A functional architecture for autonomous driving

Sagar Behere
Kungliga Tekniska Högskolan
Brinellvägen 85
Stockholm, Sweden
behere@kth.se

Martin Törngren
Kungliga Tekniska Högskolan
Brinellvägen 85
Stockholm, Sweden
martint@kth.se

ABSTRACT

As the Technology Readiness Levels (TRLs) of self-driving vehicles increase, it is necessary to investigate the Electrical/Electronic(E/E) system architectures for autonomous driving, beyond proof-of-concept prototypes. Relevant patterns and anti-patterns need to be raised into debate and documented. This paper presents the principal components needed in a functional architecture for autonomous driving, along with reasoning for how they should be distributed across the architecture. A functional architecture integrating all the concepts and reasoning is also presented.

Keywords

Autonomous driving, functional architecture, E/E architecture

1. INTRODUCTION

Autonomous driving is considered to be the 'next big thing' in the automotive domain. From the universities during the DARPA Grand and Urban challenges in 2004/2007 to technology showcases like the Google self-driving car, autonomous driving technology has shown a steady maturation. Today, most major passenger car OEMs across the world (Daimler, BMW, Audi, Ford, Nissan, Volkswagen, Volvo, ...) have active development projects in this area, and typically exciting demonstrators are exhibited every year at popular public events like the Consumer Electronics Show (CES) in Las Vegas, USA.

The autonomous driving demonstrators developed so far involve some sort of 'perception and higher intelligence' plugged on top of a base vehicle platform that usually incorporates computerized control of functions like propulsion and braking. As the technology readiness levels (TRLs) [1] increase, and autonomous features move closer to series production, it is necessary to take a deeper look at the electrical/electronic (E/E) architectures for autonomous vehicles. Factors like horizontal and vertical control hierarchies, distribution of functionality, arbitration and conflict resolution, fault propagation and isolation of system failures, system safety, optimality of implementation, cognitive complexity etc. need to be considered for each particular deployment related to autonomous driving.

Unlike the domain of task-specific algorithms (e.g. antilock braking), most of which can be objectively and quantitatively assessed at fine-grained levels, the domain of systems architecting in the automotive world is still driven largely by qualitative aspects like legacy considerations, brand values, organizational and development processes, commitments to specific technology partners and so on. The enabling technologies related to sensing, computation, communication and actuation merely form a baseline, beyond which the architecture diverges from one OEM to another. The element of 'fuzziness' in the process leading to an architectural definition, together with the fact that there is rarely a definitively best solution to any given architectural problem, constitutes a potential problem area in systems architecting. The goal of this paper is to assist architects of autonomous driving systems by raising into public debate the various considerations and solution possibilities that influence the architecture of a self-driving vehicle. These may then be evaluated within the contexts of individual projects and associated constraints, leading to specific architectural solutions.

This paper makes three contributions to the literature in this area: 1. A discussion of the key elements in a functional architecture for autonomous driving 2. A proposal on the division of the architecture into layers, and reasoning on the distribution of the architectural elements across these layers, and 3. A proposed functional architecture for autonomous driving. This architecture is undergoing continuous validation since four years at KTH The Royal Institute of Technology in Stockholm, Sweden. It has been applied to three distinct vehicle domains: A commercial heavy-duty truck which was upgraded with self-driving functionality [11], an ongoing autonomous passenger car project [3] wherein an existing vehicle in the OEMs portfolio is being upgraded towards autonomy, and a novel, legacy free, drive-by-wire electric concept vehicle developed at KTH [15]. The reason for such a broad applicability of the architecture is because it only represents the functional view of the system. The architecture focuses on one vehicle only; not a collection of vehicles as in cooperative driving and other Intelligent Transport System (ITS) scenarios. Further, the scope is restricted to the vehicle motion specific subsystems only.

WASA'15, May 4, 2015, Montréal, QC, Canada.

ACM 978-1-4503-3444-0/15/05 ...$15.00.
http://dx.doi.org/10.1145/2752489.2752491.

We use the term 'functional architecture' with a definition corresponding to the notion of 'functional concept' in the ISO26262 functional safety standard [7]. The standard defines a functional concept as, *"specification of the intended functions and their interactions necessary to achieve the desired behavior"*. A functional architecture then refers to logical decomposition of the system into components and subcomponents, as well as the data-flows between them. It does so without reference or prejudice to the actual technical implementation of the architectural elements in terms of hardware and software. An analogous term to functional architecture is 'functional view' of the architecture description. This term is recommended by ISO 42010 [8], and pertains to the architectural description of software intensive systems, from a functional *viewpoint*. In this paper we will use the terms 'functional architecture', 'functional view' and their combination 'functional architecture view' (FAV) synonymously. The FAV closely corresponds to the functional view of the system software architecture, since autonomous systems are highly software intensive.

