IEEE Open Journal of
Intelligent Transportation
Systems

Received 13 June 2023; revised 11 September 2023 and 3 November 2023; accepted 20 November 2023. Date of publication 28 November 2023; date of current version 8 December 2023.

Digital Object Identifier 10.1109/OJITS.2023.3335648

A Multi-Task Vision Transformer for Segmentation and Monocular Depth Estimation for Autonomous Vehicles

DURGA PRASAD BAVIRISETTI[1], HERMAN RYEN MARTINSEN[2], GABRIEL HANSSEN KISS[1], AND FRANK LINDSETH[1]

[1]Department of Computer Science, Norwegian University of Science and Technology, 7034 Trondheim, Norway
[2]Capgemin, 1671 Fredrikstad, Norway

CORRESPONDING AUTHOR: D. P. BAVIRISETTI (e-mail: durga.bavirisetti@ntnu.no)

The work of Durga Prasad Bavirisetti was supported in part by the ERCIM Postdoctoral Fellowship under Contract 2019-36.

ABSTRACT In this paper, we investigate the use of Vision Transformers for processing and understanding visual data in an autonomous driving setting. Specifically, we explore the use of Vision Transformers for semantic segmentation and monocular depth estimation using only a single image as input. We present state-of-the-art Vision Transformers for these tasks and combine them into a multitask model. Through multiple experiments on four different street image datasets, we demonstrate that the multitask approach significantly reduces inference time while maintaining high accuracy for both tasks. Additionally, we show that changing the size of the Transformer-based backbone can be used as a trade-off between inference speed and accuracy. Furthermore, we investigate the use of synthetic data for pre-training and show that it effectively increases the accuracy of the model when real-world data is limited.

INDEX TERMS Vision transformer, monocular depth prediction, autonomous vehicles, segmentation; multi-task.

I. INTRODUCTION

THE RESEARCH and development of autonomous vehicles [1], [12] has gained significant attention in recent years, with big companies such as Tesla, Waymo, and GM investing in new technology. Autonomous vehicles offer numerous benefits, such as reduced road accidents [36], improved traffic efficiency [14], easier accessibility for disabled and elderly people, and lower greenhouse gas emissions [17]. However, reliability is crucial to ensure the safety of human lives.

Autonomous vehicles sense their surroundings using a variety of sensors such as cameras, radar, and LiDAR. While cameras are more affordable and compact, LiDAR sensors are larger and more expensive, thus making it desirable to replace them with cameras.

The rapid advancement in the field of machine learning, due to the availability of large data and powerful computing resources, has made deep learning a practical solution for real-world problems. The field of computer vision, which applies deep learning to visual data, is crucial for autonomous driving as it enables the vehicle to gather information about its surroundings using cameras. Object detection, segmentation, and depth estimation are commonly used techniques for autonomous driving.

In 2017, Vaswani et al. [35] introduced the Transformer, a deep-learning architecture for natural language processing tasks, which achieved remarkable results. Inspired by its success, Dosovitskiy et al. [7] applied the architecture to vision tasks in 2020, leading to the creation of the Vision Transformer, which achieved new state-of-the-art results and caused a stir in the deep learning community. Since then, several new Vision Transformer architectures have been proposed.

This paper explores the use of Vision Transformers for dense prediction tasks in autonomous driving. It focuses on the tasks of monocular depth estimation and semantic segmentation and designs a multitask model that can perform both simultaneously. The effectiveness of the model will be studied through training and evaluation of multiple autonomous driving datasets. To achieve this goal the following research question (RQ)s are studied:

The review of this article was arranged by Associate Editor Xin Xia.

- RQ1. Does a multitask Vision Transformer perform better than models trained for individual tasks?
- RQ2: Can synthetic data enhance model performance when real-world data is scarce?
- RQ3: How accurate are the depth predictions from the model?

The research approach adopted in this paper is experimental, which involves exploring the relationship between a cause and its effect. The experiments will focus on examining the impact of factors such as training methods and backbone choice on performance. The performance of the models will be evaluated using both quantitative and qualitative analysis. Quantitative analysis involves computing accuracy metrics, while qualitative analysis involves visually inspecting individual predictions for insights and anomalies.

The main contributions of the proposed method are as follows:

- Most of the multitask learning methods [25], [30] use convolutional neural networks (CNNs) for joint segmentation and depth estimation. In addition, they are multi-stage methods. Unlike them, our method is a simple single-stage method that takes an image as input and performs the joint segmentation and depth estimation tasks in a single forward pass.
- For this purpose, we designed a hybrid encoding and decoding framework based on Vision transformer variants SegFormer [40] and GLPDepth [18].
- We chose the best model for each task (segmentation and depth estimation) to design a multitask model based on a thorough assessment of their advantages and drawbacks. Our selection process involved a meticulous analysis of the benefits, drawbacks, and feasibility of implementing these models.
- The inference time of our multitask model is less than that of the individual task models, even though it is performing two tasks at a time.
- Our model performed well during evaluation on unseen data (NAP Lab dataset), despite being trained on other datasets. This highlights the generalization capability of our model.
- The quantitative results of our multitask model are comparable to those of individual task models, demonstrating the capability of our model to replace the need for two different models to perform two distinct tasks.
- We show that changing the size of the Transformer-based backbone can be used as a trade-off between inference speed and accuracy.
- We have also investigated the use of synthetic data for pre-training and show that it effectively increases the accuracy of the model when real-world data is limited.

The remainder of the paper has the following organization: Section II, Background and Related Work explains the Vision Transformers and gives an overview of state-of-the-art models. Section III, Methodology, presents the selection of the semantic segmentation and depth estimation models, the functioning of chosen SegFormer and GLPDepth models, and the proposed multitask model. Section IV, Experiments and Results, details the data preparation process, metrics, hardware, and training setup. In addition, it also presents the qualitative and quantitative outcomes of the experiments conducted. Section V, Discussion, examines the results and answers the research questions. Section VI, Conclusion and Future Directions, summarizes the paper, its key findings, and suggests possible directions for future research.

II. BACKGROUND AND RELATED WORK

In this section, segmentation and depth estimation with their evaluation metrics, transformers, vision transformers, and the relevant state-of-the-art Vision Transformer models for semantic segmentation and monocular depth estimation will be presented.

Segmentation is the task of categorizing image pixels into different labels. There are three types of segmentation: semantic, instance, and panoptic. In semantic segmentation, pixels are assigned labels based on what object/structure they are part of (e.g., car, sky, building), without separating instances of the same category. In instance segmentation, each instance of a category is given a separate label, usually only for categories of interest, not the entire image. Panoptic segmentation combines semantic and instance segmentation. This paper focuses on exploring the semantic segmentation task.

Depth estimation is the task of predicting depth in a scene using images. The aim is to determine the distance from the camera to each pixel. There are two main methods for this task: stereo depth estimation and monocular depth estimation. The traditional method, stereo depth estimation, uses two cameras with a known distance between them to find the disparity between matching pixels and calculate depth. Monocular depth estimation, using deep neural networks, is a more recent approach that predicts the depth of each pixel using a single image. Monocular depth estimation can be done using either supervised learning with ground truth depth maps or self-supervised learning without ground truth. This study focuses on the supervised approach for monocular depth estimation.

The Transformer [35] is a deep learning architecture that was introduced in the 2017 paper "Attention is All You Need." It was specifically designed for processing sequential data, such as natural language, and has since become a cornerstone in the field of natural language processing (NLP). Unlike traditional recurrent neural networks (RNNs), the Transformer uses self-attention mechanisms to capture dependencies between elements in a sequence. The architecture of the Transformer consists of multiple identical layers, each composed of two sub-layers: multi-head self-attention and a feed-forward network. The multi-head self-attention mechanism allows the network to weigh the importance of different elements in the sequence. This is achieved by computing attention scores for each element in the sequence with respect to every other element. The feed-forward network

processes the information from the attention mechanism and is composed of two linear transformations followed by a non-linear activation function. The Transformer's self-attention mechanism and feed-forward network work in tandem to process the input sequence and produce an output sequence. The output of the final layer is used to make predictions, such as for language modeling or machine translation. The Transformer's use of self-attention mechanisms and parallel processing of the input sequence has proven to be highly effective, leading to its widespread use in various NLP tasks.

The Vision Transformer (ViT) [7] is a variant of the Transformer architecture that was introduced to tackle computer vision tasks. Unlike traditional convolutional neural networks (CNNs) that use convolutional layers to process image data, the ViT processes image data as sequences of feature vectors extracted from patches of the image. This allows the ViT to take advantage of the Transformer's ability to handle sequential data and process long-range dependencies between patches in an image. The network architecture of the ViT is similar to that of the Transformer, consisting of multiple identical layers, each composed of two sub-layers: multi-head self-attention and a feed-forward network. The multi-head self-attention mechanism allows the network to weigh the importance of different patches in the image and capture the relationships between them. The feed-forward network processes the information from the attention mechanism and is composed of two linear transformations followed by a non-linear activation function. The ViT architecture has been successfully applied to a variety of computer vision tasks, including image classification, object detection, and segmentation. Its ability to process image data as sequences of feature vectors allows it to effectively capture long-range dependencies between patches in an image, resulting in improved performance compared to traditional CNNs.

The following subsections discuss previous studies that employed Vision Transformers for semantic segmentation and monocular depth estimation. In selecting relevant works, three key criteria were taken into account: 1) high accuracy on autonomous driving datasets, 2) efficient, near real-time inference, and 3) significant contribution to the field.

A. RELEVANT STATE-OF-THE-ART VISION TRANSFORMER MODELS FOR SEMANTIC SEGMENTATION

The Segmentation Transformer (SETR) [44] was one of the first attempts to apply Transformers to semantic segmentation. Prior to SETR, the common approach was to use a CNN backbone, which suffered from a limited receptive field and was unable to capture long-range dependencies in images. SETR solved this issue by introducing a Transformer encoder with a global receptive field and combining it with a lightweight convolutional decoder. The resulting model achieved a mean Intersection over Union (mIoU) score of 82.15 on the Cityscapes dataset.

In May 2021, Xie et al. introduced SegFormer [40], a simple and efficient encoder-decoder design for semantic segmentation using Transformers. SegFormer's Transformer encoder is inspired by ViT [7] and has multiple modifications to enhance performance for semantic segmentation. It has a hierarchical structure, outputs multi-scale feature maps, and utilizes the efficient self-attention calculation method from PVT [37], which improves processing speed for high-resolution images. The decoder is lightweight and consists of only MLP layers. There are 6 different model sizes, with the smallest suitable for real-time applications and the largest achieving high accuracy on the Cityscapes and ADE20K datasets with a mIoU of 84.0 on the Cityscapes dataset. The model also performs well on corrupted cityscapes data, indicating its robustness and potential suitability for safety-critical tasks such as autonomous driving.

Traditionally, semantic and instance segmentation were approached as two distinct tasks, where semantic segmentation was viewed as a per-pixel classification task, and instance segmentation as a mask classification task. However, Cheng et al. proposed that the tasks could be combined and solved by a single model - the MaskFormer [4]. The architecture employs mask classification to produce semantic segmentation predictions, incorporating a Transformer decoder that predicts binary masks with respective class labels. The MaskFormer was tested with various backbones, but Swin Transformer [27] provided the best results. However, the results were limited to the Cityscapes dataset using a ResNet backbone and achieved a mIoU of 81.4.

In December 2021, Cheng et al. [3] improved upon the MaskFormer architecture with their new Mask2Former. They made several improvements to increase performance and simplify training, such as implementing masked attention in the Transformer decoder to limit attention calculations to local features, using multi-scale high-resolution features to detect small objects, and making small changes to improve performance and save training memory. Mask2Former achieved a mIoU of 84.5 on the Cityscapes dataset.

In December 2021, Jain et al. introduced SeMask [16], a solution to address the issue of the encoder struggling to capture the semantic context of the image in Vision Transformers for semantic segmentation. The architecture implements a semantic layer that follows the Transformer layer at every stage of the encoder, consisting of multiple SeMask blocks that apply semantic attention operation to feature maps. The SeMask block can be integrated with any hierarchical Vision Transformer and when used with the Swin Transformer [27] and SegFormer [40], the best-performing model achieved a mIoU of 84.98 on the Cityscapes dataset.

