LeTFuser: Light-weight End-to-end Transformer-Based Sensor Fusion for Autonomous Driving with Multi-Task Learning

Pedram Agand* Mohammad Mahdavian* Manolis Savva Mo Chen
Simon Fraser University, 8888 University Dr W, Burnaby, BC, Canada.
{pedram_agand, mohammad_mahdavian, manolis_savva}@sfu.ca, mochen@cs.sfu.ca

Abstract

In end-to-end autonomous driving, the utilization of existing sensor fusion techniques for imitation learning proves inadequate in challenging situations that involve numerous dynamic agents. To address this issue, we introduce LeTFuser, a transformer-based algorithm for fusing multiple RGB-D camera representations. To perform perception and control tasks simultaneously, we utilize multi-task learning. Our model comprises of two modules, the first being the perception module that is responsible for encoding the observation data obtained from the RGB-D cameras. It carries out tasks such as semantic segmentation, semantic depth cloud mapping (SDC), and traffic light state recognition. Our approach employs the Convolutional vision Transformer (CvT) [39] to better extract and fuse features from multiple RGB cameras due to local and global feature extraction capability of convolution and transformer modules, respectively. Following this, the control module undertakes the decoding of the encoded characteristics together with supplementary data, comprising a rough simulator for static and dynamic environments, as well as various measurements, in order to anticipate the waypoints associated with a latent feature space. We use two methods to process these outputs and generate the vehicular controls (e.g. steering, throttle, and brake) levels. The first method uses a PID algorithm to follow the waypoints on the fly, whereas the second one directly predicts the control policy using the measurement features and environmental state. We evaluate the model and conduct a comparative analysis with recent models on the CARLA simulator using various scenarios, ranging from normal to adversarial conditions, to simulate real-world scenarios. Our code is available at https://github.com/pagand/e2etransfuser/tree/cvpr-w to facilitate future studies.

* Equal contribution

1. Introduction

Many works in the autoonomous driving literature has been focusing on different aspects of perception and control tasks for safe navigation [2, 4, 8, 30, 32, 36, 37, 43]. Recent advances in end-to-end driving neural network (NN) models have demonstrated remarkable results using single modality inputs, such as image and LiDAR [16]. However, these approaches face limitations in complex urban scenarios involving adversarial situations due to their lack of 3D scene understanding [13]. Sensor fusion has shown promise in addressing these challenges by integrating multiple sensor modalities, such as cameras and LiDAR sensors, to create a more comprehensive scene representation [1, 9, 15]. Despite the improvements, these fusion methods often require large computational resources and face challenges in balancing learning signals between perception and control tasks [41]. Moreover, integrating multiple modalities with different data shapes and representations requires sophisticated preprocessing techniques such as ELPP [10], SaDMS [25], leading to increased model complexity and the potential for information loss.

Recent advancements in end-to-end autonomous driving have explored the integration of different sensor modalities, such as LiDAR, RGB and RGB-D cameras, to enhance performance. One notable approach is [41], which employs Convolutional Neural Network (CNN) to extract data features provided by a RGB-D camera and fuses them to extract future vehicle waypoints and navigational signals. Another well-known perception only method is TransFuser [7], which uses a multi-modal fusion transformer to incorporate global context and pairwise interactions into the feature extraction layers of different input modalities. We use both these methods as our baselines and we show combining ideas from these approaches could potentially lead to more robust and accurate end-to-end autonomous driving solutions. We achieved this by leveraging the strengths of both global and local context reasoning provided by transformers and CNNs and trajectory-guided control.

In this paper, we propose a novel deep neural network ar-

chitecture for end-to-end autonomous driving that leverages the complementary advantages of RGB and depth information provided by an RGB-D camera, addressing the challenges faced by existing single modality and sensor fusion approaches. Our model consists of two main modules: the perception module, which encodes high-dimensional observation data and performs semantic segmentation, semantic depth cloud (SDC) mapping and ego vehicle speed and traffic light prediction; and the control module, which decodes the features encoded by the perception module along with additional GPS, command and speedometer information to predict waypoints and control policy.

In the perception module, we utilize Convolutional vision Transformers (CvT) [39] and EfficientNet [35] to adeptly extract RGB image and SDC map features. We then fuse them using a CNN-based fusion layer. Additionally, we employ two agents in the control module to process the perception module's outputs, fostering diversified and resilient decision-making. To tackle the issue of balancing learning signals, similar to [41], we implement a Modified Gradient Normalization (MGN) method, ensuring uniform learning pace across all tasks. Finally, we evaluated our model on the CARLA simulator with various scenarios, including normal-adversarial situations, demonstrating improved performance over baseline methods.

2. Related works

Multi-Modality: Recent advancements in multi-modal end-to-end autonomous driving have highlighted the potential of using RGB images alongside depth and semantic information to enhance driving performance [44]. Studies by [2, 41] have investigated the effectiveness of incorporating depth and semantic data as intermediate representations for driving tasks. In our work, we focus on combining RGB and Depth inputs, providing complementary scene representations and are readily available in autonomous systems.

Sensor Fusion: Most sensor fusion research has focused on perception tasks such as object detection [3, 14, 22] and motion forecasting [12, 27, 42]. These approaches typically include multi-view LiDAR data or combine camera input with LiDAR data by projecting features between different spaces. ContFuse [23] is an approach fusing multi-scale RGB and LiDAR bird's eye view (BEV) features densely. However, these methods do not capture the global context of the 3D scene, which is crucial for safe navigation in challenging scenarios. We implement a multi-scale geometry-based fusion mechanism inspired by [22, 23] but find it insufficient for complex urban driving situations. Our proposed attention-based multi-modal fusion transformer overcomes this limitation by incorporating global contextual reasoning, resulting in improved driving performance.

