Remark 2: The solution c_i of (8) always exists for $q < m$ since there are m d_{ij} vectors. This result conforms with [1, Theorem 1], but is much simpler.

Step 3:

$$F = \text{diag}\,(s_1, \cdots, s_r) \tag{9}$$

$$T = \begin{bmatrix} c_1 & & 0 \\ & \ddots & \\ 0 & & c_r \end{bmatrix} \begin{bmatrix} d_{11} \\ \vdots \\ d_{1m} \\ \hline \vdots \\ \hline d_{r1} \\ \vdots \\ d_{rm} \end{bmatrix} U' = [\underset{q}{0} : \underset{n-q}{T_2}]\}r$$

$$\triangleq \bar{T}U' \tag{10}$$

$$G = (\bar{T}\bar{A} - F\bar{T}) \begin{bmatrix} I_m \\ \hline 0 \end{bmatrix} C_1^{-1} \tag{11}$$

and determine $[K:P]$ from

$$[K : P] \begin{bmatrix} T \\ \hline C \end{bmatrix} \begin{matrix} \}r \\ \}m \end{matrix} = 0. \tag{12}$$

Remark 3a: It is obvious that the above solution satisfies (3)–(5). More specifically, Steps 0, 1, and (9)–(11) guarantee condition (3) which is equivalent to $TA - F\bar{T} = GCU$.

Remark 3b: In this solution, there is no restriction on the eigenvalues of F. Although Step 1 is for real and distinct s only, it can be easily generalized [2]. In [1], the eigenvalues of F are required to be the same and be different from any eigenvalue of A. The eigenvalues of F should be stable and should also be negative enough so that the observer output can quickly reach its normal steady state value $(=0)$.

Remark 3c: Condition (3) and the stability of F guarantee that $z(t) \Rightarrow Tx(t)$ if $D(t) = 0$. This is the proof that the nonzero output of observer (2) cannot be caused by nonzero $D_1(t)$ to $D_q(t)$ if (5) is satisfied, and can be caused by any other nonzero component of $D(t)$ if (6) is satisfied [1].

Remark 3d: The observer order r is determined to be as small as possible, and to satisfy condition (6) and guarantee the existence of nonzero solution $[K:P]$ of (12). This last condition can be guaranteed if $r > n - m$, from (12). It should be noticed that from Remark 1a, the rows of T are linearly independent of the rows of C. Assuming all rows of T are linearly independent (which is true in most of the cases, even with c_i being fixed in Step 2), from (10), condition (6) can also be guaranteed if $r \geq n - q$. Since $n - q > n - m$, the upper bound of r can be set to be $n - q$.

Because the eigenvalues of F and the corresponding rows of T are ***completely decoupled,*** in actual design, r can be tried from 1 and increasing up to its upper bound ***systematically*** and recursively, and be tried by coded computer program. This property is stated in [1], and is exactly similar to that of a new design procedure for minimal order function observers [3].

Remark 3e: Having derived a clear formula for the observer order r, we can now study the effect of choosing different q, which has not been studied in [1]. From Remark 2, q must be between 1 and $m - 1$. From Remark 3d, the smaller the q, the larger the r needed to satisfy condition (6), generally. Let us set r to be its upper bound ($=n - q$ from Remark 3d); then the upper bound of the total order of $\binom{n}{q}$ observers will be

$$\binom{n}{q} r = \frac{n(n-1)\cdots(n-q)}{q(q-1)\cdots 1}. \tag{13}$$

The order of (13) is at its minimum when $q = 1$.

If we know ***a priori*** that among all n state components, only p $(<n)$ state components are possible to have failure (or only $D_1(t), \ldots, D_p(t)$ can be nonzero), then we will select the combinations of q states only among these p states ($q < \min\{p, m\}$). As a result, only $\binom{p}{q}$ observers are needed to identify and locate the failure among these p state components.

III. CONCLUSION

Mathematically, the solution to the problem formulated in (3)–(6) can be divided into two parts. First, derive each row of the solution T of (3) with complete and explicit freedom (c_i), and use this freedom to satisfy (5). Second, increase the number (r) of rows of T until both (4) and (6) are also satisfied. The main difficulty lies in the first part. Therefore, this problem can be considered as another application of the solution of (3) (see [2]). The simplicity and solution nature of the result of this paper are due to the power of this solution of (3), but do not imply lower academic and practical value as people used to do to this kind of result.

