sensors

MDPI

Article

Navigating an Automated Driving Vehicle via the Early Fusion of Multi-Modality

Malik Haris [1,*] and Adam Glowacz [2]

[1] School of Information Science and Technology, Xipu Campus, Southwest Jiaotong University, Chengdu 611756, China
[2] Department of Automatic Control and Robotics, Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering, AGH University of Science and Technology, Al. A. Mickiewicza 30, 30-059 Krakow, Poland; adglow@agh.edu.pl
* Correspondence: malikharis@hotmail.com

Abstract: The ability of artificial intelligence to drive toward an intended destination is a key component of an autonomous vehicle. Different paradigms are now being employed to address artificial intelligence advancement. On the one hand, modular pipelines break down the driving model into submodels, such as perception, maneuver planning and control. On the other hand, we used the end-to-end driving method to assign raw sensor data directly to vehicle control signals. The latter is less well-studied but is becoming more popular since it is easier to use. This article focuses on end-to-end autonomous driving, using RGB pictures as the primary sensor input data. The autonomous vehicle is equipped with a camera and active sensors, such as LiDAR and Radar, for safe navigation. Active sensors (e.g., LiDAR) provide more accurate depth information than passive sensors. As a result, this paper examines whether combining the RGB from the camera and active depth information from LiDAR has better results in end-to-end artificial driving than using only a single modality. This paper focuses on the early fusion of multi-modality and demonstrates how it outperforms a single modality using the CARLA simulator.

Keywords: artificial intelligent; end-to-end autonomous driving; safely navigation; conditional imitation learning (CIL); conditional early fusion (CEF); situation understanding; object detection; CARLA

Citation: Haris, M.; Glowacz, A. Navigating an Automated Driving Vehicle via the Early Fusion of Multi-Modality. *Sensors* **2022**, *22*, 1425. https://doi.org/10.3390/s22041425

Academic Editor: Mehmet Rasit Yuce

Received: 28 December 2021
Accepted: 11 February 2022
Published: 13 February 2022

1. Introduction

Autonomous vehicles are essential to the future of the transportation industry. As a result, developing deep learning for autonomous vehicles is essential. No one should deny that recently, deep learning and computer vision have significantly impacted the automotive industry. Deep learning is used extensively in autonomous driving and augmented reality. Only a few of the complex and specialized functions needed for autonomous driving are automatic, for example, lane-keeping assistance, emergency braking (AEB) [1], active cruise control [2], forward collision warning (FCW) [3] and crash avoidance [4]. FCW and AEB are two of the first accident-avoidance features to be tested. The vehicle can only warn the driver of an impediment in front of them in FCW mode, and the driver must then determine whether to act. In AEB, however, as the vehicle approaches the front object, the vehicle begins to act by braking. As a result of integrating intelligent sensors such as Camera, LiDAR and Radar, ego-vehicles can make these decisions; however, low-quality sensors lead to many collisions and congestion. Modular pipelines (MPs) methods and end-to-end (E2E) learning methods are two common approaches for enabling self-driving to overcome these accident and alert the driver if some obstacle is found on the driving lane line.

MPs are used in the majority of autonomous vehicles [5,6]. Perception, route planning and control are three subproblems of the autonomous driving issue, that MPs divide into smaller, simpler subproblems. The method often depends on various sensors perfectly

depicting the surrounding environment. Clear perception, tracking, mapping/localization, planning and control are the foundations of the MPs methods. This representation is then used to make a driving decision. While MPs are relatively easy to understand due to their modularity, they rely on complex intermediate representations that must be manually chosen (e.g., optical flow) and are often difficult to estimate with sufficient accuracy. Therefore, these methods may not be the best option for solving the sensorimotor control task. MPs also need significant quantities of labeled data, which may be difficult to come by, such as pixel-wise semantic segmentation [7,8] for neural network training, or high-definition maps for localization. Traffic sign recognition [9], obstacle detection [10–13], lane line recognition [14–17], monocular depth estimation [18–20], SLAM and positions recognition [21–23], and other sub-tasks, also become challenges for MPs methods.

In E2E approaches, deep neural networks are trained to directly produce the control outputs from raw sensor inputs. Imitation learning is the most popular approach [24]; it is a supervised learning approach based on human demonstrations. Using imitation learning to drive end-to-end has recently resurfaced as a topic of interest among academics. Imitation learning, like human learning, utilizes an image as an input and then predicts steering wheel angle, acceleration and deceleration values as outputs. End-to-end driving refers to the process of a model learning to drive from an expert demonstration, and it has been effectively used in lane-following [24,25] and obstacle avoidance off-road [26]. The article [27] argued that utilizing just an image as input is insufficient for determining whether the vehicle should turn left or right, or continue straight while approaching an intersection, and suggested a technique based on conditional imitation learning (CIL) and navigation commands to address this problem.

It is helpful to consider what factors an end-to-end driving system should include. We look at the issue from a variety of angles. The end-to-end driving system must start with a navigation command before following the selected route [27]. Besides this, we know it is helpful for human drivers to figure out what things are in front of the ego-vehicle. Xu et al. [28] have shown that combining a fully convolutional network (FCN) [29] with privilege learning may improve the driving model performance. Furthermore, human drivers pay close attention to the road. When a human driver sees that the traffic light has turned red, he will come to a stop, even if there are no other motorists or pedestrians in front of the ego-vehicle. Xu et al. [28] also show how segmentation information may help the model focus on the correct items. However, when the vehicle approaches an intersection, segmentation information may not be sufficient to allow the car to concentrate on valid objects, since the model cannot determine where to look and move at this moment without navigation commands. Intuitively, a branching design with a navigation command, in which various model portions handle distinct navigation circumstances, may be beneficial. In addition, human drivers can drive a vehicle safely since they can make out what class an object belongs to and its distance from the vehicle. We utilize an input RGB image and active depth information (D) from LiDAR to improve end-to-end artificial driving, rather than using only a single modality. Moreover, self-driving cars should be able to follow other vehicles. Using sequential images as input, instead of a single image, seems to have this impact. Long short-term memory (LSTM) [30] has been shown to improve driving performance in previous studies [28,31,32]. Finally, additional data, such as speed, are required in an end-to-end driving system, particularly when the speed restriction is considered.

Compared to the task in [27], our task parameters in this paper are slightly different. The traffic signal, other vehicles and pedestrians are all considered, but the speed limit is not. The model outputs include the vehicle steering angle and throttle, but the brake is not included. The brake is treated as a binary classification issue, with the vehicle either stopping or allowing driving at a steady speed. Our contributions to this study are to see whether utilizing multi-modality sensor data instead of depending on a single modality may improve the ability of an E2E driving model to assess and predict driving behavior. Color images (RGB) and depth (D) are considered single modalities; however, RGB-D is

considered multi-modal. This work is built on the conditional imitation learning (CIL) [27] CNN architecture, which can take high-level commands. We explore RGB-D via early fusion of the single model, such as RGB image and depth (D). In addition, we use the CARLA simulator [33], as do many recent studies on E2E driving [34–39]. The remainder of the research paper is structured as follows:

- Section 2—Reviews the related work carried out and developed in the past few years.
- Section 3—Presents the CEF architecture for our proposed model using the CIL.
- Sections 4 and 5—Summarize the experimental settings and the obtained results.
- Section 6—Discusses our conclusions and future work.

2. Related Work

Most research initiatives use MPs, which are the most common method of autonomous driving [40,41]. The perception stack must identify all elements of the traffic scene that are likely to be important for the driving choice, to construct the environmental model. Object detection [10,42], image segmentation [43,44] and motion estimation [45,46] are often trained and solved independently [47], with deep neural networks being used more recently for these tasks. This data may be compiled into an environment model [48,49], and a planning module creates an obstacle-free path for the control module to follow [50].

ALVINN [24] was the first imitation learning application of autonomous driving, predicting the steering angle from data from active and passive sensors. Deep learning advances have reignited interest in conditional imitation learning for autonomous driving [51]. ALVINN utilized a conditional order to display an E2E network for lanes following vacant highways, which monitored the steering angle from a single camera [52]. It learned longitudinal and transverse control through CIL, using a remote-control vehicle to execute route commands in a static environment [53]. Using numerous cameras and a 2D map for localization, it learned to navigate a road network. However, it relied on localization and route map generation and used drastically cropped images.

Pomerleau [20] and LeCun et al. [22] utilized ground vehicles and qualified deep networks to predict driver behavior using camera input. Bojarski et al. [25] show excellent results on real-world tasks, including highway following and flat-course driving. These studies focused on reactive tasks, like obstacle avoidance and lane following. However, the throttle and brake are not controlled, and lane and road changes are not considered, neither are go straight or slow down/stop maneuvers. In contrast, we propose a command-conditional formulation for more dynamic urban driving applications. Another difference is that the model is taught to control the steering angle as well as the throttle and brake, allowing it to drive itself. The decomposition of complex functions into simpler subtasks has been investigated from several perspectives.

Multiple layers of temporally extended subpolicies have been attempted using hierarchical methods to reinforce learning [54]. This kind of hierarchical breakdown is well exemplified by the choices framework [55]. This setting teaches fundamental motor abilities that may be used for various tasks [56]. For raw sensory input, hierarchical techniques were coupled with deep learning and utilized [57]. The main goal of these works was to simply profit from previous experience, and allow the hierarchical structure to uncover itself independently. This is a complicated and frequent issue, especially regarding sensorimotor skills. In addition, movement primitives have been used as building blocks in robotics to create complex motor skills [58,59]. A parameterized dynamical structure describes a simple motion, such as a strike or a throw, using movement primitives. On the contrary, the strategy we considered has more parameters and can solve more complex sensorimotor tasks, which combine perception and control. We focus on finding the next intersection or traffic light to reduce a vehicle's speed and then making a left turn while avoiding dynamic obstacles, pedestrians, or other vehicles.

