International Journal of Computer Science Engineering Research and Development (IJCSERD), ISSN 2248 – 9363(Print)
ISSN 2248 – 9371(Online), Volume 3, Number 1
Jan- March (2013), pp: 41-59
© PRJ Publication, http://www.prjpublication.com/IJCSERD.asp

IJCSERD

© PRJ PUBLICATION

AN IMPROVEMENT IN HISTORY BASED WEIGHTED VOTING ALGORITHM FOR SAFETY CRITICAL SYSTEMS

UMAMAHESWARARAO BATTA
Research Scholar
Dep. of Computer Sciences
Andhra University
Visakhapatnam, India

SEETHARAMAIAH PANCHUMARTHY
Professor
Dep. of Computer Sciences
Andhra University
Visakhapatnam, India

ABSTRACT

Fault masking is a widely used strategy for increasing the safety and reliability of computer control systems. Voting algorithms are used to arbitrate between the results of redundant modules in fault-tolerant systems. Inexact majority and weighted average voters have been used in many applications, although both have problems associated with them. Inexact majority voters require an application-speci6c 'voter threshold' value to be specified, whereas weighted average voters are unable to produce a benign output when no agreement exists between the voter inputs. The approach uses some form of voting to arbitrate between the results of hardware or software redundant modules for masking faults. Several voting algorithms have been used in fault tolerant control systems; each has different features, which makes it more applicable to some system types than others. Fault masking is one of the primary approaches to improve or maintain the normal behavior of a range of safety-critical systems. Some industrial sectors which employ such systems include process control, Transportation, nuclear power station and military applications Majority and weighted average voters have been widely used in these applications to provide error/fault-masking capability. Safety critical systems are the systems which may lead to hazards, loss of lives and great damage to the property if they fail due to errors which may lead to faults. N-Modular Redundancy or N-Version Programming along with the voter is used in the safety critical systems to mask the faults. This paper introduces a novel voting scheme based on fuzzy set theory. The voter assigns a fuzzy difference value to each pair of voter inputs based on their numerical distance. A set off fuzzy rules then determines a single fuzzy agreeability value for each individual input which describes how well it matches the other inputs. The agreeability of each voter input is then defuzzi6ed to give a weighting value for that input which determines its contribution to the voter output. The weight values are then used in the weighted average algorithm for calculating the voter 6nal output. The voter is experimentally evaluated from the point of view safety and availability, and compared with the inexact majority voter in a Triple Modular Redundant structured framework. The impact of changing some fuzzy variables on the performance of the voter is also investigated. We show that the fuzzy voter gives more correct outputs (higher availability) than the inexact majority voter with small and large errors, less incorrect outputs (higher safety) than the inexact majority voter in the presence of small errors, and less benign outputs than the inexact majority voter. In this paper different existing weighted average voting algorithms are surveyed and their merits and demerits or limitations are discussed based upon which a novel History based weighted Voting algorithm with Soft Dynamic threshold is proposed. Experimentation results of the novel voting algorithm for Triple Modular Redundant (TMR) system are compared with existing voting algorithms and the novel voter is giving almost 100% Safety if two of the three modules are error free and giving better results for one error free module. Novel voter is also giving better results for the multiple error conditions with all the modules having errors.

Index Terms: Triple Modular Redundancy, Result Amalgamation, Weighted Average Voters, History Records, Soft Dynamic Threshold, Safety Critical Systems

I. INTRODUCTION

Voter is a critical component in the implementation of N-Modular Redundant systems [13, 14]. Voting can be a hard problem in itself, for at least three reasons: i) floating point arithmetic is not exact and thus voting on floating point values requires inexact voting, ii) the output of variants (redundant modules) may be extremely sensitive to small variations in critical regions, e.g. around threshold values in the specification, and iii) some problems have multiple correct solutions (e.g. square roots) which may confuse the voter. Different voting strategies have been introduced to handle these problems; examples are maximum likelihood voter [15], predictor voters [16], stepwise negotiation voter [17] and word voters [18]. Two traditional and widely used voting algorithms are the majority and the weighted average voters. In its general form, an inexact majority voter [19] produces a correct output if the majority of its inputs match each other; that is, they are within an application-specific interval of each other. In cases of no majority, the voter generates an exception flag which can be detected by the system supervisor to drive the system towards a safe state. Efficient implementations of the majority voter have been addressed in [20-22].

The weighted average voter on the other hand, calculates the weighted mean of its redundant input values. It is useful in applications such as clock synchronization in distributed computer systems, pattern recognition and sensor planes where a result has to be generated in each voting cycle. The weights can be predetermined or can be adjusted dynamically. Calculated weights, wi, are then used to compute the voter output, $y = \Sigma wi. Xi / \Sigma wi$ where *xi* values are the voter inputs and *y* is the voter output.

The standard majority and weighted average voters are examples of two distinct groups of voting algorithms. One with a high level of safety yet with a low level of availability and the other with a low level of safety yet a high level of availability. This paper introduces a novel voter with a compromise safety performance between the standard inexact majority and weighted average voters. It also gives a higher availability than both the majority and weighted average voters.

Safety Critical systems are the systems which may lead to hazards, loss of lives or great damage to the Property if they fail. There are different domains in which safety critical control systems are used - Automotives – Drive-by-wire systems, Break by wire systems used in cars, Medicine - Infusion pumps, Caner Radiation Therapy machines etc., Military and Space applications -Rocket launchers, Satellite launchers etc., Industrial Process Control, Robotics and Consumer electronic appliances. There is a need to increase the reliability, availability and safety in all these applications. Faults that occur in these applications may lead to hazardous situations. If a single module or channel is used and when it becomes faulty due to some noise the system may fail and hazard may occur.

Hence N – Modular Redundancy or N-Version Programming along with voting technique is used to mask the faults in the faulty environments[1][2]. There are different architectural patterns [10] in which redundant modules with a voter are used in the safety critical systems. All the N-modules or N-versions [3] are designed by different teams to meet the same specifications. All these modules take the same input data, process it and generate the results which will be passed to the voter. The voter has to mask the fault by isolating or avoiding the faulty module and the correct value has to be picked by the voter. There are different types of voting algorithms [7] mentioned in the literature. Some Voting algorithms like Majority, Plurality voters [4] generate the output if the majority or required numbers of inputs to the voter are matched; otherwise it will generate no output so that the system can be taken to the fail safe state. Adaptive Majority voting algorithm [9] gives better performance by using history records. But for some safety-critical systems, there may not be any fail safe state. In such systems, the voter has to generate some value as the output using some methods like amalgamating the outputs or results of all modules, which is called as result amalgamation. Median, average, weighted average voters are some examples for the voters which amalgamate the inputs of the voter and generate some value as the voter output. History based weighted average voters consider the history of the modules and for the highly reliable module high weight is given. In this research work, Instead of harsh threshold, Soft threshold which can be changed dynamically is used to find the agreeability value of each module output with the other remaining module outputs. Harsh threshold results in agreeability value of either 0 or 1 but soft threshold method uses fuzzy Z function to generate agreeability or closeness value as shown in Figure 2.

