One solution of safety core in autonomous driving systems

Maksim Egelja, Marko Dragojević, Nikola Teslić, *Member, IEEE, Research Institute RT-RK,*

Nemanja Lukić, *Member, IEEE, Faculty of Technical Sciences, University of Novi Sad*

***Abstract*—Automotive industry now days needs one of the most sophisticated software, as that it brings m any new challenges to the world of engineering. Recently, autonomous driving became one of the biggest challenges of this field and it imposes high safety procedures.**

Work presented in this paper defines modern autonomous driving safety core which is responsible for detecting and controlling the autonomous system when safety critical event occurs either outside or inside the vehicle.

***Index terms*—Autonomous driving; ROS; safety core; driving state machine**

I. Introduction

Over the past years engineering focus was on providing driving assistance systems, now days autonomous driving became one of the most demanding software industry branch since engineers are facing with many complex problems, especially safety critical algorithms. Due to the fact that there are many sensors attached to the vehicle such as LiDAR (Light Detection and Ranging), GPS (Global Positioning System), full range and short range radars, multiple cameras there is a lot of data to collect and process in real time. Having this in mind engineers need special hardware units with very powerful processors, which are able to execute very complex algorithms[1]. Most components such as image processing, which are used for autonomous driving, are already solved as individual problems. Those components are highly important for this engineering field, but fusion of them still needs to be defined, in order to construct fully automated vehicle.

Assuming all facts mentioned above there is a lot of room for various malfunctions of either hardware or software. Here is where safety[2] part of autonomous driving system plays the role, it needs to prevent system of controlling vehicle when some of factors are incorrect.

This work was partially supported by the Ministry of Science and Technological Development of Serbia under Gant III_044009_1 TR36029

Maksim Egelja, Marko Dragojević and Nikola Teslić are with the Research Institute RT-RK, 23a Narodnog fronta, 21000 Novi Sad, Serbia (e-mail: maksim.egelja@rt-rk.com, marko.dragojevic@rt-rk.com, nikola.teslic@rt-rk.com).

Nemanja Lukić is with the Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obardovića 6, 21000 Novi Sad, Serbia (e-mail: nemanja.lukic@rt-rk.com).

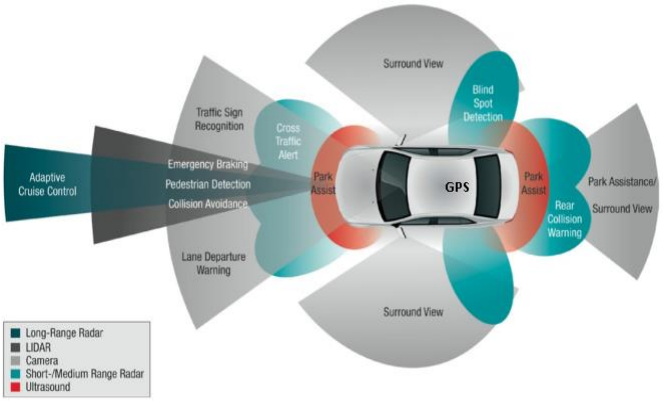

Fig. 1. Autonomous car with sensors.

Paper presents definition and implementation of autonomous driving system safety core.

The rest of material is organized as follows. At the beginning, section II describes software libraries and platform used in this research, while section III describes system architecture, where its sub sections present more detailed description of system. Section IV gives the experimental results of this work, whereas section V presents a conclusion of the demonstrated work and proposes some of its possible future improvements.

II. Platform and Software Libraries

Goal of this work was to define and make autonomous driving system safety core, capable of detecting and reacting to malfunctions of system's hardware or software components. AUTOSAR [3] is used as main development platform since it fulfills automotive safety standards. ROS [4] (Robot Operating System) and custom middleware were used as simulation and testing platform of this system. Today, ROS beside very high popularity in robot controlling software systems, finds usage in simulation and testing of autonomous driving systems. ROS specifies way of communication between nodes through specified topics. Nodes are completely independent processes and they are convenient to be used for simulating various parts of vehicle systems.

Safety core was realized using C programming language while the rest of work was realized using modern C++ programming language and tested using acceptance tests with fitnesse[5] framework, while units of this work are verified and tested using *gtest* [6] framework.

