




Toward the Design of Personalised Adaptive Driver Assistance for Truck Docking

Marc-André Buechner¹ ^a, Pedro Ribeiro¹ ^b, André Frank Krause¹ ^c, Jan Benders², Christian Ressel¹, and Kai Essig¹

¹*Rhine-Waal University of Applied Sciences, 47475 Kamp-Lintfort, Germany*

²*HAN University of Applied Sciences*

{*marc-andre.buechner2, pedro.ribeiro, andrefrank.krause, christian.ressel,*

Keywords: Advanced Driver Assistance, Personalization, Adaptive User Interface, Expertise Estimation, Machine Learning, Support Vector Machine, Truck Docking, Virtual Reality Simulator

Abstract: Backwards manoeuvring of articulated vehicles towards the loading bay at a distribution centre has been acknowledged as a significant source of accidents. Individualized driver assistance aimed to support professional truck drivers in this challenging scenario is still very much in its early days. This paper presents the ongoing design and development of a user interface and expertise estimation for adaptive and personalised driver assistance to support truck drivers through the scenario of parking a semi-trailer truck in a loading bay. The proposed system aims to adapt in real-time based on the current driving performance and adaptation rules.


1 INTRODUCTION


Performing the manoeuvre of docking a truck combination towards the loading bay of a distribution centre can be a very challenging task. This complicated manoeuvre can potentially result in accidents and costs due to damage caused, for example, by driving errors or information overload (Kural et al., 2016; Kusumakar et al., 2017). Generally, the main causes of truck accidents are related to human factors, technical failures or environmental conditions (Union, 2007) - with the first one being the most impactful cause. There is a plethora of aspects that influence the complexity of this specific task of docking a truck combination, some are driver related such as the experience, physical, emotional or perceptual factors, and others are associated with the environment and vehicle conditions. In that context, the platform VISTA-Sim (Ribeiro et al., 2021) was being developed in the project VISTA (VIsion Supported Truck docking Assistant).


VISTA-Sim uses a virtual reality (VR) environment to simulate and investigate driver performance, train drivers and develop and evaluate different forms

of context-sensitive and personalised driver assistance provided by the In-Vehicle Information System (IVIS) that is running on a virtual tablet installed in the truck cabin (see Figure 1). One of the pivotal advantages of the VR simulator is, that it allows the tracking of driver and vehicle behaviour data, which can be interpreted to discover driver problems and to adapt personalised assistance.

This paper aims to further develop the VISTA-Sim platform with a particular focus on a mechanism that will allow the personalised adaptation of driver assistance. For that, this mechanism shall be capable of learning from the driver's behaviours, estimating driver performance and deciding upon the most appropriate driver assistance. The development of such a mechanism is a challenge: Humans, particularly drivers, might be hesitant to follow AI-generated instructions that are not easily justified, interpretable and ultimately trustworthy (Arrieta et al., 2020). Therefore, this paper aims to apply the process of personalisation in which the driver assistance is tailored to match the driver's needs and preferences (Hasenjäger et al., 2019). Accordingly, it is intended to explore a hybrid approach that combines the potential of machine learning (ML) with background knowledge in the form of logical statements. This paper presents the current progress of a personalised adaptive user interface for an assistive applica-

^a  <https://orcid.org/0000-0003-0530-2975>

^b  <https://orcid.org/0000-0002-9829-8268>

^c  <https://orcid.org/0000-0002-9685-278X>

tion that supports drivers performing the backwards manoeuvring of articulated vehicles. The proposed mechanism aims to automatically adjust the driver assistance based on a performance estimation module that uses the VISTA-Sim capacities to track the driving behaviour. For example, the steering wheel angle, trailer's position and rotation, driver's head position and orientation and eye-tracking data such as the direction of the gaze ray and fixated objects can be fed into the performance estimation module. The performance-dependent assistance is adjusted by applying ML-based expertise estimation in combination with a rule-based adaptation mechanism.

