2016 IEEE 4th International Conference on Cyber-Physical Systems, Networks, and Applications

Adaptive Transient Fault Model for Sensor Attack Detection

Minsu Jo*, Junkil Park†, Youngmi Baek**, Radoslav Ivanov†, James Weimer†, Sang Hyuk Son* and Insup Lee†
*Department of Information and Communication Engineering, DGIST, Daegu, Republic of Korea
Email: {Minsu-Jo, son}@dgist.ac.kr
**CPS Global Center, DGIST, Daegu, Republic of Korea
Email: ymbaek@dgist.ac.kr
†Department of Computer & Information Science, University of Pennsylvania, Pennsylvania, USA
Email: {park11, rivanov, weimerj, lee}@seas.upenn.edu

***Abstract*—This paper considers the problem of sensor attack detection for multiple operating mode systems, building upon an existing attack detection method that uses a transient fault model with fixed parameters. For a multiple operating mode system, the existing method would have to use the most conservative model parameters to preserve the soundness in attack detection, thus not being effective in attack detection for some operating modes. To address this problem, we propose an adaptive transient fault model to use the appropriate parameter values in accordance with the change of the operating mode of the system. The benefit of our proposed system is demonstrated using real measurement data obtained from an unmanned ground vehicle.**

I. Introduction

As Cyber Physical Systems (CPS) have been widely used in various areas such as Intelligent Transportation Systems (ITS), ensuring the safety of such CPS becomes imperative. Modern CPS are equipped with multiple sensors. Some of the sensors may be vulnerable to sensor attacks such as sensor spoofing [1]–[3]. The attacked sensors may provide malicious data to the system, thus threatening the safety of the system.

Multiple sensors in CPS can provide redundant sensory information that can be employed to not only increase the robustness of the system [4]–[6] but to also defend against malicious sensor attacks [7]–[10]. For example, the speed of a vehicle can be measured from GPS, wheel encoders and IMU separately. This redundant sensory information on the speed of the vehicle can be used for detecting and/or masking some sensor attacks. [4] provides a sensor fusion algorithm for multiple redundant sensor measurements and the worst-case analysis result for the bounded number of faulty sensors. [8] provides an attack-resilient version of [4] considering a sensor communication schedule to limit the capability of the attacker. [9] extends the sensor fusion algorithm [4] incorporating measurement history into sensor fusion. The authors of [7] observe that some sensors can exhibit transient faults in the course of normal system operation (e.g., GPS in a tunnel, wheel encoder when the wheel slips), which should not be treated as sensor attacks. [7] proposes a method to detect sensor attacks based on the transient fault model, which extends the abstract sensor model in [4] to differentiate transient faults from non-transient attacks.

From the perspective of the existing sensor attack detection work in [7], we consider systems that have multiple operating modes in this work. For example, an unmanned ground vehicle can have different operating modes such as high-speed and low-speed cruise control. To use the attack detection method in [7] for such a vehicle system with multiple operating speeds, one could conservatively train one set of transient fault model parameters at the maximum operating speed, and use it for all operating speeds at runtime. However, with those model parameters, the attack detection would not be effective at lower operating speeds because we observe that there is a correlation between the vehicle's speed and the worst-case noise bound (i.e., at a lower speed, the speed sensor noise tends to be smaller, so is the worst-case noise bound). At a lower speed, one could use more precise transient fault model parameters to increase the attack detection performance while preserving the soundness of the attack detection.

In this paper, we focus on addressing the problems of the existing transient fault model, and propose an adaptive method designed to make the transient fault model adaptive to the current operating mode of the system. To do this, we propose an automatic method to train transient fault model parameters to construct the lookup table for the adaptive transient fault model parameter selection. We conduct a real-world case study using data obtained from an unmanned ground vehicle called Jackal [11], and demonstrate the effectiveness of our proposed system in attack detection for various sensor attack scenarios.

II. Problem Formulation and Preliminaries

We consider a system with multiple sensors that measures the same physical variables. Before they are fused together to be sent to the system's controller, the attack detection is performed to detect and discard the attacked sensors. In the next subsection, the sensor model that is used for sensor fusion and attack detection in this work is explained.