Bird's Eye View Strategies: In this domain, researchers either use lidar data or fuse RGB images and depth maps from a single RGB-D camera [28]. They project the depth map with the semantic segmentation to create a semantic depth cloud (SDC) with a BEV perspective. This enables the model to perceive the environment from a top-view perspective and benefits from the clearer occupied or navigable regions provided by the SDC's semantic information compared to LiDAR point clouds containing only height data [16, 33]. Huang et al. [18] fused RGB images and depth maps to capture a deeper global context, while Prakash et al. [33] combined RGB images and preprocessed LiDAR point clouds to leverage different perspectives, such as front-view and BEV. These approaches used either high-level navigational commands [18] or sparse GPS locations provided by a global planner [33] for driving. In our research, we consider using a sequence of routes instead of high-level navigational commands, as this better reflects real-world autonomous driving conditions [17].

Imitation Learning: Studies in end-to-end autonomous driving usually fall into two categories: reinforcement learning (RL) and imitation learning (IL). Authors in [21, 24] have shown the potential of RL, while IL approaches such as LBC [4] and NEAT [6] have demonstrated impressive performance. Our work adapts the auto-regression scheme used in TransFuser and its variants [20, 33].

End-to-End Autonomous Driving: End-to-end multi-task learning approaches offer benefits in training efficiency and integration simplicity. Imitation learning-based methods have been investigated for autonomous driving tasks, with the former exploring additional perception tasks to improve feature extraction [4, 19]. Combinations of various autonomous driving tasks, such as object detection, lane detection, semantic segmentation, and depth estimation have been proven to achieve incredible performance [5, 38]. In our work, we adopt a similar multi-task learning approach, but utilize depth from an RGB-D camera as input [15]. We address the imbalanced learning problem in multi-task learning by implementing a MGN algorithm [29].

3. Methodology

In this section, we present the details of our proposed approach. Our model is structured around two key components: the perception and control modules, each fulfilling a specific set of tasks to enable the vehicle navigation. In the perception side, we extract features from RGB and a BEV SDC map. Also, the model accurately predicts traffic lights and ego vehicle speed from the RGB embedded features, ensuring optimal navigation. In the subsequent control stage, our model utilizes the extracted features, current ego vehicle speed, and GPS location to provide reliable waypoints and vehicular commands.

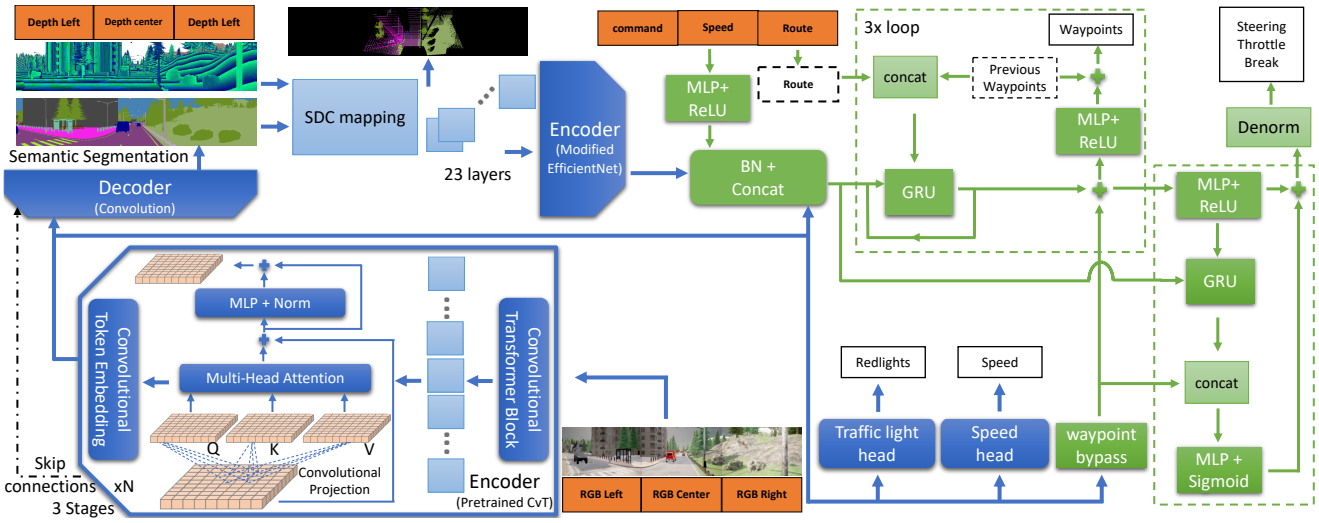

Figure 1. Model architecture. It consists of trainable and non-trainable components represented by light and dark-colored items, respectively. The perception module is denoted by blue-colored items, while the controller module is represented by green, and the inputs are orange. The white box represent different tasks that are learned simultaneously. The process inside the waypoint dashed green line box is iterated three times during the training, in which the model predicts waypoints and estimates the level of vehicular controls independently.

3.1. Perception Module

As illustrated in Fig. 1, the perception module serves to extract essential features from RGB images using CvT [39] to perform semantic segmentation and generate a BEV SDC map. Our perception module receives a total of three RGB images from three vehicle cameras, with the first camera capturing the front view angle and the other two tilted to the left and right by 60 degrees. To provide a comprehensive understanding of the environment, depth images are also captured from each camera. The front RGB and depth images have a resolution of 160×320, while the non-overlapping side cameras capture images with the resolution of 160×224. This results in a total of 160×768 pixels for both RGB and depth images.