2 RELATED WORK

Fan and Poole (Fan and Poole, 2006) defined the personalisation of digital technologies as “a process that changes the functionality, interface, information access and content, or distinctiveness of a system to increase its personal relevance to an individual or a category of individuals”. They also proposed a high-level framework to classify approaches to personalisation along three dimensions: (1) what to personalise (e.g. content and functionality of the driver assistance provided by the IVIS); (2) to whom to personalise (e.g. drivers); and (3) the explicitness of the personalisation or how the personalisation is achieved (Fan and Poole, 2006). Regarding the third dimension, there are two possibilities: (1) the implicit personalisation where the information needed for personalisation is collected automatically; and (2) the explicit personalisation where the driver has direct control over the parameterization needed for the personalisation.

One of the current trends in the automotive sector is the exploration of the personalisation of advanced driver assistance systems (ADAS) and the individualised adaptation of the driver-vehicle interaction (Hasenjäger et al., 2019; Lilis et al., 2017). Properly adapting the IVIS functionalities according to the individual characteristics and needs of a driver requires that the driver remains in the loop (Riener et al., 2016). Personalisation is seen as an important safety aspect as it has the potential to prevent distractions and cognitive overload, and ultimately reduce the risks of accidents (Lilis et al., 2017; Ulahannan et al., 2021). Additionally, according to a recent study, the personalisation of autonomy increases the familiarity and willingness to trust Autonomous vehicles (AVs) or self-driving cars (Sun et al., 2020).

As mentioned previously, there are two approaches to personalise IVIS functionalities: the explicit and the implicit approach (Fan and Poole, 2006; Stuerzlinger

et al., 2006). Although the explicit approach has the potential to promote a personalisation that is better aligned with the driver's expectation it also has some disadvantages. First, it requires drivers to direct attention and effort toward the process of personalisation, which might eventually distract them from the docking process. This direct control may also result in a limitation of the number of parameters that can be subjected to personalisation. Additionally, this limitation can be further aggravated by the “inaccessibility” of parameters which are difficult to understand (Hasenjäger et al., 2019). Finally, the driver may not be completely aware of what information is necessary to maximise the driving performance. In fact, even the safety might be compromised by a decision grounded on preferences that can prevent the presentation of critical information (Ulahannan et al., 2020; Ulahannan et al., 2021).

With the implicit approach, the information provided by the driving assistance is automatically presented or hidden. With this approach, the driver might experience that the information provided by the IVIS is not under his/her control and might not even understand the rationale of the driving assistance provided (Hasenjäger et al., 2019). However, the implicit approach has the benefit of reducing the effort, the attention and the cognitive load. Additionally, it has the potential to offer a more complex, precise and safety-oriented adaptation (Hasenjäger et al., 2019).

To implicitly (automatically) adapt the information being presented in the driving assistance, it is necessary to determine which processes will influence the adaptation. In that sense, a data-driven process is commonly employed that implies the creation of a driver model which is based on the observation of driving data (Wang et al., 2014; Lin et al., 2014). Using such a driver model, it is then possible to recognize driving behaviour or expertise in certain tasks, e.g. parking an articulated vehicle.

Besides this driver model, other information needs to be considered as an input for the adaptation (Khan and Khusro, 2020). For instance, information about the environment, the vehicle, the IVIS, as well as the preferences and profiles of the drivers. Based on such input, the application of a rule-based approach has been successfully employed to adapt the information being presented to the user (Hussain et al., 2018; Khan and Khusro, 2020). For instance, Hussain et al. (2018) followed a model-based approach to adapt the user interface. This approach is based on three different models: (1) the context model (environmental factors such as light level); (2) the user model (human factors such as experience or abilities) and (3) the device model (such as the characteristics of the hardware and

software). The resulting system is then capable to apply, at runtime, adaptation rules devised to properly adapt the user interface according to the current context. For example, whenever the light level is too low the user interface is adapted to a night mode.

3 VR-SIMULATOR PLATFORM

In our VR-Simulator (VISTA-Sim, see Figure 2), the driver wears a Virtual Reality headset that immerses the driver inside a truck cabin located at a virtual distribution centre. Inside the VR truck cabin, there is also an IVIS running on a virtual tablet computer. It is responsible to provide driving assistive information showing for example a bird-eye view of the distribution centre, the distance to the target loading bay and steering recommendations. Finally, VISTA-Sim also allows the driver to operate the truck through a steering wheel and pedals. VISTA-Sim is composed of four main components: (1) a path planner combined with the path tracking controller which runs in Simulink (MathWorks); (2) the VR-simulator that uses the Unity 3D cross-platform game engine (Unity); (3) the driver assistance Human Machine Interface (HMI); and (4) the Behaviour Analysis Module (Ribeiro et al., 2021).