978-1-5090-4403-0/16 $31.00 © 2016 IEEE
DOI 10.1109/CPSNA.2016.21

A. Abstract Sensor Model

Multiple redundant sensor values can be fused together to provide a better estimate value to a controller. To perform sensor fusion, the first thing to consider might be the choice of a sensor model. In this paper, we consider the abstract sensor model [4], [5], [12]–[15]. Unlike probabilistic sensor models [16], the abstract sensor model does not assume any noise distributions [7]. The abstract sensor model is well suited for the worst-case analysis of sensor behavior such as attack detection [7]. Marzullo has proposed an interval-based fault tolerance fusion algorithm [4], where the accuracy of those algorithms is better than individual sensor inputs, and the fusion interval size is bounded if the number of faulty sensors are bounded. An abstract sensor is represented by an interval $[y - \epsilon, y + \epsilon]$, which is made by error bound ϵ and sensor measurement y. The length of an interval determines the accuracy of the sensor. In this way, each sensor measurement is converted to an interval, and the Marzullo's fusion algorithm produces a fusion interval with respect to the assumption on the maximum number of faulty sensors [4]. This method can also be used as a conservative way to detect sensor attacks [7].

B. Attack Detection with the Transient Fault Model

Extending the abstract sensor model, the transient fault model has been developed to differentiate mere transient faults from sensor attacks [7]. The *transient fault model* (TFM) for a sensor is represented by three parameters (ϵ, e, w) where ϵ_i and e_i respectively represent the error bound and the maximum number of transient faults per given window w_i. The key concept of the attack detection method [7] is two types of pairwise comparisons of inconsistency between sensors: *weak inconsistency* (WI) and *strong inconsistency* (SI). WI between two sensors at a certain time means that at least one of the sensors provides a faulty measurement at that time. WI happens if the abstract measurements (i.e., intervals) of the two sensors do not overlap with each other. SI between two sensors at a certain time means that at least one of the sensors is non-transiently faulty at that time, thus considered to be attacked. SI happens when WI happens more frequently than a certain threshold.

The paper [7] also suggests the scheme to obtain the sensor's TFM parameters (ϵ, e, w) from training data because sensor manufactures may not provide such model parameters. To find the values for the parameters, one first needs to analyze the sensor noise of training data. In a given window size w, the transient fault number indicates the number of data points that are greater than the error bound ϵ. The maximum number of transient faults in a window becomes the parameter e. To select the best parameter set, a graph is drawn by varying the error bound (e.g., Fig. 1) and used to choose the knee point of the graph as the TFM parameter [7].

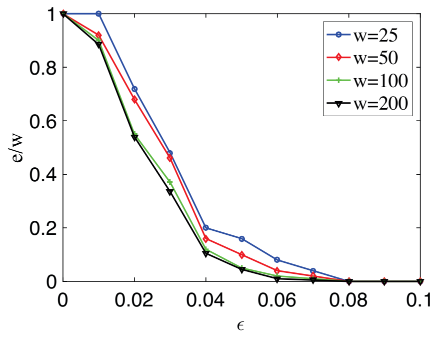


Figure 1: Example plot of error bound vs. e/w for the left encoder of Jackal robot system

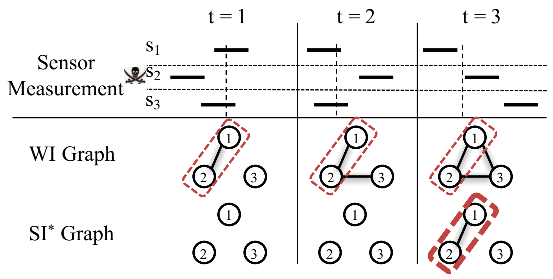


Figure 2: Example of attack detection using TFM.

Fig. 1 shows the example plot obtained from the left encoder of Jackal Robot, which will be explained in detail in Section IV. In Fig. 1, each color of the line represents the window size. The point to consider is where the slope rapidly changes, which is called the *knee point*. The knee point for the window size of 100 becomes where the error bound is 0.04 in Fig. 1. Thus, we determine the sensor's error bound to be 0.04 for this window size. The calculated TFM parameters for different window sizes are also listed in Table I. It is important to select and use proper TFM parameter values because the attack detection performance is affected by the parameters.