On the other hand, we use the CIL model to focus on early fusion and offer extra information about the expert objectives throughout the presentation. This article highlights the difficulty of learning and proposes a human policy that may be controlled. Hierarchical

methods are like the concept of researching multi-functional and parameterized controllers. Parameterized targets are employed in the area of robotics [60–62]. A generalized reinforcement learning system with parameterized values transferred between states and goals, was proposed by Schaul et al. [63]. Koltun et al. [64] investigated families of parameterized objectives in the context of navigation in a 3D environment. Javdani et al. [65] looked at a scenario in which a robot assists a human and modifies his actions based on his evaluation. Although our study uses the same technique for training a conditional controller, the model architecture and application domain are different. Our method is on the E2E spectrum, but the controller comes with commands determining the driver intention and sensory input. That eliminates any uncertainty in mapping the perceptuomotor and provides a medium for communication that can guide the autonomous vehicle like a chauffeur.

Due to safety issues, driving simulators are primarily used in training and testing tasks. Open-source applications, such as TORCS [66–68] and Grand Theft Auto V (GTA V) [69,70], are popular driving simulators used for research. However, TORCS is not photorealistic or dynamic enough, since it lacks the essential elements of the scene, such as cross-roads, oncoming traffic, pedestrians, etc. GTA V is photorealistic, yet it is a closed source with limited customization and control over the environment. We utilize the recently released open-source simulator CARLA [33], which provides better customization of realism and flexibility, addressing some of the earlier simulators' problems.

3. Methodology

We first outline the fundamental conditional imitation learning (CIL) network architecture, and then demonstrate how we integrate it with network information to exploit multi-modal perception data for early fusion.

3.1. Basic Conditional Imitation Learning Network Architecture

Consider each observation $\boldsymbol{o} = \langle \boldsymbol{i}; \boldsymbol{m} \rangle$ as containing an image $\boldsymbol{i}$ and a low-dimensional vector $\boldsymbol{m}$ which we refer to as measurements, following Dosovitskiy and Koltun [64]. A deep network represents controller $\boldsymbol{F}$. The network receives the image $\boldsymbol{i}$, the measurements $\boldsymbol{m}$, the command $\boldsymbol{c}$, and outputs the action $\boldsymbol{a}$. There are many ways to specify action spaces, such as discrete, continuous, or hybrid. Our driving experiments use 2D action spaces: steering angle and acceleration. In this case, the acceleration is negative, which represents braking or driving backward. The command c represents a category variable using a one-hot vector.

Figure 1 illustrates a basic conditional imitation learning architecture. The network takes the image, the measurements and the command as inputs. Each of these inputs is processed by its module: an image module $\boldsymbol{I}(\boldsymbol{i})$, a measurement module $\boldsymbol{M}(\boldsymbol{m})$ and a command module $\boldsymbol{C}(\boldsymbol{c})$. Convolutional networks are used for the image module, and fully connected networks are used for the measurement module. The command module assumes a discrete set of $\boldsymbol{C} = \{\boldsymbol{c}^0, \ldots, \boldsymbol{c}^K\}$ (including a default command $\boldsymbol{c}^0$ corresponding to no specific command) and introduces a specialist branch $\boldsymbol{A}^{\boldsymbol{i}}$ for each of the commands $\boldsymbol{c}^{\boldsymbol{i}}$. The command $\boldsymbol{c}$ is a switch that selects which branch will be used at any given time. The network output is specified in Equation (1):

$$\boldsymbol{F}\left(\boldsymbol{i}, \boldsymbol{m}, \boldsymbol{c}^{\boldsymbol{i}}\right) = \boldsymbol{A}^{\boldsymbol{i}}(\boldsymbol{J}(\boldsymbol{i}, \boldsymbol{m})) \tag{1}$$

This type of architecture is known as branched. It is required that branches $\boldsymbol{A}^{\boldsymbol{i}}$ learn subpolicies that correspond to different commands. In a driving scenario, one module might focus on lane following, another on the right turns, and another on left turns. All modules are linked by their perception stream.

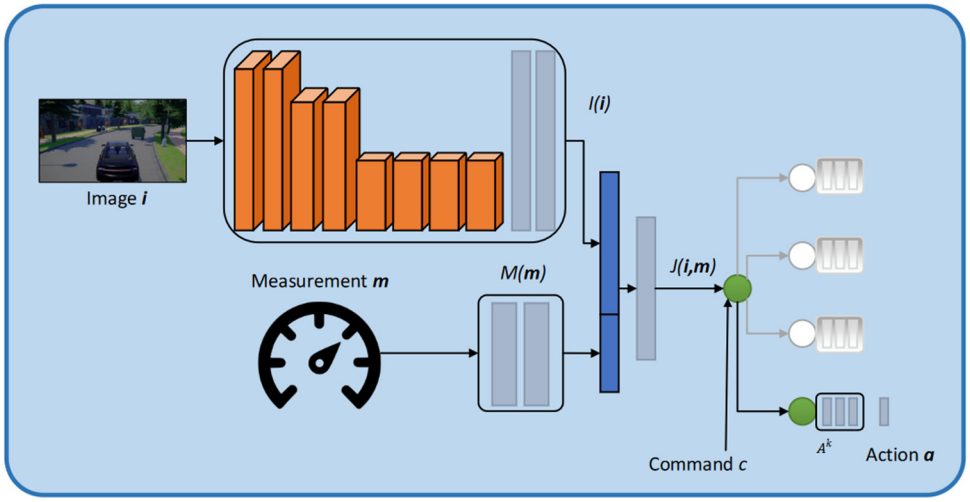


Figure 1. Fundamental conditional imitation learning (CIL) Architecture.

3.2. Early Fusion Multi-Model Network Architecture

Our proposed architecture follows the conditional imitation learning model suggested in [27]. Figure 2 depicts how we fuse the input RGB image and active depth (***D***) information from the LiDAR in the early fusion method. $\boldsymbol{I}_{i,t}$ consists of an RGB image of 200 × 88 pixels and 8 bits at each color channel, and the active depth (***D***) information from the LiDAR, $\boldsymbol{s}_{i,t}$, signifies the current vehicle speed. The command $\boldsymbol{c}_{i,t}$, and the output action $\boldsymbol{a}_{i,t}$, defines three real-value signals that determine the next maneuver, such as steering angle, throttle and brake. The control command $\boldsymbol{c}_{i,t}$ is introduced to handle complex scenarios, especially intersections, i.e., turn left, turn right, go straight, continue at the next intersection [27]. There are three continuous actions included in the action $\boldsymbol{a}_{i,t}$, namely, the steering angle for the driving wheel $\left(\boldsymbol{a}_{i,t}^{str}\right)$, the throttle setting $\left(\boldsymbol{a}_{i,t}^{acc}\right)$, and the braking action $\left(\boldsymbol{a}_{i,t}^{brk}\right)$. In order to clone human drivers' driving behavior, we can learn a deterministic policy network $\boldsymbol{F}$ via conditional imitation learning. A set of four policy branches is specifically learned to encode the hidden knowledge for each case, which is then selected for use in action prediction. Controllable imitation learning aims to determine the parameters $\boldsymbol{\theta}$ when the loss is optimal, and is defined as in Equation (2):

$$\underset{\boldsymbol{\theta}}{\boldsymbol{min}} \sum_{i}^{N}\sum_{t}^{Ti} \boldsymbol{L}(\boldsymbol{F}(\boldsymbol{I}_{i,t}, \boldsymbol{c}_{i,t}, \boldsymbol{s}_{i,t}), \boldsymbol{a}_{i,t}) \tag{2}$$

The loss function L is defined as the absolute error between the three predicted actions $\hat{\boldsymbol{a}}_{i,t}$ and the ground truth $\boldsymbol{a}_{i,t}$ with the same command:

$$\boldsymbol{L}(\hat{\boldsymbol{a}}_{i,t}, \boldsymbol{a}_{i,t}) = \left|\hat{\boldsymbol{a}}_{i,t}^{str} - \boldsymbol{a}_{i,t}^{str}\right|^2 + \left|\hat{\boldsymbol{a}}_{i,t}^{acc} - \boldsymbol{a}_{i,t}^{acc}\right|^2 + \left|\hat{\boldsymbol{a}}_{i,t}^{brk} - \boldsymbol{a}_{i,t}^{brk}\right|^2 \tag{3}$$

Figure 2 depicts the network structure diagram, while Table 1 lists the specific parameters of the network structure. The network is composed of three parts. The first part consists of the eight convolutional layers, and two fully connected layers make up the feature extraction from the ***RGB-D***. In the convolutional layer, the kernel size is five in the first layer and three in the following layers. The first, third and fifth convolutional layers have a two-step stride. In the first layer, the number of channels is 32, increasing to 256 at the eighth layer in a convolutional neural network. The fully connected layers contain 512 units in each layer; this convolutional neural network layer is elaborated in Table 1. In the second part, the speed input is processed. It consists of two layers that are fully connected. The third part consists of four identical structures, each with three fully connected layers, in which the Conv2D layer contains a convolutional layer and a dropout

layer, while the fully connected layer does not contain a dropout layer. The dropout layer randomly sets input units to zero with a frequency of rate at each step during training time, which helps prevent overfitting. A good value for dropout in a hidden layer is between 0.5 and 0.8. Here, the role of the hidden layers is to identify features from the input data and use these to correlate between a given input and the correct output. The neural networks have two main hyperparameters that control the architecture or topology of the network: the number of layers, and the number of nodes in each hidden layer. Thus, this proposed network will converge faster because it has fewer parameters to train.

