Towards Dependable Autonomous Driving Vehicles: A System-Level Approach

Junsung Kim, Ragunathan (Raj) Rajkumar
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
{junsungk, raj}@ece.cmu.edu

Markus Jochim
General Motors R&D
30500 Mound Rd.
Warren, MI, USA
markus.jochim@gm.com

ABSTRACT

Autonomous driving technologies have been emerging over the past few years, and semi-autonomous driving functionalities have been deployed to vehicles available in the market. Since autonomous driving is realized by the intelligent processing of data from various types of sensors such as LIDAR, radar, camera, etc., the complexity of designing a dependable real-time autonomous driving system is rather high. Although there has been much research on building a reliable real-time system using hardware replication, the resulting systems tend to add significant extra cost due to hardware replication. Therefore, an alternative solution would be helpful in building an autonomous vehicle in a cost-effective way. An autonomous driving system is different from the conventional reliable real-time system because it requires (1) flexible design, (2) adaptive graceful degradation and (3) effective use of different modalities of sensors and actuators. To address these characteristics, we summarize SAFER (System-level Architecture for Failure Evasion in Real-time applications) our previous work on flexible system design. We then present a conceptual framework for autonomous vehicles to provide adaptive graceful degradation and support for using different types of sensors/actuators when a failure happens. We motivate our proposed framework with various scenarios, and we describe how SAFER can be extended to support the proposed conceptual framework.

Categories and Subject Descriptors

C.0 [**Computer Systems Organization**]: GENERAL—*System architectures*; D.4 [**Software**]: Operating Systems; D.4.5 [**Operating Systems**]: Reliability—*Fault-tolerance*; D.4.7 [**Operating Systems**]: Organization and Design—*Real-time systems and embedded systems*; I.2.9 [**Artificial Intelligence**]: Robotics—*Autonomous vehicles*

General Terms

Design, Management, Reliability

Figure 1: Boss: an award-winning autonomous vehicle developed at CMU [2]

1. INTRODUCTION

One of the main reasons for the Second Industrial Revolution in the late 19^{th} century was the invention of the internal combustion engine, which enabled the automotive industry. Since then, the automotive industry has been tightly connected to daily life by providing means of efficient transportation. Technologies developed for vehicles have also been widely used in other industrial domains and vice versa.

The automotive industry has had an immense impact on our economy and society, and this trend can continue with autonomous driving vehicles [4, 14, 7, 17] such as Boss [18] depicted in Figure 1. Autonomous driving technologies can enable smart traffic control based on vehicular networks [3] and active safety features. Car manufacturers have started deploying safety features such as adaptive cruise control, stop-and-go cruise control, lane keeping and assisted lane change. As vehicles become semi-autonomous, drive autonomously on demand and eventually become fully autonomous, a multitude of computer vision, sensor fusion, signal processing and artificial intelligence problems must be solved in real-time. This trend towards a fully autonomous driving system poses challenges in achieving timeliness and reliability of the system. An example computing platform architecture [18] is illustrated in Figure 2, where complex algorithms process data from various types of sensors such as LIDAR (LIght Detection And Ranging), radar, camera, GPS, etc. and actuate a drive-by-wire controller to move the vehicle. The intelligent data handling tasks run on a group

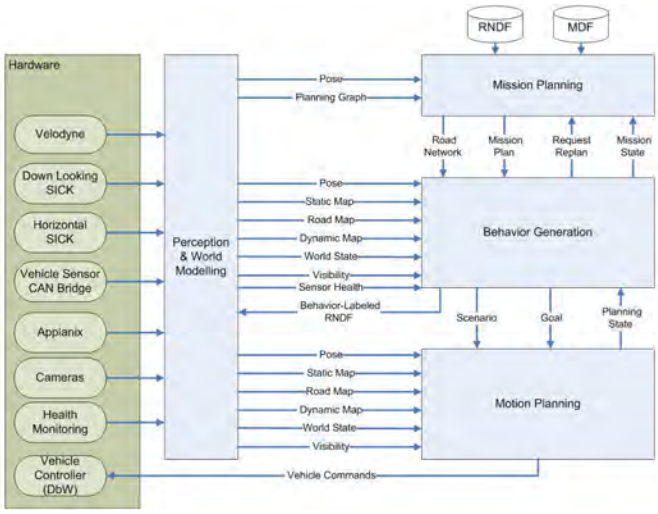

Figure 2: Boss computing platform architecture [18]

of processing units. For example, Boss is equipped with 20 processing cores, i.e. 10 Intel dual-core processors.