Yan et al. introduced the Lawin Transformer [42], a Transformer-based architecture for semantic segmentation, in January 2022. They address the issue of current Vision Transformers not producing contextual information at multiple scales, which affects performance and efficiency. The Lawin Transformer introduces a novel decoder, the large

window attention spatial pyramid pooling (LawinASPP) [42], which queries a larger area of the feature map to produce multi-scale contextual information. The decoder can be combined with any hierarchical Vision Transformer encoder and when combined with the SegFormer or Swin Transformer encoder, the Lawin Transformer shows improved accuracy and lower computational cost compared to the original SegFormer or achieves a mIoU of 84.4 on the Cityscapes dataset.

B. RELEVANT STATE-OF-THE-ART VISION TRANSFORMER MODELS FOR MONOCULAR DEPTH ESTIMATION

AdaBins [10] is a monocular depth estimation architecture introduced in November 2020 by Bhat et al. It uses a CNN encoder-decoder design with a Transformer-based building block named AdaBins to achieve good results. Depth estimation is treated as a classification task where the depth range is split into adaptive bins of varying size, determined using a mini-ViT architecture. The final prediction uses a linear combination of the bin centers to avoid sharp depth discontinuities. The AdaBins architecture achieved an absolute relative error of 0.058 on the KITTI dataset.

In March 2021, Ranftl et al. proposed the Dense Prediction Transformer (DPT) [31], a novel architecture for the tasks of monocular depth estimation and semantic segmentation. DPT adapts a traditional encoder-decoder design, replacing the CNN backbone with the ViT architecture [7]. The ViT backbone provides a global receptive field and preserves the initial feature map resolution to capture finer details. DPT is pre-trained on a massive monocular depth estimation dataset and fine-tuned on the KITTI dataset to achieve an absolute relative error of 0.062. When trained for semantic segmentation, the model achieves a mIoU of 49.02.

Kim et al. introduced GLPDepth [18], a Transformer-based architecture and training strategy for monocular depth estimation that considers both the global and local context of the image. It uses SegFormer encoder [40] to capture global dependencies and a lightweight decoder with skip connections to integrate local information. A new depth-specific augmentation technique is also introduced to improve performance, achieving an absolute relative error of 0.057 on the KITTI dataset.

The DepthFormer architecture [20] for monocular depth estimation was proposed in March 2022 by Li et al. The authors stressed the importance of capturing both global and local information from an image and achieved this by using two separate encoder branches - one using the Swin Transformer [27] to model long-range correlations and the other using convolutions for local information. The information from both branches is combined using a novel Hierarchical Aggregation and Heterogeneous Interaction (HAHI) module. The model achieved an absolute relative error of 0.052 on the KITTI dataset, which is a state-of-the-art result.

In April 2022, Li et al. introduced BinsFormer [21], a new architecture for monocular depth estimation. It is based on AdaBins with multiple modifications, including a Transformer decoder to enhance adaptive bin generation and a multi-scale design. During training, an auxiliary scene classification task is used to improve the model performance. BinsFormer achieves state-of-the-art performance with an absolute relative error of 0.052 on the KITTI dataset.

C. RELEVANT STATE-OF-THE-ART ON MULTI-TASK LEARNING

Multi-task Learning involves training shared parameters for multiple tasks to improve efficiency and accuracy by mining latent information. Prominent models in this field include Mask R-CNN [13], which combines Faster R-CNN [32] for instance segmentation and object detection. Eigen and Fergus [8] address depth prediction, surface normal estimation, and semantic labeling, while MultiNet [34] handles classification, detection, and semantic segmentation in a single model. In their study [43], the authors introduced a model that employs a multi-task learning approach to concurrently predict both the steering angle and speed command. YOLOP [39] uses CSPDarknet as its base and specializes in object detection, drivable area segmentation, and lane detection. Standley et al. [33] and Liu et al. [26] have improved multi-task training by grouping related tasks together rather than training all tasks simultaneously. In the research conducted by Liang and their colleagues [23] they devised a multitask model capable of handling object detection, semantic segmentation, and drivable area segmentation concurrently. In a separate investigation by the same team [22], they introduced the VE-Prompt framework. VE-Prompt utilizes visual exemplars to offer task-specific visual cues, effectively guiding the model in learning various tasks including object detection, semantic segmentation, instance segmentation, and lane line prediction. Furthermore, in a different study outlined in [41], Xu et al. presented a multitask learning framework based on Transformers. This framework enables the simultaneous execution of segmentation, depth estimation, and saliency detection tasks within a single model.

In their study [30], Mousavian et al. presented an approach for tackling both semantic segmentation and depth estimation tasks using deep convolutional neural networks (CNNs). Their investigation centered on the feasibility of training distinct components of the model for each task, followed by fine-tuning the entire integrated model to simultaneously enhance performance in both tasks through a single loss function. Additionally, the researchers enhanced their deep CNN by integrating a fully connected conditional random field (CRF) to boost the overall model's performance. The model's evaluation encompassed both the training and evaluation phases, conducted on the NYUDepth V2 dataset.

In another paper called "Collaborative Deconvolutional Neural Network (C-DCNN)" [25], the authors introduced an approach that jointly addresses semantic segmentation

and depth estimation as pixel-wise labeling problems. The training process involves three stages: firstly, pretraining two separate hierarchical supervised DCNNs for each task; secondly, integrating these networks through a pointwise bilinear operation to allow simultaneous learning of both tasks and finetuning; and finally, applying a fully connected Conditional Random Field (CRF) to further improve semantic segmentation performance using predicted depth and semantic labels. This model was also evaluated on the NYUDepth V2 dataset.

In both of the methods referred to in citations [25], [30], CNN-based architectures were employed. In contrast, our approach introduces a hybrid encoding and decoding framework based on Vision transformer variants SegFormer [40] and GLPDepth [18]. Unlike the methods mentioned in [25], [30], our proposed method does not require separate pretraining, finetuning, and CRF steps. Our method is a straightforward single-stage approach that takes an image as input and simultaneously performs the joint segmentation and depth estimation tasks in a single forward pass. While these previous methods reported their performance solely on the NYUDepth V2 dataset, our work presents results across three publicly available datasets: Cityscapes [5], KITTI-360 [24], and Apollo Synthetic Datasets [15], as well as on our custom NAPLab dataset. We validated the multi-task learning capabilities of our vision transformer model on our proprietary dataset, demonstrating strong performance during evaluation on previously unseen NAP Lab data, despite being trained on a different dataset. Additionally, we investigated the utility of synthetic data for pre-training and observed its effective impact on increasing model accuracy when real-world data is limited.

III. METHODOLOGY

This section presents the selection of models for the experiments in the paper and provides a detailed description of the chosen models and the proposed Multitask Model. This information is important for understanding the results and validity of the conclusions.

A. SELECTION OF MODELS

The objective of this paper is to create a multitask Vision Transformer for segmentation and depth estimation by integrating two existing models. This section examines each model and provides a justification for its selection or non-selection. The accuracy and efficiency of the models were taken into account during the decision-making process.

1) SELECTION OF SEMANTIC SEGMENTATION MODEL

The Segmentation Transformer (SETR) was an early successful attempt to use Transformers for semantic segmentation and influenced other Transformer-based models. However, it is no longer considered the best option due to advancements in computer vision and improved models with higher accuracy and efficiency.

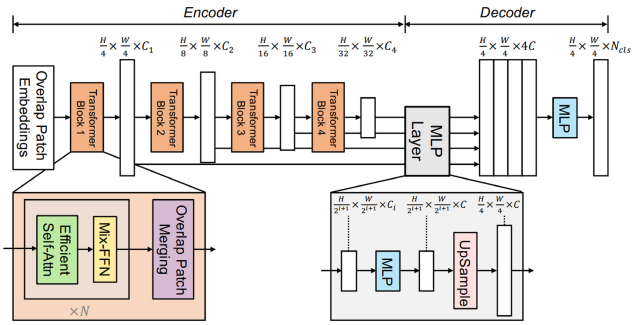


FIGURE 1. SegFormer Architecture (Image source: Xie et al. [40]).

MaskFormer and Mask2Former are intriguing models for this study due to their ability to carry out multiple segmentation tasks, and the innovative approach of using mask classification to solve semantic segmentation. Mask2Former, with its multiple improvements over MaskFormer, is considered the most promising option. However, its heavy decoder hinders its ability to perform real-time segmentation, leading to its non-selection for the experiments.

SeMask is a semantic segmentation model that improves performance by incorporating a small, task-specific layer into the Transformer encoder. Although efficient, the modifications are specific to semantic segmentation and may not transfer to other dense prediction tasks like monocular depth estimation. Therefore, SeMask will not be used in the experiments aimed at designing a multitask model with a shared encoder.

The Lawin Transformer was planned to be used in the experiments for semantic segmentation as it promised to increase accuracy and lower computational cost through its efficient decoder design. However, the results from integration with the official SegFormer code did not match the results reported in the paper. Hence, it was decided not to further explore the Lawin Transformer in this study.

The SegFormer architecture features an efficient self-attention calculation and a lightweight all-MLP decoder, making it suitable for real-time applications. Additionally, it offers 6 different backbone sizes, allowing for a balanced trade-off between speed and accuracy. Furthermore, its robustness on corrupted data makes it a strong candidate for safety-critical tasks like autonomous driving. These advantages make SegFormer the chosen model for segmentation in this study.

2) SEGFORMER

The SegFormer is a semantic segmentation model that uses an encoder-decoder design. The encoder is a hierarchical Vision Transformer inspired by ViT [7] with modifications for improved performance, while the decoder consists of MLP layers only. An overview of the architecture can be found in Figure 1.

1) *Transformer Encoder:* The SegFormer encoder takes an input image of size $H \times W \times 3$ and splits it into patches of size 4×4. These patches are then

processed through four Transformer blocks to create hierarchical feature maps. The Transformer blocks are improved versions of those used in ViT, with a reduced complexity of the multi-head self-attention operation using the sequence reduction process proposed by Wang et al. [37] with a reduction ratio R. The positional encoding used in ViT is replaced by a 3×3 convolution and an MLP, providing the model with positional information and better performance for images with varying resolutions during inference. At the end of each Transformer block, an overlapped patch merging process is applied to reduce the spatial resolution and produce hierarchical feature maps. The final feature maps have a resolution of $\frac{H}{2^{i+1}} \times \frac{W}{2^{i+1}} \times C_i$, where i refers to the current Transformer block. The feature maps are then sent to the decoder to generate the predicted segmentation mask. The SegFormer architecture has a total of 6 Transformer encoders, B0 to B5, with different sizes and the same design. B0 is the smallest model and suited for real-time applications, while B5 is the largest model with the highest accuracy.

2) *MLP Decoder:* The Segformer decoder is a simple and lightweight design, consisting only of MLP layers. Despite its simplicity, the decoder still yields high-quality predictions, thanks to the large effective receptive field provided by the hierarchical Transformer encoder which helps capture the global context of the image. The decoder follows these 4 steps to make a prediction:
   a) Each feature map F_i is transformed using an MLP layer, converting the individual channel dimension C_i to a common channel dimension C.
   b) The feature maps are upsampled to $\frac{H}{4} \times \frac{W}{4}$ and concatenated, resulting in a feature set F with a channel dimension of $4C$.
   c) F is then fused using another MLP layer, reducing the channel dimension from $4C$ to C.
   d) The final segmentation mask M is generated by passing the fused features through another MLP layer, resulting in a resolution of $\frac{H}{4} \times \frac{W}{4} \times N_{cls}$, where N_{cls} is the number of classes.

3) SELECTION OF DEPTH ESTIMATION MODEL

In this study, the focus is on investigating the application of Vision Transformers for dense prediction tasks. The depth estimation task is approached as a classification problem with AdaBins, which divides the depth range into adaptive bins. Although AdaBins achieved impressive results with a CNN backbone and limited use of Transformer architecture, it was not used in the experiments as the objective is to examine models with greater utilization of Transformers.