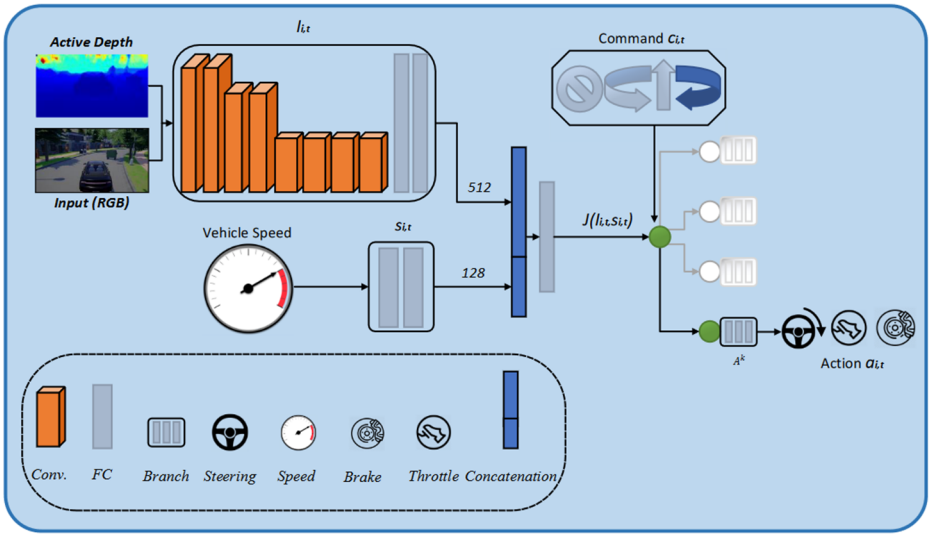


Figure 2. Early Fusion of Multi-modal Structure Diagram.

Table 1. Network Details.

Layer	Output Shape	Layer	Output Shape
Input Image	88 × 200 × 3	Input Speed	1
Conv2d_1	42 × 98 × 32		
Conv2d_2	40 × 96 × 32		
Conv2d_3	19 × 47 × 64		
Conv2d_4	17 × 45 × 64	FC	128
Conv2d_5	8 × 22 × 128		
Conv2d_6	6 × 20 × 128		
Conv2d_7	4 × 18 × 256		
Conv2d_8	2 × 16 × 256		
Flatten	8192	FC	128
FC	512		
FC	256		
FC		256	
FC	256		
FC	256	Total 4 branches for all 4 commands	
Output	1		

3.3. A Pipeline of the Early Fusion of the Multi-Model Network

The whole pipeline is visually depicted in Figure 3. Our proposed algorithm follows the conditional imitation learning model suggested in [26]. Input $\boldsymbol{I}_{i,t}$ consists of the fusion of the RGB image and the active depth (D) information from the LiDAR, $\boldsymbol{s}_{i,t}$ signifies the current vehicle speed and the command $\boldsymbol{c}_{i,t}$. In contrast to a single input mode, these multiple inputs are used for better end-to-end autonomous safety driving. In Figure 3, we can see that the convolution neural network block discussed in Section 3.2 is used to fuse RGB images with active depth (D) information from the LiDAR. Its help us to extract the most important information and associate that information with the measurement value,

such as vehicle speed, $s_{i,t}$, by concatenating. This concatenating information is inserted into the decision part and the control command values. The control command $c_{i,t}$ helps in the complex scenarios, especially intersections, i.e., follow the lane, drive straight, turn left or turn right at the next intersection. The output action $a_{i,t}$ contains three continuous actions: the steering angle for the driving wheel $\left(a_{i,t}^{str}\right)$, throttle setting $\left(a_{i,t}^{acc}\right)$ and braking action $\left(a_{i,t}^{brk}\right)$.

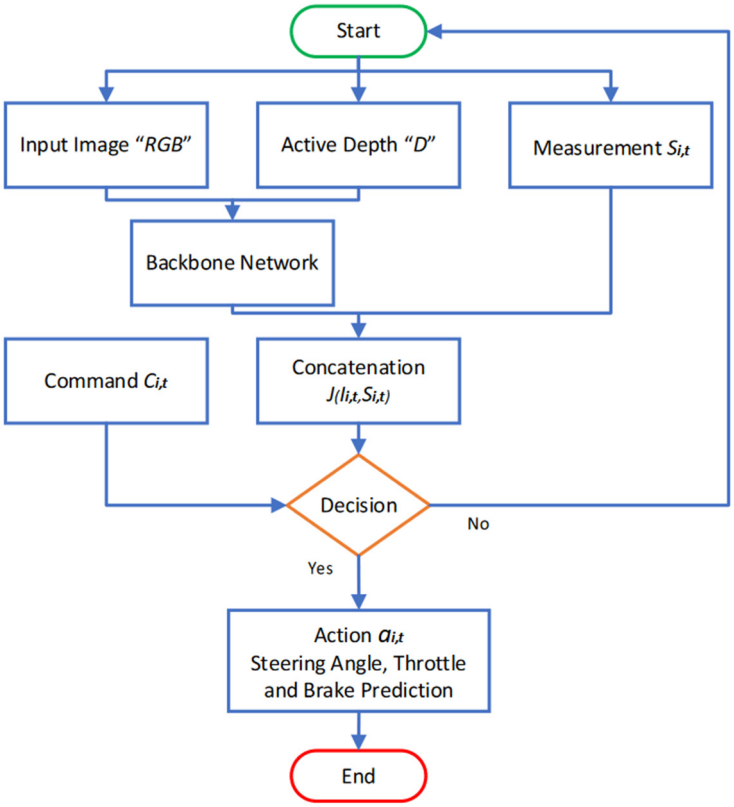


Figure 3. A Pipeline of the Early Fusion of the Multi-Model Network.

4. Experiments

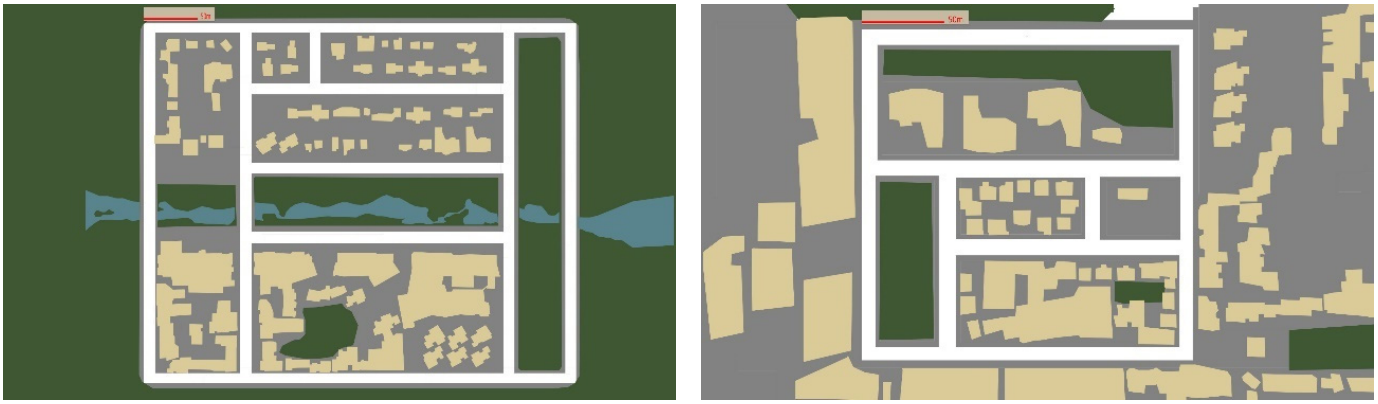
The training, validation and testing of the model proposed in this paper were performed using the TensorFlow [71] framework and cuDNN [72] kernel. The hardware equipment included a workstation with Intel@CoreTM i7-6800K CPU@3.40GHz and a GTX 1080Ti graphic card.

4.1. CARLA Simulator

Our end-to-end model was trained and tested in an open-source simulation environment for ease of use and safety. The simulation environment was chosen mainly to save data collection and labeling time. CARLA [33] is a state-of-the-art vehicle controller simulator that lets users develop their own vehicle controllers with RGB and depth cameras, and LiDAR sensors. The data about the car, such as speed, steering angle, throttle positions and brake positions, are available in simulations and the data about the environment include lane lines and traffic signs. CARLA offers more information about urban towns with different layouts. Other simulators, such as Udacity and TORCS [73], are not designed for urban area driving. Intersections, lane rules and other complexities are lacking, such as differentiating between urban driving and highway driving.

