Preprint version; final version available at https://ieeexplore.ieee.org/

HammerDrive: A Task-Aware Driving Visual Attention Model

Pierluigi Vito Amadori, *Member, IEEE*, Tobias Fischer, *Member, IEEE*, Yiannis Demiris, *Senior Member, IEEE*

***Abstract*—We introduce HammerDrive, a novel architecture for task-aware visual attention prediction in driving. The proposed architecture is learnable from data and can reliably infer the current focus of attention of the driver in real-time, while only requiring limited and easy-to-access telemetry data from the vehicle. We build the proposed architecture on two core concepts: 1) driving can be modeled as a collection of sub-tasks (maneuvers), and 2) each sub-task affects the way a driver allocates visual attention resources, i.e., their eye gaze fixation. HammerDrive comprises two networks: a hierarchical monitoring network of forward-inverse model pairs for sub-task recognition and an ensemble network of task-dependent convolutional neural network modules for visual attention modeling. We assess the ability of HammerDrive to infer driver visual attention on data we collected from 20 experienced drivers in a virtual reality-based driving simulator experiment. We evaluate the accuracy of our monitoring network for sub-task recognition and show that it is an effective and light-weight network for reliable real-time tracking of driving maneuvers with above 90% accuracy. Our results show that HammerDrive outperforms a comparable state-of-the-art deep learning model for visual attention prediction on numerous metrics with $\sim$13% improvement for both Kullback-Leibler divergence and similarity, and demonstrate that task-awareness is beneficial for driver visual attention prediction.**

***Index Terms*—Advanced Driver-Assistance Systems, Visual Attention, Task Recognition, Simulated Driving, HAMMER.**

I. Introduction

DISTRACTION, or misplaced attention, is regarded as the leading cause of vehicle accidents [1]. Several studies have confirmed this and have found distinct connections between driving accidents to some form of distraction [2], [3]. While advances in computing have fueled increasingly complex and intelligent systems for active safety assistance in driving, active monitoring still represents a challenge for the deployment of Advanced Driver-Assistance Systems (ADAS). Active monitoring systems are required to timely and reliably evaluate both the driver's actual focus of attention and the ideal focus of attention for the driving task [4], [5].

In this paper, we introduce a model for visual attention prediction in driving. In line with the literature [5], [6], we formalize the problem as human visual attention modeling, or eye fixation prediction, which has been an active research topic in computer vision, robotics and neuroscience for many years [7]. Human visual attention modeling can be described as the task of inferring the focus of attention of a human observer when looking at images or videos. Led by the pioneering work by Itti et al. [8], many studies have focused on designing computational vision attention models that can predict human eye fixations in static image observation [9], [10].

The advent of deep neural networks, together with large-scale publicly available datasets and benchmarks, have further improved static visual attention models, up to the point where it is not possible to differentiate model predictions from human fixation maps [9], [11]. However, visual attention models for static image viewing cannot address nor leverage the known correlation between human fixation patterns and time [12]. Static-scene viewing models assume the observer to have multiple seconds of observation over a single image, while dynamic-viewing, as in driving, is characterized by significantly shorter times per frame [13]. Behavioral studies have also shown that motion is a key component in human attention [14]. The ability to predict human eye fixations in dynamic environments has several real-world applications, spanning from attention-based video compressing [15] to human-robot-interactions [16], [17].

In this paper, we focus on visual attention prediction in highway driving, where it offers significant applications for ADAS [18], such as blind spot control, distraction detection or lane-change assistance [19], [20]. Recently, ADAS have experienced an increasing interest in research as they aim to improve driving safety and comfort. While active safety features, e.g., collision avoidance and lane change assistance [21], [22], have achieved notable success, many challenges still exist for driver monitoring-based assistance [4], [23]. Among these, the major source of complexity is the need to be able to predict future intentions of drivers and their attention allocation in a continuous and reliable manner [4]. Furthermore, ADAS have additional computational constraints, as they need to be deployed on platforms with limited resources, such as embedded systems [24].

In this context, we introduce a novel architecture, namely HammerDrive, for visual attention prediction in simulated driving. The proposed architecture exploits real-time maneuver recognition for task-aware visual attention modeling, as overviewed in Fig. 1. HammerDrive uses easy-to-access telemetry data from the vehicle, i.e., location, steering angle, speed and throttle, to perform reliable and real-time prediction of visual focus of the driver. Driving maneuver recognition is performed by a Hierarchical, Attentive, Multiple Models for Execution and Recognition (HAMMER) [25] network, which is a general framework for recognizing and executing actions by selecting modules depending on their prediction error. Visual attention prediction is performed via ParRMDN, a task-dependent ensemble of recurrent mixture density networks

P.V. Amadori, T. Fischer and Y. Demiris are with the Personal Robotics Lab, Dept. of Electrical & Electronic Engineering, Imperial College London.

Manuscript received April 28, 2020; revised October 01, 2020, November 26, 2020; accepted January 15, 2021.

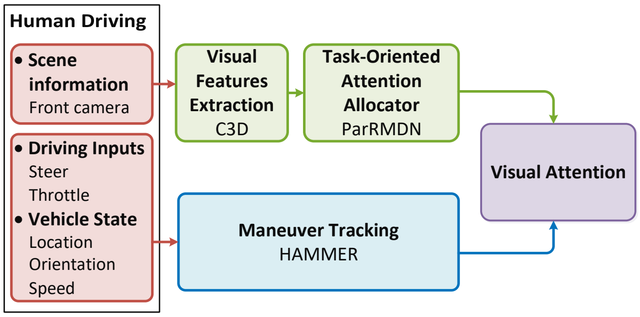


Fig. 1. Overview of HammerDrive. The proposed architecture uses scene and telemetry data (red). HAMMER (blue) uses telemetry data to track the current driving maneuver in real-time. A front camera provides scene information to a convolutional 3D (C3D) network, a feature extraction module. These features are processed by ParRMDN (green), an ensemble of neural networks, whose outputs are scaled according to the HAMMER task feedback signal.

(RMDNs) [26]. In HammerDrive, HAMMER recognizes in real-time the maneuver the driver is currently performing and sends this information to ParRMDN, which predicts the focus of attention of the driver. Although we investigate the application of the proposed framework in simulated driving, we advocate that the highly modular nature of HAMMER allows our model to be generalized also to non driving-related applications. The proposed architecture is trained and evaluated on data acquired from our custom driving simulator.

The contributions of the paper are as follows:

1) We present a novel architecture for real-time driver focus prediction, where multiple visual attention modules are dynamically activated via a task-driven attention scheduler;
2) We evaluate HAMMER task-monitoring performance over three different driving maneuvers, namely lane maintenance, lane changes to the left and right, and investigate performance-complexity trade-offs of data-driven and model-based implementations;
3) We analyze and compare the proposed task-aware visual attention architecture against RMDN in terms of information gain, Kullback-Leibler divergence, similarity and cross correlation.

The rest of the paper is organized as follows: Section II provides an overview of related works. Section III formalizes the problem of visual attention prediction and introduces the proposed architecture. Section IV focuses on an in-depth explanation of HAMMER and its implementation for simulated driving, while Section V describes both implementation and training procedure of the RMDN modules. The data acquisition methodology is described in Section VI, highlighting both the experimental setup and data collection process. Section VII analyses the results and performance achieved by the proposed architecture. Section VIII discusses limitations of HammerDrive and lists open challenges. Finally, Section IX summarizes the contributions of the paper and outlines future research directions.

II. Related Works

HammerDrive is related to computational visual attention modeling in driving (Section II-A) and biologically inspired driver behavior modeling for Advanced Driver-Assistance Systems (Section II-B).

A. *Visual Attention Modeling in Driving*

Driving represents a key application of visual attention modeling in top-down driven daily tasks [11]. We identify two main approaches to infer driver visual attention: one where it is estimated via an interior camera facing the driver exclusively [27], [28], [29], and one where it is combined with an additional camera facing the scene [5], [6], [30], [31], [32], [33].

The works in [27], [28], [29] exploit interior cameras facing the driver to estimate with high precision whether the driver is focusing their attention at the rear-view mirror, windshield or dashboard. However, such approaches cannot infer the focus of attention within the scene in front of the driver [27], [29], but only whether the driver is looking through the windshield. To estimate if the driver is looking at a crossing pedestrian or traffic lights, a secondary camera, such as the one assumed for HammerDrive or maneuver assistance [22], [34], is needed.

In line with this, Palazzi et al. in [5] introduced a multiple bottom-up branch model that employs Convolutional Neural Networks (CNNs) and leverages on visual information from the scene, motion from optical flow and semantic segmentation to refine fixation maps. Similarly, [30] introduced a fully CNN-based network for human attention prediction in driving video observation. These models achieve state-of-the-art performance in video saliency detection in driving, however they are inherently task-unaware. Sensory based, i.e., bottom-up, approaches are limited in visual attention modeling, as past literature showed that human attention patterns greatly depend on the task and sub-tasks, especially in driving [35]. Recent task-aware top-down approaches have proven to be very promising for visual attention modeling in driving, but they have been tested on datasets where observers were either looking at static images [6], [32] or not actively driving [31]. However, human vision and actions are strongly correlated, as gazes are used to collect the information required to perform an action [36]. Toward this end, HammerDrive integrates active real-time maneuver tracking and exploits this information to guide visual attention prediction towards meaningful goals for the driver in a top-down manner.

B. *Biologically Inspired Driver Behavior Modeling for ADAS*

The proposed framework for visual attention prediction is also closely related to driver modeling for Advanced Driver-Assistance Systems (ADAS). Driver modeling can be applied to a large number of problems; here, we focus on the challenging task of building human-like models of expert drivers to intervene in a shared-control manner.

