Graphic Interfaces in ADAS: from requirements to implementation

Alessio Masola
Cristian Gabbi
University of Modena And Reggio Emilia
Dept. of Physics, Informatics and Mathematics
Modena, Italy
alessiomasola@unimore.it
224915@studenti.unimore.it

Andrea Castellano
RE:Lab s.r.l
Reggio Emilia, Italy
andrea.castellano@re-lab.it

Nicola Capodieci
Paolo Burgio
University of Modena And Reggio Emilia
Dept. of Physics, Informatics and Mathematics
Modena, Italy
{nicola.capodieci,paolo.burgio}@unimore.it

ABSTRACT

In this paper we report our experiences in designing and implementing a digital virtual cockpit to be installed as a component within the software stack of an Advanced Driving Assisted System (ADAS). Since in next-generation automotive embedded platforms both autonomous driving related workloads and virtual cockpit rendering tasks will co-run in a hypervisor-mediated environment, they will share computational resources. For this purpose, our work has been developed by following a requirement-driven approach in which regulations, usability and visual attractiveness requirements have to be taken into account by balancing their impact in terms of computational resources of the embedded platform in which such graphics interfaces are deployed. The graphic interfaces we realized consist of a set of 2D frames for the instrument cluster (for displaying the tachometer and the speedometer) and a screen area in which a 3D representation of the vehicle surroundings is rendered alongside driving directions and the point-cloud obtained through a LIDAR. All these components are able to alert the driver of imminent and/or nearby driving hazards.

CCS CONCEPTS

• **Human-centered computing → Visualization application domains**; *Information visualization*; • **Computer systems organization** → *System on a chip.*

KEYWORDS

ADAS, Visualization, HMI

ACM Reference Format:
Alessio Masola, Cristian Gabbi, Andrea Castellano, Nicola Capodieci, and Paolo Burgio. 2018. Graphic Interfaces in ADAS: from requirements to implementation. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In the last years, the topic of in-vehicle Human-Machine Interface (HMI) is becoming increasingly important. Since the complexity of driving activities is significantly increasing, also the role of information systems is rapidly evolving. Traditionally, in-vehicle information systems are designed to inform the driver about dangerous situations without overloading him/her with an excessive amount of information. For this reason, the most relevant issues related to the evaluation of the HMI from a Human Factors perspective are related to inattention and workload topics. Inattention at driving, according to taxonomy proposed by Regan [19] and Cunningham [9], can be defined as *insufficient or no attention to activities critical for safe driving* and can be brought about through a number of different mechanisms such as *driver-restricted attention* (e.g. due to biological states, such as drowsiness or fatigue) or *driver misprioritized attention* (e.g. due to focusing attention on one aspect of driving to the exclusion of another which is more critical for safe driving). According to the National Highway Traffic Safety Administration (NHTSA), the time needed to consider the driver as distracted is 2.5 seconds [18]. On the other hand, the *cognitive workload* is defined as the amount of cognitive effort required to a user to process an information or complete a task [3].

In this context, the design of Human-Machine Interface for highly automated vehicles shall address a twofold challenge, related to both cases to performance. On one hand, the HMI shall comply with design regulations and ergonomics rules in order to optimize users' performances and minimize unpaired behaviour; on the other hand, it shall be implemented and deployed in order to comply with the computational capabilities in a system performance perspective. More specifically, HMI design should account for the incremental *levels* of automation. The Society of Automotive Engineering (SAE) identified five incremental levels of semi- and full automated driving, where L1-2-3 roughly represent Advanced Driving Assistance Systems (ADAS), and L4 and L5 represent fully autonomous driving (AD) systems. Higher automation levels call for adopting stringent safety standards (e.g. ISO 26262) imposed by certification authorities, since the co-existence of critical components from the ADAS (Advanced Driving Assisted System) stack, and non-critical, HMI and infotainment systems, is only possible through a carefully design of both computing hardware and software ecosystem. Researchers acknowledge that software artifacts such as Real-Time Operating Systems and hypervisors are key technologies to solve this problem [4], as they can provide the amenable property of *isolation* among components. In this work, we analyze two real use-cases under development in the EU project Prystine [10] and we highlight how our implementation efforts balanced regulations and requirements for both an ergonomics and performance perspective.

