A Lightweight CNN-Transformer Network with Laplacian Loss for Low-altitude UAV Imagery Semantic Segmentation

Wen Lu‡, Zhiqi Zhang‡, Minh Nguyen*

***Abstract*—Semantic segmentation is crucial for enabling autonomous flight and landing of low-altitude Unmanned Aerial Vehicles (UAVs) and is indispensable for various intelligent applications. However, real-time semantic segmentation is a computationally intensive task because it involves pixel-wise classification, which renders conventional semantic segmentation networks impractical for deployment on embedded systems of limited hardware resources. Moreover, variations in flight height and object appearance increase the likelihood of misjudgment in segmentation results. To address these challenges, we propose an efficient approach consisting of a CNN-Transformer network and an auxiliary loss. The encoder of the network integrates a newly designed module, which equally handles objects with varying scales. The decoder is composed of the innovative Query-Value Squeeze Axial Transformer Attention, which reduces computational complexity from quadratic in terms of image size to $\mathcal{O}(2C(H^2+W^2))$, linear in terms of image size. By incorporating Laplacian operator convolution, the novel network-agnostic loss effectively captures intricate patterns, boundaries, and small objects. This enables extra penalization of misjudgments in these areas and compels the network to focus on objects that are challenging to distinguish. Our approach attains impressive accuracy when processing 4K resolution images in real-time (15 FPS) on a mobile GPU. It demonstrates over 2x faster speed compared to representative lightweight networks, underscoring its suitability for onboard deployment.**

***Index Terms*—semantic segmentation, lightweight neural network, unmanned aerial vehicle, remote sensing.**

I. Introduction

LOW-ALTITUDE Unmanned aerial vehicles (UAVs) are widely employed in various applications, such as crop yield estimation, disaster evaluation, urban planning, and city scene understanding [1], [2], [3], [4]. Their extensive usage leads to significant economic savings and environmental benefits. The rise in low-altitude UAV presence in airspace necessitates consideration of emergency scenarios where forced landing is essential due to system malfunctions, such as data link loss, GPS failure, engine fault, and low battery. In such situations, a UAV must autonomously identify suitable landing sites that are solid, flat, spacious, and unoccupied to ensure flight safety. During an emergency landing, a UAV must have real-time perception and cognition of its surroundings to avoid obstacles without harming humans or damaging property. Image semantic segmentation is the process of classifying each pixel into specific categories. Deep learning has become the leading method in this field and has achieved significant success [5]. An onboard semantic segmentation network is a promising approach to achieving low-altitude UAV autonomous flight and landing, because it enables a UAV to semantically understand its environment, enabling it to identify safety zones and no-fly zones [6], [7], [8]. By using near real-time onboard semantic segmentation network to analyze images collected by airborne sensors, a low-altitude UAV can detect humans, vegetation, buildings, vehicles, water, and other obstacles and react accordingly to avoid them. For example, using a semantic segmentation network called adas-0001, Ryll *et al.* utilized semantic information obtained from an onboard RGB camera to plan trajectories [9]. Symeonidis *et al.* put forward a safe landing navigation pipeline for UAVs that used PSPNet [10], a semantic segmentation network, to locate areas on the ground that were most suitable for landing, such as grass or pavement, while also identifying unsuitable landing areas, including trees, people, water, and buildings [11]. In addition, instant onboard semantic segmentation can enhance the automation and intelligence of many tasks. For example, to mitigate the adverse effects of herbicides, Deng *et al.* introduced a UAV for precise spraying. They used FCN-Alexnet [12], [13], a semantic segmentation network, to create a map of weeds and provide decision support for flight route planning and spray control. Lopez *et al.* showed a UAV designed for autonomous cleaning of insulators on power lines. They employed FCN-ResNet101 [12], [14], a semantic segmentation network, to identify dirty areas, compute the trajectory, and plan arm trajectories for efficient cleaning of the insulator [15]. Onboard semantic segmentation can also detect areas of interest and reduce data transfers to ground stations by moving computation closer to the data source. To summarize, instant onboard semantic segmentation is crucial for realizing low-altitude UAV autonomous flight and landing, as well as many intelligent applications.

Although previous studies have demonstrated the feasibility of onboard semantic segmentation and its potential applications through prototypes, their methods are inadequate to support modern high-resolution airborne cameras' image size and real-time processing speed. Due to their small and lightweight nature, most low-altitude UAVs are incapable of carrying large-capacity batteries or power-hungry processors.

Wen Lu and Minh Nguyen are with School of Engineering, Computer & Mathematical Sciences, Auckland University of Technology, Auckland 1010, New Zealand (e-mail: wen.lu@autuni.ac.nz, minh.nguyen@aut.ac.nz).

Zhiqi Zhang is with School of Computer Science, Hubei University of Technology, Wuhan 430068, China and State Key Laboratory of Information Engineering in Surveying, Mapping, and Remote Sensing, Wuhan University, Wuhan 430079, China (e-mail: zzq540@hbut.edu.cn).

‡, Wen Lu and Zhiqi Zhang contributed equally to this work.

*, Minh Nguyen is the corresponding author.

Hence, the algorithms deployed on a low-altitude UAV face stringent limitations on computational resource consumption and memory occupation. Nevertheless, all of the above methods used conventional semantic segmentation networks designed for ground computers, which adopt heavy architectures to attain satisfactory accuracy but at the expense of high computational complexity and slow inference speed. Thus, these models with high computational resource consumption are unsuitable for onboard embedded systems. While some networks, including ENet [16], ERFNet [17], BiSeNetV1 [18], and BiSeNetV2 [19], employ lightweight architectures to enhance inference speed, all of them sacrifice substantial accuracy for limited acceleration, particularly on small objects and boundaries. On the other hand, the increased image resolution accompanied with development of modern sensors makes traditional Convolutional Neural Networks (CNN), which rely on fixed-sized filters, more and more uncapable to capture long-range dependencies, where are essential for semantic understanding. Vision Transformers (ViT) model global context information in images by applying self-attention mechanism, this helps in recognizing complex patterns and relationships among image elements. In recent years, ViTs have demonstrated superiority over for some computer vision tasks, such as image classification [20], object detection [21], and semantic segmentation [22]. For example, Global-Local Transformer Segmentor (GLOTS) [23] aims to achieve coherent feature representations by employing transformers for both encoding and decoding processes. Specifically, it utilizes a pre-trained transformer encoder based on Masked Image Modeling to capture semantic-rich representations from input images. Additionally, it incorporates a multiscale global-local transformer decoder to effectively leverage both global and local features. Compared with some state-of-the-art methods, GLOTS achieved better performance. Despite their success, the Transformer architecture with the full-attention mechanism requires large computational resources, which are beyond the capabilities of many mobile and embedded devices [24]. To improve ViT efficiency, Twins [25], Swin Transformer [21], Shuffle Transformer [26], and HR-Former [27] constrain self-attention within window partitions and reduce computational complexities to $\mathcal{O}(2C\sqrt{HW}HW)$ and $\mathcal{O}(2CM^2HW)$, where H and W are the height and width of the feature map, C is the dimension of the tokens, and M is the size of the local windows. However, these advances are still insufficient to satisfy the tight constraints imposed on UAV platforms. Through the integration of CNNs and Transformers, recently emerged hybrid CNN-Transformer networks utilize CNNs' proficiency in capturing local spatial features, including edges and textures, and Transformers' strength in capturing long-range dependencies. This synergy leads to more comprehensive and effective feature extraction, particularly beneficial for tasks such as semantic segmentation, which necessitates both fine-grained details and global context understanding. Furthermore, integrating diverse architectural components often leads to improved generalization and robustness. The hybrid network can capture both local and global features simultaneously, thereby mitigating the risk of overfitting and enhancing the model's capacity to generalize to unseen data. Despite these advantages, integrating CNN and Transformer architectures increases the computational complexity of the network, potentially posing challenges in processing high-resolution images for semantic segmentation tasks. This heightened complexity may impede real-time performance and scalability, particularly on resource-constrained devices. As a result, creating a semantic segmentation network that strikes an optimal balance between accuracy and speed is an urgent as well as challenging task [5].

Besides low latency processing and low memory occupation, attention should also be paid to tackle the issue of high intra-class variation caused by scale variation. As the flight height of a low-altitude UAV varies from several meters to hundreds of meters, objects of the same category in aerial images shot at different elevations can have significantly different scales. As shown in Figure 1, ground objects of the same category, such as cars or people, exhibit quite different scales on aerial images captured at flight heights of several meters, approximately 10 meters, and around 30 meters. Traditional semantic segmentation networks, such as FCN [12], UNet [29], and SegNet [30], gradually downsample feature maps to obtain large receptive fields. They search for basic features like edges and curves at low levels and extract semantic features and dependence at deeper levels. Because their convolutional kernels are sensitive to object scales, they cannot treat the same object captured at different flight heights equally. Due to the poor generalization ability of these kernels, these networks struggle to adapt to the scale variation commonly presented in UAV imagery. The issue of high intra-class variation is also reflected in the fact that objects of the same category can display a wide range of appearances, including varied shapes, colors, and patterns. As shown in Figure 1, cars have various colors, people dress in different styles of clothes, and buildings display different patterns. Another issue is the low inter-class difference, as objects of different categories can exhibit similar visual characteristics. As shown in the right subfigure of Figure 1, the objects in ubiquitous shadows are difficult to recognize. These issues intensify the ambiguity and misjudgment in the semantic segmentation of low-altitude UAV imagery.

To overcome these challenges, we present LAPNet: a lightweight semantic segmentation network specifically designed for low-altitude UAV imagery. The encoder of LAPNet incorporates a newly devised parallel connection structure module, which employs a kernel-sharing mechanism to effectively reduce the number of parameters. As for the decoder, it is comprised of the innovative Query-Value Squeeze Axial Transformer Attention. This attention mechanism omits "Keys" and significantly reduces the computational complexity from being quadratic in terms of image size to being linear. Furthermore, we introduce the Laplacian Loss as an auxiliary loss function, which focuses on capturing intricate patterns, boundaries, and small objects through the application of Laplacian operator convolution. Its effect can be observed in Figure 2, where small objects (persons, bicycles, cars, obstacles), boundaries, regions of complex patterns, and their immediate surroundings are marked out. This enables extra penalization of misjudgments in these areas and compels the network to focus on objects that are challenging to distinguish.

Fig. 1. Ground objects of the same category, such as cars or people, exhibit significant scale differences in aerial images captured at flight heights of several meters, approximately 10 meters, and approximately 100 meters. These three aerial images were obtained from the Semantic Drone Dataset (SDD) [28].

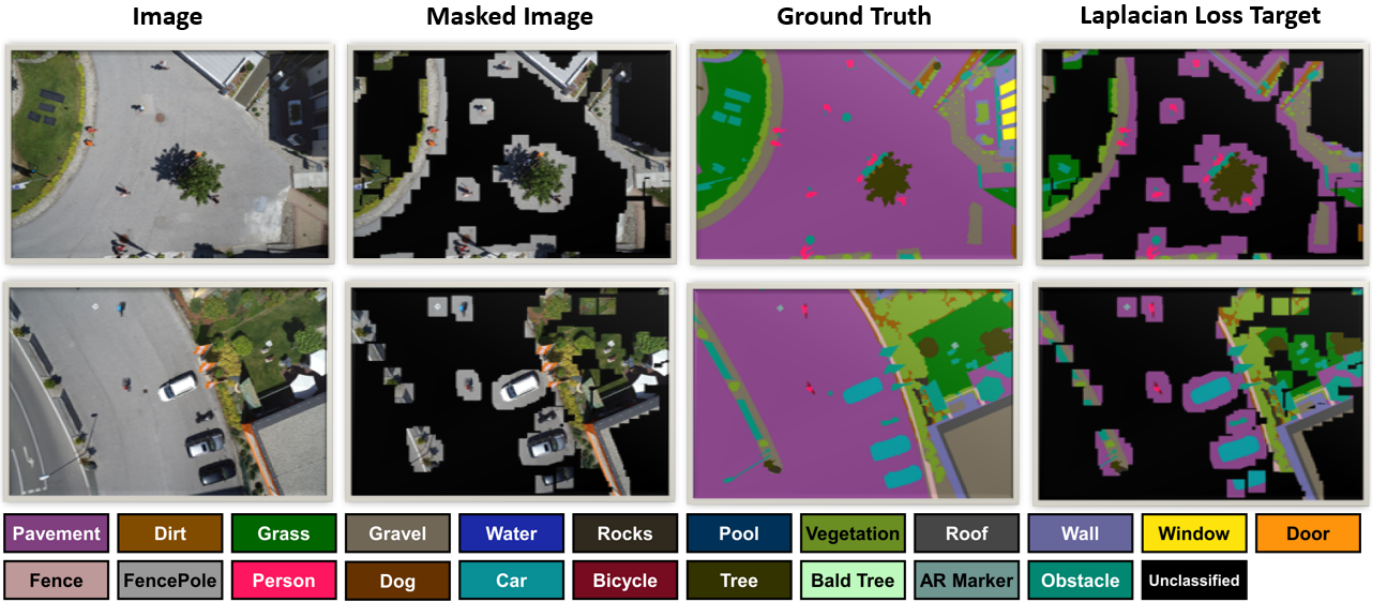

Fig. 2. Images, Masked Images, Ground Truths, and Laplacian Loss Targets of the images from the SDD dataset.

The effectiveness of our method is demonstrated in Figure 3, where LAPNets achieve significantly faster processing speed compared to existing representative networks without compromising accuracy. This exceptional trade-off between accuracy and speed allows LAPNet to efficiently meet the rigorous constraints imposed on UAV platforms, including low computational cost, minimal memory consumption, and low-latency processing. The contributions of this study can be summarized in four aspects:

1) Targeting the issue of scale variation caused by different flight heights, we have developed the Tri-branch Kernel-sharing Atrous convolution module (TKA) for feature extraction. TKA employs shared kernels to simultaneously and equally handle ground objects of inconsistent scales.
2) To capture long-range dependency at the lowest possible cost, we have designed Query-Value Squeeze Axial Transformer Attention (QVSATA) that reduces computational complexity to $\mathcal{O}(2C(H^2 + W^2))$, resulting in decreased parameters and memory usage.
3) To realize real-time onboard semantic segmentation networks within strict hardware resource limitations, we have constructed a lightweight CNN-Transformer hybrid architecture named LAPNet. Stacked TKA modules constitute its encoder, while QVSATA serves as its decoder. LAPNet achieves an optimal balance between accuracy and speed, demonstrating its potential for onboard deployment.
4) To address the challenges posed by high intra-class variation and low inter-class difference, we propose a novel network-agnostic loss called the Laplacian Loss. The Laplacian Loss captures complex patterns, boundaries, and small objects, and imposes an additional penalty on misclassifications in these areas, thereby encouraging the network to prioritize objects that are difficult to discriminate.