Building on the concept of artificial co-driver [37], authors in [38], [39] have designed biologically inspired multi-layered systems that replicate expert human behavior to provide shared-control assistance in driving. Similarly to HammerDrive, these works build on modular hierarchical systems that perform goal-oriented action prediction. The system simultaneously computes action request signals related to all plausible goals, and then sorts them according to a fitness criterion to perform take-over maneuvers. Authors in [40] have developed an assistance system that combines driver modeling to mirror human-like

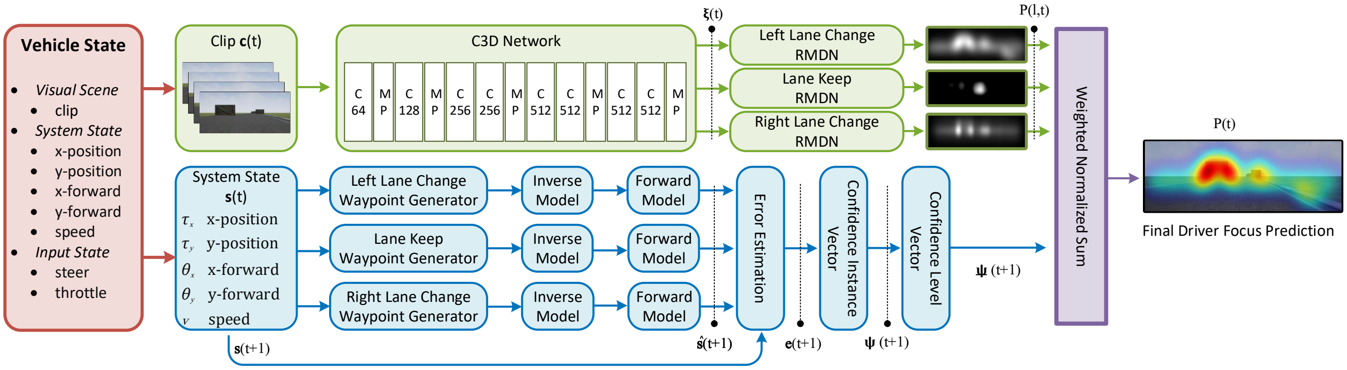


Fig. 2. Visual attention inference procedure in HammerDrive. The block to the left (red) lists the sensor readings required during validation. The bottom blocks (blue) show task tracking using HAMMER network via easy-to-access telemetry data. The top network (green) depicts both C3D and ParRMDN and represents the visual attention predicting component of HammerDrive. The outputs from both networks are combined as a weighted-normalized sum (purple).

maneuvers with driver monitoring to detect drowsiness and inattention to trigger take-overs from the system.

The proposed HammerDrive architecture is also closely related to the Adaptive Control of ThoughtRational (ACT-R) driver model from [41], [42], which has been successfully applied to describe driver behavior in a vast number of scenarios. The model builds on the ACT-R architecture [43], which exploits on modularity, seriality and parallelism to infer driver behavior and focus of attention during driving. The ACT-R model comprises of three primary components: control, monitoring and decision making. The control component directly relates to HammerDrive, as it regulates the relationship between perception and vehicle manipulation for lateral and longitudinal control, i.e., steering and accelerating, respectively.

Driver modeling can also be applied to autonomous vehicles [44], as human-like behavior is often considered safer and more acceptable for users [45]. In line with the above, the proposed framework defines a human-like computational model of visual attention, by building on biological evidence of action planning methodologies in humans [25].

III. Problem and Model

We formalize driver visual attention modeling as a special case of eye gaze fixation prediction. Eye fixation prediction is defined as a general problem of function estimation.

A. Problem Formulation

Given a group of N_{sub} subjects, we define the corresponding visual attention dataset as

$$\mathcal{D} = \left\{ (\mathbf{x}_i^s \mid \mathbf{c}_i^s)_{i=1}^{N} \right\}_{s=1}^{N_{\text{sub}}}, \qquad (1)$$

where the tuple $(\mathbf{x}_i^s \mid \mathbf{c}_i^s)$ identifies two t_f-long sequences: a) a sequence of gaze locations $\mathbf{x}_i^s = [x_i^s(t)]_{t=0}^{t_f}$ and b) a sequence of frames $\mathbf{c}_i^s = [c_i^s(t)]_{t=0}^{t_f}$, which we refer to as a *clip*. Sub-indices i and super-indices s differentiate between data samples and subjects, respectively. In other words, $x_i^s(t)$ is the instantaneous gaze location of the s-th subject while they were looking at frame $c_i^s(t)$.

The task of visual attention modeling is the derivation of a function $f(\cdot)$, that can estimate the most likely gaze location of a driver $\hat{x}_i^s(t)$ given a specific frame $c_i^s(t)$. This corresponds to the following optimization problem

$$\mathcal{P}: \quad \min_{f\in\mathcal{F}} \sum_{s=1}^{N_{\text{sub}}} \sum_{i=1}^{N} \Gamma\left(f(\mathbf{c}_i^s), \mathbf{x}_i^s\right), \qquad (2)$$

where the operator $\Gamma(\cdot)$ is used to identify the chosen loss function, in our case the negative log-likelihood, see Section V. In the following subsection, we introduce the proposed model, namely HammerDrive, to solve the optimization problem $\mathcal{P}$.

B. HammerDrive Model

The proposed HammerDrive model, visually presented in Fig. 2, is characterized by two main networks: ParRMDN, an ensemble of Recurrent Mixture Density Networks (RMDN) for visual attention prediction (top, green) and a HAMMER-based action recognition network (bottom, blue). Our architecture builds upon the concept that human visual attention can be modeled as an ensemble of bottom-up networks competing for resources, given top-down task-aware guidance. Under this assumption, the competition for resources is modeled and controlled via a task-aware top-down driven attention network.

1) **HAMMER** acts as a top-down driven attention network that recognizes in real-time the maneuver of the driver and instructs the ParRMDN network on which modules to activate. The network can be formalized as an ensemble of multiple parallel modules [25], each designed to provide the confidence/likelihood of a specific driving maneuver. The HAMMER network offers great flexibility in the implementation of its modules, allowing the coexistence of modules based on kinematic models, neural networks or Kalman filters, see [46]. After the confidence values of all modules have been computed, they are used to perform a weighted sum of ParRMDN predictions. We describe HAMMER in Section IV.

2) **ParRMDN** operates as an ensemble of RMDN modules, each trained to predict visual attention in the occurrence of a specific driving maneuver. The modules are defined according to [26] and build upon the concept of cascading a recurrent neural network with mixture density networks [47]. We perform feature extraction over a t_f-long clip of images via a pre-trained Convolutional 3D (C3D) network [48]. We introduce the formulation and training of RMDN modules in Section V.

IV. HAMMER

Task-awareness in HammerDrive is achieved by means of a HAMMER-based action recognition network [25]. The network recognizes a set of actions (maneuvers) $\mathcal{L}$ with cardinality $\|\mathcal{L}\| = L$ and consists of L pairs of inverse and forward models operating in parallel. This section provides a detailed introduction to HAMMER's paradigm, together with its pseudocode implementation in Algorithm 1.

A. Overview

Considering a maneuver $l \in \mathcal{L}$, we define the corresponding inverse model f_I^l as the function that takes as input the system state $\mathbf{s}\,(t)$ at time step t, and computes the control commands $\mathbf{a}_l\,(t)$ to be applied to the system to achieve that goal. The forward model f_F^l takes as input the current state of the system $\mathbf{s}\,(t)$ and control commands $\mathbf{a}_l\,(t)$ to compute the predicted state of the system at the next time step $\hat{\mathbf{s}}_l\,(t+1)$. Once the predicted state of the system for the l-th module has been computed by sequentially evaluating inverse and forward models, it is then compared to the actual system state at the next time step $\mathbf{s}\,(t+1)$. The comparison results in an error signal that is used to increase or decrease the confidence value of the maneuver that corresponds to the l-th module. The maneuver corresponding to the module with the highest confidence value is considered as an estimate of the driver's intention.

In our study, we focus on $L = 3$ maneuvers: 1) *lane maintenance*, 2) *lane change to the left* and 3) *lane change to the right*. Since all these maneuvers relate to control inputs over the steering wheel, the definition of corresponding inverse-forward model pairs, i.e., the cascade of $f_I^l - f_F^l,\ \forall\ l \in \mathcal{L}$, is equivalent. Although analytically equivalent, each inverse-forward model pair differs in the tangential angle assumed for the way-point. When defining the inverse and forward models for driving maneuvers, we assume the vehicle to be moving on a $\mathbb{R}^2$ workspace $\mathcal{W}$.

We define the state vector $\mathbf{s}(t)$ in time t as

$$\mathbf{s}(t) := [\tau_x(t), \tau_y(t), \theta_x(t), \theta_y(t), v(t)]\,, \tag{3}$$

where $[\tau_x(t),\ \tau_y(t),\ v(t)]$, represent the x-axis, y-axis location of the car and speed, respectively, and $[\theta_x(t),\ \theta_y(t)]$ identify the x and y-axis components of the unitary forward vector of the car, i.e., the car heading.

Similarly, we define the input state vector $\mathbf{u}(t)$ as follows

$$\mathbf{u}(t) := [w(t), g(t)]\,, \tag{4}$$

where $w(t) \in [-w_{max}, w_{max}]$ represents the steering wheel angle, with w_{max} being the maximum turning angle, and $g(t) \in [0, g_{max}]$ is the throttle pedal position of the car, with g_{max} being the maximum allowed. We normalize both steering and pedal range using a simple min-max normalization, hence $w(t) \in [-1, 1]$ and $g(t) \in [0, 1]$.

B. Inverse Model Formulation

We formulate the inverse models of HAMMER as functions that compute the steering input required to move the car from its current state $\mathbf{s}(t)$ towards the L possible behavioral locations.