The focus of this paper is to present the current work progress of the component “Driver Assistance HMI”. The focus is on the design and implementation of an adaptation mechanism that aims to provide individual driver assistance based on expertise-estimation and driving performance.

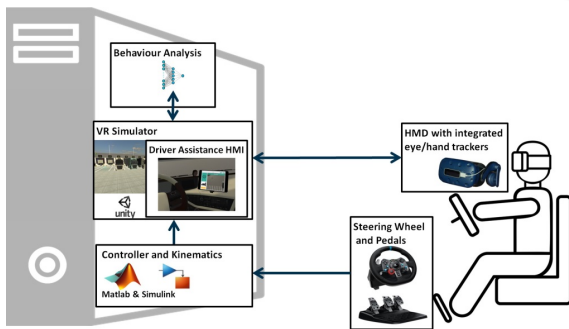


Figure 1: The high-level view of the VISTA-Sim architecture.

4 ADAPTATION MECHANISM

ADAS should provide individualised and anticipative feedback that minimizes driver distraction and provides only situation-specific, highly relevant informa-

tion. The proposed adaptation mechanism aims to adaptively control the several IVIS subsystems to provide the best fitting support for drivers given the current context. The IVIS adaptation mechanism relies on three components: (1) the Rules; (2) the Adaptor; and (3) the Models. The machine learning model that estimates driver expertise (see section 4.2) is part of the Models-component. As input, the adaptation mechanism receives a real-time expertise estimation from a driver model that is responsible to determine how good a driver is at performing the parking maneuver (Ribeiro et al., 2021). The estimation is based on real-time driving data recorded by the simulator (e.g. steering wheel angle, head movements and other data) and is used to determine when the Adaptive User Interface(AUI) should be active. For example, if the estimation predicts expert behavior, then the AUI is currently not needed. If novice driver behavior is detected, the system will adjust the IVIS in fixed intervals to match the current needs of the user.

The Rule-based component describes when and under which conditions changes to the IVIS User Interface (UI) are applied, utilizing the commonly used Event, Condition, Action (ECA) Approach (Hussain et al., 2018). The event determines when a Rule is applied, e.g. when a novice driver is detected by the performance estimator. The Action states the change that has to happen when the Event is triggered, like enabling more detailed assistance on the UI. The Condition holds information about what prerequisites need to be met so that a Rule can be applied, e.g. a disabled UI would prevent the application of any Rule with the condition “UI is on”. The ECA template was modified to have five components. First, a unique id to identify and distinguish it from other rules. Second, a description of what this rule does when and which UI elements will be affected in a human-readable way. Third, a collection of events that state when the rule is triggered. Fourth, a collection of conditions that prevent or enable it. Finally, the fifth component is a collection of actions that specify what happens to which UI element.

On a conceptual level, the Adaptor component (see Figure 3) is composed of: (1) the Evaluator; (2) the Conflict resolver; and (3) the UI-Adaptor.

Whenever the Evaluator (see Figure 4) detects that a value from the Performance estimator matches the event of one or more rules, it checks if there is any condition that prevents their application. When an event matches multiple rules, it is important to verify if the rules can be applied together. For example, the rules “In the first phase of the manoeuvre the steering assistance should be disabled” and “If the first phase of the manoeuvre duration takes longer than two min-

utes the steering assistance should be enabled” may have a potential conflict.