II. Related Work

In recent years, there has been a prosperous research focus on real-time semantic segmentation networks, driven by the increasing demand in various applications including autonomous driving, robot vision, and augmented reality. This section

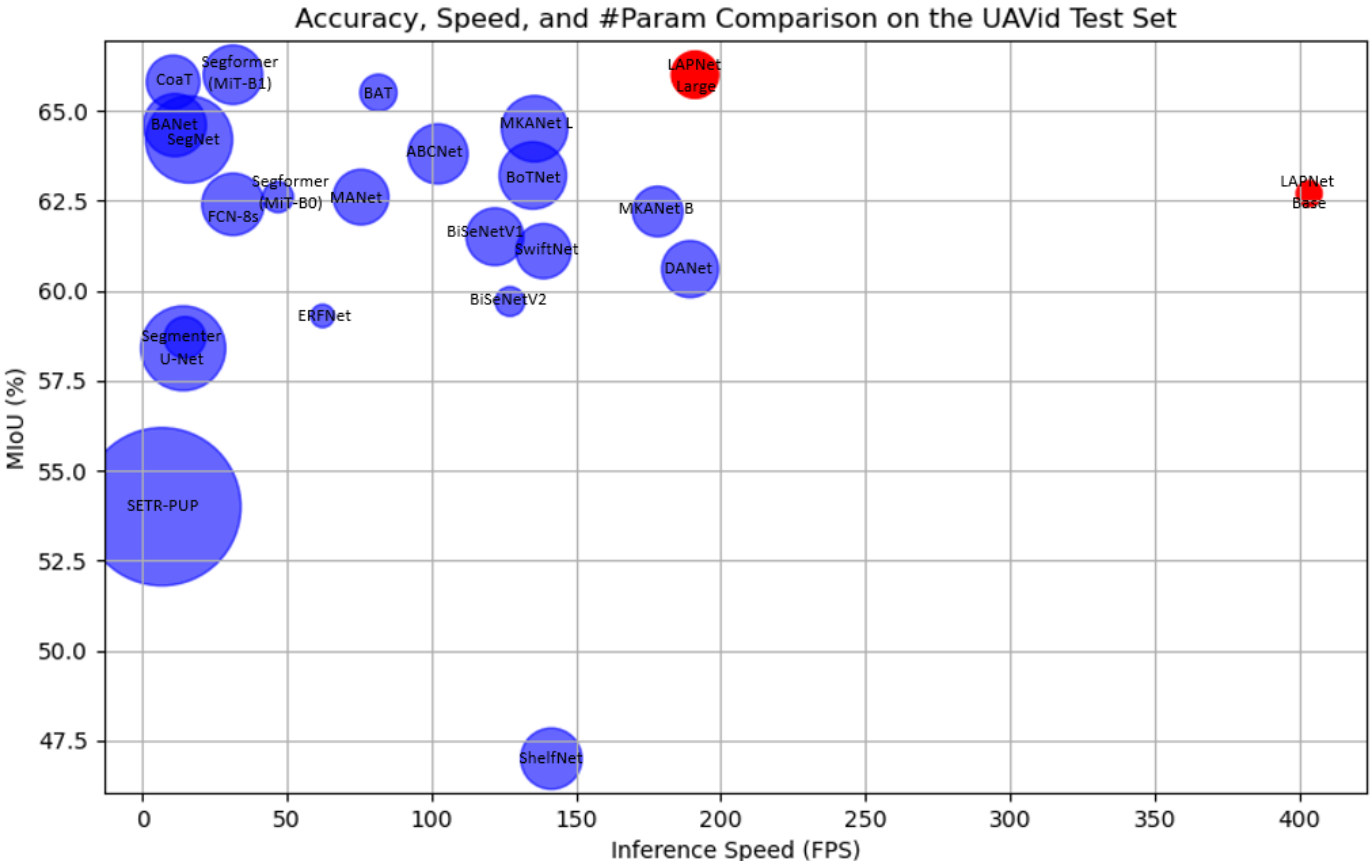

Fig. 3. Accuracy, speed, and number of network parameters comparison on the UAVid Test Set, the size of the circle represents the number of network parameters. The inference speeds were evaluated using an image size of 1024×1024 pixels on an NVIDIA RTX 3090 24G GPU. Our LAPNets (red) achieve outstanding speed-accuracy trade-off.

begins by introducing representative lightweight CNN and efficient ViT models for semantic segmentation. Subsequently, we discuss the original kernel-sharing mechanism and its limitations.

A. Lightweight CNN

ENet [16] utilizes an aggressive down-sampling approach and eliminates the final stage in order to achieve a more compact architecture. However, the marginal increase in speed is not worth the cost of a considerable accuracy drop. In general, pruning channels or abandoning stages is straightforward means to reduce computation complexity and memory consumption. Nevertheless, pruning channels would significantly weaken the learning ability of the backbone, while abandoning stages would result in an inadequate receptive field. Therefore, merely scaling down general-purpose semantic segmentation networks is not a viable solution; instead, the network architecture must be redesigned to enhance efficiency.

ERFNet [17] replaces the 3×3 convolution with a 3×1 and 1×3 convolution, which reduces the parameters by one-third. It adopts an architecture similar to ENet, with several residual downsampling blocks deployed early and a decoder that is much smaller than the encoder. While ERFNet achieves higher accuracy compared to ENet, its inference speed is slower. DABNet [31] achieves higher efficiency by utilizing a combination of depth-wise separable and asymmetric factorized convolutions. It only downsamples three times. However, due to the resulting large memory occupation, it is not suitable for embedded systems.

As accurate semantic segmentation relies on both global and local information, several methods, including BiSeNetV1 [18] and BiSeNetV2 [19], employ a dual-path architecture. In this architecture, one path maintains spatial information through wider channels and shallow layers, while the other path achieves a larger receptive field by using fewer channels and a fast-downsampling strategy. ABCNet [32] adheres to the design philosophy of BiSeNet and enhances its capabilities by incorporating an attention enhancement module, which explores long-range contextual information, and a feature aggregation module, which combines features obtained from the two paths. Hong *et al.* introduced DDRNet [33], a deep dual-resolution network comprising two deep branches. Between these branches, multiple bilateral fusions are performed. Nevertheless, in these dual-path architectures, the unbalanced computational complexity between the two branches hampers the full utilization of the parallel computational power of the GPU, resulting in a slowdown in the inference speed.

LR-ASPP [34] utilizes a lightweight backbone, MobileNetV3, to reduce latency. MobileNetV3 is specifically optimized for mobile phone CPUs through network archi-

tecture search (NAS) conducted on the ImageNet dataset [35]. However, the authors of STDC-Seg [36] argued that lightweight backbones borrowed from image classification tasks may not perform well in segmentation tasks due to the lack of task-specific design. As a result, they designed a basic module consisting of one pointwise convolution and three 3×3 convolutions connected in series, with decreasing channels. The output features of these convolutions are then concatenated and used as the input for the next module. While STDC-Seg outperformed ENet, DABNet, BiSeNetV1, and BiSeNetV2 in terms of both accuracy and speed on urban street scene datasets, the gradual expansion of the receptive field through the series connection of convolutions in its basic module makes it slower compared to the parallel connection structure.

To overcome the limitations of the aforementioned networks, our network incorporates a single-path architecture that combines local detail and global semantic learning within a single path, resulting in greater efficiency compared to the redundant dual-path architecture employed in BiSeNetV1 and BiSeNetV2. Moreover, our network's encoder utilizes a parallel connection structure for multi-scale feature extraction, and each branch in this structure has equal computational complexity.

B. Efficient ViT

The original Vision Transformer (ViT) model was introduced in 2020 as a groundbreaking approach to computer vision tasks. It replaced the traditional convolutional layers commonly used in CNNs with transformers, which were originally developed for natural language processing tasks. This substitution enabled ViT to effectively capture long-range dependencies, making it more suitable for tasks that require a comprehensive understanding of global context. However, it is important to note that transformers come with a higher computational cost compared to CNNs. The self-attention mechanism employed by transformers has a quadratic complexity, rendering it computationally expensive for high-resolution images.

Efficient ViT is an optimized version of the original ViT model, designed to improve efficiency in terms of computational complexity and memory requirements. Efficient ViT incorporates several optimizations to reduce complexity. One of the optimizations is the introduction of convolutional layers alongside transformers. These convolutional layers operate on the input image before it is fed into the transformers. By combining the strengths of both convolutional layers and transformers, Efficient ViT achieves a good balance between capturing local details (handled by convolutional layers) and global context (handled by transformers). BoTNet [37] simply replaces the spatial convolutions with global self-attention in the final three bottleneck blocks of a ResNet, this approach improves upon the baselines significantly on instance segmentation and object detection while also reducing the parameters, with minimal overhead in latency. BANet [38] consists of a dependency path and a texture path, the dependency path is conducted based on a Transformer backbone with memory-efficient multi-head self-attention, while the texture path is built on the stacked convolution operation. CoaT [39] devises a conv-attentional mechanism by realizing a relative position embedding formulation in the factorized attention module with an efficient convolution-like implementation.

Another optimization is utilizing techniques like approximate attention and sparse attention. Approximate attention techniques, such as kernelized self-attention and linearized self-attention, provide approximate solutions that trade off a loss in accuracy for significant gains in computational efficiency. Sparse attention limits the self-attention computations to a subset of tokens. For example, Token slimming [40] and Mobile-Former [41] lower down complexity by reducing the number of tokens, while Twins [25], Swin Transformer [21], Shuffle Transformer [26], and HR-Former [27] constrain self-attention within window partitions and reduce computational complexities to $\mathcal{O}(2C\sqrt{HW}HW)$ and $\mathcal{O}(2CM^2HW)$, where H and W are the height and width of the feature map, C is the dimension of the tokens, and M is the size of the local windows. However, these advances are still insufficient to satisfy the tight constraints imposed on UAV platforms.

To capture long-range dependency at the lowest possible cost, we have designed the novel Query-Value Squeeze Axial Transformer Attention that reduces computational complexity to $\mathcal{O}(2C(H^2+W^2))$, resulting in decreased parameters and memory usage.

C. Kernel-sharing Mechanism

The authors of KSAC [42] identified a weakness in the Atrous Spatial Pyramid Pooling (ASPP) [43] structure. Kernels in the branch with small atrous rates primarily focus on capturing details and handling small objects effectively, whereas kernels in the branch with large atrous rates mainly extract features of larger objects with expansive receptive fields. However, the lack of communication among these branches compromises the generalizability of individual kernels. To address this problem, the authors proposed a solution where multiple branches with different atrous rates share a single kernel. This shared kernel can scan the input feature maps multiple times, capturing information with both small and large receptive fields. Additionally, this approach allows objects of various sizes to contribute to the training of the shared kernel, resulting in an increased number of effective training samples and improved representation ability of the shared kernel. KSAC adopts the architecture of DeepLabV3+ [44]. The modified ASPP structure in KSAC includes a 1×1 convolutional branch, a global average pooling branch followed by a 1×1 convolution, and three kernel-sharing atrous convolutional branches with rates of 6, 12, and 18.

However, directly integrating KSAC into the basic module is not feasible, as KSAC can only be applied as the final stage of the encoder. Firstly, the global average pooling branch of KSAC is designed to capture image-level features. Secondly, its large atrous rates (6, 12, 18) are unsuitable for extracting low-level features, particularly in remote sensing images where objects exhibit smaller scales compared to general images. Thus, constructing a backbone by simply stacking the original KSAC structure is not feasible. Therefore, we have developed

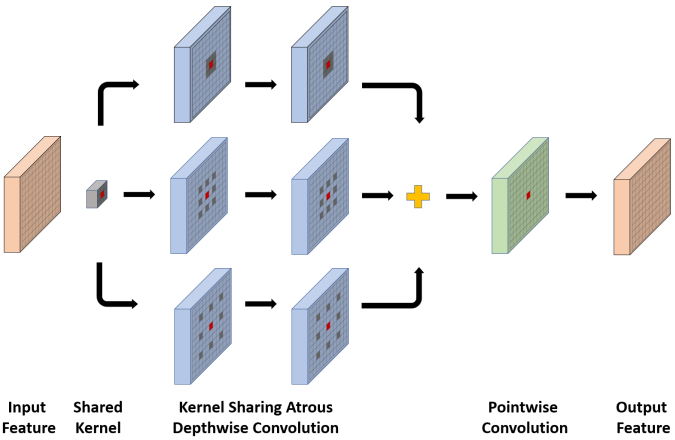

Fig. 4. Structure of the Tri-branch Kernel-sharing Atrous convolution (TKA) module.

a novel multi-branch module, as described in Section III-A. This novel module can be stacked in multiple stages to serve as the backbone for semantic segmentation.

III. Proposed Method

To perform semantic segmentation effectively, the network must capture a comprehensive view of the scene, while also preserving the intricate details. Therefore, our network adopts a hybrid architecture that incorporates CNN as the encoder and Transformer as the decoder, combining both local and global information at low computational cost. This section starts with an introduction to the Tri-branch Kernel-sharing Atrous Convolution Module, which constitutes the encoder of LAPNet. Next, we introduce the novel Query-Value Squeeze Axial Transformer Attention, which functions as the decoder of LAPNet. Lastly, we introduce the innovative Laplacian Loss.

A. Tri-branch Kernel-sharing Atrous Convolution Module

Our network's feature extraction module incorporates the kernel-sharing mechanism to address scale variation caused by varying flight heights. In contrast to KSAC, our module excludes the 1×1 convolutional branch and the global average pooling branch. Additionally, the dilation rates decrease from (6, 12, 18) to (1, 2, 3). Furthermore, regular atrous convolutions are replaced by depthwise atrous convolutions [45] to further reduce computational complexity and network parameters. As illustrated in Figure 4, the Tri-branch Kernel-sharing Atrous convolution (TKA) module is composed of three parts:

- Two levels of tri-branch 3×3 kernel-sharing depthwise atrous convolutions with dilation rates of 1, 2, and 3;
- The fusion of feature maps through addition;
- The fusion of channels through pointwise convolution.

The three branches have receptive fields of 5×5, 9×9, and 13×13, respectively, covering small-scale, medium-scale, and large-scale objects. The output feature maps of all branches are fused through addition. Therefore, if an object has a similar feature to the shared kernel, regardless of its scale, its position in the fused feature map will have a high value. Finally, a pointwise convolution is applied to the fused feature map to exchange information across channels. Assuming the number of channels in the input or output features is denoted as C, the TKA module has a total of $18C + C^2$ parameters, which is fewer than the number of parameters in a regular 3×3 convolution. Additionally, the TKA module offers the advantage of equal computational complexity among all branches, enabling full utilization of the parallel computation power of the processor.

