www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 8 August, 2013 Page No. 2649-2658

Performance Evaluation of Dynamic Fuzzy Voter Used In Safety Critical Systems

P.Babul Saheb* K.Subbarao
Dept.Of Information Technology Lakireddy Balireddy College of Engineering Mylavaram, India

ABSTRACT

Several voting algorithms have been described to arbitrate the results of redundant modules in fault-tolerant systems. The inexact majority and weighted average voters are widely used in control and safety-critical applications. Inexact majority voters require an application-specific '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. A major difficulty with inexact majority voters is the need to choose an appropriate threshold value, which has a direct impact on the voter performance. The problem of all documented weighted average voters is their inability to produce a benign output in cases of complete disagreement between the voter inputs. Both types of voters are unable to cope with uncertainties associated with voter inputs originated from erroneous software, noisy environment, or noisy hardware modules.

A voting scheme based on fuzzy set theory was introduced which softens the harsh behaviour of the inexact majority voter in the neighbourhood of the 'voter threshold' and handles uncertainty and some multiple error cases in the region defined by the fuzzy input variables. A set of fuzzy rules determines a single fuzzy agreeability value for each individual input which describes how well it matches the other inputs. 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. It is predicted that the fuzzy voter can be invaluable in at least two cases 1) as a substitute for the inexact majority voter in applications in which a small degradation in the safety performance of the system is acceptable at the cost of a large increase of its availability and a considerable decrease of its benign outputs 2) when arbitrating between the responses of dynamic channels of control systems which may include some uncertainty.

Automatic fuzzy parameter selection based dynamic fuzzy voter for safety critical systems with limited system knowledge. Existing fuzzy voters for controlling safety critical systems and sensor fusion are surveyed and safety performance is empirically evaluated. The major limitation identified in the existing fuzzy voters is the static fuzzy parameter selection. Optimally selected static fuzzy parameters work only for a particular set of data with the known data ranges. Dynamic voter is designed in such a way that it can be plugged in and used in any safety critical system without having any knowledge regarding the data produced and their ranges.

Keywords- fault tolerant system, fuzzy set theory, voter threshold, fuzzy input variables, automatic fuzzy parameter, dynamic fuzzy voter

1. INTRODUCTION

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, brake by wire systems used in cars; (medicine) infusion pumps, cancer radiation therapy machines, (military and space applications) rocket launchers, satellite launchers, etc.; and (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 [1] along with voting technique is used to mask the faults in the faulty environments [3]. There are different architectural patterns [2] in which redundant modules with a voter are used in the safety-critical systems. All the N-modules or N-versions 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 [4] mentioned. Some voting algorithms like majority, plurality voters [5] 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 [6] algorithm 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 to amalgamate the outputs or results of all modules which is called as result amalgamation [7]. Median, average, weighted average voters as given are some examples of the voters which amalgamate the inputs of the voter and generate some value as the voter output. History based weighted average voters [8] consider the history of the modules and for the highly reliable module greater weight is given. History based weighted voter [9] considers the history and the module agreeability product to calculate the weights of the modules which are used in weighted average calculation of the voter output. In modified history based weighted average voter with soft dynamic threshold [10] is designed. In this 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.

Fault-tolerance is the knowledge of manufacturing the computing systems which are able to function properly even in the presence of faults, These systems compromise wide range of applications such as embedded real-time systems, commercial interaction systems and e-commerce systems, Ad-hoc networks, transportation (including railway, aircrafts and automobiles), nuclear power plants, aerospace and military systems, and industrial environments in all of which a precise inspection or correctness validation of the operations must occur (e,g, where poisonous or flammable materials are kept)[4] In these systems, the aim is to decrease the probability of system hazardous behaviour and keep the systems functioning even in occurrence of one or more faults. Redundancy is one of the important methods in achieving fault tolerance and can be implemented in three forms including static (fault masking methods), dynamic (fault detection, fault diagnosis, fault isolation and fault location), and hybrid (masking faults and fault detection and location).

The aim in static redundancy is masking the effect of fault in the output of system; N-Modular Redundancy (NMR) and N-Version Programming (NVP) are two principal methods of static redundancy in hardware and software respectively. Three modular redundancies (TMR) is the simplest form of NMR which is formed from N=3 Voter performs a voting algorithm in order to arbitrate among different outputs of redundant modules or versions and mask the effect of fault(s) from the system output. Average voter is one of several voting algorithms which are applied in fault-tolerant control systems. Main advantages of this voter are its high availability and its potentiality to extend to large scale systems. Furthermore, in contradict with many voters like majority, smoothing and predictive; it does not need any threshold. The main problem of this voter is that whatever the number of inputs increases the complexity of its formula increases. Hence, more calculations overhead imposes and the processing speed will decrease. We use parallel algorithms on EREW shared-memory systems to present a new generation of average voter .we call as parallel average voter which provides the average voter extension without enlarging the calculations suitable for large scale systems and with optimal processing time. Basically there are two architectures for multi-processor systems. One is shared-memory multi processor system and the other is message passing [11]. In a shared-memory parallel system it is assumed n processor has either shared their public working space or has a common public memory.

Redundancy has been widely used to increase the fault tolerance of physical systems. A fault masking system uses redundant modules and a voting algorithm to mask the faulty/erroneous module outputs. The approach prevents the propagation of wrong results to the subsequent parts of a system and therefore increases the system safety. Many voting strategies [8][7][12][13] have been defined from which the majority voter and its modified versions have been widely used. In this case one of the agreed results is arbitrarily selected as the voter output. In cases of disagreement, the majority voter produces an exception code which leads the system toward a fail-safe state (in safety-critical systems) or a fail-stop state (in fail-silent systems). Therefore, the majority voter in a TMR system masks the failure of any single redundant module in each voting cycle. However, in many applications identifying faulty module(s) after masking is necessary to obtain an acceptable level of system availability. Depending on the nature of application, by identifying faulty channel(s) the corresponding module can be replaced on-line with a (one of the) back-up module (if exists) or is disconnected from the system to contribute to the voter output or is shutdown in a safe manner. These strategies are referred as 'system reconfiguration', 'system degradation' and 'fail stop' in the context of fault tolerant systems [14]. One approach to identify faulty modules is the use of their history record of within the voting algorithm. In this method a module with the worst history record is taken as the most erroneous/faulty module during the system operation. The history record of modules can also be used to improve the performance of voting algorithm. There is a little published work on the use of module history records in voting algorithms.