In the simplest of senses, an autonomous driving system may be thought of as a "cognitive driving intelligence" layered on top of a basic "vehicle platform". The cognitive intelligence is responsible for perceiving the environment, generating a feasible motion trajectory through the environment, and manipulating the vehicle platform in order to achieve the desired motion. Given this notion, some interesting questions from an architectural viewpoint are

1. What should be the principal components of the FAV of an autonomous driving system?
2. What are the various ways of distributing or allocating the FAV components across different layers of the architecture?

In the subsequent sections, we attempt to provide answers to these questions, while keeping in mind considerations which we have repeatedly experienced to be important. These considerations are: clear description of functionalities, safety, separation of concerns, and decoupling of components. The latter two facilitate change and reuse of the components, which make for flexible and evolvable architectures.

2. FUNCTIONAL COMPONENTS

We have opted to split the principal FAV components of the motion control part of the autonomous driving system into three main categories, as shown in Figure 1. These categories are related to

1. Perception of the external environment/context in which the vehicle operates
2. Decisions and control of the vehicle motion, with respect the external environment/context that is perceived
3. Vehicle platform manipulation which deals mostly with sensing and actuation of the Ego vehicle, with the intention of achieving desired motion

Each category has several components, whose functionality (from a strictly architectural perspective) will now be described.

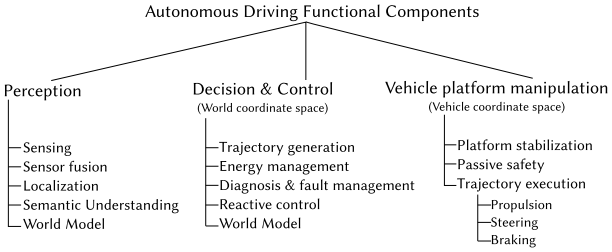

Figure 1: Components of a FAV of an autonomous driving system

2.1 Perception

A commonly heard phrase in the robotics community is, "Sensing is easy, perception is difficult.". Sensing means gathering data on physical variables using sensors, while perception refers to the semantics (interpretation and "understanding") of that data in terms of high level concepts relevant to the task being undertaken. As such, sensing is just one part of an overall perception system.

The **sensing** components can be categorized into those sensing the states of the ego vehicle and those sensing the states of the environment in which the ego vehicle operates. (sometimes referred to as the *internal* and *external* environments). A second and relevant categorization of sensor components, from the viewpoint of systems integration, depends on the amount of processing needed to extract relevant information from the sensor data. In our experience, this usually depends on the Technology Readiness Levels (TRLs) of both the sensor and the integrated system. At lower TRLs, in highly experimental vehicles and/or sensors, it is more common to work with raw sensor data and its filtering, estimation, fusion, association, and possible classification operations are performed as distinct and important parts of the overall system. An example of this is the processing of data from some multi-beam laser rangefinders, which return angle and distance measurements for each beam, typically at rates of 10Hz or more. At the opposite end of the spectrum are high TRL sensors from automotive vendors, which come packaged with data processing elements. Such sensors may directly output detected objects in the environment, along with relevant attributes of the detected objects like relative position, velocity, acceleration etc. Some sensor providers offer even higher level information, like the class of the detected object (car, truck, motorcycle, pedestrian, ...). The reason why this type of categorization is important is that it permits the systems integrator to treat the sensor component as a blackbox[1] and the sensor's output may be routed directly to the semantic understanding component or even directly to the world model. High TRL sensors are more commonly found in the automotive industry in late stage prototypes and when selecting or using them, emphasis often shifts to extra functional properties like failure probabilities, confidence levels of output data, and common mode failures with other sensors. This is because, factors like training datasets which a vendor may have used, data processing algorithms and their properties like time constants etc. tend to be a grey area for the systems integrator, yet the same factors may have a non-trivial impact on the analysis of safety

[1]Although some sensors retain the ability to transmit raw as well as processed information.

properties of the system. In comparison, when processing of raw sensor data is an explicit part of the overall system, the details tend to be more transparent, although that does not necessarily make the analysis is easier.

The **sensor fusion** component, as the name indicates, considers multiple sources of information to construct a hypothesis about the state of the environment. In addition to establishing confidence values for state variables, the sensor fusion component may also perform object association and tracking. Association refers to correlating pieces of information from multiple sensors to conclude that they refer to one and the same object. The process of tracking generates information about an object over a series of temporal readings. This information can be used either to track an object's attributes (e.g. relative velocity), or to classify the object into categories in a subsequent block (e.g. a sequence of readings is more likely than a single reading, to reveal an object as a pedestrian.). Finally, for certain system configurations, the sensor fusion block may also be used to eliminate some un-associated objects, and data that is strongly likely to be superfluous or noise. This reduces the computation and communication load on subsequent components, like the decision and control, which need to work with the perceived data.

The **localization** component is responsible for determining the location of the vehicle with respect to a global map, with needed accuracy. It may also aid the sensor fusion component to perform a task known as *map matching*, wherein physical locations of detected objects are referenced to the map's coordinate system. The localization component typically uses a combination of GPS and inertial measurement sensors. Certain algorithms try to improve on the accuracy of localization by identifying visual landmarks via cameras. The base map layers have traditionally been stored onboard, but the trend is to move towards tiled maps, where individual tiles are dynamically streamed from a service provider based on vehicle location, but which may be locally cached.

