Improvement of the driving safety using a virtual driver

Frédéric Holzmann*, Sascha Kolski†, Armin Sulzmann*, Gernot Spiegelberg*, Roland Siegwart† and Heiner Bubb‡

*DaimlerChrysler AG - Truck Product Creation
Stuttgart, Germany
Email: {frederic.holzmann, armin.sulzmann, gernot.spiegelberg}@daimlerchrysler.com

†Ecole Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, Switzerland
Email: {sascha.kolski,roland.siegwart}@epfl.ch

‡Technical University of Munich
Munich, Germany
Email: {bubb}@lfe.mw.tum.de

***Abstract*— This paper introduces a new concept for advanced driver assistance by means of a defined architecture including all system components redundant from the environment perception to the vehicle controllers. The first part describes the bottleneck of the current association driver - vehicle. After that a new solution to improve the safety on the road will be proposed. To improve the reliability of the intelligent part into the vehicle, a system will be integrated as a virtual driver. Its output will give some explanation about the quality of the possible motion vectors. The driver's command will be monitored to check its quality depending on the output of the virtual driver. The final part describes a concept of exchange of infromation to enhance the environment model will be described. In the end this paper will conclude towards future works and research issues.**

I. The SPARC concept

Road accidents are in 97% of the cases due to a human mistake [1]. About the half of these accidents would have been avoided if the driver has been warned that he evaluates wrong the surrounding environment.

DaimlerChrysler AG and its partners founded 2004 the SPARC consortium with the European Commission. The aim of this consortium is to pre-develop a new concept of active safety technology for the heavy goods vehicle and the development tools needed. The framework shown on the figure 2 will make the link from the environment sensing to the aggregates completely redundant.

The first part of the development is the transfer of the task for the vehicle coordination from the driver to the vehicle itself by using the Drive-by-Wire technology. This function will be done by a central redundant powertrain controller that will execute a given motion vector (acceleration, steering angle).

Later a virtual driver could to be integrated into the vehicle. On one hand its output could be used as feedback by the driver. On the other hand if the driver fails, it could brake the vehicle at the last moment before the point of no return. This monitoring function will be integrated into a decision control to check the compliance of the driver's command with the output of the virtual driver. Nevertheless the vehicle with this technology will never be allowed to drive autonomously with this function.

II. Copying the Nature: using the driver as model

The driver is the single link between the environment and the command of the vehicle. If the driver fails, the complete command level is lost and the vehicle becomes indubitably quickly dangerous for the driver and the peoples around. Therefore each relevant part of his work has to be integrated into a virtual driver that will work parallel to the driver.

The driver's main task is to reach safely a predefined goal with his vehicle. His main task will be split between three concurrent functions like in [2] : (1) *navigation* - finding the right road segments one after the other, (2) *planning* - managing the local environment around the vehicle by understanding the scenario, and (3) *coordination* - control activities in order to realise safely the scenario.

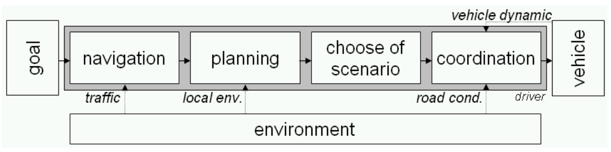

Fig. 1. Model of a driver

III. Technology to integrate into the vehicle

The navigation task could be helped nowadays with any infotainment system like the TLA from SiemensVDO. It bases on a stored map and a DGPS. It delivers time after time information to help the driver to choose the best road to drive. This part of the driving will not be supported by the SPARC project because its focuses only on the planning and the control parts as safety relevant technologies.

The planning function has to realise the desire of the navigation function based on some beliefs, like an agent [3]. The information coming from the infotainment could be taken into account but will always be checked with some other intern sensors. Like the humans, the planning function senses the environment with different sensors. The interpretation of the information leads to an environment model. It is used to enable the different possible scenarios.