TABLE I
The Encoder of LAPNet.

Stage	Output Size	Operation	Output Channels
Input Image	1024×1024		3
Stage 1	512×512	ConvS2	$c/2$
Stage 2	256×256	ConvS2	c
Stage 3	128×128	ConvS2	$c \times 2$
	128×128	TKA	$c \times 2$
Stage 4	64×64	ConvS2	$c \times 4$
	64×64	TKA	$c \times 4$
Stage 5	32×32	ConvS2	$c \times 8$
	32×32	TKA	$c \times 8$

ConvS2: 3×3 convolution of stride 2, followed by batch normalization and ReLU activation.
TKA: Tri-branch Kernel-sharing Atrous convolution module.
c: the channel count of the output from the stem stage.

Low-altitude UAV imagery exhibits significant scale variation as a result of varying flight altitudes. Considering its alignment with the design purpose and advantages of the TKA module, we stack TKA modules in multiple stages, forming the encoder of our lightweight CNN-Transformer network called LAPNet specifically designed for UAV imagery semantic segmentation.

As detailed in Table I, each stage begins with a 3×3 convolution of stride 2, followed by batch normalization and ReLU activation. From Stage 3 onward, the TKA module is applied. The number of channels, c, determines the width of the backbone. LAPNet has 2 typical sizes: Base (c = 60) and Large (c = 120).

B. Query-Value Squeeze Axial Transformer Attention

The Query-Value Squeeze Axial Transformer Attention (QVSATA) first pools the input feature maps along the horizontal or vertical axis, condensing them into a column or row, and subsequently applies self-attention within that column or row. Denote the input tokens as $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$, Query $\mathbf{q}$ and Value $\mathbf{v}$ are linear projections of $\mathbf{x}$:

$$\mathbf{q} = \mathbf{W_q x} \tag{1}$$

$$\mathbf{v} = \mathbf{W_v x} \tag{2}$$

where $\mathbf{W_q}, \mathbf{W_v} \in \mathbb{R}^{C \times C}$ are learnable weights.

Horizontal/vertical squeeze is implemented by taking average of $\mathbf{q}$ and $\mathbf{v}$ on the horizontal/vertical direction:

$$\mathbf{q_h} = \frac{1}{W}(\mathbf{q}^{\rightarrow(C,H,W)}\mathbb{1}_W)^{\rightarrow(H,C)} \tag{3}$$

$$\mathbf{q_v} = \frac{1}{H}(\mathbf{q}^{\rightarrow(C,W,H)}\mathbb{1}_H)^{\rightarrow(W,C)} \tag{4}$$

$$\mathbf{v_h} = \frac{1}{W}(\mathbf{v}^{\rightarrow(C,H,W)}\mathbb{1}_W)^{\rightarrow(H,C)} \tag{5}$$

$$\mathbf{v_v} = \frac{1}{H}(\mathbf{v}^{\rightarrow(C,W,H)}\mathbb{1}_H)^{\rightarrow(W,C)} \tag{6}$$

where $\mathbf{z}^{\rightarrow(\cdot)}$ means permuting the dimension of tensor $\mathbf{z}$ as given, and $\mathbb{1}_m \in \mathbb{R}^m$ is a vector with all elements equal to 1.

We render $\mathbf{q_h}$, $\mathbf{v_h}$, $\mathbf{q_v}$, and $\mathbf{v_v}$ to be aware of their position by introducing the learnable positional embeddings $\mathbf{b_h^q}$, $\mathbf{b_h^v} \in \mathbb{R}^{H\times C}$, and $\mathbf{b_v^q}$, $\mathbf{b_v^v} \in \mathbb{R}^{W\times C}$.

QVSATA is expressed as:

$$\begin{aligned}\mathbf{y}_{i,j} = &\, softmax((\mathbf{q}_{\mathbf{h}(i)} + \mathbf{b}^{\mathbf{q}}_{\mathbf{h}(i)})(\mathbf{v_h} + \mathbf{b_h^v})^\top)\mathbf{v_h} \\ &+ softmax((\mathbf{q}_{\mathbf{v}(j)} + \mathbf{b}^{\mathbf{q}}_{\mathbf{v}(j)})(\mathbf{v_v} + \mathbf{b_v^v})^\top)\mathbf{v_v}\end{aligned} \tag{7}$$

The computational complexity of squeezing the Query ($\mathbf{q}$) and Value ($\mathbf{v}$) is $\mathcal{O}(2C(H+W))$, while for the Axial Multi-head Self-attention, it is $\mathcal{O}(2C(H^2+W^2))$. As a result, our proposed QVSATA achieves a reduced total computational complexity of $\mathcal{O}(2C(H^2+W^2))$, when compared to the Regular Multi-head Self-attention ($\mathcal{O}(2C(HW)^2)$) and other Efficient Multi-head Self-attentions ($\mathcal{O}(2C\sqrt{HW}HW)$ or $\mathcal{O}(2CM^2HW)$). In contrast to other Multi-head Self-attentions that incorporate Query, Key, and Value, our approach simplifies the process by eliminating Key and substituting it with Value, thereby reducing the number of parameters as well as computational complexity.

C. Network Architecture

Our research objective is to develop a lightweight semantic segmentation network capable of achieving real-time inference speed (15 FPS) on resource-constrained devices with computational capacities lower than 5 TFLOPS when processing 4K images. When these preconditions or design specifications are established, it becomes evident that while the computational complexity of our transformer mechanism QVSATA is lower than that of other efficient Transformer mechanisms, achieving real-time inference speed is challenging unless QVSATA is exclusively applied to the final layer feature map, which has the smallest dimensions.The other reason is that the last stage extracts dense and high-level semantic features that encode complex spatial patterns and relationships, which are ideally suited for the transformer attention mechanism, enhancing the model's ability to refine segmentation predictions by comprehensively understanding the image context. Conversely, shallower layers tend to capture local details and fine structures within the image. However, their high-dimensional features often demonstrate sparsity and redundancy, leading to inefficient use of computational resources. This inefficiency may result in suboptimal performance when employing transformer attention, as the model may struggle to capture meaningful interactions between features effectively.

The architecture of LAPNet is depicted in Figure 5, while its decoder is displayed in the lower section. Initially, the Stage 5 output features are compressed by reducing the channel count from 8c to 2c using pointwise convolution, followed by batch normalization and ReLU activation. Next, the compressed features are enhanced using the QVSATA. Then, the enhanced features propagate information across channels through pointwise convolution. Subsequently, the sigmoid function is applied to convert the features into gates, allowing control over the strength of the compressed features through multiplication. By allocating varying degrees of importance to different parts of the compressed features, this attention mechanism enables the model to selectively process and utilize information that is most relevant for the task.

Since deeper stage features have a larger receptive field and contain rich semantic information, we use them to filter lower stage features, which contain local detail information. By assigning higher weights to relevant features or areas of interest, this operation helps capture important information and relationships between elements in the feature map. Specifically, a pointwise convolution and sigmoid function are applied on the gated features first, and then the resulting probability map is upsampled 4x by bilinear interpolation and afterwards multiplied with Stage 3 features.

A crucial requirement for the task of semantic segmentation is the network's ability to simultaneously capture a comprehensive view of the scene and preserve both the details and semantics of the image. In order to achieve efficient and effective integration of these elements, firstly, the gated features (which encapsulate abundant semantic information) are upsampled 4x by bilinear interpolation. Subsequently, they are merged with the attentioned Stage 3 features (which contain intricate detail information) through concatenation. Afterwards, the merged features are fed into the segmentation head, as shown in Figure 6, to generate class logits. Finally, the class logits are upsampled 8x using bilinear interpolation to restore them to the original input image size, before being used as input for the loss functions.

While residual connections have been widely employed in many semantic segmentation networks to facilitate backpropagation, we choose not to incorporate them into our network architecture due to the associated high memory usage, which would be impractical for embedded systems. To prevent gradient vanishing or exploding and enhance the feature extraction capability of the backbone, we introduce three auxiliary semantic segmentation heads on top of the output features from Stage 3 to Stage 5 during the training phase, as depicted at the upper part of Figure 5. The output class logits from the three auxiliary semantic segmentation heads are upsampled by factors of 8x, 16x, and 32x respectively before being passed to their corresponding auxiliary loss functions. During the inference phase, these auxiliary heads are discarded, resulting in no additional computational cost.

D. Laplacian Loss

We define areas with significant fluctuations in pixel values as Pixel Impure Areas (PIA). Complex patterns and boundaries are typically found in these areas, which tend to contain more valuable information compared to areas with narrow pixel

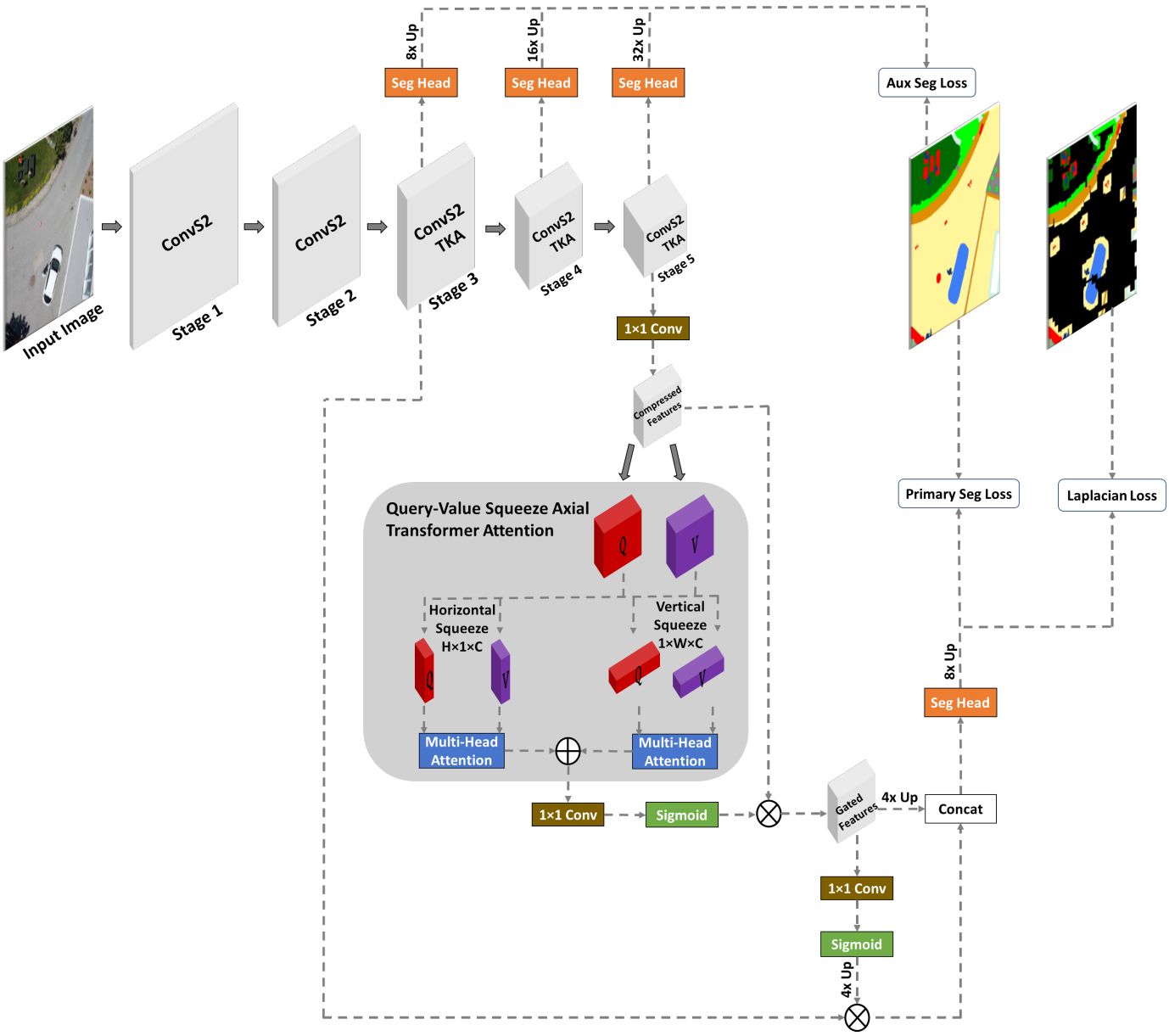

Fig. 5. The Architecture of LAPNet.
ConvS2: 3×3 convolution of stride 2, followed by batch normalization and ReLU activation.
TKA: Tri-branch Kernel-sharing Atrous convolution module.

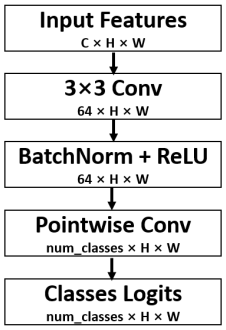

Fig. 6. Semantic Segmentation Head.

value fluctuations. Furthermore, PIA often exhibit high intra-class variation. For instance, as depicted in Figure 1, human appearances can vary greatly due to differences in hair color, hairstyle, and clothing choices, including hats and helmets. Empirical observations indicate that prediction errors are more prone to occur along boundaries and with small objects [46]. To emphasize the importance of PIA, we assign higher weights to these areas during the loss calculation. By employing the Laplacian operator convolution, which effectively detects image edges by calculating the gradients of neighbouring pixels, we can filter out most interior textures and retain object boundaries by setting an appropriate threshold value. However, since the remaining edges are very thin, a dilation operation is