The **semantic understanding** component is likely to be the most "fuzzy" for some readers. Words like 'semantic' are not exactly common in the vocabularies of practicing automotive architects. To some extent, this is the component in which the balance shifts from sensing to perception. More concretely, the semantic understanding component can include classifiers for detected objects, and it may annotate the objects with references to physical models that predict likely future behavior. Detection of ground planes, road geometries, representation of driveable areas may also happen in the semantic understanding component. In specific cases, the semantic understanding component may also use the ego vehicle data to continuously parameterize a model of the ego vehicle for purposes of motion control, error detection and potential degradation of functionality.

The **world model** component holds the state of the external environment as perceived by the ego vehicle. Conceptually, it is possible to think in terms of an extended or compound world model that includes the internal states of the ego vehicle, thus simultaneously representing the vehicle internal and external worlds. However, in practice, we have experienced that such compound world models rarely exist, because requirements and technologies at the technical architecture level usually lead to separated, optimized implementations. Also, the producers, consumers, and processing involved of data originating from the ego vehicle and its external environment, have qualitative differences. Not having a compounded world model does not eliminate a great deal of value, and having a compounded world model does not add a great deal of value. Therefore, in most practical implementations, the world model component as described here, only represents the external world of the ego vehicle. We like to characterize the world model component as either passive or active. A passive world model is more like a data store and may lack semantic understanding of the stored data. Therefore, it can not, by itself, perform physics related computations on the data it contains, to actively predict the state of the world given specific inputs. The active world model, on the other hand, may incorporate kinematic and dynamic models of the objects it contains, and be able to evolve beliefs of the world states when given a sequence of inputs. Other components (like decision and control) may then request a set of predictions of future world states, for a specific set of inputs, in order to determine the optimal inputs to be applied. The passive world model, as described above, is by far the most common in the autonomous driving projects we have encountered. In fact, there are efforts to create a (semi-) standardized representation [6] of the world in the form of so called 'Local dynamic maps' (LDM) [13]. An LDM is technically implemented as a database, but can be conceptually thought of as a layered map. The bottom-most layers represent the most static beliefs about the world, while the topmost layers represent the most dynamic, in the sense of time. For example, the lowermost layer may be populated with a static map of the immediate surroundings of the vehicle (roads, permanent features, etc.). The layer above it may be populated with more-or-less static road objects (traffic lights, lane markings, guard rails). The next layer may contain temporary objects like diversions due to construction work. The final layer would be populated by fast-moving objects detected by the rest of the perception system (other vehicles, pedestrians, etc.). The world model component typically provides an interface to query its contents, add and remove data, concurrency, access control, replication over distributed computational media etc. In specific cases, it also holds historical information about some or all of its contents.

2.2 Decision and control

The decision and control category refers to those functional components which are concerned by the vehicle characteristics and behavior in the context of the external environment it is operating in. Reference is made to the vehicle as a whole, and the way it moves in its environment, energy and fault management concerns that affect the vehicle's motion to its destination, as well as reactive control to unexpected events in the environment. The specific details of the vehicle platform that actually generate the desired external behavior and characteristics are not of prime interest.

The **trajectory generation** component repeatedly generates a set of obstacle free trajectories in the world coordinate system and pick an optimal trajectory from the set. The generation and/or selection of an optimal trajectory is constrained by factors like limitations of platform motion (e.g. non-holonomicity), energy availability, and the state of the platform with regards to faults and failures.

The emergence of dedicated, holistic **energy management** components is a relatively recent phenomenon, stimulated by the growth of hybrid and electric vehicles. This

component is usually split into closely-knit sub-components for battery management and regenerative braking. Since energy is a system-wide concern, it is not uncommon for the energy management component to have interfaces with other vehicular systems like HVAC, lights, chassis, and brakes. For autonomous driving, sensors and associated fusion and computation silicon may account for a significant energy consumption.

Diagnosis and fault management throughout the system components is an integral part of any well designed architecture. In the context of decision and control, this refers to identifying the state of the overall system with respect to available capabilities. The identified state would be used to influence behavior like redundancy management, systematic degradation of capabilities, triggering transitions to and from safe states, and potential driver handover.