This paper is organized as follows: in sections 2 and 3 we provide background information with regards to previous work and related standards for HMI design in ADAS scenarios. In sections 4 and 5 we detail our implementation choices for the 2D and 3D interfaces and we highlight the reasons for which our implementation choices are compliant with the mentioned requirements. Conclusive remarks are given in section 6.

2 RELATED WORK ON HMI BASED ON A USER-CENTERED DESIGN APPROACH

Human-Machine Interfaces have been usually designed mostly to include information related to the vehicle state.

The look-and-feel of in-vehicle HMI is influenced by ergonomics principles and considerations as well as by guidelines and regulations coming from standardization bodies. For example, the design of controls, indicators and telltales shall be compliant with the ISO 2575:2010. However, for road vehicles, standards are used for non-binding prescriptive use only: for example, reference icons are reported in the standard, but each designer (and/or automotive manufacturer) has significant room to customize the design solutions.

Modern vehicles, equipped with digital and reconfigurable dashboards, are able to inform the driver with a lot of information, e.g. coming from vehicle's sensors. This raises new complexity and new research topic in this field. From the ergonomics perspective, the new challenge is related to the calibration of the amount of information to be provided to the user. According to the different situations, the vehicle should be able to adapt the level and the complexity of the information. For example, when no danger is expected, the vehicle could provide the user with more details about the surrounding environment and allow him/her to activate and handle infotainment features. When there is a dangerous situation, the vehicle should minimize the information provided in order to focus the driver attention on the action to be accomplished.

The spread of highly automated vehicles is raising new challenges also from the HMI point of view. In particular, in vehicles equipped with Level 2 and Level 3 Automated Driving Functions (ADFs) the HMI continues to play an essential role. It shall be able to promptly inform the driver about the responsibility related to each agent, to keep the driver in the control loop when his/her attention is needed and to support the transition of control (e.g. when the human is expected to resume the vehicle control due to a vehicle request, i.e. the so-called Take Over Request). Recently, new research trends are emerging in this domain. Cooperative automated vehicles have been equipped with cooperative User Interfaces. In these prototypes, the role of the HMI is changing to adapt with the situation. The scope of these tools is to increase the awareness of the driver by explaining the decisions of the automation. In fact, several studies [14] [7] show that informing the driver about the rational of the decision-making is able to increase the acceptance and the trust in the automated system. This approach has been called *negotiation-based interaction* opposed but at the same time built upon the traditional *warning-based approach*. This is because, for critical situations, the traditional paradigm based on warnings is still needed for minimizing the amount of information provided to the driver and ensure a fast and safe reaction and/or transition of control. Other non-safety-critical information, however, can be provided through more complex and rich modalities. In this way, the system becomes able to exploit data coming from different sources, such as sensors for environment and driver monitoring, that generate a lot of information potentially able to enrich the driving experience and have an impact on safety. In order to fulfil this requirement, another trend recently emerged is the design of three-dimensional interfaces. This topic has become highly significant since, through the 3D mapping, the spatial awareness of the users can be increased. Several studies show that 3D view is able to improve the recognition of the semantics of spatial information [1], providing robust information. This led to an increase of the cognitive workload, that is however compensated by the accuracy of the perception. In order to avoid negative implications related to 3D in automotive, sources indicate that only certain information can be provided through three-dimensional representations. Contextual information only (i.e. data representing in 3D the environment) are suitable to be represented in 3D [13]. However, the system should always be able to shift to a simplified view when critical situations occur.

3 REQUIREMENTS FOR ADAS GRAPHIC INTERFACES

A comprehensive summary of laws and regulations with regards to HMI design in ADAS scenarios can be found in [11]. After evaluating six different ISO standards (ISO 11428, 15005, 16951, 2575, 15408-2, 26262) authors in [11] infer that design requirements fall in seven distinct categories.