The following three properties are desirable in building a cost-effective autonomous driving system: flexibility, adaptive graceful degradation and smart use of sensor/actuator modalities. *Flexibility* can be accomplished in designing a reliable computing platform for autonomous driving. Due to the cost-sensitiveness of the automotive industry, hardware replication that raises vehicle costs significantly may not always be an attractive option. Nowadays, even the same model may have different features per unit, and the amount of required hardware may be different to reduce vehicle costs. Therefore, improving the reliability of autonomous vehicles should be achieved in a flexible manner so that it can also be applicable to different hardware configurations.

Adaptive graceful degradation can be supported in various features. For example, if autonomous vehicles can pull over to the shoulder lane when a failure happens, it may be acceptable as long as drivers and passengers are guaranteed to be safe. This may cause fewer needs for redundant resources than supporting full-fledged recovery modes, and using fewer resources is always beneficial in a mass-production industry. Therefore, it is important to be able to support graceful degradation in an adaptive manner so that autonomous vehicles can recover a failure with less resources.

The effective use of *different sensor/actuator modalities* on autonomous vehicles is essential. As depicted in Figure 2, autonomous vehicles have various sensor modalities providing 360-degree coverage. Many analog sensors are prone to intermittent faults, so using different sensor modalities is better than duplicating the same type of sensors because different types of sensors typically react to the same environmental condition in diverse ways. Suppose a vehicle is equipped with radars for blind spot detection. If a backward-looking radar does not work properly, a vision algorithm detecting obstacles from images obtained through a backward-looking camera can be used. A similar approach is also applicable to actuators. An autonomous vehicle may use a low-grade sensor with complex data-processing algorithms after a high-grade sensor with simple algorithms fails, until the vehicle can safely stop.

To address the flexible system design, we proposed a layer called SAFER (System-level Architecture for Failure Evasion in Real-time applications) to embody configurable task-level fault-tolerance features to tolerate *fail-stop* processor and/or task failures for distributed embedded real-time systems [11, 9]. For the other two aspects, we present a conceptual framework for autonomous vehicles to provide graceful degradation and support for using different types of sensors/actuators when a failure happens. We motivate the necessity of our proposed framework with various usage scenarios.

The rest of this paper is organized as follows. Section 2 summarizes SAFER, and Section 3 presents our conceptual framework to support adaptive graceful degradation and the effective use of different sensor/actuator modalities. We summarize our work in Section 4.

2. AN OVERVIEW OF SAFER

SAFER was proposed for use in a real-time fault-tolerant computing platform for autonomous driving features. The computing platform must be based on a robust functional architecture that allows to perform the repeating sequence of perception, computation and vehicle control in the presence of possible system failures. In [11, 9], we proposed and implemented a distributed layer called SAFER to incorporate configurable task-level fault-tolerance support using hot standbys, cold standbys and task re-execution. SAFER tolerates fail-stop processor failures and/or task failures for distributed embedded real-time systems such as an autonomous vehicle.

The SAFER layer comprises SAFER daemons and a library. One daemon runs on each processing board, and the SAFER daemons have a master-slave architecture. One of the SAFER daemons takes the role of the master that controls the slave SAFER daemons managing tasks on each node and monitoring the health status of processing boards and tasks. The library leveraging Linux/RK [13] enables any task running on the SAFER layer to be periodic with configurable parameters. Those parameters for each task are loaded by the library when the task is launched. For the communication layer, an inter-process communication primitive such as IPC [16] can be used. The overall architecture of SAFER is illustrated in Figure 3.