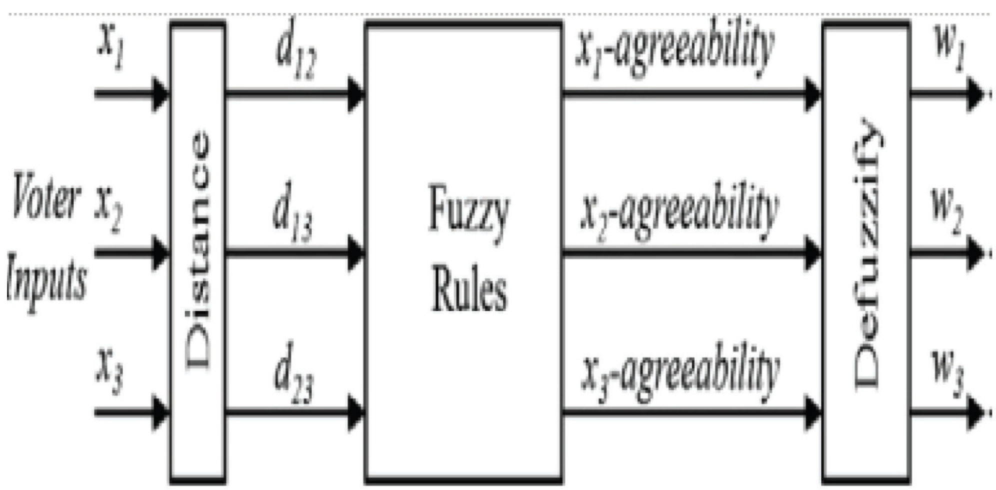
This Research Paper is organized as follows:
Section II is the literature survey of the existing voting algorithms.
In Section 3, Implementation of fuzzy voter

In Section 4 Improvement in History based weighted Voting algorithm with soft dynamic threshold is given
In Section 5, Experimental method and Test Harness is described
In Section 6, Experimental results are analyzed
In Section 7, Conclusions and Future works are given.

2. RELATED WORKS

Fuzzy voter designed in this paper is nothing but a softened inexact majority voter. In this voter there is a need for two thresholds. All the distance values or agreement distance values for each pair of module outputs, below the lower threshold are considered as complete agreement cases. The distances above the upper threshold are considered as complete disagreement cases. The middle distance values between lower and upper thresholds are processed using fuzzy approach. In this fuzzy approach three parameters p, q and r are used which will decide small, medium and large membership values as shown in Figure 1. Rule based fuzzy inference step along with centroid norm for defuzzification are used in this voter. But the fuzzy parameter values are statically selected in this voter and the performance of the voter varies with variation of these fuzzy parameter values. Static selection of fuzzy threshold parameter values is a major limitation in this voter. There is a need for automatic dynamic selection of values for these parameters for any kind of dynamically varying input dataset.

Fig.1: Structure of a three-input fuzzy voting unit



Fuzzy voting approach is given by Blank et al. (2010) for sensor fusion for the systems with low system information. In this fuzzy voter design, only fuzzy membership functions are used instead of fuzzy rule based inference. Scores are assigned for each sensor based upon these fuzzy membership function values and then fused output value is calculated as a weighted average using these scores. Computational complexity is reduced compared to the rule based fuzzy and centroid norm for defuzzification. Performance of this membership function based voter is little lower compared to the rule based fuzzy voter but computational complexity is far reduced. In this voter also, optimal values are selected for fuzzy parameters statically based on trial and error method. These fuzzy parameters may not be efficient for the other data or dynamically changing data. The steps involved in the static fuzzy voting approach given by Blank et al. (2010) for sensor fusion are given below.

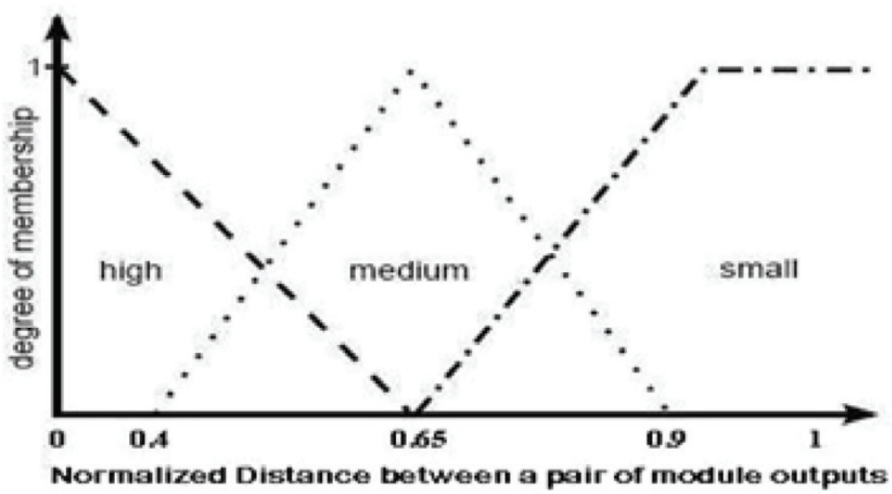
1) Find the Eucledian distance between each pair of module outputs i, j.

dij=|xi-xj|

```
For eah module i = 1 to n
    For each module j = 1 to n
        DM (i, j) = dij
    End
End
```

2) From the distance matrix DM by copying the upper triangle matrix elements except the diagonal elements into the vector ND.
3) Normalize the module agreement distances in the set ND in the 0 – 1 scale before using them in the fuzzy membership functions.
4) Specific shapes used for fuzzy membership functions are shown in Figure 2.

Figure 2: Fuzzy membership functions for static fuzzy voter



The concept of three point's fuzziness is when the fuzzy value boundaries and its most probable or most advisable value are known. Such way of the definition of fuzzy value can be described by the triangular membership function. With the normalized module agreement distance values, compute the membership vector µx(dij) for the agreement sets. Later express degree of agreement in the module outputs in a closed linear form. Therefore it is Possible to calculate the membership values µset(dij) very efficiently.

