Contents lists available at Mapta Publishing Group

Vol. 1, No. 2, August, 2018

Mapta Journal of Electrical and Computer Engineering (MJECE)

journal homepage: http://maptapublishing.com/index.php/mjece

ISSN: 2631-5718

Comparative Parallel plurality Voting Algorithm for Fault-Tolerant Medical Robot

Sharareh Jareh[a,*], Abbas Karimi[b]

[a] Department of Computer Engineering, Faculty of Engineering, Islamic Azad University, Ashtian Branch, Ashtian, Iran
[b] Department of Computer Engineering, Faculty of Engineering, Islamic Azad University, Arak Branch, Arak, Iran

Article Info

Article history:

Received Jul 7th, 2018
Revised Aug 16th, 2018
Accepted Sep 22th, 2018

Keyword:

Comparative parallel plurality
Voting algorithm
Triple modular redundancy
Fault tolerant

Abstract

Safety Critical systems from which medical robots if get stopped or get defected because of their faults they may result in death. This paper provides a fault tolerant multilayer framework for safety critical systems. Triple Modular redundancy method (TMR) uses comparative parallel plurality voter in sensor section. A TMR voter circuit that its work is tolerating the faults in in sensor section will choose one of three exits of sensor for processing. The performance of voting algorithm is calculated by a set of fault injection experiments. Up to now different algorithms are provided for voting while in this paper comparative parallel plurality algorithm is provided because of high speed of processing and low time complexity in comparison with other algorithm.

1. Introduction

We can say that robot is a multitasking electro-mechanic device or programmable smart software which does different tasks. There are different types of robots which are invented in recent decades. Variety of robots like mobile robots, industrial robots, service robots, vision robots, military robots, automatic multipurpose robots, health care robots, educational robots, toy robots, washer robots, cleaner robots, gardening robots, and laundry robots. In this paper, we have focused on medical robots and the robots which help us for managing the health of patient. Variety of surgery and medical systems are developed in different practical domains and they do more accurate surgeries [1]. The automatic health care systems involves many trends such as giving services to patient; taking care of patient and giving right dosage of drug in determined time [2], biologic cyber integrated strong systems along with body of human which are designed to control important parameters, can improve health and quality of life considerably [3, 4]. The work atmosphere of the medical robots is getting close to this domain. This case needs to hardware and software components which are provided considering the safety requirements in all the stages of development of system [5, 6]. In modern medical robot systems the different surgery assignments must be implemented with different sensor and stimulation abilities. Complexity and scale of systems is growing increasingly and meet the operational requirements (i.e. actual sensor time stimulation, different sensor feedbacks of sensor, controlling the robots in high frequency) and non-operational requirements (i.e. safety, efficiency, high processing speed, and reliability). In large scale and highly complicated system, the trend of obtaining the high speed of system gets much more complicated. In medical robots such issues have not been the center of attention. The computer of robot uses sensor for determining the faults between intended and felt of common situation, speed, and forces. If a sensor is damaged or faces with fault remains undiscovered, the controller will develop wrong information about the points and claws and consequently it will be faulty considerably [7-10]. Different methods of voting are described in scientific works: the examples are plurality, Propagation, Median, Weighted Average, Predictor, negotiator step by step voting, maximum probability Voters, and Parallel Average Voting [11-14]. For comparing the importance of voting algorithms different performances and criterion can be defined (for example, reliability, availability, safety [15-18]. plurality voter is one of the several algorithm used in fault-tolerant control system. The main advantage of this voter is availability [5, 19-21] and high reliability [22]. In this paper parallel algorithms are used in

* Corresponding author: sharareh.jareh@gmail.com

medical systems to provide a new generation of voting called parallel plurality voter (ppv) for increasing the speed of processing and decreasing the time complexity of algorithm [23].

2. Triple modular redundancy

A fault that occurs in medical programs may result in dangerous situation. If an individual sensor is used and it is faulty by attention to noise, the system gets failed and fault occurs. Therefore, N modular redundancy and or N version programing with voting method is used for covering the fault in a faulty environment. There is different architecture pattern in which the modules of redundancy with voter in safety critical systems is used [7]. Triple modular redundancy (TMR) is a fault tolerance using three module or sensor which work together and have similar input signals. The output of TMR is one of the selected three outputs by voting method. The redundancy sensors for system safety can get possible by different methods. The simplest method use two sensors, the initial sensor and support sensor. The complicated sensors of redundancy can be obtained by many sensors in a matrix (sensor array). In this study, redundancy sensors are improved using TMR [24].

