Cognitive Systems in Intelligent Vehicles
A new Frontier for Autonomous Driving

Elvio Amparore[1], Marco Beccuti[1], Simona Collina[2],
Flavia De Simone[2], Susanna Donatelli[1], Fabio Tango[3]

[1]*Department of Computer Science, Università di Torino, C.so Svizzera 187, Torino, Italy*
[2]*Facoltà di Scienze della Formazione, Università degli Studi Suor Orsola Benincasa, Napoli, Italy*
[3]*Centro Ricerche FIAT (CRF), Via Principe Amedeo, 12, Torino, Italy*
{amparore, beccuti, susi}@di.unito.it, simona.collina@unisob.na.it, fladesimone@gmail.com, fabio.tango@crf.it

Keywords: Partial Autonomous Driver Assistance Systems (PADAS) and Advanced Driving Assistance systems (ADAS), Intelligent Vehicles, Autonomous Driving, Human-automation Interaction, Artificial Cognitive Systems, Markovian Decision Process (MDP), Petri Nets (PN).

Abstract: This position paper introduces the concept of "artificial co-pilot" (that is, a driver model), with a focus on driver's oriented cognitive cars, in order to illustrate a new approach for future intelligent vehicles, which overcomes the limitations of nowadays models. The core consists in adopting the human cognitive framework for vehicles, following an artificial intelligent approach to take decisions. This paper illustrates in details these concepts, as they are under development in the EU co-funded project HOLIDES.

1. INTRODUCTION

Nowadays, automation systems to support, or even to replace, human drivers have become a trend in the current Intelligent Transportation Systems research. They are called Advanced Driver Assistance Systems (ADAS) or Partially Autonomous Driving Assistance Systems (PADAS), depending on the level of automation considered; anyway, their goal is to strengthen driver's sensing ability, to warn / inform in case of errors and to reduce the control efforts of the vehicle itself. In fact, drivers are limited in recognizing, interpreting, understanding and operating in critical situations; moreover, they are prone to errors and to get tired (many accidents are due to human wrong behaviour, drowsiness, or inattention in general (Tango, 2013)). Therefore, these ADAS/PADAS can effectively avoid some accidents, by cooperating with the driver and assuring the mutual understanding between the human-agents and the machine-agents, in order to reduce or avoid conflicts. This principle of smart collaboration between humans and machines have been the focus of a number of theoretical studies, such as (Inagaki, 2008), (Flemisch, 2003), (Heide and Henning, 2006), (Li, 2012), in which full automation can be regarded as one extreme point of interaction spectrum.

In particular, for Li and colleagues, the concept of a "cognitive vehicle" was proposed and defined as cognitive driving assistance systems, which – utilizing the findings of multidisciplinary engineering and cognition science – is able to monitor and detect the errors of human drivers and correctly respond / intervene to avoid accidents. As mentioned by Da Lio and colleagues, depending on its application context, a system capable to determine how a human expert should drive, can be regarded as an artificial co-driver, which is considered a *symbiotic system*, that is, it is described using the rider-horse metaphor (or H-metaphor), in which an animal can "read" human intentions and, reciprocally, the rider can "read" the animal's intentions.

The goal of this position paper is to illustrate a new approach for the implementation of this virtual driver (hereafter, co-pilot), which adopts a human cognitive framework as basis and follows an artificial intelligence approach. This activity is carrying out inside the European co-funded project HOLIDES, whose main goal is to design adaptive cooperative systems, focussing on the optimization of the distribution of workloads between humans

and machines, to compensate losses of capacities for instance in stress situations (http://www.holides.eu/).