3.1.1 Convolutional Vision Transformer

The cornerstone of our feature extractor is the CvT [39] that has been pretrained on the ImageNet [11] shown in bottom left section of the Fig. 1. This particular network was selected for its unique ability to leverage both convolution and transformer modules, providing unparalleled benefits to our feature extraction process. Convolution layers excel at extracting local features, while transformers are known for their ability in global feature extraction and learning. By combining the strengths of these two powerful techniques, CvT ensures that our feature extractor is capable of capturing both local and global features, resulting in truly comprehensive visual representations.

CvT-13 [39], a light version of the CvT, has been carefully selected for its exceptional performance and fewer number of parameters. As one can see in Fig. 1 , it has been designed with three main stages, each of which incorporates a Convolutional Token Embedding (CTE) to process the 2D input images. The features extracted by the CTE are then normalized and passed through a Convolutional Transformer Block (CTB). To apply both convolutional and attention layers, the CTB uses a depth-wise separable convolution operation known as Convolutional Projection to create the query, key, and value embeddings. These embeddings are then passed through a transformer module to extract global features, ultimately resulting in highly accurate and comprehensive feature maps. The last layer of CvT is a fully connected layer used for image classification that we remove from the model. As a result, the RGB extracted features contains 384 features, each with a size of 10×48.

3.1.2 Semantic Segmentation

After the feature maps are extracted, we use them in different sections of the model, as they contain valuable information. First, they are utilized to train a semantic segmentation decoder capable of accurately identifying 23 different classes depicted in the Fig. 1 as ”Decoder”. Later, we use this to create a BEV SDC map. To achieve this, we have developed a segmentation decoder that consists of three convolutional layers and a final pointwise convolution with sigmoid activation. By leveraging skip connections, we can effectively capture both local and global features, resulting in accurate and comprehensive segmentation maps.

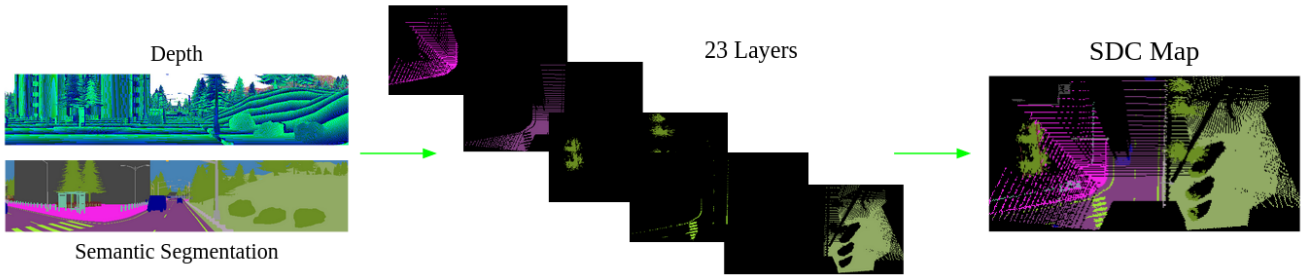

Figure 2. Semantic Depth Cloud (SDC) map created from three depth and semantic segmentations acquired from the three cameras attached to the vehicle. The SDC mapping creates three separate BEV maps using estimated semantic segmentation and attaches them together.

3.1.3 Semantic Depth Cloud

In order to enhance the understanding of the scene, we have utilized a method that involves the creation of a BEV Semantic Depth Cloud using the provided depth images and estimated semantic segmentation, in addition to the RGB images [28]. The SDC map represents the ego-vehicle's surrounding environment and contains richer information with respect to the lidar data due to the 23-layered classified information. Each layer represents one class of environment object. According to the Fig. 2, the SDC is created without considering the height information (y-axis), as a BEV perspective requires only the x-axis and z-axis. The process of creating the SDC involves separately generating a SDC map for each camera, and then merging them together after rotating the side maps with an appropriate angle.

We define a 64-meter distance range in front of the vehicle and 32 meters to each side of the camera, creating a coverage area of 64×64 square meters. The SDC maps have dimensions of 160×320 square centimeters for the front and 160×224 for the sides. We generate a transformation matrix for the x-axis using camera parameters and normalize the coordinates to align with the SDC tensor spatial dimension. One-hot encoding is applied to yield a 23-channel SDC tensor. The resulting maps are copied to an empty tensor with dimensions of 160×768, with side maps rotated at a 42-degree angle. For feature extraction, we use the compact EfficientNet-B1 [35] network to generate a tensor of 192 features, each with a size of 10×48.

3.2. Controller Module

The control module depicted as green squares in the Fig. 1, receives RGB, SDC and navigational measurements features and uses them to predict the vehicle's future waypoints. The navigational measurements includes route location, navigational command provided by the global planner and the ego vehicle speed. The navigational command specifies the vehicle's general direction, such as left, right, forward, stop, etc., and is defined as a one-hot vector. Subsequently, the control module predicts the appropriate vehicular control, including steering, throttle, and brake, based on the predicted waypoints and fused features. To predict the vehicle's future waypoints, we employ a gated recurrent unit (GRU) similar to [28]. The GRU is a suitable choice as it addresses the vanishing gradient problem while maintaining a better performance-cost ratio compared to other RNN methods. To train a control model that predicts current control actions based on current input, behavior cloning is commonly used but relies on the assumption of independent and identically distributed (IID) data, which is not valid for closed-loop tests [40]. To overcome this issue without resorting to reinforcement learning, we used a similar trick to [40] for predicting multi-step control actions into the future. To this end, we first employ a waypoint branch that utilizes fused features and environment-agent static knowledge through waypoint bypass. Further, we deploy a dynamic branch to capture the environment-agent dynamic interaction given the learned static knowledge. The dynamic branch provides dynamic information such as object motion and traffic light changes, while the waypoint branch incorporates static information like curbs and lanes and improves spatial consistency across both branches.