In [15] the cumulative number of times each version (software module of a TMR system) has participated in a majority agreement (exact, bit-by-bit majority) is taken as a history measure of versions. In cases of disagreement between version results, the output of a version with the highest history count is chosen as the voter output. Voter has lower failure probability (defined as the ratio of failed computations to total computations) over the standard majority voter. The three-domain predictor voter [56] uses the fault record of variants in producing voter output in cases of disagreement. In this algorithm, a number is associated with each variant which denotes the number of cycles in which the variant has not contributed toward a majority consensus by that cycle. The number is incremented each time a variant result does not participate in the consensus. In cases of complete disagreement, the closest variant result to the estimated output value with a distance within a predefined predicted threshold is selected as the voter output if its associated fault number is lower than the average fault record value (at any time, the average

fault record value is the mathematical average of variants fault number which has been counted by that time). If the associated fault number of selected variant result is less than the average fault record value, an exception flag is generated, that is the variant is interpreted as a low reliable module. Both of the mentioned research works use the 'recent history' of modules in disagreement cases to force an output to occur. It will be indicated that a majority voter in which the output selection mechanism[16] uses the recent history of modules in agreement voting cycles, produces more correct and less incorrect results than the standard majority voter during *n* runs.

1.1 PURPOSE

When facing the need to perform low-level sensor fusion with only very limited knowledge available one has to come up with an alternative to the well known and proven Kalman filter. A very interesting candidate for such applications is the so called fuzzy (or soft) voter. This algorithm makes use of fuzzy logic principles to fuse signals in an efficient way and provides a figure of merit as well as sensor monitoring capabilities with very moderate demand for computation performance and memory. A computational efficient alternative implementation of soft voting for embedded applications. The most common approach chosen to master low level signal fusion is the Kalman filter. It is very popular and commonly used for a very broad range of purposes like localization[17], street traffic modelling[18] , sensor calibration[19], or for economic processes[20]. This is because Kalman filtering allows for superior fusion performance and has the ability to estimate (i. e. overcome periods of insufficient measurement by means of forward projection of the last valid state). The price for such outstanding properties, however, is the need for extensive knowledge of the system's properties and characteristics. Among the parameters that have to be acquired prior to being able to use the filter is the system and measurement model that expresses the relation among the system inherent states to the measured quantities and the respective noise.

Furthermore, a few assumptions concerning the correlation of the process noise over time and its statistical distribution have to be assured in order to be able to derive a functional system. In many cases information of such level of detail is too costly to obtain for the intended application or cannot be acquired at all. This is especially true when sensor fusion in dynamical sensor networks like Zig Bee, Bluetooth information exchange among collaborating machines of varying manufacturers are concerned. Here, not necessarily all devices that are able to provide additional information are known. Therefore, they cannot be analyzed and modeled in advance as would be required for the Kalman filter. Thus, an alternative approach that requires very little or even no knowledge about the system components and their collaboration has to be found. Besides the ability to function with very limited system knowledge there are several other demands when dealing with low level multi-sensor fusion. Among these are high robustness, computational efficiency, a high degree of accuracy and ease of use just to name a few. Commonly used methods in this field are the weighted average method, various voting algorithms based on crisp numbers and the already mentioned Kalman filter as well as other more elaborate estimation and inference methods. Another quite popular method in this field is applied in fuzzy techniques, that are utilized in different applications including sensor fusion.

2.2 PROJECT SCOPE

Increasing dependence is being placed on computer-based systems in many applications. For example, several modern commercial aircraft use "flyby- wire" flight control systems with limited mechanical or analogue electronic connection between the pilot and the aircraft control surfaces. In such systems the demand signals from the pilot to the control surfaces are routed through digital flight control computers. Redundancy is required to achieve stringent reliability, availability or safety constraints depending upon the specific application. Redundancy can include information redundancy (for example, through coding techniques), temporal redundancy (for example, retry) or resource redundancy(through replicated or standby sub-systems called "variants"). One approach to arbitrating between variants uses voting algorithms. Voting algorithms seek to mask erroneous results by accepting as correct the result produced by a Majority of variants. This approach assumes that a minority of failed variants produce identical results. The behaviour of voting algorithms where there is no agreement between results is the focus of this study. In some systems, the probability of multiple simultaneous errors is considered negligible. In safety critical systems, however, it is necessary to consider the behaviour of fault-tolerant mechanisms, such as voters, in worst case conditions. Such events may be improbable but can have catastrophic consequences. Thus multiple error scenario should be considered in the dependability analysis.

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-specific '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. Neither voter type is able to cope with uncertainties associated with the voter inputs.A novel voting scheme based on fuzzy set theory[21] which softens the harsh behaviour of the inexact majority voter in the neighbourhood of the 'voter threshold', and handles uncertainty and some multiple error cases in the region defined by the fuzzy input variables[7].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. The percentage of the benign outputs of the majority voter that are successfully handled by the fuzzy voter (resulting in correct outputs) is more than the percentage of those that are unsuccessfully resolved by the fuzzy voter (resulting in incorrect outputs). The fuzzy voter is also a useful voting algorithm when arbitrating between the responses of dynamic channels of control. The definition of the fuzzy rules used in the voter is also likely to play an important part

in the performance of the voter. The fuzzy voter can be invaluable in at least two cases as a substitute for the inexact majority voter in applications in which a small degradation in the safety performance of the system is acceptable at the cost of a large increase of its availability, and a considerable decrease of its benign outputs and when arbitrating between the responses of dynamic channels of control systems (e.g., response of redundant sensors or controllers) which may include some uncertainty.