User input handling: the direct interaction possibilities provided to the user must be strongly restricted and minimized. Moreover, the response time between a user input (e.g. interacting with a touchscreen) and the corresponding processing time (e.g. change in the visual representation) must be bound to a maximum of 250ms.

Dealing with multiple windows: the user must not be able to arbitrarily create different windows for showing different visual elements, not he or she should be able to decide for their positioning. If a GUI is composed of multiple visual elements then country-specific laws will impose constraints with regards to positioning and screen-area occupancy, e.g. the speedometer must always be visible and occupy x% of the viewable surface.

Three sets of requirements are related to *scheduling and virtualization*. As highlighted in previous sections, ADAS embedded platforms are commonly virtualized through a hypervisor. Inter-VM communication must be secure and access to GPU computing power must be deterministically scheduled [5, 12]. A single scheduling policy must be active in a given time instant, although dynamic policy reconfigurations are allowed. For instance, going through a state in which the vehicle is driven by the AI to a state in which the human user is called to regain the vehicle control could trigger significant variations to the system schedule and this also applies to the GUI-related tasks; in such cases, GPU arbitration policies must change accordingly.

The final category of requirements relates to *monitoring and logging capabilities*. This allows the system to detect unexpected behaviors and therefore deploy mitigating strategies for safety-threatening situations. Regarding graphics, quantities to monitor are user input response times and screen draw frequencies of critical components to render (e.g. visual signals for the state of the vehicles).

3.1 Requirements for in-vehicle HMI homologation

Specific normative standards regulate the information that must be provided to homologate a road vehicle. These regulations slightly change in different countries, and some of them regulates specific features (e.g. information related to electric vehicles). As a reference guideline, the part 571 of Federal Motor Vehicle Safety Standards [18] regulates some information always needed to homologate a vehicle information system as well as the way in which this information shall be provided. These data can be grouped in 4 categories: (i) vehicle indicators (e.g. lights, high-beam, indicators, windshield actuators), (ii) system failures and limitations (malfunction in brake system, anti-lock system, variable brake proportioning system, issues in brake pressure, low brake fluid condition, ADAS malfunction), (iii) vehicle functions issues (e.g. fuel level, engine oil pressure, electrical charge, engine stop, parking brake applied, cruise control), (iv) comfort information (all info related to HVAC). Per each of this topics, the suggested pictorial representation, wording/abbreviation, function, illumination, and colour is reported.

Moreover, regulations suggest that all the information shall be minimized in order to avoid an overload that could potentially create conflicts in driver's capabilities.

However, these standards do not take into account the requirements related to the different levels of automation, and foresee the different ADAS as distinctly separated modules, without intercepting the user perspective, i.e. the concrete action that he/she should perform when driving a partial automated vehicle.

4 2D INSTRUMENT CLUSTER

The 2D interface that we implemented and describe in this section accounts for the fact that HMIs in semi of fully-autonomous vehicles should follow the key design principles and requirements we summarized in previous sections. A key requirement we introduced refers to the minimal intrusiveness, meaning that the Graphic Interface must not distract the driver, nor it shall significantly increase the cognitive workload. At the same time it must be designed to increase the trust in automation, and reduce potential anxiety arising from the unexpected variation of roles during the transaction between manual-to-automatic, or the more critical automatic-to-manual transition.

2D HMI for L1/2 vehicle

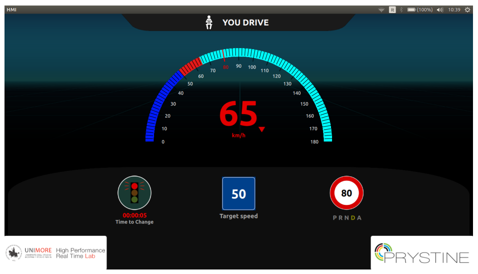


Figure 1: HMI for traffic light time-to-green with optimal speed of approach - landscape view

Figure 1 shows an HMI specifically designed for one of the use-cases of the European Prystine project [10], namely, a scenario where a connected traffic lights sends its status (color, and time-to-change) to the vehicle. The car is then able to compute the optimal approach speed by adjusting its acceleration. A L3/4 vehicle can issue a request for taking over the longitudinal control, while a lower-level (at L2) can simply inform the driver of the desired/recommended speed. Figure shows the HMI for this latter case, and we see how this information is restricted to the bottom/left of the screen, with minimal cognitive overhead for the driver

The current representation, as reported in this paper, is designed to be implemented as an instrument cluster, i.e. positioned behind the steering wheel. For this reason, a horizontal depiction with a mask frame has been conceptualized since it fits current vehicles' layout. However, one of the trends in automotive HMI design is aimed at minimizing the number of graphical interfaces, in order to provide all the information in one place, both for ergonomics, design and architectural reasons.