Algorithm 1 HAMMER Algorithm

```
Input: State vectors $\mathbf{s}(t)$ and $\mathbf{s}(t+1)$
Output: Recognized maneuver $l$
  Receive new observation from system $\mathbf{s}(t+1)$
  for $l \in \mathcal{L}$ do
      Compute way-point direction $\phi_l$
      Compute steering input $\hat{w}(\phi_l)$ via Eq. (5)
      Compute predicted state input $\hat{\mathbf{s}}(t+1)$ via Eq. (6)
      Compute error value $\mathbf{e}(t+1)$ via Eq. (8)
  end for
  Identify most probable module as $\arg\min \mathbf{e}(t+1)$
  Compute confidence instance $\boldsymbol{\psi}(t+1)$ via Eq. (9)
  Compute confidence level vector $\underline{\boldsymbol{\psi}}(t+1)$ via Eq. (10)
  return Identify recognized maneuver as $\arg\max \underline{\boldsymbol{\psi}}(t+1)$
```

We employ an orientation correction approach [49], where we discretize a unidirectional road into three separate virtual lanes [4]. Given the current state of the car $\mathbf{s}(t)$ we define three way-points or maneuver goals, one per behavioral location and located at fixed road-distance from the location of the vehicle. After the way-points have been identified and their location has been computed, the algorithm evaluates the angular distance between the direction of the car in its current state, i.e., $[\theta_x, \theta_y]$, and the tangential angle of the l-th way-point ϕ_l. The angular differences identify the set of steering angles required for the car to proceed towards the corresponding way-points and is used to compute an estimate of the steering input $\hat{w}(t)$ via linear mapping. We define the inverse model function for steering as

$$f_I^l : \hat{w}(\phi_l) = \frac{(\theta - \phi_l)}{w_{max}} = \frac{(\arctan(\theta_y/\theta_x) - \phi_l)}{w_{max}}, \tag{5}$$

where θ represents the direction of the car or yaw and ϕ_l identifies the tangential angle for the l-th way-point. Time-dependence of parameters has been removed to ease notation.

C. Forward Model Formulation

The forward model computes the expected behavior of the vehicle, given its current state and inputs. The formulation of a forward model is inherently a trade-off between accuracy and complexity. In our case, we favor simplicity over accuracy, as the proposed architecture is required to rapidly and simultaneously compute the expected behavior of the vehicle for multiple maneuvers. We assume a kinematic bicycle model, which was shown to achieve good accuracy when modeling vehicle behavior in real-driving scenarios, despite its simplicity [50]. Under these assumptions, if the time interval between observation δ_t is small, the car moves in the same direction of its rear wheels, i.e., the car direction (θ_x, θ_y) in our case. Following the same notation as for the inverse model, we formulate the predicted state of the car for the l-th goal as

$$f_F^l : \hat{\mathbf{s}}_l(t+1) = \left[\begin{array}{llll} \hat{\theta}_x & = \cos(\theta + \hat{w}_l \cdot w_{max}) & & \\ \hat{\theta}_y & = \sin(\theta + \hat{w}_l \cdot w_{max}) & & \\ \hat{v} & = v_0 + \dot{v} \cdot \delta_t & = v_0 + a \cdot \delta_t & \\ \hat{\tau}_x & = x_0 + \dot{x} \cdot \delta_t & = x_0 + \hat{v} \cdot \cos(\theta) \cdot \delta_t & \\ \hat{\tau}_y & = y_0 + \dot{y} \cdot \delta_t & = y_0 + \hat{v} \cdot \sin(\theta) \cdot \delta_t & \end{array}\right]. \tag{6}$$

While we presented a kinematics-based approach for HAMMER, its high modularity allows the inclusion of more complex models, such as inverse-forward pairs of Neural Network (NN) models, which we will evaluate in Section VII-C.

D. Confidence Extractor

Once inverse-forward model pairs have been computed for all the considered maneuvers, their outputs are used at a prediction verification stage to generate a set of error signals. We define the $\mathbb{R}^{L\times 1}$ error vector $\mathbf{e}(t+1)$ as

$$\begin{aligned}\mathbf{e}(t+1) &= [e_l(t+1),\ \forall\ l \in \mathcal{L}] \\ &= [\epsilon(\hat{\mathbf{s}}_l(t+1), \mathbf{s}(t+1)),\ \forall\ l \in \mathcal{L}]\,,\end{aligned} \tag{7}$$

where $e_l(t{+}1)$, $\hat{\mathbf{s}}_l(t{+}1)$ and $\mathbf{s}(t{+}1)$ identify the error value, the predicted state and the actual state corresponding to the l-th maneuver, respectively. Here, the operator $\epsilon\left(\hat{\mathbf{s}}_l(t+1), \mathbf{s}(t+1)\right)$ is used to represent the error function that compares the state vectors received in argument. Without loss of generality, we compute $\epsilon(\cdot)$ as the Euclidean distance between the predicted forward direction vector of the vehicle for the l-th maneuver $[\hat{\theta}_{x,l}, \hat{\theta}_{y,l}]$ and its true direction. Therefore, we have

$$\epsilon(\hat{\mathbf{s}}_l(t+1), \mathbf{s}(t+1)) = \sqrt{(\hat{\theta}_{x,l} - \theta_x)^2 + (\hat{\theta}_{y,l} - \theta_y)^2}. \tag{8}$$

The error vector $\mathbf{e}(t{+}1)$ is then used to compute a confidence instance vector $\boldsymbol{\psi}(t+1)$. The confidence instance is a $L \times 1$ vector whose elements are all zeros, except for the element that corresponds to the goal with the lowest error value, which is given a unitary value:

$$\psi_l(t+1) = \begin{cases} 0 \text{ if } l \neq \arg\min \mathbf{e}(t+1) \\ 1 \text{ if } l = \arg\min \mathbf{e}(t+1) \end{cases}. \tag{9}$$

Confidence instances are then collected in a confidence level vector $\underline{\boldsymbol{\psi}}(t+1)$ according to a confidence window of length c_w. We define the confidence level vector as the weighted sum of the c_w previous confidence instances:

$$\underline{\boldsymbol{\psi}}(t+1) = \sum_{i=0}^{c_w} a_i \boldsymbol{\psi}(t+1-i), \tag{10}$$

where a_i identifies the time-based weights for the confidence instances. In our implementation, unless differently specified, we assume a confidence window $c_w = 1s$ as it shows a positive trade-off between real-time predictions and performance (see Section VII-C), and we consider a simple linear weighting scheme for confidence instances where recent instances have higher influence for task recognition than past ones, i.e., $a_i = (c_w - i)/c_w$.

Finally, the task corresponding to the highest element in the confidence level vector is identified as the maneuver that the driver is currently performing. This signal is propagated through the network and is used to perform a weighted sum of the visual attention prediction outputs of ParRMDN, as shown in Fig. 2. More details on visual attention networks are provided in the next section.

V. RMDN MODULE

This section introduces the modules of ParRMDN. The network is an ensemble of RMDN networks following a single C3D feature extraction network [26], [48], see Fig. 3. The feature vectors are then input to the L modules, each characterized by a cascade of a long short-term memory network (LSTM) [51] and a mixed density network (MDN).

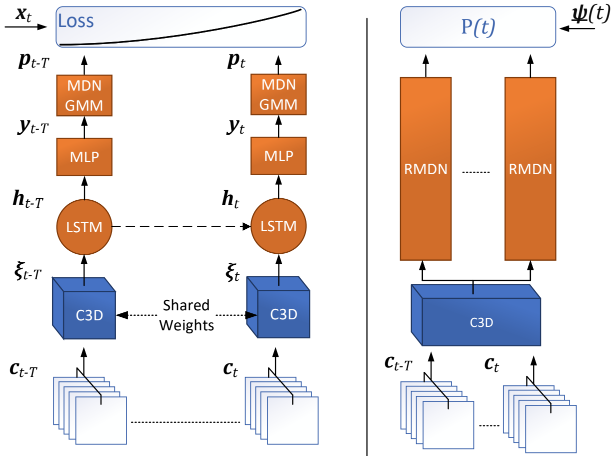


Fig. 3. ParRMDN module block diagram. During training (left), a T-long sequence of 16 frames clips $\mathbf{c} = \{\mathbf{c}_{t-T}, ..., \mathbf{c}_t\}$ is input to a C3D network (pre-trained on Sports-1M), which outputs a sequence of feature vectors $\mathbf{x} = \{\mathbf{x}_{t-T}, ..., \mathbf{x}_t\}$. The sequence is fed to a LSTM network, whose hidden states $\mathbf{h}_t$ are projected via multilayer perceptron (MLP) to a vector of Gaussian parameters $\mathbf{p}_t$. The loss is the negative log-likelihood of the ground-truth gaze location against the mixture of Gaussians $\mathbf{p}_t$ via Eq. (19). During inference (right), each clip of a T long sequence is first processed using a shared C3D network. The sequential output $\mathbf{x}$ is then simultaneously fed to the RMDN modules, each providing a maneuver-dependent bivariate Gaussian prediction. The predictions are then combined according to Eq. (21).

A. Model Formulation

We assume a dataset with structure $\mathcal{D} = \left\{\left(\mathbf{v}^i, \mathbf{x}^i, l^i\right)\right\}_{i=1}^N$, where N identifies the number of available triples data-points. Here, $\mathbf{v}^i$ represents the video data, $\mathbf{x}^i$ is used to represent the ground-truth gaze locations and l^i the associated sub-task. Video data $\mathbf{v}^i$ comprises $T = 14$ overlapping clips $\mathbf{c}_t^i$ of $t_f = 16$ frames, i.e., $\mathbf{v}^i = \left(\mathbf{c}_t^i\right)_{t=0}^{T-1}$, which corresponds to scene information spanning $T + t_f = 30$ frames, i.e., $1s$. Gaze data points $\mathbf{x}^i$ are tuples of $(x, y)_i$ positions, normalized to $[0, 1]$.