Module scores are computed instead of making use of fuzzy rule set for inference step. Scores for module i and j which have dij as the module agreement distance can be calculated as follows:

scorei += µhigh(dij) + µmed(dij) - µlow(dij)
scorej += µhigh(dij) + µmed(dij) - µlow(dij)

Initially, all the module scores are initialized to zero. For each normalized distance of module pairs in closed linear form, scores are updated by accumulating with the newly computed score for the corresponding module.

For example for TMR system, for each of the normalized distances d12, d23 and d31, corresponding module scores are updated as given.
For normalized distance d12,

Score1+= µhigh(d12)+ µmed(d12)- µlow(d12)
Score2+= µhigh(d12)+ µmed(d12)- µlow(d12)

For normalized distance d23,

Score2+= µhigh(d23)+ µmed(d23)- µlow(d23)
Score3+= µhigh(d23)+ µmed(d23)- µlow(d23)

For normalized distance d31,

Score3+= µhigh(d31)+ µmed(d31)- µlow(d31)
Score1+= µhigh(d31)+ µmed(d31)- µlow(d31)

6) If any module score is negative, normalize all the module scores to make all of them positive.

Calculate the voter output as a weighted average of scores and module output values.
If *i n scorei* = _1 >0
Output = *i n scorei xi* = _1 * / *i n scorei* = _1
Otherwise
Output = (x1+x2+……..+xn)/n

Figure 3: The distance matrix

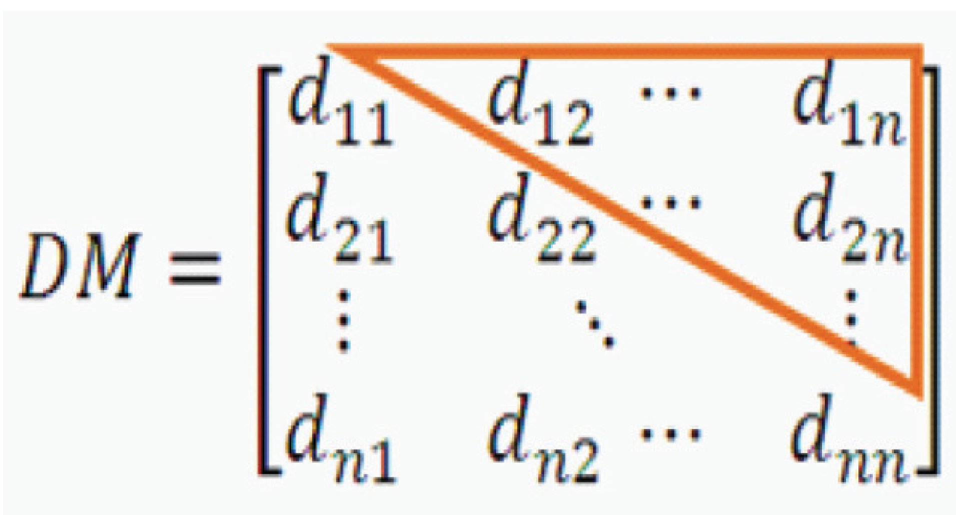
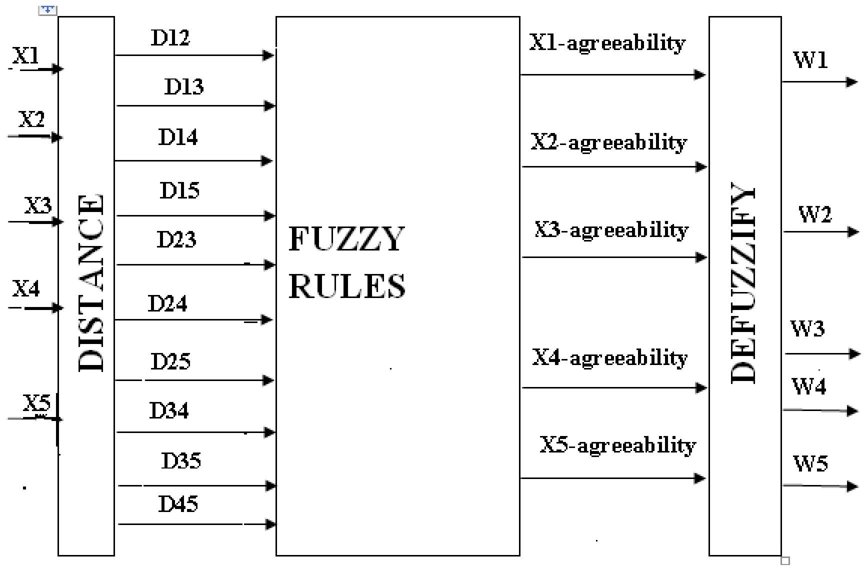


Fig.4: Structure of a five-input fuzzy voting unit



Basic Lorkzok's Standard weighted average voting algorithm (Lorkzok WA):

In this voting algorithm [8] weights are calculated based on the distances between the module outputs as given below

$$Wi = \frac{1}{1+\prod_{\substack{i=1,j=1 \\ i!=j}}^{N} \frac{d^2(xi,xj)}{\alpha^2}}$$

Where d (xi,xj) is the distance between the output values of module i and module j and a is a scaling factor.

After assigning the weights, output of the voter is calculated as follows:

$$Y = \sum_{i=1}^{N} \left(\frac{wi}{s}\right) xi$$

Where *s* is the sum of all the weights In this algorithm, reliability of the modules in the previous voting cycles called history is not considered.

History based weighted average voting algorithm Algorithm for building history records:

History records [6] are built based on the reliability of the modules. If a module has contribution for the majority consensus of the outputs of all the modules in a particular voting cycle, then a Boolean variable is set to 1 otherwise

Cleared to 0 the cumulative sum of this Boolean variable up to the current voting cycle is calculated which is the

History record of a particular module A module with high cumulative sum value is the highest reliable module and

The one with low cumulative sum value is less reliable module.

This history value is normalized by dividing it by the cycle number and is called as the state indicator Pi of the

Module I. There are two versions of history based weighted average voters called state indicator based and

Module elimination based weighted average voting algorithms as described in the reference [6].

In the state indicator based weighted average voting

Algorithm (HWA1), weights are assigned based on the state indicator Pi value

Wi = pi

In module elimination based weight assignment *(HWA2)* method if state indicator value of a module is less than the average state indicator value of all the modules then weight for that module is assigned as zero and eliminated from contributing to the voter output.