In order to fuse the extracted features from RGB images and the SDC map, we concatenate them and apply a batch normalization (BN) layer to them and then concatenate it with the measurement tensor. The GRU in waypoint branch takes the fused features as the initial hidden state, and the inputs include the current waypoint in the BEV space, the route location coordinate transformed to the BEV space. The initial waypoint coordinate is always positioned at $(0, 0)$, the bottom-center point of the SDC map. To transform the global coordinates (x_g, y_g) to local coordinates (x_l, y_l), we use the Eq. 1. We then add the next hidden state from the GRU to waypoint bypass, which is the RGB features that have passed through a biasing module. The biasing module consists of adaptive global pooling and a linear layer applied to the RGB extracted features. Finally, a sigmoid function is applied to convert all values between 0 and 1. We apply a multi-layer perceptron (MLP)

network containing two linear layers and a rectified linear unit (ReLU) to the biased GRU hidden state to obtain normalized control commands in the range of 0 to 1.

$$\begin{bmatrix} x_l \\ y_l \end{bmatrix} = \begin{bmatrix} cos(90+\theta_v) & -sin(90+\theta_v) \\ sin(90+\theta_v) & cos(90+\theta_v) \end{bmatrix}^T \begin{bmatrix} x_g - x_v g \\ y_g - y_v g \end{bmatrix} \tag{1}$$

The GRU in the dynamic branch takes the same fused features as the waypoint branch for the initial hidden state to improve consistency and the inputs include the predicted vehicular command from the waypoint branch. The result then concatenate with the same waypoint bypass, representing the abstract static coarse simulator and then fed to MLP and sigmoid to create adjusted control output.

To determine the suitable vehicular control, we compute them in two different ways. First, we denormalized the summation of the predicted vehicular command from waypoint branch and adjusted control from dynamic branch. Second, we use two separate PID controllers to predict the vehicle controls, one for finding the steering command (lateral) and the other for finding the throttle and brake (longitudinal), using the predicted waypoints and the current speed. Our control policy is similar to [28] that calculates the control commands using both methods based on the scenario. During driving, the vehicle relies on the first two waypoints to calculate its next destination by taking their average. However, we also generate a prediction for a third waypoint, which supplies additional data to the GRU and waypoint prediction layer. Furthermore, this approach enables the MLP and PID agents to receive identical information, since the last biased hidden state contains the details from the second waypoint.

4. Experiments

4.1. Dataset

We use CARLA [13] (0.9.10) for the simulating environment which has 8 available towns for training and testing. We train our model on the 210 GB publicly available dataset by TransFuser[1] for the experiment. All 8 towns were used for training and the dataset includes approximately 2500 routes through junctions with an average length of 100m and around 1000 routes along curved highways with an average length of 400m. To generate data for training purposes, an expert policy is formulated, which employs privileged information obtained from the simulator to control the driving process [4]. The expert's waypoints serve as ground-truth labels for the imitation loss, making the expert comparable to an automatic labeling algorithm. To accomplish lateral control, the expert policy follows the path generated by the A* planner, and a PID controller is used to minimize the angle of the vehicle towards the next waypoint in the route, which is at least 3.5 meters away. Meanwhile, longitudinal control is performed using a version of model predictive control, which differentiates between 3 target speeds. The standard target speed is 4.0 m/s, but the speed is reduced to 3.0 m/s when the expert is inside an intersection. Additionally, if an infraction is predicted, the target speed changes to 0.0 m/s, bringing the vehicle to a halt. The longitudinal and lateral controllers use PID values of $K_p = 5.0, K_i = 0.5, K_d = 1.0$ and $K_p = 1.25, K_i = 0.75, K_d = 0.3$, respectively. For the running average of Both controllers' integral term, we use a buffer of size 40.

Originally, all three RGB images and depth maps are retrieved at a resolution of 480×960 then cropped to 160×320 to avoid distortion. Thus, all three RGB images and depth maps are represented as $R \in 0, \cdots, 255^{3\times160\times320}$ which is the set of 8-bit value in a form of RGB channel $(C) \times height(H) \times width(W)$. Then, the true depth value for each pixel i in the depth map can be decoded by:

$$\mathbb{R}_i^{dec} = \frac{R_i + 256G_i + 256^2 B_i}{256^3 - 1} \times 1000, \tag{2}$$

where $\mathbb{R}_i^{dec}$ is the decoded true depth of pixel i, (R_i, G_i, B_i) are stored 8-bit value of pixel i, 256 is the highest decimal value of 8-bit, and 1000 is the actual depth range of RGB-D camera in meters. Also, each semantic segmentation ground truth is represented as $R \in 0, 1^{23\times160\times320}$ where 23 is the number of classes with value of 1 or 0 showing if the pixel belongs to the class or not. The object classes for the semantic segmentation are according to [28]. Moreover, the waypoints are represented in BEV space with $\omega\rho_i = (xi, yi)_{i=1}^3$. The center $(0, 0)$ of the BEV space (local vehicle coordinate) is on the ego vehicle itself positioned at the bottom-center point. The model estimates the vehicular controls in a normalized range of 0 to 1, then they will be denormalized to their original value with steering $\in [-1, 1]$, throttle $\in [0, 0.75]$, and brake $\in \{0, 1\}$. For the traffic light state prediction, we set the value to 1 if a red light appeared. Meanwhile, speed measurement (in m/s) and GPS locations are sparse, and high-level navigational commands are one-hot encoded.

