One solution of keep lane function and curvature calculation with SD map data using ROS

Maksim Egelja, Stevan Stević, Nikola Teslić, Member, IEEE, Research Institute RT-RK,

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

***Abstract*—Software in automotive industry today is a very popular branch and brings many new challenges to the world of engineering. Recently, autonomous driving became one of the biggest challenges of this field. Goal is to develop a system capable of controlling vehicle almost completely independently.**

Work presented in this paper defines modern autonomous driving algorithm, which includes functionality of keeping autonomous vehicle in appropriate lane using SD (Standard Definition) map data instead of HD (High Definition). This algorithm is based on data fusion from camera senor and map data.

***Index terms*—Autonomous driving; ROS; vehicle control based on camera and map; keep lane function; curvature calculation**

I. Introduction

Recently, autonomous driving became one of the most popular software industry branches since engineers are working on providing highly automated driving systems, while over the past few years more focus is given on providing driving assistance. Due to the fact that there are many sensors attached to the vehicle such as GPS (Global Positioning System), LiDAR (Light Detection and Ranging), full range and short range radars, multiple camera, ultra sonic sensors[1], there is a lot of data to collect and process in real time. Having this in mind engineers need huge computing power, special hardware units with very powerful processors, which are able to execute very complex algorithms. Sensors and their field of operation are illustrated in figure 1.

This work was partially supported by the Ministry of Science and Technological Development of Serbia under the project No. TR36029, year 2019

Maksim Egelja, Stevan Stević and Nikola Teslić are with the Research Institute RT-RK, 23a Narodnog fronta, 21000 Novi Sad, Serbia (e-mail: maksim.egelja@rt-rk.com, stevan.stevic@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, dragan.samardyija@rt-rk.uns.ac.rs).

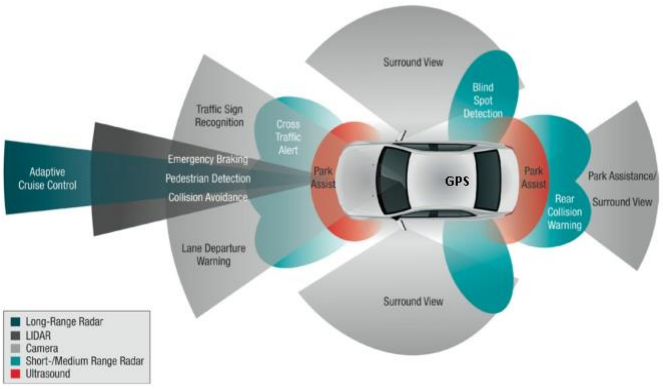

Fig. 1. Autonomous car with sensors.

Most components such as image processing[2], object detection and tracking[3], 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. In order to achieve this goal, engineers need to combine a great deal of research from different areas to produce highly automated vehicle pilot. Paper presents definition and implementation of road curvature calculation using fusion of camera sensor and SD instead of HD map data.

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 make autonomous vehicle system capable of controlling vehicle on roads where HD maps are absent. ROS [4] (Robot Operating System) was used as a platform of this system. Today, ROS is a very popular software system used for robot control, where the self driving vehicle can be defined as robot. Units of a system are implemented as ROS nodes where nodes are processes that

are completely independent and can communicate with each other through specified topics.

Complete work was realized using modern C++ programming language and tested using acceptance tests with simulator and 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 the system is shown in figure 2.

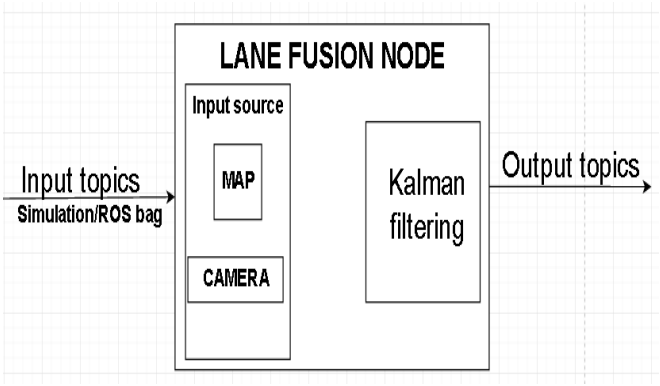

Fig. 2. Illustration of simulated hardware and signals between them.

Lane fusion node is main part of this autonomous vehicle subsystem. As its name says, this node is in charge of fusion data collected from camera and maps, it works very efficient when using HD maps, goal of this research is to make similar efficiency while using SD maps with less data available. Besides Lane fusion there are other nodes present, they are mainly used for providing data to lane fusion node. Since verification of algorithm presented in this work is done using acceptance tests, there are nodes which are in charge of simulation framework. In further text, more detailed explanation of system and HD maps is given.