Reactive control components are used for immediate (or "reflex") responses to unanticipated stimuli from the environment. Existing vehicle features like collision mitigation by braking may be considered as reactive control. These components execute in parallel with the nominal system, and if a threat is identified, their output overrides the nominal behavior requests. Their sense-plan-act loops are typically at least an order of magnitude faster than the nominal system loop. It is sometimes the case that what is considered reactive behavior in the presence of unexpected events, can be dealt with by very fast deliberative behavior. For example, consider the Autonomous Emergency Braking (AEB) feature in some passenger cars. This is considered a reactive function, that monitors a small subset of sensors (compared to full autonomous driving) and initiates braking action in case of imminent collision with a moving or stationary object. The function is constantly active (when enabled) and may generate a deceleration demand that overrides other demands on the propulsion subsystem. However, if the perception and trajectory generation components are sufficiently fast, they could detect the threat and generate appropriate trajectories (in this case, strong deceleration) as part of their normal operation, negating the need for a specialized AEB system. (Such a specialized system may still be implemented as a redundancy measure or for supervisory control, if a system safety analysis suggests provable improvement in safety or decrease of ASIL requirements on other parts of the system. However, it wouldn't be a functional necessity.) The call for reducing reactive functionality in favor of fast, deliberative functionality needs to be taken in consultation with domain experts and by considering the specific algorithms involved, and the characteristics of their technical implementation. In particular, worst case execution times and end-to-end timing analyses are important factors.

The **world model** part of Decision and Control refers to an active world model, as described previously in Section 2.1. If such a world model is present, it may be used by the decision and control components to generate a set of potential futures for a given set of input actions. The most desirable future will then be determined, and the corresponding input will be the one actually used by the system.

2.3 Vehicle platform manipulation

This category groups the components that are directly responsible for the motion of the vehicle. They abstract the principal actuation subsystems, and also provide a minimum level of stability to the platform while it is in motion. Although not directly related to propulsion, components related to passive vehicle safety and occupant protection may be included in this category, since they are closely related to scenarios arising from undesirable propulsion and may be triggered by the decision and control components.

The **platform stabilization** components are usually related to traction control, electronic stability programs, and anti-lock braking features. Their task is to keep the vehicle platform in a controllable state during operation. Unreasonable motion requests may be rejected or adapted to stay within the physical capabilities and safety envelope of the vehicle.

The **trajectory execution** components are responsible for actually executing the trajectory generated by Decision and Control. This is achieved by a combination of longitudinal acceleration (propulsion), lateral acceleration (steering) and deceleration (braking). Most recent vehicles already incorporate such components and they may be considered "traditional" from the perspective of autonomous driving development.

3. FUNCTIONALITY DISTRIBUTION

As mentioned in section 1, we consider an autonomous vehicle architecture as broadly comprising of a vehicle platform and a cognitive driving intelligence. These two parts may be considered as two distinct layers of the architecture. It is then necessary to consider at least the following two questions

1. What kind of information should flow between the cognitive driving intelligence and the vehicle platform layers?

2. What changes are necessary/desirable in a given vehicle platform, if it will be controlled by a computer, and not a human being?

Answers to both questions depend on the distribution of functionality between the vehicle platform and the cognitive driving intelligence. Currently, most autonomous driving experiments that build on existing, in-production vehicles follow a typical pattern: The vehicle contains a network of electronic control units (ECUs) controlling the basic vehicle propulsion (lateral and longitudinal acceleration, braking). The vehicle manufacturer usually builds a "gateway" that allows the experimenters to send a limited set of commands to the ECUs in the vehicle network. These commands are usually set-points for the various control loops that exist in the vehicle platform. For example, the cognitive driving intelligence may continuously regulate the set-point of the cruise-control function in the vehicle.

From a theoretical viewpoint, the distribution of functionality between the cognitive driving intelligence and the vehicle platform can lie between two extremes, as shown in Figure 2. In the Figure, the vertical placement of the components (above or below dotted line) denotes which layer they are allocated to. It is only this placement that is important; the horizontal placement and location relative to other components in the same layer are purely asthetic, and carry no meaning. On one extreme, Figure 2 (a), the cognitive driving component directly controls the torque outputs of the vehicle platform actuators, in a so-called "distributed I/O approach". There is no greater intelligence in the vehicle

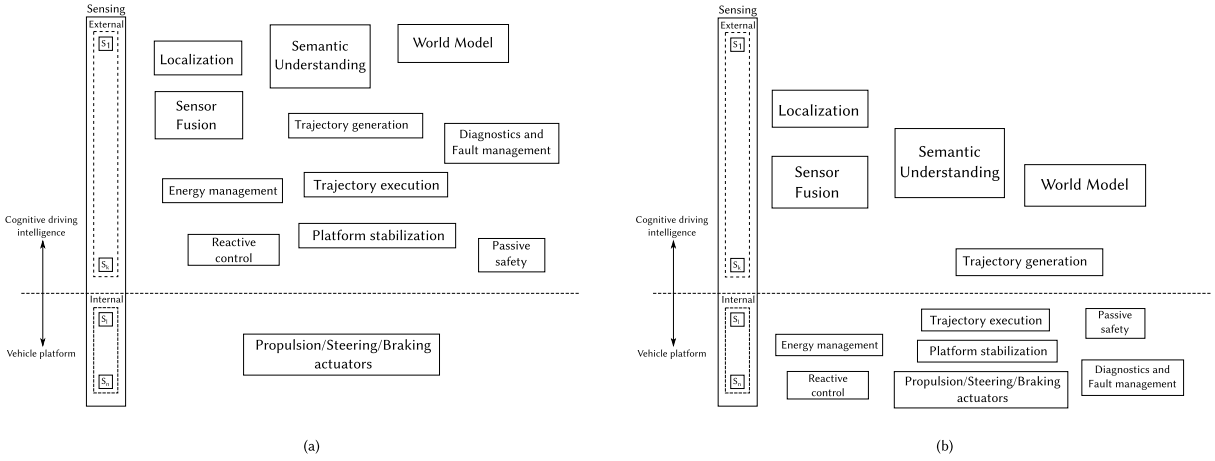

Figure 2: Distribution of functional components across the layers in an autonomous driving architecture