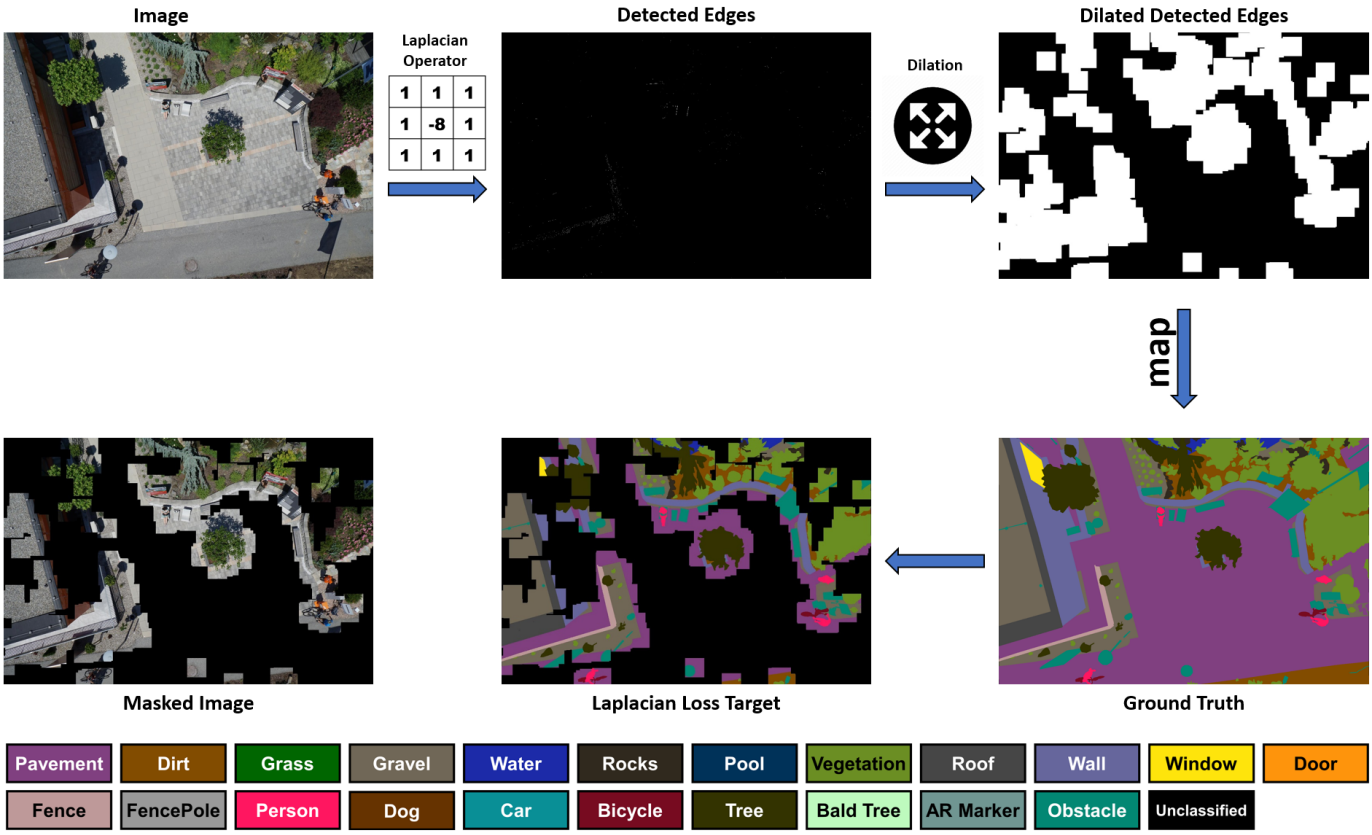

Fig. 7. The procedure of generating the Laplacian Loss target, dilation rate $d = 50$ pixels.

performed to encompass the entire bodies of small objects such as humans and obstacles. This approach enables the detection of complex patterns, boundaries, and small objects. Finally, we use these identified areas to generate targets for the proposed Laplacian Loss by mapping them to the corresponding ground truth.

The procedure is illustrated in Figure 7 and detailed in Algorithm 1. By setting an appropriate threshold value t in Algorithm 1, less noticeable edges, such as the grain of roads and the texture of grass, are filtered out, while more prominent edges, such as persons, obstacles, cars, individual plants, and complex patterns, are extracted. Therefore, t is a hyperparameter that controls the sensitivity of edge detection. For the sake of generality, we set $t = 0.5$ for the figures and subsequent experiments. As illustrated in the top-middle subfigure of Figure 7, the detected edges are thin and scarce. Therefore, a dilation operation is applied to expand these edges into areas. The dilation rate d is another hyperparameter that influences the contribution of surrounding pixels to the Laplacian Loss calculation. To achieve optimal results, d should be set to encompass the entirety of small objects and their immediate surroundings, while excluding irrelevant areas in the distance. Therefore, we empirically set d to 50 pixels for the figures and subsequent experiments. As shown in Figure 8, small objects (persons, bicycles, obstacles), boundaries, regions of complex patterns, and their immediate surroundings are marked out.

As depicted on the right side of Figure 5, the upsampled

Algorithm 1 Generate Laplacian Loss Target

Input: Image $\mathbf{X}$, Ground Truth $\mathbf{Y}$, Laplacian Operator $\mathbf{K}$, Threshold t, Dilation Rate d.
Output: Laplacian Loss Target $\hat{\mathbf{Y}}$.

$\mathbf{X_b} \leftarrow Norm(|Conv(\mathbf{X}, \mathbf{K})|) > t$
$\mathbf{X_d} \leftarrow Dilate(\mathbf{X_b})$
$\hat{\mathbf{Y}} \leftarrow \mathbf{Y} \otimes \mathbf{X_d}$
return $\hat{\mathbf{Y}}$

$\otimes$ denotes elementwise multiplication.

class logits and the Laplacian Loss targets are passed to the loss functions for Laplacian Loss calculation. The network is guided by the Laplacian Loss to prioritize complex patterns and boundaries, resulting in improved discrimination between small objects and their surrounding environment. The Laplacian Loss is a network-agnostic loss that can be applied on any semantic segmentation network, and common loss functions like the cross-entropy or Dice loss can be utilized to compute the Laplacian Loss.

The hyperparameters, namely the threshold t and the dilation rate d, can be adjusted based on the average object scale present in the dataset. Fine-tuning the hyperparameters typically leads to improved results. However, to ensure generality, we refrained from fine-tuning the hyperparameters during our experiments. Instead, we used straightforward and intuitive

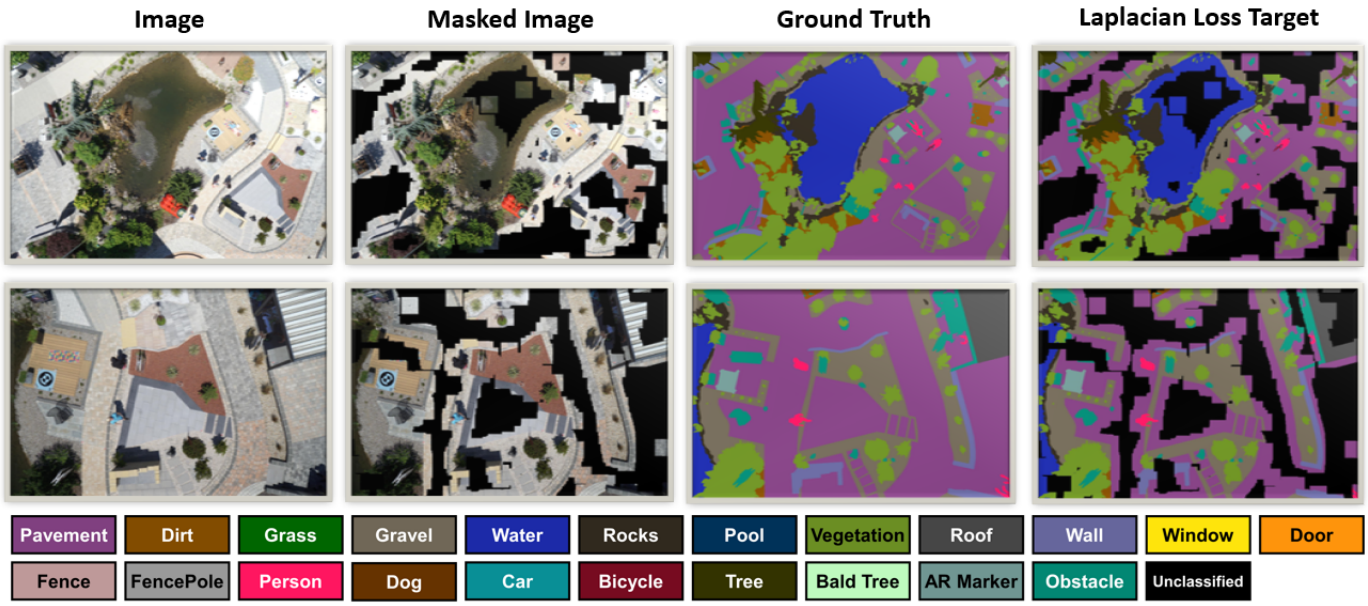

Fig. 8. Images, Masked Images, Ground Truths, and Laplacian Loss Targets of the images from the SDD dataset.

values ($t = 0.5$, $d = 50$) to assess the impact of the Laplacian Loss on prediction accuracy.

The total loss L_t is the weighted sum of the primary semantic segmentation loss L_p, the supplementary Laplacian Loss L_s, and the auxiliary semantic segmentation losses L_a:

$$L_t = w_1 \times L_p + w_2 \times L_s + w_3 \times L_a \tag{8}$$

Similar to the Laplacian Loss L_s, conventional loss functions, such as Cross-Entropy or Dice, can be utilized to compute both the primary semantic segmentation loss L_p and the auxiliary semantic segmentation losses L_a. We employed Cross-Entropy as the loss functions in the subsequent experiments, in that case, the total loss L_t is:

$$\begin{aligned} L_t = &- w_1 \times \sum_{i=1}^{N}\sum_{j=1}^{n} y_{ij} \cdot log(p_{ij}) - w_2 \times \sum_{i=1}^{N'}\sum_{j=1}^{n} y_{ij} \cdot log(p_{ij}) \\ &- w_3 \times \sum_{k=1}^{3}\sum_{i=1}^{N}\sum_{j=1}^{n} y_{ij} \cdot log(p_{ij}^{k}) \end{aligned} \tag{9}$$

where N is the total pixels within a batch, N' is the considered pixels (the pixels whose values equal 1 in the Dilated Detected Edges Map, as illustrated in the upper-right corner of Figure 7) for the Laplacian Loss, n is the number of classes, y_{ij} is the true probability distribution over classes (often one-hot encoded), p_{ij} is the predicted probability distribution over classes by the primary head, and p_{ij}^{k} is the predicted probability distribution over classes by the auxiliary head k.

In the following experiments, we assigned $w_1 = 1$ and $w_2 = 2$ to enforce a triple misjudgment penalty on PIA, thus directing the network's attention towards challenging objects that are difficult to discriminate. As the auxiliary losses do not have a direct impact on prediction accuracy, we assigned a lower weight to them by setting $w_3 = 0.15$.

In recent years, several studies have proposed a range of boundary losses aimed at enhancing the accuracy of semantic segmentation. For example, STDC-Seg [36] applies the Laplacian operator on the ground truth to generate the binary detail labels. However, the binary labels lack class information, along with the labels' scarcity due to absence of dilation operation, their effect is not significant. MKANet [47] applies the Sobel operator on the ground truth to produce boundary labels, whereas MGTT [48] utilizes the Connected Component Algorithm for the same purpose. However, employing ground truth for generating boundary labels is suitable primarily for land-cover classification in satellite imagery, where boundaries are often indistinct due to low spatial resolution. These methods fail to capture intricate patterns within large ground objects and address the challenge of high intra-class variation. In contrast, our method directly applies the Laplacian operator to the images, a technique better suited for high-spatial resolution, low-altitude UAV imagery.

IV. Experiments

In order to verify the effectiveness of LAPNet and the Laplacian Loss, we conducted experiments on three UAV datasets and compared our method with various methods.

A. Experimental Setting

Network training was conducted using an NVIDIA RTX 3090 24G GPU, while network inference speed was measured using an NVIDIA RTX 2060 Max-Q 6G Mobile GPU (FP32: 4.55 TFLOPS), which has similar computational capacity to that of an NVIDIA Jetson AGX Orin embedded system (FP32: 5.33 TFLOPS). Network performance was assessed by the Mean Intersection over Union (MIoU) metric defined as:

TABLE II
COMPARISON ON THE UAVID TEST SET.

Method	Class IoU (%)								MIoU	FPS[1]	FPS[2]	#Param	MACs
	Clutter	Build.	Road	Tree	Vege.	M Car	S Car	Human	(%)	3090	2060M	(M)	(G)
Heavyweight CNN													
Dilation Net [50]	45.4	80.7	65.1	73.8	45.5	53.6	24.5	0.00	48.6	-	-	-	-
MSD [4]	57.0	79.8	74.0	74.5	55.9	62.9	32.1	19.7	57.0	-	-	-	-
U-Net [29]	61.8	82.9	75.2	77.3	62.0	59.6	30.0	18.6	58.4	14.1	OOM	28.0	1057.9
FCN-8s [12]	63.9	84.7	76.5	78.3	61.9	65.9	45.5	22.3	62.4	31.3	OOM	15.1	322.8
SegNet [30]	65.6	85.9	79.2	78.8	63.7	68.9	52.1	19.3	64.2	16.1	OOM	29.5	645.9
Lightweight CNN													
ShelfNet [51]	44.1	76.9	61.4	73.2	43.4	52.6	21.0	3.6	47.0	141.4	4.6	14.6	46.7
ERFNet [17]	64.5	85.6	77.3	77.9	62.2	60.6	46.1	0.00	59.3	62.3	1.9	**2.1**	59.4
BiSeNetV2 [19]	61.2	81.6	77.1	76.0	61.3	66.4	38.5	15.4	59.7	127.1	4.4	3.4	49.3
DANet [52]	64.9	85.9	77.9	78.3	61.5	59.6	47.4	9.1	60.6	189.4	6.3	12.6	39.6
SwiftNet [53]	64.1	85.3	61.5	78.3	76.4	51.1	62.1	15.7	61.1	138.7	4.5	11.8	51.6
BiSeNetV1 [18]	64.7	85.7	61.1	78.3	77.3	48.6	63.4	17.5	61.5	121.9	4.2	12.9	51.8
MANet [54]	64.5	85.4	77.8	77.0	60.3	67.2	53.6	14.9	62.6	75.6	-	12.0	51.7
CANet [55]	66.0	86.6	62.1	79.3	**78.1**	47.8	**68.3**	19.9	63.5	-	-	-	-
ABCNet [32]	67.4	86.4	81.2	79.9	63.1	69.8	48.4	13.9	63.8	102.2	3.7	14.0	62.9
MKANet B [47]	63.3	83.2	79.6	77.9	60.8	69.6	43.1	20.4	62.2	178.2	6.3	9.9	25.8
MKANet L [47]	66.1	85.2	81.1	79.3	62.4	70.3	49.8	22.1	64.5	135.6	4.6	17.1	43.9
Transformer													
SETR-PUP [22]	55.6	77.8	71.6	73.5	54.2	55.9	26.9	16.5	54.0	6.7	OOM	97.6	-
Segmenter(ViT-T) [56]	64.2	84.4	79.8	76.1	57.6	59.2	34.5	14.2	58.7	14.7	OOM	6.7	26.8
Segformer(MiT-B0) [57]	63.2	84.8	77.9	77.8	59.7	69.2	43.3	25.2	62.6	46.9	OOM	3.7	27.2
Segformer(MiT-B1) [57]	66.6	86.3	80.1	79.6	62.3	72.5	52.5	**28.5**	**66.0**	31.3	OOM	13.7	63.3
CNN-Transformer													
BoTNet [37]	64.5	84.9	78.6	77.4	60.5	65.8	51.9	22.4	63.2	135.0	3.8	17.6	49.9
UAVSNet [58]	64.9	86.0	77.0	77.6	62.0	67.0	54.3	26.0	64.4	-	-	-	-
BANet [38]	66.7	85.4	80.7	78.9	62.1	69.3	52.8	21.0	64.6	11.2	-	15.4	-
BAT [59]	66.2	85.1	81.2	79.8	63.2	72.8	52.1	23.3	65.5	81.6	2.9	5.3	47.2
CoaT-Mini [39]	**69.0**	**88.5**	80.0	79.3	62.0	70.0	59.1	18.9	65.8	10.6	0.2	11.1	104.8
LAPNet Base	63.6	83.1	79.4	78.2	61.0	69.7	43.8	22.5	62.7	**403.5**	**16.2**	2.5	**7.7**
LAPNet Large	66.4	85.5	**81.5**	**80.5**	63.4	**73.4**	52.4	24.5	**66.0**	191.1	6.5	8.6	25.4

[1] The inference speeds were measured using a cropped image size of 1024×1024 pixels on an NVIDIA RTX 3090 24G GPU.
[2] The inference speeds were measured using the original image size of 3840×2160 pixels on an NVIDIA RTX 2060 Max-Q 6G Mobile GPU.
OOM denotes out of GPU memory.
Multiply-Accumulate Operations were measured using a cropped image size of 1024×1024 pixels.
- means that the performance is not reported by the authors and the code is not released on the GitHub for us to measure.

$$\text{MIoU} = \frac{1}{N}\sum_{c=1}^{N}\frac{TP_c}{TP_c + FP_c + FN_c}. \tag{10}$$

where N represents the number of classes and TP_c, FP_c and FN_c denote the number of true positive pixels, false positive pixels, and false negative pixels, respectively, in Class c.