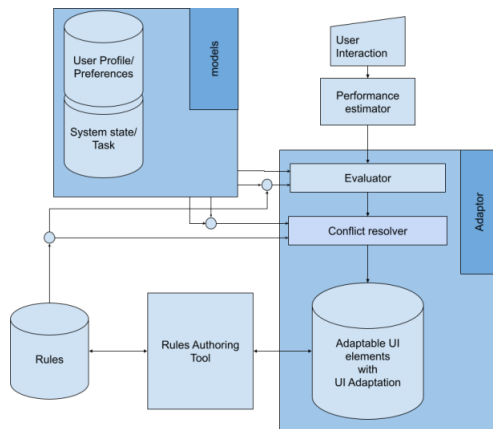


Figure 2: Detailed view of the AUI Evaluator module of the Adaptor-component. It uses four boolean decisions to select a fitting set of rules for the current context.

In that sense, the role of the Conflict resolver component is to detect and solve conflicts between the two rules. The conflicts will be detected by checking the actions that are associated with the respective rules. For example, by analysing if the rules affect the same UI object or enable the same UI modality. After detection, conflict resolution strategies must be applied. Although this component is still under development, it is envisioned to extend rules with priorities and/or the expected utility of an action (e.g. a rule with the “safety” property will have a higher priority than one without). Both approaches allow to resolve conflicts between rules by excluding rules that fit less the current requirements of docking safely. If there are conflicting rules remaining after the conflict resolution, then they are all equally fit for the current context and the resolver will return the first. Ultimately, even the active involvement of the driver in the process of conflict resolution is under consideration. For example, a user’s preference for a certain rule can be used to select this rule among the remaining rules after conflict resolving. As the last step, a UI-Adaptor component changes specific UI- elements according to the specified action of the prioritised rule.

The Models component contains information that facilitates the selection of the rules according to their applicability. Examples are task phases, user preferences or user abilities. Since this component is still at an early stage of conceptualization it will not be further discussed in this paper.

4.1 Implementation in Unity

The functional prototype is being implemented in Unity 3D according to the conceptualization men-

tioned in Section 4. The adaptation process starts in the Performance estimator where a generic Unity-Event is triggered whenever the driver’s expertise hits a value range that corresponds to a level of expertise (e.g. novice). Each level of expertise has an event attached and when triggered. It invokes a list of Rule IDs (strings keys) that are supposed to be executed. For example, for the event “driver expertise is equal to Novice” we could have one or more rules associated, such as “enable HMI”.

The Rule IDs are then processed in the Evaluator-Class where they are used as keys to select the correct Rule-Object (Rule) from a map. The implementation of a Rule contains its Actions, which are composed of a reference to all the Adaptable-UI-Objects that are affected by it as well as which methods are to be called and their parameters. Afterwards, the Actions reference is used to fetch its objects from the Object-Map to adapt the UI.

The implementation of the adaptation does not have a singular UI Adaptor. Instead, the adaptation is handled by each Adaptable-UI-Object by a script that inherits from a virtual base class which provides them with the Execute-Method that is called whenever a rule is applied. The individual version of Execute-Method handles how the adaptation is done. The current implementation of the Adaptation mechanism is not yet capable of taking system states or the user into account or resolving conflicts.

4.2 Driver Expertise Estimation Model

Using machine learning, the system can adapt to the driver based on previous behaviour. If the trained system detects unexpected driver behaviour in a familiar situation, it can trigger warnings or supportive feedback. Here, a ML-model was trained to predict the driver’s expertise on-the-fly within a certain timeframe. If a situation of low expertise is detected, the system will provide context sensitive feedback to ease the truck docking process.

Assuming that support can be helpful especially during initial training of the docking process (e.g. for novices), the ML-classifier should provide well-balanced classification accuracy for both expert- and novice behaviour. That way, novices will be provided with extensive support, while expert drivers will see only very specific, non-disturbing support.

To train a ML-model, a small dataset was collected using the VISTA Sim: Four participants with no prior truck driving experience performed a familiarization session at the first day of docking training. Three days later, the first training session started, followed by another session the next day and a final session

at the third day. Each day, five to six trials were recorded. Recording of a trial was stopped if docking failed after seven forward-backward reposition-attempts (turns). The maximum turn amount was based on a previous pilot-dataset, where an already-trained driver completed docking with no more than five turns. Two extra turns were given to ease initial learning progress and to acquire more trials with a rating above zero (see below).