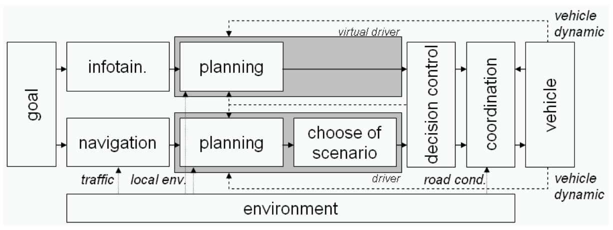

Fig. 2. Redundant data flow into the vehicle

IV. Virtual driver

Let define (E) the environment around the vehicle as a finite union of discrete elements. The type of elements could be limited to the most likely objects (personal car, truck, pedestrian, shield etc.). The others objects present in the environment will be defined as unknown.

i different sensors (S) are used to make a representation of the environment. This representation is a list of elements sensed (E) and the probabilities (P) of the sensing.

$$S : \{\forall i, E \longrightarrow \{E_i, P\}\}_i \tag{1}$$

The representation of the environment integrates some noise coming from the sensors themselves. Therefore a data fusion (DF) is used to improve the results if the reliability of the sensors is proven.

$$DF : \cap\{\{E_j, P_j \mid S(j)\ coherent\} \longrightarrow \{E^\star, P^\star\}\}_{i\leq j} \tag{2}$$

This model of environment is defined in the virtual driver (DR) as a blackboard. In order to manage several scenarios, different knowledge sources (KS) will be integrated. Each knowledge source will integrate a scenario with the validity range and a local model of the environment:

- A scenario definition (S).
- A validity range: ($E^{\star\star}$), sub-part of the model of the environment ($E^\star$), ($E^{\star\star} \subset E^\star$). This range is the union of obligatory elements and the exclusion of additional parts. If the knowledge source integrates all the information about the environment, ($E^{\star\star}$) will be the same as ($E^\star$). In that case, the model used by this source will be general enough to be used in each case.
- A longitudinal and a lateral controllers: $M_{long.}$, $M_{lat.}$. Their outputs are a range for the acceleration (γ) and for the steering (θ). An optimum of the scenario will be set with its quality or dangerousness (Q). The quality is a number between 0 and 255 [1]. If the number is bigger than 100, the scenario will not provoke any kind of accident. Normally this optimum would be the same as the driver's command if the driver were an optimal controller.

$$M : \begin{cases} M_{long.} : E^{\star\star} \longrightarrow \begin{cases} \gamma_{min}, \gamma_{max} \in \mathbb{R} \\ (\gamma_{opt}, Q) \in [\mathbb{R} \times \mathbb{N}] \end{cases} \\ M_{lat.} : E^{\star\star} \longrightarrow \begin{cases} \theta_{min}, \theta_{max} \in \mathbb{R} \\ (\theta_{opt}, Q) \in [\mathbb{R} \times \mathbb{N}] \end{cases} \end{cases} \tag{3}$$

- The probability ($P^\star$) of the elements will be used with the matching rate (D) of the validity range ($E^{\star\star}$) on the environment model ($E^\star$) to define the scenario's adaptation A as product of the both.

$$KS : E^{\star\star} \longrightarrow \{M, A\} \mid A = P^\star \cdot D \tag{4}$$

If the sensors gives a perfect model of the environment ($E^\star = E$), and the model used by the knowledge source matches perfectly all the environment, A will be set at 100%. If A becomes to sink to zero, the virtual driver will have problem to sense its environment and to understand it. Therefore the quality of its output has to be put in perspective. On the road, more than one scenario could be enabled at the same time. The final proposition (P) is also the union of the k different single scenarios.

$$P = \cap\{M, A\}_k \tag{5}$$

To describe practically the final proposition (P), the quality of each possible motion vector will be stored inot a motion vectors map (MVM) in the theoretic dynamic range of the vehicle:$[-10m/s^2, 10m/s^2] \times [-55^o, 55^o]$. The steps to discretise the map are lower than the smallest feasible step of the engine, the brake units and the steering unit: $0.1\,m/s^2$, $0.1\,rad$.