Fig. 2 shows an example to explain how the attack detection method using TFM works to detect the attack. First of all, the interval is constructed for each sensor using its sensor measurement and error bound for each time round as shown in the top of Fig. 2. The vertical dotted line indicates where the true value is. When intervals do not intersect with each other at a certain time, WI occurs, as there is an edge between sensor s_1 and s_2 in the WI graph at $t = 1$ in Fig. 2. Suppose that for each sensor, at most one fault is allowed within a time window of size 3 (i.e., $e = 1$, $w = 3$). At time 3, three occurrences of WI between sensors s_1 and s_2 are observed during the last three consecutive rounds. This implies that at least one sensor between s_1 and s_2 does not conform to its transient fault model. Thus, SI between s_1

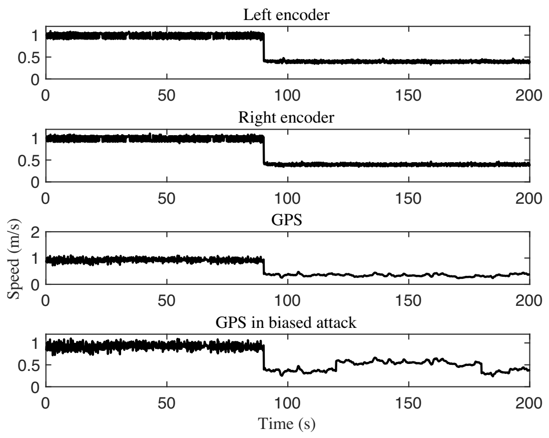


Figure 3: Sensor data for the example scenario and attack in GPS.

and s_2 occurs at time 3, and an attack detection alarm is raised.

C. Motivation and Example

In this subsection, we illustrate a motivating example where the attack detection fails due to the use of the conservative TFM parameter in a multiple operating mode system. To illustrate this example, we use the data from the Jackal robot system to be explained in Section IV, and perform transient fault modeling as shown in Fig. 1.

We consider a multiple operating mode system and an example scenario where the system's reference speed decreases from 1 m/s to 0.4 m/s. We examine the consequence of the existing attack detection system [7] which uses the fixed conservative TFM parameter.

The data for the example scenario is collected by changing the vehicle's speed from 1.0 m/s to 0.4 m/s as shown in Fig. 3, which shows the sensor measurements for left and right encoders and GPS. A simulated bias attack (magnitude of 0.22) is added to the GPS data when the system runs at the low speed mode (0.4 m/s), as shown at the bottom graph in Fig. 3.

Fig. 4 shows the result of attack detection using the existing system [7] with the window size of 100. Fig. 4 shows the number of WI occurrences (i.e., y axis) and the red line representing the threshold of the number of WI occurrences to raise SI (which is the sum of the number of faults allowed for two sensors), where:

- WI (L. enc, R. enc): WI between left encoder and right encoder
- WI (L. enc, GPS): WI between left encoder and GPS
- WI (R. enc, GPS): WI between right encoder and GPS

If the number of occurrences of WI goes over the red line in the graph, it indicates that there is an SI detected, meaning that there exists a non-transient fault (assumed to be an

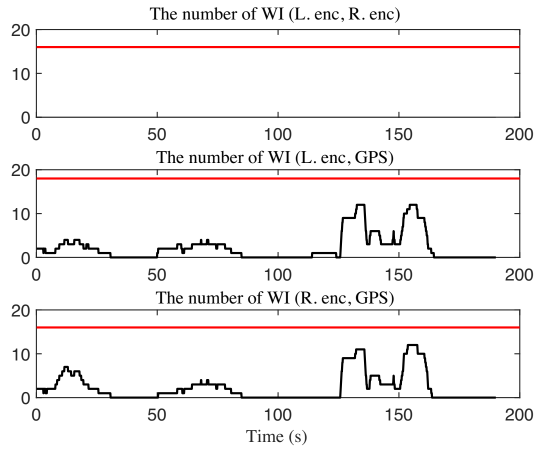


Figure 4: The attack detection result of the existing system.

attack) present in the system. In Fig. 4, the result shows that the existing system does not detect the attack present in the GPS signal in this attack scenario.