The DepthFormer model was not used in the experiments either, despite having two branches (Transformer-based and CNN-based) for gathering both global and local information from the image and producing remarkable results on the KITTI dataset. The lack of availability of official code at the time of experimentation made incorporating the model difficult and time-consuming.

BinsFormer, one of the best models for monocular depth estimation on the KITTI dataset, could not be utilized in the experiments due to the unavailability of the paper and official code at the time of experimentation.

The GLPDepth was chosen as the monocular depth estimation model for this study due to its lightweight decoder that effectively merges global and local contexts to generate accurate depth predictions. Its lightweight design also makes it suitable for real-time use and its compatibility with SegFormer, the selected segmentation method, allows for easy integration into a multitask model. The compatibility with SegFormer was the main reason for choosing GLPDepth over other methods for monocular depth estimation.

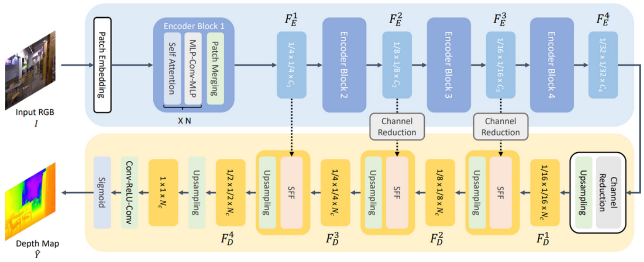


FIGURE 2. The architecture of GLPDepth (Image source: Kim et al. [18].)

4) GLPDEPTH

This section explains the functioning of GLPDepth, the selected depth estimation model. It merges the hierarchical Vision Transformer encoder used in SegFormer with a unique lightweight decoder for monocular depth estimation. The model's performance is further improved by the introduction of Vertical Cut-Depth, a depth-specific augmentation technique. A visual representation of the architecture is provided in Figure 2.

1) *Transformer Encoder:* The GLPDepth architecture incorporates the SegFormer encoder by Xie et al. [40] to generate hierarchical feature maps from the input image for the decoder. The original implementation uses the SegFormer B4 backbone for depth prediction, but the experiments will test smaller backbone sizes to evaluate their impact on performance.
2) *Lightweight Decoder:* The GLPDepth decoder takes the final feature map F_E^4 from the encoder and processes it through a channel reduction and bilinear upsampling step, resulting in a feature map of size $\frac{1}{16}H \times \frac{1}{16}W \times N_C$. The feature map then passes through multiple consecutive layers that include Selective Feature Fusion (SFF) modules and bilinear upsampling. The SFF module combines global and local features from the current decoder feature map and the encoder feature map, creating rich feature

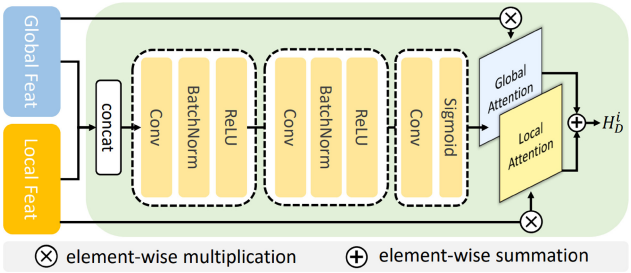


FIGURE 3. The GLPDepth Selective Feature Fusion (SFF) module (Image source: Kim et al. [18].)

maps. After multiple SFF layers and upsampling, the decoder feature map is transformed to $H \times W \times N_C$. To produce the final depth prediction, the feature map is sent through two convolutional layers and a sigmoid function, producing a depth map that is multiplied by the maximum depth value of the dataset to obtain the predicted distance in meters. The SFF module is depicted in Figure 3.

3) *Vertical CutDepth:* The GLPDepth model introduces a new augmentation method named Vertical CutDepth, which is specifically designed for depth estimation and inspired by CutDepth. The method improves the diversity of the dataset by placing a random crop of the ground truth depth map into the RGB image. Unlike CutDepth, Vertical CutDepth does not crop the depth map vertically to better preserve the important vertical information. This method is specific to depth estimation and its effects on other dense prediction tasks are unknown. However, it was not used in the experiments for this study as the goal was to design an architecture for both depth estimation and segmentation.

In short, the proposed method uses SegFormer for semantic segmentation and GLPDepth for monocular depth estimation. Both models use the same Transformer backbone and have lightweight decoders suited for real-time use. The architecture of each model is explained already in Sections III-A2 and III-A4, and their combination into a multitask model is detailed in the next section.

5) PROPOSED MULTITASK MODEL

The multitask model performs monocular depth estimation and semantic segmentation, using SegFormer and GLPDepth. The model can generate predictions for both tasks with a single forward pass, resulting in improved accuracy and efficiency.

1) *Architecture:* The multitask architecture leverages the SegFormer encoder, to obtain hierarchical features from the input image. These feature maps are then directed to two individual decoders, one for semantic segmentation and the other for monocular depth estimation. These decoders generate the ultimate predictions for each task. A visualization of the proposed architecture can be seen in Figure 4, with further elaboration on the segmentation and depth decoders found in Section III-A2 and III-A4 sections, respectively.

2) *Loss Function:* The model is trained to perform both semantic segmentation and monocular depth estimation simultaneously by employing two individual loss functions, specific to each task. The cross-entropy loss, a commonly used method, is utilized for the segmentation task.

$$L_{CE} = -\sum_{i}^{n} y_i \log(\hat{y}_i) \tag{1}$$

For the segmentation task, the cross entropy loss uses the ground truth label y_i and the softmax probability for the i-th class $\hat{y}_i$ to calculate the loss. For the depth estimation task, the scale-invariant log scale loss, previously used in GLPDepth, is employed.

$$L_{SI} = \frac{1}{n}\sum_{i}^{n} d_i^2 - \frac{1}{2n^2}\left(\sum_{i}^{n} d_i^2\right) \tag{2}$$

The scale-invariant log scale loss calculates the difference between the log of the ground truth depth value y_i and the log of the predicted depth value $\hat{y}_i$ for each pixel ($d_i = \log y_i - \log \hat{y}_i$). During the training process, the two task-specific loss functions are combined into a single loss function to calculate the total loss.

$$L = L_{CE} + L_{SI} \tag{3}$$

IV. EXPERIMENTS AND RESULTS

This section provides an overview of the software, hardware, training configurations used for the proposed method, and other methods used for comparison. It covers the various experiments conducted and the datasets used in the study. The section also presents an analysis of the experimental results.

A. DATASETS

In our study, we conducted experiments using four datasets: Cityscapes [5], KITTI-360 [24], Appolo Synthetic [15] and NAPLab dataset (Our own data). The specifics of each dataset, such as the selection of training and testing data, the number of classes, and the class types for both RGB and depth images of each dataset are discussed in the subsequent subsections. It is to be noted that due to the limited annotated frames in the NAPLab dataset, the models cannot be trained or fine-tuned on it. However, we used this data to evaluate the performance of the models trained on the above-mentioned data sets.

1) CITYSCAPES

The Cityscapes dataset [5] includes 5,000 frames with fine annotations. Out of these, one can use 2,975 frames for training, 500 for validation, and 1,525 for testing.

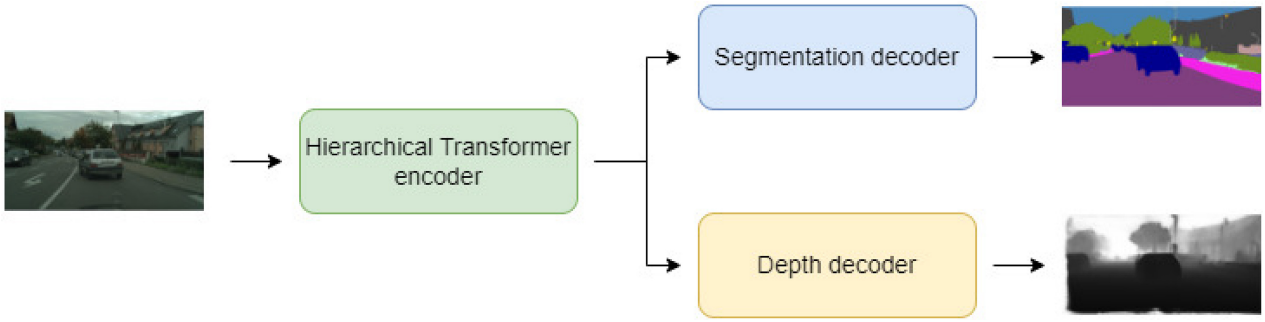


FIGURE 4. The proposed multitask architecture.

However, the annotations for the test set are not available for use during experimentation, leaving 3,475 frames for training and validation. Each frame comes with RGB images captured by the left front-facing camera, annotations for semantic segmentation, pre-computed disparity maps, and camera parameters (both intrinsic and extrinsic). Blurred RGB images are also included for visualization purposes.

The Cityscapes dataset contains high-resolution images (2048x1024), but it is not possible to perform real-time inference with the selected models using these images. To make real-time inference feasible, the images can be resized to a smaller resolution (1024x512), which is half of the original size.

To use disparity maps for training, they must be converted to depth maps through some calculations. The first step is to calculate the disparity values from the raw images using an equation:

$$d = \frac{float(p) - 1.0}{256.0} \tag{4}$$

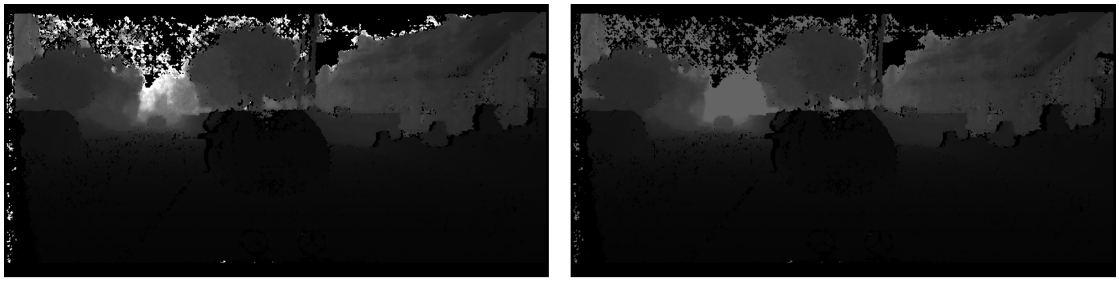
where the calculation of disparity values d from raw pixel values p is done. The depth values are calculated with some intrinsic and extrinsic camera parameters using another equation as:

$$D = \frac{B \times f_x}{d} \tag{5}$$

where the disparity value (d), baseline (B), and focal length (fx) are used to calculate the depth value (D) in meters through the specified equation. The resulting depth maps range from 0 to approximately 470 meters, however, it was noticed that the high depth values were noisy and inaccurate, as shown in Figure 5a. Despite testing the depth maps for training, the evaluation metrics failed to converge as expected. As a result, it was decided to clip the depth maps at 100 meters, meaning values greater than 100 meters would be set to 100. This decision was made as most of the significant objects in the image fall within this range, as seen in the clipped depth map in Figure 5b. The Cityscapes dataset is labeled with 30 semantic classes, but only 19 of them are typically used for training and validation because the other classes are rare. These 19 commonly used classes with their corresponding colors are listed in Table 1. To train using the segmentation data, images with values corresponding to the train IDs of the classes are generated using the official Cityscapes scripts.

TABLE 1. Cityscapes classes and color palette.

Train ID	Name	Color
0	Road	
1	Sidewalk	
2	Building	
3	Wall	
4	Fence	
5	Pole	
6	Traffic Light	
7	Traffic Sign	
8	Vegetation	
9	Terrain	
10	Sky	
11	Person	
12	Rider	
13	Car	
14	Truck	
15	Bus	
16	Train	
17	Motorcycle	
18	Bicycle	



(a) Depth **(b)** Depth clipped

FIGURE 5. Cityscapes depth map before and after clipping at 100m.

2) KITTI-360

The KITTI-360 dataset [24] includes raw perspective images, 2D semantic labels, raw LiDAR scans, vehicle poses, and intrinsic and extrinsic camera parameters. The dataset includes 61,280 annotated images that are divided into a train set and a validation set. The train set has 49,004 frames, while the validation set has 12,276 frames.