It is important to collect appropriate data to achieve imitation learning. The simulation environment is based on CARLA, with Logitech G29 Driving Force Racing Wheel for driver input, and FFB Checker for force feedback. As shown in Figure 4, CARLA provides maps

and sample views of Town 1, used for training and Town 2, used exclusively for testing. Town 1 has 2.9 km of road and 11 intersections, while Town 2 has 1.4 km of road and eight intersections. It provides a professionally designed environment with buildings, vegetation and traffic signs, as well as vehicular and pedestrian traffic. Before data collection, we redefined the scope of the steering angle. The Logitech G29 Driving Force Racing Wheel supports a rotation angle of [−900,900], the same as a real car, but in the CARLA simulator, the steering wheel angle is set to −1 or 1.



(**a**) (**b**)

Figure 4. Aerial viewpoints of (**a**) Town 1 and (**b**) Town 2.

We gathered the driving behavior data of a human driver under Left, Right, Follow, and Straight instructions (in the acquisition, the front car was a Lincoln MKZ 2017, made available by CARLA). During this time, the vehicle's speed at the time, the RGB camera frame data from the front-view camera, and the active depth information from the LiDAR, were also collected. We set the camera at the same position and parameters (such as FoV, resolution and type) as [27]. In all commands except the Follow command, autonomous vehicles had to avoid obstacles. Therefore, we collected data to reduce the risk of collision with vehicles and pedestrians in front of Right, Left and Straight.

4.2. Data Collection

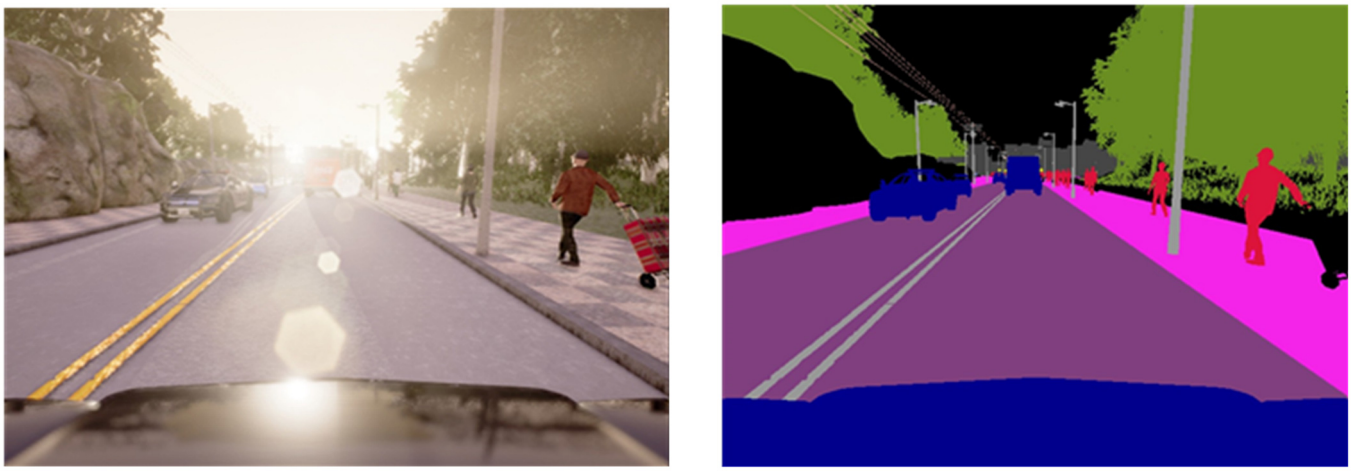
The collected data set is the same as [74], since the driving time was 25 h in Town 1, by balancing different weather environments. In short, this dataset was generated by a hard-coded self-pilot with full access to all CARLA-sensitive driving data. When traveling on a straight road, the autopilot maintained a steady speed of 31.8 km per hour, slowing down or stopping at the next intersection before turning. The raw data that we processed and synchronized at 10 fps includes the following input details:

1. The center front camera and the two side cameras at 30° left and right. The only camera utilized for autonomous driving is the one in the center. In a self-driving vehicle, the images from the side cameras are solely used during training to mimic the driving error recovery graph [25]. The collection includes 2.5 million RGB pictures with a resolution of 800 × 600 pixels and relevant ground truth. The RGB input image is cropped to eliminate the sky and extremely near region, then scaled to provide a channel resolution of 200 × 88 pixels in our early fusion model. Data augmentation is essential for good generalization in our initial experiments. During network training, we conduct augmentation online. We add a random subset of transformations of random sampling magnitudes to each image to be presented to the network. Changes in brightness, contrast, lighting and Gaussian noise are all part of the transformation. Gaussian blur, salt and pepper noise, and area dropout are some of the effects available. Geometric improvements, such as translation and rotation, are not performed, since the control command is not invariant to these transformations.

2. We develop an upper bound driver using perfect semantic segmentation in this work. In order to develop this upper bound, the twelve semantic classes of CARLA are mapped to five, which we consider sufficient. We keep the original road surfaces, vehicles and pedestrians, while lane markings and sidewalks are given the status of 'lane limits' (Towns 1 and 2 only have roads with one go and one return lane, separated by continuous double lines), and the remaining seven classes are given the status of 'other'.
3. Premebida et al. [75] obtained depth data from the LiDAR point cloud, and we think RGB images include dense depth data. The CARLA depth of ground truth comes straight from the Z-buffer used for displaying the simulation, and it is extremely accurate. The depth value is 24-bit encoded and spans from 0 to 1000 m, implying a depth precision of around 1/20 mm. An active sensor range coverage and depth accuracy significantly exceed a passive sensor. The use of depth data has been post-processed to make it more realistic. The Velodyne information from the KITTI dataset [76] provides a realistic sensor reference. First, we reduce the depth value only to examine pixels inside the 1100-m range, i.e., pixels with outside values in the depth image. On the other hand, this range does not contain any depth information. Second, we re-quantify the depth value within 4cm of the original. Third, we fix the pixels that have no data. Finally, we apply a median filter to prevent establishing precise depth boundaries between objects. New depth images are utilized during training and testing. Figure 5 shows an example of a CARLA depth image and the equivalent post-processed version.

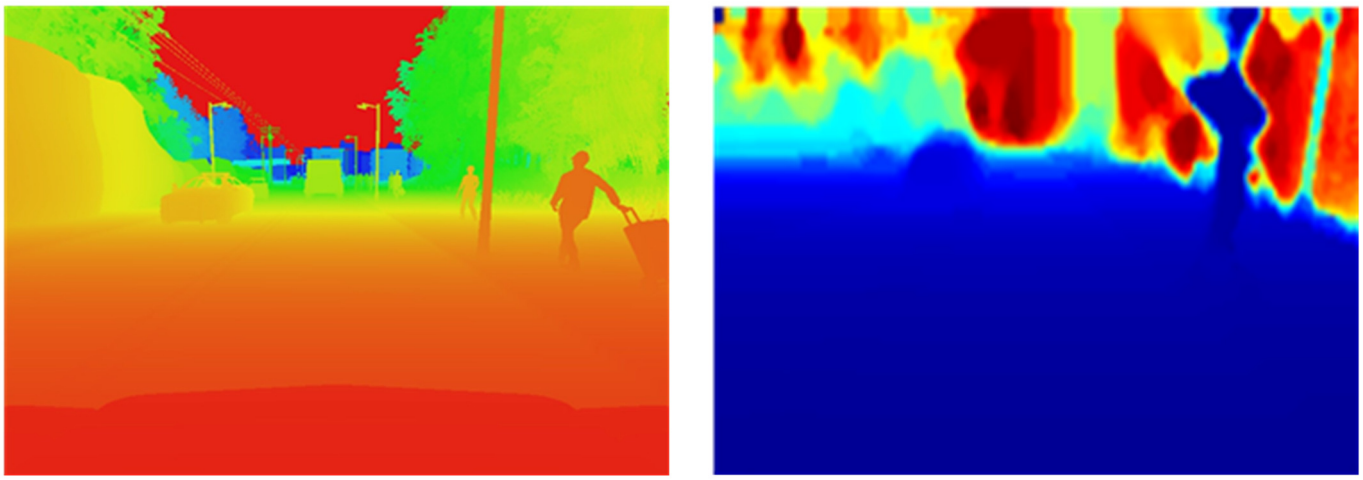
4.3. Training

Given the autopilot ground truth values, the networks are trained to minimize the mean-squared error of the steering angle, throttle and brake values. We follow the multi-tasking training approach where we have multi-tasks for four corresponding network heads for left-branch, right-branch, straight-branch, and slow down for the speed/stop-branch. According to the selected command, the loss is minimized for the current active head when one head is activated. The network is trained for 200 epochs with a batch size of 16. For optimization, the ADAM optimizer [77] is used where $\beta 1 = 0.9$ and $\beta 2 = 0.99$, and there is an initial learning rate of $\alpha = 0.0002$.



(**a**) (**b**)

Figure 5. *Cont.*



(**c**) (**d**)

Figure 5. Images utilized in our proposed model. (**a**) Input RGB image, (**b**) Semantic Segmentation Ground Truth, (**c**) CARLA depth ground truth, (**d**) Post-processed image of an active depth.

5. Experiment Results

5.1. Success Rate Comparison with Previous Methods

We compared our proposed model to previous single-modality methods discussed in the related work section, to assess its superiority further. This comparison is entirely based on the CARLA benchmark results because not all relevant papers offer comprehensive training techniques or training datasets. The benchmark test included six different driving activities with increasing complexity, including driving in a straight line, driving through a single turn, navigating through the town, taking several turns, and navigating through the town with random obstacles, such as vehicles, obstacles and pedestrians, among others. The agent starts at a predetermined place in town and completes a task within a certain amount of time. The time restriction is equivalent to the time it takes to travel 31.8 km per hour on the best route to the destination. We compared our conditional early fusion (CEF) method to the listed models in Table 2. These results are reproduced from the following manuscript listed in Table 2.