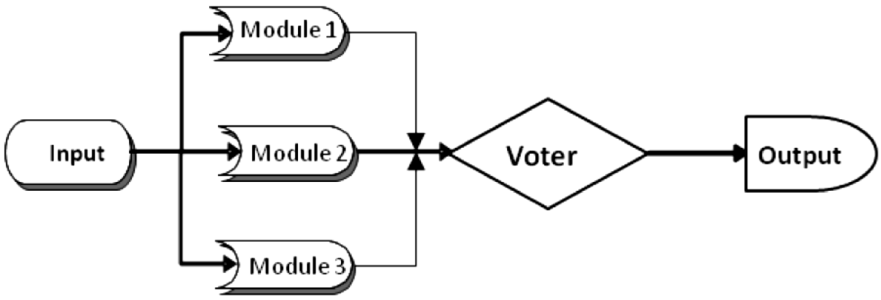

Figure 1. Triple modular redundancy system

In fault covering approach the hardware modules or software versions are repeated and then voting is used for selecting between results to cover the effect of two or more performing time fault. N version programming and N modular redundancy are the known fault covering methods. The simplest form of N modular redundancy approach is triple modular redundancy model. Here, three module works together in parallel and three outputs is calculated by voter and depending on the implementing voting method in voter the output is presented. The TMR diagram is shown in figure 1.

2.1. Fault-tolerant controller of medical robot

The fault-tolerant controller of robot system can be divided in two layers: one servo and a connector monitor. The servo layer of provided framework includes robot and robot controlling computer. Robot controller calculated the required momentum to applying to each motor to move the robot from a given situation to the next intended situation. The information about present position or the speed of the robot claws to controller get improved by internal sensors of robot. The computer of robot use the sensors for sending the position of claws, speed and/ or force and determining the faults between intended values. All of the sensors usually have a form of noise. If a sensor is damaged and the fault is remained undiscovered, the controller get wrong information about claws and result like before will be faulty considerably. Therefore, the intactness of sensors is very important for robots for completing their assignments and consequently focusing on algorithms for providing the possible fastest fault tolerance of sensor. The connection between robot unit and control unit includes sensors and A/D and D/A transformers. Sensors samples from the present position of mechanical part and turns it to analog signals (voltage or current). The A/D transformer turns the signal and sends it to control unit. The intended path and other control parameter in supplied to control unit by user. The control parameters determine the characteristics of robot unit. The control unit considering the present situation of robot and intended situation calculates the required force. The calculated value gets turned to analog signal and the signal stimulates the motor to move the robot [25]. The middle layer includes the robot programmer and fault tolerance routines. Fault tolerance manager (FTM) is a component which provides the fault tolerance through robot and middleware application. In other word, FTM manages the things in middleware and the use of fault will occur in middleware program. In this point the fault tolerance is used by the time that the computer system or its elements is designed. Fault tolerance makes system able to avoid service cancelling using elements change of initial approaches. Fault tolerance services are presented by software but its embedded form may be in hardware. Fault tolerance software implementation makes programmer able to investigate important data in time determining points. Fault tolerance Robot controller uses physical redundancy and different practical software to providing vast domain of software and hardware fault tolerance. Fault tolerance methods developed for making reliable robotic control systems in this research can be sued in programs which

need high degree of safety like services and investigation in free space of station, keeping and cleaning of the trash in nuclear facilities, patient monitoring, and handling the duties in medical equipment assignments [26, 27].

2.2. Test model

The test model is used for simulation. The below schematic shows the results obtained from TMR system based on majority voter which are presented in figure 2:

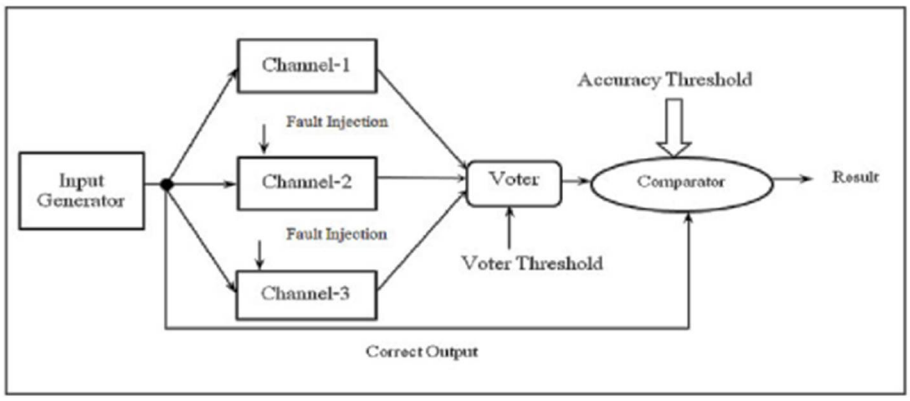

Figure 2. Test model

The fault is injected in two channels and the input data producer is used for production. The producer of output data produces true outputs in every cycle and the random faults distributed in this output get considered true uniformly and get in. now this damaged outputs get injected to plurality voter and the result of voter get compared with true output. If the output of voter is in threshold of accuracy of conceptual true output, the output of voter is considered true while if the difference is more than the threshold, the output is considered wrong [24].