The KITTI-360 dataset has a resolution of 1408x376, which is low enough to allow real-time inference but the image height of 376 is not divisible by 32, a requirement of the chosen models. To resolve this, 24 pixels are cropped from the top of the image resulting in a resolution of 1408x352. During validation, a different crop will be used as proposed by Garg et al. [11].

The raw LiDAR point clouds must be transformed into 2D depth maps for training purposes. The transformation is done using the official KITTI-360 scripts and involves applying a transformation matrix to all the points to convert the LiDAR coordinate frame to the camera coordinate frame, which is calculated using an equation.

$$T_{L\to k} = T_{0\to k} \times T_{L\to 0} \tag{6}$$

here, $T_{L\to 0}$ transformation matrix represents the process of transforming 3D points from the LiDAR coordinate frame to the left perspective camera. The $T_{0\to k}$ matrix, which transforms the left perspective camera to other cameras, is set to the identity matrix as only the left perspective camera will be used for the experiments. The projected points in the camera coordinate frame are then transformed into 2D depth maps through the use of intrinsic camera parameters. These maps have a depth range of 0 to 80 meters.

The semantic classes for KITTI-360 align with those utilized for Cityscapes, as listed in Table 1. As a result, KITTI-360 will also use the same 19 classes for training and evaluation. Images with train IDs for the semantic classes must be generated to train the models.

3) APOLLO SYNTHETIC DATASET

The dataset [15] consists of 273,000 frames in total. To improve the balance between classes and scenes, a subset of the dataset was selected for use in the experiments.

The dataset, originally consisting of 273,000 frames, was reduced to 45,235 frames after only half of the available highway sequences from the dataset were used due to data imbalance, and excluding 125 frames due to missing depth annotations. To create a balanced dataset, the frames were split into a training set and a validation set based on the virtual scene level, with the *Road_Loop_with_Intersections* scene used for validation and the remaining scenes for training. The final training set has 40,195 frames and the validation set has 5,040 frames.

The images in the Apollo Synthetic Dataset have a resolution of 1920x1080, but to perform real-time inferences, the resolution needs to be reduced. A common approach is to decrease the resolution to 960x540, which is half the original size. However, due to the design of the chosen models, the image size must be divisible by 32, so the final resolution selected is 960x544.

The depth values are decoded from the depth images using the following equation:

$$D = \left(R + \frac{G}{255.0}\right) \times 655.36 \tag{7}$$

where, the depth value in meters, D, is calculated by using the normalized float values of the red and green channels, R and G, of a pixel in the image. This calculation produces depth values ranging from 0 to 655.35 meters with a precision of 1 cm. Despite the high accuracy of the depth maps, the depth range is clipped to 200 meters, as is done with the Cityscapes and KITTI-360 datasets. This larger max depth value is made possible due to the accuracy of the depth maps in the Apollo Synthetic Dataset.

The Apollo Synthetic Dataset has 24 classes for semantic segmentation that were modified to match the classes in Cityscapes, resulting in 14 classes that will be used in the experiments. These classes are shown in Table 2 and have a color palette similar to Cityscapes.

4) NAPLAB

The NAPLab dataset comprises 10 frames that feature semantic segmentation annotations exclusively. The resolution of the images is 1920 × 1080, however, to facilitate real-time inference, they have been resized to 960 × 544.

TABLE 2. Apollo synthetic dataset classes and color palette.

Train ID	Class Name	Original Classes	Color
0	Road	Road	
1	Sidewalk	Sidewalk	
2	Building	Building	
3	Pole	Pole, Street Light	
4	Traffic Light	Traffic Light	
5	Traffic Sign	Traffic Sign	
6	Vegetation	Vegetation	
7	Terrain	Terrain	
8	Person	Pedestrian	
9	Car	Coupe, SUV, Hatchback, Van	
10	Truck	Truck, Pickup Truck	
11	Bus	Bus	
12	Motorcycle	Motorcyclist	
13	Bicycle	Cyclist	

The semantic segmentation class definitions in the dataset match those of Cityscapes (Table 1).

B. METRICS

Evaluation of segmentation methods can be performed using various metrics. Pixel accuracy, which is calculated as the percentage of image pixels labeled correctly, is a simple method but can provide misleading results if the class distribution in the image is unbalanced. A more reliable evaluation metric is the Intersection over Union (IoU) or Jaccard index.

The recent research in monocular depth estimation uses metrics introduced by Eigen et al. [9]. This overview covers the commonly used evaluation metrics for comparing predicted and actual values. The absolute error shows the difference between the two but does not take into account the size of the actual value, leading to potential issues when comparing values of varying magnitudes. To remedy this, the absolute relative error divides the absolute error by the actual value to incorporate the magnitude. The absolute relative error, squared relative error, and root mean squared error are often used to evaluate model performance. Threshold accuracy measures the percentage of image pixels that have a maximum ratio between predicted and actual values below a specified threshold, with common threshold values being 1.25, 1.252, and 1.253.

C. HARDWARE

The experiments in this paper required significant computational resources and were carried out using two solutions: the IDUN cluster and a virtual machine provided by NTNU. The IDUN cluster at NTNU is a large computing cluster for research purposes, equipped with A100, V100, and P100 GPUs, which were utilized for training the models. Training multiple GPUs was necessary due to the high memory consumption of the models. The virtual machine from NAPLab with an A10 virtual GPU with 23 GB RAM and a graphical user interface was used for ease of development, benchmarking, inference, and visualization of results.

D. TRAINING SETUP

The multitask model is trained on three datasets selected specifically for this purpose. The number of training epochs varies depending on the size of each dataset: 300 epochs for Cityscapes, 30 epochs for the Apollo Synthetic Dataset, and 20 epochs for KITTI-360. During training, the Adam optimizer [19] is utilized with a learning rate of 1.0×10^{-4} and a batch size of 12. Augmentations are performed using the Albumentations library [2], including techniques such as *HorizontalFlip*, *RandomBrightnessContrast*, *RandomGamma* and *HueSaturationValue*. The model's initial weights are from ImageNet [6], pre-trained for optimal performance.

E. RESULTS

The results of the experiments conducted for this study are presented in this section. The following experiments were carried out to answer the RQs posed in Section I:

1) Multitask Training Comparison - The multitask training method was compared with individual task training for segmentation and depth to examine its impact on model performance.
2) Backbone Size Analysis - The multitask model was trained with different backbone sizes to evaluate how the choice of backbone affects performance.
3) Synthetic Data Pre-training - The multitask model was pre-trained on a synthetic dataset to improve its accuracy.
4) Depth Prediction Validation - The accuracy of the predicted distances to objects in images was evaluated.
5) Evaluation on NAPLab Data - The multitask model trained on Cityscapes was tested on NAPLab data.

1) MULTITASK TRAINING COMPARISON

The aim of the experiment was to determine the effect of multitask training on performance. The approach involved training the model for both semantic segmentation and monocular depth estimation simultaneously. Three models were trained: one for segmentation only, one for depth estimation only, and one for both tasks. The models were evaluated on the Cityscapes, KITTI-360, and Apollo Synthetic datasets using the SegFormer B2 backbone for all models.

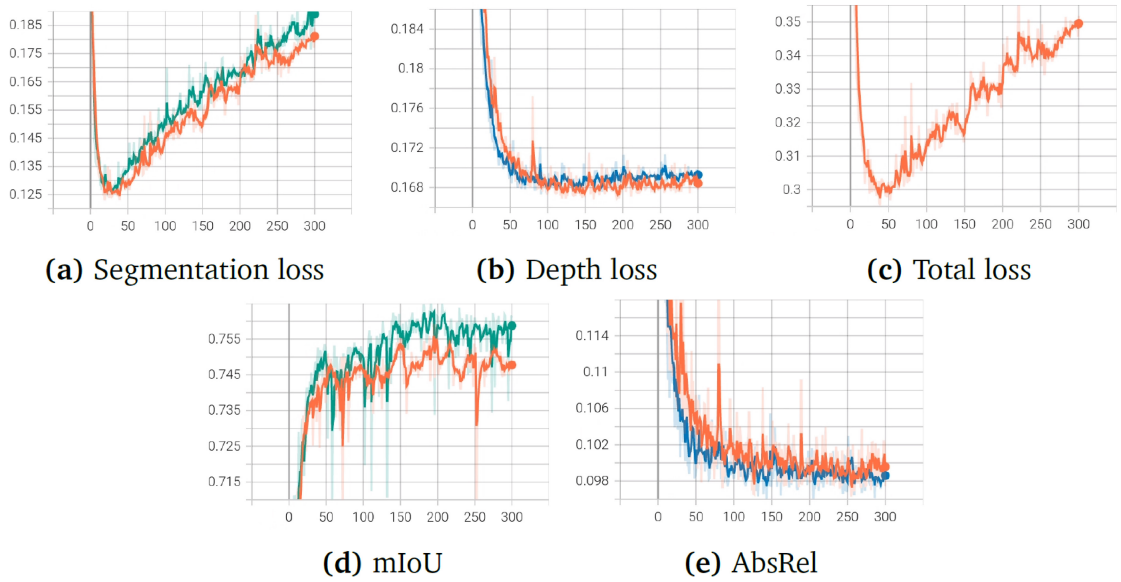


FIGURE 6. Comparison of Multitask Training on Cityscapes dataset. The graphs illustrate the validation loss and evaluation metrics during training, where orange represents multitask, blue represents depth, and green represents segmentation.

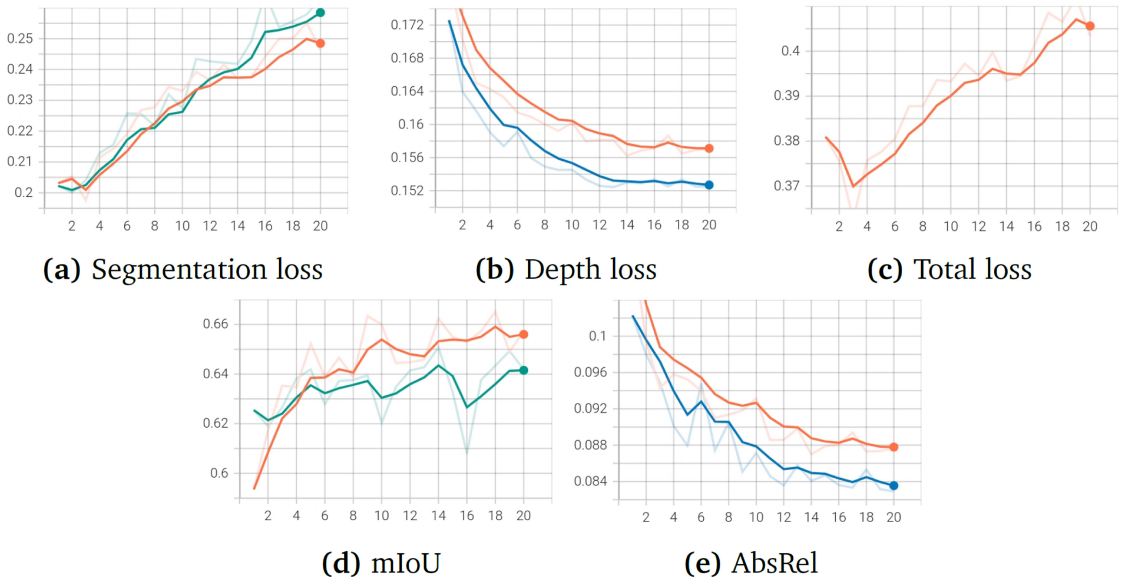


FIGURE 7. Comparison of Multitask Training on KITTI-360 dataset. The graphs illustrate the validation loss and evaluation metrics during training, where orange represents multitask, blue represents depth, and green represents segmentation.