Given the i-th datapoint of $\mathcal{D}$, RMDN first extracts the C3D features of the input clip as $\boldsymbol{\xi}_t = \text{C3D}(\mathbf{c}_t^i)$. This operation is performed independently from the task, i.e., ParRMDN only requires a single C3D network. C3D is defined as [48]: C64-MP-C128-MP-C256-C256-MP-C512-C512-MP-C512-C512-MP, where C represents a three-dimensional convolutional layer, MP is the max-pooling layer and the number specifies the number of kernels of the layer. The first MP layer has kernel (1, 2, 2), whereas all others have a (2, 2, 2) kernel.

Once the vector of features $\boldsymbol{\xi}_t$ has been computed, the LSTM network operates as follows:

$$i_t = \sigma(W^i h_{t-1} + I^i \boldsymbol{\xi}_t + b_i) \tag{11}$$

$$f_t = \sigma(W^f h_{t-1} + I^f \boldsymbol{\xi}_t + b_f) \tag{12}$$

$$o_t = \sigma(W^o h_{t-1} + I^o \boldsymbol{\xi}_t + b_o) \tag{13}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W^c h_{t-1} + I^c \boldsymbol{\xi}_t + b_c) \tag{14}$$

$$h_t = o_t \odot \tanh(c_t), \tag{15}$$

where i_t, f_t, o_t, c_t and h_t identify input gate, forget gate, output gate, memory cell and hidden representation, respectively. Operators σ and $\tanh$ represent the element-wise sigmoid function and hyperbolic tangent function, respectively. The parameters to be learned are W^*, I^* and b^*, where $*$ is used to represent $\{i, f, o\}$.

The hidden representation h_t of the LSTM network is input to a linear layer that projects the parameters to a mixture of C bivariate Gaussians to compute the two-dimensional visual attention map. Given a re-parameterization of the model as a Gaussian Mixture Model (GMM) with C components, we have

$$\mathbf{y}_t = W_y h_t + b_y = \{(\tilde{\pi}_t^c, \tilde{\mu}_t^c, \tilde{\sigma}_t^c, \tilde{\rho}_t^c)\}_{c=1}^C \tag{16}$$

where W_y and b_y are the weights and bias of the projecting linear layer, respectively. Since the outputs of the linear layer are unbounded real numbers, we normalize their values to define a valid probability distribution $\mathbf{p}_t = \{(\mu_t^c, \pi_t^c, \sigma_t^c, \rho_t^c)\}_{c=1}^C$ as follows [47]:

$$\mathbf{p}_t = \left\{ \begin{array}{l} \pi_t^c = \frac{\exp(\tilde{\pi}_t^c)}{\sum_{c'=1}^M \exp(\tilde{\pi}_t^{c'})} \\ \mu_t^c = \tilde{\mu}_t^c \\ \sigma_t^c = \exp(\tilde{\sigma}_t^c) \\ \rho_t^c = \tanh(\tilde{\rho}_t^c) \end{array} \right\}_{c=1,}^{C} \tag{17}$$

where μ_t^c and σ_t^c are means and standard deviations of the bivariate Gaussians, respectively, π_t^c are the mixing coefficients with $\sum_{c=1}^C (\pi_t^c) = 1$ and ρ_t^c identify the correlations between variables.

B. Model Training

The only trainable components of HammerDrive are the modules of ParRMDN. We train the modules individually on behavioral-aware sub-datasets $\mathcal{D}_l$, which we define as subsets of $\mathcal{D}$, i.e., $\mathcal{D}_l \subset \mathcal{D}$,$\forall\ l \in \mathcal{L}$. Analytically we have

$$\mathcal{D}_l = \left\{ \left(\mathbf{v}^i, \mathbf{x}^i, l^i \right) \right\}_{i=1}^{N_l}, \ s.t.\ l^i = l \ \forall\ i \in 0, ..., N_l, \tag{18}$$

where N_l identifies the number of data-points available in the sub-dataset corresponding to the l-th maneuver. For each $\mathcal{D}_l$, 80% of the sub-dataset is used for training and the remaining 20% is used for testing.

The training of ParRMDN modules is performed by optimizing the negative log-likelihood of the ground-truth gaze locations $\mathbf{x}_i$ for the i-th frame sequence $\mathbf{v}_i$:

$$\Gamma(\mathbf{v}_i, \mathbf{x}_i) = \sum_{t=0}^{T-1} \sum_{j=1}^{A} - \log \left(\sum_{c=1}^{C} \pi_t^c \mathcal{N}(a_{t,j}^i; \mu_t^c, \sigma_t^c, \rho_t^c) \right). \tag{19}$$

For the interested reader, more details on RMDNs and their training can be found in [47].

C. Task Awareness Integration

We have seen that ParRMDN operates as an ensemble of RMDN modules, each trained to predict the maneuver-dependent visual attention of the driver. At inference stage, all modules produce different predictions from the same video data and they are averaged according to the confidence level vector given by HAMMER (see Eq.(10)). Given the instantaneous prediction of a single RMDN module

$$\mathrm{P}(l, t) = \sum_{c=1}^{C} \pi_t^c \mathcal{N}(\mu_t^c, \sigma_t^c, \rho_t^c), \tag{20}$$

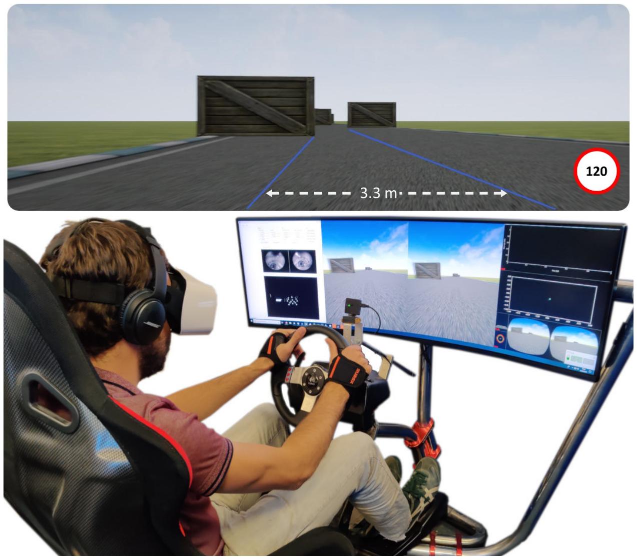


Fig. 4. Driving simulator setup. Top: simulated environment with an overlay of the three virtual lanes considered, their width and the free-flow speed. Note that none of these overlays were visible to participants during the experiment. Bottom: the participant wears a VR headset with integrated eye tracker. The screen shows the scene displayed to the participant and real-time sensor readings for monitoring purposes.

the output of HammerDrive is computed as:

$$\mathrm{P}(t) = \Big[\underline{\psi}_l(t+1) \cdot \mathrm{P}(l, t) \Big]_{l=1}^{L}. \tag{21}$$

VI. Experiments

We evaluate HammerDrive on a dataset that we collected on our custom designed driving virtual reality (VR) simulator (see Fig. 4). The experiment was designed to collect physiological signals that were unobtrusive, namely gaze locations from a VR headset, and easy-to-access telemetry data from the vehicle, namely steering angle, throttle, speed, location and scene information. This study has been approved by the Ministry of Defence Research Ethics Committee (MoDREC).

A. Participants

We recruited twenty participants (mean age 24.2, standard deviation 4.5), all experienced drivers with normal or corrected vision. Before the trial, participants were introduced to the sensors, experimental setup and a brief explanation of the task to be performed. The eye-tracking sensors were calibrated at the beginning of each trial. To avoid learning effects, each participant was allowed a demo trial to familiarize themselves with the driving simulator before performing the actual experiment.

B. Setup

We set up a realistic driver-in-the-loop simulation for the experiment (see Fig. 4). The setup comprised of a physical simulator, a VR headset with integrated eye gaze tracking, which required an infra-red camera mounted above the steering wheel and a screen to monitor the participants. The simulated driving environment was custom developed and designed using the Unreal Engine (https://www.unrealengine.com). The engine is known to provide state-of-the-art near photo-realistic rendering quality and cutting edge realistic physics, and has been successfully used in the past literature for a range of applications, including realistic driving simulators for autonomous vehicle research [52], [53].

C. Experimental Procedure

The experiment required each participant to drive for five minutes along a straight 10m-wide highway on our simulator. Multiple rectangular shaped obstacles are placed on the track, and the participants are asked to avoid these obstacles while driving at a free-flow speed of 120km/h. For the purpose of simulation, the road is discretized into three lanes with ~ 3.3m width and the obstacles are randomly placed at one of three lanes at a fixed distance between each other. The obstacles are 3.5m wide, so that they entirely block a lane in width, and they are 120m distant from each other. Therefore, given a speed of 120km/h, participants pass an obstacle approximately every 3.5 seconds. The total length of the highway is 15km, however participants were asked to drive five minutes per session, which, at a speed of 120km/h corresponds to 10km.

The experiment and the simulated scenario are designed to reduce inter-participant differences in the rate of driving maneuvers, the environment experienced and the cognitive states. We achieve this by maintaining all participants' vehicles to drive at a constant velocity, i.e., 120km/h, even when avoiding the obstacles. While there are no theoretical constraints to implementing braking and turns in HammerDrive, we restrict driving maneuvers to lane changes. This ensures that all drivers perform the same number of driving maneuvers, and that each maneuver is performed in the same amount of time, i.e., every $\sim$ 3.5 seconds. Besides ensuring that all drivers experienced similar levels of cognitive states and engagement, the chosen scenario, i.e., a highway driving with lane-changes, also ensures that the collected dataset captures a richer set of human behaviors for each of the considered maneuvers.