This problem also shows the fundamental importance of (3), which is essential to all basic design problems, old and new (such as this problem and the problem of LTR [4]), in state-space control theory. Compared to the Lyapunov equation ($TA - AT = C$) and the Sylvester equation ($TA - FT = C$) which are more concerned with system analysis only, and to the Riccati equation which is more concerned with the quadratic optimal system design only, (3) is more important as long as the design problems are concerned. Because the analytical and restriction free solution T (with complete and explicit freedom) of (3) is derived only recently [2], we expect more and more important applications to be found based on this solution.

REFERENCES

[1] W. Ge and C. Z. Fang, "Detection of faulty components via robust observation," *Int. J. Contr.,* vol. 47, pp. 581–600, Feb. 1988.

[2] C. C. Tsui, "A complete analytical solution to $TA - FT = LC$ and its applications," *IEEE Trans. Automat. Contr.,* vol. AC-32, p. 742, Aug. 1987.

[3] ——, "A new algorithm for the design of multifunctional observers," *IEEE Trans. Automat. Contr.,* vol. AC-30, p. 89, Feb. 1985.

[4] ——, "On preserving the robustness of an optimal control system with observers," *IEEE Trans. Automat. Contr.,* vol. AC-32, p. 823, Sept. 1987.

[5] R. V. Beard, "Failure accommodation in linear systems through self-reorganization," Man Vehicle Lab., Cambridge, MA, Rep. MVT-71-1, Feb. 1971.

[6] H. L. Jones, "Failure detection in linear systems," Ph.D. dissertation, Dep. Aeronaut. Astronaut., M.I.T., Cambridge, Sept. 1973.

[7] A. S. Willsky, *Automatica,* vol. 12, p. 601, 1976.

[8] X. Singh, X. Hassan, X. Chen, X. Li, and X. Pan, *Proc. IEE,* vol. 130, part D, p. 243, 1983.

[9] R. Isermann, *Automatica,* vol. 20, p. 387, 1984.

[10] K. Janssen and P. M. Frank, "Component failure detection via state estimation," in *Preprints, IFAC 9th World Congr.,* vol. 1, 1984, pp. 147–152.

[11] P. M. Frank and D. Hongy, "Component failure detection via non-linear state observers," presented at the IFAC Workshop on Fault Detection and Safety in Chemical Plants, Kyoto, Japan, 1986.

[12] X. Shoureshi and X. Hostein, "Failure detection and isolation using system structure knowledge and eigenvalue sensitivity," in *Proc. ACC,* Seattle, WA, 1986, pp. 1933–1938.

[13] X. Hodzic and D. Siljak, "Failure detection in large space systems, in *Proc. 7th IFAC/IFORS Symp. Identification and Process Parameter Est.,* York, England, 1986, pp. 273–279.

Performance Analysis of Voting Strategies for a Fly-by-Wire System of a Fighter Aircraft

C. SUBRAMANIAN AND D. K. SUBRAMANIAN

Abstract—Findings of studies on input processing of a digital fly-by-wire system of a fighter aircraft are presented. Objectives were to select a suitable software structure complying with reliability and fault tolerance requirements and to assess its computational load. Ramp and constant input signals with noise were studied based on Monte-Carlo methods. Voting strategies studied and compared include lower-median, upper-

Manuscript received July 18, 1987; revised May 22, 1988. Paper recommended by Associate Editor, B. Sridhar.

The authors are with the Indian Institute of Science, Bangalore, India.

IEEE Log Number 8927761.

median, and weighted average. Execution times and memory requirements of each strategy have also been assessed.

I. INTRODUCTION

There has been considerable research activity in the area of fault tolerant systems which include computer embedded systems required to perform accurately and reliably in spite of failures within the system. A few of the noteworthy architectures in this area are SIFT, FTMP, and JPL–STAR. A recent trend in the design of flight control systems for fighter aircraft is the trade-off between static stability and performance. This is achieved through digital fly-by-wire (DFBW) which has to be fault tolerant for safety reasons [1]–[3]. Studies on input processing of a fault tolerant computing system for DFBW system and the findings of those studies are dealt with in this note. The aim of the study was to select a suitable software structure on an input processing system which meets the capacity requirements along with reliability and fault tolerance requirements.

II. ARCHITECTURE

A hierarchical shared resources quadruplexed architecture has been chosen. The computations are carried out at two levels; input–output processing, and control law computations. Fig. 1 shows the schematic of DFBW computations. The I/O processors acquire data from sensors, process it, and vote on it prior to sending it to control processors.