2. EXISTING SYSTEM

Existing fuzzy voters for controlling safety critical systems and sensor fusion are surveyed and safety performance is empirically evaluated. 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.Optimally selected static fuzzy parameters work only for a particular set of data with the known data ranges.

3. PROPOSED SYSTEM

A dynamic or automatic fuzzy parameter selection method for fuzzy voters is proposed based on the statistical parameters of the local set of data in each voting cycle. Proposed dynamic fuzzy voter is dynamically self configurable.This dynamic fuzzy voter can be used for any systems with little system knowledge and for any input data ranges. The dynamic fuzzy voter can configure itself for any kind of dynamically changing data and it is the first attempt to our knowledge to automate a fuzzy voter. Proposed Dynamic fuzzy voter is giving safety if two of the three modules of the TMR System are error free.

4.1 PROBLEM DEFINITION

Designing automatic fuzzy parameter selection based dynamic fuzzy voter for safety critical systems with limited system knowledge.Optimally selected static fuzzy parameters work only for a particular set of data with the known data ranges.A dynamic or automatic fuzzy parameter selection method for fuzzy voters is proposed based on the statistical parameters of the local set of data in each voting cycle.

4. IMPLEMENTATION & RESULTS

5.1 INTRODUCTION

Existing fuzzy voters for controlling safety critical systems and sensor fusion are surveyed and safety performance is empirically evaluated. The major limitation identified in the existing fuzzy voters is the static fuzzy parameter selection. Optimally selected static fuzzy parameters work only for a particular set of data with the known data ranges. A dynamic or automatic fuzzy parameter selection method for fuzzy voters is proposed based on the statistical parameters of the local set of data in each voting cycle. Safety performance is empirically evaluated by running the static and dynamic fuzzy voters on a simulated triple modular redundant (TMR) system for 10000 voting cycles. Experimental results show that proposed Dynamic fuzzy voter is giving almost 100% safety if two of the three modules of the TMR System are error free.

Dynamic voter is designed in such a way that it can be plugged in and used in any safety critical system without having any knowledge regarding the data produced and their ranges. Fuzzy voter designed 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. 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.

In the static fuzzy voter given in the reference, optimal fuzzy bandwidth is selected based on trial and error method. After some number of trials for the specific input data, optimal values are selected for ‘a’, ’b’ and ‘c’ to decide the fuzzy bandwidth. These optimal values for ‘a’,’ b’ and ’c’ might work only for that input date ranges but this might not work for other input data or dynamically changing data set. So this method cannot be used for those systems with little system knowledge where input data ranges cannot be predicted before only. It is a major drawback with the static method of deciding the fuzzy bandwidth and there are more chances of misclassification of the data. Proposed dynamic fuzzy voter is dynamically self configurable. Dynamic fuzzy bandwidth calculation procedure is given above. This dynamic fuzzy voter can be used for any systems with little system knowledge and for any input data ranges.

The dynamic fuzzy voter can configure itself for any kind of dynamically changing data and it is the first attempt to our knowledge to automate a fuzzy voter. This dynamic fuzzy voter can be just used anywhere in the N-Modular Redundant Systems or Sensor Fusion applications without calibrating it for any threshold values and without knowing anything about those systems.

5.2 PROPOSED METHOD FOR AUTOMATIC DYNAMIC SELECTION OF FUZZY PARAMETERS

Fuzzy bandwidth can be dynamically changed for every voting cycle by setting the values of ‘a’, ‘b’, and ‘c’

1) Calculate the mean of normalized distances in the vector ND formed

$X' = 1/N \sum ndi$

where *ndi* is the i th element in the ND vector and N is the total number of elements (normalized distances) in the ND vector.

2) b = mean (ND)

3) b = x

a = b - standard deviation (ND)

$a = b - \sqrt{\sum}(ndi-x2)/N$

c = b + standard deviation (ND)

$c = b + \sqrt{\sum}(ndi-x)2/N$

'a' to 'c' is the range for medium member function or the fuzzy bandwidth. 0 to 'a' is the high agreement range and beyond 'c' is the low agreement or complete disagreement range. In this automatic dynamic fuzzy parameter selection, local mean of the normalized distances vector for a particular voting cycle is calculated and assigned to parameter 'b'. Standard deviation of the normalized distances subtracted from mean is assigned to fuzzy parameter 'a' and the standard deviation added to mean is assigned to parameter 'c'. Instead of taking the static optimized values for fuzzy parameters, dynamic values based upon the data in a particular voting cycle are assigned to fuzzy parameters.

5.3 STATIC FUZZY VOTER VS. DYNAMIC FUZZY VOTER

In the static fuzzy voter given in the reference, optimal fuzzy bandwidth is selected based on trial and error method. After some number of trials for the specific input data, optimal values are selected for 'a', 'b' and 'c' to decide the fuzzy bandwidth. These optimal values for 'a',' b' and 'c' might work only for that input date ranges but this might not work for other input data or dynamically changing data set. So this method cannot be used for those systems with little system knowledge where input data ranges cannot be predicted before only. It is a major drawback with the static method of deciding the fuzzy bandwidth and there are more chances of misclassification of the data.