To limit situations where drivers can avoid multiple subsequent obstacles without performing a lane change, we employ a custom discrete distribution for obstacle lane placement. Given the number of obstacles between the current obstacle and the previous one on the i-th lane d_i, we define the next obstacle placement probability distribution on lane i as

$$p(i) = \frac{e^{d_i/\delta}}{\sum_i e^{d_i/\delta}}, \tag{22}$$

where δ identifies the distance between two adjacent obstacles. This ensures that if the i-th lane has not been blocked for the past 4 obstacles, the probability of blocking that lane is e^4 times higher than that of the most recently blocked lane. Note that obstacle placement is designed such that obstacles only cover one virtual lane, i.e., two of the three virtual lanes are empty so that drivers are always able to avoid the obstacles.

D. Dataset Collection

For each participant, we collected (see Section IV-A): 1) instantaneous two-dimensional gaze locations for both left and right eye ($\{ts, x_r, y_r, x_l, y_l\}$), 2) state information of the vehicle ($\{\tau_x, \tau_y, \theta_x, \theta_y, v\}$), 3) driver inputs ($\{w, g\}$) (all at 60 Hz) and scene observed (200×112 sized RGB images at 30Hz). The integrated eye-tracker in the VR headset provides gaze data as a set of two-dimensional coordinates, which correspond to the human gaze location observed on the left and right eye screens, as seen through the headset lenses. The coordinate system of the screens employed by the VR headset is normalized to the range $[-1, 1]$ along both x-axis and y-axis, so that its origin is $(0, 0)$, bottom-left corner is $(-1, -1)$ and top-right corner is $(1, 1)$. Each participant performed a drive of 5 minutes, while their gaze, the scene they observed and their behavioral telemetry were recorded.

TABLE I
SUMMARY OF DATASET FEATURES.

Dataset	Frames	Subj.	Annotations	Active
HammerDrive	180,000	20	GMap, Tlmy, DrInp	Yes
Deng et al. [31]	74,825	28	GMap	No
Palazzi et al. [5]	555,000	8	GMap, Tlmy	Yes
Pugeault & Bowden [54]	158,668	1	GMap, Tlmy, DrInp	Yes

For our dataset, we recorded a total of 9000 frames and 18000 samples per data stream per participant. We recap the features of our dataset in Table I and compare it with related datasets. The table indicates the total number of frames, the number of subjects, the annotations included and whether the subjects were actively driving during the experiments. Here, GMap, Tlmy and DrInp indicate gaze maps, telemetry and driver inputs, respectively.

VII. RESULTS

In this section, we present and discuss the performance of the proposed task-aware visual attention model for driving, namely HammerDrive. Results are computed using a 5-fold validation and results are averaged over 20 realizations.

We compare the proposed model against the state-of-the-art in deep learning based visual attention modeling, RMDN [26], [5]. To the best of the authors' knowledge, there is only a limited number of deep learning based video saliency models [11], [55] to date, and RMDN is still referenced as a state-of-the-art performer when evaluated on the challenging HOLLYWOOD-2 dataset, [26], [56], [55]. We first show the performance of task-aware ParRMDN modules; these results correspond to the performance we would achieve under the assumption that an oracle is providing the task currently being performed by the driver (Section VII-B). We then show the performance achieved by HAMMER in tracking the driving maneuvers as a function of the number of available data-points and its robustness to data-driven and model-based implementations (Section VII-C). Finally, we show the performance of the proposed model when both HAMMER and ParRMDN are jointly operating (Section VII-D). We evaluate the performance of the proposed scheme in terms of multiple visual attention metrics, i.e., Kullback-Leibler divergence (KL), cross-correlation (CC), similarity (Sim) and information gain (IG) [57]. We describe each of the chosen metrics in the following subsection.

A. Evaluation Metrics

The metrics are chosen as they all provide different insights on the prediction provided by the model. In order to compute these metrics, we assume both predicted locations and ground-truth gaze patterns to be two separate two-dimensional distributions of probabilities, i.e., P for model prediction and Q^{gt} for ground-truth.

KL is a general measure that evaluates the difference between two probability distributions, analytically

$$KL(\mathrm{P}, Q^{gt}) = \sum_i Q_i^{gt} \log\left(\epsilon + \frac{Q_i^{gt}}{\epsilon + \mathrm{P}_i}\right), \tag{23}$$

where $\sum_i$ identifies the sum over all the pixels and ϵ is a regularizing factor. Therefore, lower values indicate that the two densities are more similar.

The CC metric computes how correlated P and Q^{gt} are:

$$CC(\mathrm{P}, Q^{gt}) = \frac{cov\,(\mathrm{P}, Q^{gt})}{cov(\mathrm{P}) \times cov(Q^{gt})}, \tag{24}$$

where the $cov(\cdot)$ operator identifies the covariance. Clearly, higher values of CC indicate that the two densities follow a similar behavior.

The Sim metric numerically evaluates the similarity between two distributions, analytically

$$Sim(\mathrm{P}, Q^{gt}) = \sum_i \min(\mathrm{P}_i, Q_i^{gt}). \tag{25}$$

As it follows, Sim values are constrained between [0,1], where 0 corresponds to highly dissimilar distributions and 1 to perfectly identical distributions.

Finally, IG is computed as the gain introduced by the predicted distribution when compared to a systematic bias, in our case a centered prior Gaussian, and the ground-truth distributions. Analytically, it is computed as

$$IG(\mathrm{P}, Q^b) = \frac{1}{N} \sum_i M_i^{gt} \left[\log_2\left(\epsilon + \mathrm{P}_i\right) - \log_2\left(\epsilon + B_i\right)\right], \tag{26}$$

where M^{gt} is a binary map of gaze locations, and B is the distribution of the chosen bias.

B. ParRMDN Modules

Table II includes the visual attention inference performance for each of the ParRMDN modules, under the assumption that an oracle is providing the correct task currently performed. As mentioned in Section V-B, ParRMDN modules are trained on the training portion of the corresponding maneuver sub-datasets (i.e., 80% of $\mathcal{D}_l,\ \forall\ l \in \mathcal{L}$). To highlight the benefits of the proposed task-aware module training, we collect performance of each module both on sequences corresponding to the same maneuver the module was trained for and on sequences corresponding to different maneuvers. For instance, for the left lane change module, $\mathcal{D}_{\text{left}}$ identifies the performance on left lane change dataset, while sequences that belong to other maneuvers are presented as $\mathcal{D}_{\neq\text{left}} = \mathcal{D}_{\text{keep}} \cup \mathcal{D}_{\text{right}}$. Performance of each module are averaged over 25 test sequences of 25 frames each.

As shown in Table II, task-aware module performances are always higher when evaluated on test sequences from the corresponding task-aware sub-dataset, e.g., left module on $\mathcal{D}_{\text{left}}$, whereas performance decrease when the module is tested on different task sub-datasets $\mathcal{D}_{\neq\text{left}}$. Performance gaps for KL, CC, Sim show that task-aware modules are able to better predict the focus of the driver when the task being performed is its corresponding one.At the same time, large gaps on IG show that predictions on corresponding tasks are considerably more meaningful than the ones during different tasks.

TABLE II
TASK-DEPENDENT MODULE PERFORMANCE

	Mod. Left		**Mod. Keep**		**Mod. Right**		**No Task**	**RMDN**
	$\mathcal{D}_{\text{left}}$	$\mathcal{D}_{\neq\text{left}}$	$\mathcal{D}_{\text{keep}}$	$\mathcal{D}_{\neq\text{keep}}$	$\mathcal{D}_{\text{right}}$	$\mathcal{D}_{\neq\text{right}}$	$\mathcal{D}$	$\mathcal{D}$
KL ↓	$\underline{0.72}$	0.91	$\underline{0.70}$	0.78	$\underline{0.71}$	0.84	0.78	0.81
CC ↑	$\underline{0.72}$	0.63	$\underline{0.72}$	0.65	$\underline{0.73}$	0.64	0.69	0.69
Sim ↑	$\underline{0.56}$	0.48	$\underline{0.55}$	0.49	$\underline{0.56}$	0.48	0.52	0.48
IG ↑	$\underline{4.00}$	3.22	$\underline{3.16}$	2.89	$\underline{5.49}$	3.15	4.02	3.76

↑ Metric where higher values indicate better performance.
↓ Metric where lower values indicate better performance.

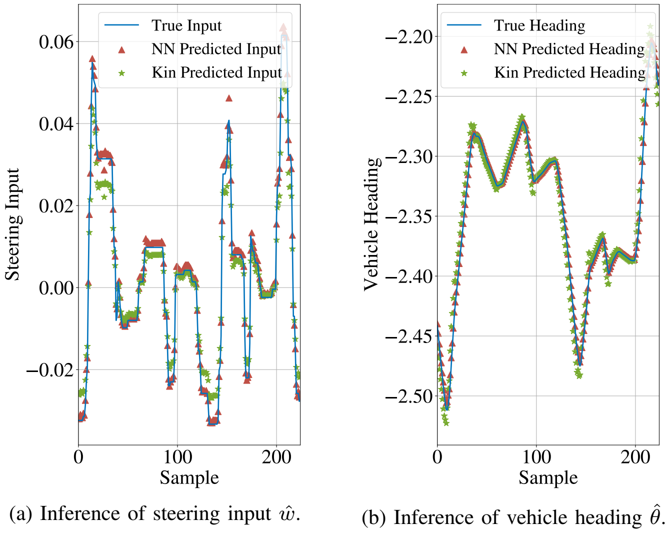
Table II also includes the performance achieved by task-unaware HammerDrive, identified by No Task, and RMDN. For task-unaware HammerDrive, we set the confidence level vector to $\underline{\psi}(t) = [0.33, 0.33, 0.33]$. This corresponds to a case where no information on the maneuvers is available, hence HammerDrive assumes all modules to have equal weights at the weighted sum stage. As we can see, task-based modules achieve higher performance than task-unaware HammerDrive, proving that the performance gap is motivated by task-awareness. On the other hand, task-unaware HammerDrive shows comparable and for some metrics better performance than RMDN. These results are not surprising, as task-unaware HammerDrive corresponds to an ensemble of RMDN modules, each trained on a specific portion of the dataset. The performance gap between task-aware modules, task-unaware HammerDrive and RMDN confirms the findings from the literature [5], [35], [41], which showed that gaze behavior of drivers is indicative of their current maneuver.