A. HD map

Todays, well known, maps which are used in navigation devices and mobile phones are primary meant for humans, due to the fact that they can understand simple visual or vocal instructions provided from various devices. In autonomous diving era, where machines have to make decisions on the road, need for new set of maps is occurred. These maps are specially built for robotic systems, specifically they have very high precision, at centimeter level. This high precision is needed because self-driving car needs extremely precise instructions how to maneuver around 3D (3 Dimension) space. Tolerance for error is high in most of every day cases but there are a lot of situations where car has no room for error such as driving on the road build next to the cliff. Maps should contain where the lanes are, what is position of all traffic signalization, where the road boundaries are, maps also must contain data such as where the curves are and how high the curves are. When it comes to a self-driving car to precisely locate itself, lot of sensors are involved, visual sensors have a lot of limitations. Here the LiDAR sensor plays a role it precisely measures the depth or distance in a 3D space with lasers. However, both LiDAR and cameras as visual sensors, typically work together, running very fast, multiple times per second.

B. HD maps layers

An HD Map is organized into five layers. They are a base map (standard definition map), geometric map, the semantic map, map priors, and real-time knowledge as shown on figure 3.

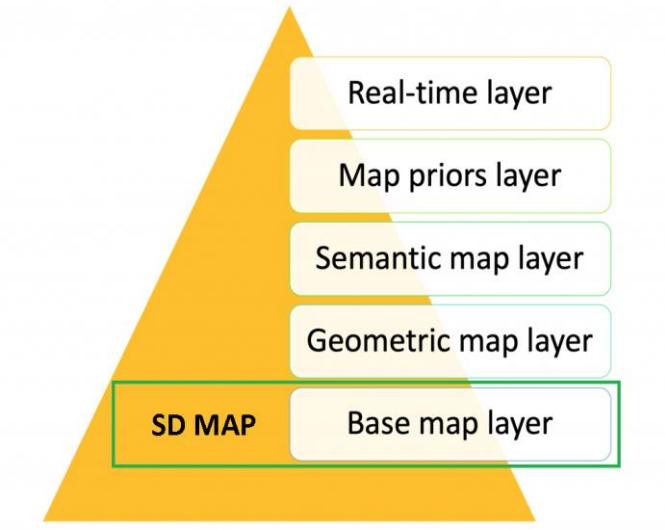

Fig. 3. Illustration of HD map layers.

Geometric Map layer contains of raw data collected by various visual sensors such as LiDAR and camera. Data is reconstructed and stored in this map layer. This layer contains information about positions of static objects near the road, e.g. buildings.

Semantic Map layer is geometrics layer upgrade were semantic objects are added. Semantic objects can be either 2D or 3D objects such as intersections, lane boundaries, parking places, traffic signalization, etc. These objects contain information about lane change restrictions, one way roads, usual traffic speeds, etc.

Map priors layer contains dynamic information such as average waiting time on traffic lights, probability of finding parking spot, etc. Autonomy algorithms commonly use this data in models as inputs and combines them with other real-time information.

Real-time knowledge layer is dynamically updated contains real-time traffic information. This data can also be shared in real time between autonomous vehicles.

C. Kalman filer

Kalman filtering[7], also known as LQE (linear quadratic estimation) is an algorithm that uses measurements observed over the time, such as from sensors, containing raw senor data with statistical noise and other interference. This algorithm produces estimates of unknown variables by estimating a probability distribution over variables. These estimates are more accurate than those based on a single measurement alone. As such, it is a common sensor fusion and data fusion

algorithm.

The Kalman filter has numerous applications in engineering, mostly for guidance and navigation. These filters also are one of the main topics in the field of robotic motion planning and control, and they are included in trajectory optimization.

The algorithm works in a two-step process. In first, prediction step, the Kalman filter produces current system state variables estimates, along with their uncertainties. When next measurement arrives, usually corrupted with random noise and some other errors, estimates are updated using a weighted average method, which means that more weight will be given to estimates with higher certainly. Algorithm is recursive and can be run in real time, using only the present input measurements and the previously calculated state with its uncertainty matrix. There are extensions and generalizations of the algorithm developed, such as extended Kalman filter and unscented Kalman Filter which are used for nonlinear systems.

D. Lane fusion node

The lane fusion node is component in the perception module of autonomous driving stack, responsible for providing fused lanes output, which is the internal representation of lanes, lane geometry and lane attributes. Lane fusion fuses sensor and map data according to provide fused lanes output shown on the figure 4.