2. MODELLING THE DRIVER

Theories of cognition can be divided in two separate classes according to the role that context plays in cognitive processes. The implication for the artificial systems lays at the core of the debate among these different perspectives. The new embodied view, aiming at reducing the relationship between the individual and the environment in the cycle perception-action, suggests that information needed to act on the environment are given by the context, without any intervention of high cognitive processes (as in (Da Lio, 2013)). On the contrary, classic views of cognition divide between low and high cognitive processes being the high level of processing abstract and independent from the sensor modality through which information is acquired. Active Control of Thought – Reflexive (hereafter ACT-R) is a computational model aimed to simulate the behaviour of a driver (Salvucci, 2006) following the second perspective. ACT-R emphasizes the effort to integrate different sources as the task that a person is going to perform, the artefact necessary to perform the task and the cognition through which a person perceives, reasons and acts. ACT-R is an example of how cognitive processes are inserted into computational models to simulate driving behaviour. However, it reflects also the limits and the gaps between research on cognition and their implementations. The cognitive module is embodied in nature but inserted in a modular architecture and without a clear explanation about how the different processes interact each other.

Starting from ACT-R, but with the specific aim to improve safety control and to reduce the number and the impact of human errors in human-machine interaction, the Cognitive Architecture for Safety Critical Task Simulation (hereafter CASCaS) architecture has been developed. As described by Lüdtke, Weber, Osterloh and Wortelen (2009), the CASCaS model is a three layers architecture, which distinguishes the human behaviour on the base of different intentional demands:

1) *autonomous behaviour*: the level of "acting without thinking"; it is the level of the manual control which controls everyday low level actions;

2) *associative behaviour*: the level of actions based on plans elaborated in familiar contexts;

3) *cognitive behaviour*: the level of elaboration of new plans in new contexts.

In short, the functioning of the model is based on rules stored at the associative level in a memory component. Each rule has an "if…then" structure which relies an action on a goal, a series of subgoals and conditions imposed by the context.

Unlike the ACT-R architecture, CASCaS model allows parallelism between autonomous and associative layer simulating the human ability to manage two tasks simultaneously.

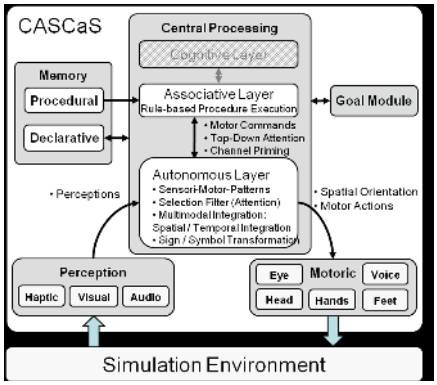


Figure 1: the cognitive architecture CASCaS.

For the implementation of the co-pilot in the HOLIDES vehicle demonstrator, we have adopted the CASCaS architecture as a basis, reproducing the autonomous behaviour and the associative behaviour into the co-pilot architecture (the cognitive layer can be foreseen as a further step of model development). The adopted probabilistic approach is described in the following sections.

3. IMPLEMENTING THE DRIVER MODEL

The new artificial driver solution we propose exploits probabilistic techniques, in particular Markov Decision Process (MDP) (Howard, 1960; Bellman, 1957) which we briefly recall in the following to pave the way to the explanation of how MDPs are been applied inside the CASCaS architecture, for the case under study.

3.1 Markov Decision Processes

MDP is a mathematical formalism introduced in the 1950s by Bellman and Howard (Howard, 1960; Bellman, 1957) in the context of operations research and dynamic programming. It has been used in a wide area of disciplines including economics, manu-

facturing, robotics, automated control and communication systems. More precisely, it is a stochastic control process, where, at each time step, the process is in some state $s \in S$, and a decision maker may choose any action $a \in A$ that is available in *s*. Then, the process responds by randomly moving into a new state *s'* according to a specified transition probability, and giving to the decision maker the corresponding reward (cost) $R_a(s,s')$ (depending by the chosen action and by the source and destination state).

A key notion for MDPs is the *strategy*, which defines the choice of action to be taken after any possible time step of the MDP. Analysis methods for MDPs are based on the identification of the strategies that maximize (or minimize) a target function based on the MDP's rewards (or costs).