Wi = 0 if Pi < Pavg

Where Pavg = (P1+P2+P3+............+PN)/N

Otherwise Wi = Pi

If we consider Triple Modular Redundancy (TMR), these two versions work well if the same two modules

Consistently reliable and the other module generates outputs with some error. But in the reality, any module

May fail randomly and generate erroneous outputs. The existing history based weighted average algorithms failed

To produce the correct results even though majority of the modules have generated the error free outputs. This

Problem occurred since values for weights are assigned only based upon history. The module which generates

Correct output in the present cycle may be neglected and zero or less weight may be assigned for that module if it

Has poor history record. Hence proper weight is not given for the degree of closeness or agreement of a

Module with other module outputs.

Weighted average voter with Soft Threshold (WA ST):

In this voting algorithm [5] Degree of Closeness is calculated. Degree of closeness of each module with

Other module is calculated and average agreement value is calculated and assigned as a weight for that module.

Threshold is made soft by using a roll-off constant which is tenable. But in this algorithm history is not used. This

Algorithm generated no output or benign output if all the weights of all the modules are assigned zero value.

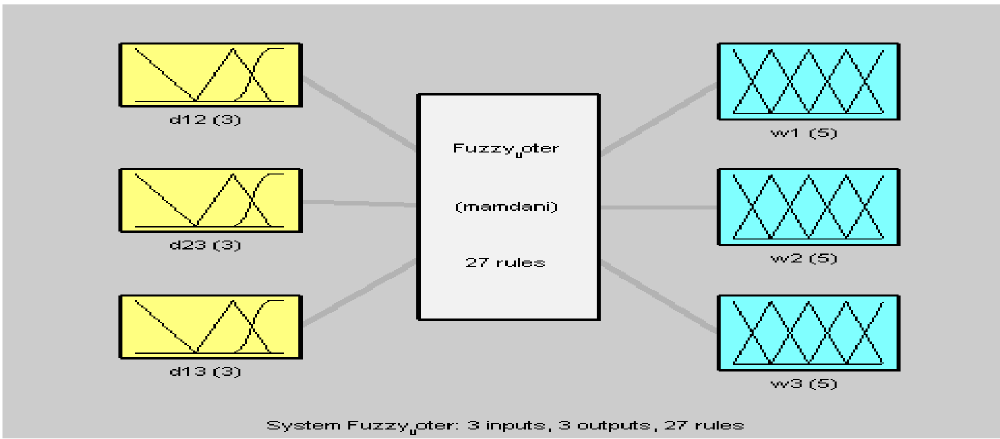
In Reference [11], Modified History based weighted average voting with soft dynamic threshold is given. In

This work, the threshold is calculated based upon the notional correct output of the voter. It is difficult to predict the voter output before only to decide the threshold. It is a major limitation in this voter. In Reference [12], a neural network based voter is designed and the neural network is trained using feed forward error back propagation algorithm. It is time taking process to train the network.

3. FUZZY VOTER

The fuzzy voter described herein uses fuzzy logic to generate the weights required for calculating a weighted average voter output. Fig. 5 shows the basic structure of a three-input fuzzy voter.

Fig 5: structure of 3 input voter



3.1. Calculating the fuzzy difference of input pairs

The first step in the approach requires the de6nition of a fuzzy difference variable to describe each pair of inputs to the voter. For each pair xi and xj with numerical distance dij , based on the triangular membership functions shown in Fig 6. , we de6ne a fuzzy difference variable represented by a set of membership grades µA(dij) where A: {small; medium; large}.Where symmetrical sets are used, this requires two parameter values to be speci6ed. Based on the numerical difference between any two inputs, a non-zero membership grade will be assigned to one or two of the fuzzy sets defined for the corresponding fuzzy difference variable. For convenience triangular fuzzy membership functions are used. which a ramp function in place of the traditional hard (that is, discontinuous) threshold found in traditional inexact majority voters.

Fig 6: Definition of the difference variable membership functions

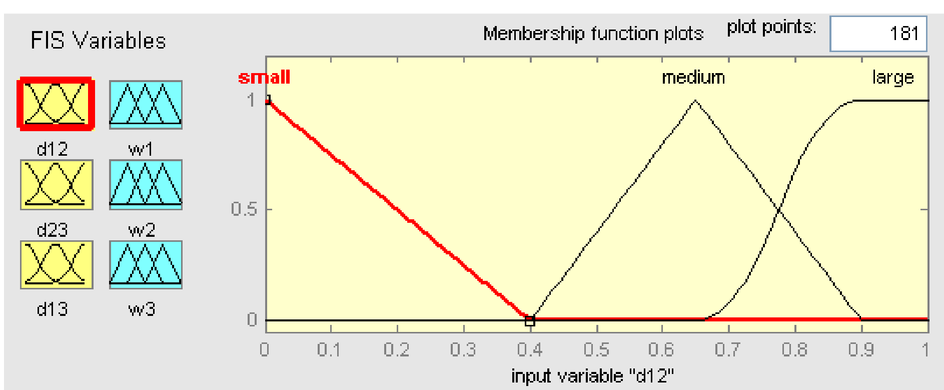


Fig. 6 shows two qualitatively different fuzzy difference variables. In the first case, there is a significant region in which two inputs which differ by a non-zero amount are regarded as being in de6nite agreement; an intermediate region in which the difference is specified using linguistic variables that may be true to a lesser of greater extent (for example, the difference between two inputs may be such that a non-zero membership is awarded to the small and medium fuzzy difference sets); and a third region which identi6es inputs that are in definite disagreement. In the case of the second fuzzy variable, there is no region of definite agreement speci6ed, although there is (as one would expect) a region ofde6nite disagreement.

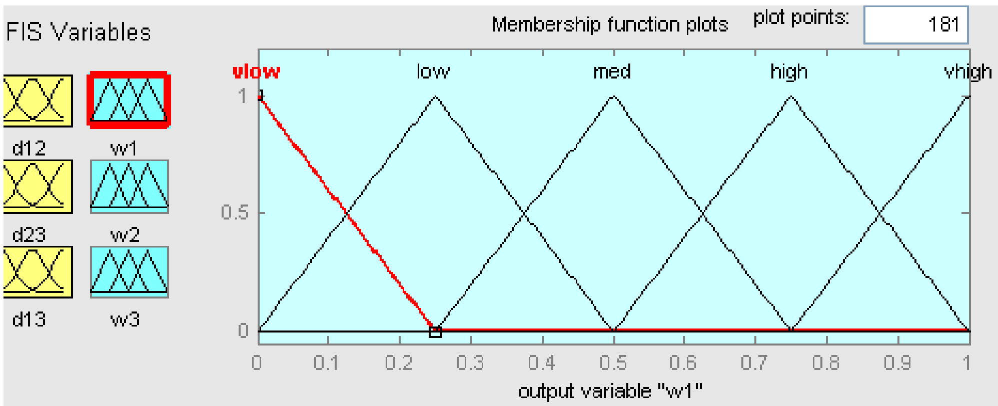


Fig 7: Definition of the output fuzzy variable membership functions

The final output value y for an m-way fuzzy voter is obtained by weighting each input signal xi with the calculated weight wi:

$$y = \Sigma wi \, . \, xi \, / \, \Sigma wi$$

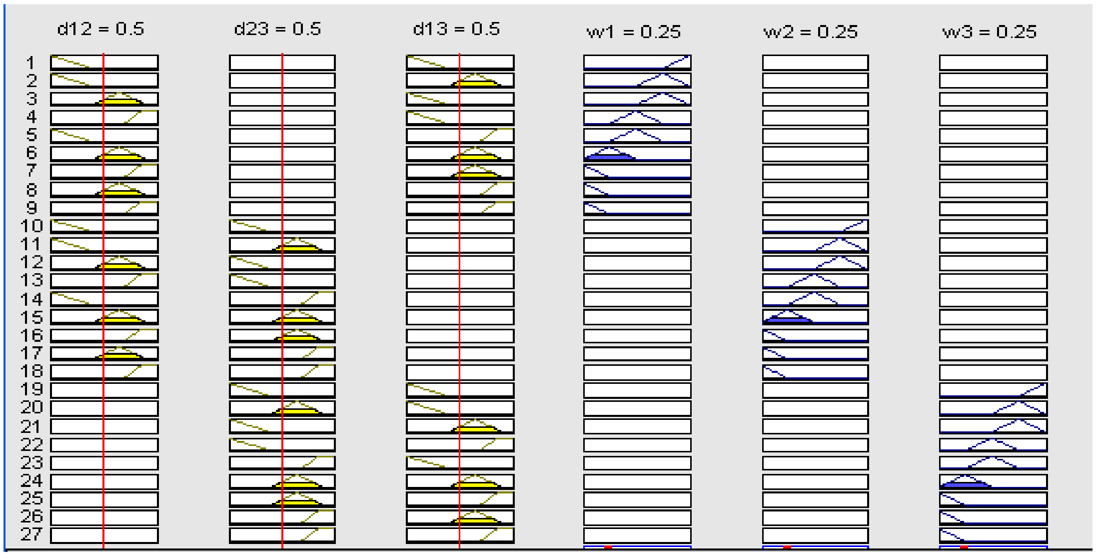
3.3. Fuzzy rule set definition

Table 1 shows a rule matrix that summarizes one possible set of fuzzy rules for combining and
Mapping fuzzy difference values onto a fuzzy agreeability value in a 3-input system.

TABLE1: Rule matrix for fuzzy input variables

&		Dij		
		small	Medium	large
Dik	small	vhigh	Med	high
	medium	med	Low	vlow
	large	high	Vlow	vlow

Fig 8: The Rule Viewer for fuzzy voter as follows



4. AN IMPROVEMENT IN HISTORY BASED WEIGHTED VOTING ALGORITHM FOR SAFETY CRITICAL SYSTEMS

A correctly functioning weighted average voter always generates the weighted mean of its inputs that is identical to or in between the inputs that the majority voter would select as in agreement. It is obvious that the output of inexact majority and weighted average voters for all agreement voting cycles are similar. This implication leads us to introduce a novel voter that is a combination of majority and weighted average voters. It performs as a majority voter in agreement cases, and functions as a weighted average voter in disagreement voting cycles. The voter is less complex and quicker than the weighted average voter, since in majority of the cases it does not perform the relatively time consuming weighted averaging procedure.
A novel history based weighted average voting algorithm with soft dynamic threshold is given below:

1. Let x_1, x_2, …x_m be the voter inputs and y its output.
2. The distance between the output of module *xi* and output of modulej xj is calculated as
   dij = |xi-xj|
3. Closeness index *Sij* using following formula

$$Sij = \begin{cases} 1 & \text{if } dij<=VT \\ 1-\frac{|dij-vt|}{n*vt-vt} & \text{if } dij<=n*VT \\ 0 & \text{if } dij>n*VT \end{cases}$$

   Where n is a variable that can be assigned a value >=2 to make the threshold soft. And dij is the distance between i and j module outputs and *vt* is the voting threshold.
4. Calculate History values using the procedure given in the Reference [6] but use n*VT as the threshold for agreement while calculating history records. Find the Normalized history values for each module by dividing the history with cycle number.

5. History and Closeness Product (H) for each module as follows
   Hi = Pi*(Σ Sij/N-1)
   Where N is the Total number of modules And Pi is a normalized history value of the module i and Pi=Histi /cycleno
6. Normalized History average Pavg
   Pavg $= \sum_{i=1}^{N} \frac{pi}{n}$
7. Calculate the weights for all N modules as follows
   For i =1 to N
   if HCPi=0 AND Pi<Pavg
   Wi=0
   Otherwise
   Wi=2*Hi

8. If all the weights are equal to zero in the worst case, Modify the weights as follows
   Wi=pi2
   for i= 1 to N
9. Calculate weighted average using the weights
   $Y = \frac{\sum_{i=1}^{N} wi\,.\,xi}{\sum_{i=1}^{N} wi}$
   Where Wi is the weight of i th module and xi is the output of i th module

5. EXPERIMENTAL METHODOLOGY

5.1. Test Harness: Test Harness for experimentation with voting algorithms is shown in the Figure 9. Below.

The input generator produces one notional correct result in each voting cycle. This sequence of numbers identical correct results expected from redundant modules. Copies of the notional correct result are presented to each saboteur in every voting cycle. The saboteurs can be programmed to introduce selected module error amplitudes, according to selected random distributions. The symptom of errors appears to the voter as numerical input values. A comparator is used to check for agreement between the notional correct result and the output of the voter under test at any voting cycle. However, for simplicity, issues associated with ensuring synchronization of the inputs to the voter and to the saboteurs are ignored.