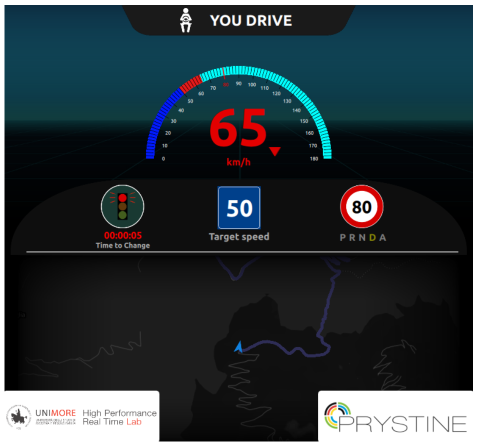


Figure 2: HMI for traffic light time-to-green with optimal speed of approach - portrait view

This is becoming particularly true for partially automated vehicles, such as Tesla Model 3, that integrates all the information in

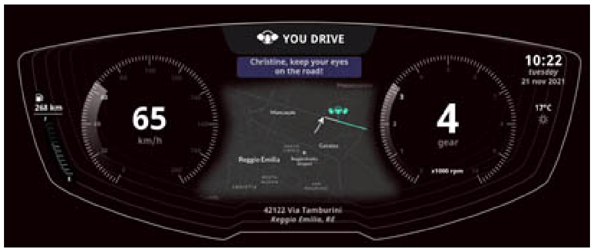


Figure 3: Step 1: simple warning

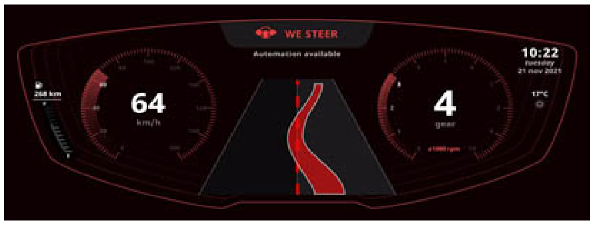


Figure 4: Step 2: lateral control

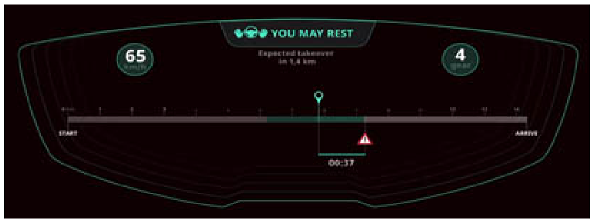


Figure 5: Step 3: full control

a single central display. This trend enables the implementation of systems able to exploit the vehicle space constraints, for example through vertical information systems (as already implemented in top-level vehicles on the market, such as Tesla Model S and Ferrari Roma). For these reasons, the integrated vertical version (the so-called "portrait" version) of the same HMI concept has to be available too, so to allow us the integration within the same frame of both driving- and infotainment-related information (as displayed in Figure 2).

2D HMI for L3/4 vehicle
Another interesting example, taken again by the use-cases of the Prystine project [10, 15], is a scenario where a L4 vehicle takes incremental control over an increasingly distracted driver. Figures 3, 4 and 5 show how the HMI look-and-feel changes, respectively to: i) issue a warning to the distracted driver, ii) taking over lateral control to avoid lane deviations, and iii) taking full control of the vehicle for a well-defined amount of time/kilometers. This is accomplished in three steps. Starting from the first HMI, the graphical elements that reflect the three system statuses increasingly change, and finally the latter one, completely removes the information on current speed and gear, which becomes useless, in favor of the time-to-control, and has a more relaxing layout, with green and white color. The transaction to full automation is now complete.