Heartbeat signals from the library of each task are used for detecting task/processor failures. The standby node determines the failure of the primary if heartbeat messages of its primary are missed several times. We named this failure detection scheme as *time-based* failure detection. A task failure can also be directly detected by the SAFER daemon if the SAFER daemon catches an OS signal caused by an unexpected task failure. On failure detection, appropriate recovery can be initiated. We named this failure detection scheme as *event-driven* failure detection.

SAFER offers the following properties: (a) Each task can be backed up by zero, one or more standby(s), (b) Each standby

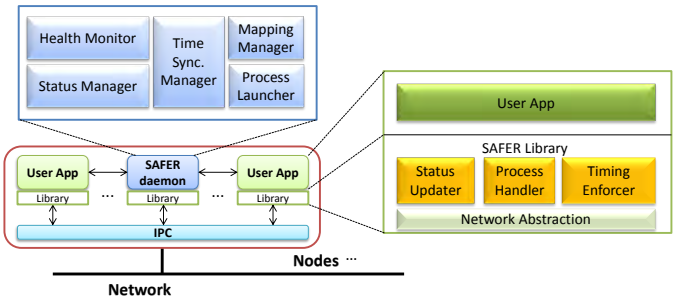

Figure 3: The overall architecture of SAFER [9]

can be either a cold standby or a hot standby, (c) A primary task and each of its standby(s) are allocated to different processor boards to avoid common failure modes, (d) Failure detection and recovery latencies are analyzed and can be guaranteed, (e) State transfer is supported for seamless recovery, and (f) An event-based failure detection method and task re-execution are supported for local task failures.

The worst-case response time analysis on SAFER shows that a hot standby can recover the failure of its primary within $kT_{heartbeat} + D_{primary} + d$, where k is a configurable positive integer, $T_{heartbeat}$ is the period of heartbeat signals, $D_{primary}$ is the deadline of the primary, and d is the network delay when the time-based failure detection method is used. A cold standby can recover the failure of its primary within $kT_{heartbeat} + nD_{primary} + d$, where n is a positive integer greater than or equal to 1 depending on the timing relation between the primary and its cold standby(s). These analysis results are also added to our model-based design tool, SysWeaver [6], for proper parameter choices at the design time.

The implementation of SAFER on Ubuntu 10.04.4 LTS and our measurements demonstrated that our goals are achievable. The evaluation results in [9] show that the failure detection time linearly increases as the period of heartbeat signals increases for the time-based failure detection method because the waiting time from the standby increases linearly. When the event-based failure detection method is used, the failure detection latency is reduced significantly. It was illustrated that the recovery time of a task is related to its task period. A case study on Boss showed that the failure detection and recovery procedures supported by SAFER do not cause any behavioral difference at the level of autonomous driving [9].

3. ADAPTIVE RESOURCE MANAGEMENT

As just described, SAFER addresses the necessity of flexible design primitives in building a reliable real-time system such as an autonomous vehicle. We now describe a conceptual framework for autonomous vehicles to provide graceful degradation and support for using different types of sensors/actuators when a failure happens.

3.1 Adaptive Graceful Degradation

Graceful degradation is a well-established approach to maintain limited functionality in a system with a component failure. The basic idea behind is to avoid a potential undesirable event by providing restricted features to accommodate the reduced resources due to a failure. When it comes to autonomous vehicles, graceful degradation should be appropriately adjusted depending on different situations. Suppose a failure on a processing board running vision algorithms to detect pedestrians. If a vehicle with the failure is driving on a highway, the vehicle may notify its driver of the failure and keep driving. If the vehicle is in an urban area, pedestrians are highly likely to be present. Hence, the vehicle may run the pedestrian detection algorithms in a degraded mode (possibly with a low frame rate) on another live processing board and also slow down the vehicle. This adaptive resource management can be realized by adjusting the period of a target task.