platform, which then represents just a set of distributed inputs/outputs. The cognitive driving component then needs intimate familiarity with the vehicle platform, and it would be difficult to de-couple and reuse the one without the other. The other extreme, Figure 2 (b), treats both the cognitive driving intelligence as well as the vehicle platform as two cooperating, relatively autonomous entities. Neither knows the intimate details about the other, and the driving intelligence makes motion demands of the vehicle platform in world coordinates, which the latter makes a best effort to fulfill. The task of the driving intelligence is to perceive the world and make motion requests in this world, while the task of the vehicle platform is to realize the desired motion requests while keeping its own features and limitations in mind. In such an ideal de-coupling, the same driving intelligence should be able to operate a variety of vehicle platforms, provided the acceleration interface remains the same.

The functionality distribution shown in Figure 2 (a) leads to simplicity of the vehicle platform, but it has significant drawbacks from the viewpoint of complexity, separation of concerns, and technical feasibility. In order to perform closed-loop propulsion control of the vehicle platform, the driving intelligence would need a fairly detailed model of the platform, including its dynamics and the constraints on the vehicle actuators and sensors. Performing fine-grained (low time horizon) control of the actuators by using motion feedback from the perception system places unreasonably high demands on the technical implementation and performance of the perception system. The functionality distribution shown in Figure 2 (b) on the other hand, is attractive because it enables a relatively clean separation of concerns. The driving intelligence need not be concerned with the finer details of how the motion it desires is achieved. The vehicle platform need not be concerned with how and why the motion commands are generated - only whether they are realizable and if so, how to best realize them given the current platform capabilities. Concepts related to stabilization of the platform, like traction control, anti-lock brakes etc. are transparently realized by the vehicle platform, without the driving intelligence having to be aware of them.

Our recommendation, based on three different self-driving platforms and projects, as well as the recommended architecting best practices of separating concerns and loose couplings, is to achieve as clean a split as possible, between the driving intelligence and vehicle platform [(Figure 2 (b)]. This lowers the cognitive complexity [10] (cognitive effort needed to understand a model) of the architecture, as well as reduces the potential for feature interaction and other undesirable emergent behavior. It also enables better reuse of the driving intelligence and vehicle platform in other projects. That said, any assumptions made regarding the behavior and performance of the vehicle platform need to be made explicit. This is especially true for end-to-end latencies on the fulfillment of acceleration requests, and interpretation of sensor data by the controllers in the vehicle platform. The approach (of Figure 2 (b)) places high demands on the functionality available in the vehicle platform, with regards to its abilities for keeping the platform stable, and retaining basic self-protection measures which may include reactive control. In practice, this is unlikely to be an issue because the high-end vehicles of most automotive OEMs today already incorporate such functionality and it is these high-end vehicles that are the most likely candidates for receiving upgrades to self-driving functionality. The principal modifications needed to these vehicles, as self-driving vehicle platforms, would be in the area of sub-system redundancy, to increase the platform reliability and safety.

4. FUNCTIONAL ARCHITECTURE

This section briefly presents a functional architecture for autonomous driving, that has emerged from our work. It brings together all the functional components described so far, and distributes them over the cognitive driving intelligence and vehicle platform components. Also, it follows our proposal of achieving a relatively clean split between the two. For technical and practical reasons, some components like energy management and diagnostics are allocated to both the vehicle platform as well as the driving intelligence. However, each allocation has slightly different responsibilities and scope of operation.

As shown in Figure 3, the sensing and world model components, although conceptually unified, are split into those dealing with the external environment of the vehicle, and those dealing with the Ego vehicle platform. The split helps to achieve separate technical implementations, if required,

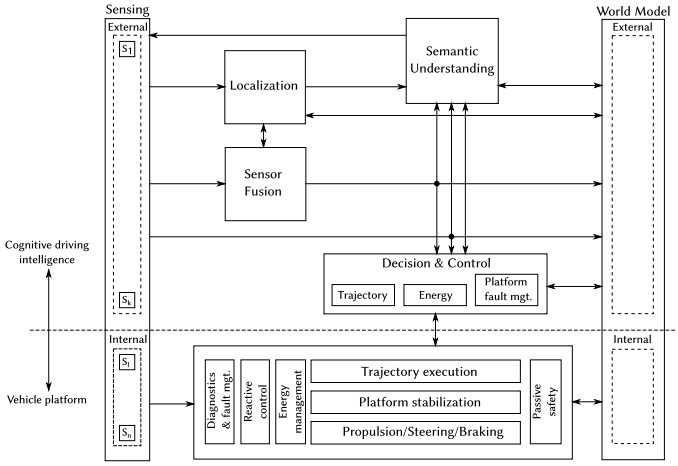

Figure 3: A functional architecture for autonomous driving architecture

when the functional architecture is eventually refined to a technical architecture, following the ISO26262 process. The inter-component arrows in Figure 3 represent data-flows in the direction of the arrow. As shown, the outputs of the sensing components go to the rest of the perception and decision and control components, either directly or indirectly, depending on the level of processing and fusion that is needed.