Table 2. Result Reproduction Details.

Model	Abbreviation	Manuscript
Modular Perception	MP	[33]
Reinforcement Learning	RL	[33]
Controllable Imitative Reinforcement Learning	CIRL	[35]
Multi-Task Learning	MT	[36]
Conditional Affordance Learning	CAL	[37]

The comparison results are shown in Figure 6, with a reference line for a clear values analysis. In the face of a complicated traffic situation, the early fusion of the RGB image and the active depth method had a better success rate on the CARLA benchmark. On the other hand, this early-fusion technique may be used in combination with the CAL and CIRL procedures, which are highly compatible. This comparison to prior studies backs up our assertion that multi-modality may aid E2E driving. As shown by Figure 6, our proposed model performed better than nearly all the tasks, except for the training town task under changing weather conditions, where modular perception performed slightly better. In contrast, we can also see that the results of reinforcement learning can vary greatly. Reinforcement learning was trained using more data; 12 days of driving, compared with 14 h for imitation learning. A second explanation is that urban driving is more difficult than

most tasks, especially asynchronous reinforcement learning. Moreover, a driving scenario is more complex than reinforcement learning maze navigation, for instance, because the agent needs to deal with vehicle dynamics and visual perception in a cluttered dynamic environment.

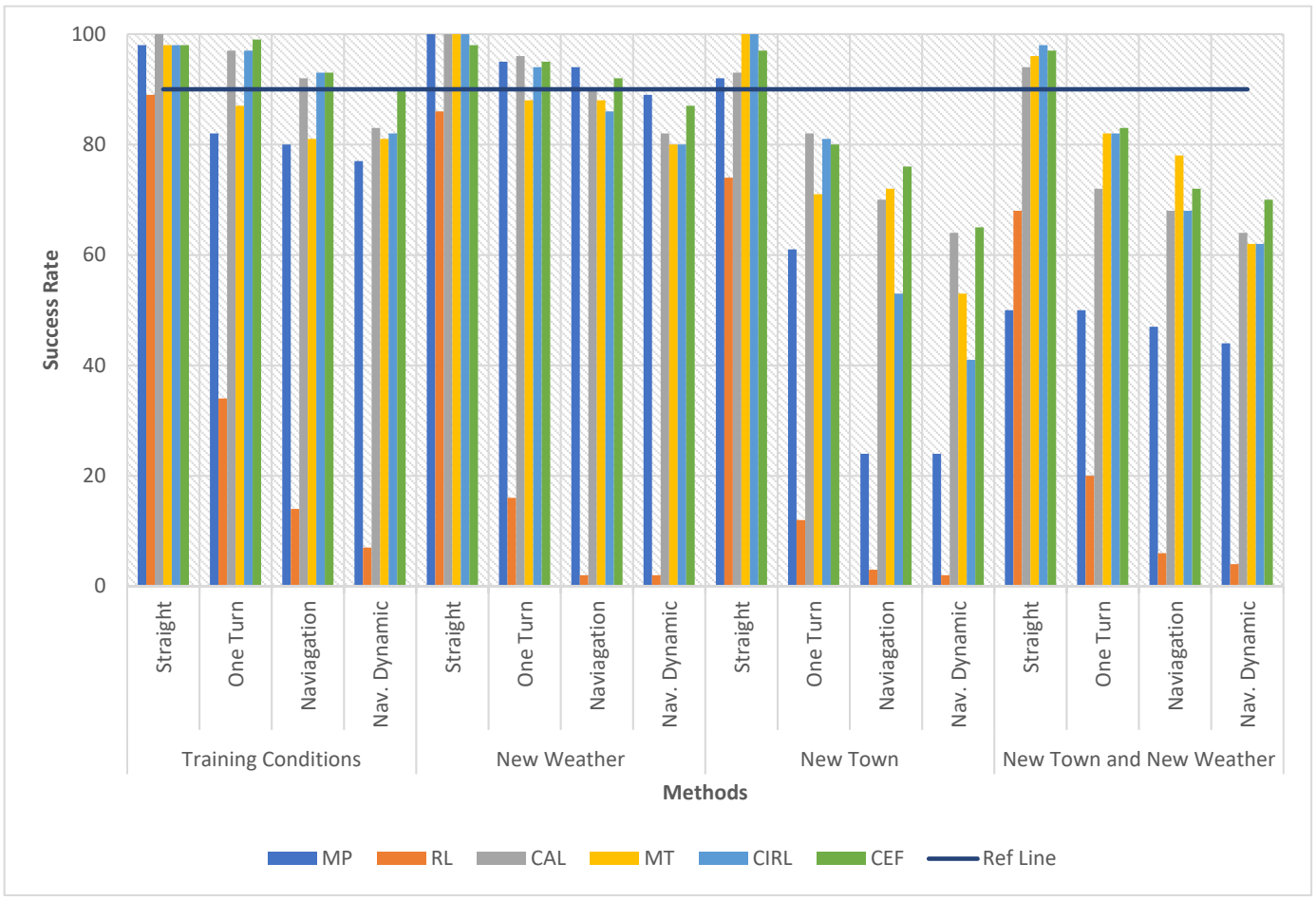


Figure 6. Success Rate Comparison with Previous Methods.

According to the benchmark, towns and weather conditions are divided into four models, under each of which the four driving tasks have to be tested in 25 episodes. These are the four models relating to towns and weather in the benchmark:

- Training conditions: driving (i.e., running the episodes) in the same conditions as the training set (Town 1, four weather conditions).
- New Town: driving under the four weather conditions of the training set but in Town 2.
- New Weather: driving in Town 1 but not seen at training time under the two weather conditions.
- New Town and Weather: driving in conditions not included in the training set (Town 2, two weather conditions).

5.2. Performance of Single & Multi-Models

The suggested model performance against the original CARLA benchmark is shown in Table 3. We incorporated a model trained in perfect semantic segmentation (SS) based on the five classes examined here for autonomous driving, as illustrated in Figure 5. Therefore, we think of this model as an upper limit. Its average production is ≥96%, and it has reached 100% on numerous occasions. Table 2 shows that our suggested multi-model may drive

correctly in CARLA if we give appropriate input. Indeed, we can observe that active depth provided complete information for E2E driving, and that its performance in an untrained context is considerably superior to RGB. In most instances, however, multi-modality (RGB-D) performs better than utilizing just one model, such as RGB or simply D. The most apparent scenario is a new town and weather conditions with dynamic objects, i.e., under challenging circumstances. Single modality of RGB can obtain a success rate of 53.5%, D alone 71.62%. At the same time, the early fusion of multi-modality achieved a success rate of 94.3%. The primary issue we aimed to address in this article, is whether the early fusion of multi-modality may enhance the performance of conditional imitation learning over a single model.

Table 3. Mean and Standard Deviation of Success Rate on the Original CARLA Benchmark.

			Active		Estimated				Active		Estimated	
Task	**SS**	**RGB**	**D**	**EF**	**D**	**EF**	**SS**	**RGB**	**D**	**EF**	**D**	**EF**
	Training Conditions						New Town					
Straight	98.0	96.3	98.7	98.3	92.3	97.3	100	84.0	94.3	96.3	78.3	71.6
One Turn	100.0	95.0	92.0	99.0	84.6	96.3	96.6	68.0	74.3	79.0	46.3	47.0
Navigation	96.0	89.0	89.3	92.6	75.3	94.3	96.0	59.6	85.3	90.0	45.6	46.6
Nav. Dynamic	92.0	84.0	82.6	89.3	71.0	90.6	99.3	54.3	70.3	84.3	44.3	46.6
	New Weather						New Town and Weather					
Straight	100.0	84.0	99.3	96.0	92.0	84.6	100	84.6	97.3	97.3	78.0	89.3
One Turn	100.0	76.6	94.6	94.6	93.3	80.6	96.0	66.6	72.7	82.7	62.6	64.0
Navigation	95.3	72.6	89.3	91.3	73.3	80.6	96.0	57.3	84.0	92.7	55.3	60.7
Nav. Dynamic	92.6	68.6	90.0	86.0	76.6	77.3	98.0	46.7	68.3	94.0	54.0	49.3

5.3. Grad-CAM Visualizations

Figure 7 presents a Grad-CAM visualization of different methods using random samples from training conditions for qualitative analysis [78]. Using Grad-CAM, we can investigate what aspects of the input significantly affect the output. Our first step was to visualize the salient parts of the structure that contributed to control output. Figure 7 shows how Grad-CAM attention maps cover useful regions, such as the center line or the border between the road and sidewalk. Our proposed method looks at narrow, yet important, points, such as the center of the road and the leading vehicle, while CIL might highlight irrelevant information and MT appears to be distracted by large regions. We visualized attention maps based on traffic light predictions as a second step. MT and CIRL widely capture the traffic light, including the pole, when it is red, whereas our proposed approach attentively looks at it, but the red color point is not included.