In a real-time system such as an autonomous vehicle, most tasks deal with a periodic sequence of perception, computation and control. By adjusting task periods, *utilization*[1] that can be treated as workload can be regulated. Lowering the utilization of a task creates more slack for other tasks to use. In other words, a framework for *adaptive graceful degradation* is desirable in order to run critical tasks with limited resources due to a failure. For example, the vision algorithms mentioned above can be run on an another live processing board along with tasks that are adjusted to have lower utilization.

The SAFER layer is a good underlying framework to support adaptive graceful degradation. When a processing board failure is detected by the SAFER layer, the cold standbys of the primary tasks on the failed board can be activated to run elsewhere. If resources are limited, the SAFER daemon and the library can work together to make sure that all required tasks are executed in a degraded manner. The schedulability of the adjusted tasks can be guaranteed by using admission control algorithms or response-time tests that can handle varying periods [10, 5, 8]. Depending on a given condition, the best configuration parameters can be adaptively set by the SAFER daemon. Accordingly, heartbeat signals and task status transfer should be adjusted as well.

3.2 Smart Sensor/Actuator Control

Autonomous vehicles are equipped with different kinds of sensors such as LIDAR, radar, camera, thermal imaging camera, ultrasonic sensor, GPS receiver, IMU (Inertial Measurement Unit), vehicular communication receiver, and so on. Hence, we can leverage sensor modalities to achieve a common goal. For example, autonomous vehicles can detect obstacles using camera images, point clouds from LIDARs, radar data and a list of detected obstacles through vehicular communication. Various sensor modalities can improve the possibility of obstacle detections and be more helpful for autonomous vehicles to recover from a sensor failure. Consider a vehicle equipped with a LIDAR sensor and vehicular communication. LIDAR can be hampered by occlusions that limit the ability to recognize the surroundings. This limitation can be significantly mitigated by using sensor data from other vehicles through vehicular communication. A similar concept is also applicable to actuators. Differential braking can be used to recover a steering failure using different

[1]The utilization of a task is defined as the ratio of worst-case execution time of a task to its period.

actuator modalities.

The SAFER layer can be extended to effectively use such sensor/actuator modalities. The SAFER layer should have the capability to detect sensor anomalies that are different for each type of sensors, which can be a plug-in module for the SAFER daemon. When a sensor failure happens, the SAFER daemon can trigger a different configuration using different types of sensors to recover from the failure. Since different data-processing algorithms are mandatory for different types of sensors, the logical combination among algorithms are given *a priori* as configuration parameters. Then SAFER can assign suitable amount of resources.

It must be noted that this method is also closely related to adaptive graceful degradation because various algorithms using different types of sensors even for a common goal may consume significantly different amount of resources. For example, we can use a vision-based algorithm to localize the vehicle when GPS signal is absent, but the vision-based algorithm tends to consume more CPU resources. In that sense, we may consume more computing resources even for a degraded mode operation with GPS failure.

4. SUMMARY

The framework underlying autonomous vehicles must consider, among other needs, three important aspects: (1) flexible system design, (2) adaptive graceful degradation and (3) effective use of sensor/actuator modalities. By extending SAFER (System-level Architecture for Failure Evasion in Real-time applications), our previous work that addresses flexible system design, we have proposed a conceptual framework for autonomous vehicles to support adaptive graceful degradation and leverage various modalities of different sensors and actuators. We have motivated the necessity of our proposed framework with various usage scenarios based on our experiences with autonomous vehicle development.

5. REFERENCES

[1] Dozens injured in 259-car autobahn pile up. `http://www.thelocal.de/national/20090720-20701.html` as of Sep 24, 2012. The Local.

[2] Tartan Racing at Carnegie Mellon. `http://www.tartanracing.org` as of Sep 24, 2012.