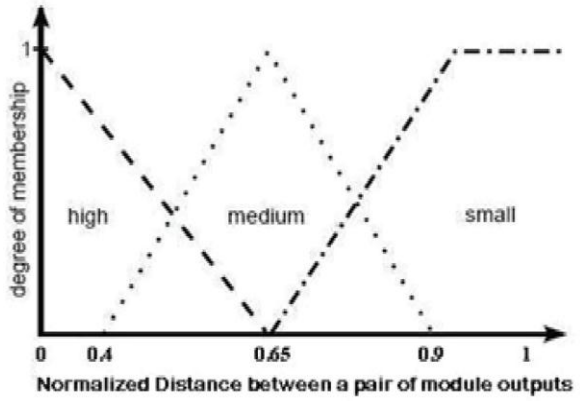

Fig 1 static fuzzy band width

Proposed dynamic fuzzy voter is dynamically self configurable. Dynamic fuzzy bandwidth calculation procedure is given above. This dynamic fuzzy voter can be used for any systems with little system knowledge and for any input data ranges. The dynamic fuzzy voter can configure itself for any kind of dynamically changing data and it is the first attempt to our knowledge to automate a fuzzy voter. This dynamic fuzzy voter can be just used anywhere in the N-Modular Redundant Systems or Sensor Fusion applications without calibrating it for any threshold values and without knowing anything about those systems.

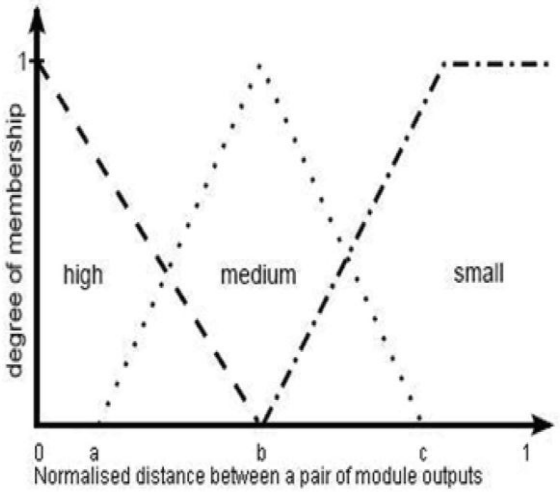

Fig 2 dynamic fuzzy bandwidth

5.4 EXPERIMENTAL SET UP

Test Harness: Test harness for experimentation with voting algorithms is shown in Figure. Cyclic data like sin wave is generated using the equation given.
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. Accuracy or Acceptance Threshold is taken as 0.2 and 0.1 and safety performance is evaluated. 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 parameter Safety as given below:
Safety = 1-(nic/n) (12)
where
nc = Number of correct results given by a voter
nic = Number of Incorrect results given by voter

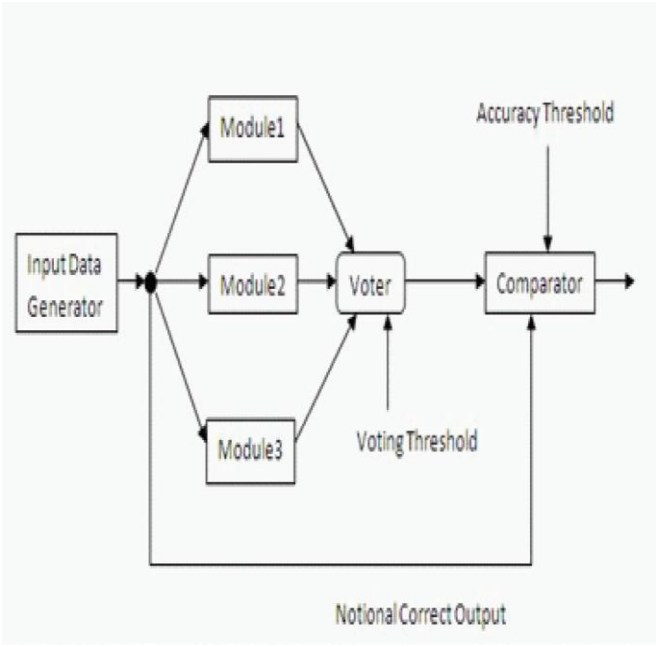

Fig 3 Experimental set up

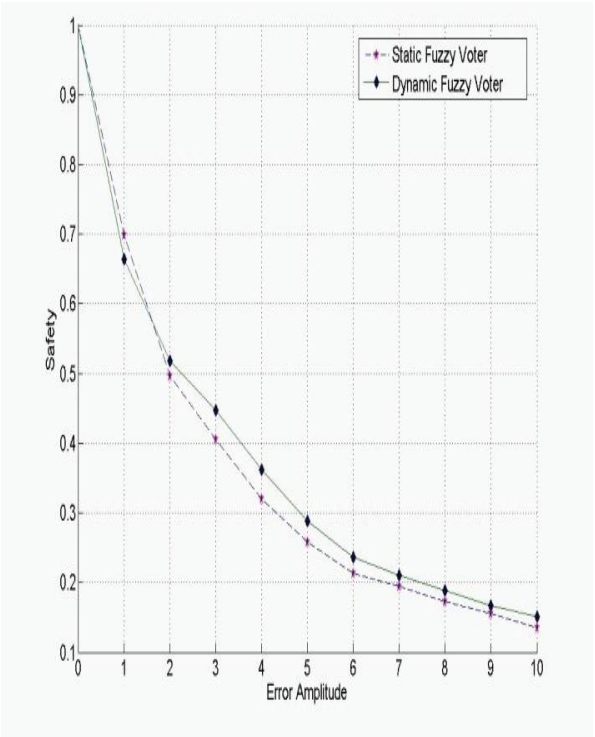

Fig 4 SS1 :Safety performance of static and dynamic fuzzy voters for scenario 1 (accuracy threshold = 0.2)

5.5 Test & Result analysis:

Performance Evaluation of Static and Dynamic Fuzzy Voters

Scenario 1: One error free module and others perturbed with large errors. In this scenario, Two modules are perturbed with the large errors in the error amplitude range [-e +e] and the other module is error free. For each error amplitude e=0 to e=10 and e value incremented by one, voters are run for 10000 voting cycles and how many times each voter is giving the correct output (safety performance) is recorded. From voting cycle 0 to 3999, module1 is error free and module2 and module3 are perturbed with the error in the range [-e +e]. From Voting cycle 4000 to 6999, module3 is error free and the other two modules have the error in the range [-e +e]. For the remaining 7000 to 10000 voting cycles, module2 is error free and other modules are perturbed with the errors in the range [-e +e]. Safety performance is plotted as shown