The first scenario, called 1WN, involves training the model on all available maps and route sets except for the Town05 , which are reserved for validation. The model is then evaluated on both Town05 short and long routes, consisting of 32 short and 10 long routes, to assess its performance. The evaluation is conducted in clear noon condition in a normal situation, and all non-player characters follow the traffic rules. In second scenario, 1WA, the model is tested under abnormal non-player characters (NPC) behavior, which may lead to collisions, such as pedestrians suddenly crossing the street or bicyclists appearing. Addi-

[1] https : / / github . com / autonomousvision / transfuser/tree/2022

tionally, the traffic light manager may intentionally create a state with double green lights at an intersection, simulating emergency situations where an ambulance or firefighter may skip the traffic light. Along with properly driving the ego vehicle, the model is expected to react safely and avoid collisions in these adversarial situations.

4.2. Implementation Details

In order to concurrently acquire knowledge in various tasks through end-to-end learning, it is crucial to establish several loss functions beforehand. For the purpose of semantic segmentation loss function ($\mathcal{L}_{SEG}$), a mixture of binary cross-entropy and dice loss is utilized and can be computed through following equation.

$$\mathcal{L}_{SEG} = \Big(\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\Big) + \Big(1 - \frac{2|\hat{y} \cup y|}{|\hat{y}| + |y|}\Big) \tag{3}$$

where N is the number of pixel elements at the output layer of the semantic segmentation decoder. Then, y_i and $\hat{y}_i$ are the value of ith element of the ground truth y and prediction $\hat{y}$, respectively. Meanwhile, we use a simple L1 loss for the other tasks: traffic light state loss ($\mathcal{L}_{TL}$), stop sign loss ($\mathcal{L}_{SS}$), steering loss ($\mathcal{L}_{ST}$), throttle loss ($\mathcal{L}_{TH}$), brake loss ($\mathcal{L}_{BR}$), velocity loss ($\mathcal{L}_{VE}$) and waypoints loss ($\mathcal{L}_{WP}$). This approach enables us to benefit from both distribution-based and region-based losses simultaneously [29]. Providing supplementary loss criteria to the semantic segmentation task is imperative, since the remaining components of the network structure rely on it. In the case of three predicted waypoints, only ($\mathcal{L}_{WP}$) necessitates averaging. Ultimately, the comprehensive loss that encompasses all tasks can be calculated as follows:

$$\begin{aligned}\mathcal{L}_{TOTAL} = {} & \alpha_1\mathcal{L}_{SEG} + \alpha_2\mathcal{L}_{TL} + \alpha_3\mathcal{L}_{SS} + \\ & \alpha_4\mathcal{L}_{ST} + \alpha_5\mathcal{L}_{TH} + \alpha_6\mathcal{L}_{BR} + \alpha_7\mathcal{L}_{VE} + \alpha_8\mathcal{L}_{WP}\end{aligned} \tag{4}$$

The loss weight for each task is denoted by $\alpha_{1,\cdots,7}$. We utilize the MGN algorithm [29] to adaptively adjust the loss weights for each training epoch. To achieve this, we employ the Adam optimizer with a decoupled weight decay of 0.001, and train the model until it reaches convergence [26]. Initially, the learning rate is set to 0.0001 and gradually halved if the validation metric shows no decline for three consecutive epochs. Furthermore, to prevent unnecessary computational expenses, training is halted if there is no progress for 15 consecutive epochs or reached the maximum of 40 epochs. Our model is implemented using the PyTorch framework [31] and trained on an NVIDIA GeForce RTX-3090 with a batch size of 20.

4.3. Evaluation Metrics

As per the CARLA leaderboard evaluation setting, we have employed the driving score (DS) as our principal metric. The higher the DS value, the more exemplary the driving ability. The DS can be computed using the following:

$$DS = \frac{1}{N_r}\Sigma_{i=1}^{N_r} RC_i IP_i \tag{5}$$

To calculate the driving score (DS) for a given route (DS_i), we use the product of two factors: the percentage of the route that was completed correctly (RC_i) and the corresponding infraction penalty (IP_i). We then compute the average of all DS_i values over the total number of routes (N_r) to obtain the final driving score. To calculate RC_i, we divide the distance driven correctly on the route by the total length of the route. This calculation excludes any incorrect paths taken (e.g., driving on sidewalks). To calculate IP_i, we use the following formula:

$$IP_i = \prod_{j}^{M} (p_i^j)^{\#infractions_j}, \tag{6}$$

where M represents the types of infractions for evaluations, the ideal IP_i at the beginning of the evaluation is 1.0, and it decreases each time an infraction occurs. We consider same penalties for different infractions as [7].

4.4. Baselines

We have opted to compare our metrics with some of the cutting-edge techniques in autonomous driving. The necessary inputs for inference time in each method are presented in the inputs column of the table 1, where RGB-D denotes the RGB camera and depth information, whereas RGB-L represents the RGB camera and LiDAR information. "-F" denotes inputs with only the front sensors, while "-A" signifies inputs of all left, front, and right sensors.