III. System Architecture

As explained before, ROS nodes were used as system units. Schematic view of realized safety core is shown in figure 2.

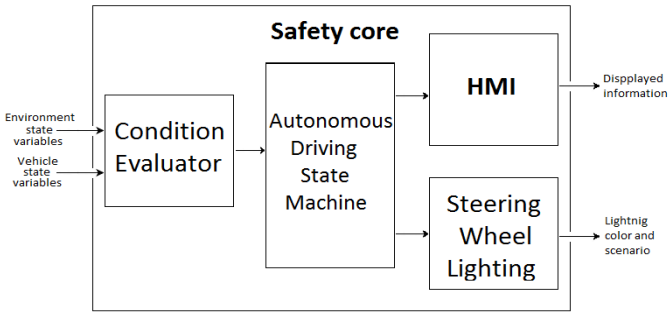

Fig. 2. Illustration of simulated hardware and signals between them.

Autonomous driving state machine is heart of the system. As its name says, this unit is in charge of automated driving system state. It switches states depends on various signals and internal parameters to keep vehicle and its passengers safe. Beside this unit there are condition evaluator, HMI(Human Machine Interface) and steering wheel lightning control units. Condition evaluator node analyses a lot of different signals such as door open signal, fasten seatbelt signal, etc. Once all signals are in right condition this part of system will allow activation of automated driving. Purpose of HMI and steering wheel lightning component is to inform the driver about autonomous driving system state and warn him about safety critical situations where human intervention is necessary.

All software components are implemented through AUTOSAR specific standard. Every software component is written using C programming language and has standardized architecture. Components are consist of receiver, core and sender part. Since AUTOSAR framework is used for implementation communication between components has to be done through particular memory zones through interfaces. Receiver part of component has to take all input interfaces from memory to components internal representations, then core part has to deal with component logic and sender part is there to write component output data back to communication memory. In further text more detailed explanation of each system unit is given.

A. *Autonomous driving state machine*

As previously mentioned autonomous driving state machine has to determine exact state of entire automated driving system.

It has several possible states:

- OFF the initial state, it changes when driver wants to activate autonomous vehicle pilot.
- READY state signify that all sensors and actuators as hardware units and all software components are ready to take over control of the vehicle. Also that means that all other conditions such as weather and traffic conditions are fulfilled.
- ACTIVE state represents fully automated vehicle driving. When system is in this state vehicle is capable of driving at certain speed and road conditions without human interaction.
- ACTIVE TOR (Take Over Request) state occurs when automated driving system detects situation which is impossible to handle by itself. In this state system notifies that human intervention is needed in certain period of time. There are three TOR levels, low, medium and high priority take over requests. Each priority level represents amount of time until system activates minimal risk maneuver.
- ACTIVE MRM (Minimal Risk Maneuver) is state of autonomous driving state machine which is entered when TOR timer e lapses without driver intervention. In this state system will guide the vehicle to complete safe stop. If driver take over control of vehicle during active TOR state, active MRM state will not be triggered.

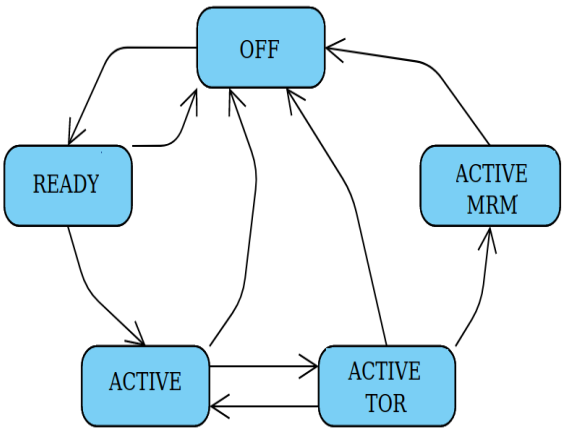

Fig. 3. Autonomous driving state machine state diagram.

As Autonomous driving state machine is FSM(Finite State Machine) it has its state transfer functions. Figure 3 shows its state transfer diagram using these functions. State transfer functions defines to which state this component under certain conditions will transfer from the current one. In further text description of state transfer is given.