III. INPUT PROCESSING

The input processing is an integrated function of health monitoring, signal selection, and redundancy management which comprise channel rejection, reconfiguration, and channel reacceptance. Table I gives the strategy for signal selection, channel rejection, and reacceptance based on redundancy levels. The health monitoring is effected through self-monitoring and cross-channel monitoring.

The redundancy with simplicity is expected to provide the necessary degree of reliability and fault tolerance. The effect of redundancy is harnessed through voting. A voter is defined as the following:

$$V_{out}=f(V_1, \cdots, V_i, \cdots, V_n) \tag{1}$$

where V_{out} is voter output and Vi is ith channel input.

The median select is a commonly used voting technique. To avoid the effects of transients due to channel switching, on-line weighted average (WA) voting is preferred. In WA voting,

$$V_{out}=\frac{\sum_{i=1}^{n} W_i V_i}{\sum_{i=1}^{n} W_i} \tag{2}$$

where Wi is the weighting factor of channel i. A typical set of weighting factors [4], [5] of a quadruplex system are

$$W_1=\{1+(V_1-V_2)^2(V_1-V_3)^2(V_1-V_4)^2\}^{-1} \tag{3}$$

$$W_2=\{1+(V_2-V_3)^2(V_2-V_4)^2(V_2-V_1)^2\}^{-1} \tag{4}$$

$$W_3=\{1+(V_3-V_4)^2(V_3-V_1)^2(V_3-V_2)^2\}^{-1} \tag{5}$$

$$W_4=\{1+(V_4-V_1)^2(V_4-V_2)^2(V_4-V_3)^2\}^{-1}. \tag{6}$$

If cross-channel differences are small then

$$V_{out}=\frac{\sum_{i=1}^{n} V_i}{4}. \tag{7}$$

Weighting factor cap ($\hat{W}i$) used for checking tolerance for acceptance is

$$\hat{W}_i=\frac{W_i}{W_1+W_2+W_3+W_4} \quad \text{for } i=1, 2, 3, 4. \tag{8}$$

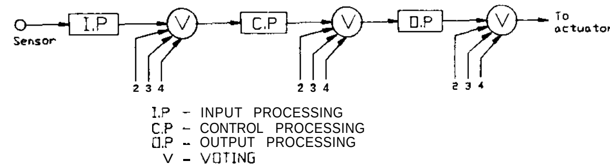

Fig. 1. Schematic of digital fly-by-wire system computations (channel #1).

Strategy / Redundancy level	Signal Selection	Channel Rejection	Channel Reacceptance
Quadruplex	i) Lower Median ii) Upper Median iii) Weighted Average	Tolerance Set for bound Check, Cross comparison	Consistent good behaviour after rejection for a predetermined duration
Triplex	i) Mid-value select ii) Weighted Average	- do -	- do -
Duplex	Average	- do -	Not possible
Simplex	Based on health Monitoring	Tolerance Set for bound check	- do -

Upon the occurrence of a failure in the system, it is reconfigured to a lower level of redundancy. The voted output being the most likely value of data, it is taken as reference for comparison. Two cases on permissible tolerance are considered: 1) 1 percent of voted value; and 2) 20 percent of $\hat{W}i$. Rejection and/or reacceptance of a channel are based on these values and a time factor A t which is the duration of time allowed (say, time for five consecutive samples) for the channel to recover.

IV. SIMULATION

The simulation [5], [6] conducted covers data processing, rejection, and reacceptance and voting. The aims of simulation were 1) to compare different voting strategies and to select the best candidate voter, and 2) to facilitate the selection of a suitable processor based on computational load requirements. A Monte-Carlo method of simulation on PDP-I 1 has been carried out. The modeling of sensor input and noise and simulation of input processing are explained below.

A. Sensor-Input Modeling

Sensor-input signals have been modeled as ramp and constant and are as follows:

$$\text{Ramp signal, } X(t)=a\,\frac{t}{n} \tag{9}$$

where a is a constant corresponding to the slope of ramp, t is a sampling instant represented by a number, and n is the total number of samples during the simulation period T. Typical values are: $a = 3750$ since the signal range (0–5 V) is divided into 4096 levels (max.) and $n = 300$.

$$\text{Constant signal, } X(t)=b \tag{10}$$

where b (=2048) is a constant value.