Static voter takes fuzzy parameters a=0.4, b=0.65 and c=0.9 for all the voting cycles. Dynamic voter detects outliers based upon the local information for each voting cycle data values. For the module output data set in each voting cycle, mean value of the normalized distances of the module pairs is calculated. [mean-std mean+std] is taken as the range for medium fuzzy membership function (std – standard deviation). [0 a] is complete agreement range taken as high membership function and beyond 'c' is the complete disagreement range taken for low membership function. Fuzzy parameters 'a', 'b' and 'c' are dynamically changed for each voting cycle.

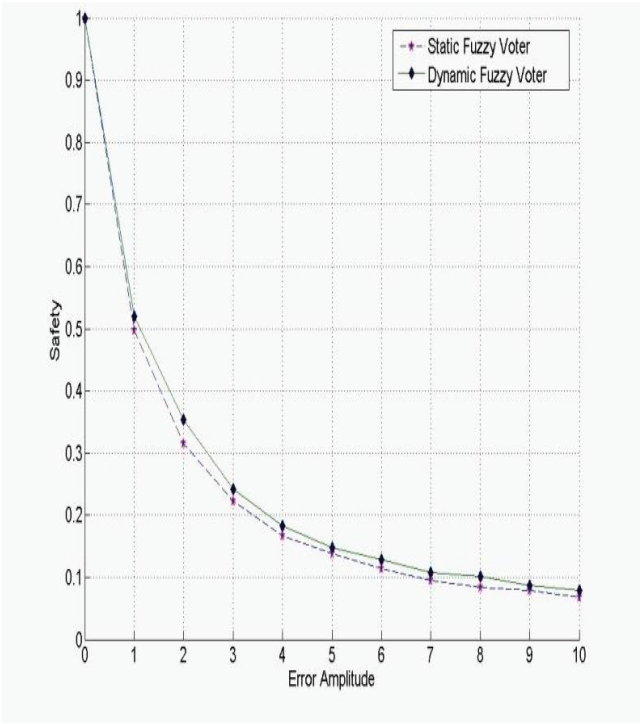

Fig 5 SS2:Safety performance of static and dynamic fuzzy voters for scenario 1 *(acceptance threshold = 0.1)*

Dynamic fuzzy voter is giving better safety performance for large error amplitude errors since it is taking the threshold values for each local data set. Statically selected values are able to efficiently find the outliers up to [-2 +2] error range but for large error amplitudes safety performance is low compared to the dynamic fuzzy voter as the fixed threshold values failed to find the outliers for some data sets in some voting cycles.

Major limitation in this method is wrong modules with error may satisfy the majority consensus and they will be nearer to the mean and the correct module without error is eliminated assuming it as an outlier

Scenario 2: One error free module and others perturbed with small errors. In this scenario, Two modules are perturbed with the small errors in the error amplitude range [-e +e] and the other module is error free. For each error amplitude e=0 to e=2 and e value incremented by 0.2, voters are run for 10000 voting cycles and how many times each voter is giving the correct output (safety performance) is recorded. From voting cycle 0 to 3999, module1 is error free and module2 and module3 are perturbed with the error in the range [-e +e]. From Voting cycle 4000 to 6999, module3 is error free and the other two modules have the error in the range [-e +e]. For the remaining 7000 to 10000 voting cycles, module2 is error free and other modules are perturbed with the errors in the range [-e +e]. Safety performance is plotted as shown

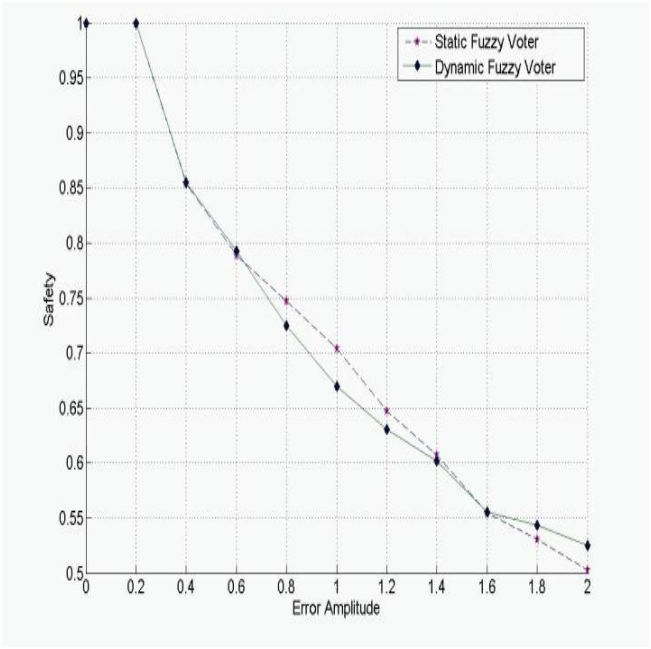

Fig 6 SS3: Safety performance of static and dynamic fuzzy voters scenario 2*(accuracy threshold = 0.2)*

For small errors, static fuzzy voter is giving slightly better safety performance as the selected fixed values for fuzzy parameters a=0.4,b=0.65 and c=0.9 are able to detect the outliers correctly. But the same fuzzy parameter values may not work for the other set of data in some other range or dynamically varying data and it is difficult to select optimum values for the fuzzy parameters where we need to go for the automatic dynamic selection of fuzzy parameters.