4.1 Technical design challenges

From the technical viewpoint the main challenge is being able to design a graphically scalable and composable interface using non-computationally intensive components. Under this light, it is clear that, especially at higher level of automation, the HMI should not occupy a significant amount of computational resources, which are needed by the software stack to accomplish its highly-critical tasks in real-time. The HMI for the Prystine vehicle employs lightweight components based on the QT graphical library [22], and engineers designed a system where it runs on a dedicated board (e.g., a tablet, or touchscreen), receiving from the AD stack the meta-information to show via custom lightweight yet reliable UDP connection, using a highly compact data format [8]. In those cases where the HMI does not use a dedicated computer, we resort to real-time extensions to the operating system of the in-vehicle domain controller to provide the necessary spatial and time isolation among software components, ultimately making the driving stack, and the graphical HMI co-existing on the same computing platform without interfering each other. For instance, Prystine employs an NVIDIA Drive PX [17] board, and in this case the HMI exclusively owns only one of the two integrated GPUs, leaving the other one, and the two discrete GPUs, to the highly critical tasks of the driving software.

4.2 Regulation compliance

In order to design an operational interface (i.e. close to a product ready to be deployed on real vehicles), all the homologations requirements have been taken into account. For example, in every condition the driver will be able to monitor the most relevant driving-related information, such as speed, warnings and relevant environmental information. These data have been represented in compliance (from a graphical and textual point of view) with relevant standards [18]. Moreover, considerations about legibility of textual information have been done, in order to allow the user to always be able to read all relevant information. This influenced the text type and size, considered from a realistic distance between driver' eye and the centre of the cockpit [2].

For more complex information, we followed a task-based approach, meaning that the priority in visualization has been given to the relevant information actually needed by the driver to dynamically accomplish with his/her task or sub-task in every moment. One of the crucial aspects to be addressed was to always inform the driver about his/her role, meaning that the vehicle shall always inform the driver about who is in charge of the driving task. In order to comply with this, we designed a dedicated area in the upper part of the instrument cluster where this information is always displayed. This area is made of a pictorial representation plus a textual label. Since one of the concept behind the design of this HMI is to foster a cooperative approach, we decided to use a colloquial communication tone and information not related to the vehicle state but related to the action requested to the driver (e.g. "You Drive!" instead of "Manual Mode" when manual driving is expected).

Moreover, we adopted two design strategies, both finalized at designing an uncluttered HMI: (i) we changed the representation of some information according to the control authority; (ii) we decide to include or remove some information to be provided according to the level of control. For example, as shown in Figure 4a, when the vehicle control is fully in charge of the driver, we focused on driving related information, such as vehicle speed, gear, RPM and map; when the control is fully in charge of the automation, we removed some not relevant information (e.g. the gear and the RPM) and we re-sized other info to provide more high-level visibility: for example, the map has been translated into a visualization of the overall trip status, including possible insights of potential obstacles detected in the near future. The third pillar implemented in the HMI concept was to provide explanations, in order to balance the workload and increase the driver awareness. For example, when the vehicle control is shared between the two agents, we provided an animated representation (as shown in Figure 4b) to explain (i) the reason that led to a shared control modality and (ii) the action requested to the driver. All these actions were finalized at increasing the overall trust in the system.

5 3D REPRESENTATION OF VEHICLE SURROUNDINGS

Even if 2D graphics was the first solution to be implemented for virtual cockpits, recent advancements in embedded and integrated GPU devices allows the system integrator to provide for a 3D render of the surrounding area. According to [20] the trend to merge in the same display both 2D and 3D graphics is getting more and more popular. The amount of information normally hidden to the driver that a 3D representation can offer significantly grows if the vehicle is also connected through a smart city infrastructure powered by IoT devices. To this purpose we designed and implemented a virtual cockpit able to leverage 3D real-time graphics for rendering information related a really-existing instrumented urban area in the city of Modena (northern Italy), called *Modena Automotive Smart Area* (MASA)[1]. This one square-kilometer wide area is enhanced with smart cameras, several IoT devices, edge and fog nodes able to perform real-time data processing and communicate such data to ADAS-equipped cars through a high speed network. The data collected by the car is then rendered in our 3D visualization surface in order to show road users that would be hidden by either the driver line of sight or car's sensors, traffic lights' timing, driving directions and other generic hazards (see figure 6).