The cross-entropy loss function was employed for the main semantic segmentation loss, auxiliary semantic segmentation loss, and the Laplacian Loss, while AdamW [49] was selected as the optimizer. The base learning rate was set to 0.001, implementing cosine decay.

B. Experimental Results on the UAVid Dataset

UAVid is a dataset for UAV imagery semantic segmentation, comprising high-resolution (4K) oblique-view urban scene images captured at an approximate flying height of 50 meters [4]. A total of 420 images were officially split into three sets: a training set consisting of 200 images, a validation set consisting of 70 images, and a test set consisting of 150 images. The ground truths of the test set are withheld for benchmarking. The dataset contains 8 classes: building, road, tree, low vegetation, static car, moving car, human, and clutter.

The networks were trained for 1000 epochs, with a batch size of 12. A warmup strategy was employed during the initial 50 epochs. Data augmentations, such as random flipping, random rotation, random scaling of rates (0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.25, 1.5, 2.0), random cropping into size 1280×1280 pixels, and color jitterring, were employed on the input images during the training process.

According to Table II, LAPNet Large demonstrates superior performance in terms of both accuracy and speed. LAPNet Large achieves an accuracy that is 5.4% higher than DANet while maintaining approximate inference speed. It also performs inference at a speed six times faster than Segformer (MiT-B1) while maintaining equal accuracy. Among CNN-Transformer hybrid models, LAPNet Large exhibits slightly higher accuracy compared to CoaT. However, it achieves an inference speed that is 18 times faster than CoaT, as measured using a cropped image size of 1024×1024 pixels on an RTX 3090 GPU. The performance gap increases to 32 times faster when evaluating the original 4K image size on an RTX 2060 Mobile GPU. The expanded superiority is attributed to the QVSATA, which reduces computational complexity from quadratic in terms of image size to linear in terms of image size.

TABLE III
COMPARISON ON THE UDD6 VALIDATION SET.

Method	Class IoU (%)						MIoU (%)	FPS 2060M
	Other	Facade	Road	Vegetation	Vehicle	Roof		
Heavyweight CNN								
U-Net [29]	58.1	67.3	65.2	89.3	62.6	82.5	70.8	OOM
FCN-8s [12]	58.7	67.7	66.1	88.4	66.2	83.5	71.8	OOM
DeepLabV3+ [44]	70.8	72.3	71.2	81.8	62.9	80.2	73.2	OOM
SegNet [30]	60.7	71.9	68.7	89.0	67.6	86.5	74.1	OOM
ACNet [60]	71.2	73.2	71.5	82.8	64.2	81.8	74.1	-
Lightweight CNN								
ENet [16]	56.0	63.4	66.8	89.2	55.7	85.6	69.5	3.8
BiSeNetV1 [18]	57.8	64.4	65.6	90.0	59.0	84.2	70.2	3.9
BiSeNetV2 [19]	59.5	68.0	66.8	89.8	58.7	84.4	71.2	4.1
DDRNet23 [33]	59.6	66.9	66.2	90.4	62.6	85.1	71.8	3.3
LR-ASPP [34]	57.7	68.2	67.1	89.9	63.0	86.3	72.0	5.9
DABNet [31]	59.3	69.4	69.4	89.8	59.3	86.9	72.4	4.1
ERFNet [17]	59.2	72.3	67.4	89.1	65.4	86.7	73.4	1.8
DANet [52]	71.6	72.6	71.3	82.1	63.1	81.3	73.7	5.8
STDC1-Seg [36]	60.5	68.8	69.2	90.3	66.6	87.5	73.8	5.2
MKANet B [47]	61.9	70.3	69.5	90.6	67.1	87.2	74.4	5.7
MKANet L [47]	63.4	72.2	72.1	90.5	66.9	87.8	75.5	4.1
Transformer								
SETR [22]	69.6	70.9	69.8	79.6	61.8	79.9	71.9	OOM
Swin Transformer [21]	70.2	71.8	69.9	82.1	62.3	80.0	72.7	OOM
OCRNet [61]	71.8	73.4	71.0	81.9	63.6	81.6	73.9	OOM
UAVSNet [58]	60.9	71.7	68.4	89.4	**70.5**	86.9	74.6	-
Segformer [57]	71.8	73.1	72.3	82.5	68.6	80.8	74.9	OOM
CSWin Transformer [62]	**73.6**	**74.9**	**73.8**	84.8	64.9	86.9	76.5	OOM
CNN-Transformer								
BoTNet [37]	61.2	69.8	69.5	89.9	66.8	87.6	74.1	3.5
LETNet [63]	62.0	70.3	69.6	90.2	67.4	87.5	74.5	4.1
CoaT-Mini [39]	62.7	71.5	70.6	90.3	67.7	88.1	75.2	0.2
BAT [59]	64.4	73.2	73.1	90.5	68.2	88.2	76.3	2.8
LAPNet Base	61.8	70.6	70.7	90.6	65.6	87.4	74.5	**15.1**
LAPNet Large	64.6	74.1	72.8	**90.9**	68.6	**88.3**	**76.6**	6.0

The inference speeds were measured using the original image size of 4096×2160 pixels on an NVIDIA RTX 2060 Max-Q 6G Mobile GPU.
OOM denotes out of GPU memory.
- means that FPS is not reported by the authors and the code is not released on the GitHub for us to measure FPS.

Among the networks examined, LAPNet Base is the only one capable of performing real-time (15 FPS) 4K resolution image inference on an RTX 2060 Mobile GPU, while still achieving comparable accuracy. In contrast, none of the Heavyweight CNNs and Transformers can load images at their original resolution due to insufficient GPU memory. In comparison to BoTNet and ABCNet, LAPNet Base achieves a remarkable fourfold increase in speed while only experiencing a marginal decrease in accuracy of 0.5% and 1.1%, respectively. When evaluated with a cropped image size of 1024×1024 pixels on an RTX 3090 GPU, LAPNet Base demonstrates inference speeds that are five and eight times faster than those of MANet and Segformer (MiT-B0), respectively, while exhibiting 0.1% higher accuracy.

The notable efficiency of LAPNets aligns with their network parameter count and computational complexity, as demonstrated in the final two columns of Table II, where they exhibit fewer network parameters and significantly lower MACs compared to other models.

C. Experimental Results on the UDD Dataset

The Urban Drone Dataset (UDD) [64] contains a variety of urban scenes in four Chinese cities and was collected by a UAV in oblique view and nadir view at altitudes between 60 and 100 meters. It includes 6 classes: facade, road, vegetation, vehicle, roof, and other. The image size is either 4096×2160 pixels or 4000×3000 pixels. The dataset is officially divided into a training set and a validation set. As in previous works, we report experimental results on the validation set.

The networks were trained for 1000 epochs, with a batch size of 12. A warmup strategy was employed during the initial 50 epochs. Data augmentations, such as random flipping, random rotation, random scaling of rates (0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.25, 1.5, 2.0), random cropping into size 1280×1280 pixels, and color jitterring, were employed on the input images during the training process.

As presented in Table III, similar to the experiments conducted on the UAVid dataset, both Heavyweight CNNs and Transformers face limitations in loading images at their original resolution due to insufficient GPU memory. However, LAPNet Base stands out as the only network capable of achieving real-time (15 FPS) inference speed while processing images at the native 4K resolution. Additionally, LAPNet Large surpasses other networks in terms of both accuracy and inference speed metrics.

In this dataset, long range receptive field is an important factor for segmentation accuracy, as evidenced by the performance disparity observed between the Swin Transformer and CSWin Transformer. While Swin Transformer divides the input features into non-overlapping windows and performs

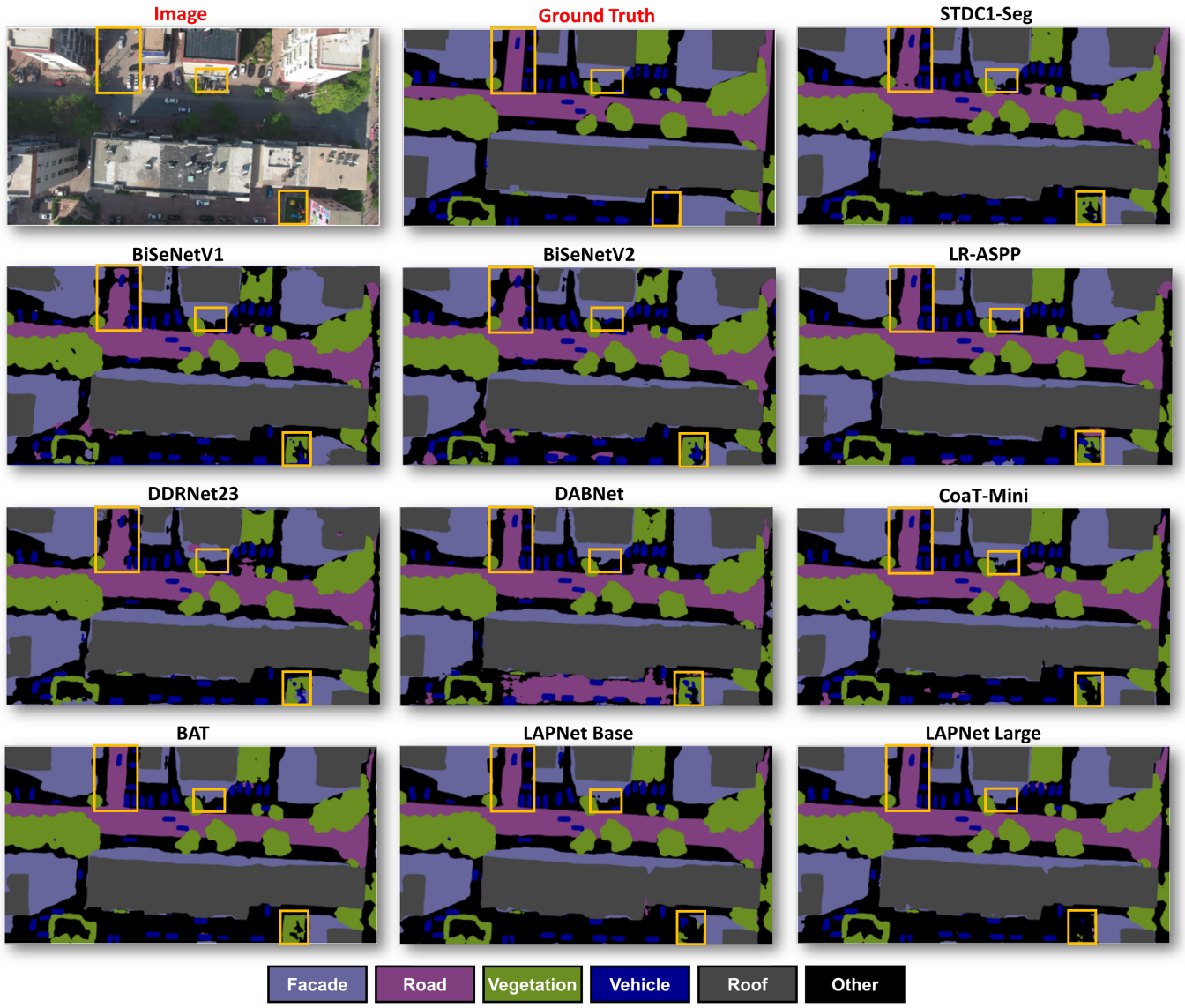

Fig. 9. A comparison of the results predicted by various methods on the UDD dataset.

self-attention within each local partition, CSWin Transformer divides the input feature into stripes of equal width and performs self-attention within the horizontal and vertical stripes that form a cross-shaped window. Consequently, the stripe attention mechanism employed in the CSWin Transformer excels at modeling dependencies over longer ranges. The QVSATA employed in LAPNet shares similarities with the stripe attention mechanism. However, it enhances its ability to capture a global receptive field by averaging the Query q and Value v in the horizontal/vertical direction prior to performing vertical/horizontal axial self-attention. Hence, LAPNet Large achieves a slightly higher accuracy compared to CSWin Transformer.

Figure 9 shows that half of the area is exposed to sunlight, while the other half is in shadow. Shadows are prevalent in urban areas and pose difficulty even for human discrimination of different classes within them. The UDD dataset presents a low inter-class difference, which poses challenges for semantic segmentation models. The predicted results demonstrate that LAPNet surpasses other lightweight networks in effectively recovering details of roads, facades, and objects in shadows, highlighting its superior urban scene understanding. This advantage can be attributed to LAPNet's kernel-sharing mechanism, where objects of varying sizes contribute to the training of shared kernels, enhancing their feature representation ability.