[3] S. Azimi, G. Bhatia, R. Rajkumar, and P. Mudalige. Vehicular networks for collision avoidance at intersections. *SAE Int. J. Passeng. Cars âĂŞ Mech. Syst.*, 4(1):406 – 416, 2011.

[4] BMW ConnectedDrive. Take Over Please! `http://www.bmw.com/com/en/insights/technology/connecteddrive/2010/future_lab/index.html#/1/4` as of Aug 20, 2012.

[5] G. Buttazzo, G. Lipari, and L. Abeni. Elastic task model for adaptive rate control. In *Real-Time Systems Symposium, 1998. Proceedings., The 19th IEEE*, pages 286 –295, dec 1998.

[6] D. de Niz, G. Bhatia, and R. Rajkumar. Model-based development of embedded systems: The sysweaver approach. In *Real-Time and Embedded Technology and Applications Symposium, 2006. Proceedings of the 12th IEEE*, pages 231 – 242, april 2006.

[7] GM Press. Self-Driving Car in Cadillac's Future. `http://media.gm.com/media/us/en/cadillac/news.detail.html/content/Pages/news/us/en/2012/Apr/0420_cadillac.html` as of Aug 20, 2012.

[8] R. Guerrra and G. Fohler. A gravitational task model for target sensitive real-time applications. In *Real-Time Systems, 2008. ECRTS '08. Euromicro Conference on*, pages 309 –317, july 2008.

[9] J. Kim, G. Bhatia, R. R. Rajkumar, and M. Jochim. SAFER: System-level Architecture for Failure Evasion in Real-time Applications. In *Proc. of the 33rd IEEE Real-Time Systems Symposium (RTSS)*, 2012.

[10] J. Kim, K. Lakshmanan, and R. R. Rajkumar. Rhythmic tasks: A new task model with continually varying periods for cyber-physical systems. In *Proceedings of the 2012 IEEE/ACM Third International Conference on Cyber-Physical Systems*, ICCPS '12, pages 55–64, Washington, DC, USA, 2012. IEEE Computer Society.

[11] J. Kim, R. R. Rajkumar, and M. Jochim. SAFER: System-level Architecture for Failure Evasion in Real-time Applications. In *Proc. of the 4th Workshop on Adaptive and Reconfigurable Embedded Systems*, 2012.

[12] T. Lomax, D. Schrank, and S. Turner. 2011 urban mobility report. *Texas Transportation Institute*, 2011.

[13] S. Oikawa and R. Rajkumar. Portable RK: a portable resource kernel for guaranteed and enforced timing behavior. In *Proceedings of the fifth IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 1999.

[14] S. Pinkow. Continental Tests Highly-Automated Driving. `http://www.conti-online.com/generator/www/com/en/continental/pressportal/themes/press_releases/3_automotive_group/chassis_safety/press_releases/pr_2012_03_23_automated_driving_en.html` as of Aug 20, 2012.

[15] M. Schneider. Few guidelines exist on when to shut down roads, Feb 1st, 2012. Associated Press.

[16] R. Simmons. Inter Process Communication (IPC). `http://www.cs.cmu.edu/~ipc` as of Aug 20, 2012.

[17] S. Thrun. What we're driving at. `http://googleblog.blogspot.com/2010/10/what-were-driving-at.html` as of Aug 20, 2012.

[18] C. Urmson, J. Anhalt, H. Bae, J. A. D. Bagnell, C. R. Baker , R. E. Bittner, T. Brown, M. N. Clark, M. Darms, D. Demitrish, J. M. Dolan, D. Duggins, D. Ferguson, T. Galatali, C. M. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, S. Kolski, M. Likhachev, B. Litkouhi, A. Kelly, M. McNaughton, N. Miller, J. Nickolaou, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, V. Sadekar, B. Salesky, Y.-W. Seo, S. Singh, J. M. Snider, J. C. Struble, A. T. Stentz, M. Taylor , W. R. L. Whittaker, Z. Wolkowicki, W. Zhang, and J. Ziglar. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I*, 25(8):425–466, June 2008.