Our 3D graphics are built using a thin abstraction layer able to wrap both OpenGL [21] and OpenGLES [16] API profiles, called *raylib*[2]. The city map 3D mesh is obtained using information extracted from the OpenStreetMap (OSM) database[3]. Delineation of road lanes and pedestrian crossings are rendered through a 3D pointcloud sensed from the car's LiDAR and blended in the static 3D mesh alongside with roads, buildings and pavements.

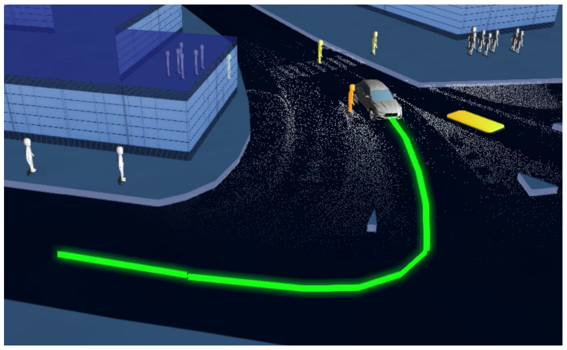
5.1 Rendering traffic information in 3D

In order to avoid excessive cognitive load for the user, our design provides for a simplified and non-realistic render of a small subset of objects that surrounds the ego-vehicle. With *ego-vehicle* we refer to the visual representation of the currently driven car: visually, the ego-vehicle is represented as the only fully textured 3D object that has a significantly higher triangle-count mesh compared to nearby road users (pedestrians and other vehicles). In Figure 6 it is shown as the white car. Pedestrians and other cars are represented as simplified place-holders. As visible in Figure 6a, buildings are drawn as transparent, hence allowing the driver to see road users within corners that would otherwise be hidden from both the driver eyes and the limited range of the ego-vehicle sensors. The smart city infrastructure is able to communicate to the car the exact position of those road users. We also render the path the ego-vehicle is supposed to take, that it is highlighted in a glowing green trajectory (Figure 6a) and the state of traffic lights, in the form of countdown to transition from green to yellow and red and viceversa (Figure 6b, implementing the 3D HMI for the aforementioned Prystine use-case). Hazardous situations are depicted by shading nearby road users using a heatmap coloring scheme that wraps an ideal ellipse centered on the ego-vehicle [18]: according to how distant a pedestrian or a nearby car is located with respect to the center of this ellipse, its color will transition from a neutral tint (white or light blue) to a yellow and then to an orange/reddish color. If positions and velocities of the ego-vehicle and the users are such to constitute an immediate danger, our 3D render would make the driver aware by using sounds and a 2D popup alternative frame so to properly signal these situations.