1) *Training:* The models were trained based on the instructions in Section IV The training process of the models was monitored using graphs for three datasets: Cityscapes, KITTI-360, and Apollo Synthetic Dataset which can be found in Figures 6, 7, and 8 respectively. In all the datasets, the segmentation loss began to increase after a few training epochs, which is a commonly observed sign of overfitting. Despite this, the mean Intersection over Union (mIoU) continued to improve. On the other hand, the depth loss showed a smoother convergence pattern compared to the segmentation loss throughout the training process. Comparing the convergence of the individual task models and the multitask model, it can be seen that both models generally converge similarly and no significant differences are apparent from the training graphs.
2) *Qualitative Analysis:* The qualitative results for the models applied to Cityscapes, KITTI-360, and Apollo Synthetic Dataset are presented in Figures 9, 10, and 11, respectively. The multitask model produced similar results to the individual task models, and only the predictions of the multitask model are shown. The model performed well in generating convincing

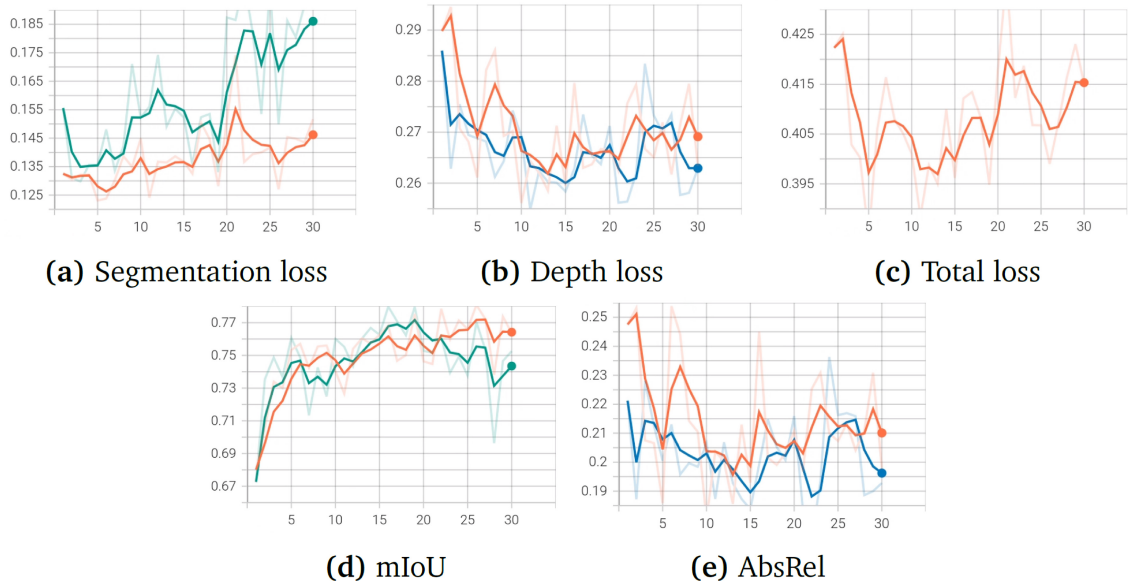


FIGURE 8. Comparison of Multitask Training on Apollo Synthetic Dataset. The graphs illustrate the validation loss and evaluation metrics during training, where orange represents multitask, blue represents depth, and green represents segmentation.

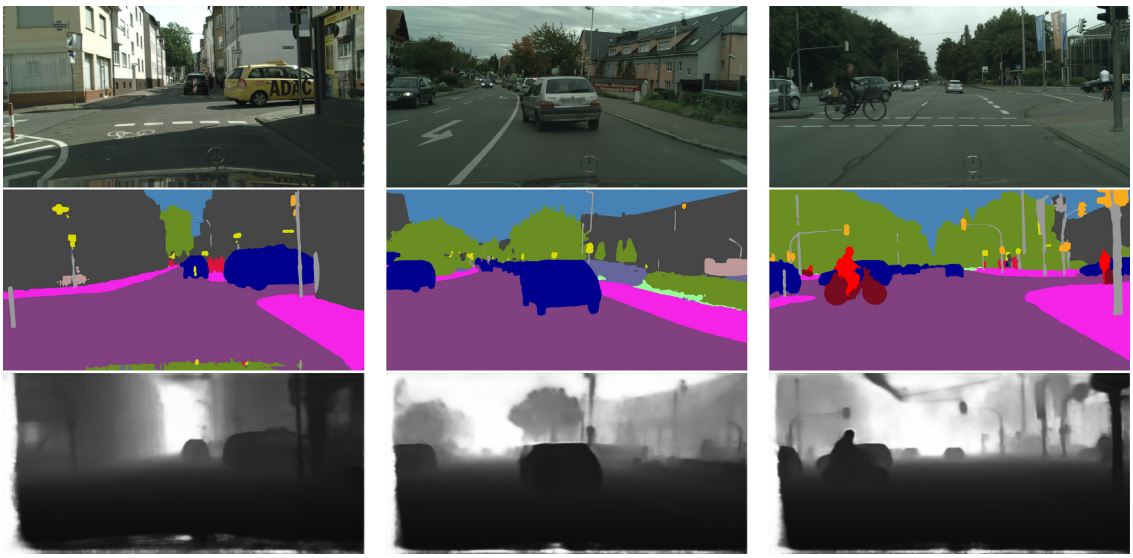


FIGURE 9. Qualitative analysis of Multitask Training on Cityscapes dataset. The results shown were generated using the B2 multitask model and are presented in three rows. The first row displays the original image, the second row shows the predicted segmentation mask, and the third row displays the predicted depth map.

predictions across all datasets. However, the model faced challenges in segmenting the sky region in the Apollo Synthetic Dataset as it was not labeled in the ground truth annotations. The model also struggled to generate accurate depth predictions for the top part of the KITTI-360 images as the ground truth depth maps only covered the bottom part of the images.

3) *Quantitative Analysis:* The quantitative results are displayed in Table 3. The individual task model performed similarly or better than the multitask model in terms of absolute relative error for depth, with only small differences. For segmentation, the individual task model showed the best performance on Cityscapes while the multitask model performed better on the other two datasets. The inference speed was comparable across all datasets, as the images had a similar number of pixels to process. Despite performing an additional task, the multitask model had a similar inference speed to the segmentation model.

2) BACKBONE SIZE ANALYSIS

The purpose of the experiment is to examine the impact of the size of the Transformer backbone on the model performance. Three sizes of the SegFormer backbone, B0, B2, and B4, are used in combination with the multitask model and trained and evaluated on Cityscapes, KITTI-360, and Apollo Synthetic Dataset.

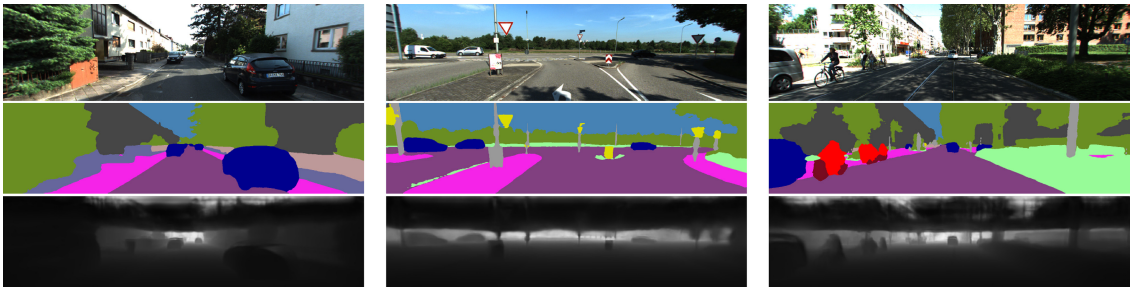


FIGURE 10. Qualitative analysis of Multitask Training on KITTI-360 Dataset. The results shown were generated using the B2 multitask model and are presented in three rows. The first row displays the original image, the second row shows the predicted segmentation mask, and the third row displays the predicted depth map.

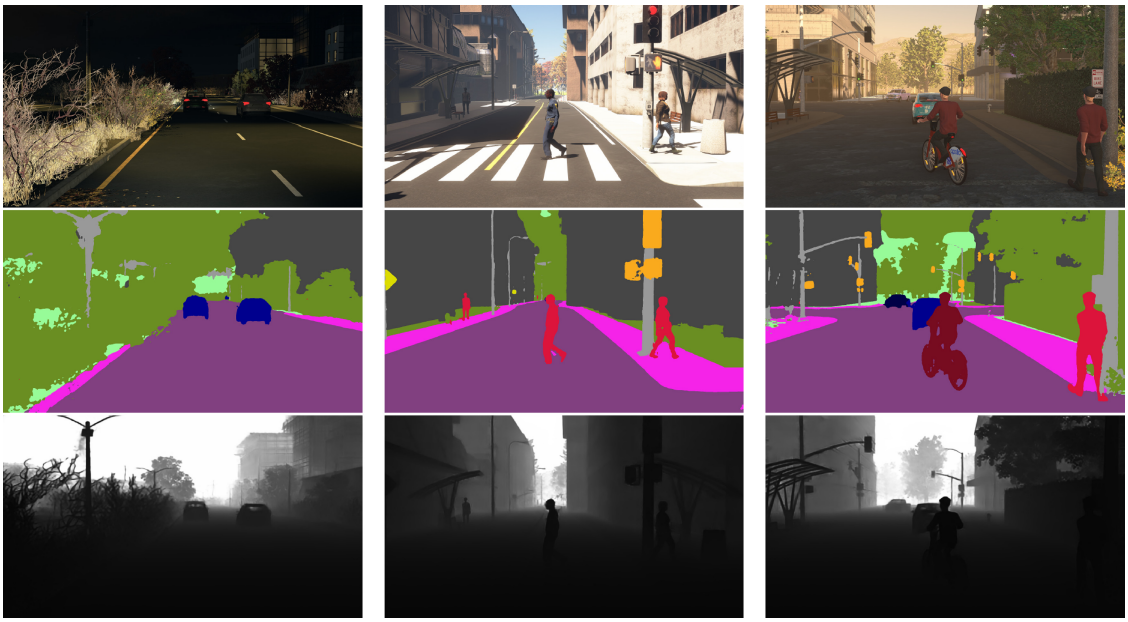


FIGURE 11. Qualitative analysis of Multitask Training on Apollo Synthetic Dataset. The results shown were generated using the B2 multitask model and are presented in three rows. The first row displays the original image, the second row shows the predicted segmentation mask, and the third row displays the predicted depth map.

TABLE 3. Quantitative comparison of multitask training.

Model	FPS	AbsRel	SqRel	RMSE	RMSElog	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	mIoU	mAcc	aAcc
Cityscapes											
Depth only	30	**0.096**	**1.213**	**6.289**	**0.162**	0.904	**0.977**	0.991	-	-	-
Seg only	21	-	-	-	-	-	-	-	**76.53**	**83.98**	**95.87**
Multitask	18	**0.096**	1.266	6.317	**0.162**	**0.905**	**0.977**	**0.992**	75.93	83.54	95.80
KITTI-360											
Depth only	32	**0.083**	**0.372**	**2.905**	**0.147**	**0.912**	**0.972**	**0.989**	-	-	-
Seg only	21	-	-	-	-	-	-	-	65.07	73.33	**93.39**
Multitask	19	0.087	0.386	2.938	0.150	0.907	**0.972**	**0.989**	**66.52**	**74.23**	93.37
Apollo Synthetic Dataset											
Depth only	30	**0.173**	6.993	18.550	**0.275**	**0.792**	**0.902**	**0.954**	-	-	-
Seg only	21	-	-	-	-	-	-	-	77.98	**85.23**	94.55
Multitask	18	0.181	**6.047**	**17.190**	0.277	0.783	0.897	**0.954**	**78.09**	84.59	**94.88**

1) *Training:* The experiment tested three different sizes of the SegFormer backbone (B0, B2, and B4) with the multitask model on Cityscapes, KITTI-360, and Apollo Synthetic Dataset. The training graphs for each dataset can be seen in Figure 12, 13, and 14. The segmentation loss for all backbone sizes started increasing after a few training epochs, which is a sign of overfitting. However, the problem was less prominent for the B0 backbone, which had a slower convergence rate. The depth loss converged better and improved for a