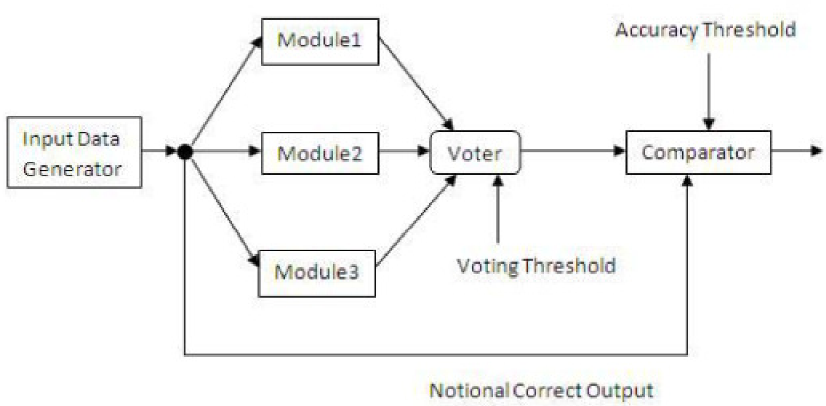


Figure 9: Experimental Setup to evaluate the Performance of a Voter

A voter threshold *(*dynamic or fixed)*, VT,* is used to determine the maximum acceptable divergence of voter inputs in each voting cycle from the notional correct result, and an accuracy threshold, *AT*, is used in comparator to determine if the distance between the notional correct result and the voter output is within acceptable limits. In this framework, the accuracy threshold is chosen equal to the voter threshold in each voting cycle. A voter result which has a distance from the notional correct answer less than the accuracy threshold is taken as a correct output, otherwise it is considered as an incorrect output. This is a valid assumption in a many real time systems in which the discontinuity between consecutive correct variant results is small (Bennett, 1994). Hence, the presence a large discontinuity is indicative an error and can be detected by the acceptance tests. Where the voter cannot reach an agreement between the outputs of saboteurs, it produces a of default value that moves the system toward a fail-safe or fail-stop state. Such voter output is called a disagreed (benign) result. It is also assumed that all voters perform correctly. This assumption is made due to the fact that the voting algorithm is usually a simpler program than the modules it monitors.

Cyclic data like Sin wave is generated using the equation
Given below
Input data = 100 + 100 * sin (t)
Sample rate t is taken as 0.1.
Generated input data is given to each of the modules and the random error of uniform distribution is injected
Into each of the required module in the required range [-e, +e]. Initially generated input data before injecting the
Error is considered as the notional correct output. Fixed voting Threshold and Accuracy Threshold are
Considered as 0.5
For the Soft Dynamic Threshold methods, Voting Threshold can be varied dynamically. For the Weighted Average voter with Soft Dynamic Threshold the tunable parameter is taken as 5. For the Novel Algorithm n=5 taken which is same as tunable parameter of Weighted Average voter with Soft Dynamic Threshold so that results can be compared
Based on the n value threshold is changed. Closeness Index varies as shown in the Figure 10. Below based upon
The distance measure.

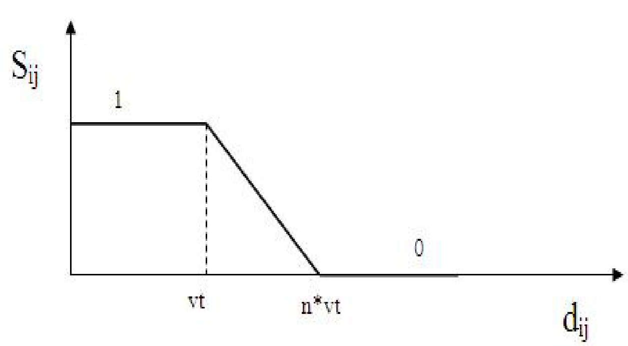


Figure 10: Closeness Index (Sij) versus distance measure (dij) in the novel algorithm

The generated output by the voter is compared with the notional correct output and if the difference is less

Than the accuracy threshold value, it is considered as the correct result otherwise incorrect result. Each set of Experiment is performed for 10000 runs and the number of correct results (nc) and number of incorrect results (nic) are counted.

Then the performance of the voter is evaluated by using the parameters Availability and Safety as given below:

Availability = nc /n

Safety = 1-(nic/n)

Where nc = Number of correct results given by a voter

nic = Number of Incorrect results given by a voter and

n= Total number of runs or voting cycles

Safety (S): Since from a safety viewpoint the smallest number of agreed but incorrect outputs is desirable for a given voter, the safety measure can be defined as: $S = (1\text{-}n\ ic\ /\ n)$. Thus S $[0\ 1]$ and ideally $S=1$.

Reliability (R): A voter which produces more correct results among its total outputs can be interpreted as more reliable voter. Reliability is defined as the ratio of correct voter outputs to the number of voting actions:

$R= n\ c$ / n. Thus $R[0\ 1]$ and ideally $R=1$.

Within the test harness the following parameters can be adjusted.

- The value of consensus threshold.
- Number of voter inputs. The test harness provides a facility to define 3, 5 and 7 inputs voters.
- Input data trajectory and sample rate. Different types of input data trajectory can be selected within the test harness. The frequency of data arrival (sample rate of input data) is also adjustable.
- The value of accuracy threshold.
- The amplitude of injected errors. The amplitude of injected errors can be expressed as a function of the input signal. If δ is defined as the maximum amplitude of errors during a particular test consisting of n voting cycles, and A is taken as the maximum amplitude of input data, the δ / A will be the maximum error-to-signal ratio (ESR) of that particular test.
- Number of injected errors. One or two or three saboteurs may be programmed to simulate variant results' errors.
- Error persistence time. The experimental harness has capability to select error persistence time,

  Error arrival intervals and its activation period Errors can be permanent, transient, and intermittent.
- Error Distribution. A variety of error distributions including uniform, exponential, normal and

  Poisson distributions with adjustable parameters have been defined within the test harness.

5. EXPERIMENTAL RESULTS

Empirical evaluation of the safety performance of the voters is done by running each voter for 10000 voting cycles.

Fig 11: Safety Comparison with all modules having equal amplitude errors (Small Errors) for 10000 runs

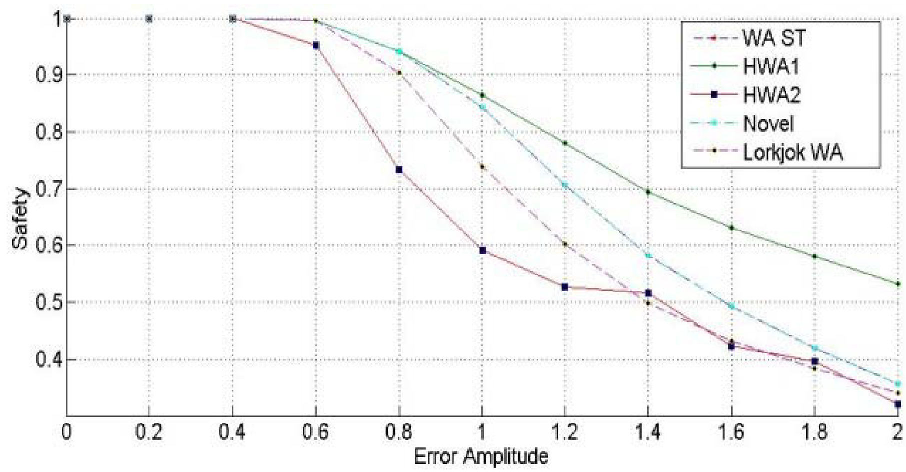


Figure 11. Shows safety performance for small errors for the large and small errors, proposed novel algorithm has better safety performance than the Module elimination version of the History based weighted average voter, Lorkzok's weighted average voter and Weighted average voter with Soft dynamic Threshold. But in this case, State Indicator version of the History based weighted average voter is performing better than all other voting algorithms since it considers only history to assign the weights.