C. Robustness to Data-Driven and Model-Driven approaches

Given the flexibility of the HAMMER formulation, we can follow either a model-driven or a data-driven approach during forward-inverse model pairs design. In this section, we compare two implementations of HAMMER in terms of model accuracy and task recognition accuracy as a function of confidence window length c_w. Kin identifies the kinematics-based model-driven implementation (see Section IV), whereas NN represents the NN-based data-driven approach.

In the NN approach, we define forward and inverse models as single multilayer perceptron feed forward networks with a single hidden layer of 8 units. Forward and inverse model networks have been trained on the telemetry data of a single participant and are then tested on unseen participants' data.

Model accuracy is shown in Figs. 5a and 5b, for inverse and forward model, respectively. In Fig. 5a, we compare the true steering input $w_{(t+1)}$ with its estimated value $\hat{w}_{(t+1)}$ from both Kin and NN inverse models. In a similar manner, Fig. 5b shows a comparison between the true direction vector $[\theta_x, \theta_y]_{(t+1)}$ and its estimation from the Kin and NN forward models, i.e., $[\hat{\theta}_x, \hat{\theta}_y]_{(t+1)}$. Both figures show that a data-driven approach provides a more accurate representation of the vehicle kinematics, when compared to an analytic model-driven approach. Both models offer accurate representations of the 1-step vehicle behavior, which is a fundamental requirement



(a) Inference of steering input $\hat{w}$.

(b) Inference of vehicle heading $\hat{\theta}$.

Fig. 5. Inverse and forward module inference performance. NN identifies prediction from shallow data-driven model, while Kin stands for the prediction achieved via Kinematic model from Eq.(5) and Eq.(6) for inverse and forward module, respectively. Vehicle heading is computed as $\hat{\theta} = \arctan(\theta_y/\theta_x)$.

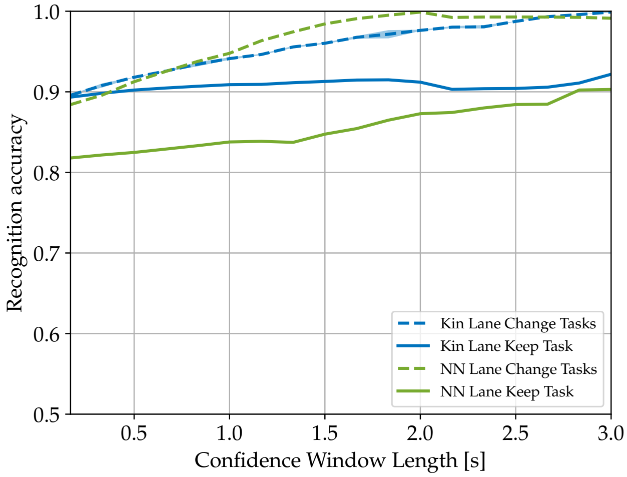


Fig. 6. HAMMER task recognition accuracy as a function of the length of the confidence window. NN identifies data-driven implementation of HAMMER, while Kin stands for a kinematic model implementation.

for HAMMER, as it ensures that both forward and inverse models predictions are reliable for maneuver tracking.

Although the performance shown in Fig. 5 are promising, we are more interested in how the two implementations affect HAMMER in its ability to correctly track maneuvers on a longer prediction horizon. In Fig. 6 we show the task recognition accuracy of NN-based and Kin-based HAMMER. The NN-based implementation shows higher task recognition accuracy during active tasks, namely lane changes on both left and right lane, while lane maintenance suffers by lower accuracy, due to jerky movements that often happen during lateral control. These movements can be mistakenly interpreted by a very accurate model, such as NN. On the other hand, these events are filtered by Kin-based HAMMER, whose accuracy is ~95% in both scenarios with $c_w = 1s$, i.e., the value considered for HammerDrive. Since each maneuver instance normally lasts $3.5s$, these results indicate that HAMMER can reliably track and recognize a maneuver $2.5s$ in advance, or before, if we assume shorter c_w.

D. HammerDrive Performance

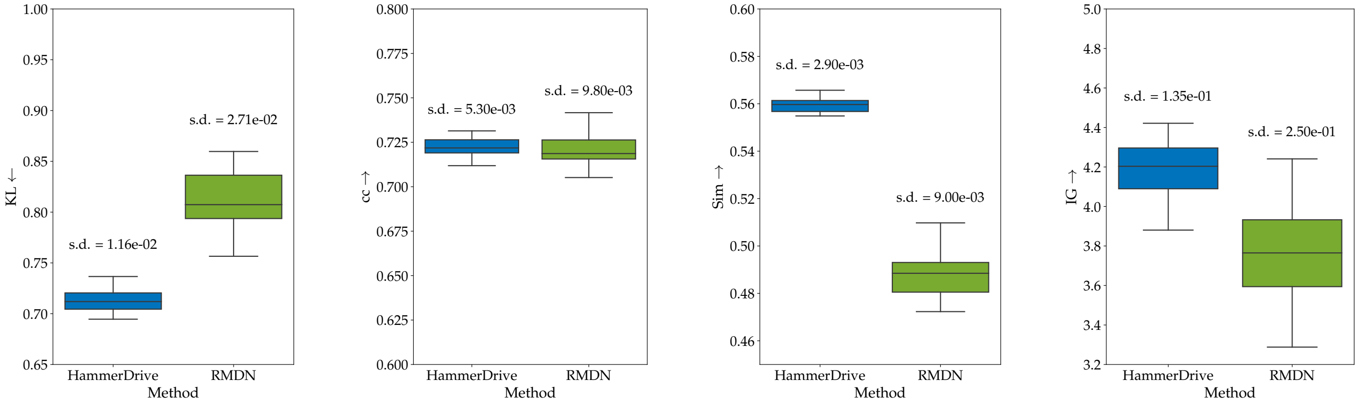
We collect HammerDrive performance in Fig. 7 for all presented metrics. Results show that task-awareness is greatly beneficial to achieve reliable visual attention predictions, with clear gains over standard RMDN. Figs. 7a and 7c highlight that HammerDrive prediction follows the actual focus of attention of the driver more closely than RMDN, as the proposed method outperforms the state-of-the-art on KL and Sim, with improvements on performance of $\sim 13\%$ in both cases. Similarly, Fig. 7d indicates that HammerDrive is able to provide more meaningful predictions as achieved IG are $\sim 12\%$ higher than the ones achieved by RMDN. Fig. 7b shows that both methods are able to capture the dynamics of focus of attention, as they both reach comparable CC values. It is important to stress that HammerDrive is very consistent in its prediction and performance in comparison with RMDN. As shown in Fig. 7, all performance metrics for HammerDrive have lower variation with smaller standard deviations.

We provide a qualitative assessment of the predicted visual attention distribution in Fig. 8, which includes predictions for all driving maneuvers considered in addition to a failure case. It is interesting to notice that, on the failure case portrayed in Fig. 8d, the driver was looking at the road in the immediate surrounding of the vehicle, without paying much attention to the obstacles ahead. If we compare this with HammerDrive's prediction, we can see that the model expected the driver to be looking at the obstacles instead. Therefore, Fig. 8d shows a case where HammerDrive could be applied in ADAS as the driver's focus is not on relevant areas of the scene.

Since the final goal of the proposed HammerDrive architecture resides in its ability to be implemented and operate in ADAS, we evaluate its computational time during inference. Given a clip of 16 frames, C3D feature extraction and Par-RMDN inference only require 6ms and ~5ms, respectively. The total inference time is 11ms, which corresponds to an inference rate of $\sim$ 90Hz. Since the scene information is captured at 30Hz, we can conclude that the proposed HammerDrive is able to operate in real-time.

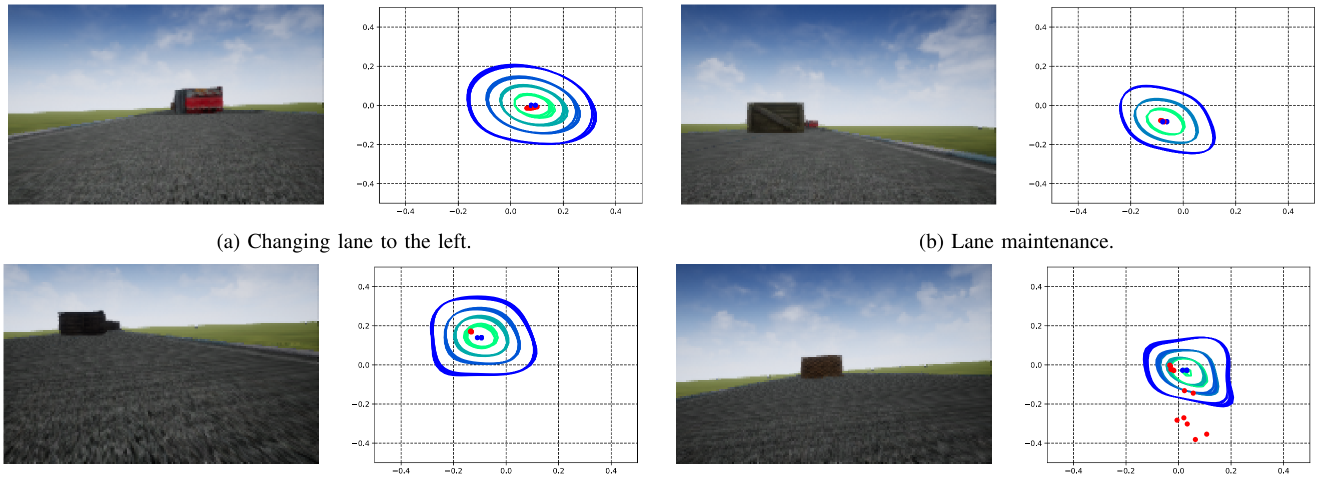
VIII. Discussion, Limitations and Open Challenges

We introduced HammerDrive, a model for driver visual attention prediction that dynamically integrates and exploits task-awareness. Our model builds on the assumption that a driver gaze pattern is driven by the goal of completing the maneuver he is currently performing. This assumption has been validated in the past literature by numerous studies and models, such as the ACT-R driver cognitive model [41], [43]. In ACT-R a two-state process is assumed for lateral control: an initial state for salience perception, where the driver shifts the visual attention to compute the steering needed to achieve a goal, and a motor state, wherein the estimated steering for the chosen goal is acted on. In a similar manner, HammerDrive uses HAMMER to compute all possible maneuvers and to select the most probable. Once the most likely maneuver has



(a) Box plot for KL metric. (b) Box plot for CC metric. (c) Box plot for Sim metric. (d) Box plot for IG metric.