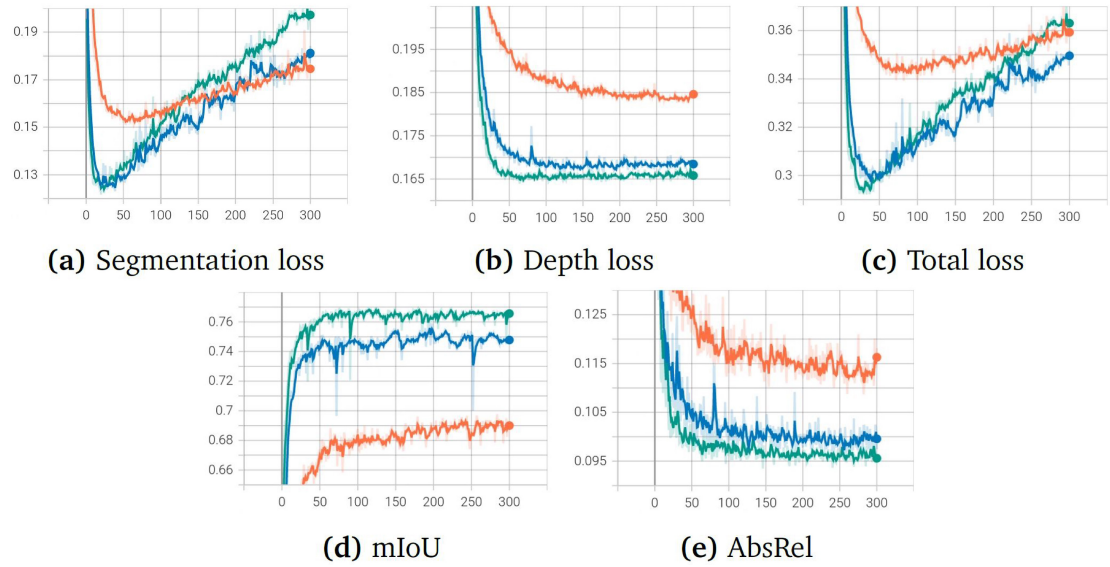


FIGURE 12. Comparison of different Backbone sizes on Cityscapes dataset. The graphs depict the validation loss and evaluation metrics while the models were being trained. The orange color represents the B0 backbone, blue represents the B2 backbone, and green represents the B4 backbone.

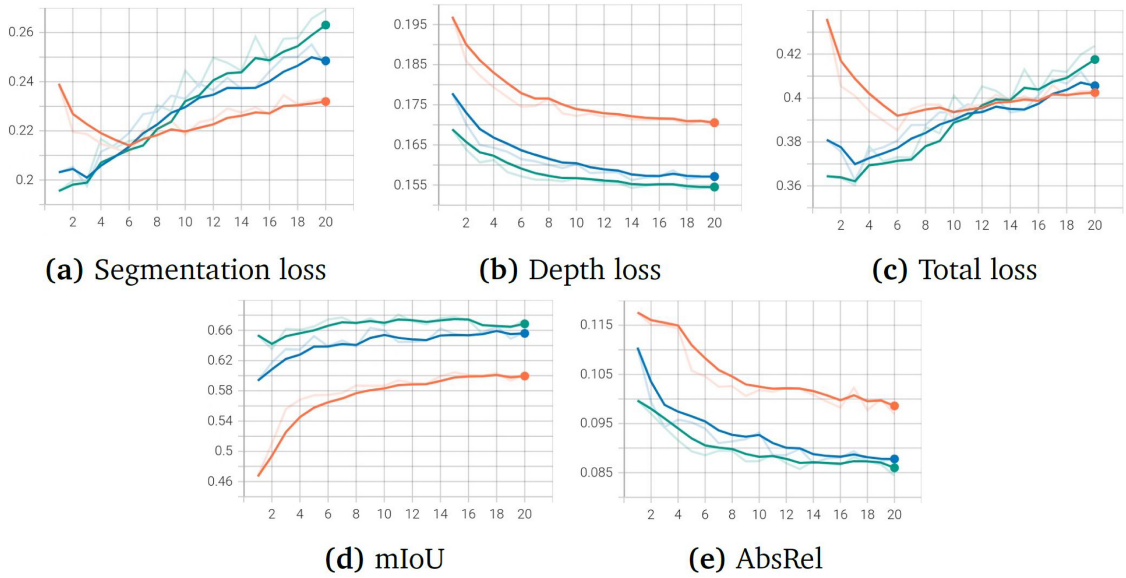


FIGURE 13. Comparison of different Backbone sizes on KITTI-360 dataset. The graphs depict the validation loss and evaluation metrics while the models are being trained. The orange color represents the B0 backbone, blue represents the B2 backbone, and green represents the B4 backbone.

longer period of time. In terms of accuracy, the B4 backbone performed the best, B2 the second best, and B0 performed the worst.

2) *Qualitative Analysis:* The results of the segmentation and depth tasks are shown in Figures 15 and 16 respectively. The two largest models in the experiment perform better in semantic segmentation, showing better distinction between the road and sidewalk and accurately segmenting small objects and thin structures. In-depth estimation, the largest models produce clearer depth maps and detect small objects and thin structures more effectively. The results are shown for the Cityscapes, KITTI-360, and Apollo Synthetic Datasets and compare the original images with the results from Model B0, B2, and B4.

3) *Quantitative Analysis:* As depicted in Table 4, the model featuring the biggest backbone typically achieves the best accuracy for both tasks, with the exception being depth estimation on the Apollo Synthetic Dataset, where Model B2 outperforms Model B4. In terms of computational speed, Model B0 stands out with its impressive 56-61 FPS performance, which is well within the realm of real-time processing. Model B2 also operates close to real-time with a frame rate of 18-19 FPS, while the more robust Model B4 operates at 11-12 FPS.

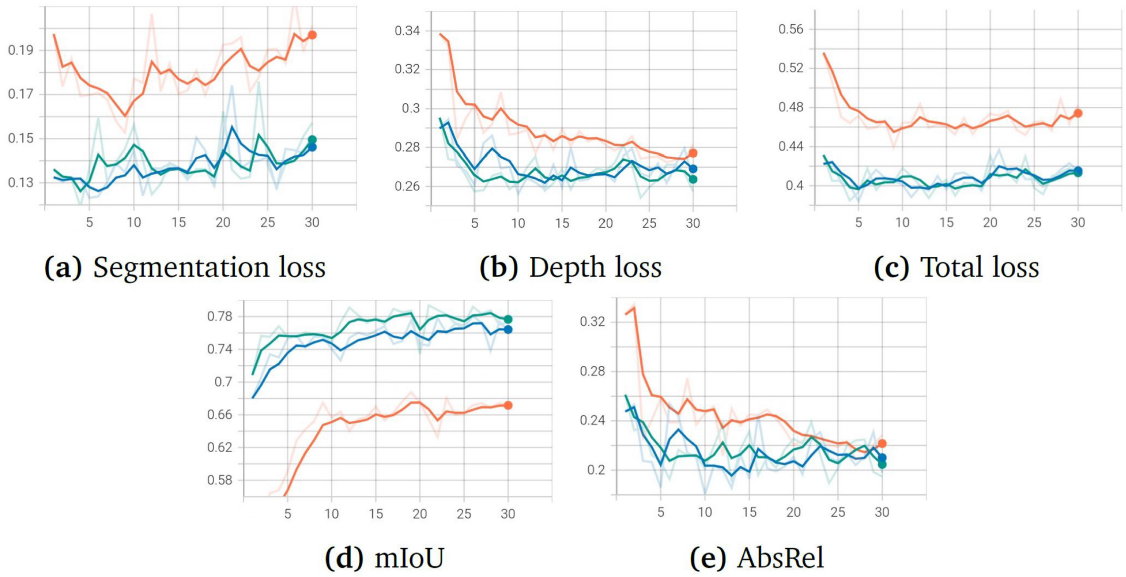


FIGURE 14. Comparison of different Backbone sizes on Apollo Synthetic Dataset. The graphs depict the validation loss and evaluation metrics while the models are being trained. The orange color represents the B0 backbone, blue represents the B2 backbone, and green represents the B4 backbone.

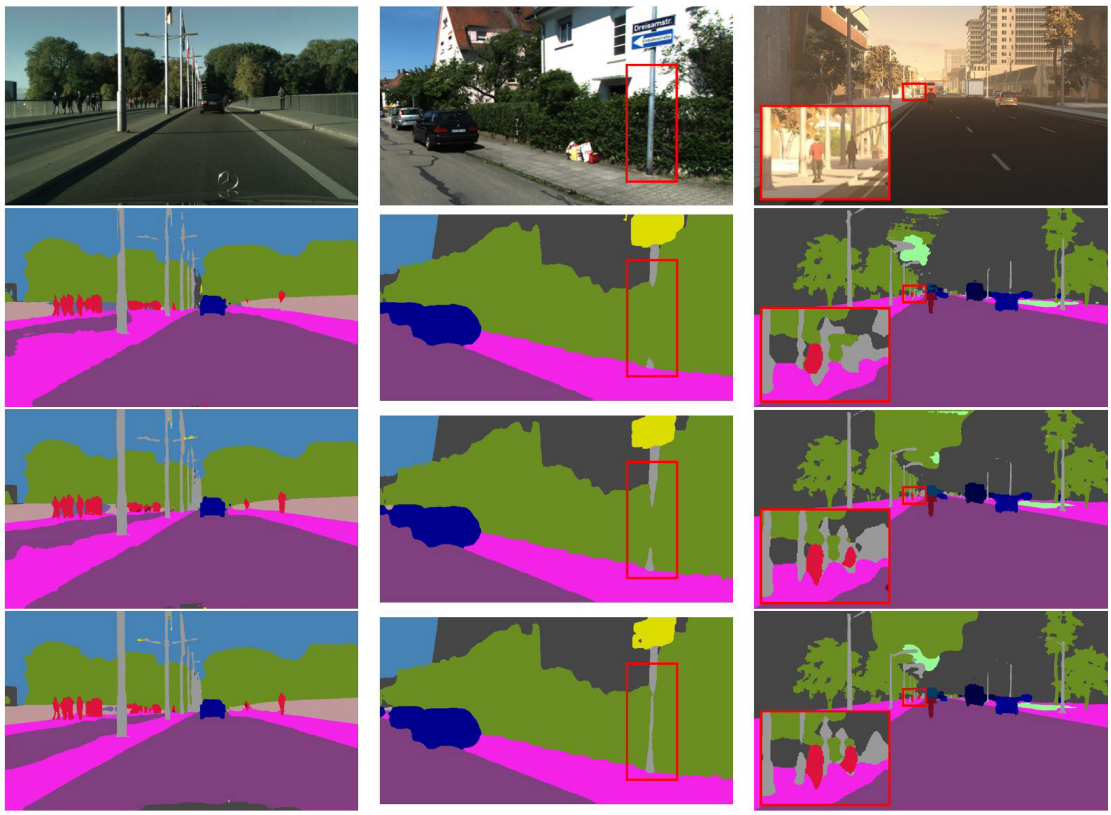


FIGURE 15. The results of the comparison of different Backbone sizes for the segmentation task can be seen in the first, second and third columns of the images, which represent Cityscapes, KITTI-360, and Apollo Synthetic Dataset, respectively. The first row shows the original image, while the second, third, and fourth rows show the results obtained using the B0, B2, and B4 backbones, respectively. The images have been cropped for better visual clarity.

3) SYNTHETIC DATA PRE-TRAINING

This section explores the impact of pre-training on a large synthetic dataset. Specifically, we pre-trained a multitask model that uses a SegFormer B2 backbone on the Apollo Synthetic Dataset and then fine-tuned it on Cityscapes. We compared the model's performance to that of one trained exclusively on Cityscapes.