D. Problem Statements

In this subsection, we formulate our problem statements. The first problem that we consider is as follows:

Problem 1: Adapt the attack detection method to use the transient fault model parameters which are suitable to the system's operating mode.

To address this problem, we, in the next section, propose an attack detection method which uses a lookup table that contains the values of parameters for the transient fault model, which are trained for each operating speed. As the transient fault modeling is currently done manually, it is necessary to automate the transient fault modeling process to perform it multiple times for each operating speed. Thus, the second problem that we consider is as follows:

Problem 2: Given a training data, automate the transient fault modeling procedure (i.e., heuristic algorithm to find the knee point)

To address this problem, we propose an automatic transient fault modeling method to be explained in the next section.

III. Adaptive Transient Fault Model

In this section, we explain our proposed system using an adaptive transient fault model. Our proposed system uses the lookup table which contains multiple TFM parameter values trained for the different operating modes of the system. To construct such a lookup table, we propose an automatic process for the transient fault modeling which has been done only manually in the existing work [4].

A. Detection Scheme with Adaptive Transient Fault Model

Fig. 5 shows the architecture of our proposed system. First of all, the measured values from sensors are sampled at a

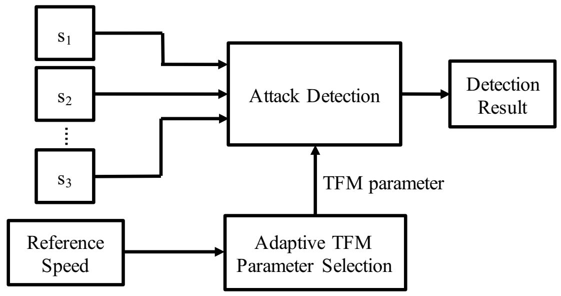


Figure 5: The architecture of our proposed system.

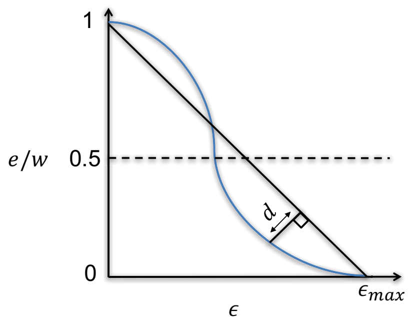


Figure 6: A sample graph of ϵ vs. e/w

certain sampling rate. For example, in our case study to be explained in the next section, we use 10 Hz of sampling rate because the attack detection is performed based on the lowest frequency among sensors (i.e., GPS in our system). The system uses sampled measurements and the parameters from the lookup table to detect if there is any attack present to the system. Then, when the reference speed changes, the values for TFM parameters are also updated according to the system's reference speed. When the speed is not listed on the lookup table, the system uses the parameter values of the lowest speed which is higher than the reference speed in the lookup table. In the next subsection, we will explain how to construct the lookup table, which is used in the adaptive TFM parameter selection.

B. *Automating the Transient Fault Modeling Process*

Selecting the suitable values for TFM parameters is important for the attack detection performance. In the previous work [4], transient fault modeling has been done manually. Therefore, it is necessary to repeat the modeling process for the different sets of training data in order to construct the lookup table for a multiple operating mode system. Therefore, we propose an automatic method to find the proper values for parameters using the characteristic of knee point (i.e., rapid increase of slope), thus reducing the manual efforts for transient fault modeling.