For each scenario, the quality of each motion vector will be extrapolated with two Gaussian curves ($\mathcal{G}$), one for the acceleration and one for the steering angle.

[1]This value has been chosen arbitrarily in order to use the type *char* (8 bits).

$$\mathcal{G}_{long.}:\begin{cases}\forall\gamma\in[\gamma_{min},\gamma_{opt.}]\\ \gamma_{min}\in[-10\,m/s^2,\gamma_{opt.}],\\ Q(\gamma)=Q\cdot e^{-\frac{(\gamma-\gamma_{opt.})^2}{2\cdot k^2}}\\ with\,k\in\mathbb{R},k=||\frac{\gamma_{opt.}-\gamma_{min}}{\sqrt{3}}||_2\\ otherw.\,Q(\gamma)=Q\end{cases}\quad(6)$$

$$\mathcal{G}_{lat.}:\begin{cases}\forall\gamma\in[\gamma_{opt.},\gamma_{max}]\\ \gamma_{max}\in(\gamma_{opt.},10m/s^2),\\ Q(\gamma)=Q\cdot e^{-\frac{(\gamma_{opt.}-\gamma)^2}{2\cdot k^2}}\\ with\,k\in\mathbb{R},k=||\frac{\gamma_{max}-\gamma_{opt.}}{\sqrt{3}}||_2\\ otherw.\,Q(\gamma)=Q\end{cases}\quad(7)$$

These curves are defined only in the ranges of the final proposition (P) and maximal at the optimum. If a minimum or a maximum does not exist, the quality of the motion vectors between the optimum and this extreme will be set at the quality (Q). An extreme may not be defined for example if there is no vehicle ahead or behind the SPARC vehicle.

$$\forall(\gamma,\theta),\begin{cases}\exists l\mid(\gamma,\theta)\in P_l,\\ MVM(\gamma,\theta)=\sum_l(\sqrt{\mathcal{G}_{long.}(\gamma)\cdot\mathcal{G}_{lat.}(\theta)})\\ otherw.\,MVM(\gamma,\theta)=0\end{cases}\quad(8)$$

An example of the such the motion vectors map is given in the figure 5. This map is generated if the vehicle is integrated into a platoon. In that case, there is a maximal and a minimal acceleration allowed. The steering angle range is computed to stay on the current straight lane without oscillations larger than $0.5\,m$.

A. Extereoceptive sensors

During the SPARC project three different sensors will be integrated. Their functionalities overlap themselves in order to improve their results and check their validity. The camera module from the EPFL with the image processing is the most intelligent sensor used here. It could deliver information about the road ahead the vehicle and about the objects around it. It will also be possible to trigger the camera to check the truth of the outputs of the other sensors.

The two other sensors are specified because they are dedicated either to the road detection or to the objects detection. The GPS with the map database will be delivered by Siemens VDO. It could describe the road ahead the vehicle as long as its database is up-to-date. The model based on the curvature will be used to decrease the speed in advance like a curve warner. The radars integrated by DaimlerChrysler extract the position and the relative speed of the objects.

The integration additional cameras for the dead angles monitoring permits the virtual driver to have theoretically an improved vision of the environment as the driver.

B. Knowledge sources

Some algorithms have already been integrated into the multiagent system:

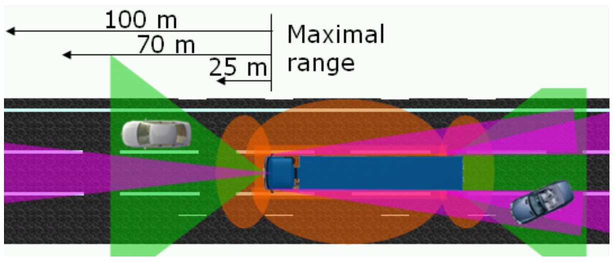

Fig. 3. Overlapping of the sensors

- Speed control : used if there is no vehicle ahead
- Distance control : to manage safely the distance between the vehicle and the other one ahead
- Overtaking control and lane changing
- Crossing assistant

These sources need to split the environment into parallel lanes and crossing roads. As long as there is no crossing road defined, the three first sources may be enabled. In that case the pattern on figure 4 is used to check the assumptions like enough place to change the lane, distance ahead enough etc. For each of the agents, a logic based algorithm check all the rules defined by the rules on the roads.