In our experience, it is useful to establish a data link between localization and sensor fusion. Certain sensors may exhibit repeatable tendencies at fixed locations along specific routes, like increase in false positives, dropouts etc. Changing the level of confidence in a sensor, based on geographical location is an interesting line of research and the architecture should not be a limiting factor.

Another interesting data link in Figure 3 is the connection from the semantic understanding component to the sensor components. This is useful in at least three scenarios. Firstly, some specialized autonomous driving situations benefit from so-called *focused attention* mechanisms. Focused attention means exploring a specific part of the environment more deeply. This may require physical motion of the sensors and/or configuration changes to the sensors (panning a camera to a different field of view, changing the 'zoom' of a lens, etc.). Today, most sensors of most autonomous vehicles are physically fixed to a constant pose with respect to the vehicle coordinate system. But in the domain of mobile and cognitive robotics, it is quite common to have, for example, a pan-tilt-zoom camera to aid the robot in a search task. Secondly, calibration changes to the sensors may be needed at runtime (e.g. changing exposures based on time of day, triggering re-calibration if changes in physical alignment are suspected). Thirdly, if communication transceivers are considered as a kind of sensor/actuator, the semantic understanding component can use it to respond to incoming communication requests, publish ego vehicle information and make asynchronous requests for information. Such communication requirements are often an integral part of scenarios like cooperative driving, where a vehicle maintains constant communication to the infrastructure and other vehicles in the vicinity.

The decision and control components include energy management from the perspectives of mission completion and overall vehicle energy demands (interior and exterior lights, HVAC). This is in contrast to the energy management component in the vehicle platform, that manages blending of hybrid propulsion systems, regenerative braking, and parts of the battery charge management and cell load balancing in electrical vehicles.

The reactive control in this particular architecture is allocated to the vehicle platform, since our particular technical implementations of the perception and decision and control components have not been fast enough to deal with unexpected events as part of the deliberative control. Also, having the reactive control in the vehicle makes it easier to assure a basic level of self-protection for the vehicle platform, in case the cognitive driving system is totally disabled. Since the passive safety components like airbags, pre-tensioners of seat belts etc. are very tightly coupled to the vehicle platform, and unlikely to be easily reused in other vehicle platforms, their control components are also a part of the vehicle platform.

We have not shown the interactions between the functional components in the vehicle platform, keeping space limitations of this paper in mind. Recent vehicles already incorporate components like platform stabilization, reactive control and abstraction of the motion control actuators. Thus, the novelty in the vehicle platform is lower, compared to that of the driving intelligence. Nevertheless, it is important to clarify that the physical actuation systems are abstracted by the 'Propulsion/Steering/Braking' component in Figure 3.

So far in this paper, we have completely neglected to describe the reverse or return flow of data from the vehicle platform to the driving intelligence. Partially this is because the contents of the dataflow depend not only on the distribution of functionality, but also on the particular algorithms within each functional component. Although, it is tempting to consider only one-way communication from the driving intelligence to the platform, and letting the perception system close the loop, in practice, the platform can provide a constant flow of states and possible asynchronous notifications that are useful for feedback and feedforward to the driving intelligence. This is particularly true in case of degraded platform functionality, where the driving intelligence must quickly make sure that the generated trajectories are still realizable.

4.1 Comparison with similar architectures

Given the proliferation of autonomous driving projects, it is useful to compare the proposed architecture with those from ongoing/previous projects. While a detailed comparison of architectures is beyond the immediate scope of this paper, we would nevertheless like to point out some relevant projects and highlight key similarities and differences.

Close comparisons can be made with the architecture of Bertha, the Mercedes Benz S-class vehicle, that recently(2014) completed a 103 mile autonomous drive from Mannheim to Pforzheim [16]. In the system overview presented in [16], components like perception, localization, motion planning, and trajectory control are clearly identified. This agrees well with the components we have described in this paper. In addition, our paper explicitly discusses components for platform stabilization, energy management and fault diagnosis. Further, our architecture makes a clear distinction