B. Noise Modeling

The input to processor is $Y(t)$ which is the sensor-input $X(t)$ modulated by the noise $N(t)$ as follows:

$$Y(t)=X(t)+N(t) \tag{11}$$

where $N(t) = K.\text{RAN}\,[S(t), S(t-1)]$, K is variable gain coef., RAN [...] is the random function assumed to be uniformly distributed, and $S(t)$ is the random number generated at time t.

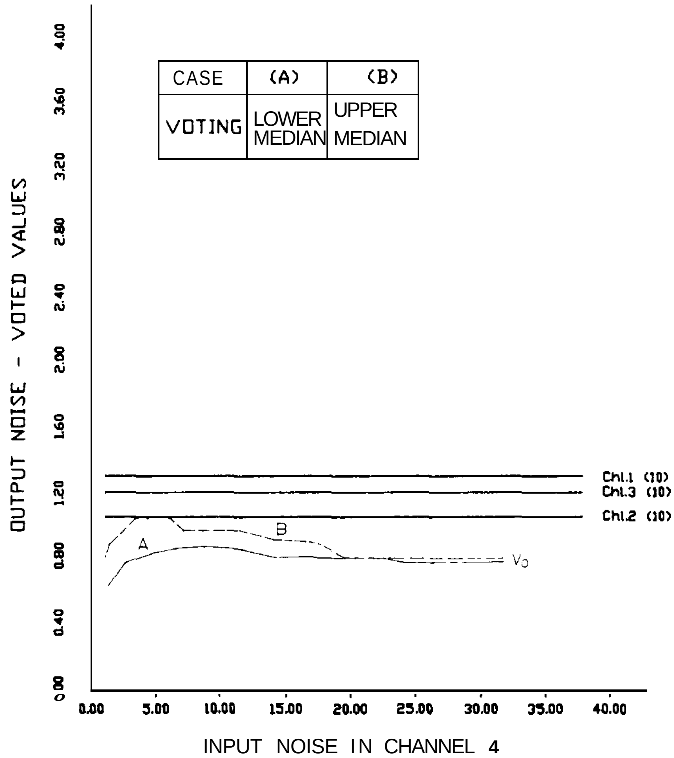

Fig. 2.

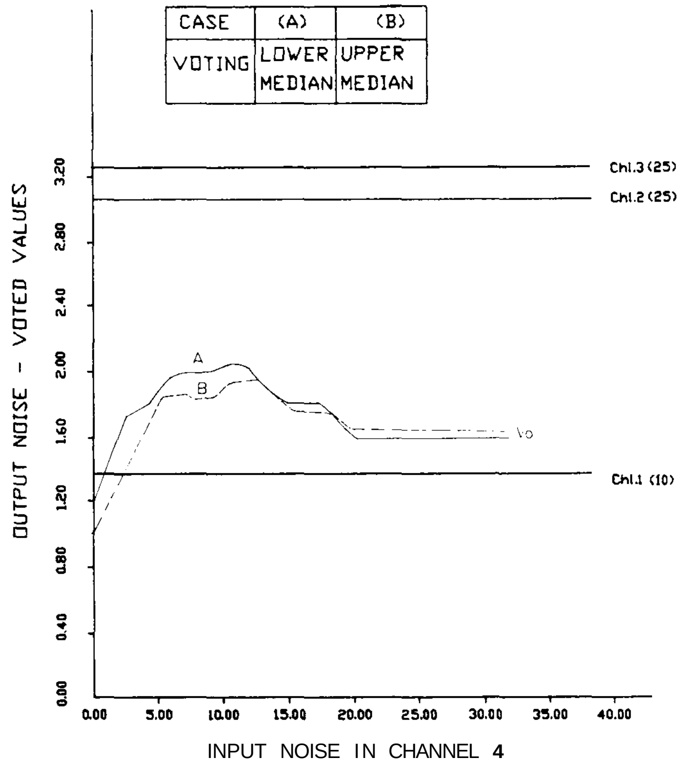

Fig. 3.

C. Process *Simulation*

The activities of each input processor have been simulated through a main program (MP) and its associated routines. There are four modules simulating four input processors. MP gets the sensor number as per the multiplexing scheme and the magnitude of signal sensed at the corresponding time of simulation. Limit checks for boundary and rate of change are conducted on this signal. Then smoothing is carried out by averaging samples as a greater number of samples than required are taken for a fixed duration of time. The same procedure is repeated at all intervals of simulation, and hence for the full range of simulated time.

