Comprehensive DSRC Communication Testing Design and Support for Next-Generation Smart Vehicles

Qingquan Zou[1, a], Wei Zhang[2,b], Min Huang[2,c], Wenhan Huang[2,d], Sijun Li[2,e]

[1] SAIC Limited, No. 489 Weihai Road, Shanghai, China

[2] TongJi University, No. 4800 Caoan Road, Shanghai, China

[a]zouqingquan@saicmotor.com, [b]zhang_wi@tongji.edu.cn, [c]297259020@qq.com, [d]424215623@qq.com, [e]524993618@qq.com

Keywords: Test design, DSRC communication, V2X, smart vehicles, test support system.

Abstract. DSRC plays an important role in current V2X communication market, but its performance in real environment is still not very clear which raise potential dangerous to both the passengers and the vehicle factory's business. It's important for the vehicle designer to know the communication performance and limitations. Testing in realistic environment mimicking actual scenarios is currently the best method. However, there're too many factors affect the communication performance such as the road difference, velocity, distance, protocol design policy and current network traffic load. The testing itself is challenging so we proposed a systematic framework to do the DSRC testing and developed related testing programs. This system greatly improves the testing efficiency. The testing shows that the DSRC technology can satisfy most of the V2X applications but the application itself should have additional consideration on packet loss problem and shouldn't exceed the bandwidth limit.

Introduction

Smart vehicle is important in the future transportation world. They're expected to improve the transportation efficiency and to decrease the potential accidents [1]. And the vehicle to X (X here can be another vehicle, road-side facilities, and people) communication (V2X) is crucial to support all above applications. Unfortunately, the communication performance under different circumstance and the technical limitations are still not very clear in real environment, which raises potential dangerous for the vehicle design because this affect the vehicle "reliability" and "driving safety". Changan University made the Dedicated Short Distance Communication (DSRC) testing in 2014 [2,3]. They choose ZigBee and RFID for DSRC testing due to technical restrictions. Since there're very few reference, most of the advances borrows the hardware/software and standard from the wireless sensor network fields [4,5]. However, these are not industry accepted V2X communication technologies. That's why TongJi Unversity and SAIC Limited issues a joint project to do V2X communication testing in outside realistic environments.

Currently the IEEE 1609.3 and 1609.4 standard in the DSRC world are active players in this market due to their maturity. The 1609.4 standard is mainly for data link layer or medium access layer. It improves the traditional popular 802.11 wireless protocol to adapt to the moving circumstance [6]. IEEE 1609.3 implements a light weight network layer based on 1609.4 standard in order to support some typical V2X applications [7].

Testing Analysis and Design

The V2X communication testing usually involves several moving vehicles and requires high performance and accurate data logging. High precision time and geographical position information must be recorded at the same time. And all the vehicles must be synchronized in time accurately before testing. This is accomplished by using GPS. Another challenging is the data analysis. Different from most of the classical testing, this one requires comprehensive analysis of all test data to get final

reports rather than the "report from a single test" model. The whole process can be abstracted as the following:

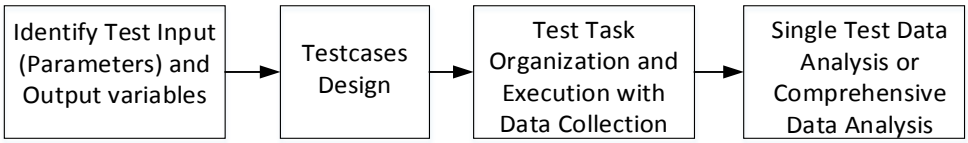

Fig. 1: Basic testing work organization.

