Automated valet parking with EB robinos
Automated valet parking with EB robinos

Highly automated driving (HAD) raises the complexity within vehicles tremendously due to many different components that need to be coordinated and combined. A change in current state-of-the-art development procedures is needed to manage this increased complexity. As a result, the use of a software framework that is based on a standardized architecture with open interfaces can reduce this complexity, but also cost and effort. This leads to higher competitiveness and, vice versa, the delivery of better products to the end consumer. The industry needs to shift its focus and also think about best practices in terms of functional software rather than only in terms of hardware architectures.

EB robinos

The automotive industry still faces the challenge of how to coordinate and combine separate parts, so that they cover all needs – from functionality to efficiency as well as safety and security. A high degree of complexity results in the need to incorporate elements such as sensors, fusion, function, and control components as well as actuators from the perspective of both hardware and software. EB robinos, an application layer software framework with open interfaces and software modules for highly automated driving, is especially designed to reduce the complexity of highly automated driving and application development.

It enables appropriate combination and interaction of all the related elements that form an automated driving system, from development to mass production.

Figure 1: The EB robinos architecture
Automated valet parking with EB robinos

EB robinos architecture

EB robinos comprises a software framework based on an architecture with open interfaces and software modules for highly automated driving.

The software architecture, depicted in Figure 1, follows a classical robotic architecture based on the sense-plan-act principle outlined by Boyd, 1976. First, the sensor data is converted into abstracted data structures to render subsequent layers independently of the individual sensor type. It is, thus, easier to replace sensors or enhance the sensor setup with new sensors. This sensor data is processed by different modules in the following sensor data fusion step. The results of the sensor data fusion process are shown in function-specific views. These views summarize the fused sensor data to provide a specialized view for the functions that follow. The simplest form has only one functional view for all behaviors. However, this often results in a view that becomes unusable because it contains a lot of information that is not needed by many of the behaviors. The architecture therefore provides different views for different behaviors. This also enables the developer to tune the view to the special needs of a behavior. For example, an emergency brake behavior needs the object hypothesis immediately after the initial detection (e.g. to pre-charge brakes), whereas an adaptive cruise control may require a more stable object hypothesis, since rapid reaction is not so much of an issue. Another advantage of different views is the ability to adapt the amount of data to different system architectures as the performance of communication buses may differ.

The HAD functions are provided by the center part of the architecture. This architecture splits the overall system into small behaviors, each for a specialized situation. This enables the developer to concentrate on a specific task. Moreover, the system can be enhanced over time by the addition of more and more behaviors alongside the original ones. A need to moderate the access to actuators comes up, if multiple vehicle control behaviors are used. This need is fulfilled by situative behavior arbitration. It collects the demands of all behaviors for controlling the car and decides which one or which set of behaviors is allowed to access the actuators.

These commands are then executed by the components of the motion management or the HMI management layers. The system applies vehicle-independent control and optimization algorithms in these layers to execute the requested commands. These vehicle-independent commands are then converted into vehicle/actuator specific commands by the abstraction layer on the right-hand side.

Automated valet parking

with EB robinos

Automated valet parking is a fully automated function. Its development and application is a manageable task that requires many software algorithms that are essential for fully automated driving of vehicles in other environments (e.g. highways, residential areas). The vehicle is driven at low speed within a local environment (e.g. parking lot). Therefore, it is possible to use this function in a private parking area. According to local country regulations, a safety driver might be obligatory in the vehicle at all times.

Function definition

The automated valet parking in this example executes the following function definition:

The driver starts the application. It perceives its surroundings and detects static as well as slowly moving obstacles (e.g. pedestrians). This data is used to adapt the path of the vehicle from its starting position to a previously set parking spot. The vehicle then follows this path until it has reached its final destination. The path is updated continuously and adjusted, if necessary, to ensure that no collisions take place during moving.

Realization

The function described in previous definition can be easily realized by using the EB robinos software framework. The requirements for realization are:

- The test area is located in a private area
- The test vehicle is equipped with high precision environmental perception sensors (e.g. high definition RADAR or LIDAR) as well as input data for the positioning component (e.g. IMU/GPS) that are accessible by a car PC
- The test vehicle actuators are fully accessible by a car PC
- The test vehicle sensor and actuator data is converted to the EB robinos data structures in the abstraction layers shown in Figure 1
Automated valet parking with EB robinos

In this example, Valerie, one of EB’s highly automated driving (HAD) test vehicles (see Figure 3), is used to realize this application. She is a regular production vehicle additionally equipped with a high precision DGPS receiver, surround view cameras (not used for this application), ultrasonic sensors, and IBEO Lux LIDAR sensors. It also is retrofitted with actuators for disabled people that provide full drive by wire features to the control PC.

This application does not need all the EB robinos modules, as shown in Figure 2. It is essential to know the position of the vehicle in order to plan and track a path from the actual position to the destination. This precondition is realized by the positioning module, which fuses the vehicle motion data (e.g. tire sensors) with a global position received by, e.g. a DGPS sensor. The other tasks are easier to perform if the exact vehicle position is known. The next task in the function definition is to monitor the area around the vehicle. This is done by the grid fusion module that, inter alia, comes with a built-in Dempster-Shafer occupancy fusion (Wu et. al. 2002). The data from different environmental perception sensors can be fed directly into the grid fusion sensor models to create a representation of the vehicles surroundings.

The grid fusion data is then accessed by the path planning module which allows the system to plan a path from A to B, i.e. from the current position to the set destination. This task is fulfilled by various algorithms that can be run in parallel if necessary. In this example, a hybrid A*-algorithm is used for simplicity. It searches for the path to the destination by taking the motion model of the vehicle into account and avoids all perceived obstacles.

The latitudinal and longitudinal controller modules calculate the actuating variables for the vehicle’s pedals and steering wheel if a path has been found. The software component that controls the overall procedure is part of the behavior. It is named Mission Control and contains a state machine that tells the vehicle when to start the application (according to the function definition) as well as stopping it when the final position has been reached.

The data from the sensors is fed into the grid fusion module which is configured to cover an area of 50m x 50m at a resolution of 10cm. These parameters have proven to be sufficient for the sensor data and the application. The content of the grid fusion module is shown in Figure 4. It follows the Dempster-Shafer paradigm and derives the drivability of a cell from the LIDAR and ultrasonic sensor measurements.
The path planning module plans the path based on this data. The algorithms avoid all occupied cells, which are by definition not drivable. In Figure 5, the algorithm has planned a path that requires it to reverse first and then follow an S-curve to reach its parking position.

Summary

This technical paper shows how EB robinos can be used to implement an automated valet parking application. After a short introduction on EB robinos, the function definition of the valet parking system is described, which resulted in a subset of components required. A description is given of how these components can be used for automated valet parking. Finally, the EB HAD test vehicle is introduced and the implementation of automated valet parking in this vehicle is described.

EB robinos accelerates the HAD development by applying ready to use software components. HAD development is made easier by the correct combination and configuration of the available software components with well-defined interfaces. The EB robinos approach enables car manufacturers to focus on the development of end consumer driving experience and HAD functions.

Bibliography


Author:
Dr.-Ing. Sebastian Ohl
Senior Expert Driver Assistance
About EB Automotive

Elektrobit (EB) is an award-winning and visionary global supplier of embedded and connected software solutions and services for the automotive industry. A leader in automotive software with over 25 years serving the industry, EB’s software powers over 70 million vehicles and offers flexible, innovative solutions for connected car infrastructure, human machine interface (HMI) technologies, navigation, driver assistance, electronic control units (ECUs), and software engineering services. EB is a wholly owned subsidiary of Continental.