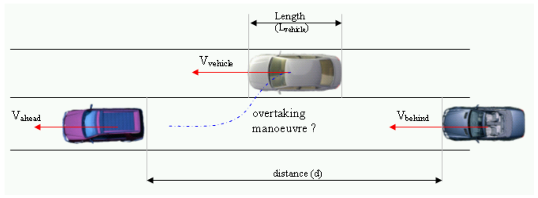

Fig. 4. Pattern used for a straight road with 3 lanes

On the figure 5, the vehicle is driven between two personal cars on the same lane. It is also possible for the driver to overtake or to follow the vehicle ahead. On the motion vectors map, the two scenarios are enabled. The gaussian curve in the middle of the map represents the vehicle following and the other one is more on the left (lane changing) and upper (acceleration) than the first scenario, it represents the overtaking.

C. Motion vectors map

As shown in the figure 2, the driver tests all the different possible scenarios in parallel and choose the one at the last moment with his personal criterion. A naïve copy of the driver model would integrate the chose of the command into the virtual driver. This integration is not acceptable because a single motion vector from the virtual driver is not enough to test the quality of the driver's command. Actually having the same outputs never occurs in practice. The driver could not been modelled with any pattern that matches his actions exactly but with some global transfer functions [4]. Only

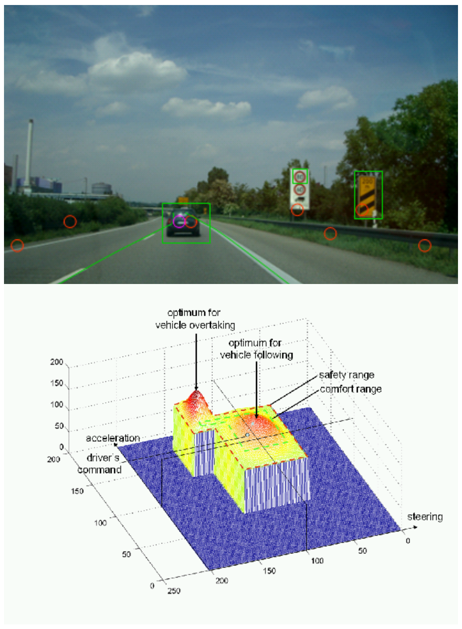

Fig. 5. Motion vectors map

with the two motion vectors, it will not be possible to check if they are coherent or not.

V. Decision Control

A. First generation - local computing

The decision control checks the driver's command by using the motion vectors map. A basic safety concept has been quickly developed. Its goal is to always have the motion vector to be realised by the vehicle into the safe range defined by the union of the different enabled scenarios (P). In that case if the driver's command is dangerous, the closest safe point from the driver's command will be send to the powertrain controller instead of the driver's command. Parallel to this function a local feedback could computed the local gradient of the motion vectors map.

The example presented on figure 6 is the output of a dynamic window computed when a pedestrian comes abruptly ahead the vehicle on a straight line. On the Boolean map (figure on the top right), the driver's command (green point) is out of the safe range. Therefore the decision control sends the closest safe point (red point) to the powertrain controller.

For the safe range, the map will be improved by taking the dangerousness into account. The dangerousness is a theoretical concept that could be model with the minimal distance to the road or to the object.