Fig. 6 shows a sample plot of the proportion of faults in a window (e/w) against the error bound (ϵ), which can be

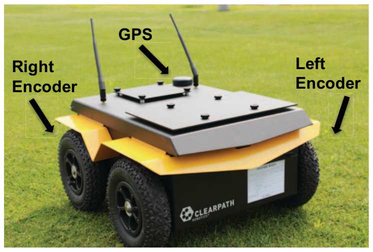


Figure 7: Jackal robot.

drawn following the description on transient fault modeling in Section II and in [4]. Let ϵ_{max} denote the maximum value of the error bound. To find the knee point of the graph, we make the descending diagonal line from the point $(0, 1)$ to the point $(\epsilon_{max}, 0)$ in the graph, which has the function of $y = -\frac{1}{\epsilon_{max}}x + 1$. According to the Lemma 4 of [4], we only consider the points under 0.5 of e/w as the parameter values because otherwise no detection can be made by the attack detector. From each original point of the graph, we calculate the perpendicular distance to the diagonal line (i.e., the shortest distance from a point to the diagonal line). Finally, we select the point as the TFM parameter which has the longest perpendicular distance. The reason is that, as decreases in the graph, the perpendicular distance gradually increases at first, becomes the maximum at the knee point, and decreases again after passing that point. In the next section, we will show the result of applying this automatic transient fault modeling method to a real world data set obtained from an unmanned ground vehicle.

IV. Case Study

In this section, we evaluate our proposed system by conducting a real-world case study using an unmanned ground vehicle called Jackal [12]. We compare the performance of our system with that of the existing system [7].

A. *Jackal Robot System Description*

Jackal in Fig. 7 is an electric unmanned ground vehicle that has many sensors including two encoders, IMU, and GPS [12]. We use both encoder sensors for the left and right wheel and GPS to measure the velocity of Jackal. To evaluate our proposed system explained in the previous section, we first gather data for each sensor by driving Jackal on straight lines at a constant speed. Both encoder sensors provide measurements at 20 Hz and GPS sensor offers measurements at 10 Hz. The attack detection is performed at 10 Hz which is the lowest frequency of the sensors.

B. *Lookup Table*

This subsection presents the constructed lookup table for the Jackal robot system applying the automatic transient fault modeling method of Section III. We ran the Jackal robot in various operating modes (i.e., different reference speeds

Table I: TFM parameters of the Jackal Robot's sensors for different running speeds.

(a) Parameters for left encoder.

v	$w = 25$			$w = 50$			$w = 100$			$w = 200$		
	ϵ	e	e/w	ϵ	e	e/w	ϵ	e	e/w	ϵ	e	e/w
0.4 m/s	0.04	5	0.2	0.04	7	0.14	0.04	12	0.12	0.04	19	0.095
0.7 m/s	0.09	6	0.24	0.07	7	0.14	0.07	12	0.12	0.07	19	0.095
1.0 m/s	0.1	4	0.16	0.1	7	0.14	0.1	9	0.09	0.1	13	0.065
1.3 m/s	0.14	3	0.12	0.13	7	0.14	0.13	8	0.08	0.13	10	0.05
1.6 m/s	0.16	4	0.16	0.16	5	0.1	0.16	7	0.07	0.16	10	0.05

(b) Parameters for right encoder.

v	$w = 25$			$w = 50$			$w = 100$			$w = 200$		
	ϵ	e	e/w	ϵ	e	e/w	ϵ	e	e/w	ϵ	e	e/w
0.4 m/s	0.04	5	0.2	0.04	8	0.16	0.04	12	0.12	0.04	18	0.09
0.7 m/s	0.07	4	0.16	0.07	7	0.14	0.07	10	0.1	0.07	14	0.07
1.0 m/s	0.1	3	0.12	0.1	4	0.08	0.1	7	0.07	0.09	11	0.055
1.3 m/s	0.13	3	0.12	0.13	4	0.08	0.13	6	0.06	0.13	8	0.04
1.6 m/s	0.16	43	0.12	0.16	5	0.1	0.16	6	0.06	0.16	7	0.05

(c) Parameters for GPS.