3. Parallel algorithm

Using the parallel algorithms is one of the most useful methods of showing the problems of sequential plurality voters for large scale systems like public health systems, geographical information systems, data continuity, sensor nets mobile robots, and so on [28]. In this section a new parallel plurality voting algorithm will be presented for medical robots. First, sequential plurality voting will be introduced and then the parallel plurality algorithm will be introduced and described by inspiration of functions of divide and conquer method and algorithm and Brent's theorem.

3.1. Sequential plurality voting

An algorithm for sequential parallel voting in a completely sequential atmosphere for n inputs is provided [29]:

Procedure Exact Sequential Plurality Voting (Totally Ordered)

Sort (ascending order) in place the set of records (xi , vi) with xi as key. Use the end –marker

(xn+1, vn+1) = (∞, 0)

y:=z:=x1;

u:=w:=v1;

For i=2 to n+1 do

While xi=z do

u:=u+vi;

i:=i+1;

End While.

If u>w Then

w:=u;

y:=z;

End If.

z:=xi;

u:=vi;

End For.

End.//end of procedure//

3.2. Parallel plurality voting

Parallel system of medical robot with n processor will be intended and for presenting parallel plurality voting (PPV) in fault tolerant medical robot the following hypotheses will be considered:

- The array A[1…n] with n element includes an...,a2,a1 in which i=1….n and ai is the output unit or module.
- The modules of redundancy, n, are considered as force 2.
- The divide and conquer is used for algorithm implementation.
- For improve of the algorithm, the number of required processors is equal to number of sub-arrays for example it is considered as P.

Optimum Pseudo-code of parallel plurality voter (PPV) is provided as following:

Pseudo-code of parallel plurality voter

Procedure Exact Parallel-Plurality Voting *(Totally Ordered Space)*

Input: X is an array of the *n* elements x1,x2,…,xn where $n=2k$.

Output: Return *Y* as the output of the Parallel Plurality Voting (PPV).

Step 1. // Partitioning *X* //

X is subdivided into log *p n*

n = subsequences *Xi* of length *log n* each, where $1 \leq i \leq p$;

Step 2. // Find the number of iterative elements (k) in each partition //

(2.1) $k \leftarrow 1$

(2.2) For i=1 to p do in Parallel

Zi ← X[(i-1)log n +1]

ui ← 1 // set tally for each element //

For j=((i-1)log n +2) To Min{i*log n, n} Do

If Xj=Zi Then

Find iterative Elements in each partition.

ui ← ui +1 // update tally //

Else

k ← k+1

Zk ← Xj

uk ← vj

End If.

End For.

End For.

Step 3.

(3.1) IF k=n Then // n is the number of elements //

“There is no plurality agreement”

Else

Z is subdivided into log *p k*

k = subsequence's *Zi* of

Length *(log k)* each, where *1≤i≤p;*

(3.2) Broadcast *Zi* to processor *Pi*

(3.3) *L← 1*

// Find Repetitive Elements in other partitions and update its tally //

(3.4) For i=0 to ^log p - 1h do

For j=1 to p do in Parallel

IF (j is Odd) Then

Tj ← Z[(j-1)*log k +1]

∀□S ∈□ {((j-1)*log k +2) and Min {2(i+j)*log k,(j-1)k}}

Merge each of pair subsequence (Tj ,Tj+1) and do in Parallel.

IF Tj=Zs Then

uj ← uj +1

Else

W(0,L) ← Tj; W(1,L) ← uj;

L ← L+1; Tj=Zs; uj ← us

End IF.

End IF.

End for.

End for.

// Matrix W is an amalgamation of array X and the number of each element iteration(tally) //

Step4.

(4.1) For i=1 to (L/log L) do in Parallel

Y(0,i) ← W(0, (i-1)*log L +1)

Y(1,i) ← W(1, (i-1)*log L +1)

For j=((i-1)*log L +2) To Min{i*log L, L} do

If Y(1,j)> Y(1,i) Then

Y(1,i) ← Y(1,j)

Y(0,i) ← Y(0,j)

End If.