If there is no safe motion vector, an enhancement of the dynamic window needs to be used to manage the collision. A collision for a vehicle depends on different factors. The three more important factors are:

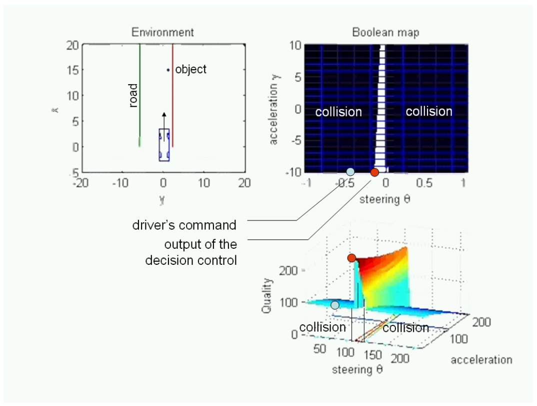

Fig. 6. Motion vectors map realised with a dynamic window

- Type of contact: personal car - personal car, truck - pedestrian, personal car - road etc.
- The speeds (v)
- The relative angle (ζ): a lateral collision for a vehicle is more dangerous than at the front.

An over-simple pattern was extracted from these remarks in order to conciliate each factor and always be lower than the given limit of 100. The quality is supposed to be the inverse of the energy $\mathcal{E}$.

$$\mathcal{E} = (M_{vehicle} \cdot v^2 + M_{other} \cdot v_{other}^2)\frac{cos(\zeta)}{2} \quad (9)$$

The two weights are statically defined for a personal car, a truck or a pedestrian. For the road, the weight is supposed to be null.

B. Second generation - Integration of comfort parameters

This first method leads the system to stay in a safe physical sate. In that case the vehicle will avoid any accident but the comfort will not be taken into account. Therefore two different ranges will be extracted from the motion vectors map. The first range is already used for the safety function as limit to avoid any accident: (SR) on the figure 5. The second one is the comfort range (CR) included into the safe range.

The comfort range is used as long as it exists. It is practically a subpart of the safety range with a maximal acceleration of $0.7\,m/s^2$, maximal deceleration of $-0.3\,m/s^2$ and a steering range depending of the speed (3^o in practice for a speed of $120\,km/h$).

The bottleneck of the first method comes from the research of the closest safe point. If the system has to find a safe point, it could choose the best one. For our system, the best point is one of the optimums defined by (P).

The decision control has also first to extract the optimums from the map. The optimums will be sorted depending on their likelihood with the driver's command. This leads to the chose of the driving optimum. The feedback could also been the vector in the direction of the optimum. Its length will be distance between the two motion vectors on the map.

C. *Finding the optimums*

The algorithm that finds the different optimums bases on a quick sort. As a lot of motion vectors get a null quality, a first copy of the motion vectors with a not null quality is made in order to erase the irrelevant vectors. An histogram of the different qualities results of this first step.

Than the histogram will be read from the best quality down to zero. As soon as a motion vector is find on the histogram, its position will be stored in the list of optimums. After that it will be erased from the histogram. Recursively the neighbours will be erased too as long as they get a lower quality. After that, the Gaussian curve corresponding to the found optimum will be completely erased from the histogram. An iterative loop will after that searches the next best quality in the histogram as long as it is not voided. The last vectors alone on an area are automatically local minimum.

D. *Choosing the driving optimum*

The movement of the optimums are extrapolated with a Lagrange polynomial function. A square is matched on the map, centred on the extrapolation points with a size of 5^o and $1\,m/s^2$. These squares as area of possible next positions of the optimums are compared to the driver's command position.

To extrapolate the position of the driver's command on the map for the next moment, the graph of its positions is filtered to extract a mean square function. The tangent of this function on the current point will give the direction of the next command. The intersection of the path of the driver's command with any square for an optimum will permit to choose the right optimum that the driver wants to realise.

This assumption will be verified later by giving a feedback to reach this optimum. If the driver reacts negatively to this feedback, the chose of the optimum will be supposed wrong. In that case an other optimum will be selected until the system finds the right one.