Test case design is the key in the whole workflow. The first step is to identify the factors which affect DSRC performance. Tables 1 lists all the factors considered in the test:

Table 1: Testing inputs variables (containing parameters)

Input	Possible Values	Description
x_1: Network Layer	MAC, NET	
x_2: Field	City road, Country Road, High way, Open field	In which the test vehicles running.
x_3: velocity	0, 0-20, 20-40, 40-60, 60-80, 80-100, 100-120	Km/h
x_4: distance	0-5, 5-15, 15-35, 35-75, 75-150, 150-250, 250-750, >750	*m*
x_5: Traffic load factor	Light, medium, heavy	Implemented by tuning the packet sending rate.
x_6: Neighbor vehicle inference strength	0%, 25%, 50%, 75%, 100%	100% is the full sending speed
x_7: Antenna count	1 or 2	
Motor switch on/off x_8: disturbance	None or exist.	

After identifying the factors, the test cases can be derived from them. Attention that all the above factors are the "input" or control parameters to a test. The final output are network indexes indicating the quality of service. There most important expected output is lost rate, communication delay an bandwidth. So the test model can be abstracted as:

$$X := \{x_i, i \in [1,..,8]\}, Y := \{y_1 : lossrate, y_2 : delay, y_3 : bandwidth\}$$

Where *X* is input and *Y* is the output. Each combination of *X* values should be designed as an independent test cases. The test process is essentially to get the value of *Y* when given a determined set of *X* values.

However, the number of test cases will explosion rapidly because the combination of all *X* value set are huge. The following policy are introduced in real testing work organization to help simplify the work and decreasing the testing cost:

1) Task aggregation: Task replaces the test to help organizing resources. A task is usually respond to a test but may also response to a sequence test if these tests can be fulfilled in a single driving trial;
2) Continuous test in the outside: this requires the support of the field testing software;
3) Backend IT system support to improve data management and analyzing efficiency.

Testing System Architecture and Developing

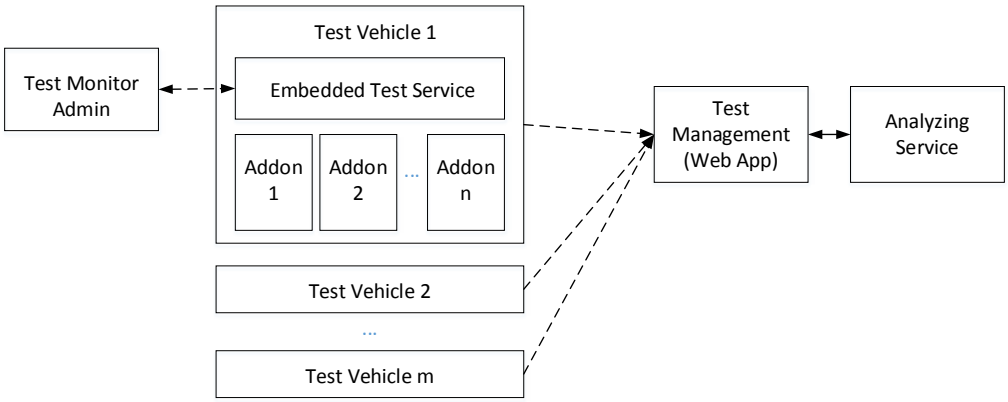

Fig. 2 Test System Architecture

Due to the complexity of the whole testing organization, the testing process highly depends on the test system support to improve efficiency as illustrated in Fig.2. The whole implementation adopts the micro-service architecture and restful interface techniques. C, Java and Python are used in developing. It can be divided into field side testing support and backend testing support.

Field Side Test Application. This includes the test service, test addon and test monitor. The test service is running inside the DSRC devices. It accepts the commands from the admin (monitor) program and then interprets the commands such as START and STOP. The monitor is a QT based GUI application running on Windows to help select testing.

Different tasks usually require different packet sending/receiving policies. So the system provides more than 6 testing addons to help implementing different tests. Addon is currently implemented as an independent program and can be called or killed by the test service. For example, and simple loss rate test requires a low rate frame sending addon and a corresponding receiver addon. Generally, the addon program can forward the transmitted (TX) frames and received (RX) frames to the test service and let the test service to write them to log file.