As our first baseline, we have selected X13 [28], which is an end-to-end approach that uses RGB-D data. This algorithm mainly relies on CNN and EfficientNet to extract features for identifying the vehicle's navigational waypoints. We trained two different versions of X13: the first one (X13-F) is identical to the model presented in the paper, and the second one (X13-A) includes all three RGB-D data from left, front, and right sensors. This approach aims to demonstrate the potential of utilizing multiple sensors and facilitate a fair comparison between the two models. Furthermore, we have chosen two versions of the Transfuser method [33]. In the first version called TF-F , the algorithm utilizes a combination of ResNet and transformer architecture to process an RGB image and LiDAR data. In the second version, TF-A, we fed three RGB image and LiDAR data and scale the network to match the input.

Table 1. Performance comparison of LetFuser (ours) with baselines: X13-F/A [28], TF-F/A [34], and Expert

Experiment	Model	Inputs	Normal Clear Noon (1WN)			Adversarial Clear Noon (1WA)		
Town5			DS	RC	IPS	DS	RC	IPS
short	X13-F	1 RGB-D	32.814	68.284	0.468	28.359	58.792	0.480
	X13-A	3 RGB-D	48.833	75.824	0.588	37.263	73.986	0.466
	TF-F	1 RGB-L	17.800	19.864	0.942	23.641	24.373	0.953
	TF-A	3 RGB-L	12.494	16.315	0.886	11.349	14.675	0.843
	Ours	3 RGB-D	**66.012**	**99.717**	0.663	**51.669**	**91.918**	0.574
	Expert	*	99.919	99.919	1.00	79.675	95.349	0.833
long	X13-F	1 RGB-D	8.381	63.633	0.194	7.601	48.114	0.246
	X13-A	3 RGB-D	7.670	48.039	0.291	11.866	52.424	0.456
	TF-F	1 RGB-L	22.456	24.509	0.950	12.964	15.393	0.910
	TF-A	3 RGB-L	7.111	7.351	0.963	8.829	9.001	0.971
	Ours	3 RGB-D	13.943	**63.685**	0.215	**20.584**	47.186	0.493
	Expert	*	60.808	100	0.608	23.344	58.872	0.619

Table 2. Model Specifications

Model	Total Parameters	GPU memory
X13-F	20985934	2920 MB
X13-A	20985934	4958 MB
TF-F	66218754	3898 MB
TF-A	66401154	5015 MB
Ours	31331865	3761 MB

4.5. Results

In this section, as described in Section 4, we evaluate the proposed model in various scenarios. Table 1 presents final results for our method and the baselines. Please note that a higher IP or RC does not necessarily indicate better driving performance. A vehicle may complete all routes and receive a high RC, but drive poorly, resulting in low IP and DS, or vice versa. As can be seen from the results of the Table 1, our method achieved the highest DS scores for both the 1WN and 1WA scenarios, with RC rates of 99.717 and 91.918, respectively, in Town5 short routes. These findings demonstrate both the accuracy and conservativeness of our approach. While X13-F and X13-A performed reasonably well, X13-A showed better performance due to its greater knowledge of the environment. However, TF-F and TF-A both exhibited poor performance in terms of DS and RC. This is likely due to the highly conservative nature of these methods, which cause them to stop frequently during driving, resulting in high IPS but low DS.

Town5's long routes pose a greater challenge, particularly in adversarial scenarios, where rare accidents or infractions can result in reduced DS. Our method was successful in driving properly, but due to limitations in the training dataset, accidents occurred in some cases. The dataset only collected RGB image data from the top of the expert vehicle, which made it difficult to avoid accidents in situations where the ego vehicle was close to the front vehicle and only the top of the vehicle was visible. As a result, the reported numbers are slightly lower than expected. However, our method achieved better metrics compared to the baselines, demonstrating its great ability. TF-F aimed to drive conservatively, which resulted in high DS due to better IPS but lower RC. In contrast, other baselines achieved much higher RC with a lower IPS. Another crucial factor we focused on is developing an end-to-end algorithm that is lightweighted and can be quickly trained with a single advanced GPU. Table 2 illustrates total parameters and GPU memory usage for each baseline as well as our method.

4.6. Ablation studies on task specific learning

We conducted three ablation studies to evaluate effectiveness of different modules. The ablations include the middle 60-degree SDC map section (no side SDC), replacing CvT with EfficientNet (no CvT), and removing vehicular controls (no VC). The results are presented in Table 3. Additionally, we analyzed the model's performance in handling multiple perception and control tasks simultaneously by conducting a comparative study with task-specific models to evaluate their intuitive performance on each task independently. The metrics scoring including binary cross entropy BCE_{SEG} for semantic segmentation, and accuracy Acc_{TL} for traffic light state similar to [28]. Mean absolute error is used to justify the model's performance in predicting ego vehicle speed prediction (MAE_{SP}), waypoints (MAE_{WP}), steering (MAE_{ST}), throttle(MAE_{TH}), and brake (MAE_{BR}), which are the same function used for their loss calculation. We also present the first epoch number with the best validation result ($epoch^*$).