Then, voting on accumulated values and health monitoring of channels through cross-checking are performed. The voting strategies simulated are lower median (LM), upper median **(UM),** and two cases of weighted average (WA). In order to study the effect of varying noise in a faulty channel (say, no. **4),** the noise levels in other channels are maintained constant and the noise in the faulty channel is varied throughout its range. **rms** value of noise in voted output is plotted against the noise in channel **4.** The noise levels of the other three channels are also plotted for comparison.

The simulation program is written mostly in Pascal with certain portions of it in Fortran and MACRO-11 to take advantage of the facilities available on the PDP-11 system. It exploits the features of RSX-11 M as well as facilities for intertask communication and sharing of common memory. The plotting of results of simulation is done off-line through a Fortran routine. A few of the plots are given in Figs. 2-6.

V. RESULTS

The results of simulation are as follows. For moderate noise levels, lower median is better (Fig. 2). For considerably high noise levels both LM and UM are equally good (Fig. 3). WA(1) gives voted output with less noise compared to WA(2) or LM (Fig. 4). For moderate noise levels, WA(1) and WA(2) are equally good and considerably better than LM (Fig. *5).* For high noise levels, WA(1) and WA(2) only are good and LM is not acceptable (Fig. *6).*

Studies carried out on development systems on memory and execution time requirements of different voting strategies revealed the following:

Memory requirements of algorithm for WA(1) are double that for LM and 25 percent more than that for WA(2) (refer to Col. 3 of Table II).