The figure 7 shows the extrapolation that permits to find

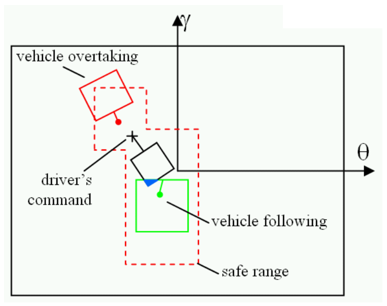

Fig. 7. Use of the filtering of the driver's command to choose the right driving optimum

the intersection of the driver's command with a scenario, in that case *vehicle following*. This method shows better results than using only the distance, as soon as the driver changes his strategy. In that case, the system could not observe that the driver's command tends to go away from the optimum.

E. *Transition path*

It is not always possible to go directly from the driver's command to the optimum. If there is any local minimum between the two points, having the transition near this minimum could lead to an accident if the quality is too low on this point. Therefore as soon as a local minimum is detected, the transition path has to be compute step by step. In practice, a local minimum could be found if there is for example three overlapped scenarios.

The transition path to follow from the driver's command to the optimum is based on the meshing of the map with a Delaunay's algorithm. A graph of the triangles with the bond between them is computed. The bond between two triangles is the apex if it corresponds to an optimum. The edges connected to a minimum could also been a bond because there is always a safe point between a minimum and an optimum.

The triangle including the driver's command is used as starting triangle for the exhaustive research of the best path.

Finding the best path needs a criterion to satisfy. The possible paths are qualified by the distance to the straight lane between the driver's command and the optimum.

The figure 8 shows a Delaunay's meshes for the points defined on the map. The optimum to reach is in blue and the others are in green. In red are the local minimum to avoid. The cross corresponds to the driver's command.

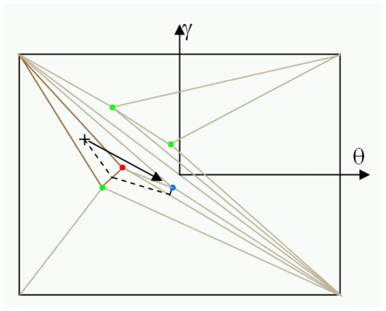

Fig. 8. Use of a Delaunay's algorithm to find a path

VI. Tests of the system - Limitation of the technology

Some tests show clearly the limit of the sensors like with the temporary lanes shown on figure 9. In this case, the virtual driver wanted to steer in order to change to another lane. The drivers tried to keep the command of the vehicle in order to stay on the right lane in spite of the feedback, which was supposed to improve the driving. But the system had without doubt overwritten the driver's command.

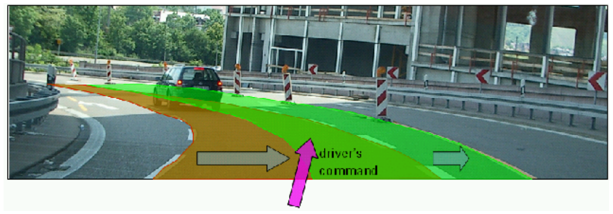

Fig. 9. Mistake from the image processing

A temporal monitoring of the driver's command gives an information to the reachable limits. If the limit of the driver's command tends to a place near an optimum, the environment model may have an offset or the planning function not been optimal. But if the limit of the driver's command does not correspond to any optimum, the redundancy of the drivers gets an incompatibility of its functions. The decision control has to analyse the both *pilot* functions to find where the failure comes from:

$$\underbrace{E \xrightarrow{\text{S}} E_i \xrightarrow{\text{DF}} E^\star}_{\text{sensing}} \quad \underbrace{\xrightarrow{\text{KS}} M \xrightarrow{\text{MVM}} P}_{\text{planning}}$$

The incompatibility is either due to different inputs or different planning functions. As the environment model is supposed to be quite the same in term of number and rough position of the elements, the planning functions may have an error. But it is easier to give first a feedback to both pilots in order to check if they get all the relevant information. Therefore an exchange of the environment model will be done and a modification of the comportment of at least one pilot may come.