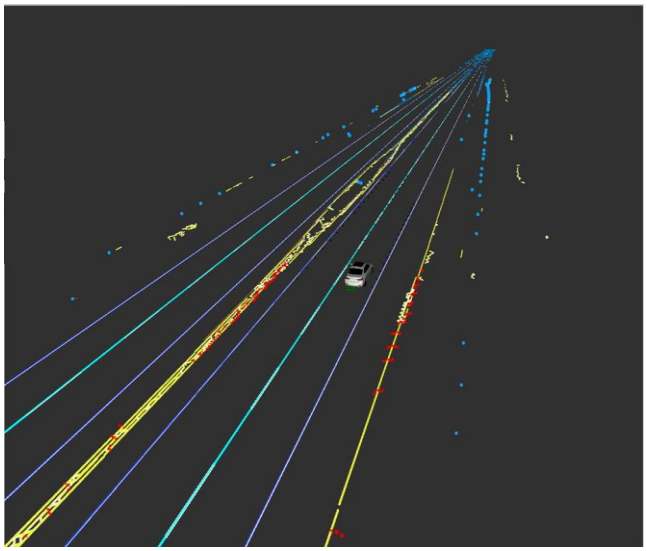

Fig. 4. Graphical view of fused lanes.

This node is using Kalman filtering algorithm for making predictions if sensor data is not very reliable, e.g. camera lens is dirty or there are bad weather conditions, fog, sun shining directly to camera sensor and etc.

Since we are using ROS as our development platform lane fusion component is distributed on two layers, the communication layer, handles all information exchange through specified ROS topics and the domain layer containing sensor and map data processing units. Figure 5 shows architecture diagram of lane fusion component.

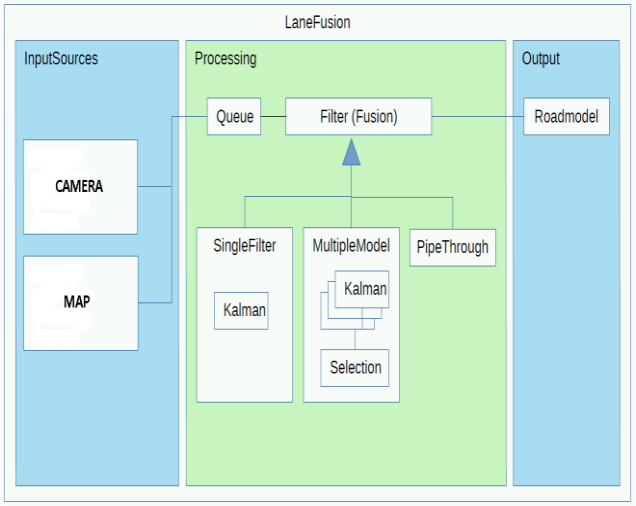

Fig. 5. Lane fusion software architecture diagram.

Communication layers main class, LaneFusionApplication, is responsible for communication with other autonomous driving system components. It includes communication interface objects used for publishing and subscribing messages.

Domain layers main class, LaneFusion, is the entry point in this layer. It comprises objects for each input source and a fusion algorithm called FusionStrategy. This part of a system is triggered cyclically by the LaneFusionApplication. When triggered, it sets up a queue of observations and gives it to fusion algorithm.

E. Map source

Map source is subcomponent of lane fusion node, it is in charge of providing map data to fusion algorithm. Also it calculates road curvature using data available.

HD map data curvature calculation is done by observing center lines from each lane on the road while curvature calculation based on SD map data is done using only one center line per road. All curvature calculations are done by using *intel AdLib* library, part of code where curvature calculation performs is attached in listing 1.

```
AdLibCurveFitting<Point2d> curve_fitting(polyline);

if (curve_fitting.GetStatus() == 0)
{
   auto curvature =
curve_fitting.CalculateCurvatureAt(arc_length.value());

   if (curvature.has_value())
   {
      return curvature.value();
   }
}
```

Listing 1 Curvature calculation C++ code

IV. Results

Testing of system is done in multiple ways, firstly acceptance tests are performed to check lateral displacement of the autonomous vehicle.
Tests are executed multiple times and in different scenarios:

- Straight road only
- Straight then curved road
- Straight then curved then straight road
- Curved road only with different curve radius
- Combined scenario with real world road example

All acceptance tests are done using fitnesse open source framework by checking vehicle driven distance and lateral vehicle displacement, results of combined test scenario are shown in figure 6.