It has been proved that the maximal (or minimal) reward and its associated optimal strategy for an MDP can be computed in polynomial time using linear programming techniques. However this is not practical for large MDPs, and alternative solution techniques based on iterative methods have been proposed, such as *value* iteration and *policy* iteration. Roughly speaking, value iteration (Bellman, 1957) consists in the successive approximation of the required values. At every iteration, a new value for a state is obtained by taking the maximum (or minimum) of the values associated with the state's outgoing actions. A value of an action is derived as a weighted sum over the values, computed during the previous iteration, of the possible next states, and where the weights are obtained from the probability distribution associated with the actions. Each iteration can be performed in time $O(|S|^2 \cdot |A|)$, where *S* is the state space set and *A* the set of all the possible actions. Instead the policy iteration algorithm (Howard, 1960) alternates between a *value determination phase*, in which the current policy is evaluated, and a *policy improvement phase*, in which an attempt is made to improve the currently computed policy. The policy improvement step can be performed in $O(|S|^2 \cdot |A|)$, while the value determination phase in $O(|S|^3)$ by solving a system of linear equations. In this regard, a critical issue for the application of MDPs to realistic complex problems is scalability with respect to the MDP size: for MDPs with very large or infinite state space, these algorithms may be inapplicable, and approximate solution techniques are the only viable approach.

In this paper we focus on sparse sampling techniques (Kearns et al., 1999), which do not need a complete description of the MDP, but that only require access to a *generative model* that can be queried to generate, from an initial state, a smaller MDP that is still sufficient to compute a near-optimal strategy. Hence, the complexity of these approaches does not have dependence on the global MDP size, but it is exponential in the solution horizon time (which depends on the desired degree of approximation of the optimal policy).

Obviously a crucial aspect of this technique is the definition of the generative model which, taking in input a state-action pair $\langle s, t \rangle$, must be able to randomly generate a next state *s'* according to a transition probability $P_{s,a}(\cdot)$.

In this paper, we propose to exploit the high level formalism, called *Markov Decision Petri Net* (MDPN) as starting point for the generative model. A MDPN models a system in terms of its events, while for an MDP the system evolution has to be expressed by explicitly describing all possible states, actions and probabilistic transitions. The high level description of the MDPN can ease the modeller task and can reduce the risk of introducing errors.

3.2 Markov Decision Petri Nets

The MDPN formalism provides a graphical description of the system, where a complex non-deterministic or probabilistic behaviour is described as a composition of simpler nondeterministic or

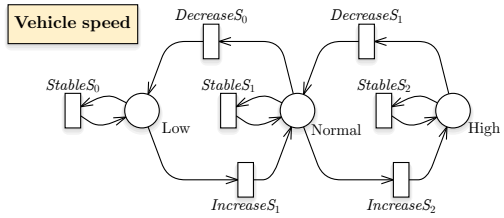
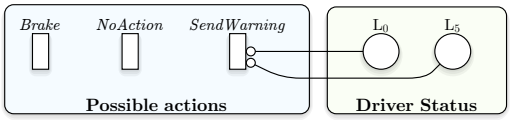


Figure 2: sub-net for the vehicle speed.

probabilistic steps in which the probabilistic behaviour is clearly distinct from the non-deterministic one. In details, a MDPN model is composed by two Petri nets: the probabilistic subnet N^{pr} (enriched with a transition weight function) and the non-deterministic subnet N^{nd}. These subnets represent the probabilistic and non-deterministic behaviours of the underlying MDP, respectively.

Figure 2 shows a simple probabilistic sub-net N^{pr} modelling the vehicle speed. According to PN notation the places, graphically represented as circles, correspond to the state variables of the system (i.e. *Low*, *Normal* and *High*), while the transitions (graphically represented as boxes) correspond to the events that can induce a state change (i.e. *DecreaseS*$_i$, *IncreaseS*$_i$, and *StableS*$_i$). The arcs connecting places to transitions and vice versa express the relation between states and event occurrences. Each



place can contain tokens, drawn as black dots. The number of tokens in each place defines the state, called "marking". The evolution of the system is

Figure 3: sub-net for the co-pilot's actions.

given by the firing of an enabled transition[1], which removes a fixed number of tokens from its input places and adds a fixed number of tokens into its output places (according to the cardinality of its input/output arcs).