The final dataset consists of 74 trials. Each trial in the dataset was rated by the first-author on a scale from 0 to 5 using the docking quality as an objective criterion (rating: 0 = did not park; 1,2: unloading not possible; 3,4: unloading possible; 5: perfect docking). This docking-quality rating was used to separate trials into novice- and expert trials. The dataset contains a large number of trials rated with 0, underlining that truck-docking is indeed a quite complicated task. Most machine learning methods work best given a balanced dataset with equal number of trials per class. Because of the comparatively small number of trials and the strong imbalance, the dataset was split into only two classes: expert- and novice trials. All trials with a zero-rating ('did not park') were assigned to the novice class (n=41) and all other trials were assigned to the expert class (n=33). With these two classes, the dataset is still moderately imbalanced.

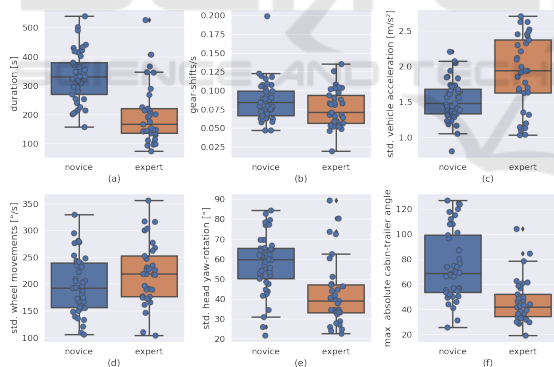


Figure 3: Features for ML-learning, extracted from the dataset. Some features correlate with expertise, e.g. head horizontal movements (e) and the cabin-trailer angle (f).

The recorded dataset was analysed with respect to several key performance indicators suggested by a professional truck-driving instructor (for details see (Ribeiro et al., 2021)). Here, we focus on performance indicators that are locally observable inside the truck cabin: docking duration, gear shifts, vehicle acceleration, steering wheel movements, head movements and the cabin-trailer angle. Figure 5 shows that some of the analysed performance indicators clearly correlate with the expertise level, visible by non- or slightly overlapping box-plots.

For comparison with ML-trained based classifiers, a simple baseline-classifier (BC) was implemented using the three features with strong predictive power (vehicle acceleration, head movements and cabin trailer angle, see Figure 5). The BC performs simple thresholding per feature and combines the feature-specific classification results using majority voting. Optimal threshold values were selected by grid-search. With two classes, we have a binary classification task: Expert trials were assigned to the positive class, and novice trials to the negative class. Applied to the complete dataset and full trial-length, the BC yields within-class accuracies of 82% for the expert class (i.e. true-positive rate (TPR), also known as recall / sensitivity) and 90% for the novice-class (true negative rate (TNR), also known as specificity) and a balanced accuracy (the average of TPR and TNR) of 86%. This is a promising but probably over-optimistic result, because the BC-thresholds were adjusted given the full dataset.

A classifier trained on full trials can only provide an expertise estimation at the end of a trial. This estimation could be used to provide adequate feedback for the next docking attempt, but a better solution would be a classifier that can estimate driver performance within a shorter timeframe (sliding window approach). Therefore, support vector machines (SVM) were trained on three different, bootstrapped datasets consisting of sub-trials with a predefined duration (25 seconds, 50s and 75s). From each trial in the dataset, 50 sub-trials were randomly sampled. The SVMs (RBF kernel, $C=0.001$) were trained using leave-one-out cross-validation, with generalisation evaluated at the level of individual trials. Hence, each trial was left out once, resulting in 74 folds. The training set of each fold was balanced using the synthetic minority over-sampling technique (SMOTE, (Chawla et al., 2002)), because SVM training is very sensitive to class-imbalance (Akbani et al., 2004). Cross-validation was repeated ten times, because SMOTE is stochastic.

SVM-Training Results: The balanced accuracy depends - as expected - on the window size. It drops only slightly from a window size of 75s to a window size of 50s (see Figure 6). Hence, a duration of 50 seconds appears to be a suitable compromise between “classifier response delay” (time until the first expertise estimation is available) and accuracy. Accuracy – especially for the expert class – drops noticeably compared to observing full trials (see baseline classifier). This means that the proposed hybrid approach, mixing ML and the rule-based approach might help to reduce uncertainty regarding observed driver expertise within shorter timeframes.