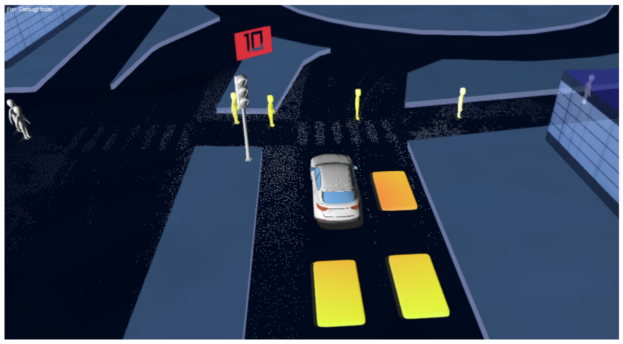
5.2 ISO regulation compliance

With regards to user input and window positioning requirements, our solution relies on a single window for the 3D render, hence no input is required from the user to alter the currently displayed visuals. In our case, responsiveness requirements relate to the latency introduced when road users positions are communicated through external sensors, pre-processed and therefore rendered. Through extensive testing we were able to bound such time in below 100ms worst case, hence well below the ISO standard requirement. Pre-processing of data involves aggregating nearby road users positions and reprojecting them through a homography to the respective locations, hence translating from GPS coordinates to the local coordinate system used in our 3D rendering. As far as scheduling is concerned, we are able to bound both period, CPU and GPU compute time for our 3D graphic application: for testing and measurements we used the NVIDIA Jetson TX2[4] which is a well-known development board featuring a six-core CPU and a high performance integrated GPU fully compliant the latest OpenGL and OpenGLES profiles. Such development platform allowed us to measure framerates (e.g. redraw frequency) in the range of 40 to 250 FPS (frame per second). Such measures implied stress-testing our application by drawing unrealistically large numbers of road users. We then activate virtual synchronization on the V-BLANK signal of our display device, hence further bounding the redraw period to multiple of 16.6 ms (60 Hz). This allows the system designer to easily account for such periods and CPU/GPU processing time for scheduling purposes. We refer the reader to the following paper for an in-depth understanding of GPU scheduling on NVIDIA platforms [6].

[1]https://www.comune.modena.it/modena-smart-community/smart-mobility/masa-modena-automotive-smart-area
[2]https://www.raylib.com/
[3]https://www.openstreetmap.org/
[4]https://devblogs.nvidia.com/jetson-tx2-delivers-twice-intelligence-edge/



(a) Ego-vehicle with driving directions highlighted as a trajectory



(b) Ego-vehicle waiting for the green traffic light

Figure 6: Visual information displayed in our 3D Virtual Cockpit

6 CONCLUSION AND FUTURE WORK

In this paper we described our experiences in designing a HMI in the context of partially and fully autonomous vehicles. We detailed our implementation choices that were driven by regulations and requirements in the context of ergonomics and safety. With this work we aim to provide useful insights for system designer in the way in which such regulations might be interpreted from various certification authorities. Our work is based on both 2D and 3D graphic interfaces that were specifically designed to maximize visual appealing without resorting to excessive computational load for the in-vehicle computers or excessive cognitive load for the human user. For validating our work, our solution leveraged a really existing smart urban area in Northern Italy in the context of the Prystine and CLASS European Projects.

ACKNOWLEDGMENTS

This work was supported by the Prystine Project, funded by Electronic Components and Systems for European Leadership Joint Undertaking (ECSEL JU) in collaboration with the European Union's H2020 Framework Programme and National Authorities, under grant agreement no. 783190. This work is also supported by the EU H2020 program under the CLASS project, grant No. 780622.