End For.

End For.

// Return the Result if PPV exists //

(4.2) For i=0 to $\left\lceil \log \frac{L}{\log L} \right\rceil - 1$ do

For $j = 1$ to $\left(\frac{L}{\log L}\right)$ do in parallel

For j=1 to (L/log L) do in Parallel

If (j is Odd) Then

If Y(1,(j-1)*log L +1))> Y(1,(j*log L +1)) Then

Result ← Y(0,(j-1)*log L +1))

Else

Result ← Y(0,(j*log L +1))

End IF.

End IF.

End For.

End For.

End.

In the next section four steps is tested for PPV algorithm and is compared with time complexity of sequential algorithm. Optimum parallel plurality voting will be described with least time complexity and least number of processors respectively. Moreover, the Brent theorem is used for dividing the input elements to predetermined groups [30].

4. Results and discussion

In this section comparison between time complexities of sequential plurality voting algorithm and parallel plurality voting algorithm will be provided. For doing it, Ts(n) is defined as a function of time of sequential plurality voting algorithm running and Tp(n) is defined as a function of running time of parallel voting algorithm in which P is the number of sensors. As it is mentioned in sequential plurality voting subsection, sequential plurality needs to Ts(n)=O(nlogn) time complexity while the parallel algorithm consumes constant O(1) time for dividing the X array to P sub-array. "partitioning X" indicates the maximum length of "Longn" in step one. Finding the repetitive elements in X array and so on, the value of counter k in each section, and P, need to O(logn) time complexity in step 2. In step 3, if the value of k is equal to number of inputs of X, the algorithm do not have more result since it is not reached to plurality agreement. The time complexity of step 3.1 is O(1). If the value of k opposites n, the number of counter elements will be arranged and in matrix Z in step 2 using step 1 method will be saved to P partition. Time complexity of processes in step 3.2 and 3.3 and 3.4 are O(1), O(1), and O(logk) respectively which are less than or equal to O(logn). The result of step 3 is shown in matrix W in which the row zero addresses the input value of algorithm and row one shows the label of each element in partition. In step four, the next row in several partitions must be compared for finding the repetitive elements. (step 4.1) finally, the plurality element is considered in step 4.2. processes in step 4 altogether need to O(log1)<O(logk)<O(logn) so:

Tp(n)= O(1)+O(log n)+ O(log k)+ O(log l)

We conclude that complete complexity of parallel plurality voting for $l \leq k \leq n$ is:

Tp(n)=O(log n)

Considering the required running time of counter and number of processors, the cost and time complexity of algorithm will be optimum.

5. Conclusions

As it is mentioned in the results and discussion section the running time of sequential algorithm is Ω(n log n) while it is O(log n)for parallel algorithm. Therefore, complexity of parallel plurality voting algorithm is less than sequential and the speed of processing in medical robots will be high considerably.

References

[1] R.H. Taylor, A. Menciassi, G. Fichtinger, P. Fiorini, P. Dario. Medical robotics and computer-integrated surgery. Springer handbook of robotics. Springer2016. pp. 1657-84.

[2] J.B. Prajapati, N.K. Modi, R.S. Patel. A Scrutiny of Automated Healthcare System with SFT. International Journal of Computer Science and Communication Networks. (2011) 305-9.

[3] A. Facchinetti, G. Sparacino, C. Cobelli. Modeling the error of continuous glucose monitoring sensor data: critical aspects discussed through simulation studies. SAGE Publications2010.

[4] F. Koushanfar, M. Potkonjak, A. Sangiovanni-Vincentell. Fault tolerance techniques for wireless ad hoc sensor networks. SENSORS, 2002 IEEE. IEEE2002. pp. 1491-6.

[5] D.M. Blough, G.F. Sullivan. A comparison of voting strategies for fault-tolerant distributed systems. Proceedings Ninth Symposium on Reliable Distributed Systems. IEEE1990. pp. 136-45.

[6] A.G. Khiabani, R. Babazadeh. Design of robust fractional-order lead–lag controller for uncertain systems. IET Control Theory & Applications. 10 (2016) 2447-55.

[7] G. Latif-Shabgahi, A. Hirst, S. Bennett. A novel family of weighted average voters for fault-tolerant computer control systems. 2003 European Control Conference (ECC). IEEE2003. pp. 642-6.

[8] G. Latif-Shabgahi, J.M. Bass, S. Bennett. History-based weighted average voter: a novel software voting algorithm for fault-tolerant computer systems. Proceedings Ninth Euromicro Workshop on Parallel and Distributed Processing. IEEE2001. pp. 402-9.