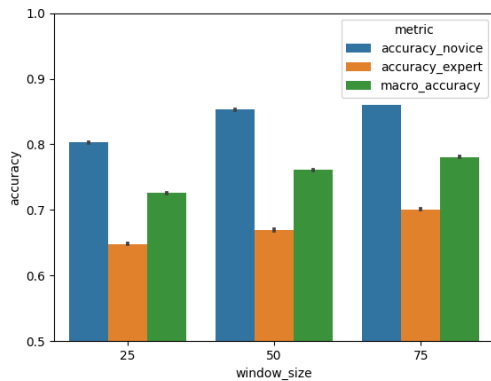


Figure 4: The performance of trained SVMs depends on the observation timeframe (window size in seconds).

5 CONCLUSIONS & FUTURE WORK

In this paper, we have presented ongoing work that targets the design and development of an adaptation mechanism for driver assistance. This adaptation mechanism will provide individualised driver assistance based on estimated driver expertise and aims to support professional truck drivers in the challenging task of backwards manoeuvring articulated vehicles towards a loading bay at a distribution centre.

Rule-based adaptation: As a next step, the design of the adaptation rules will be guided by a detailed analysis of the thinking aloud protocols and questionnaires collected during recording of the dataset, e.g. problems experienced while docking. The observations and comments will inform a post-interview on which the adaptation rules will be based. The not yet fully implemented AUI components will be further developed. The development of a task model will require a task analysis (a breakdown into phases, tasks, and subtasks) of the docking manoeuvre. Further, a detection mechanism is required to determine in which phase the driver currently is. The rule adaptation needs such a detector to know what the driver is currently supposed to do to apply rules that are made for this phase/task or sub-task. The Driver model (i.e. an individual profile) will require further research on user profiles for automotive applications. Individual driver profiles will allow the system to also take the current driver requirements (preferences, impairments, skills, etc.) into account when applying rules. For example, such an individual user model could be initialized once by the driver and then continuously updated with a running-average of previous docking-performances to provide a robust and stable adaptation of the UI. For the Resolver, the concept described in section 4 needs to be implemented, and the priority

and the expected utility properties need to be defined further.

Machine learning: The current dataset is comparatively small. Larger or more complex models like recurrent neural networks (RNN) or attention-based RNN (Vaswani et al., 2017) might show overfitting. Therefore, it is planned to evaluate and compare special machine learning methods well suited for small datasets, e.g. few-shot learning (Wang et al., 2020). Further, methods with a low energy consumption and fast training times (e.g. random projection methods, (Krause et al., 2013)) will be explored. This is important for the future goal of continuous, “lifelong” machine learning (Chen and Liu, 2018) on a low-power, embedded system inside a vehicle to continuously update an individual expertise estimator model for each driver.

Future work: In the future, it is also planned to do a study regarding implicit and explicit adaptation because - as shown in previous studies (Hussain et al., 2018; Bongartz et al., 2012; Todi et al., 2021) - frequent, unnotified changes can frustrate and confuse the user. To answer, if implicit or explicit adaptation is better for the AUI of the VISTA-Sim, it is planned to do an A/B test where one group performs docking with an implicit and the other with an explicit version of the AUI. Explicit here would likely mean by voice command e.g. “Can you help me” which would trigger the adaptation of the AUI. To not waste too many resources by implementing both versions a Wizard of Oz approach is likely.

ACKNOWLEDGEMENTS

This Interreg V-A project was co-financed by the European Union via the INTERREG Deutschland-Niederland programme, the Ministerium für Wirtschaft, Innovation, Digitalisierung und Energie of Nordrhein-Westfalen and the Province of Gelderland.