1) *Training:* In accordance with section IV, the models were trained, and Figure 17 shows their training graphs. Pre-training on the synthetic dataset improved

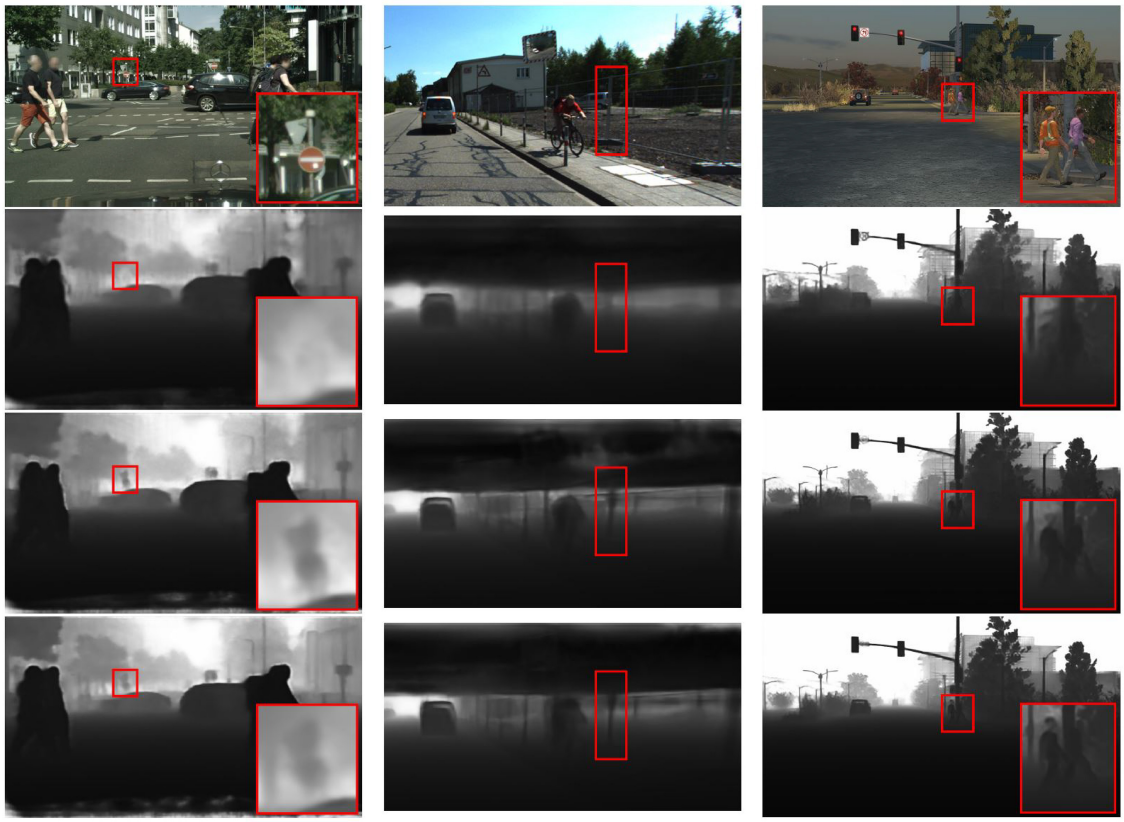


FIGURE 16. This experiment evaluates the qualitative results of the depth estimation task, which are shown through images cropped for better visualization. The images are from the Cityscapes, KITTI-360, and Apollo Synthetic Dataset, with the original image in the first row, and the results from B0, B2, and B4 models in the subsequent rows.

TABLE 4. Backbone size analysis: Quantitative results.

Backbone	FPS	AbsRel	SqRel	RMSE	RMSElog	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	mIoU	mAcc	aAcc
Cityscapes											
B0	57	0.109	1.435	6.999	0.177	0.876	0.969	0.990	69.78	78.34	94.85
B2	18	0.096	1.266	6.317	0.162	0.905	0.977	**0.992**	75.93	83.54	95.80
B4	12	**0.093**	**1.263**	**6.251**	**0.160**	**0.909**	**0.978**	**0.992**	**76.94**	**84.83**	**95.94**
KITTI-360											
B0	61	0.097	0.440	3.172	0.163	0.889	0.967	0.987	60.44	69.27	92.69
B2	19	0.087	0.386	2.938	0.150	0.907	**0.972**	**0.989**	66.52	74.23	93.37
B4	12	**0.084**	**0.377**	**2.925**	**0.149**	**0.909**	**0.972**	**0.989**	**68.08**	**75.75**	**93.58**
Apollo Synthetic Dataset											
B0	56	0.208	7.096	17.750	0.298	0.761	0.881	0.945	68.76	77.16	92.93
B2	18	**0.181**	**6.047**	**17.190**	**0.277**	**0.783**	**0.897**	0.954	78.09	84.59	**94.88**
B4	11	0.190	6.960	18.390	0.278	0.775	0.896	**0.955**	**79.37**	**85.80**	94.69

both the loss and absolute relative error for depth estimation, but there was no observed improvement for semantic segmentation. Unfortunately, the model pre-trained on synthetic data had to be stopped after 150 epochs, but there is potential for better performance with a longer training period. However, given that the last phase of training typically sees only slight improvements, the difference in performance would likely be negligible.

2) *Results:* Table 5 presents the quantitative results, showing that the model pre-trained on synthetic data outperforms the original model in all-depth evaluation metrics. However, there is no observed improvement in semantic segmentation with pre-training on the synthetic dataset. The models with and without synthetic dataset pre-training had no significant differences in their predictions, so there are no qualitative results provided for this experiment.

4) DEPTH PREDICTION VALIDATION

This section presents the validation of depth prediction, which assesses the model's capability to estimate the distance

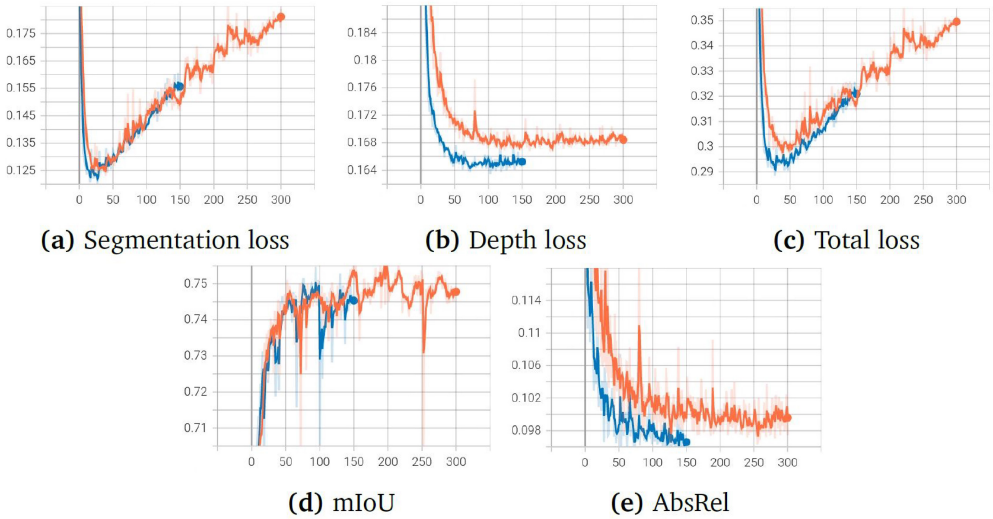


FIGURE 17. The graphs depict the validation loss and evaluation metrics of training with and without pre-training on synthetic data. The orange line represents training without pre-training, while the blue line represents training with pre-training.

TABLE 5. Quantitative results of pre-training on synthetic data.

Cityscapes										
Pre-trained	AbsRel	SqRel	RMSE	RMSElog	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	mIoU	mAcc	aAcc
No	0.096	1.266	6.317	0.162	0.905	0.977	**0.992**	**75.93**	83.54	95.80
Yes	**0.094**	**1.185**	**6.227**	**0.159**	**0.908**	**0.978**	**0.992**	75.48	**83.64**	**95.83**

between objects in an image. Six images, three from Cityscapes and three from KITTI-360 are used, as shown in Figure 18. The multitask model produces a depth prediction for each image, and the predicted depth of an object, marked with a red circle, is compared to the ground truth depth using the average of multiple depth values in the same region. Two different versions of the multitask model are used, one trained on Cityscapes and one on KITTI-360, enabling an assessment of how accurately the model performs on the training dataset as well as on new and unseen data.

Table 6 displays the results of the depth validation. The results demonstrate that the model's predicted depth is the most precise for the dataset it was trained on, with a prediction error of less than a meter. However, when the model is tested on an unfamiliar dataset, the prediction error is significant, and it seems that the Cityscapes model overestimates the distance on the KITTI-360 dataset, while the KITTI-360 model underestimates the distance on the Cityscapes dataset.

TABLE 6. The results of the depth validation are presented, where the estimated distance to individual objects in images from Cityscapes and KITTI-360 datasets are evaluated, as depicted in Figure 18. Two models are used for evaluation, one trained on Cityscapes and the other trained on KITTI-360. The distance values are expressed in meters.

Cityscapes				
Trained on	Image	Ground truth	Prediction	Error
Cityscapes	1	9.26	9.50	0.24
	2	12.31	11.67	0.64
	3	35.09	36.40	1.31
KITTI-360	1	9.26	6.47	2.79
	2	12.31	6.64	5.67
	3	35.09	18.45	16.64
KITTI-360				
KITTI-360	1	4.83	4.31	0.52
	2	17.13	16.57	0.57
	3	34.14	36.75	2.61
Cityscapes	1	4.83	9.31	4.48
	2	17.13	26.90	9.76
	3	34.14	67.89	33.76

5) EVALUATION ON NAPLAB DATA

In this section, the performance of the multitask model on the NAPLab dataset is evaluated. Due to the limited annotated frames in the dataset, the model cannot be trained or fine-tuned on it. Instead, an evaluation is carried out using a model trained on the Cityscapes dataset. Experimental setup in Section IV-E2 is followed, where three different model sizes (B0, B2, and B4) are tested.

Table 7 displays the quantitative results, as no ground truth depth data is available, only evaluation metrics for semantic segmentation are presented. The highest mIoU score is achieved by the largest model, while the smallest model achieves the lowest score, similar to the results of Section IV-E2. Figure 19 provides qualitative results where the predicted segmentation masks look similar to the ground truth masks.

(a) Image 1 (b) Image 2 (c) Image 3

FIGURE 18. The validation of depth involves verifying the predicted distance to an object in each image (see Table 6 also for better understanding.). Objects selected for validation are highlighted with a red circle, with the Cityscapes dataset presented in the first row and the KITTI-360 dataset in the second row.

TABLE 7. Quantitative results on NAPLab data.

Backbone	FPS	mIoU	mAcc	aAcc
B0	56	50.49	66.49	92.94
B2	18	55.36	**74.12**	94.15
B4	11	**59.97**	73.73	**94.36**

V. DISCUSSION

The section examines the implications of the experimental outcomes on the suitability of the model for use in autonomous driving. Additionally, it considers the limitations of this study and endeavors to address the research questions in Section I.

6) THE POTENTIAL APPLICATIONS IN THE FIELD OF AUTONOMOUS DRIVING

Real-time performance is an important requirement for deep learning architectures in autonomous vehicles, with a frame rate of approximately 30 FPS considered real-time. In Section IV-E2, different backbone sizes were tested and the B0 model achieved a frame rate of about 60 FPS, while the B2 model achieved roughly 20 FPS. However, experiments were conducted on powerful GPUs that may not be available in an autonomous vehicle, and additional time is required for data loading, post-processing, and decision-making. TensorRT can optimize the model for fast inference and may help increase inference speed. In addition to inference speed, accuracy is also important in autonomous driving, and the largest models (B2 and B4) outperformed B0 in Section IV-E2. To increase performance, the resolution of input images can be increased to capture finer details, but this may decrease the frame rate. Therefore, different image sizes should be tested to find a balance between accuracy and inference speed.

A. THE LIMITATIONS OF THE STUDY

In this section, the limitations of the study are discussed, as well as some possible solutions to address them.

1) EARLY OVERFITTING

Sections IV-E1 and IV-E2 showed that the segmentation loss in the model increased early in the training process, indicating overfitting, but the model's accuracy continued to improve. This unexpected outcome may be due to the cross-entropy loss used for segmentation, which is calculated differently from accuracy. The lack of augmentation techniques used during training could be another reason for overfitting, and alternative techniques that work for both depth estimation and semantic segmentation could be explored. The hyperparameters were not extensively tested during training, and the learning rate was identified as a crucial parameter that could be optimized through the use of a learning rate scheduler or manual decrease of the learning rate after the initial training epochs. The use of individual learning rates for each task-specific decoder could also help achieve better accuracy for both depth and segmentation.