When comparing the accuracy and predicted results of LAPNet Base and LAPNet Large, a notable improvement in segmentation accuracy is observed by increasing the width of the network (as measured by the channel count of the output feature from the stem stage). This improvement can be attributed to the presence of a higher number of shared kernels, which provide richer feature representations and, consequently, enhance the network's feature extraction ability. Moreover, LAPNet can be easily scaled according to available computational resources by adjusting the value of c in Table

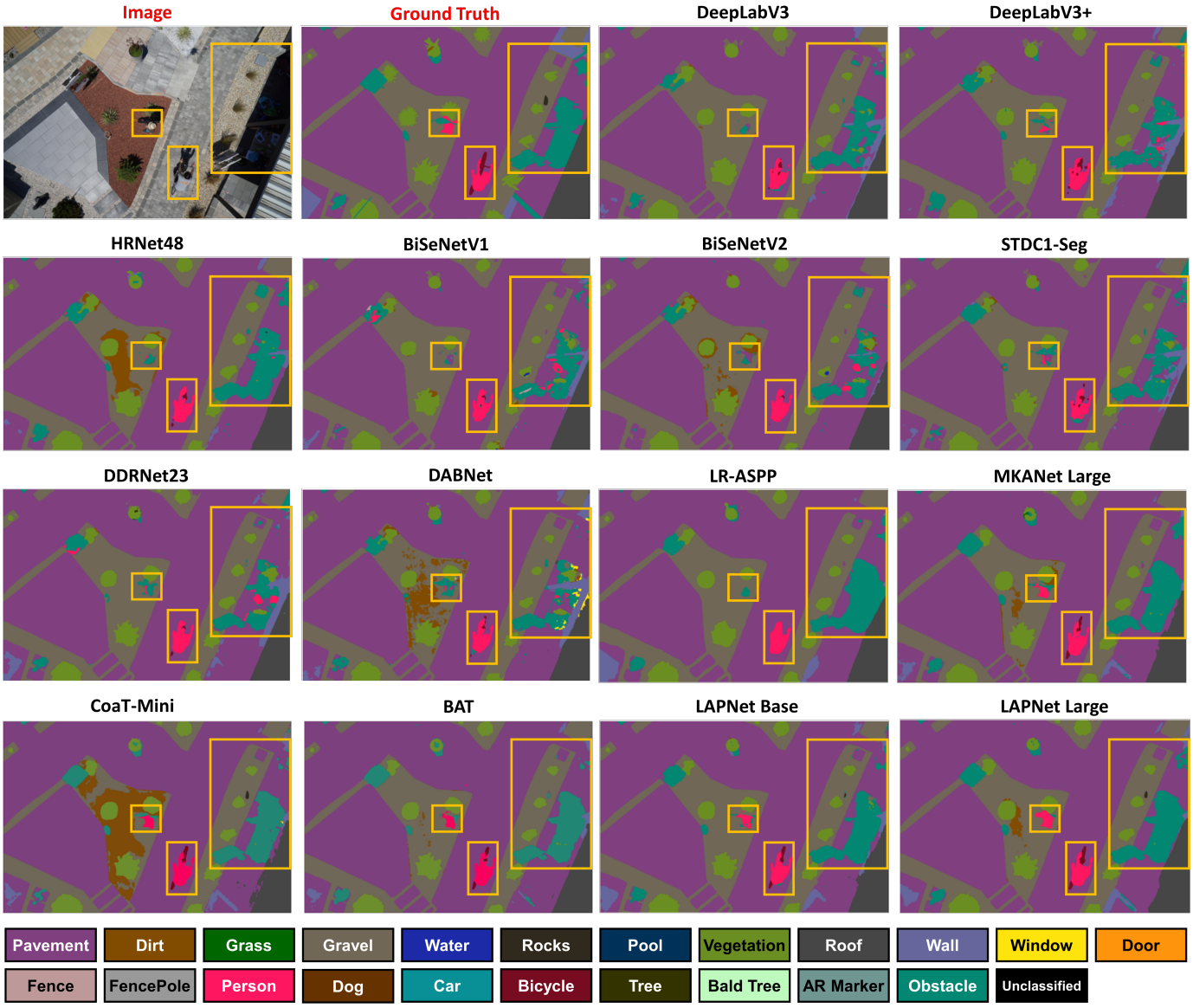

Fig. 10. A comparison of the results predicted by various methods on the SSD dataset.

I.

D. Experimental Results on the SDD Dataset

The Semantic Drone Dataset (SDD) [28] is dedicated to enhancing the safety of autonomous drone flight and landing procedures through semantic understanding of urban scenes. The dataset captures a residential community comprising over 20 houses situated in a small European town. The UAV images were collected from a nadir view at altitudes ranging from 5 to 30 meters. A high-resolution camera with a resolution of 4000×6000 pixels was employed for image acquisition. The dataset contains 22 classes: paved area, dirt, grass, gravel, water, rocks, pool, vegetation, roof, wall, window, door, fence, fence pole, person, dog, car, bicycle, tree, bald tree, AR marker, and obstacle. The proportion of each class is displayed in Table IV. In contrast to other UAV imagery datasets (Chen *et al.*, 2018; Lyu *et al.*, 2020) with fewer than 10 classes, the object classification in this dataset is highly detailed. Certain classes, such as fence poles and bicycles, have a small scale, while other classes, specifically persons, dogs, and obstacles, exhibit significant intra-class variation. Additionally, their low proportions, approximately 1%, present additional challenges for semantic segmentation models. Nonetheless, these classes hold substantial importance for ensuring flight safety and enabling autonomous landing. The images and ground truths are displayed in the first and third column of Figure 8, with the class legend provided at the bottom. The dataset consisting of 400 publicly available images was divided into training, validation, and test sets at a ratio of 7:1:2.

The networks were trained for 800 epochs, with a batch size of 5. A warmup strategy was employed during the initial 50 epochs. Data augmentations, such as random flipping, random rotation, random scaling of rates (0.8, 0.9, 1.0, 1.1, 1.25), random cropping into size 1536×1536 pixels, and color

TABLE IV
COMPARISON ON THE SDD TEST SET.

	Class IoU(%)																					MIoU	FPS
Method	**Pave.**	**Dirt**	**Gras.**	**Grav.**	**Wat.**	**Rock**	**Pool**	**Vege.**	**Roof**	**Wall**	**Win.**	**Door**	**Fen.**	**Pole**	**Per.**	**Dog**	**Car**	**Bic.**	**Tree**	**Bald**	**Mark.**	**Obst.**	**(%)**	**2060M**
Prop. (%)	37.6	3.3	20.0	7.2	2.3	0.7	0.7	6.9	7.5	2.7	0.6	0.1	1.0	0.1	1.1	0.1	0.8	0.2	2.1	1.4	0.2	3.5		
Heavyweight:																								
DeepLabV3 [43]	90.1	49.1	93.1	68.4	90.3	36.8	95.3	66.0	79.3	49.9	34.0	0.0	25.3	0.0	53.7	21.1	88.8	55.8	50.8	53.2	79.0	54.9	56.1	OOM
DeepLabV3+ [44]	90.5	48.7	92.6	68.9	90.7	37.5	93.6	66.3	81.4	53.0	33.6	3.0	28.8	2.2	52.7	26.3	90.4	55.2	55.0	51.1	77.3	56.7	57.1	OOM
HRNet48 [65]	89.3	51.2	91.9	68.2	89.9	39.5	92.8	67.3	77.3	53.7	32.3	0.8	29.1	4.4	55.0	46.7	83.3	59.8	64.7	50.1	76.6	57.4	58.2	OOM
Lightweight:																								
ENet [16]	89.8	46.7	80.1	60.6	87.7	6.6	83.8	60.9	75.1	22.3	0.0	0.0	0.0	0.0	31.9	0.0	50.1	0.0	47.9	43.9	31.4	33.1	38.7	4.7
ERFNet [17]	91.2	48.3	92.1	77.9	87.3	27.7	90.5	59.2	75.0	28.3	13.0	0.0	2.6	0.0	37.8	0.0	76.0	12.8	52.0	47.3	55.1	39.4	46.1	2.4
BiSeNetV1 [18]	88.2	48.3	92.7	72.8	87.8	38.9	89.0	65.6	69.3	40.6	23.2	0.0	20.1	0.5	49.1	24.9	71.8	57.0	55.6	47.0	75.8	50.0	53.1	5.2
BiSeNetV2 [19]	91.6	47.0	91.2	77.9	85.1	37.9	81.2	61.9	77.6	38.0	17.8	0.0	27.6	0.0	48.5	21.2	87.4	57.2	49.3	51.2	75.5	51.5	53.5	5.3
STDC1-Seg [36]	93.9	52.0	93.5	81.2	91.4	45.4	94.8	70.6	80.8	50.8	34.6	0.0	37.7	0.7	59.9	37.9	91.1	62.8	67.7	54.5	77.6	59.8	60.8	6.8
DDRNet23 [33]	91.6	54.4	93.7	72.8	92.5	47.5	95.9	69.7	80.1	55.8	38.2	0.0	37.5	1.9	57.3	30.5	93.1	67.1	63.2	61.5	76.3	59.9	60.9	4.5
DABNet [31]	93.4	54.6	93.1	78.5	92.1	48.9	92.3	70.7	82.1	50.6	39.9	0.0	38.6	2.6	59.6	29.1	90.4	63.3	69.6	51.1	82.1	58.0	60.9	5.4
LR-ASPP [34]	91.8	52.3	93.9	72.6	90.9	49.3	96.0	70.4	83.1	54.7	38.2	1.3	38.6	2.4	63.7	39.2	92.3	62.5	72.9	56.3	77.6	60.7	61.8	7.6
MKANet B [47]	93.5	52.9	94.3	78.3	93.7	57.2	97.6	73.7	83.1	57.6	52.4	0.6	42.9	7.6	65.7	51.7	93.3	68.8	74.0	64.1	82.5	64.3	65.9	7.2
MKANet L [47]	94.5	55.3	94.8	79.5	92.0	**62.0**	97.9	73.5	85.6	67.4	56.1	10.1	48.6	9.8	70.3	59.4	94.6	75.5	74.5	64.7	86.3	68.6	69.1	5.5
CNN-Transformer																								
BoTNet [37]	94.1	53.8	94.2	80.5	93.5	55.5	97.5	71.4	84.3	57.6	43.3	0.0	41.8	7.3	63.9	53.1	93.1	68.6	70.9	60.6	85.0	64.1	65.2	4.5
LETNet [63]	94.8	52.9	94.3	82.5	93.6	53.3	97.5	72.7	85.1	58.8	47.4	0.1	44.2	10.1	67.7	53.3	93.6	70.6	73.2	61.2	86.7	65.6	66.3	5.2
CoaT-Mini [39]	93.2	53.4	94.4	74.7	92.0	60.9	97.5	73.5	84.7	65.2	57.8	0.0	51.7	9.2	71.3	60.2	93.9	72.8	75.0	61.1	85.0	67.4	67.9	0.2
BAT [59]	95.3	53.7	94.8	83.3	91.4	59.7	97.6	72.7	86.1	63.0	54.2	1.3	52.1	12.2	72.1	64.0	93.5	73.7	74.1	65.8	83.2	69.3	68.8	3.6
LAPNet Base	94.9	53.2	94.4	83.6	**94.4**	55.8	97.0	71.9	83.6	58.9	49.3	4.6	48.1	8.4	69.2	65.9	94.6	68.6	72.3	**67.4**	84.7	66.6	67.6	**15.8**
LAPNet Large	**95.9**	**55.7**	**94.9**	**84.4**	91.1	61.9	**98.4**	**74.1**	**86.5**	**68.5**	**60.2**	**23.2**	**53.4**	**13.4**	**72.4**	**74.1**	**95.4**	**76.1**	**77.0**	67.2	**86.8**	**70.3**	**71.9**	7.6

The inference speeds were measured on an NVIDIA RTX 2060 Max-Q 6G Mobile GPU. The experiments were conducted using a downscaled image size of 3072×2048 pixels. The predictions from the downscaled images were later upscaled to the original size of 6000×4000 pixels.
OOM denotes out of GPU memory.

jitterring, were employed on the input images during the training process. To prevent GPU memory overflow caused by the large original image size of 4000×6000 pixels, we downscaled the images to 2048×3072 pixels before feeding them into the networks. Subsequently, the network's output class logits were upscaled back to the original size.

As present in Table IV, LAPNet Large continues outperforming other networks in terms of accuracy while also demonstrating faster inference speed. It exhibits superiority primarily in small-scale classes (e.g., door, fence, fence pole, bicycle, bald tree, and AR marker), as well as classes with high intra-class variation (e.g., person, dog, wall, window, and obstacle). Notably, these classes are the ones that frequently occur in the Laplacian Loss target, providing evidence for the effectiveness of the Laplacian Loss. Moreover, considering the significance of these classes in flight safety and autonomous landing, our method demonstrates further advantages in this domain.

Consistent with the experimental results of the UAVid dataset and UDD dataset, LAPNet Base emerges as the only network capable of achieving real-time inference speed (15 FPS), which is more than two times faster than other lightweight networks in the comparison. While sacrificing only 1.5% MIoU, it achieves three times faster inference speed than MKANet Large, which ranks second in accuracy among the networks. Due to limitations in GPU memory, heavyweight networks are still incapable of handling the downscaled image size of 3072×2048 pixels.

The predicted results shown in Figure 10 demonstrate that both LAPNet Base and Large outperform other networks in recovering details of persons, bicycles, and obstacles in shadows. This finding aligns with the Class IoU values presented in Table IV, illustrating that LAPNets exhibit a significant advantage over other networks in these classes. This superiority can be attributed to two factors. First, the TKA module, featuring multi-scale receptive fields while preserving spatial resolution, facilitates the establishment of object dependencies and maintains spatial details. Second, the Laplacian Loss effectively identifies small objects (person and bicycle) and boundary regions (shadows), enabling the network to prioritize these challenging areas during training.

V. ABLATION ANALYSIS

This section begins with an analysis of the effect of kernel-sharing mechanism. Subsequently, we demonstrate the effectiveness of QVSATA, which serves as a crucial component within the decoder. Finally, the impact of Laplacian Loss is evaluated.

A. The Effect of Kernel-sharing Mechanism

A variant network was constructed to evaluate the effect of kernel-sharing mechanism by replacing kernel-sharing atrous convolutions with regular atrous convolutions. Table V presents results demonstrating that the kernel-sharing atrous convolution improves the MIoU by 2.6% while maintaining the same computational cost as regular atrous convolution. Consequently, the kernel-sharing mechanism enhances the generalization ability of kernels by scanning objects at multiple scales. This leads to the derivation of more discriminative feature embeddings, which facilitate semantic understanding.

B. The Effectiveness of QVSATA

To assess the effectiveness of the QVSATA, a variant network was constructed by simply concatenating features from Stage 3 and Stage 5. This network was compared to LAPNet, which incorporates the QVSATA as a crucial component within its decoder. Table VI presents the results showing

TABLE V
COMPARISON OF LAPNET LARGE WITH AND WITHOUT KERNEL-SHARING MECHANISM ON THE SDD DATASET.

