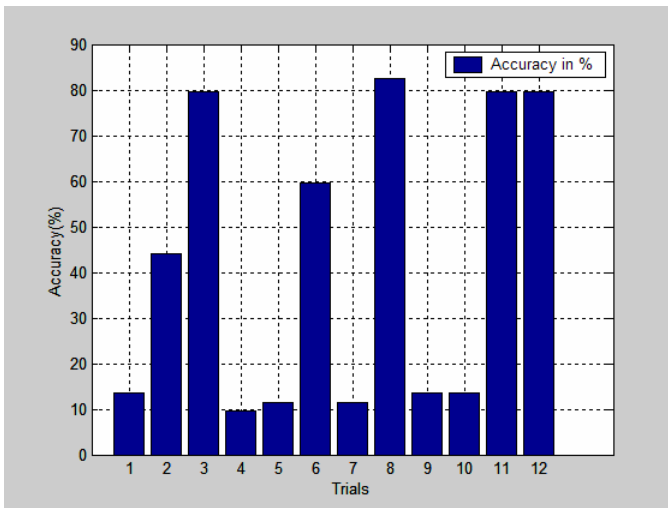


Fig. 8. Defuzzified Accuracy input in percentile.

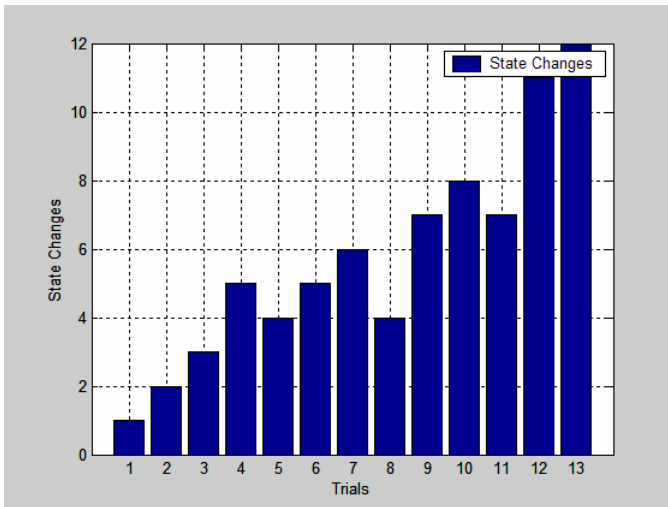


Fig. 9. States visited during trials.

REFERENCES

- [1] Grantner, J.L., Fodor, G.A., "Fuzzy Automaton for Intelligent Hybrid Control Systems", FUZZ-IEEE'02. *Proceedings of the 2002 IEEE International Conference on*, Volume: 2, 12-17 May 2002 Pages:1027 – 1032
- [2] Grantner, J.L., Fodor, G., Driankov, D., "Hybrid Fuzzy-Boolean Automata for Ontological Controllers", *Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, Volume: 1, 4-9 May 1998 Pages:400 - 404 vol.1
- [3] Grantner, J.L., Patyra, M.J., "Reprogrammable Fuzzy Logic Finite State Machine Model", *NAFIPS. 1996 Biennial Conference of the North American Fuzzy Information Processing Society*, 19-22 June 1996 Pages:492 – 496
- [4] Fodor G., "Ontologically Controlled Autonomous Systems: Principles, Operations and Architecture". Kluwer Academic Publishers, Boston/Dordrecht/London 1998
- [5] Grantner J. L., Gottipati R., Pernalet N., Fodor G., Edwards S., "Intelligent Decision Support System for Eye-Hand Coordination Assessment". 2005 *FUZZ-IEEE Conference*, Reno, Nevada, May 22-25, 2005