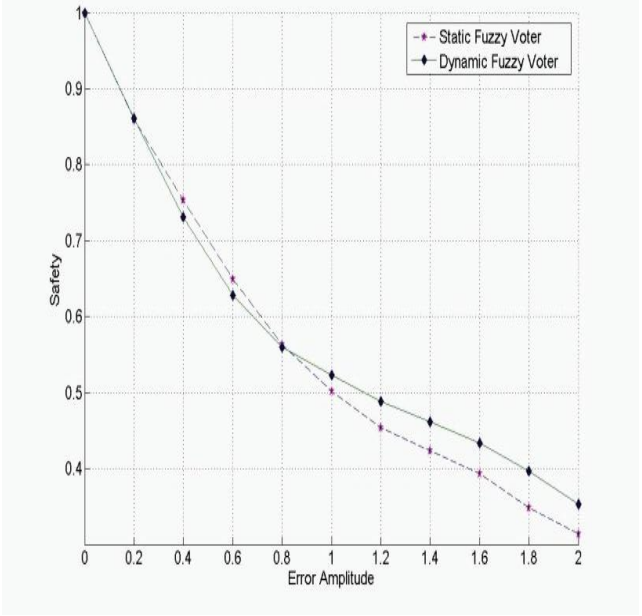

Fig 7 SS4: Safety performance of static and dynamic fuzzy voters for scenario 2 (accuracy threshold = 0.1)

Scenario 3: One error free module and others perturbed with large errors. In this scenario, two modules are perturbed with the large errors and the other module is error free. For each error amplitude e=0 to e=10 and e value incremented by one, voters are run for 10000 voting cycles and how many times each voter is giving the correct output

(safety performance) is recorded. From voting cycle 0 to 3999, module1 is error free and module2 is perturbed with the error in the range of [-e +e] and module3 is perturbed with the error in the range [-2e +2e]. From voting cycle 4000 to 6999, module3 is error free and module1 is perturbed with the error in the range [-e +e] and module2 is perturbed with the error in the range [-2e +2e]. For the remaining 7000 to 10000 voting cycles, module2 is error free module1 is perturbed with the error in the range [-2e +2e] and module3 is perturbed with the error in the range [-e +e]. Safety performance is plotted as shown

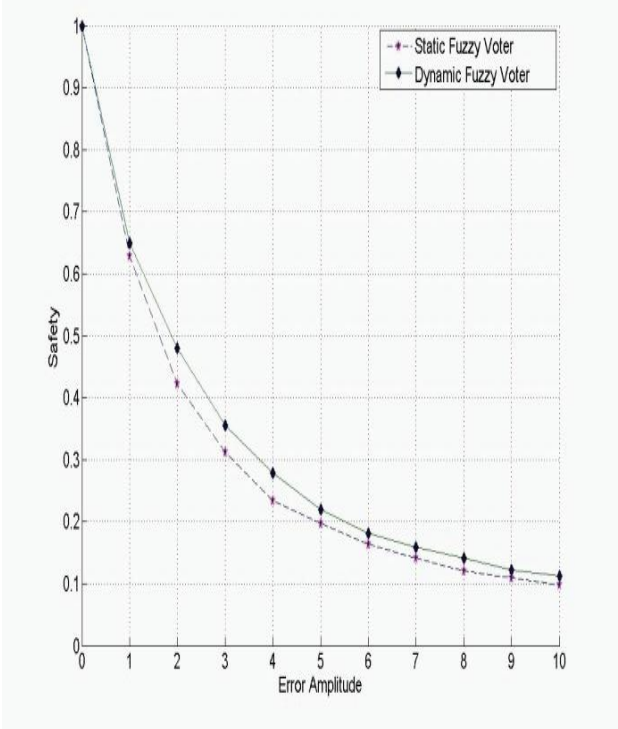

Fig 8 SS5: safety performance of static and dynamic fuzzy voters for scenario 3 (accuracy threshold = 0.2)

Dynamic fuzzy voter is giving better safety performance compared to static fuzzy voter since it is calculating the local mean for each data set and deciding the fuzzy bandwidth.

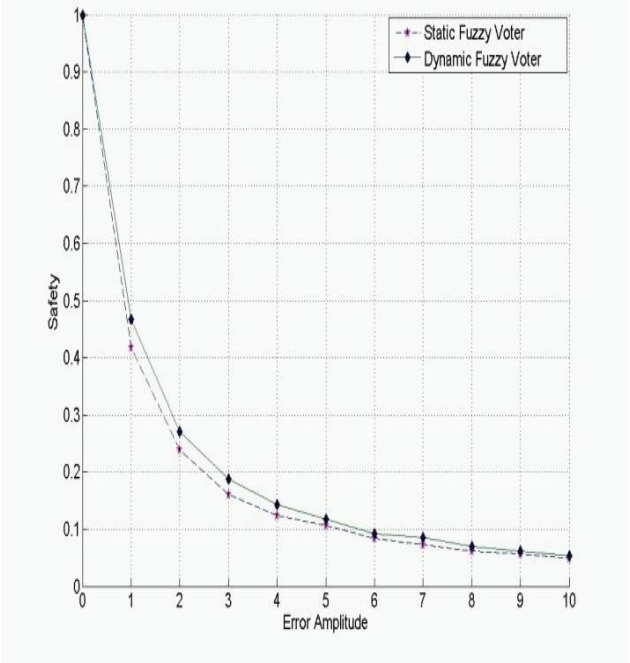

Fig 9 SS6 :safety performance of static and dynamic fuzzy voters for scenario 3 (accuracy threshold = 0.1)

Scenario 4: One error free module and others perturbed with small errors. In this scenario, two modules are perturbed with the large errors and the other module is error free.