Fig. 7. Performance comparison between task-aware HammerDrive and RMDN. Except for KL, where lower values correspond to predictions closer to the ground-truth, for all other parameters higher values correspond to better performance, as presented in detail in Section VII-A. For each box, the central line represents the median, the edges of the box correspond to the 25th and 75th percentiles and the whiskers identify minimum and maximum values.



(a) Changing lane to the left. (b) Lane maintenance.

(c) Changing lane to the right. (d) Failure case.

Fig. 8. Qualitative performance of HammerDrive during the three driving tasks. For each maneuver, the image on the left depicts the scene observed by the driver. The right plot shows the sequence (10 steps) of ground-truth gaze locations (red dots) and the corresponding sequence of predicted visual attention distributions $\mathrm{P}(t)$, together with their highest mode (blue dots).

been computed, this information is provided to ParRMDN which infers where drivers should be focusing their attention to safely complete such a maneuver. These assumptions lead to a reactive implementation of driver visual attention prediction.

Although the use of telemetry-based maneuver tracking for task-awareness has shown impressive results in the literature [20], numerous studies have proven that there is a strong correlation between driver gaze patterns and future maneuvers [58], [59]. In fact, a lane change is often characterized by glances to interior and side mirror and to the side window before any signs of the maneuver appear in the telemetry data. Therefore, the inclusion of gaze information as an additional input to HammerDrive opens to interesting directions for future studies, as it could potentially lead to better and more diverse maneuver-tracking. Additionally, providing the system with information on gaze history and driver future intentions could be leveraged to perform better visual focus predictions.

For our study, we collected the gaze patterns of multiple drivers in a high-speed highway driving simulator and without external distractions. In this scenario, we assumed that the driving task could be divided into a set of simple maneuvers and implemented HAMMER, a network for real-time maneuver tracking. Our results showed that HAMMER is a flexible network, allowing for both kinematic-based models and neural network-based components, and that it can reliably track lane change maneuvers. Although the application of HammerDrive to lateral control maneuvers is supported by previous studies from the literature [19], [58], it would be worth studying how HAMMER can be expanded to track a larger set of maneuvers, such as turns and accelerations. In these scenarios, drivers have complete control over the speed of the vehicle, directly impacting both the duration and the diversity of maneuvers. These considerations go beyond expanding the maneuver tracking capabilities of HAMMER. Maneuvers for longitudinal control of the vehicle, such as braking and coasting, lead to more diverse, yet equally correct, set of scan patterns. Also, while HammerDrive can predict multimodal gaze distributions, our results showed that driver gaze patterns tended to be predominantly unimodal in the proposed scenario, due to the presence of single obstacles ahead of the driver. Therefore, investigating how HammerDrive can adapt and scale to more complex scenarios represents a clear challenge for the future.

In this paper, we have addressed the problem of driver focus prediction as a task that jointly involves top-down processes, i.e., the maneuver-tracking via our HAMMER network, and bottom-up processes, i.e., the visual-attention modeling via our ParRMDN. To achieve this, our experiment was designed to collect the gaze patterns of multiple human drivers, while they experienced a similar environment and performed the same maneuvers. Under these assumptions, we have shown that HammerDrive is able to reliably predict driver gaze and that task-awareness plays a critical role for this. Despite the flexibility offered HammerDrive, driver gaze modeling still represents a very challenging and complex task, as human drivers are characterized by different gaze patterns, even when performing the same maneuver. Among these, we identify three fundamental aspects that future models should consider and are often not taken into account when modeling visual attention in driving: self-pacing, spare capacity and peripheral vision. Drivers can modulate the complexity of the driving task by self-pacing in order to meet additional task demands, without leading them into a distracted state [60]. In a similar way, while driving, humans still have additional spare capacity, as proven by [61], where drivers were able to occlude their vision while driving without impacting their safety. Understanding and modeling these concepts could have tremendous benefits, as the ability to know when and where to glance off-road and correctly anticipate hazardous situations is a key ability for safe driving [62]. Finally, it would also be interesting to evaluate and model the role of peripheral vision, and its differences with foveal vision, in driving, as studies have shown that humans can use peripheral vision to complete numerous tasks [63].

IX. Conclusions

In this paper, we addressed the problem of task-aware visual attention prediction in driving. To solve this, we proposed HammerDrive, a learnable architecture that uses easy-to-access data from the vehicle. We developed a realistic virtual-reality driving simulator and collected a dataset of multimodal data from a cohort of 20 participants. We performed extensive experiments and compared HammerDrive against a state-of-the-art deep learning model for visual attention prediction. Our results indicate that task-awareness is beneficial for visual attention prediction and that it can be leveraged using telemetry data to achieve more robust and reliable predictions.

Our study focused on the application of HammerDrive in a highway driving scenario and without external distractions. The extension of the simulated scenario with dynamic obstacles and a more complex environment represents an important direction for future works. Additionally, it would be interesting to investigate how HammerDrive is affected when considering additional factors, such as distraction and different cognitive states of the driver.

Acknowledgments

This work was supported in part by UK DSTL/EPSRC Grant EP/P008461/1, and a Royal Academy of Engineering Chair in Emerging Technologies to Yiannis Demiris.

References

[1] National Highway Traffic Safety Administration, *Traffic Safety Facts: Distracted Driving in Fatal Crashes*, 2019.

[2] Y. Liao, G. Li, S. E. Li, B. Cheng, and P. Green, "Understanding driver response patterns to mental workload increase in typical driving scenarios," *IEEE Access*, vol. 6, pp. 35 890–35 900, 2018.

[3] Y.-K. Wang, T.-P. Jung, and C.-T. Lin, "EEG-based attention tracking during distracted driving," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, no. 6, pp. 1085–1094, 2015.

[4] B. Morris, A. Doshi, and M. Trivedi, "Lane change intent prediction for driver assistance: On-road design and evaluation," in *IEEE Intell. Veh. Symp.*, 2011, pp. 895–901.

[5] A. Palazzi, D. Abati, S. Calderara, F. Solera, and R. Cucchiara, "Predicting the Driver's Focus of Attention: the DR(eye)VE Project," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1720–1733, 2018.

[6] T. Deng, K. Yang, Y. Li, and H. Yan, "Where does the driver look? top-down-based saliency detection in a traffic driving environment," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 2051–2062, 2016.

[7] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 185–207, 2013.

[8] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, 1998.

[9] C. Wloka, I. Kotseruba, and J. K. Tsotsos, "Active fixation control to predict saccade sequences," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3184–3193.

[10] N. D. B. Bruce, C. Catton, and S. Janjic, "A deeper look at saliency: Feature contrast, semantics, and beyond," in *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 516–524.

[11] A. Borji, "Saliency prediction in the deep learning era: Successes and limitations," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019, to appear.

[12] J. Coull, "fMRI studies of temporal attention: allocating attention within, or towards, time," *Cogn. Brain Res.*, vol. 21, no. 2, pp. 216–226, 2004.

[13] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Predicting human eye fixations via an LSTM-based saliency attentive model," *IEEE Trans. Image Process*, vol. 27, no. 10, pp. 5142–5154, 2018.

[14] J. M. Wolfe and T. S. Horowitz, "What attributes guide the deployment of visual attention and how do they do it?" *Nat. Rev. Neurosci.*, vol. 5, no. 6, pp. 495–501, 2004.

[15] H. Hadizadeh and I. V. Bajic, "Saliency-aware video compression," *IEEE Trans. Image Process*, vol. 23, no. 1, pp. 19–33, 2014.

[16] T. Fischer, H. Jin Chang, and Y. Demiris, "RT-GENE: Real-Time Eye Gaze Estimation in Natural Environments," in *Eur. Conf. Comput. Vis.*, 2018, pp. 339–357.

[17] H. Admoni and B. Scassellati, "Social eye gaze in human-robot interaction: A review," *J. of Human-Robot Interact.*, vol. 6, no. 1, pp. 25–63, 2017.

[18] K. Bengler, K. Dietmayer, B. Farber, M. Maurer, C. Stiller, and H. Winner, "Three decades of driver assistance systems: Review and future perspectives," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 4, pp. 6–22, 2014.

[19] J. Nilsson, J. Silvlin, M. Brannstrom, E. Coelingh, and J. Fredriksson, "If, when, and how to perform lane change maneuvers on highways," *IEEE Intell. Transp. Syst. Mag.*, vol. 8, no. 4, pp. 68–78, 2016.

[20] Y. Xing, C. Lv, H. Wang, H. Wang, Y. Ai, D. Cao, E. Velenis, and F. Wang, "Driver lane change intention inference for intelligent vehicles: Framework, survey, and challenges," *IEEE Trans. on Veh. Tech.*, vol. 68, no. 5, pp. 4377–4390, 2019.