2) FEW AVAILABLE DATASETS

The study focuses on using a multitask model for monocular depth estimation and semantic segmentation. Finding datasets annotated for both tasks was challenging as most publicly available datasets focus on one task. Three datasets were chosen, all of which contain annotations for both tasks but have their own issues. Cityscapes is small and its depth maps are not as accurate as LiDAR depth maps, while KITTI-360 has a much larger dataset but sparse depth maps. The lack of annotated real-world data led to the use of synthetic data for the evaluation of the multitask model. The chosen dataset, Apollo Synthetic Dataset, has a large number of annotated frames but cannot substitute real-world data completely due to domain shift. Semi-supervised and self-supervised methods were not explored in the study, but they could be interesting to look into as they require less annotated data. However, annotated data for both tasks would still be necessary to evaluate the model's performance.

For more theoretical background, additional experiments, and in-depth explanation of the proposed study, please refer to our work [28].

VI. CONCLUSION AND FUTURE DIRECTIONS

The study explored the use of Vision Transformers for dense prediction tasks in autonomous driving. The goal was to develop a multitask model that can perform monocular depth estimation and semantic segmentation simultaneously while being able to operate in real-time. A literature review was conducted to identify state-of-the-art Vision Transformers, and two models, SegFormer and GLP-Depth, were selected

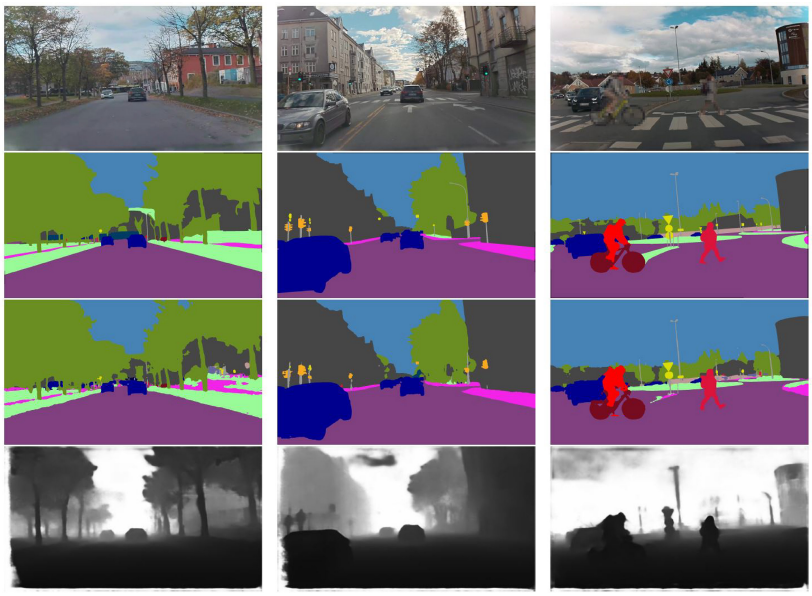


FIGURE 19. The qualitative results of NAPLab data, where pedestrians and license plates are blurred for anonymity purposes. The first row displays the original image, followed by the ground truth segmentation mask in the second row, the predicted segmentation mask in the third row, and the predicted depth map in the fourth row.

for the proposed multitask architecture. The effectiveness of the proposed model was evaluated with three different datasets and compared to individual task models. The results demonstrated that the multitask model achieves comparable accuracy to the individual task models and significantly lowers the total inference time for both tasks. The choice of backbone was found to be able to control the ratio between inference speed and accuracy. An experiment was conducted to investigate the impact of pre-training the model on a large synthetic dataset, which improved depth estimation accuracy significantly while maintaining segmentation accuracy. However, the model struggled on new and unseen datasets when estimating depth accurately.

The potential research ideas in Vision Transformers can be explored to further improve our work. One idea is to explore semi-supervised or self-supervised learning approaches, which require less annotated data during training and can save time and money. Another idea is to investigate the use of mixing multiple depth estimation datasets during training to improve the model's performance across multiple datasets without additional fine-tuning. This approach could force the model to rely on more general features and not features specific to a single dataset.

Another interesting direction to explore is multi-sensor fusion. As the technology continues to evolve, multi-sensor fusion [29], [38] is likely to play a central role in the advancement of automated driving systems. Extensive hyperparameter analysis for the enhanced performance of the proposed method can also be conducted. These ideas could be interesting to investigate further in future research.

REFERENCES

[1] J. Betz et al., "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 458–488, 2022.

[2] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, p. 125, Feb. 2020.

[3] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proc. CVPR*, 2021, pp. 1–10.

[4] B. Cheng, A. G. Schwing, and A. Kirillov. "Per-pixel classification is not all you need for semantic segmentation." 2021. [Online]. Available: https://medium.com/@HannaMergui/maskformer-per-pixel-classification-is-not-all-you-need-for-semantic-segmentation-1e2fe3bf31cb

[5] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding." 2016. [Online]. Available: https://arxiv.org/abs/1604.01685

[6] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[7] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. ICLR*, 2020, p. 6.

[8] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2650–2658.

[9] D. Eigen, C. Puhrsch, and R. Fergus. "Depth map prediction from a single image using a multi-scale deep network." 2014. [Online]. Available: https://arxiv.org/abs/1406.2283

[10] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," 2020, *arXiv:2011.14141*.

[11] R. Garg, V. K. BG, G. Carneiro, and I. Reid, "Unsupervised CNN for single view depth estimation: Geometry to the rescue," 2016, *arXiv:1603.04992*.

[12] A. Gholamhosseinian and J. Seitz, "Vehicle classification in intelligent transport systems: An overview, methods and software perspective," *IEEE Open J. Intell. Transp. Syst.*, vol. 2, pp. 173–194, 2021.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2961–2969.

[14] A. J. Huang and S. Agarwal, "Physics-informed deep learning for traffic state estimation: Illustrations with LWR and CTM models," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 503–518, 2022.

[15] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The ApolloScape open dataset for autonomous driving and its application," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2702–2719, Oct. 2020.

[16] J. Jain et al., "Semask: Semantically masked transformers for semantic segmentation," 2021, *arXiv:2112.12782*.

[17] G. Jornod, A. Pfadler, S. Carreira, A. El Assaad, and T. Kürner, "Fuel efficient high-density platooning using future conditions prediction," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 786–798, 2022.
[18] D. Kim, W. Ga, P. Ahn, D. Joo, S. Chun, and J. Kim, "Global-local path networks for monocular depth estimation with vertical cutdepth," 2022, *arXiv:2201.07436*.
[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[20] Z. Li, Z. Chen, X. Liu, and J. Jiang, "Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation," 2022, *arXiv:2203.14211*.
[21] Z. Li, X. Wang, X. Liu, and J. Jiang, "Binsformer: Revisiting adaptive bins for monocular depth estimation," 2022, *arXiv:2204.00987*.
[22] X. Liang, M. Niu, J. Han, H. Xu, C. Xu, and X. Liang, "Visual exemplar driven task-prompting for unified perception in autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 9611–9621.
[23] X. Liang, Y. Wu, J. Han, H. Xu, C. Xu, and X. Liang, "Effective adaptation in multi-task co-training for unified autonomous driving," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 19645–19658.
[24] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2D and 3D," 2021, *arXiv:2109.13410*.
[25] J. Liu, Y. Wang, Y. Li, J. Fu, J. Li, and H. Lu, "Collaborative deconvolutional neural networks for joint depth estimation and semantic segmentation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5655–5666, Nov. 2018.
[26] X. Liu et al., "Gpt understands, too," *AI Open*, to be published.
[27] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," 2021, *arXiv:2103.14030*.
[28] H. R. Martinsen, "Autonomous driving: Vision transformers for dense prediction tasks," M.S. thesis, Dept. Comput. Sci., Norwegian Univ. Sci. Technol., Trondheim, Norway, 2022.
[29] Z. Meng, X. Xia, R. Xu, W. Liu, and J. Ma, "HYDRO-3D: Hybrid object detection and tracking for cooperative perception using 3D LiDAR," *IEEE Trans. Intell. Veh.*, vol. 8, no. 8, pp. 4069–4080, Aug. 2023.
[30] A. Mousavian, H. Pirsiavash, and J. Košecká, "Joint semantic segmentation and depth estimation with deep convolutional networks," in *Proc. IEEE 4th Int. Conf. 3D Vis. (3DV)*, 2016, pp. 611–619.
[31] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," 2021, *arXiv:2103.13413*.
[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 91–99.
[33] T. Standley, A. Zamir, D. Chen, L. Guibas, J. Malik, and S. Savarese, "Which tasks should be learned together in multi-task learning?" in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9120–9132.
[34] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "MultiNet: Real-time joint semantic reasoning for autonomous driving," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2018, pp. 1013–1020.
[35] A. Vaswani et al., "Attention is all you need," 2017, *arXiv:1706.03762*.
[36] K. Vellenga, H. Steinhauer, A. Karlsson, G. Falkman, A. Rhodin, and A. C. Koppisetty, "Driver intention recognition: State-of-the-art review," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 602–616, 2022.
[37] W. Wang et al., "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," 2021, *arXiv:2102.12122*.
[38] Z. Wang, Y. Wu, and Q. Niu, "Multi-sensor fusion in automated driving: A survey," *IEEE Access*, vol. 8, pp. 2847–2868, 2019.
[39] D. Wu et al., "YOLOP: You only look once for panoptic driving perception," *Mach. Intell. Res.*, vol. 19, no. 6, pp. 550–562, 2022.
[40] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Proc. NeurIPS*, 2021, pp. 1–14.
[41] X. Xu, H. Zhao, V. Vineet, S. Lim, and A. Torralba, "MTFormer: Multi-task learning via transformer and cross-task reasoning," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 304–321.
[42] H. Yan, C. Zhang, and M. Wu, "Lawin transformer: Improving semantic segmentation transformer with multi-scale representations via large window attention," 2022, *arXiv:2105.15203*.
[43] Z. Yang, Y. Zhang, J. Yu, J. Cai, and J. Luo, "End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perceptions," in *Proc. IEEE 24th Int. Conf. Pattern Recognit. (ICPR)*, 2018, pp. 2289–2294.
[44] S. Zheng et al., "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," 2020, *arXiv:2012.15840*.

DURGA PRASAD BAVIRISETTI received the Ph.D. degree in signal and image processing from VIT in 2016. He was a Postdoctoral Researcher with Shanghai Jiao Tong University. He is a Researcher and an Algorithm Developer specializing in machine learning, deep learning, and computer vision. He also worked as an Algorithm Expert with the Innovation Center, Alibaba Group and was a Visiting Researcher with UBC Okanagan and the University of Warsaw. He is currently a Researcher with NTNU, Norway, working on machine vision for autonomous driving and medical image computing.

HERMAN RYEN MARTINSEN received the bachelor's degree in engineering (majoring in data) and the master's degree in data technology (specializing in databases and search) from the Norwegian University of Science and Technology. He is a Software Engineer with Capgemini and internships with Uninett AS and SINTEF Ocean.

GABRIEL HANSSEN KISS received the Diploma degree in computer science engineering from the Technical University of Cluj-Napoca, Romania, and the Ph.D. degree in engineering from KU Leuven, Belgium. He is an Associate Professor with the Computer Science Department, NTNU, Trondheim, and a Senior Engineer at the Operating Room of the Future, St Olavs University Hospital. His main area of expertise is related to vision computing, medical image processing and visualization, digital twins, and ultrasound technology. Special interests include extended reality, volumetric data visualization, image registration, and fusion for ultrasound-related applications.

FRANK LINDSETH received the B.Sc. degree in engineering from BIH, Bergen, Norway, and the M.Sc. degree in mathematical science and the Ph.D. degree in computer science from the Norwegian University of Science and Technology, where he is a Professor with the Department of Computer Technology and Informatics, Faculty of Information Technology and Electrical Engineering. He was also a Senior Research Scientist with SINTEF Medical Technology, where he works in the field of Image Guided Interventions / Surgical Navigation. His areas of expertise include image guided surgery, navigation and tracking technology, medical image processing and analysis, medical visualization, computer graphics, and augmented reality. He is also skilled in human-computer interaction, GUI design, open source software development, and project management.