REFERENCES

- Akbani, R., Kwek, S., and Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *European conference on machine learning*, pages 39–50. Springer.
- Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al. (2020). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115.
- Bongartz, S., Jin, Y., Paternò, F., Rett, J., Santoro, C., and Spano, L. D. (2012). Adaptive user interfaces for smart environments with the support of model-based languages. In *International Joint Conference on Ambient Intelligence*, pages 33–48. Springer.
- Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.
- Chen, Z. and Liu, B. (2018). Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207.
- Fan, H. and Poole, M. S. (2006). What is personalization? perspectives on the design and implementation of personalization in information systems. *Journal of Organizational Computing and Electronic Commerce*, 16(3-4):179–202.
- Hasenjäger, M., Heckmann, M., and Wersing, H. (2019). A survey of personalization for advanced driver assistance systems. *IEEE Transactions on Intelligent Vehicles*, 5(2):335–344.
- Hussain, J., Ul Hassan, A., Muhammad Bilal, H. S., Ali, R., Afzal, M., Hussain, S., Bang, J., Banos, O., and Lee, S. (2018). Model-based adaptive user interface based on context and user experience evaluation. *Journal on Multimodal User Interfaces*, 12(1):1–16.
- Khan, I. and Khusro, S. (2020). Towards the design of context-aware adaptive user interfaces to minimize drivers’ distractions. *Mobile Information Systems*, 2020.
- Krause, A. F., Essig, K., Piefke, M., and Schack, T. (2013). No-prop-fast-a high-speed multilayer neural network learning algorithm: mnist benchmark and eye-tracking data classification. In *International Conference on Engineering Applications of Neural Networks*, pages 446–455. Springer.
- Kural, K., Besselink, I., Xu, Y., Tomar, A., and Nijmeijer, H. (2016). A driver support system for improved maneuvering of articulated vehicles using an unmanned aerial vehicle. In *HVT14: International Symposium on Heavy Vehicle Transport Technology, Rotorua, New Zealand*.
- Kusumakar, R., Kural, K., Tomar, A. S., and Pyman, B. (2017). Autonomous parking for articulated vehicles. *HAN University of Applied Science, Netherlands*.
- Lilis, Y., Zidianakis, E., Partarakis, N., Antona, M., and Stephanidis, C. (2017). Personalizing hmi elements in adas using ontology meta-models and rule based reasoning. In *International Conference on Universal Access in Human-Computer Interaction*, pages 383–401. Springer.
- Lin, N., Zong, C., Tomizuka, M., Song, P., Zhang, Z., and Li, G. (2014). An overview on study of identification of driver behavior characteristics for automotive control. *Mathematical Problems in Engineering*, 2014.
- Ribeiro, P., Krause, A. F., Meesters, P., Kural, K., van Kolfshoten, J., Büchner, M.-A., Ohlmann, J., Ressel, C., Benders, J., and Essig, K. (2021). A vr truck docking simulator platform for developing personalized driver assistance. *Applied Sciences*, 11(19):8911.
- Riener, A., Boll, S., and Kun, A. L. (2016). Automotive user interfaces in the age of automation (dagstuhl seminar 16262). In *Dagstuhl reports*, volume 6. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
- Stuerzlinger, W., Chapuis, O., Phillips, D., and Roussel, N. (2006). User interface façades: towards fully adaptable user interfaces. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 309–318.
- Sun, X., Li, J., Tang, P., Zhou, S., Peng, X., Li, H. N., and Wang, Q. (2020). Exploring personalised autonomous vehicles to influence user trust. *Cognitive Computation*, 12(6):1170–1186.
- Todi, K., Bailly, G., Leiva, L., and Oulasvirta, A. (2021). Adapting user interfaces with model-based reinforcement learning. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–13.
- Ulahannan, A., Cain, R., Thompson, S., Skrypchuk, L., Mouzakitis, A., Jennings, P., and Birrell, S. (2020). User expectations of partial driving automation capabilities and their effect on information design preferences in the vehicle. *Applied Ergonomics*, 82:102969.
- Ulahannan, A., Thompson, S., Jennings, P., and Birrell, S. (2021). Using glance behaviour to inform the design of adaptive hmi for partially automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(5):4877–4892.
- Union, I. R. T. (2007). A scientific study “etac” european truck accident causation: Executive summary and recommendations.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- Wang, W., Xi, J., and Chen, H. (2014). Modeling and recognizing driver behavior based on driving data: A survey. *Mathematical Problems in Engineering*, 2014.
- Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34.