From the off state system goes to ready when user wants to activate automated pilot only if all hardware and software units are working properly. If something goes wrong system will be set back to off state or everything went before goes to active state system will check all road and weather conditions. When vehicle figure out road situation or some of sensor reports an error state system will switch to active tor state where driver is informed about the problem and amount of time he has until minimal risk maneuver is performed. If driver takes over the control until tor time period and handles potentially risky situation, vehicle will continue to drive by itself, system will be back to active state. From all above mentioned states system can go back to off if user wants to switch of automated driving pilot.

B. *Condition evaluator*

This software component has to observe a large amount of signals and make sure that neither hardware or software

components are working properly in context of automate vehicle pilot. Condition evaluator has to check some very basic signals such as are all doors on vehicle closed or are all passengers are fasten seat belt beside these there are some very complex signals to analyze such as weather conditions observed through camera or driver fatigue. All of signal checking hast to be done in real time. Having this in mind we had to provide communication interface fast enough and a lot of computing power and optimizations. Due to these facts we chose ROS as our communication platform and classic AUTOSAR as implementation platform.

Weather and driver fatigue signals analysis is done through multiple cameras. Before signal from camera even comes to condition evaluator a lot of image processing has to be done how could we make filtration and use only the most important data.

After all checking finishes this software components sends its output to autonomous driving state machine, it contains of state machine state to be allowed and TOR time if some of the signals to be checked is in error state.

C. *Human Machine Interface*

Now days we have modern, digital HMIs present on the automotive market. As all digital systems it provides as a huge amount of possibilities to be shown. In figure 4 illustration of modern human machine interface is given.

Fig. 4. Illustration of HMI used in research.

Beside its regular functions, to show vehicle speed and engine RPM(Rotation Per Minute) HMI in this system has few more functionalities. It has to inform driver about vehicle state in automated diving mode. HMI shows when automated driving mode is activated and if preciously mentioned TOR occurs lets the driver know. This software component is in direct communication with autonomous driving state machine as a heart of a system. It is highly important in this system because provides information about autonomous driving system to driver which is crucial for safety.

D. *Steering wheel lightning*

Today we are witnesses of LED(Light Emitting Diode) lightning era. As a modern system autonomous car has large amount of LEDs built in steering wheel. These LEDs have safety critical information function. Figure 5 shows the concept of LED steering wheel.

Fig. 5. Illustration steering wheel when autonomous pilot activated.

When system is activated and vehicle is driving on its own steering wheel is emitting green light, when some TOR occurs depending on its priority steering wheel is changing its color to blue, for low priority TORs, yellow, for medium priority and red, for the most urgent situations. Also red color is used when MRM is triggered.

Beside colors this component also changes lightning scenario based on TOR priority:

- Solid light - used when system is active.
- Pulsing light - used when medium priority TOR occurs.
- Blinking light - turns on when high priority TOR has been triggered.

Since modern LEDs are very powerful source of light and vehicle can be driven in very bright or extremely dark environment, dimming has to be performed. This is done by reading multiple brightness sensors which measures vehicles environment lightning and establishing minimal amount of light visible to human eye under current lightning conditions, since driver who is responsible for taking over control of the vehicle can always notice the safety warning and at the same time not be blinded by too high steering wheel illumination.

IV. RESULTS

Since this research is realized in simulated environment and ROS is used as communication platform challenge was simulation and acceptance testing. All previously described software components needs to have wrapper implemented which is used to extract input and output data since software components are implemented according to AUTOSAR standard. Beside extraction these wrappers are used to transfer data over the ROS topics and instantiate ROS communication nodes for each of the software components.

Acceptance tests are performed using fitnesse open source framework. Tests are simulating various hardware or software malfunctions and verifies system behavior under given set of input parameters. In figure 6 results of acceptance tests is shown.