[21] B. Fröhlich, M. Enzweiler, and U. Franke, "Will this car change the lane? Turn signal recognition in the frequency domain," in *IEEE Intell. Veh. Symp.*, 2014, pp. 37–42.

[22] A. Jain, H. S. Koppula, B. Raghavan, S. Soh, and A. Saxena, "Car that knows before you do: Anticipating maneuvers via learning temporal driving models," in *IEEE Int. Conf. Comput. Vis.*, 2015, pp. 3182–3190.

[23] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, "Learning-based approach for online lane change intention prediction," in *IEEE Intell. Veh. Symp.*, 2013, pp. 797–802.

[24] Y. Liu, P. Lasang, S. Pranata, S. Shen, and W. Zhang, "Driver pose estimation using recurrent lightweight network and virtual data augmented transfer learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3818–3831, 2019.

[25] Y. Demiris and B. Khadhouri, "Hierarchical attentive multiple models for execution and recognition of actions," *Robot. Auton. Syst.*, vol. 54, no. 5, pp. 361–369, 2006.

[26] L. Bazzani, L. Torresani, and H. Larochelle, "Recurrent mixture density network for spatiotemporal visual attention," in *Int. Conf. Learn. Repres.*, 2017.

[27] F. Vicente, Z. Huang, X. Xiong, F. De la Torre, W. Zhang, and D. Levi, "Driver gaze tracking and eyes off the road detection system," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2014–2027, 2015.

[28] T. Billah, S. M. Rahman, M. O. Ahmad, and M. Swamy, "Recognizing distractions for assistive driving by tracking body parts," *IEEE Trans. Circuits and Syst. for Video Tech.*, vol. 29, no. 4, pp. 1048–1062, 2018.

[29] Y. Xing, C. Lv, H. Wang, D. Cao, E. Velenis, and F.-Y. Wang, "Driver activity recognition for intelligent vehicles: A deep learning approach," *IEEE Trans. Veh. Tech.*, vol. 68, no. 6, pp. 5379–5390, 2019.

[30] Y. Xia, D. Zhang, J. Kim, K. Nakayama, K. Zipser, and D. Whitney, "Predicting driver attention in critical situations," in *Asian Conf. Comput. Vis.*, 2018, pp. 658–674.

[31] T. Deng, H. Yan, L. Qin, T. Ngo, and B. S. Manjunath, "How do drivers allocate their potential attention? Driving fixation prediction via convolutional neural networks," *IEEE Trans. Intell. Transp. Syst.*, 2019.

[32] T. Deng, H. Yan, and Y.-J. Li, "Learning to boost bottom-up fixation prediction in driving environments via random forest," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 9, pp. 3059–3067, 2017.

[33] A. Tawari and B. Kang, "A computational framework for driver's visual attention using a fully convolutional architecture," in *IEEE Intell. Veh. Symp.*, 2017, pp. 887–894.

[34] Y. Yao, M. Xu, Y. Wang, D. J. Crandall, and E. M. Atkins, "Unsupervised traffic accident detection in first-person videos," in *IEEE Intern. Conf. on Intell. Robots and Syst.* IEEE, 2019, pp. 273–280.

[35] M. Land and B. Tatler, *Looking and acting: Vision and eye movements in natural behaviour*. Oxford University Press, 2009.

[36] M. M. Hayhoe, "Vision and action," *Annu. Rev. Vis. Sci.*, vol. 3, pp. 389–413, 2017.

[37] M. Da Lio, F. Biral, E. Bertolazzi, M. Galvani, P. Bosetti, D. Windridge, A. Saroldi, and F. Tango, "Artificial Co-Drivers as a universal enabling technology for future intelligent vehicles and transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 244–263, 2015.

[38] M. Da Lio, A. Mazzalai, and M. Darin, "Cooperative intersection support system based on mirroring mechanisms enacted by bio-inspired layered control architecture," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 5, pp. 1415–1429, 2018.

[39] M. Da Lio, A. Mazzalai, K. Gurney, and A. Saroldi, "Biologically guided driver modeling: The stop behavior of human car drivers," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2454–2469, 2018.

[40] C. Sentouh, A. Nguyen, M. A. Benloucif, and J. Popieul, "Driver-Automation cooperation oriented approach for shared control of lane keeping assist systems," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 5, pp. 1962–1978, 2019.

[41] D. D. Salvucci, "Modeling driver behavior in a cognitive architecture," *Human Factors*, vol. 48, no. 2, pp. 362–380, 2006.

[42] D. D. Salvucci and R. Gray, "A two-point visual control model of steering," *Perception*, vol. 33, no. 10, pp. 1233–1248, 2004.

[43] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind." *Psychological review*, vol. 111, no. 4, p. 1036, 2004.

[44] P. Bosetti, M. Da Lio, and A. Saroldi, "On curve negotiation: From driver support to automation," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2082–2093, 2015.

[45] D. Shin, B. Kim, K. Yi, A. Carvalho, and F. Borrelli, "Human-centered risk assessment of an automated vehicle using vehicular wireless communication," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 2, pp. 667–681, 2019.

[46] D. Ognibene and Y. Demiris, "Towards active event recognition." in *Int. Joint Conf. on Artif. Intell.*, 2013.

[47] A. Graves, "Generating sequences with recurrent neural networks," *arXiv:1308.0850*, pp. 1–43, 2013.

[48] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *IEEE Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.

[49] M. Kok, J. Hol, and T. Schön, "Using inertial sensors for position and orientation estimation," *Found. and Trends in Signal Process.*, vol. 11, pp. 1–153, 2017.

[50] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *IEEE Intell. Veh. Symp.*, 2015, pp. 1094–1099.

[51] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[52] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conf. on Robot Learn.*, 2017.

[53] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2018, pp. 621–635.

[54] N. Pugeault and R. Bowden, "How much of driving is preattentive?" *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5424–5438, 2015.

[55] W. Wang, J. Shen, J. Xie, M. Cheng, H. Ling, and A. Borji, "Revisiting video saliency prediction in the deep learning era," *IEEE Trans. Pattern Anal. and Mach. Intell.*, 2019, to appear.

[56] S. Gorji and J. J. Clark, "Going from image to video saliency: Augmenting image salience with dynamic attentional push," in *IEEE Conf. on Comput. Vis. and Pattern Recog.*, 2018, pp. 7501–7511.

[57] M. Kummerer, T. S. Wallis, and M. Bethge, "Saliency benchmarking made easy: Separating models, maps and metrics," in *Eur. Conf. Comput. Vis.*, 2018, pp. 770–787.

[58] S. Martin, S. Vora, K. Yuen, and M. M. Trivedi, "Dynamics of driver's gaze: Explorations in behavior modeling and maneuver prediction," *IEEE Trans. Intell. Veh.*, vol. 3, no. 2, pp. 141–150, 2018.

[59] A. Doshi and M. M. Trivedi, "On the roles of eye gaze and head dynamics in predicting driver's intent to change lanes," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 453–462, 2009.

[60] K. Kircher and C. Ahlstrom, "Minimum required attention: A human-centered approach to driver inattention," *Hum. Factors*, vol. 59, no. 3, pp. 471–484, 2017.

[61] K. Kircher, T. Kujala, and C. Ahlström, "On the difference between necessary and unnecessary glances away from the forward roadway: An occlusion study on the motorway," *Hum. Factors*, vol. 62, no. 7, pp. 1117–1131, 2020.

[62] C. C. McDonald, A. H. Goodwin, A. K. Pradhan, M. R. Romoser, and A. F. Williams, "A review of hazard anticipation training programs for young drivers," *J. of Adolesc. Health*, vol. 57, no. 1, pp. S15–S23, 2015.

[63] R. Rosenholtz, "Capabilities and limitations of peripheral vision," *Annu. Rev. Vis. Sci.*, 2016.

Pierluigi Vito Amadori (S'14, M'17) received the M.Sc. degree (Hons.) in Telecommunications Engineering from the University of Rome La Sapienza, Rome, Italy, in 2013 and the Ph.D. degree in Electronic Engineering from the Department of Electrical & Electronic Engineering, University College London, London, U.K., in 2017.

He currently holds a position as a Postdoctoral Research Associate at the Personal Robotics Laboratory at Imperial College London, London, U.K. His main research interests include driver monitoring, user modeling and driving assistance systems.

Tobias Fischer (M'16) received the B.Sc. degree from Ilmenau University of Technology, Germany, in 2013, the M.Sc. degree in Artificial Intelligence from the University of Edinburgh, U.K., in 2014, and the Ph.D. degree from the Personal Robotics Lab, Imperial College London, London, U.K, in 2018.

His research interests include both computer vision and human vision, visual attention and computational cognition. He is interested in applying this knowledge to cognitive robotics.

Dr. Fischer was a recipient of the Queen Mary Award for the Best U.K. Robotics PhD Thesis in 2018 and the Eryl Cadwaladr Davies prize for the best departmental thesis in 2017-2018.

Yiannis Demiris (SM'03) received the B.Sc. (Hons.) degree in artificial intelligence and computer science and the Ph.D. degree in intelligent robotics from the Department of Artificial Intelligence, University of Edinburgh, Edinburgh, U.K., in 1994 and 1999, respectively.

He is a Professor with the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., where he is the Royal Academy of Engineering Chair in Emerging Technologies, and the Head of the Personal Robotics Laboratory. His current research interests include human-robot interaction, machine learning, user modeling, and assistive robotics. He has published more than 200 journal and peer-reviewed conference papers in the above areas.

Prof. Demiris was a recipient of the Rectors Award for Teaching Excellence in 2012 and the FoE Award for Excellence in Engineering Education in 2012. He is a Fellow of IET, BCS, and Royal Statistical Society.