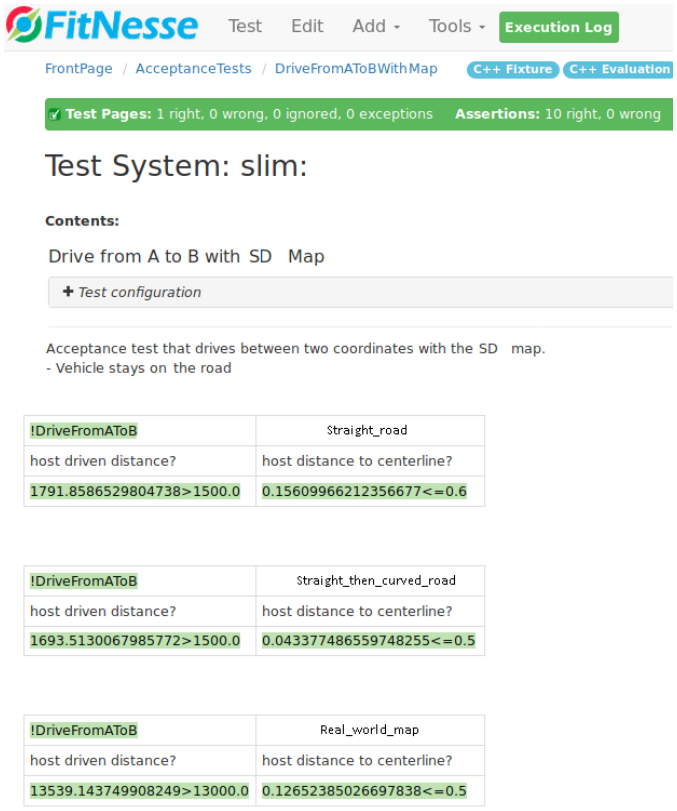

Fig. 6. Acceptance test results.

Unit and integration testing is done in this research too. It is done by using *google test* and *google mock* library. Mocking the objects is designed with use of DI (Dependency Injection) design pattern.

Further data analysis is done in this research too, curvature calculated using SD map is compared to values obtained from HD map. In the upper diagram of following figure 7 graphical view of comparison curvature values over the time is given, while the lower diagram shows curvature comparison with its standard deviation.

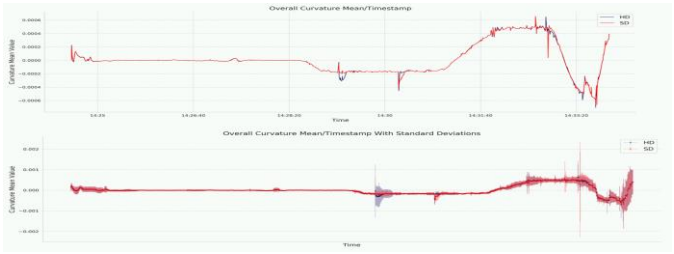

Fig. 7. Output of testing script used to check system functionality.

V. Conclusion

Contribution of this work is successful implementation of usage SD instead of HD map data in self-driving cars.
This research shows how fusion of SD map and camera data can be done in autonomous driving system. Demonstration includes definition and development of algorithms, based on data fusion, for vehicle control using camera and SD map data. Algorithms shown in this investigation are:

- Keep lane function where software components are keeping a vehicle in current lane.
- Curvature calculation where software units calculate curvature of the road in front of vehicle using SD map data which consist of much smaller amount of information compared to HD map.

The fact that embedded systems like self-driving car, has a limited amount of memory, usage of SD data is very useful, also all parts of the world will not have HD maps available, so system must rely on only available SD map data. By selection of ROS as the base platform of this work, program code is platform independent. Next step could be transferring the whole system to other platforms such as *AUTOSAR*[8] framework.

References

[1] H. Somogyi, D. Pup, P. Körös, A. Mihaly, A. Soumelidis "Research of Required Vehicle System Parameters and Sensor Systems for Autonomous Vehicle Control", 12th SACI, Timisoara, Romania, pp. 27-32, May 2018
[2] H. Kurihata "Rainy weather recognition from in-vehicle camera images for driver assistance", IEEE Proceedings. Intelligent Vehicles Symposium, Las Vegas, NV, USA, pp. 205-210, June 2005.
[3] T. Ogawa, H. Sakai, Y. Suzuki, K. Takagi and K. Morikawa, "Pedestrian detection and tracking using in-vehicle lidar for automotive application", Baden-Baden, Germany, pp. 734-739, June 2011.
[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] J. Xi, Kalman filter and its application foundation[M], National Defense Industry Press, 2001.
[8] AUTOSAR Technical overview, 2007, AUTOSAR Specification Release 3.0, Retrieved on 28/11/2007.