REFERENCES

[1] Patrick Bader, Niels Henze, Nora Broy, and Katrin Wolf. 2016. The Effect of Focus Cues on Separation of Information Layers. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 509–514.
[2] Charles Bigelow and Steve Matteson. 2011. Font Improvements in Cockpit Displays and their Relevance to Automobile Safety. In *Society of Information Displays 2011 Vehicle Displays and Interfaces Symposium, University of Michigan-Dearborn*.
[3] Deborah A Boehm-Davis, Wayne D Gray, Leonard Adelman, Sandra Marshall, and Robert Pozos. 2003. *Understanding and measuring cognitive workload: A coordinated multidisciplinary approach*. Technical Report. GEORGE MASON UNIV FAIRFAX VA DEPT OF PSYCHOLOGY.
[4] Paolo Burgio, Marko Bertogna, Nicola Capodieci, Roberto Cavicchioli, Michal Sojka, Přemysl Houdek, Andrea Marongiu, Paolo Gai, Claudio Scordino, and Bruno Morelli. 2017. A software stack for next-generation automotive systems on many-core heterogeneous platforms. *Microprocessors and Microsystems* 52 (2017), 299–311.
[5] Nicola Capodieci, Roberto Cavicchioli, and Marko Bertogna. 2018. NVIDIA GPU scheduling details in virtualized environments: work-in-progress. In *Proceedings of the International Conference on Embedded Software*. IEEE Press, 12.
[6] Nicola Capodieci, Roberto Cavicchioli, Marko Bertogna, and Aingara Paramakuru. 2018. Deadline-based scheduling for GPU with preemption support. In *2018 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 119–130.
[7] Andrea Castellano, Massimo Fossanetti, Elisa Landini, Fabio Tango, and Roberto Montanari. 2020. Automation as Driver Companion: Findings of AutoMate Project. In *International Conference on Intelligent Human Systems Integration*. Springer, 1048–1054.
[8] Roberto Cavicchioli. 2019. The MASA Protocol. https://github.com/HiPeRT/MASA_protocol.
[9] Mitchell Cunningham and Michael A Regan. 2015. Autonomous vehicles: human factors issues and future research. In *Proceedings of the 2015 Australasian Road safety conference*, Vol. 14.
[10] Norbert Druml, Georg Macher, Michael Stolz, Eric Armengaud, Daniel Watzenig, Christian Steger, Thomas Herndl, Andreas Eckel, Anna Ryabokon, Alfred Hoess, Sumeet Kumar, George Dimitrakopoulos, and Herbert Roedig. 2018. PRYSTINE - PRogrammable sYSTems for INtelligence in automobilEs. https://doi.org/10.1109/DSD.2018.00107
[11] Simon Gansel, Stephan Schnitzer, Frank Dürr, Kurt Rothermel, and Christian Maihöfer. 2013. Towards virtualization concepts for novel automotive HMI systems. In *International Embedded Systems Symposium*. Springer, 193–204.
[12] Cheol-Ho Hong, Ivor Spence, and Dimitrios S Nikolopoulos. 2017. GPU virtualization and scheduling methods: A comprehensive survey. *ACM Computing Surveys (CSUR)* 50, 3 (2017), 1–37.
[13] Ho Gi Jung, Dong Suk Kim, Pal Joo Yoon, and J Hie Kim. 2006. 3D vision system for the recognition of free parking site location. *International journal of automotive technology* 7, 3 (2006), 351–357.
[14] Jeamin Koo, Jungsuk Kwac, Wendy Ju, Martin Steinert, Larry Leifer, and Clifford Nass. 2015. Why did my car just do that? Explaining semi-autonomous driving actions to improve driver understanding, trust, and performance. *International Journal on Interactive Design and Manufacturing (IJIDeM)* 9, 4 (2015), 269–275.
[15] Mauricio Marcano, Sergio E. Diaz, Joshué Pérez, Andrea Castellano, Elisa Landini, Fabio Tango, and Paolo Burgio. 2020. Human-Automation Interaction Through Shared and Traded Control Applications. In *Intelligent Human Systems Integration 2020 - Proceedings of the 3rd International Conference on Intelligent Human Systems Integration (IHSI 2020): Integrating People and Intelligent Systems, February 19-21, 2020, Modena, Italy*. 653–659. https://doi.org/10.1007/978-3-030-39512-4_101
[16] Aaftab Munshi, Dan Ginsburg, and Dave Shreiner. 2008. *OpenGL ES 2.0 programming guide*. Pearson Education.
[17] NVIDIA. 2017. NVIDIA DRIVE PX: scalable AI Supercomputer For Autonomous Driving. https://www.nvidia.com/en-us/self-driving-cars/drive-px/
[18] United States Department of Transportation. 2020. Federal Motor Vehicle Safety Standards (FMVSS). https://www.nhtsa.gov/laws-regulations/fmvss.
[19] Michael A Regan, Charlene Hallett, and Craig P Gordon. 2011. Driver distraction and driver inattention: Definition, relationship and taxonomy. *Accident Analysis & Prevention* 43, 5 (2011), 1771–1781.
[20] Stephan Schnitzer, Simon Gansel, Frank Dürr, and Kurt Rothermel. 2016. Real-time scheduling for 3d gpu rendering. In *2016 11th IEEE Symposium on Industrial Embedded Systems (SIES)*. IEEE, 1–10.
[21] Mark Segal and Kurt Akeley. 2010. The OpenGL graphics system: a specification, version 4.5 (core profile). *Retrieved May* 16 (2010), 2010.
[22] The QT Company. 2020. QT. Cross-platform development for embedded and desktop. https://www.qt.io/.