The pilot receives information coming from the environment model of the virtual driver. The feedback will be done by using an haptic, visual or auditive feedback. Unfortunately the driver cannot get the all information right at the same moment. The capacity to receive additional information abruptly and to understand them right and quickly is depending on the driver's capacity. Therefore the driver will only get a part of the environment model ($\tilde{\hat{\text{E}}}$). If the driver were really good trained, $\tilde{\hat{\text{E}}}$ would be $E^\star$ the same.

$$\begin{cases} E_{pilot,new} = E_{pilot,former} \cap \tilde{\hat{\text{E}}}_{virtual,former} \\ E_{virtual,new} = E_{virtual,former} \cap \tilde{\hat{\text{E}}}_{pilot,former} \end{cases} \tag{10}$$

It is realtive easy to get the information coming from the virtual driver. But the information coming from the pilot for the virtual driver is more difficult to extract.To have an idea of the information the driver has, the decision control has theoretically to inverse the *pilot* function as it get only the output.

$$P \xrightarrow{\text{MVM}^{-1}} M \xrightarrow{\text{KS}^{-1}} \tilde{\hat{\text{E}}}^\star \tag{11}$$

The inversion of the function MVM could be made without difficulties. The problem of the inversion is for the knowledge in KS. This knowledge is unknown and depending on the driver. Therefore the research of the possible input is made with different model stored in the decision control.

Up to now, only the lateral control of the vehicle made by the driver could be right analysed. Actually Simmala [5] found that the drivers respond to an emergency with a ballistic jerk of the steering wheel. This jerk could be found by analysing the driver's command extrapolation explained before. In that case a dead time of 1.20 sec are taken into account as reaction time in order to compute with the vehicle speed, where the problem comes from. After that, the sensors could be triggered in order to check the validity of this reflex.

If there is no modification of the outputs (long incompatibility), the decision control will automatically switch the virtual driver off. In that case it is up to now not possible to know which pilot makes the mistake.

VII. Conclusion and Future Works

This paper described a new concept for advanced driver assistance based on the complete redundancy of the link from the environment sensing to the vehicle control. This radical modification of the architecture induces the development of a virtual driver.

This virtual driver generates a planning based on a model of the environment. The current model of the virtual driver uses a blackboard with different knowledge sources. In the future, this multiagent system will be improved by integrating a reflex function. This modification will need to change the architecture of the multiagent system and using a ContractNet approach. The bid will base on the computation time needed by the different agents. The maximal time will depend on the time to collision (TTC) with the objects.

The decision control integrated into the vehicle for the drivers' monitoring is up to now not capable to manage a long incompatibility. First the reverse computing for the longitudinal control must be realised. Secondly, the use of the fitness of the driver and the adaptation (A) have to been taken into account. With these both additional inputs, it will possible to balance the two commands gradually.

References

[1] J. Grendel, "Sicherheitsanalyse im strassenverkehr," tech. rep., Bundesanstalt fr Strassenwesen, Mensch und Sicherheit, 1993.
[2] H. Lunenfeld and G. J. Alexander, "A user's guide to positive guidance," 1990.
[3] R. Deters, "Developing and deploying a multi-agent system," 2000.
[4] L. kuang Chen and A. G. Ulsoy, "Driver mode uncertainty," June 1999.
[5] S. H., "Driver/vehicle steering response latencies," pp. 683–692, 1981.
[6] *International Conference on Intelligent Robots and Systems*, (Grenoble (France)), IEEE, Sept. 1997.
[7] A. G. S. Goerzig and P. Levi, "Realzeitfaehige multiagentenarchitektur fuer autonome fahrzeuge," pp. 44–55, Nov. 1999.
[8] T. M. I. S. R. Association, *MISRA-C:2004 - Guidelines for the use of the C language in critical systems*. oct 2004.
[9] A. K. R. Nathan Gartner, Carrol J. Messer and R. J. Koppa, *Traffic Flow Theory*. 1975.