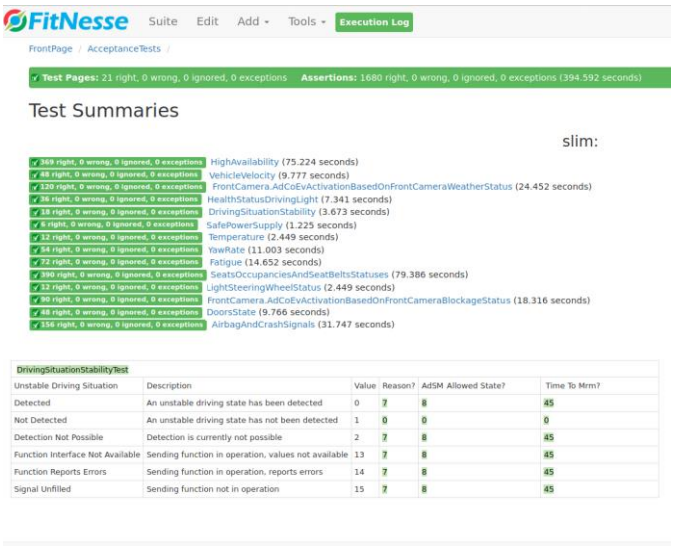

Fig. 6. Acceptance test results.

These tests are testing system form high level, they simulates vehicle systems malfunctions and sends that simulated signals to condition evaluator which has to process the data and forward it to autonomous driving state machine which puts system in appropriate state depending on failure level. When state is determined, it is sent to steering wheel LEDs and human machine interface in order to inform the driver. Acceptance tests are checking all the data flow from simulated failure to information which driver receives through system components and make sure that safety core is performing without any mistakes.

Integration and unit testing is performed with usage *google test* and *google mock* library. Unit tests are verified software components functionalities while integration tests are verified connection between components and system functionalities all together.

V. Conclusion

Contribution of this work is successful implementation of safety core which can be used in highly automated vehicle pilots.

Investigation shows how one way of automated driving safety core implementation. Demonstration includes definition and development of software components which are mandatory for safety critical events which occurs while automated vehicle pilot is active.

Demonstrated safety core include following functionalities:

- Detection of hardware or software malfunctions using various health checking methods such as watchdog timers, direct health state signal checking, etc.
- Detection of complex traffic situation or bad weather conditions where sensors attached to vehicle cannot work properly.
- Informing the diver about detected situations and requesting take over in certain period of time.
- Performing minimal risk maneuver if driver does not take over control of the vehicle and led it to complete safe stop.

Since this research shows solution of safety core implementation in simulated environment, one of possible future improvements could be adapting the system to run on specialized hardware. Also more sensors and signals can be added to the system to make it more complex. As ROS is used as base communication platform of the system, it can be adapted to work on AUTOSAR Adaptive[7] platform which is more often used in modern vehicles software solutions.

References

[1] J. Levinson , J. Askeland , J. Becker , J. Dolson , D. Held , S. Kammel , J. Z. Kolter , D. Langer , O. Pink , V. Pratt , M. Sokolsky , S. Ganymed , D. Stavens , A. Teichman , M. Werling , S. Thrun "Towards fully autonomous driving: Systems and algorithms", IV, pp.163-168 , Baden-Baden, Germany, June 2011.

[2] L. Daxue, A. Xiangjing, S. Zhenping, H. Hangen "Active Safety in Autonomous Land Vehicle", PEITS, pp. 476 - 480, Guangzhou, China, August 2008.

[3] AUTOSAR Technical overview, 2007, AUTOSAR Specification Release 3.0, Retrieved on 28/11/2007.

[4] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng "Ros: an open-source robot operating system", ICRA Workshop on Open Source Software, vol. 3, no. 3.2, pp. 5, June 2009

[5] G.K. Hanssen, and B. Haugset, Automated Acceptance Testing Using Fit, in 42d Hawaiian International 20 Conference on System Sciences (HISS'09). 2009, IEEE Computer Society: Hawaii, USA. P.1-8.

[6] J. Swaminathan "Test-Driven Development" in "Mastering C++ Programming", Birmingham, United Kingdom, Packt Publishing, September 2017

[7] S. Fürst and M. Bechter, "AUTOSAR for Connected and Autonomous Vehicles: The AUTOSAR Adaptive Platform", 2016 46th Annual IEEE/IFIP Int. Con. on Dependable Systems and Networks Workshop (DSN-W), 2016.