Safety of different voting algorithms with two error free modules for 10000 runs is compared in Figure.

In this upto 3000 cycles' module1 and module2 are error free and module3 is perturbed with an error in the range

[-e, +e], From 3001 to 7000 cycles module1 and 3 are error free and module2 is perturbed with an error in the range [-e, +e], From 7001 to 10000 cycles modules 2 and 3 are error free where as module1 is perturbed with an error in the range [-e,+e].The two History based weighted average versions called State Indicator based version and Module Elimination based version failed to give 100% safety even though two modules are error free. The reason is, much

Importance is given for previous reliability history but in the current voting cycle, things may be different. A module which has got good history so far may be perturbed with errors in the current voting cycle. But due to its past reliability history, It is given high weight and the erroneous module contributes much for the voter output. This is a major limitation in the two versions of history based weighted average voter which has been Overcome in the Novel Algorithm by taking the History and Closeness Product (HCP) into consideration as given

In the algorithm while assigning the Weights assigned and outputs for the given input values are shown in the Table 2. And Table 3. For Module Elimination based weighted average voter and proposed novel History based weighted average voter with Soft Dynamic Threshold respectively. In the Table 2. And Table 3. Column headings are given

Below x – Notional Correct output x1, x2, x3 are the outputs generated by module1, module2 And module3 respectively H1,H2,H3 are history values of the modules and w1,w2,w3 are the weights assigned for the modules and HWA O/P is the output produced by the module elimination version of History based weighted average voter. NH1, NH2, NH3 are the history values and N_w1, N_w2, N_w3 are the weights assigned for the
Modules and N_O/P is the output produced by the proposed novel algorithm.

In the Table 2 and Table 3 Third module is perturbed with error up to 20 th cycle where as remaining two modules are error free and there onwards for the remaining voting cycles, Second module is perturbed with error where as remaining two modules are error free. Module Elimination version of History based weighted average voter results are compared with the Novel voting algorithm. If the same two modules are consistently error free, module elimination based version is producing the correct results. But practically this is not possible. Any module may be inconsistent and fail randomly at the runtime. Cycle no 21 onwards, second module is perturbed with errors. But module elimination version gives importance to the previous history and hence gives high weight to the erroneous module2 as shown in Table.3. Due to this high weight, it contributes much for the result. Module elimination version needs some recovery time. Whereas, the novel algorithm considers History and Closeness or Consensus for Majority of each module to assign the weights and is able to produce correct outputs as shown in Table.3, if any two modules are error free

TABLE 2: OUTPUTS GENERATED BY HISTORY BASED WEIGHTED AVERAGE VOTER

Cycle no	x	X1	X2	X3	H1	H2	H3	W1	W2	W3	o/p
16	199.749	199.749	199.749	195.921	16	16	1	1	1	0	199.749
17	199.957	199.957	199.957	205.465	17	17	1	1	1	0	199.957
18	199.166	199.166	199.166	200.042	18	18	1	1	1	0	199.166
19	197.385	197.385	197.385	190.56	19	19	2	1	1	0	197.385
20	194.63	194.63	194.63	184.68	20	20	2	1	1	0	194.63

TABLE 3: OUTPUTS GENERATED BY PROPOSED NOVEL VOTER

Cycle no	x	X1	X2	X3	NH1	NH2	NH3	N_w1	N_w2	N_w3	o/p
16	199.749	199.74	199.74	195.92	16	16	1	1	1	0	199.749
17	199.957	199.95	199.95	205.46	17	17	1	1	1	0	199.957
18	199.166	199.16	199.16	200.04	18	18	2	1.8125	1.81249	0.1806	199.208
19	197.385	197.38	19738	190.56	19	19	2	1	1	0	17.385
20	194.63	194.63	194.63	184.68	20	20	2	1	1	0	194.63
21	190.93	190.93	191.42	190.93	21	21	3	2	2	0.2857	191.163
22	186.321	186.32	191.55	186.32	22	21	4	1	0	0.1818	186.321

7. CONCLUSIONS & FUTURE WORK

In this work, a Novel History based weighted average voter with Soft Dynamic Threshold is designed and safety performance is evaluated empirically for 10,000 voting cycles on a Triple Modular Redundant system (TMR). *Reliability history* of the modules and *closeness* or *agreeability* of a module output with other module outputs (majority consensus) in a voting cycle are used to assign the weights for the individual modules and final output is generated by calculating the weighted average of all the module outputs. The Novel voting algorithm is performing better and giving almost 100% safety if majority of the modules are error free which is the much needed behaviour for fault masking in the practical applications. Novel voting

Algorithm is also giving better safety performance for the multiple error scenarios compared to the other history based weighted average voters.

Majority consensus is established if the majority of the modules generate the same output values, which need not be correct. Majority of modules may coincidentally generate the erroneous same output and may cause for the majority consensus and contribute for the final output. This can be overcome using forecasting and prediction algorithms like double exponential smoothing and

Interpolation to predict the cyclic pattern data output for the current cycle based on the outputs of the past cycles and it remains the future work.

REFERENCES

[1] J.-C. Laprie, “Dependable computing and fault-tolerance: concepts and terminology,” in Digest of Papers FTCS’15: IEEE 15th Annu. Int.Symp. Fault-Tolerant Computing Systems, Ann. Arbor, MI, June 1985,pp. 2–11.

[2] B. W. Johnson, Design and Analysis of Fault-Tolerant Digital Systems New York: Addison-Wesley, 1989.

[3] L. Chen and A. Avizienis, “N-Version Programming: a fault-tolerance approach to reliability of software Operation,” in Digest of Papers FTCS’8: IEEE 8th Annu. Int. Symp. Fault-Tolerant Computing Systems, Toulouse,France, June 1978, pp. 3–9. York: Addison-Wesley, 1989.