Figure 3 shows a non-deterministic subnet N^{nd} in which the automatic driver can choose among three possible actions: *break*, *do no action*, or *send a warning*.

Observe that N^{pr} and N^{nd} can share places (as shown in Figure 3, where the places $L_0 \ldots L_5$ belong to a probabilistic sub-model describing the level of driver's attention), but they cannot share transitions.

Moreover, an MDPN model must have an associated reward function defined in terms of its places' markings and of transition firings; such reward function is used to compute the corresponding MDP reward to be optimized.

The generation of the MDP corresponding to a given MDPN has been described in details in (Beccuti et al., 2007): it consists of (1) a composition step, merging the two subnets in a single net, (2) the generation of the RG of the composed net, (3) two reduction steps transforming each PR and ND sequence in the RG into a single MDP transition.

[1] A transition is enabled if each input place contains a number of tokens greater or equal than a given threshold, and each inhibitor place contains a number of tokens strictly smaller than a given threshold.

3.3 The MDPN Model

The MDPN model that we use to derive the MDP of our co-driver requires defining first a multi-interval discretization of all the continuous-valued measures collected by the sensors (i.e. frontal Long Range Radar, Lane Recognition Camera and rear Short Range Radar for the blind-spot areas). Obviously a higher number of intervals increase the quality of the solution, but it makes the model more complex. Therefore, the most appropriate trade-off is an important part of our planned investigation during the HoliDes project.

The second step is a careful selection of which system's components have to be modelled. Our initial proposal is to consider the following system's components:

- *A vehicle* component describing the vehicle dynamic status (according to the information available on CAN bus);
- *A driver* component describing the driver status as reported in section 2;
- *An obstacle* component describing the obstacle status in terms of its relative speed and position (longitudinal and lateral) w.r.t. our vehicle;
- *An action* component describing the possible actions (e.g. to break, to do no action, to send a warning) that the artificial driver can execute.

It naturally follows that the first three components will be used to generate the corresponding N^{pr} net, while the last one the N^{nd} net.

Moreover, the reward function for this MDPN model can be defined by combining the following transition reward:

- if action *Break* is selected then it returns $Cost_{Break}$;
- else if action *SendWarning* is selected then it returns $Cost_{SendWarning}$ else it returns 0;

with the following marking reward:

- if place *Collision* is marked then it returns $Cost_{Collision}$ else it returns 0;

with $Cost_{Collision} \gg Cost_{Break} \geq Cost_{SendWarning}$.

This obtained reward function is hence able to assure that the system goal is to avoid collision minimizing the total number of actions *Break* and *SendWarning*. Obviously, more complex reward functions could be also investigated during the project.

3.4 Integration in the vehicle

Vehicle integration is shown in Figure 4, where the On-line Sparse Sampling Algorithm (OSSA) uses the MDP derived automatically by MDPN model as generative model. Practically, data collected by the vehicle's sensors are discretized to map them on a specific MDP state s.

Then, such MDP state is passed as input to the OSSA, which will return a small "sub-MDP" to be solved to derive a near-optimal strategy.

In details, starting from state s the OSSA will query the generative model N times on each possible pair $\langle s, a_i \rangle$. Then, this step is recursively applied on any generated states up to a selected time horizon H.

This essentially generates a sub-MDP with a tree

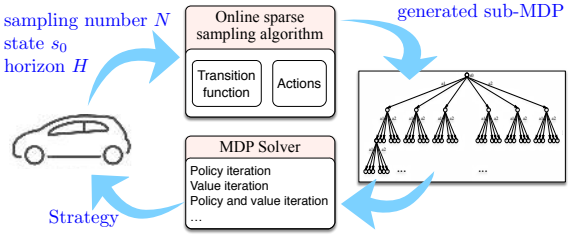


Figure 4: schema of vehicle integration.

structure (as shown in Figure 4) where the number of children for each node s is $N \cdot |A_s|$, assuming A_s to be the set of all the available actions in s. Moreover H gives the tree depth.