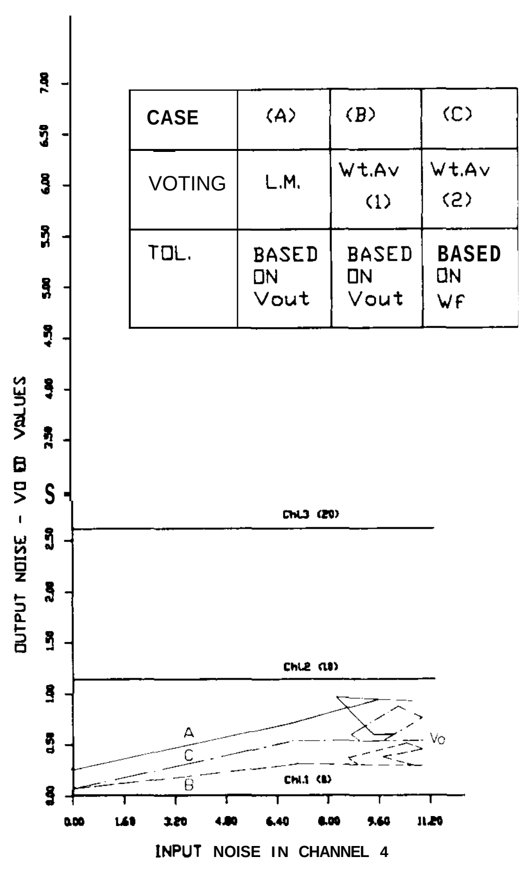

Fig. 4

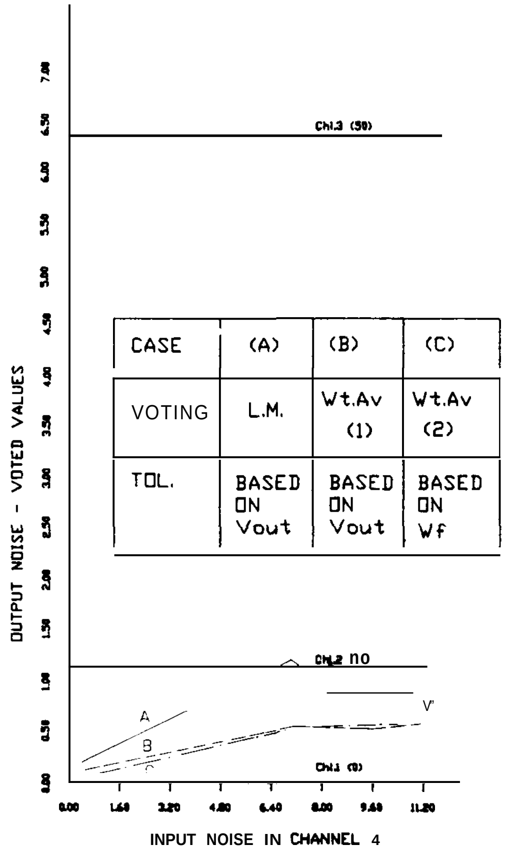

Fig. 5

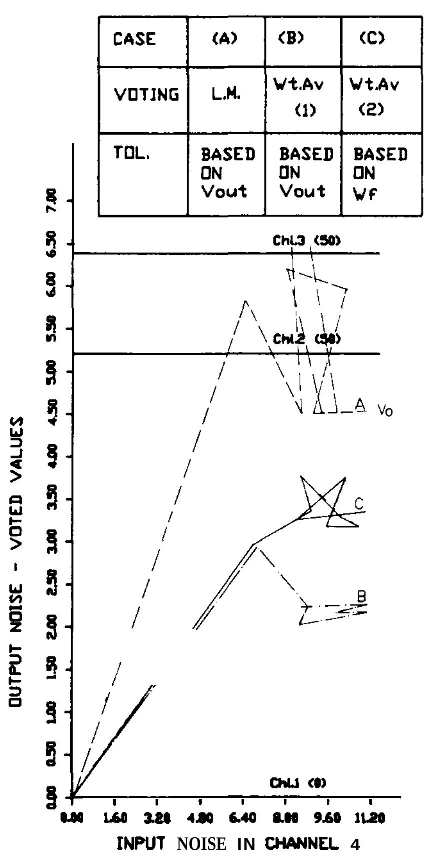

Fig. 6

TABLE II

Sl No	Operation	Memory size (in words of 16 bits)	Approx Exec. Time (microsec)
CASE 1- L.M. SELECTION			
1	Data aquisition	192	240.0
2	Data processing	422	527.5
3	Voting	1082	1352.5
		1706	2120.0
CASE 2- WEIGHTED AVERAGE-WA(1)			
1	Data aquisition	206	257.5
2	Data processing	478	597.5
3	Voting	2386	2982.5
		3070	3837.5
CASE 3- WEIGHTED AVERAGE-WA(2)			
1	Data aquisition	206	257.5
2	Dato processing	478	597.5
3	voting	1564	1955.0
		2243	2810.0

Execution time of WA(1) is double that of LM and 50 percent more than that of WA(2) (refer to Col. 4 of Table II).

VI. Conclusions

The following are the conclusions drawn. Noise in voted output is less than that in any channel. This is due to the fact that the voting rejects the channel with high noise thus affecting filtering. LM is superior to UM although their computational requirements are comparable. WA strategies are superior to median selections. Although the computer overheads of these strategies are much higher than that of LM, the advent of fast and less expensive VLSI technology has overcome these shortcomings. The simulation of input cases is limited to ramp and constant signals and the noise is assumed to be uniformly distributed.

References

[1] M. Pease *et al.*, "Reaching agreements in the presence of faults," *J. Ass. Comput. Mach.*, vol. 21, pp. 228–234, Apr. 1980.

[2] J. E. Potter *et al.*, "Thresholdless redundancy management with arrays of skewed instruments," AGARDorgraph 224 on Integrity in Electron. Flight Contr. Syst., Apr. 1977.

[3] A. K. Caglayan *et al.*, "Systems approach to software fault tolerance," in *Proc. AIAA Comput. Aerospace V Conf.*, Long Beach, CA, Oct. 21–23, 1985, pp. 361-369.

[4] R. B. Broen, "New voters for redundant systems," *Trans. ASME, J. Dynam. Syst. Measurement and Contr.*, pp. 41-45, Mar. 1915.

[5] Anon, "Simulation and experimental studies on I/O processors of flight control computer," Hindustan Aeronautics Limited, Int. Rep. D/ATG/FBW/001, June 1986.

[6] C. Subramanian *et al.*, "Simulation of sensor management in a quadruplexed digital fly-by-wire system," in *Proc. Introductory Workshop on Fault-Tolerant Comput.*, Bangalore, India, July 24–25, 1986, pp. 6.1-6.10.

Comments on "On the Stability of Discrete-Time Sliding Mode Control Systems"

U. KOTTA

Abstract—In the above paper,[1] stability conditions for discrete-time sliding mode control systems are presented, which require the stabilizing control to be upper and lower bounded. The purpose of this correspon-

Manuscript received March 29, 1988; revised August 8, 1988.

The author is with the Department of Control Systems, Institute of Cybernetics, Estonian Academy of Sciences, Akadeemia tee 21, Tallinn 200108 USSR.

IEEE Log Number 8928712.

[1] S. Z. Sarptürk, Y. Istefanopulos, and O. Kaynak, *IEEE Trans. Automat. Contr.*, vol. AC-32, pp. 930–932, Oct. 1987.

0018-9286/89/0900-1021$01.00 © 1989 IEEE