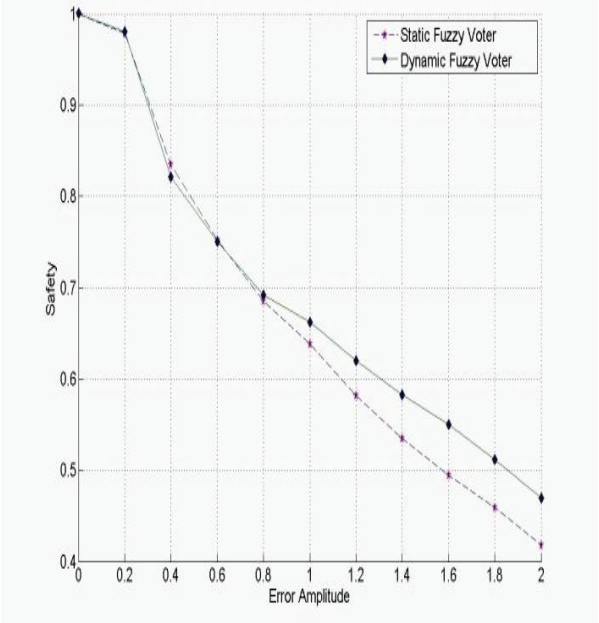

Fig 10 SS7:Safety performance of static and dynamic fuzzy voters for scenario 4 (accuracy threshold = 0.2)

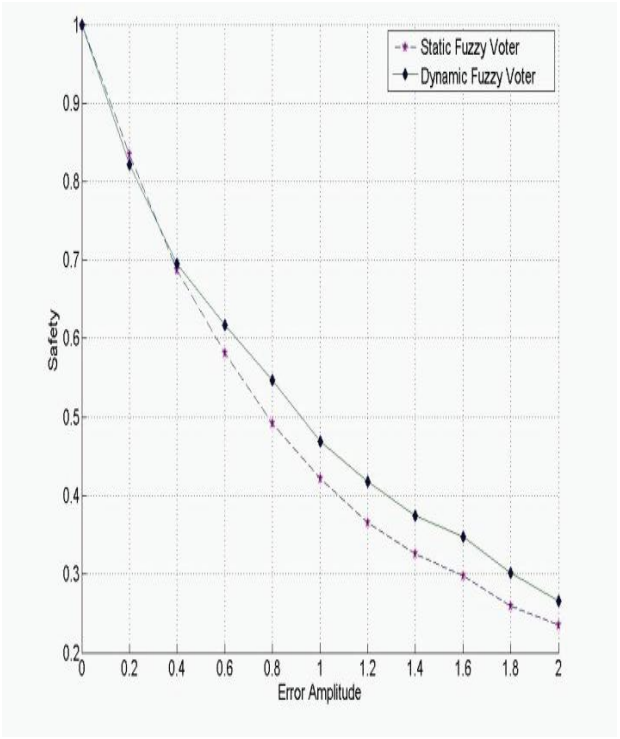

Fig 11 SS8: Safety performance of static and dynamic fuzzy voters for scenario 4(accuracy threshold = 0.1)

Scenario 5: Two error free modules and the other module perturbed with large errors. In this scenario two modules are error free and the other module is perturbed with the large errors.

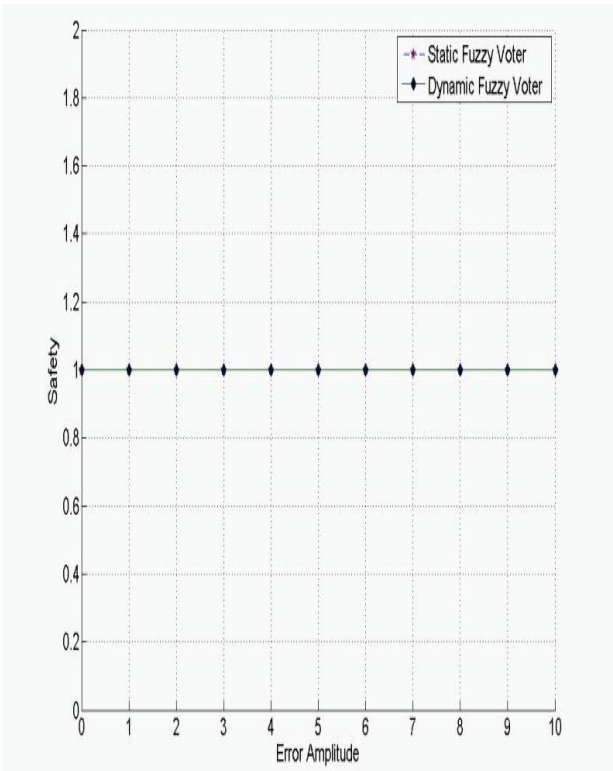

Fig 12 SS9:Safety performance of static and dynamic fuzzy voters for scenario 5 (acceptance threshold = 0.1)

Both the static and dynamic fuzzy voters are giving 100% safety. Since two modules are error free, majority consensus is satisfied and the other module's output is considered as the outlier and eliminated from contributing to the final voter output. The major limitation in this approach is, if the modules which have error satisfy the majority consensus, other module which produce the correct output is considered as the outlier and eliminated from contributing to the voter output. Dynamic fuzzy voter gives 100% correct outputs when two modules are error free. Dynamic fuzzy voter detected the erroneous module as an outlier as it is far from the other module outputs and avoided from contributing to the voter output.

5. CONCLUSION & FUTURE WORK

A self configurable dynamic fuzzy voter using statistical parameters is designed and safety performance is compared with the existing static fuzzy voter. In the existing static fuzzy voter, optimal fuzzy parameters are selected globally which are fixed throughout all the voting cycles, which is not a better idea. This static method is useful only in the situations where the data ranges are known based upon which optimal fuzzy parameters are selected to decide the fuzzy bandwidth. Dynamic fuzzy voter can dynamically configure itself for any data of any ranges as it decides the fuzzy parameters based upon the local data of a particular voting cycle, using statistical parameters like mean and standard deviation. This dynamic fuzzy voter can be used in any safety critical system without having much knowledge about the system, data and ranges of data. The safety performance of the static and dynamic fuzzy voters are compared empirically by running for 10000 voting cycles on a TMR simulator system. The dynamic fuzzy voter is given almost100% safety if two modules are error free and giving better safety performance than the static fuzzy voter if one module is error free and two modules have errors. Though there is no great improvement in the safety performance with the dynamic method, it is useful method since it automates the fuzzy voting technique.