between the vehicle platform and cognitive driving intelligence layers, as a guideline for allocating components. Such a layered approach is also found in some other project architectures, like the European HAVE-IT project [4], which separates the architecture into three layers: 'Driver interface components', 'Command layer' and 'Execution layer'. The execution layer corresponds closely to our 'vehicle platform' in that it pertains to drivetrain control and *"..to perform the safe motion control vector."* [4]. The HAVE-IT architecture explicitly acknowledges the presence of the driver and driver interface components, including an HMI. We have not covered those in our architecture description, because we chose to focus exclusively on the motion control aspect of the architecture. The architecture of Stanford University's DARPA Urban Challenge entry, Junior [12], is also relevant. This is an early example of an autonomous driving architecture and the interface to the VW Passat vehicle seems to be via steering/throttle/brake controls, rather than direct longitudinal and lateral acceleration demands. This architecture also explicitly includes a component/layer for 'Global Services' dealing with functionality like data logging, file systems, and inter-process communication. We do not describe these service because they do not strictly fit into an architecture's functional view, as defined in section 1. Similarities to our architecture include explicit division of the architecture into a sensor interface, perception, navigation, and a vehicle interface. Another similar functional architecture was presented by the company TRW, at the Tech. AD conference [2] in Berlin in February 2016. We have not found an academic reference to this architecture, but note that it splits the architecture into 'Observation', 'Perception', 'Decision and Planning' and 'Realization'. The world modeling, sensor fusion, and world model components seem to be combined under 'Perception'.

The explicit recognition of the semantic understanding component and the world model, to the best of our knowledge, is unique to our architecture description.

5. DISCUSSION

One useful way to assess the architecture is to compare its components and their functionality with the capabilities of a human driver. We have chosen to make the comparison by discretizing the human driver's activities into a set of categories, and then reasoning on the vehicle architecture effects of each category. One of the models to discretize the human driver's activities is the Observe-Orient-Decide-Act (OODA) model [5]. The OODA loop, shown in Figure 4, was developed by US military strategist John Boyd, and has found wide acceptance in many fields of human endeavor related to rapid judgment and decision making in an uncertain environment.

Broadly speaking, the Observe, Orient and Decide parts of the loop, as applied to overall vehicle motion, would be performed by the cognitive driving intelligence components. The Observe part corresponds to the sensor components in the architecture, whereas the Orient and Decide parts correspond to the semantic understanding and Decision and Control components respectively. The Act part of the OODA loop corresponds to the vehicle platform. It is important to point out that this mapping is with respect to the overall vehicle motion. For the operation of individual parts, like the vehicle control, similar OODA loops or other patterns may be present at a more fine-grained level. Both the observe and orient parts can apply to internal contexts i.e. the Ego vehicle states, as well as to external contexts i.e. the environment within which the Ego vehicle operates.

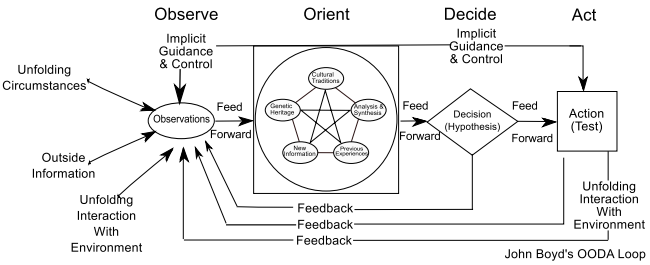

Figure 4: The OODA loop. [source: adapted from CC-BY 3.0 licenced image by Patrick Edwin Moran]

With regards to observation (sensing), the suite of sensors on an autonomous vehicle may collectively match (and even exceed) the sensory capabilities of ordinary human drivers. However, the Orientation part, corresponding to the perception and interpretation of sensory data remains far superior in humans. This is because humans can construct the external context far more rapidly than computers can, and they can reason more broadly and deeply about it than a computer can. Furthermore, humans can also apply rapid learning to their orientation capabilities. With computers, the extent of learning is still rather limited and restricted to specific contexts. Learning also presents a particularly difficult problem when it comes to product deployment. OEMs prefer to extensively validate the core orientation and decision making software prior to commercial deployment. Once the vehicles are in the field, any modifications to the software would usually require the validation not just of the individual modifications but of the entire modified vehicle. Revalidating the entire vehicle every time a relatively small level of learning occurs is not feasible with current methods of validation and system construction. Therefore, incorporating any learning into a deployed vehicle is not just a matter of regularly pushing software updates to the vehicle code. We foresee that addressing this challenge would require advances in two distinct areas: The virtual testing and validation of vehicles, and the development of 'correctness-by-construction' methods. The former would be needed in a scenario where field vehicles submit learned information to the OEM "cloud" where the learning can be extensively tested on virtual vehicles in virtual scenarios, in accelerated time. Only that learning which measurably improves the vehicle performance would be accepted and subsequently deployed. The latter advancement (in correctness-by-construction methods) is crucial in order to remove the need for verification of the entire vehicle, when changes are made to specific sub-systems. Such methods should assure, for example, a 'side-effect-free' integration of individual components. On the topic of learning, any improvements to the cognitive driving intelligence need to be categorized, for example, as rule based, knowledge based, and skill based [14]. Improvements in some specific categories are likely to be incorporated far more rapidly than in others. An example of this is where the vehicle relies on constantly updating maps supplied from off-board sources. The map may be considered a knowledge source, and any updates to the map that contain instances of more and detailed information, in

previously known categories, would help the same vehicle software to make potentially better decisions. In contrast, improvements to a pedestrian detecting algorithm would be considered a 'skill based' improvement and would need to be validated far more thoroughly than the map case.