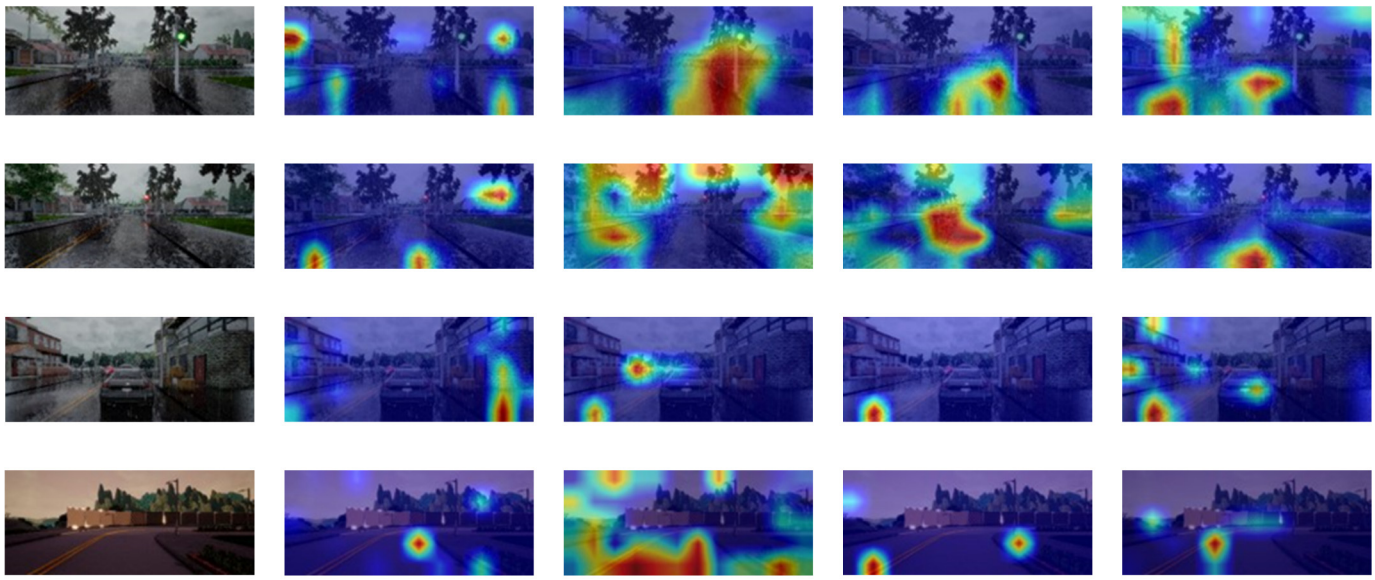
5.4. Prediction of Steering Wheel Angle

To further analyze our proposed model goodness, we compared it to predicting the steering wheel angle using single and multi-modality methods. The mean square error (MSE) loss and forward-facing camera were used to train the first model. The MSE loss and RGB and depth fusion were utilized in the second model (D). The steering wheel angle was standardized between -1.0 and $+1.0$, as illustrated in Figure 8, with positive values indicating right-side rotation, and negative values indicating left-side rotation. We used the linear transformation Equation (4) to normalize the value:

$$\theta_{normalized} = -0.5 + max\left(0,\ min\left(1.0, \frac{\theta_{raw} - \theta_{min}}{\theta_{max} - \theta_{min}}\right)\right) \quad (4)$$

where, $\theta_{normalized}$ is the normalized steering angle between -1 and $+1$; θ_{raw} is the actual steering wheel angle (in radians) measured from the vehicle; θ_{max} and θ_{min} represent the

maximum and minimum steering wheel angle, respectively. Similarly, we normalized throttles values between −1 and 1.



(**a**) Input Image (**b**) CIL (**c**) MT (**d**) CIRL (**e**) Ours

Figure 7. Qualitative Results Analysis by Grad-CAM Visualization.

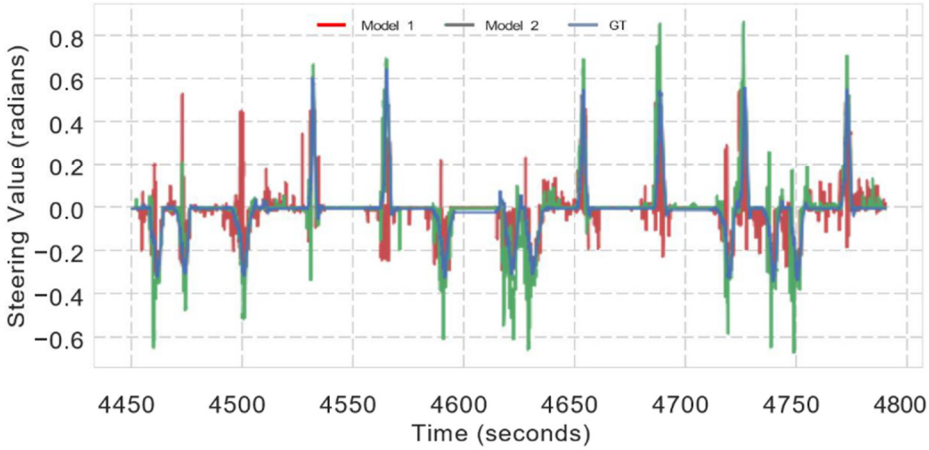


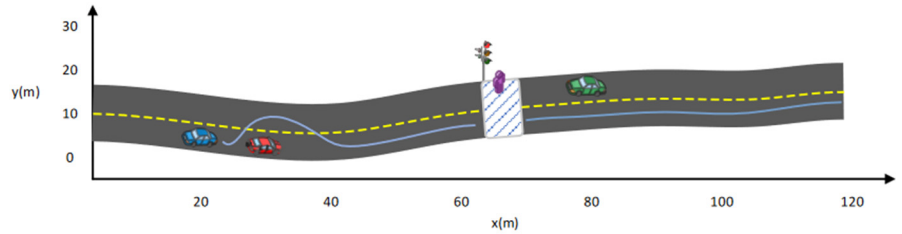
Figure 8. Detailed evaluation of two driving models with the prediction of steering wheel angle.

The ground truth steering signal over time is presented in the blue line. Figure 8 shows the predictions of the basic CIL model (Model 1) and the early fusion of the multi-model (Model 2) in red and green, respectively. The model predictions have a significant qualitative difference: both often depart from the ground reality.

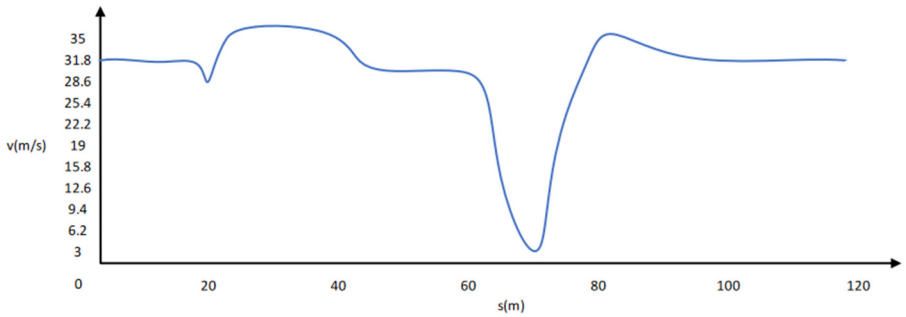
5.5. Path Planning

Specifically, here are the results include waiting for pedestrian crossings and overtaking a vehicle in front. There is a predefined map with vehicles and pedestrians on it. Cubic splines describe the road edges. White dashed lines indicate the center lines of these roads. A zebra crossing is represented by the rectangle box with white and blue lines on the road. The yellow dash lines show the vehicle path. Figures 9 and 10a,b show path-planning moments, overall trajectories, and the speed graph of the vehicle, respectively. Here s and v are the shift and speed. The speed graph in this paper represents total values, which account for many factors, including decreasing speed, throttling, and air and ground resistance.

Figure 10b shows the first left turn effect of the road on the second curve. Meanwhile, the blue car curve effect does not appear on the x-axis because it is also moving, and its curve does not exit under 120 m once the agent crosses it.

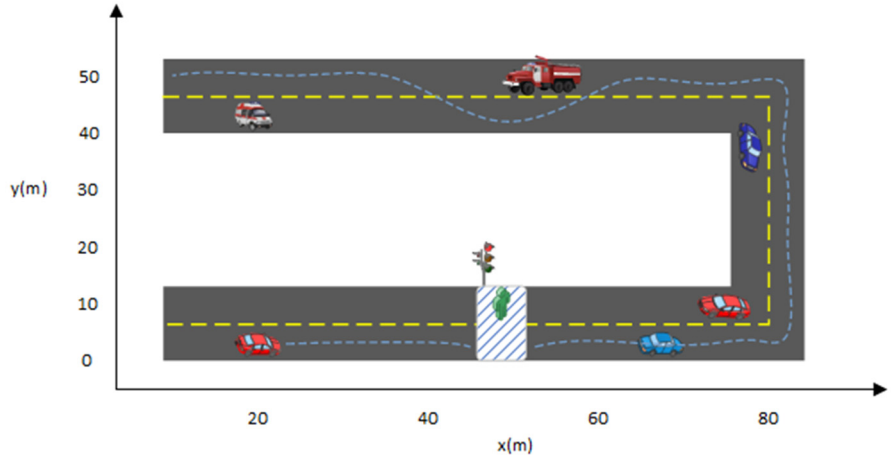


(**a**)



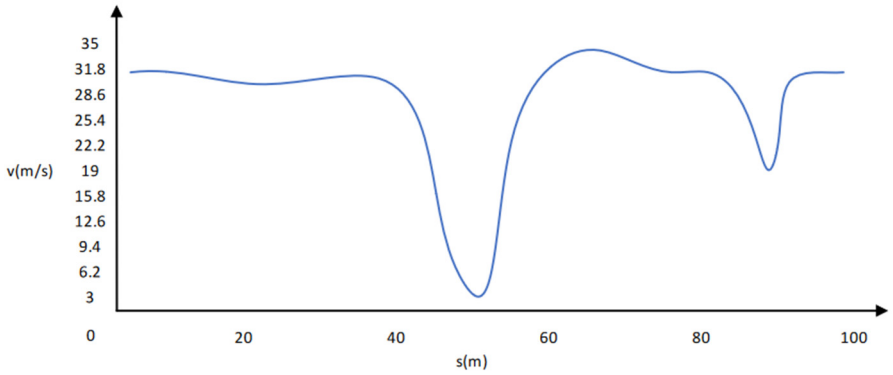
(**b**)

Figure 9. Dynamic path planning results in a straight road with multiple challenges. (**a**) Vehicle overtakes the vehicle in front and waits for the green signal, (**b**) Speed Curve.