Finally this generated sub-MDP is solved (using policy or value iteration algorithms) to derive a near-optimal strategy, which is used to suggest the next action to the current driver.

4. DISCUSSION AND CONCLUSIONS

Researchers have widely investigated the possibility to reduce or eliminate the accidents due to driver's errors or inappropriate behaviors, by using specific ADAS/PADAS applications that warn the driver or even by using automated systems that can replace the human user, by taking control of the vehicle in a proper time. In this position paper, we have selected an appropriate cognitive model and related architecture (CASCaS) of the driver and implemented an artificial co-pilot starting from it and reproducing the autonomous and associative layers. To achieve that, we follow a probabilistic approach, described in terms of Markov Decision Petri Net formalism. In Figure 5, the architectural scheme of the co-pilot is shown. Under normal condition, the driver (the human-agent) perceives the environment, evaluating the possible risks (using the information from the co-pilot as a support). Based on these results, the driver formulates an intention and plans the next action (a trajectory in the future), which are implemented by acting on the pedals and on the steering. In the meanwhile, if a co-pilot is present, it analyzes the environment as well, and predicts the possibility to have a crash or a potentially critical situation. Thus, the co-pilot assesses risks creating its own driving plan, comparing this maneuver with the one that the driver is actually performing and taking into consideration the intention of the driver. This determines how dangerous a given situation can be, and thereby the level of automation which is necessary (e.g. by displaying a warning signal or some information to the human-agent, or whether an automatic intervention is needed).

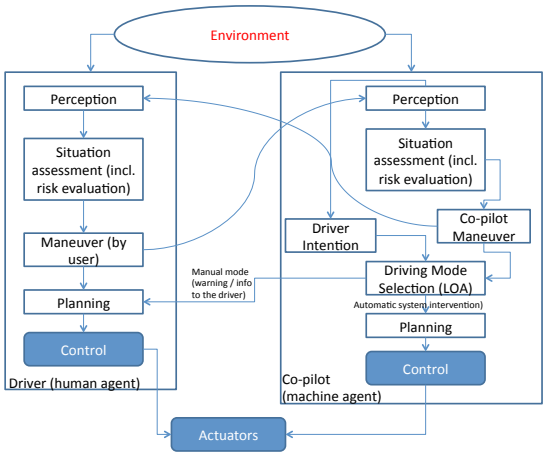


Figure 5: co-pilot scheme in HoliDes.

With respect to the current state of the art, we consider the works of Da Lio and colleagues, of Wu and colleagues and also of Li and colleagues. In (Da Lio, 2006) the perception-action framework is considered (embodied view); in this paper, we regard the "classical" view of cognition as the most appropriate, because we can reproduce the different levels of cognition in a hierarchical way which can be reproduced in a system architecture and implemented by a computational point of view. In this context, our choice of using CASCaS is motivated by its goal-oriented model, for which its predictions are easier to be generalized respects to a task-oriented model (e.g. it can be applied to automotive domain or to aeronautics domain, indifferently), by using a probabilistic approach, such as the one described.

In this sense, we follow the line indicated by Li and colleagues, with their concept of "cognitive car", where our co-pilot can be regarded as an instantiation. Another contribution of our work is about the understanding of which functions can be

automated next and to which extent. This could provide a kind of roadmap towards the realization of fully autonomous vehicles in a near or far future, where the co-pilot can substitute the human driver (although the introduction of more automated human-machine systems should occur gradually).

Finally, it is worth to mention the work of NaiQi Wu and colleagues, since they adopted Coloured Hybrid Petri Nets (CHPN). Their goal was to show a model based on CHPN to describe cooperation behaviour between a driver and a co-pilot in an ADAS application. They showed that their model is deadlock-free and conflict-free. In our case, Petri Nets are used instead as a high level formalism for the MDP, which derives the strategy of the co-pilot itself and the intelligence of the automated system.