The 'Act' part of the OODA loop is implemented via the vehicle platform. The main changes to the vehicle platform, when driven by a computer, relate to increased redundancy and self-protection. Redundancy needs to be provided not just to ensure safety, but also to increase the availability (non-disruption of provided service) in order to complete the mission. The redundancy refers not just to actuation mechanisms but also to the process of providing degraded service by potentially exploiting the so called inherent redundancy within the vehicle. An example of inherent redundancy would be 'steering-by-braking', a function that is not at all available to a human driver today. Given that the steering performance by braking is unlikely to match the performance of an actual steering system, there will be limits to the vehicle motion when steering-by-braking is utilized. This is essentially degradation of performance and a systematic exploration of various degradation modes is a necessary activity when considering the replacement of a human by a driving computer.

Finally, it is unlikely that in the near future, computers will demonstrate reasoning capabilities that match or exceed humans. Also, autonomous vehicles are envisioned as just one component of a future Intelligent Transport System(ITS). Both these factors point to the need for some sort of off-board intelligence (human or machine) that will assist the autonomous vehicle in a multitude of ways. Therefore, it is not unreasonable to expect that the two layers of the autonomous driving architecture mentioned so far, the vehicle platform and the driving intelligence, will be augmented by a third "cloud" based layer. The redistribution of functionality between on-board and off-board vehicle systems will thus be an interesting area of exploration.

6. CONCLUSIONS

The description of a functional architecture at such a high level of abstraction, as described in this paper, is just a preliminary step in the overall architecting process. Yet, it already yields enough material for an engaging discussion, with far reaching consequences for the overall system design. In architecting, there are no uniquely and definitively correct solutions and at this point, it is difficult to say anything beyond, "These ideas are derived from the state of practice, and have worked well for us.". We believe that patterns exist for autonomous driving architectures, and these patterns ought to be documented and debated. In this paper, we have contributed to the effort by describing the principal functional components needed for autonomous driving, together with some reasoning regarding their distribution across the architecture. A specific architecture incorporating the ideas has also been presented.

References

[1] European Commission, G. Technology readiness levels (TRL), HORIZON 2020 - WORK PROGRAMME 2014-2015 General Annexes, Extract from Part 19 - Commission Decision C(2014)4995. http://ec.europa.eu/research/participants/data/ref/h2020/wp/2014_2015/annexes/h2020-wp1415-annex-g-trl_en.pdf.

[2] Tech.AD - The Road towards Autonomous Driving, Berlin 2015. http://autonomous-driving-tech-berlin.we-conect.com/en/.

[3] The FUSE project. Functional safety and evolvable architectures for autonomy. http://www.fuse-project.se/.

[4] The HAVE-it EU project. Deliverable D12.1 Architecture document. http://haveit-eu.org/LH2Uploads/ItemsContent/24/HAVEit_212154_D12.1_Public.pdf.

[5] John Boyd. The essence of winning and losing. Unpublished lecture notes, 1996.

[6] ETSI TR 102 863 V1.1.1 Local Dynamic Map (LDM) - Rational for and guidance on standardization. http://www.etsi.org/deliver/etsi_tr/102800_102899/102863/01.01.01_60/tr_102863v010101p.pdf, 2011.

[7] ISO 26262:2011 Road vehicles - Functional safety, 2011.

[8] ISO 42010: Systems and software engineering - Recommended practice for architectural description of software-intensive systems, 2011.

[9] P. Derler, E. A. Lee, M. Torngren, and S. Tripakis. Cyber-physical system design contracts. In *ICCPS '13: ACM/IEEE 4th International Conference on Cyber-Physical Systems*, April 2013.

[10] H. Kopetz. The Complexity Challenge in Embedded System Design. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 3–12. IEEE, May 2008.

[11] J. Mårtensson, A. Alam, and S. Behere. The Development of a Cooperative Heavy-Duty Vehicle for the GCDC 2011: Team Scoop. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1033–1049, Sept. 2012.

[12] M. Montemerlo et al. Junior: The stanford entry in the urban challenge. *Journal of Field Robotics*, 2008.

[13] Z. Papp, C. Brown, and C. Bartels. World modeling for cooperative intelligent vehicles. In *Intelligent Vehicles Symposium, 2008 IEEE*, pages 1050–1055, June 2008.

[14] J. Rasmussen. Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-13(3):257–266, May 1983.

[15] O. Wallmark et al. Design and implementation of an experimental research and concept demonstration vehicle. In *Vehicle Power and Propulsion Conference (VPPC), 2014 IEEE*, pages 1–6, Oct 2014.

[16] J. Ziegler et al. Making bertha drive: An autonomous journey on a historic route. *Intelligent Transportation Systems Magazine, IEEE*, 6(2):8–20, Summer 2014.