(**a**)

Figure 10. *Cont.*



(**b**)

Figure 10. Dynamic path planning results for the U shape road with multiple challenges. (**a**) Vehicle overtakes the vehicles in front, takes two left turns, and waits for the green signal, (**b**) Speed Curve.

6. Conclusions

This paper presents a comparison of single- and multi-modal perception data for end-to-end driving. We focused our analysis on the RGB and depth data, since they are usually available in autonomous vehicles through cameras and active sensors, such as LiDAR. In this study, we examined the driving performance of single-modal CIL, MP, RL, CAL, MT and CIRL models, as well as multi-modal CIL (RGB, depth) models according to early fusion paradigms using the CARLA simulation environment. The depth information available in CARLA is post-processed in order to obtain a more realistic range of distances and depth accuracy. This depth is also used to train a depth estimation model so that the experiments cover multi-modality, not only from a multisensory perspective (RGB and active depth), but also from a single-sensory perspective (RGB and estimated depth). Our results show that the multi-modal CIL models significantly improve performances in all scenarios, especially in "New Town" and "New Town and New Weather" compared to the single models.

There are a number of limitations to the method presented here that we believe are essential to resolve, on order to achieve (and exceed) human-level driving. This means that the user cannot access long-term dependencies and cannot make sense of the road scene. In addition, this method lacks a long-term planning model that is important for safely interacting with occluded dynamic agents. There are many promising directions for future research. The time and safety measures needed for a closed-loop policy are major limitations; thus, robustly evaluating a policy offline and quantifying its performance is a critical research area.

In the future, a video might be used to provide temporal information as well. In this way, the end-to-end driving model is able to distinguish between dynamic and static objects more precisely. Furthermore, it would be interesting to see if the proposed method could be applied to real-world data, such as the BDD100K dataset [79], then to demonstrate how a real car would behave under our control. We could also try to improve the model's ability to avoid collisions by including other useful information. In urban areas, it is possible to achieve autonomous driving at the L4 level or even at the L5 level with simple navigation instructions if the generalization abilities of this model are strong enough.

Author Contributions: Conceptualization, M.H.; methodology, M.H.; software, M.H.; validation, M.H. and A.G.; formal analysis, M.H.; investigation, M.H.; resources, M.H.; data curation, M.H.; writing—original draft preparation, M.H.; writing—review and editing, A.G.; visualization, M.H.; supervision, A.G.; project administration, M.H. and A.G.; funding acquisition, A.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the AGH University of Science and Technology, Grant No.: 16.16.120.773.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Garcia-Bedoya, O.; Hirota, S.; Ferreira, J.V. Control system design for an automatic emergency braking system in a sedan vehicle. In Proceedings of the 2019 2nd Latin American Conference on Intelligent Transportation Systems (ITS LATAM), Bogota, Colombia, 19–20 March 2019.
2. Perrier, M.J.R.; Louw, T.L.; Carsten, O. User-centred design evaluation of symbols for adaptive cruise control (ACC) and lane-keeping assistance (LKA). *Cogn. Technol. Work* **2021**, *23*, 685–703. [CrossRef]
3. Haris, M.; Hou, J. Obstacle Detection and Safely Navigate the Autonomous Vehicle from Unexpected Obstacles on the Driving Lane. *Sensors* **2020**, *20*, 4719. [CrossRef] [PubMed]
4. Qin, Y.; Hashemi, E.; Khajepour, A. Integrated Crash Avoidance and Mitigation Algorithm for Autonomous Vehicles. *IEEE Trans. Ind. Inform.* **2021**, *17*, 7246–7255. [CrossRef]
5. Hrovat, D.; Hubbard, M. Optimum Vehicle Suspensions Minimizing RMS Rattlespace, Sprung-Mass Acceleration and Jerk. *J. Dyn. Syst. Meas. Control* **1981**, *103*, 228–236. [CrossRef]
6. Huang, Q.; Wang, H. *Fundamental Study of Jerk: Evaluation of Shift Quality and Ride Comfort*; SAE Technical Paper; State Key Laboratory of Automotive Safety and Energy Tsinghua University: Beijing, China, 2004.
7. Lv, Q.; Sun, X.; Chen, C.; Dong, J.; Zhou, H. Parallel Complement Network for Real-Time Semantic Segmentation of Road Scenes. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–13. [CrossRef]
8. Hamian, M.H.; Beikmohammadi, A.; Ahmadi, A.; Nasersharif, B. Semantic Segmentation of Autonomous Driving Images by the combination of Deep Learning and Classical Segmentation. In Proceedings of the 2021 26th International Computer Conference, Computer Society of Iran (CSICC), Tehran, Iran, 3–4 March 2021.
9. Zhou, K.; Zhan, Y.; Fu, D. Learning Region-Based Attention Network for Traffic Sign Recognition. *Sensors* **2021**, *21*, 686. [CrossRef]
10. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef]
11. Yang, B.; Luo, W.; Urtasun, R. Pixor: Real-time 3d object detection from point clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7652–7660.
12. Qi, C.R.; Liu, W.; Wu, C.; Su, H.; Guibas, L.J. Frustum pointnets for 3d object detection from rgb-d data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 918–927.
13. Haris, M.; Glowacz, A. Road object detection: A comparative study of deep learning-based algorithms. *Electronics* **2021**, *10*, 1932. [CrossRef]
14. Haris, M.; Glowacz, A. Lane Line Detection Based on Object Feature Distillation. *Electronics* **2021**, *10*, 1102. [CrossRef]
15. Haris, M.; Hou, J.; Wang, X. Multi-scale spatial convolution algorithm for lane line detection and lane offset estimation in complex road conditions. *Signal Process. Image Commun.* **2021**, *99*, 116413. [CrossRef]
16. Haris, M.; Hou, J.; Wang, X. Lane Lines Detection under Complex Environment by Fusion of Detection and Prediction Models. *Transp. Res. Rec.* **2021**, 03611981211051334. [CrossRef]
17. Haris, M.; Hou, J.; Wang, X. Lane line detection and departure estimation in a complex environment by using an asymmetric kernel convolution algorithm. *Vis. Comput.* **2022**, 1–20. [CrossRef]
18. Gurram, A.; Urfalioglu, O.; Halfaoui, I.; Bouzaraa, F.; López, A.M. Monocular depth estimation by learning from heterogeneous datasets. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Suzhou, China, 26–30 June 2018; IEEE: Suzhou, China, 2018; pp. 2176–2181.
19. Gan, Y.; Xu, X.; Sun, W.; Lin, L. Monocular depth estimation with affinity, vertical pooling, and label enhancement. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 224–239.
20. Fu, H.; Gong, M.; Wang, C.; Batmanghelich, K.; Tao, D. Deep ordinal regression network for monocular depth estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2002–2011.
21. Shin, Y.-S.; Park, Y.S.; Kim, A. Direct visual slam using sparse depth for camera-lidar system. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; IEEE: Brisbane, QLD, Australia, 2018; pp. 5144–5151.
22. Qiu, K.; Ai, Y.; Tian, B.; Wang, B.; Cao, D. Siamese-ResNet: Implementing loop closure detection based on Siamese network. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Suzhou, China, 26–30 June 2018; IEEE: Suzhou, China, 2018; pp. 716–721.
23. Yin, H.; Tang, L.; Ding, X.; Wang, Y.; Xiong, R. Locnet: Global localization in 3d point clouds for mobile vehicles. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Suzhou, China, 26–30 June 2018; IEEE: Suzhou, China, 2018; pp. 728–733.
24. Pomerleau, D.A. Alvinn: An Autonomous Land Vehicle in a Neural Network. Available online: https://proceedings.neurips.cc/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf (accessed on 28 December 2021).
25. Bojarski, M.; Del Testa, D.; Dworakowski, D.; Firner, B.; Flepp, B.; Goyal, P.; Jackel, L.D.; Monfort, M.; Muller, U.; Zhang, J.; et al. End to End Learning for Self-Driving Cars. *arXiv* **2016**, arXiv:1604.07316.
26. Muller, U.; Ben, J.; Cosatto, E.; Flepp, B.; Cun, Y.L. Off-Road Obstacle Avoidance Through End-to-End Learning. Available online: https://proceedings.neurips.cc/paper/2005/file/fdf1bc5669e8ff5ba45d02fded729feb-Paper.pdf (accessed on 28 December 2021).