	Class IoU(%)																					MIoU
Method	**Pave.**	**Dirt**	**Gras.**	**Grav.**	**Wat.**	**Rock**	**Pool**	**Vege.**	**Roof**	**Wall**	**Win.**	**Door**	**Fen.**	**Pole**	**Per.**	**Dog**	**Car**	**Bic.**	**Tree**	**Bald**	**Mark.**	**Obst.**	**(%)**
Prop. (%)	37.6	3.3	20.0	7.2	2.3	0.7	0.7	6.9	7.5	2.7	0.6	0.1	1.0	0.1	1.1	0.1	0.8	0.2	2.1	1.4	0.2	3.5	
LAPNet Large Sharing	**95.9**	**55.7**	**94.9**	**84.4**	**91.1**	**61.9**	**98.4**	**74.1**	**86.5**	**68.5**	**60.2**	**23.2**	**53.4**	**13.4**	**72.4**	**74.1**	**95.4**	**76.1**	**77.0**	**67.2**	**86.8**	**70.3**	**71.9**
LAPNet Large Regular	95.2	54.1	94.6	82.5	91.0	59.8	97.5	73.6	85.0	64.4	57.8	9.3	52.7	11.2	72.3	62.1	94.7	73.7	74.9	64.6	86.1	68.0	69.3

TABLE VI
COMPARISON OF LAPNET LARGE WITH DIFFERENT DECODERS ON THE SDD DATASET.

	Class IoU(%)																					MIoU
Method	**Pave.**	**Dirt**	**Gras.**	**Grav.**	**Wat.**	**Rock**	**Pool**	**Vege.**	**Roof**	**Wall**	**Win.**	**Door**	**Fen.**	**Pole**	**Per.**	**Dog**	**Car**	**Bic.**	**Tree**	**Bald**	**Mark.**	**Obst.**	**(%)**
Prop. (%)	37.6	3.3	20.0	7.2	2.3	0.7	0.7	6.9	7.5	2.7	0.6	0.1	1.0	0.1	1.1	0.1	0.8	0.2	2.1	1.4	0.2	3.5	
LAPNet Large with QVSATA	**95.9**	**55.7**	**94.9**	**84.4**	91.1	**61.9**	**98.4**	**74.1**	**86.5**	**68.5**	**60.2**	**23.2**	**53.4**	**13.4**	**72.4**	**74.1**	**95.4**	**76.1**	**77.0**	**67.2**	**86.8**	**70.3**	**71.9**
LAPNet Large with QKVSATA	95.6	54.6	94.7	83.6	**93.3**	61.0	98.1	73.9	85.9	68.1	58.2	22.4	53.3	12.5	71.7	66.2	95.3	75.8	76.6	66.2	86.7	70.1	71.1
LAPNet Large without QVSATA	94.9	54.9	94.7	81.4	90.8	60.5	98.1	73.4	85.6	63.7	52.4	1.8	53.2	7.8	71.2	59.3	95.2	75.7	74.1	65.3	85.9	68.0	68.5

that the QVSATA enhances the MIoU by 3.4%, indicating its ability to effectively integrate lower stage features, which capture local detail information, with deeper stage features, which provide rich semantic information.

To analysize the influence of omitting "Keys" in Axial Multi-head Self-attention, we also built a variant network without eliminating "Keys", denoted as LAPNet Large with QKVSATA. As presented in Table VI, LAPNet Large with QVSATA performs better than LAPNet Large with QKVSATA while has lower computational complexity.

Conventional transformer attention mechanisms introduce a substantial number of parameters that require learning during training. Consequently, larger datasets are necessary to accurately estimate these parameters and mitigate the risk of overfitting. However, in contrast to widely used general-purpose image datasets like ADE20K, existing UAV datasets are relatively small, typically comprising only several hundred images. Training a large model with limited data samples increases the likelihood of overfitting. To enhance the generalization capability of our model to unseen data, we opted to reduce the number of network parameters. Furthermore, in contrast to conventional transformers, where the QKV attention primarily functions as a feature extractor in the encoder, our QVSATA attention output serves as an importance weights matrix, or gated matrix. It dynamically reweights the features extracted at Stages 3 and 5 by performing element-wise multiplication. Consequently, the role of the Value component is not to learn feature representation but to extract deep-layer semantic relationships. Our proposition is that incorporating the Value component into token relationship computation (by eliminating Key and substituting it with Value) can enhance the modeling of this gating mechanism.

C. The Impact of Laplacian Loss

As presented in Table VII, the Laplacian Loss improves the MIoU metrics by 1.2% on the UDD dataset. It improves the accuracy of all classes, with the facade, road, and other benefiting the most. Figure 11 serves as evidence of improved accuracy for the facade class, as the Laplacian Loss enables better reconstruction of pillars and exterior walls of the buildings.

As a network-agnostic loss, the Laplacian Loss can be applied on any semantic segmentation network, therefore we conducted comparison experiments with three different networks (LAPNet Large, LR-ASPP, and BiSeNetV1) on the SDD dataset to evaluate the impact of Laplacian Loss. As presented in Table VIII, the Laplacian Loss increases the MIoU metrics of LAPNet Large, LR-ASPP, and BiSeNetV1 by 4.9%, 3.4%, and 4.7%, respectively. It also increases the accuracy of all classes, with small-scale classes (e.g., rock, door, fence, fence pole, bicycle, and AR marker) and classes with high intra-class variation (e.g., person, dog, wall, window, and obstacle) most benefitted from it. This aligns with the predicted results illustrated in Figure 12, where the Laplacian Loss better preserves the details of people, bicycles, and obstacles. Hence, the utilization of the Laplacian Loss can enhance a UAV's ability to recognize humans and identify no-fly zones during an autonomous emergency landing.

VI. DISCUSSION

Targeting the issue of scale variation caused by different flight heights, we have developed a repeatable feature extraction module that utilizes shared kernels to simultaneously and equally handle ground objects of multiple scales. Ablation analysis experiments were conducted to validate the effectiveness of the module's parallel branches and kernel-sharing mechanism. To capture long-range dependency at the lowest possible cost, we have designed an efficient axial transformer attention named QVSATA that reduces computational complexity from quadratic in terms of image size to linear in terms of image size. To realize real-time onboard semantic segmentation networks within strict hardware resource limitations, we have constructed a lightweight CNN-Transformer hybrid architecture named LAPNet. Stacked TKA modules constitute its encoder, while QVSATA serves as its decoder. When evaluated using an NVIDIA RTX 2060 Max-Q 6G Mobile GPU (FP32: 4.55 TFLOPS), which possesses comparable computational capacity to the NVIDIA Jetson AGX Orin embedded system (FP32: 5.33 TFLOPS), LAPNet Base

TABLE VII
COMPARISON OF LAPNET LARGE WITH AND WITHOUT THE LAPLACIAN LOSS ON THE UDD DATASET.

Method	Class IoU (%)						MIoU (%)
	Other	Facade	Road	Vegetation	Vehicle	Roof	
LAPNet Large with Laplacian Loss	**64.6**	**74.1**	**72.8**	**90.9**	**68.6**	**88.3**	**76.6**
LAPNet Large without Laplacian Loss	62.9	71.5	71.0	90.6	68.2	88.1	75.4

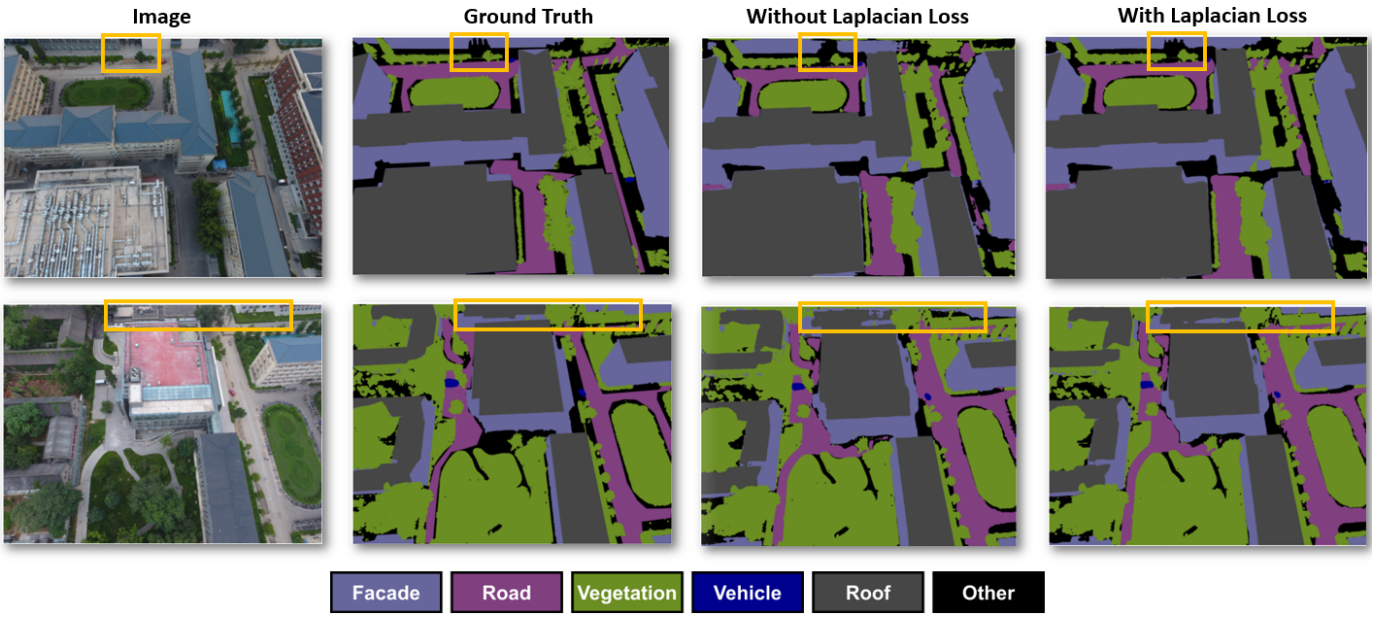

Fig. 11. A Comparison of the results predicted by LAPNet Large with and without the Laplacian Loss on the UDD dataset.

TABLE VIII
COMPARISON OF MODELS WITH AND WITHOUT THE LAPLACIAN LOSS ON THE SDD DATASET.

Method	Class IoU(%)																					MIoU (%)
	Pave.	Dirt	Gras.	Grav.	Wat.	Rock	Pool	Vege.	Roof	Wall	Win.	Door	Fen.	Pole	Per.	Dog	Car	Bic.	Tree	Bald	Mark.	Obst.
Prop. (%)	37.6	3.3	20.0	7.2	2.3	0.7	0.7	6.9	7.5	2.7	0.6	0.1	1.0	0.1	1.1	0.1	0.8	0.2	2.1	1.4	0.2	3.5	
LAPNet Large with Lap.	**95.9**	**55.7**	**94.9**	**84.4**	91.1	**61.9**	**98.4**	**74.1**	**86.5**	**68.5**	**60.2**	**23.2**	**53.4**	**13.4**	**72.4**	**74.1**	**95.4**	**76.1**	**77.0**	**67.2**	**86.8**	**70.3**	**71.9**
LAPNet Large without Lap.	95.2	54.0	94.6	83.5	90.6	56.8	97.6	73.1	85.5	64.0	48.6	3.1	44.8	8.5	68.7	55.1	93.6	69.3	74.1	64.3	83.4	65.7	67.0
LR-ASPP with Lap.	94.1	53.8	94.2	74.5	**91.5**	55.5	97.5	71.4	84.3	57.6	43.3	6.1	43.8	7.3	63.9	53.1	93.1	68.6	74.9	60.6	81.0	64.1	65.2
LR-ASPP without Lap.	91.8	52.3	93.9	72.6	90.9	49.3	96.0	70.4	83.1	54.7	38.2	1.3	38.6	2.4	63.7	39.2	92.3	62.5	72.9	56.3	77.6	60.7	61.8
BiSeNetV1 with Lap.	93.0	50.5	93.6	75.4	90.1	45.5	90.9	69.3	73.2	48.6	36.2	0.2	30.5	3.8	57.4	29.1	77.7	62.5	60.2	49.8	78.1	55.9	57.8
BiSeNetV1 without Lap.	88.2	48.3	92.7	72.8	87.8	38.9	89.0	65.6	69.3	40.6	23.2	0.0	20.1	0.5	49.1	24.9	71.8	57.0	55.6	47.0	75.8	50.0	53.1

emerges as the only network capable of real-time (15 FPS) 4K resolution image inference while maintaining commendable accuracy. LAPNet Large surpasses other lightweight networks by a substantial margin in terms of both accuracy and speed. Our further observation reveals that adding a single line of code, $torch.compile$, from PyTorch 2.0 accelerates LAPNet by approximately 10%. Moreover, optimizing LAPNet with TensorRT results in a significant speed boost, several times faster. The efficient utilization of hardware resources exhibited by LAPNet highlights its suitability for onboard deployment.

In contrast, none of the Heavyweight CNNs and Transformers can load images at their original resolution due to insufficient GPU memory. Some prior studies employed preprocessing operations, such as downsampling or cropping, to reduce the size of the original images before inputting them into the network. Nevertheless, downsampling tends to blur essential details, including the geometric shape and structural content of objects, which consequently hinders the accurate discrimination of small objects and diminishes the advantages of employing a complex architecture. The cropping of original images into small patches leads to the loss of long-range contextual information, resulting in misjudgments. Furthermore, the process of restoring the predicted results to their original size introduces additional latency. Therefore, LAPNet offers significant advantages over Heavyweight CNN and Pure Transformer models when applied for high-resolution imagery on embedded systems.