[9] G. Latif-Shabgahi. A novel algorithm for weighted average voting used in fault tolerant computing systems. Microprocessors and Microsystems. 28 (2004) 357-61.

[10] J.L. Gersting, R.L. Nist, D.B. Roberts, R. Van Valkenburg. A comparison of voting algorithms for n-version programming. Proceedings of the Twenty-Fourth Annual Hawaii International Conference on System Sciences. IEEE1991. pp. 253-62.

[11] L. Chen, A. Avizienis. N-version programminc: A fault-tolerance approach to rellablllty of software operatlon. Twenty-Fifth International Symposium on Fault-Tolerant Computing, 1995,'Highlights from Twenty-Five Years'. IEEE1995. p. 113.

[12] M.D. Krstic, M.K. Stojcev, G.L. Djordjevic, I.D. Andrejic. A mid-value select voter. Microelectronics Reliability. 45 (2005) 733-8.

[13] P. Singamsetty, S. Panchumarthy. A novel history based weighted voting algorithm for safety critical systems. Journal of Advances in Information Technology. 2 (2011) 139-45.

[14] P.R. Lorczak, A.K. Caglayan, D.E. Eckhardt. A theoretical investigation of generalized voters for redundant systems. [1989] The Nineteenth International Symposium on Fault-Tolerant Computing Digest of Papers. IEEE1989. pp. 444-51.

[15] A. Karimi, F. Zarafshan. PAV: parallel average voting algorithm for fault-tolerant systems. International Journal of Advanced Computer Sciences and Applications. 2 (2011).

[16] A. Karimi, F. Zarafshan, A.B. Jantan, S. Al-Haddad. New Parallel N-Input Voting for Large Scale Fault-Tolerant Control Systems. Journal of Electronic Science and Technology. 9 (2011) 174-9.

[17] A. Karimi, F. Zarafshan, A.B. Jantan, S. Al-Haddad. An efficient Markov model for reliability analysis of predictive hybrid m-out-of-n systems. International Journal of Physical Sciences. 6 (2011) 6981-94.

[18] M.Z. Shahrak, S. Mohammadi. Middle east user navigation in online social networks and interactions in e-commerce, an analogy. Advances in Computer Science: an International Journal. 3 (2014) 32-6.

[19] A. Karimi, F. Zarafshan, A. Jantan, A.R. Ramli, M.I.B. Saripan, S. Al-Haddad. Exact parallel plurality voting algorithm for totally ordered object space fault-tolerant systems. Pertanika Journal of Science and Technology. 20 (2012) 89-96.

[20] D. Domis, M. Trapp. Integrating safety analyses and component-based design. International Conference on Computer Safety, Reliability, and Security. Springer2008. pp. 58-71.

[21] M. Zakershahrak, A. Sonawane, Z. Gong, Y. Zhang. Interactive plan explicability in human-robot teaming. 2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN). IEEE2018. pp. 1012-7.

[22] S. Yacoub, X. Lin, J. Burns. Analysis of the reliability and behavior of majority and plurality voting systems. Hewlett-Packard Development Company, LP: Palo Alto, CA, USA. (2002).

[23] M.Y. Jung. A layered approach for identifying systematic faults of component-based software systems. Proceedings of the 16th international workshop on Component-oriented programming. ACM2011. pp. 17-24.

[24] G. Latif-Shabgahi, M. Tokhi, M. Taghvaei. Voting with dynamic threshold values for real-time fault-tolerant control systems. Proceedings of 16th IFAC World Congress, Prague, Czech Republic2005.

[25] J.J. Craig. Introduction to robotics: mechanics and control, 3/E. Pearson Education India2009.

[26] D. Brugali, A. Shakhimardanov. Component-based robotic engineering (part ii). IEEE Robotics & Automation Magazine. 17 (2010) 100-12.

[27] A. Karimi, F. Zarafshan, S. Al-Hadad. Availability analysis of predictive hybrid M-Out-of-N systems. TELKOMNIKA (Telecommunication Computing Electronics and Control). 12 (2014) 437-46.

[28] B. Parhami. Parallel threshold voting. The Computer Journal. 39 (1996) 692-700.

[29] M. Zaker Shahrak. SECURE AND LIGHTWEIGHT HARDWARE AUTHENTICATION USING ISOLATED PHYSICAL UNCLONABLE FUNCTION. (2016).

[30] R.P. Brent. Algorithms for minimization without derivatives. Courier Corporation2013.

[31] A.C.D.-o. Soil, Rock. Standard test methods for one-dimensional consolidation properties of soils using incremental loading. ASTM International2004.