27. Codevilla, F.; Miiller, M.; Lopez, A.; Koltun, V.; Dosovitskiy, A. End-to-End Driving Via Conditional Imitation Learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; IEEE: Brisbane, QLD, Australia, 2018; pp. 4693–4700. [CrossRef]
28. Xu, H.; Gao, Y.; Yu, F.; Darrell, T. End-to-end learning of driving models from large-scale video datasets. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2174–2182.
29. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; Volume 1, pp. 431–440.
30. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
31. Eraqi, H.M.; Moustafa, M.N.; Honer, J. End-to-end deep learning for steering autonomous vehicles considering temporal dependencies. *arXiv* **2017**, arXiv:1710.03804.
32. Hou, Y.; Hornauer, S.; Zipser, K. Fast recurrent fully convolutional networks for direct perception in autonomous driving. *arXiv* **2017**, arXiv:1711.06459.
33. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An open urban driving simulator. In Proceedings of the Conference on Robot Learning, PMLR, Mountain View, CA, USA, 13–15 November 2017; pp. 1–16.
34. Wang, Q.; Chen, L.; Tian, W. End-to-end driving simulation via angle branched network. *arXiv* **2018**, arXiv:1805.07545.
35. Liang, X.; Wang, T.; Yang, L.; Xing, E. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. In Proceedings of the The European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 584–599.
36. Li, Z.; Motoyoshi, T.; Sasaki, K.; Ogata, T.; Sugano, S. Rethinking self-driving: Multi-task knowledge for better generalization and accident explanation ability. *arXiv* **2018**, arXiv:1809.11100.
37. Sauer, A.; Savinov, N.; Geiger, A. Conditional affordance learning for driving in urban environments. In Proceedings of the Conference on Robot Learning, PMLR, Zürich, Switzerland, 29–31 October 2018; pp. 237–252.
38. Müller, M.; Dosovitskiy, A.; Ghanem, B.; Koltun, V. Driving policy transfer via modularity and abstraction. *arXiv* **2018**, arXiv:1804.09364.
39. Rhinehart, N.; McAllister, R.; Levine, S. Deep imitative models for flexible inference, planning, and control. *arXiv* **2018**, arXiv:1810.06544.
40. Thrun, S.; Montemerlo, M.; Dahlkamp, H.; Stavens, D.; Aron, A.; Diebel, J.; Fong, P.; Gale, J.; Halpenny, M.; Hoffmann, G.; et al. Stanley: The robot that won the DARPA Grand Challenge. *J. Field Robot.* **2006**, *23*, 661–692. [CrossRef]
41. Ziegler, J.; Bender, P.; Schreiber, M.; Lategahn, H.; Strauss, T.; Stiller, C.; Dang, T.; Franke, U.; Appenrodt, N.; Keller, C.G.; et al. Making Bertha Drive—An Autonomous Journey on a Historic Route. *IEEE Intell. Transp. Syst. Mag.* **2014**, *6*, 8–20. [CrossRef]
42. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
43. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
44. Li, Y.; Qi, H.; Dai, J.; Ji, X.; Wei, Y. Fully convolutional instance-aware semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2359–2367.
45. Sun, D.; Yang, X.; Liu, M.-Y.; Kautz, J. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8934–8943.
46. Güney, F.; Geiger, A. Deep discrete flow. In Proceedings of the Asian Conference on Computer Vision, Hyderabad, India, 13–16 January 2006; Springer: Berlin/Heidelberg, Germany, 2016; pp. 207–224.
47. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Rob. Res.* **2016**, *32*, 1231–1237. [CrossRef]
48. Zhang, H.; Geiger, A.; Urtasun, R. Understanding high-level semantics by modeling traffic patterns. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, NSW, Australia, 1–8 December 2013; pp. 3056–3063.
49. Geiger, A.; Lauer, M.; Wojek, C.; Stiller, C.; Urtasun, R. 3D Traffic Scene Understanding From Movable Platforms. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 1012–1025. [CrossRef] [PubMed]
50. Schwarting, W.; Alonso-Mora, J.; Rus, D. Planning and Decision-Making for Autonomous Vehicles. *Annu. Rev. Control Robot. Auton. Syst.* **2018**, *1*, 187–210. [CrossRef]
51. Bojarski, M.; Yeres, P.; Choromanska, A.; Choromanski, K.; Firner, B.; Jackel, L.; Muller, U. Explaining how a deep neural network trained with end-to-end learning steers a car. *arXiv* **2017**, arXiv:1704.07911.
52. Hubschneider, C.; Bauer, A.; Weber, M.; Zöllner, J.M. Adding navigation to the equation: Turning decisions for end-to-end vehicle control. In Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), Yokohama, Japan, 16–19 October 2017; IEEE: Yokohama, Japan, 2017; pp. 1–8.
53. Amini, A.; Rosman, G.; Karaman, S.; Rus, D. Variational end-to-end navigation and localization. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; IEEE: Montreal, QC, Canada, 2019; pp. 8958–8964.
54. Barto, A.G.; Mahadevan, S. Recent Advances in Hierarchical Reinforcement Learning. *Discret. Event Dyn. Syst.* **2003**, *13*, 41–77. [CrossRef]
55. Sutton, R.S.; Precup, D.; Singh, S. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.* **1999**, *112*, 181–211. [CrossRef]

56. Konidaris, G.; Kuindersma, S.; Grupen, R.; Barto, A. Robot learning from demonstration by constructing skill trees. *Int. J. Robot. Res.* **2011**, *31*, 360–375. [CrossRef]
57. Kulkarni, T.D.; Narasimhan, K.R.; Saeedi, A.; Tenenbaum, J.B. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *arXiv* **2016**, arXiv:1604.06057.
58. Kober, J.; Bagnell, J.A.; Peters, J. Reinforcement learning in robotics: A survey. *Int. J. Robot. Res.* **2013**, *32*, 1238–1274. [CrossRef]
59. Pastor, P.; Hoffmann, H.; Asfour, T.; Schaal, S. Learning and generalization of motor skills by learning from demonstration. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; IEEE: Kobe, Japan, 2009; pp. 763–768.
60. Da Silva, B.; Konidaris, G.; Barto, A. Learning parameterized skills. *arXiv* **2012**, arXiv:1206.6398.
61. Deisenroth, M.P.; Englert, P.; Peters, J.; Fox, D. Multi-task policy search for robotics. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; IEEE: Hong Kong, China, 2014; pp. 3876–3881.
62. Kober, J.; Wilhelm, A.; Oztop, E.; Peters, J. Reinforcement learning to adjust parametrized motor primitives to new situations. *Auton. Robot.* **2012**, *33*, 361–379. [CrossRef]
63. Schaul, T.; Horgan, D.; Gregor, K.; Silver, D. Universal value function approximators. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; pp. 1312–1320.
64. Dosovitskiy, A.; Koltun, V. Learning to act by predicting the future. *arXiv* **2016**, arXiv:1611.01779.
65. Javdani, S.; Srinivasa, S.S.; Bagnell, J.A. Shared autonomy via hindsight optimization. *Robot. Sci. Syst.* **2015**. [CrossRef]
66. Chen, C.; Seff, A.; Kornhauser, A.; Xiao, J. Deep Driving: Learning Affordance for Direct Perception in Autonomous Driving. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015.
67. Al-Qizwini, M.; Barjasteh, I.; Al-Qassab, H.; Radha, H. Deep learning algorithm for autonomous driving using googlenet. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Redondo Beach, CA, USA, 11–14 June 2017; IEEE: Los Angeles, CA, USA, 2017; pp. 89–96.
68. Huang, J.; Tanev, I.; Shimohara, K. Evolving a general electronic stability program for car simulated in TORCS. In Proceedings of the 2015 IEEE Conference on Computational Intelligence and Games (CIG), Tainan, Taiwan, 31 August 2015–1 September 2015; IEEE: Tainan, Taiwan, 2015; pp. 446–453.
69. Richter, S.R.; Vineet, V.; Roth, S.; Koltun, V. Playing for data: Ground truth from computer games. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 102–118.
70. Ebrahimi, S.; Rohrbach, A.; Darrell, T. Gradient-free policy architecture search and adaptation. In Proceedings of the Conference on Robot Learning, PMLR, Mountain View, CA, USA, 13–15 November 2017; pp. 505–514.
71. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv* **2016**, arXiv:1603.04467.
72. Chetlur, S.; Woolley, C.; Vandermersch, P.; Cohen, J.; Tran, J.; Catanzaro, B.; Shelhamer, E. cuDNN: Efficient Primitives for Deep Learning. *arXiv* **2014**, arXiv:1410.0759.
73. Wymann, B.; Espié, E.; Guionneau, C.; Dimitrakakis, C.; Coulom, R.; Sumner, A. Torcs, the Open Racing Car Simulator. 2000. Available online: http//torcs.sourceforge.net (accessed on 6 March 2021).
74. Codevilla, F.; López, A.M.; Koltun, V.; Dosovitskiy, A. On Offline Evaluation of Vision-Based Driving Models. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 236–251.
75. Premebida, C.; Carreira, J.; Batista, J.; Nunes, U. Pedestrian detection combining RGB and dense LIDAR data. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; IEEE: Chicago, IL, USA, 2014; pp. 4112–4117.
76. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? the KITTI vision benchmark suite. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
77. Kingma, D.P.; Ba, J.L. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
78. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.
79. Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V.; Darrell, T. Bdd100k: A diverse driving dataset for heterogeneous multi-task learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2636–2645.