Table 3. Ablation study with task specific metrics

Model	Acc_{TL}	MAE_{SP}	BCE_{SEG}	MAE_{WP}	MAE_{ST}	MAE_{TH}	MAE_{BR}	$epoch^*$
X13-F	0.9846	NA	0.1591	0.0792	0.0173	0.0482	0.0236	30
X13-A	**0.9882**	NA	0.0648	0.07878	0.0195	0.04254	0.01989	19
Ours no side SDC	0.9812	0.2786	0.0647	0.08275	0.02285	0.0531	0.03323	15
Ours no CvT	0.988	**0.13443**	0.0634	0.07307	**0.0173**	0.0510	0.0199	11
Ours no VC	0.9839	0.3051	0.0621	0.0817	0.01879	0.0531	0.02567	19
Ours	0.987	0.2524	**0.0620**	**0.0729**	0.0182	**0.0445**	**0.0185**	21

Table 3 shows that our approach was more efficient compared to X13, as it can achieve its minimum only after 21 epochs. Our approach also outperformed X13 in terms of task-specific performance metrics, with more accurate results for vehicular control and waypoint prediction. Although X13-A had the best traffic light accuracy, our approach provided a more comprehensive solution for autonomous driving. The ablation study showed that removing the side maps in SDC decreased the accuracy of SDC, resulting in less accurate vehicular control and waypoint prediction. Conversely, replacing the CvT with Effnet improved the speed, traffic light prediction, and steering control, but reduced the accuracy of other vehicular control commands that require more in-depth knowledge. Finally, removing the vehicular control module improved waypoint predictions but decreased the performance of vehicular control commands. This is due to the lack of an estimator in the system to learn dynamic behavior in the environment, including other vehicles and traffic light state.

Furthermore, the results of the ablation study highlight the importance of fusing multiple sensor inputs and the use of deep neural network architectures to achieve better performance in autonomous driving tasks. Future research could explore more efficient architectures for autonomous driving tasks, such as utilizing attention mechanisms instead of concatenation or integrating reinforcement learning algorithms to improve decision-making in dynamic environments. Moreover, the integration of explainable AI techniques would enable us to gain a better understanding of the decision-making process and enhance the transparency and interpretability of autonomous systems.

5. Discussions and conclusions

This study introduces an end-to-end deep multi-task learning model that can concurrently manage perception and control tasks learning for an autonomous driving vehicle. The model is designed to address a point-to-point navigation task in which the vehicle is required to follow a predetermined route sequence established by a global planner. The CARLA simulator, featuring four diverse scenarios, is utilized to evaluate the model and examine various driving aspects. Furthermore, a comparative analysis is conducted with recent models to establish the model's performance.

The use of a SDC map plays a crucial role in achieving robust scene understanding, a capability that would otherwise be lost. LiDAR sensors provide data on the height dimension, whereas SDC offers semantic information pertaining to each class on every layer, thus facilitating the acquisition of valuable insights by the model. Additionally, concatenation of RGB and SDC features in the fusion block allows for the autonomous learning of the relationship between the features, thus preventing information loss. By utilizing two agents that represent distinct aspects of driving, the proposed model can generate a wider range of driving options, allowing for greater distances to be covered while achieving a more optimal trade-off between the percentage of completed routes and the incurred infraction penalties. Incorporating traffic light and ego vehicle speed information can enhance the RGB feature extraction process. In order to balance the features extracted from SDC map, RGB, and simulation measurements, we employed batch normalization for more accurate waypoint and control prediction. Also, we incorporated additional metadata, such as command, route, and ego vehicle speed into the model to improve perception. The GRU in dynamic branch predicts required adjustment to vehicular control obtained from waypoint branch, by incorporating dynamic coarse simulator.

The experimental results yielded several noteworthy observations. Firstly, it was determined that all models struggle in adversarial scenarios that necessitate advanced perception and control modules. Secondly, the proposed model achieved the highest driving score (DS) due to its ability to respond effectively to anomalies. The model's performance was further bolstered by its ability to maintain a balance between route completion (RC) and infraction penalty (IP) by leveraging the expertise of two agents to understand various aspects of driving. Thirdly, the proposed model was shown to be more efficient, featuring fewer trainable parameters and utilizing less GPU resources compared to its nearest competitor in each scenario. Finally, augmenting the center camera and depth with left and right sensors as a single image enabled the ego vehicle to broaden its domain knowledge to encompass a perpendicular viewpoint while simultaneously reducing the input-output time, thereby facilitating more efficient training.

References

[1] Pedram Agand, Mahdi Taherahmadi, Angelica Lim, and Mo Chen. Human navigational intent inference with probabilistic and optimal approaches. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8562–8568. IEEE, 2022. 1

[2] Aseem Behl, Kashyap Chitta, Aditya Prakash, Eshed Ohn-Bar, and Andreas Geiger. Label efficient visual abstractions for autonomous driving. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2338–2345. IEEE, 2020. 1, 2

[3] Can Chen, Luca Zanotti Fragonara, and Antonios Tsourdos. Roifusion: 3d object detection from lidar and vision. *IEEE Access*, 9:51710–51721, 2021. 2

[4] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR, 2020. 1, 2, 5

[5] Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, et al. Persformer: 3d lane detection via perspective transformer and the openlane benchmark. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVIII*, pages 550–567. Springer, 2022. 2

[6] Kashyap Chitta, Aditya Prakash, and Andreas Geiger. Neat: Neural attention fields for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15793–15803, 2021. 2

[7] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *Pattern Analysis and Machine Intelligence (PAMI)*, 2022. 1, 6

[8] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338, 2019. 1

[9] Yaodong Cui, Ren Chen, Wenbo Chu, Long Chen, Daxin Tian, Ying Li, and Dongpu Cao. Deep learning for image and point cloud fusion in autonomous driving: A review. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):722–739, 2021. 1