v	$w = 25$			$w = 50$			$w = 100$			$w = 200$		
	ϵ	e	e/w	ϵ	e	e/w	ϵ	e	e/w	ϵ	e	e/w
0.4 m/s	0.12	2	0.08	0.11	6	0.12	0.11	11	0.11	0.08	36	0.18
0.7 m/s	0.14	2	0.08	0.14	7	0.14	0.14	8	0.08	0.12	18	0.09
1.0 m/s	0.16	3	0.12	0.14	7	0.14	0.14	9	0.09	0.12	20	0.1
1.3 m/s	0.16	4	0.16	0.14	8	0.16	0.14	11	0.11	0.12	19	0.095
1.6 m/s	0.2	4	0.16	0.16	9	0.18	0.16	11	0.11	0.14	16	0.08

such as 0.4 m/s, 0.7 m/s, 1.0 m/s, 1.3 m/s and 1.6 m/s) and collected the training data from the runs. The transient fault modeling result for all speeds is summarized in Table I, where v denotes the speed that varies from 0.4 m/s to 1.6 m/s, and the window size w varies from 25 to 200. Table I demonstrates that different speeds yield different TFM parameters. We observe the relationship between the error bound (ϵ) and the speed (v) such that the error bound increases as the speed becomes faster. We also observe that the larger window size yields a smaller error bound.

This lookup table is used to select the proper TFM parameter values given a reference speed. If the reference speed is not contained in the lookup table, our system chooses the parameters of the next higher speed rather than the next lower speed. The reason for this is to be sound in the attack detection and avoid any false alarms. For instance, if the system's reference speed is 0.5 m/s, the parameters of 0.7 m/s are used for the given reference speed because the lookup table (Table I) contains no entry for 0.5 m/s.

To validate the results obtained from our automatic modeling method proposed, we also manually conducted the TFM parameter selection for all the cases above. We observed that the result of the automatic method coincides with the one that is found manually in all cases.

C. Evaluation on Motivating Example

We recall the motivating example in Section II. In this subsection, we compare our proposed system with the existing system [7] using the motivating example. The existing system uses the most conservative TFM parameter values such that the error bounds are 0.1, 0.1 and 0.14 for left and right encoders and GPS respectively. This system uses these parameter values for all operating speeds such as 1 m/s and 0.4 m/s. In contrast, our proposed system uses the adapted TFM parameter values according to the given reference speed to the system. At 1 m/s, our system uses the same TFM parameters to the ones that the existing system uses while our system at 0.4 m/s uses the error bounds of 0.04, 0.04 and 0.11 for left and right encoders and GPS respectively. We note that both the existing system and our proposed system make no false alarm in the normal case where there is no attack present.

Now, we consider the attack scenario of the motivating example, where the biased attack (magnitude of 0.22) is added in GPS as shown in Fig. 3. In contrast to the existing system which uses the most conservative TFM parameters, our proposed system uses the adapted TFM parameters according to the system's reference speed. Fig. 8 shows the number of WI occurrences as the result of attack detection. The red line indicates the threshold to raise an alarm for attack detection. We observe in Fig. 8 that our proposed system quickly detects the attack of this example while the existing system could not detect it as shown in Fig. 4.

Moreover, we also consider another scenario, a variation of the motivating example, where the random attack (of uniform distribution with magnitude of 0.22) is manifested

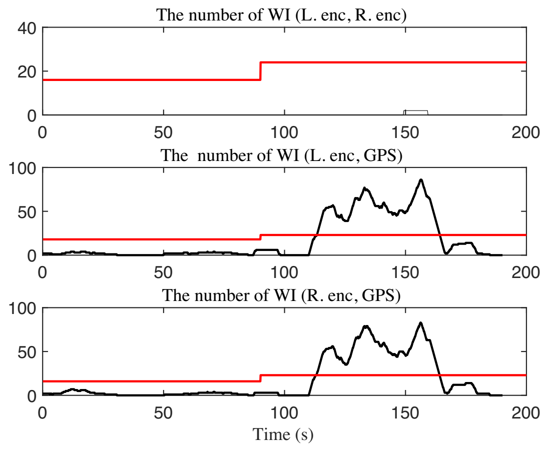


Figure 8: The attack detection result of our system for the example attack scenario.

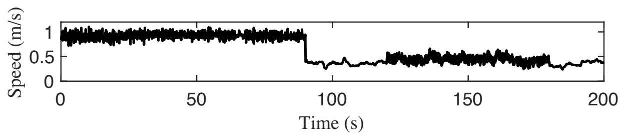


Figure 9: Random attack in GPS.