[4] D. M. Blough and G. F. Sullivan, “A comparison of voting strategies for fault-tolerant distributed systems,” in Proc.IEEE 9th Symp. Reliable Distributed Systems, Huntsville, Alabama, Oct. 1990, pp. 136–145.

[5] G. Latif-Shabgahi, "A Novel Algorithm for Weighted Average Voting Used in Fault-Tolerant Computing Systems," Microprocessors and Microsystems, vol. 28, pp. 357-361,2004.

[6] G. Latif-Shabgahi, J. M. Bass, and S. Bennett, "History- Based Weighted Average Voter: A Novel Software Voting Algorithm for Fault-Tolerant Computer Systems," Euromicro Conference on Parallel, Distributed, and Network-Based Processing, pp. 402-409, 2001

[7] G. Latif-Shabgahi, Julian M. Bass and Stuart Bennett, “A taxonomy for software voting algorithms used in safetycritical Systems,” IEEE Trans.Reliability, vol. 53, no. 3, pp 319-328, Sept. 2004

[8] P. R. Lorczak , A. K. Caglayan, and D. E. Eckhardt, "A Theoretical Investigation of Generalized Voters for Redundant Systems," presented at FTCS-19. Digest of Papers. Nineteenth International Symposium on Fault-Tolerant Computing" chicago, USA, 1989.

[9] G. Latif-Shabgahi, S. Bennett, "Adaptive majority voter: a novel voting algorithm for real-time fault-tolerant control Systems," in 25th Euromicro Conf., vol. 2, 1999, pp. 113- 120

[10] S.Phani Kumar, P.S.Ramaiah,V.Khanaa, "Architectural Patterns to Design Software Safety based Safety-Critical Systems" Proceedings of ICCCS' 11 International Conference on Communication, Computing & Security,Feb 12-14, 2011, pp:620-623 ACM New York, NY, USA ©2011 ISBN: 978-1-4503-0464-1

[11] M.Das, S.Battacharya, "A Modified History Based Weighted Average Voting with Soft-Dynamic Threshold," in ACE '10 Proceedings of the 2010 International Conference on Advances in Computer Engineering, ISBN: 978-0-7695-4058-0 doi>10.1109/ACE.2010.45

[12] Zarafshan, F, Latif-Shabgahi, G.R., Karimi, "A Novel Weighted Voting algorithm based on Neural Networks for Fault-Tolerance Systems",ICCSIT, 2010, IEEE International Conference on Computer Science and Information Technology, pp.135-139

[13] S. Lee, J.-I. Jung, and I. Lee. Voting Structures for Cascaded Triple Modular Redundant Modules, IEICE Electronics Express, 2007, 4 (21): 657-664.

[14] A. B. Baykant. Hierarchical Triple-Modular Redundancy (H-TMR) Network For Digital Systems, OncuBilim Algorithm And Systems Labs. 2008, 8 (5), URL: http://arxiv.org/ftp/arxiv/papers/0902/0902.0241.pdf

[15] Y. W. Leung. Maximum Likelihood Voting for Fault-Tolerant Software with Finite Output Space", IEEE Trans. on Reliability, 1995, 44 (3): 419-427.

[16] J. M. Bass, P. R. Croll, P. J. Fleming, and L. J. C. Woolliscroft. Three Domain Voting in Real-Time Distributed Control Systems", Proc. of 2nd IEEE Euromicro Workshop on Parallel & Dist. Processing, 1994, pp. 317-324

[17] K. Kanekawa, H. Maejima, H. Kato, and H. Ihara Dependable On-Board Computer Systems with a New Method: Stepwise Negotiated Voting", Proc. of FTCS'19: IEEE 19th Ann. Int. Symp. On Fault-Tolerant Computing Systems, 1989, Chicago, USA, pp. 13-19.

[18] S. Mitra and E. J. McCluskey. Design of Redundant Systems Protected Against Common-Mode Failures", Technical Report, CRC-TR-00-2, Stanford University, 2000, URL: http://crc.stanford.edu.

[19] P. R. Lorczak, A. K. Caglayan and D. E. Eckhardt. A Theoretical Investigation of Generalised Voters", Proc. Of IEEE 19th Ann. Int. Symp. on Fault-Tolerant Computing Systems, 1989, Chicago, USA, June, pp. 444-451.

[20] H. Yu and A. Xu. Design of a Fault-Tolerant Voter for Safety Related Analog Inputs", Proc. of 3rd Int. Conf. on Advanced Computer Theory and Engineering, ICACTE, 2010, Chengdu, China.

[21] K. Kwiat, A. Taylor, W. Zwicker, D. Hill, S. Wetzonis, and S. Ren. Analysis of Binary Voting Algorithms for Use in Fault Tolerant and Secure computing", 2010, URL: http://lcs.syr.edu/faculty/tang/Teaching/CSE791-Spring11/Papers/Voting.pdf

[22] S. Askari, B. Dwivedi, A. Saeed, and M. Nourani. Scalable \Mean Voting Mechanism for Fault Tolerant Analog Circuits", Proc. of 4th Int. Workshop on Design and Test (IDT), 2009, Riyadh, Nov. 15-17, pp. 1-6

[23] tarun dhar diwan and upasana sinha, "eye tracking and detection by using fuzzy template matching and parameter based judgment" International journal of Computer Engineering & Technology (IJCET) Volume 4, Issue 1 (2013), ISSN Print : 0976 - 6367,ISSN Online: 0976 – 6375. Published By IAEME

ABOUT AUTHERS

Dr.Panchumarthy Seetha Ramaiah obtained his Ph.D in Computer Science from Andhra University in 1990. He is presently working as a professor of Computer Science in the department of Computer Science and Systems Engineering, Andhra University College of Engineering, Visakhapatnam- INDIA. His research interests are safety critical systems, VLSI design, Embedded Systems and Fault Tolerant Systems. Dr.P. Seetha Ramaiah is the Principal Investigator for several Defence R&D projects and Department of Science and Technology projects of the Government of India in the areas of Embedded Systems and robotics. He has published twelve journal papers, and presented twenty International Conference papers in addition to twenty one papers at National Conferences in India.

B.Uma Maheswara Rao obtained his M.Tech (CSE) in Computer Science from Andhra University in 2008. He is presently as a Research Scholar of Computer Science in the department of Computer Science and Systems Engineering, Andhra University College of Engineering, Visakhapatnam- INDIA. His research interests are safety critical systems, VLSI design, Embedded Systems and Fault Tolerant Systems. He has published one journal papers.