To sum up, in this position paper we have attempted a new technological system for co-pilot implementations, using MDP for the computational implementation of the cognitive system. Since HOLIDES project have just started October 2013, the development of the proposed framework is in its initial phase. The next steps consist now in the preparation and execution of the experimental phase on the field with the demonstrator vehicle to collect real-time and on-line data for the tuning and the evaluation of the MDP co-pilot. This phase, together with the prototype set-up, is foreseen in this year and at the beginning of 2015; while the final implementation and assessment of the co-pilot will be the activity to carry out within the end of the project (August 2016).

One important future work will be to investigate the possibility of extending this approach using Partially Observable MDP (POMDP) (Leslie, 1995)). Indeed, a POMDP is a generalization of an MDP, in which an agent must base its decisions on incomplete information about the state of the environment. Hence, POMDP can be used more efficiently to model systems where the agent cannot directly observe the complete underlying state. However, POMDPs are often computationally intractable to solve a real system and its approximate solution techniques for POMDPs could not provide a sub-optimal solution that satisfies the time constraints imposed by our application.

In addition, another important achievement is represented by the full exploitation of the CASCaS framework, in particular for the cognitive behaviour. In this context, the integration of driver's state classifier inside the co-pilot (driver state becomes an input in this case) is a crucial point for deciding the level of automation.

ACKNOWLEDGEMENTS

This research has been performed with support from the EU ARTEMIS JU project HoliDes (http://www.holides.eu/) SP-8, GA No.: 332933. Any contents herein reflect only the authors' views. The ARTEMIS JU is not liable for any use that may be made of the information contained herein.

REFERENCES

Tango, F., and Botta, M. (2013). Real-Time Detection System of Driver Distraction Using Machine Learning. *IEEE Transactions on Intelligent Transportation Systems Journal*, vol. 14, no. 2, pp. 894-905, DOI 10.1109/TITS.013.2247760

Inagaki, T. (2008). Smart collaboration between humans and machines based on mutual understanding. *Annu. Rev. Control*, vol. 32, no. 2, pp. 253–261.

Flemisch, F. O., Adams, C. A., Conway, S. R., Goodrich, K. H., Palmer, M. T., and Schutte, P. C. (2003). The H-Metaphor as a Guideline for Vehicle Automation and Interaction. *Control*, no. December, pp. 1 – 30.

Heide, A. and Henning, K. (2006). The 'cognitive car': A roadmap for research issues in the automotive sector. *Annual Reviews in Control*, vol. 30, no. 2, pp. 197–203.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. Artificial Intelligence, 101:99-134, 1995.

Li, L., Wen, D., Zheng, N.-N., and Shen, L.-C. (2012). Cognitive Cars: A New Frontier for ADAS Research. *IEEE Transactions on Intelligent Transportation Systems Journal*, vol. 13, no. 1, pp. 395–407.

Salvucci, D. D. (2006). Modeling driver behavior in a cognitive architecture. *Human Factors*, *48*, pp. 362-380.

Lüdtke, A. Weber, L., Osterloh, J.-P., Wortelen, B. (2009). Modeling Pilot and Driver Behaviour for Human Error Simulation. *Conference Proceedings, HCI 2009, Digital Human Modeling,* San Diego, Springer: LNAI 2009.

Moore, R., and Lopes, J. (1999). Paper templates. In *TEMPLATE'06, 1st International Conference on Template Production*. SCITEPRESS.

Smith, J. (1998). *The book*, The publishing company. London, 2nd edition.

NaiQi Wu, Feng Chu, Mammar, S., MengChu Zhou (2011). Petri Net Modeling of the Cooperation Behavior of a Driver and a Copilot in an Advanced Driving Assistance System. *In Intelligent Transportation Systems, IEEE Transactions on (Volume:12 , Issue: 4)*. Date of Publication: Dec. 2011.

M. Beccuti, G. Franceschinis, and S. Haddad. Markov Decision Petri Net and Markov Decision Well-Formed Net Formalisms. *In 28th International Conference on application and theory of Petri nets and other models of concurrency (ICATPN'07)*, volume 4546 of Springer LNCS, pages 43-62. June 25-29, 2007, Siedlce, Poland.