Table II: False alarm rate.

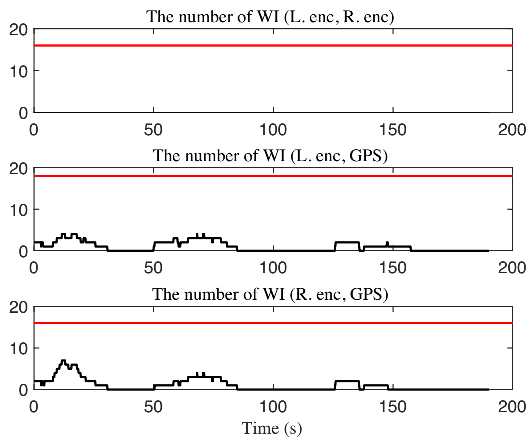
Detector	Existing System	Proposed System
False Alarm Rate (%)	0 %	0 %

instead of the biased attack. The magnitude of the attack is decided to be roughly as large as the size of the largest error bound (i.e., GPS). Fig. 9 shows the GPS measurements where the random attack is added. Fig. 10a shows that the existing system does not detect any attack while our proposed system adapts to the speed and uses the proper parameter values, thus being capable of detecting the attack of this scenario as shown in Fig. 10b.

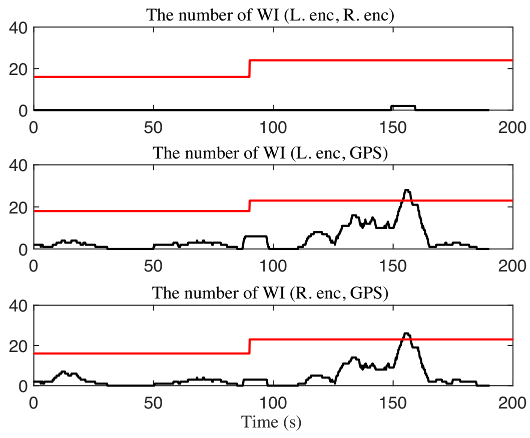
Besides the biased and the random attacks, there might be a type of attacks where the attacker has the full knowledge of the system including the detection scheme [7], [8]. This kind of attack may be able to remain undetected. Thus, such attacks are not considered in our case study. However, note that the attacker's capability should be limited in order to remain stealthy from the detector (e.g., conforming to the sensor's TFM).

D. Further Evaluation

To provide a more thorough comparison, we employ 12 different test data sets obtained from the multiple runs of the Jackal robot system where the operating mode changes from the high speed mode (i.e., 1 m/s) to the low speed mode (i.e., 0.4 m/s). First of all, we calculate the false alarm rate for both systems in the normal operation where there is



(a) The attack detection result of the existing system.



(b) The attack detection result of our system.

Figure 10: The attack detection result in the random attack scenario.

Table III: Detection rate.

(a) Detection rate (%) for biased attack.

Magnitude	Existing System	Proposed System
0.33	100 %	100 %
0.22	13.8 %	100 %
0.11	5.5 %	58.3%

(b) Detection rate (%) for random attack.

Magnitude	Existing System	Proposed System
0.33	97.2 %	100 %
0.22	2.7 %	100 %
0.11	0 %	11.1 %

no attack present. Table II summarizes the result that both systems did not make any false alarms.

Now, we calculate the detection rate for both detection systems against random[1] and biased attacks of varied mag-

[1]The uniform distribution is used.

nitudes (0.11, 0.22 and 0.33) manifesting in one of the sensors. Table III shows the detection rates for both types of attacks of different magnitudes. When the magnitude of the attack is large (i.e., 0.33), both systems have an equally good performance in the case of biased attacks, while our proposed system performs slightly better than the existing system for random attacks. The reason is that since the attack magnitude is large, both systems are able to easily detect the attacks. On the other hand, when the magnitude of the attack is small (i.e., 0.11), both systems can detect less attacks, but we can see that our proposed system has a better performance than the existing system. Lastly, if the attack magnitude is medium, our proposed system outperforms the existing system.