[10] Giuseppe D'Amico, Aldo Amodeo, Ina Mattis, Volker Freudenthaler, and Gelsomina Pappalardo. Earlinet single calculus chain–technical–part 1: Pre-processing of raw lidar data. *Atmospheric Measurement Techniques*, 9(2):491–507, 2016. 1

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3

[12] Nemanja Djuric, Henggang Cui, Zhaoen Su, Shangxuan Wu, Huahua Wang, Fang-Chieh Chou, Luisa San Martin, Song Feng, Rui Hu, Yang Xu, et al. Multixnet: Multiclass multistage multimodal motion prediction. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 435–442. IEEE, 2021. 2

[13] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. 1, 5

[14] Sudeep Fadadu, Shreyash Pandey, Darshan Hegde, Yi Shi, Fang-Chieh Chou, Nemanja Djuric, and Carlos Vallespi-Gonzalez. Multi-view fusion of sensor data for improved perception and prediction in autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2349–2357, 2022. 2

[15] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2020. 1, 2

[16] Angelos Filos, Panagiotis Tigkas, Rowan McAllister, Nicholas Rhinehart, Sergey Levine, and Yarin Gal. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? In *International Conference on Machine Learning*, pages 3145–3153. PMLR, 2020. 1, 2

[17] Chunzhao Guo, Takashi Owaki, Kiyosumi Kidono, Takashi Machida, Ryuta Terashima, and Yoshiko Kojima. Toward human-like lane following behavior in urban environment with a learning-based behavior-induction potential map. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1409–1416. IEEE, 2017. 2

[18] Zhiyu Huang, Chen Lv, Yang Xing, and Jingda Wu. Multimodal sensor fusion-based deep neural network for end-to-end autonomous driving with scene understanding. *IEEE Sensors Journal*, 21(10):11781–11790, 2020. 2

[19] Keishi Ishihara, Anssi Kanervisto, Jun Miura, and Ville Hautamaki. Multi-task learning with attention for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2902–2911, 2021. 2

[20] Bernhard Jaeger. *Expert drivers for autonomous driving*. PhD thesis, Master's thesis, University of Tübingen, 2021. 1, 3, 8, 13, 2021. 2

[21] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8248–8254. IEEE, 2019. 2

[22] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7345–7353, 2019. 2

[23] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 641–656, 2018. 2

[24] Xiaodan Liang, Tairui Wang, Luona Yang, and Eric Xing. Cirl: Controllable imitative reinforcement learning for

vision-based self-driving. In *Proceedings of the European conference on computer vision (ECCV)*, pages 584–599, 2018. 2

[25] Wei Liu, Liyan Ma, Bo Qiu, Mingyue Cui, and Jianwei Ding. An efficient depth map preprocessing method based on structure-aided domain transform smoothing for 3d view generation. *PloS one*, 12(4):e0175910, 2017. 1

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[27] Gregory P Meyer, Jake Charland, Shreyash Pandey, Ankit Laddha, Shivam Gautam, Carlos Vallespi-Gonzalez, and Carl K Wellington. Laserflow: Efficient and probabilistic object detection and motion forecasting. *IEEE Robotics and Automation Letters*, 6(2):526–533, 2020. 2

[28] Oskar Natan and Jun Miura. End-to-end autonomous driving with semantic depth cloud mapping and multi-agent. *IEEE Transactions on Intelligent Vehicles*, 2022. 2, 4, 5, 6, 7

[29] Oskar Natan and Jun Miura. Towards compact autonomous driving perception with balanced learning and multi-sensor fusion. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):16249–16266, 2022. 2, 6

[30] Eshed Ohn-Bar, Aditya Prakash, Aseem Behl, Kashyap Chitta, and Andreas Geiger. Learning situational driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11296–11305, 2020. 1

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[32] Aditya Prakash, Aseem Behl, Eshed Ohn-Bar, Kashyap Chitta, and Andreas Geiger. Exploring data aggregation in policy learning for vision-based urban autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11763–11773, 2020. 1

[33] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7077–7087, 2021. 2, 6

[34] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 7

[35] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 2, 4

[36] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7153–7162, 2020. 1

[37] Gustavo Velasco-Hernandez, John Barry, Joseph Walsh, et al. Autonomous driving architectures, perception and data fusion: A review. In *2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 315–321. IEEE, 2020. 1

[38] Dong Wu, Man-Wen Liao, Wei-Tian Zhang, Xing-Gang Wang, Xiang Bai, Wen-Qing Cheng, and Wen-Yu Liu. Yolop: You only look once for panoptic driving perception. *Machine Intelligence Research*, pages 1–13, 2022. 2

[39] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22–31, 2021. 1, 2, 3

[40] Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. *arXiv preprint arXiv:2206.08129*, 2022. 4

[41] Yi Xiao, Felipe Codevilla, Akhil Gurram, Onay Urfalioglu, and Antonio M López. Multimodal end-to-end autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):537–547, 2020. 1, 2

[42] Zhishuai Zhang, Jiyang Gao, Junhua Mao, Yukai Liu, Dragomir Anguelov, and Congcong Li. Stinet: Spatio-temporal-interactive network for pedestrian detection and trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11346–11355, 2020. 2

[43] Albert Zhao, Tong He, Yitao Liang, Haibin Huang, Guy Van den Broeck, and Stefano Soatto. Sam: Squeeze-and-mimic networks for conditional visual driving policy learning. In *Conference on Robot Learning*, pages 156–175. PMLR, 2021. 1

[44] Brady Zhou, Philipp Krähenbühl, and Vladlen Koltun. Does computer vision matter for action? *Science Robotics*, 4(30):eaaw6661, 2019. 2