To address the challenges posed by high intra-class variation and low inter-class difference, we propose a novel loss called the Laplacian Loss. The Laplacian Loss captures complex patterns, boundaries, and small objects, and imposes an additional penalty on misclassifications in these areas, thereby encouraging the network to prioritize objects that are difficult to discriminate. The Laplacian Loss is network-

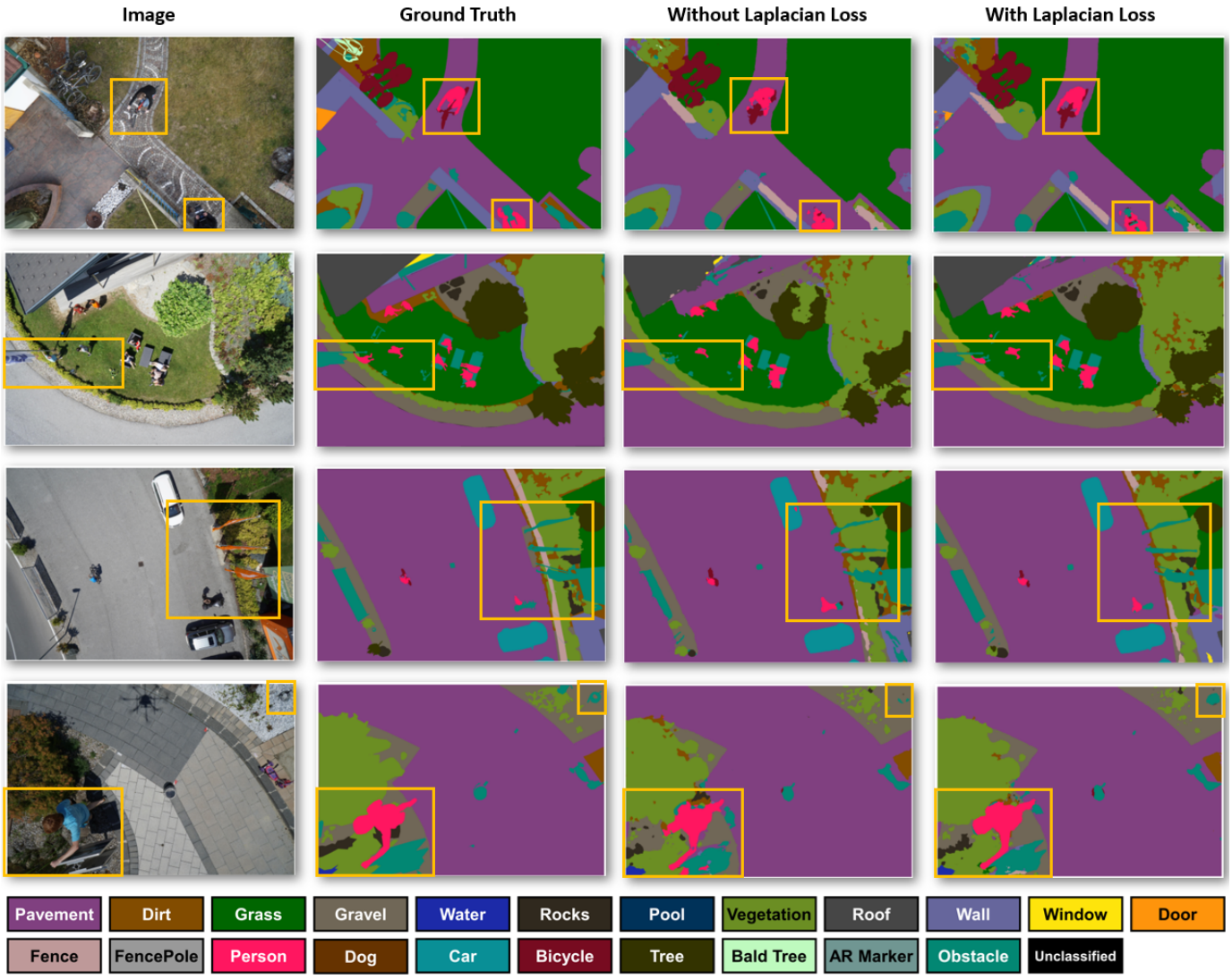

Fig. 12. A Comparison of the results predicted by LAPNet Large with and without the Laplacian Loss on the SDD dataset.

agnostic, therefore can be applied on any semantic segmentation network. Common loss functions like the cross-entropy or Dice loss can be utilized to compute the Laplacian Loss. The ablation analysis experiments demonstrate that the use of Laplacian Loss results in increased accuracy across all classes, particularly benefiting small-scale classes and classes with high intra-class variation. Visual interpretations also show that the Laplacian Loss promotes boundary reconstruction and preserves fine details of people, bicycles, and obstacles. Hence, the utilization of the Laplacian Loss can enhance a UAV's ability to recognize humans and identify no-fly zones during an autonomous emergency landing.

VII. Conclusions

Real-time onboard semantic segmentation is essential for UAV autonomous landing and various intelligent applications. However, due to hardware resource constraints, conventional semantic segmentation networks face limitations in loading high-resolution images, and existing lightweight networks suffer from significant accuracy compromises while remaining far from achieving real-time processing speed. To address this misalignment, we propose a semantic segmentation approach that offers fast inference speed, low memory usage, and favorable accuracy. Our proposed LAPNet achieves an exceptional balance between accuracy and speed, operating at least twice as fast as existing representative lightweight networks, while also achieving significantly better accuracy on three low-altitude UAV imagery datasets. These results underscore the potential of LAPNet for onboard deployment. In future research, we will investigate effective methods, including network pruning, parallel optimization, and hardware acceleration, to facilitate the adaptation and deployment on embedded systems.

Author Contributions

Conceptualization, Wen Lu, Zhiqi Zhang, and Minh Nguyen; Data curation, Wen Lu; Formal analysis, Wen Lu; Funding acquisition, Zhiqi Zhang; Investigation, Wen Lu;

Methodology, Wen Lu; Project administration, Minh Nguyen; Resources, Zhiqi Zhang; Software, Wen Lu; Supervision, Minh Nguyen; Validation, Zhiqi Zhang and Minh Nguyen; Visualization, Wen Lu; Writing – original draft, Wen Lu; Writing – review & editing, Zhiqi Zhang and Minh Nguyen. All authors have read and agreed to the published version of the manuscript.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers and members of the editorial team for their comments and suggestions. This work was supported by the National Key R&D Program of China (2022YFB3902800), the National Natural Science Foundation of China (No.61901307), Scientific Research Foundation for Doctoral Program of Hubei University of Technology (No.BSQD2020054).

Wen Lu received the B.Eng. degree in Materials Physics from Wuhan University of Technology in 2007, the M.Eng. degree in Computer Science and Technology from Hubei University of Technology in 2023. He is currently working toward the Ph.D. degree in computer and information sciences with the School of Engineering, Computer & Mathematical Sciences, Auckland University of Technology.

His research interests include computer vision, remote sensing, machine learning, and deep learning.

Zhiqi Zhang received a B.Sc. degree in Geographic Information Systems from Huazhong Agricultural University, a B.Eng. degree in Computer Science and Technology from Huazhong University of Science and Technology, an M.Eng. degree in Computer Technology from Wuhan University, and a Ph.D. degree in photogrammetry and remote sensing from Wuhan University in 2006, 2006, 2015, and 2018.

He is currently an Associate Professor with the School of Computer Science, Hubei University of Technology. His research interests include system architecture, algorithm optimization, AI, and high-performance processing of remote sensing.

Minh Nguyen received the B.Sc. degree in computer science, and the M.Sc. and Ph.D. degrees in computer vision from The University of Auckland, Auckland, New Zealand, in 2007, 2010, and 2014 respectively.

Since 2017, he has codirected the Centre for Robotics and Vision with Auckland University of Technology (AUT). Currently, he is the Head of the Department of Computer Science and Software Engineering with AUT, leading a team of 40 faculty members. His research interests include computer vision, AI, virtual/augmented reality, computer–human interaction, knowledge representation, and machine learning.

REFERENCES

[1] Z. Song, Z. Zhang, S. Yang, D. Ding, and J. Ning, "Identifying sunflower lodging based on image fusion and deep semantic segmentation with uav remote sensing imaging," *Computers and Electronics in Agriculture*, vol. 179, p. 105812, 2020.

[2] R. Gupta and M. Shah, "Rescuenet: Joint building segmentation and damage assessment from satellite imagery," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 4405–4411.

[3] W. Boonpook, Y. Tan, and B. Xu, "Deep learning-based multi-feature semantic segmentation in building extraction from images of uav photogrammetry," *International Journal of Remote Sensing*, vol. 42, no. 1, pp. 1–19, 2021.

[4] Y. Lyu, G. Vosselman, G.-S. Xia, A. Yilmaz, and M. Y. Yang, "Uavid: A semantic segmentation dataset for uav imagery," *ISPRS journal of photogrammetry and remote sensing*, vol. 165, pp. 108–119, 2020.

[5] W. Fang, L. Xiaoyan, W. Qixiong, and L. Lu, "Aerial-bisenet: A real-time semantic segmentation network for high resolution aerial imagery," *Chinese Journal of Aeronautics*, vol. 34, no. 9, pp. 47–59, 2021.

[6] C. Papaioannidis, I. Mademlis, and I. Pitas, "Autonomous uav safety by visual human crowd detection using multi-task deep neural networks," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 074–11 080.

[7] S. Nedevschi *et al.*, "Semantic segmentation learning for autonomous uavs using simulators and real data," in *2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 2019, pp. 303–310.

[8] L. Bartolomei, L. Teixeira, and M. Chli, "Perception-aware path planning for uavs using semantic segmentation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5808–5815.

[9] M. Ryll, J. Ware, J. Carter, and N. Roy, "Semantic trajectory planning for long-distant unmanned aerial vehicle navigation in urban environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1551–1558.

[10] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.

[11] C. Symeonidis, E. Kakaletsis, I. Mademlis, N. Nikolaidis, A. Tefas, and I. Pitas, "Vision-based uav safe landing exploiting lightweight deep neural networks," in *2021 The 4th International Conference on Image and Graphics Processing*, 2021, pp. 13–19.

[12] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[15] R. Lopez Lopez, M. J. Batista Sanchez, M. Perez Jimenez, B. C. Arrue, and A. Ollero, "Autonomous uav system for cleaning insulators in power line inspection and maintenance," *Sensors*, vol. 21, no. 24, p. 8488, 2021.

[16] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *arXiv preprint arXiv:1606.02147*, 2016.

[17] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, 2017.

[18] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 325–341.

[19] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation," *International Journal of Computer Vision*, vol. 129, no. 11, pp. 3051–3068, 2021.

[20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[21] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted

windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[22] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr *et al.*, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6881–6890.

[23] Y. Liu, Y. Zhang, Y. Wang, and S. Mei, "Rethinking transformers for semantic segmentation of remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, 2023.

[24] W. Zhang, Z. Huang, G. Luo, T. Chen, X. Wang, W. Liu, G. Yu, and C. Shen, "Topformer: Token pyramid transformer for mobile semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 083–12 093.

[25] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen, "Twins: Revisiting the design of spatial attention in vision transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9355–9366, 2021.

[26] Z. Huang, Y. Ben, G. Luo, P. Cheng, G. Yu, and B. Fu, "Shuffle transformer: Rethinking spatial shuffle for vision transformer," *arXiv preprint arXiv:2106.03650*, 2021.

[27] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang, "Hrformer: High-resolution transformer for dense prediction," *arXiv preprint arXiv:2110.09408*, 2021.

[28] G. U. of Technology, "Semantic drone dataset," 2020, accessed 5 July 2022. http://www.dronedataset.icg.tugraz.at.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[30] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[31] G. Li, I. Yun, J. Kim, and J. Kim, "Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation," *arXiv preprint arXiv:1907.11357*, 2019.

[32] R. Li, S. Zheng, C. Zhang, C. Duan, L. Wang, and P. M. Atkinson, "Abcnet: Attentive bilateral contextual network for efficient semantic segmentation of fine-resolution remotely sensed imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 181, pp. 84–98, 2021.

[33] Y. Hong, H. Pan, W. Sun, and Y. Jia, "Deep dual-resolution networks for real-time and accurate semantic segmentation of road scenes," *arXiv preprint arXiv:2101.06085*, 2021.

[34] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[36] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking bisenet for real-time semantic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 9716–9725.

[37] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 16 519–16 529.

[38] L. Wang, R. Li, D. Wang, C. Duan, T. Wang, and X. Meng, "Transformer meets convolution: A bilateral awareness network for semantic segmentation of very fine resolution urban scene images," *Remote Sensing*, vol. 13, no. 16, p. 3065, 2021.

[39] W. Xu, Y. Xu, T. Chang, and Z. Tu, "Co-scale conv-attentional image transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9981–9990.

[40] Y. Tang, K. Han, Y. Wang, C. Xu, J. Guo, C. Xu, and D. Tao, "Patch slimming for efficient vision transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 165–12 174.

[41] Y. Chen, X. Dai, D. Chen, M. Liu, X. Dong, L. Yuan, and Z. Liu, "Mobile-former: Bridging mobilenet and transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5270–5279.

[42] Y. Huang, Q. Wang, W. Jia, Y. Lu, Y. Li, and X. He, "See more than once: Kernel-sharing atrous convolution for semantic segmentation," *Neurocomputing*, vol. 443, pp. 26–34, 2021.

[43] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[44] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[45] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[46] Y. Yuan, J. Xie, X. Chen, and J. Wang, "Segfix: Model-agnostic boundary refinement for segmentation," in *European Conference on Computer Vision*. Springer, 2020, pp. 489–506.

[47] Z. Zhang, W. Lu, J. Cao, and G. Xie, "Mkanet: A lightweight network with sobel boundary loss for efficient land-cover classification of satellite remote sensing imagery," *arXiv preprint arXiv:2207.13866*, 2022.

[48] W. Lu and M. Nguyen, "A lightweight transformer with multi-granularity tokens and connected component loss for land cover classification," *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

[49] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[50] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[51] J. Zhuang, J. Yang, L. Gu, and N. Dvornek, "Shelfnet for fast semantic segmentation," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019, pp. 847–856.

[52] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3146–3154.

[53] M. Oršić and S. Šegvić, "Efficient semantic segmentation with pyramidal fusion," *Pattern Recognition*, vol. 110, p. 107611, 2021.

[54] R. Li, S. Zheng, C. Zhang, C. Duan, J. Su, L. Wang, and P. M. Atkinson, "Multiattention network for semantic segmentation of fine-resolution remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–13, 2021.

[55] M. Y. Yang, S. Kumaar, Y. Lyu, and F. Nex, "Real-time semantic segmentation with context aggregation network," *ISPRS journal of photogrammetry and remote sensing*, vol. 178, pp. 124–134, 2021.

[56] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 7262–7272.

[57] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 077–12 090, 2021.

[58] S. Kumar, A. Kumar, and D.-G. Lee, "Uavsnet: An encoder-decoder architecture based uav image segmentation network," *arXiv preprint arXiv:2302.13084*, 2023.

[59] W. Lu, L. Wei, and M. Nguyen, "Bi-temporal attention transformer for building change detection and building damage assessment," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2024.

[60] J. Fu, J. Liu, Y. Wang, Y. Li, Y. Bao, J. Tang, and H. Lu, "Adaptive context network for scene parsing," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6748–6757.

[61] Y. Yuan, X. Chen, X. Chen, and J. Wang, "Segmentation transformer: Object-contextual representations for semantic segmentation," *arXiv preprint arXiv:1909.11065*, 2019.

[62] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo, "Cswin transformer: A general vision transformer backbone with cross-shaped windows," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 124–12 134.

[63] G. Xu, J. Li, G. Gao, H. Lu, J. Yang, and D. Yue, "Lightweight real-time semantic segmentation network with efficient transformer and cnn," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[64] Y. Chen, Y. Wang, P. Lu, Y. Chen, and G. Wang, "Large-scale structure from motion with semantic constraints of aerial images," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*. Springer, 2018, pp. 347–359.

[65] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5693–5703.