V. Conclusion

In this paper, we addressed the problem of sensor attack detection for multiple operating mode systems. The existing system uses the most conservative TFM parameter to be sound in detection, thus not being effective to use for a multiple operating mode system. We proposed the adaptive transient fault model to address the problem of the existing system. Our system uses the suitable parameter values in accordance with the reference speed (i.e., system's operating mode) using the lookup table method. The lookup table is constructed using the automatic transient fault modeling method which is presented in this paper. By conducting a real-world case study, we demonstrated that our proposed system outperforms the existing system in various attack scenarios.

For our future work, we plan to study the online learning scheme for finding the proper TFM parameter values at runtime. Additionally, we plan to incorporate system dynamics in our attack detection system to improve the detection performance against various stealthy and collusion attacks.

Acknowledgement

This research was supported in part by Global Research Laboratory Program (2013K1A1A2A02078326) through NRF, by the DGIST Research and Development Program (CPS Global Center) funded by the Ministry of Science, ICT & Future Planning, and by NSF CNS-1505799 and the Intel-NSF Partnership for Cyber-Physical Systems Security and Privacy. This material is based on research sponsored by DARPA under agreement number FA8750-12-2-0247. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. The first two authors contributed equally to this work.

References

[1] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *SEC'11: Proc. 20th USENIX conference on Security*, 2011, pp. 6–6.

[2] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental security analysis of a modern automobile," in *SP'10: IEEE Symposium on Security and Privacy*, 2010, pp. 447–462.

[3] A. H. Rutkin, "'Spoofers' Use Fake GPS Signals to Knock a Yacht Off Course," MIT Technology Review, August 2014.

[4] K. Marzullo, "Tolerating failures of continuous-valued sensors," *ACM Trans. Comput. Syst.*, vol. 8, no. 4, pp. 284–304, Nov. 1990.

[5] R. R. Brooks and S. S. Iyengar, "Robust distributed computing and sensing algorithm," *Computer*, vol. 29, no. 6, pp. 53–60, Jun. 1996.

[6] Y. Zhu and B. Li, "Optimal interval estimation fusion based on sensor interval estimates with confidence degrees," *Automatica*, vol. 42, no. 1, pp. 101–108, 2006.

[7] J. Park, R. Ivanov, J. Weimer, M. Pajic, and I. Lee, "Sensor attack detection in the presence of transient faults," in *Cyber-Physical Systems (ICCPS), 2015 ACM/IEEE International Conference on*, 2015.

[8] R. Ivanov, M. Pajic, and I. Lee, "Attack-resilient sensor fusion," in *DATE'14: Design, Automation and Test in Europe*, 2014.

[9] ——, "Resilient multidimensional sensor fusion using measurement history," in *HiCoNS'14: High Confidence Networked Systems*, 2014.

[10] ——, "Attack-resilient sensor fusion for safety-critical cyber-physical systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 15, no. 1, p. 21, 2016.

[11] Clear Path Robotics, "Jackal Robot System," 2016, http://www.clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/.

[12] B. Ao, Y. Wang, L. Yu, R. R. Brooks, and S. Iyengar, "On precision bound of distributed fault-tolerant sensor fusion algorithms," *ACM Computing Surveys (CSUR)*, vol. 49, no. 1, p. 5, 2016.

[13] D. Dolev, N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl, "Reaching approximate agreement in the presence of faults," *Journal of the ACM (JACM)*, vol. 33, no. 3, pp. 499–516, 1986.

[14] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the ACM (JACM)*, vol. 32, no. 2, pp. 374–382, 1985.

[15] N. H. Vaidya, L. Tseng, and G. Liang, "Iterative approximate byzantine consensus in arbitrary directed graphs," in *Proceedings of the 2012 ACM symposium on Principles of distributed computing*. ACM, 2012, pp. 365–374.

[16] A. S. Willsky, "A survey of design methods for failure detection in dynamic systems," *Automatica*, vol. 12, no. 6, pp. 601–611, 1976.