Backend Side Application. Responds to manage the data upload and analyzing. It's a web based application running on Linux server so that it can accessed remotely. It also provides plenty of reports.

Data Analyzing Service. Called by the web application to fulfill the data analyzing tasks. It's developed by Python. It can extract target data from the entire logging database, process them, and then save the results into database for reporting. Here's a analyzing task for loss rate evaluation:

Algorithm: loss_rate_evaluation;
Input: log data files;
Output: loss rate distribution and loss rate
Begin
    Extract the original interested test data from the full data log into logstream;
    For each packet, extract the geographical position data from the logstream;
    Estimate the vehicle velocity at the TX/RX time of each packet;
    Iterate each TX packet, and searching its corresponding RX packets. Attention the RX packet may be lost in the transmission.
    For each packet pair
        Estimate the distance and transportation delay;
    End
    Group all the TX/RX pair according to velocity and distance;
    Average the loss rate in each group;
    Return tuple {velocity, distance, loss rate} which can be easily visualized;

Do average of all groups' loss rate;
Return the average loss rate;
End

System Implementation. The final DSRC testing design and support system is mainly developed by Java except the data analyzing algorithms are developed by Python. The web based graphical user interface support flexible combinations of multiple dimensions as user query input including network layer, velocity, distance, sending rate, disturbance type and more. The server side interprets user input and do statistics based on OLAP techniques. Table. 2 lists some output of the system:

Table. 2: Some Statistics of the DSRC Testing and Support System

Network Layer	Environment	Bandwidth (B/s)	Delay (us)	Loss Rate (%)
Data Link Layer	Country Road	303434	1363	18.57
	City Road	299072	1838	0.95
	Highway	413212	1377	1.49
Network Layer	Country Road	330993	3416	18.79
	City Road	202307	3208	1.09
	Highway	380209	3465	0.59

Conclusion

The DSRC (1609.3 and 1609.4) are optimized for moving vehicle communications. We greatly improved the testing efficiency based on the above testing support system. And it lay a solid foundation for future large scale connected vehicle testing. Most of the communication delay is under 3ms. Some may exceed 4ms. Which means it's pretty fast for V2X communication. However, the loss rate isn't stable. Fortunately, the 1609.3 and 1609.4 won't repeat too many times like the traditional 802.11 WLAN. They will report failure as quickly as possible in order to adapt to the fast moving scenarios. This policy can decreases the communication time but may probably increasing the loss rate. This means the V2X applications will encounter higher packet loss due to both the environment factors and the protocol themselves. The application layer developing must consider this and avoid violating the bandwidth communication restrictions.

Acknowledgements

This work was financially supported by Shanghai science and Technology Commission and SAIC Motor Corporation Limited.

References

[1] Safety Pilot Project, http://safetypilot.umtri.umich.edu/.
[2] Z.T. Duan, J. Kang, L. Tang, M.M. Lu, D. Wang, C. Wang, Wireless network testbed oriented internet of vehicles, Journal of Traffic and Transportation Engineering, Vol.14.2(2014), p.104-111.
[3] D. Wang, Design of Road to Vehicle Wireless Network Performance Measurement Testbed for Internet of Vehicles, A Dissertation for the degree of master, Chang'an University (2014).
[4] H. Vlado, K. Andreas, W. Andreas, W. Adam, TWIST: A Scalable and Reconfigurable Testbed for Wireless Indoor Experiments with Sensor Networks, In Proc. of the 2nd Intl. Workshop on Multi-hop Ad hoc Networks: from Theory to Reality (2006).
[5] Computer Science Department,University of Virginia.VineLab WirelessTestbed. http://www.cs.virginia.edu/~whitehouse/research/testbed/index.html.
[6] 1609.3 IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services, http://standards.ieee.org/findstds/standard/1609.3-2010.html.
[7] 1609.4 IEEE Standard for Wireless Access in Vehicular Environments (WAVE)--Multi-channel Operation, http://standards.ieee.org/findstds/standard/1609.4-2010.html.