One limitation identified with the dynamic fuzzy voter is the approach used for outlier detection. If two modules of TMR system, which have errors, wrongly or coincidentally satisfy the majority consensus and then the other module output which is actually correct is considered as an outlier. Hence there is a need to consider the module reliability history also apart from the statistical parameters, in the calculation of the fuzzy parameters and this remains the future work. This work can be extended to design Interval type fuzzy voter using fuzzy sets and proposed automatic parameter selection method may be applied to increase the safety performance. There is also scope for designing neuro-fuzzy voting system.

6. REFERENCES

[1]Chen, L., & Avizienis, A. (1978). N-Version programming: a fault-tolerance approach to reliability of software operation. In Proceedings of the 8th Annual International Symposium on Fault-Tolerant Computing Systems (pp. 3-9).

[2]Kumar, S. P., Ramaiah, P. S., & Khanaa, V. (2011). Architectural patterns to design software safety based safety-critical systems. In Proceedings of the International Conference on Communication, Computing & Security (pp. 620-623).

[3]Laprie, J.-C. (1985). Dependable computing and fault-tolerance: concepts and terminology. In Proceedings of the IEEE 15th Annual International Symposium on Fault-Tolerant Computing Systems (pp. 2-11).

[4]Latif-Shabgahi, G., Bass, J. M., & Bennett, S. (2004). A taxonomy for software voting algorithms used in safety-critical systems. *IEEE Transactions on Reliability*, *53*(3), 319–328. doi:10.1109/ TR.2004.832819

[5]Blough, D. M., & Sullivan, G. F. (1990). A comparison of voting strategies for fault-tolerant distributed systems. In Proceedings of the IEEE 9th Symposium on Reliable Distributed Systems (pp. 136-145).

[6]Latif-Shabgahi, G., & Bennett, S. (1999). Adaptive majority voter: a novel voting algorithm for real-time fault-tolerant control systems. In Proceedings of the 25th Euromicro Conference (Vol. 2, pp. 113-120).

[7]Lorczak, P. R., Caglayan, A. K., & Eckhardt, D. E. (1989). A theoretical investigation of generalized voters for redundant systems. In Proceedings of the Nineteenth International Symposium on Fault- Tolerant Computing (pp. 444-451).

[8]Latif-Shabgahi, G., Bass, J. M., & Bennett, S. (2001). History-based weighted average voter: a novel software voting algorithm for fault-tolerant computer systems. In Proceedings of the Euromicro Conference on Parallel, Distributed, and Network- Based Processing (pp. 402-409).

[9]Singamsetty, P., & Panchumarthy, S. (2011). A novel history based weighted voting algorithm for safety critical systems. Journal of Advances in Information Technology, 2(3), 139–145. doi:10.4304/ jait.2.3.139-145

[10]Das, M., & Battacharya, S. (2010). A modified history based weighted average voting with soft-dynamic threshold. In Proceedings of the International Conference on Advances in Computer Engineering (pp. 217-222).

[11] C.-L. Lei and H.-T. Liaw, "Efficient Parallel Algorithms for Finding the Majority Element," Journal of Information Sceince and Engineering, vol. 9, pp. 319-334,1993.

[12] Kanekawa, K**.,** Maejima, H., Kato, H., and Ihara, H. (1989). "Dependable On-Board Computer Systems with a New Method: stepwise Negotiated Voting", Digest of papers IEEE 19th Int. Ann. Symp. on Fault-Tolerant Computing Systems, Chicago, June 1989, pp. 13-19.

[13] Bass, J. M. (1995). "Voting in Real-Time Distributed Computer Control Systems*"*, PhD thesis, Department of Automatic Control and System Engineering, The University of Sheffield, Sheffield, UK.

[14]Johnson, B. W. (1989). Design and analysis of fault-tolerant digital systems. Reading, MA: Addison- Wesley.

[15] Gersting, J. L**.,** Nist, R. L., Roberts, D. B., and Van Valkenburg, R. L. (1991). "A Comparison of Voting Algorithms for N-Version Programming", Digest of papers IEEE 24th Ann. Hawaii Int. Conf. on Systems Sciences, Vol. 2, pp. 253-62.

[16] Latif-Shabgahi, G. Bass, J. M., and Bennett, S. (1998b). "Component-Oriented Voter Model for Dependable Control Applications", Proc. of the Int. Conference on Control'98, University of Wales, Swansea, UK, Sept. 1998

[17] N. Schmitz, J. Koch, M. Proetzsch, and K. Berns. Fault-tolerant 3D localization for outdoor vehicles. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 941–946, Beijing, China, October 9-15 2006.

[18]G.J. Grindey, S.M. Amin, E.Y. Rodin, and A. Garcia-Ortiz. Kalman filter approach to traffic modeling and prediction. In Intelligent Transportation Systems, volume 3207, pages pp. 234–240, Pittsburgh, PA, USA, October 1997. SPIE.

[19] V. Cevher and J. H. McClellan. Sensor array calibration via tracking with the extended kalman filter. In ICASSP 01 Proceedings of the Acoustics, Speech, and Signal Processing, pages pp. 2817–2820, Washington, DC, USA, 2001. IEEE Computer Society.

[20] G.K. Pasricha. Kalman filter and its economic applications. Technical report, University of California, Santa Cruz, CA, USA, October 2006.

[21]Latif-Shabgahi, G., & Bennett, S. (1999). Adaptive majority voter: a novel voting algorithm for real-time fault-tolerant control systems. In Proceedings of the 25th Euromicro Conference (Vol. 2, pp. 113-120).