Distortion Map-Guided Feature Rectification for Efficient Video Semantic Segmentation

Jingjing Xiong, *Student Member, IEEE,* Lai-Man Po, *Senior Member, IEEE,* Wing-Yin Yu, *Student Member, IEEE,* Yuzhi Zhao, *Graduate Student Member, IEEE,* and Kwok-Wai Cheung, *Member, IEEE*

***Abstract*—To leverage the strong cross-frame relations of videos, many video semantic segmentation methods tend to explore feature reuse and feature warping based on motion clues. However, since the video dynamics are too complex to model accurately, some warped feature values may be invalid. Moreover, the warping errors can accumulate across frames, thereby resulting in degraded segmentation performance. To tackle this problem, we present an efficient distortion map-guided feature rectification method for video semantic segmentation, specifically targeting the feature updating and correction on the distorted regions with unreliable optical flow. The updated features for the distorted regions are extracted from a light correction network (CoNet). A distortion map serves as the weighted attention to guide the feature rectification by aggregating the warped features and the updated features. The generation of the distortion map is simple yet effective in predicting the distorted areas in the warped features, i.e., moving boundaries, thin objects, and occlusions. In addition, we propose an auxiliary edge-semantics loss to implement the distorted region supervision with classes. Our network is trained in an end-to-end manner and highly modular. Comprehensive experiments on Cityscapes and CamVid datasets demonstrate that the proposed method has achieved state-of-the-art performance by weighing accuracy, inference speed, and temporal consistency on video semantic segmentation.**

***Index Terms*—Video semantic segmentation, feature warping and correction, deep neural networks, optical flow.**

I. Introduction

SEMANTIC segmentation is a fundamental and core task in computer vision. Recently, semantic segmentation has achieved unprecedented progress benefited from the prosperity of deep convolutional neural networks [1]–[4], transformer frameworks [5]–[7], etc. With the dramatically accelerating pace in the development and adoption of new technologies, single-image semantic segmentation no longer meets the requirements for some emerging applications. For instance, the rapid advancements and broad prospects of robotics, autonomous driving, and video surveillance technology have a stronger dependence on video semantic segmentation. Videos involve a much larger volume of data and rich spatial-temporal information [8]–[12]. Video semantic segmentation aims to assign dense class labels for each frame in a video. Moreover, the real-time applications put forward higher demands of efficiency on video semantic segmentation tasks.

J. Xiong, L. M. Po, W. Y. Yu, Y. Zhao are with Department of Electrical Engineering, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong SAR (e-mail: jingxiong9-c@my.cityu.edu.hk; eelmpo@cityu.edu.hk; wingyinyu8-c@my.cityu.edu.hk; yzzhao2-c@my.cityu.edu.hk).

K. W. Cheung is with School of Communication, The Hang Seng University of Hong Kong, Hang Shin Link, Siu Lek Yuen, Shatin, Hong Kong SAR (email: keithcheung@hsu.edu.hk).

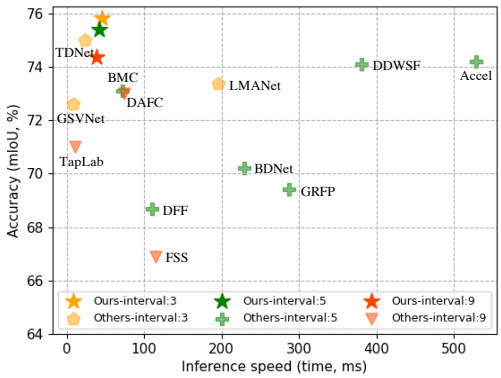

Fig. 1. Comparison of the accuracy and time of recent video semantic segmentation methods under different keyframe intervals $\{3, 5, 9\}$ on Cityscapes dataset. The pentagrams indicate our methods with various keyframe intervals. Data from Table I.

A comparison of main-stream video semantic segmentation methods is shown in Fig. 1. In general, the most straightforward way to perform video semantic segmentation is to extend the image segmentation models to video segmentation directly, treating videos as uncorrelated frames and segmenting the videos in a frame-by-frame fashion, as shown in Fig. 2 (a). However, there exist two main problems: 1) It cannot leverage the spatial continuity and temporal information in consecutive frames, which produce temporally inconsistent segmentation results on video; 2) The consecutive frames of a video share a large portion of similar spatial information. The image semantic segmentation models completely ignore the spatial consistency in videos and segment each frame individually, leading to the heavy and redundant computation. To take advantage of the spatial and temporal information inside videos, Fig. 2 (b) assumed that the high-level features of the keyframe can be reused and warped to generate the features for the next consecutive non-keyframes [13]–[15]. The feature warping process relies heavily on the estimation of motion clues, such as motion vector, optical flow, etc. However, it remains a challenging problem for authentic scene dynamics estimation. The invalid optical flow in the regions like moving thin objects, moving boundaries, occlusions may cause unsatisfactory segmentation results [13], [14], [16]–[19], which are illustrated in Fig. 3. The subsequent works follow the idea of feature warping and try to design a correction module to revise the warped features in the distorted areas [20]–[23], as shown in Fig. 2 (c). In addition, Fig. 2 (d) directly propagated the labels [24], [25], and/or correct the wrongly

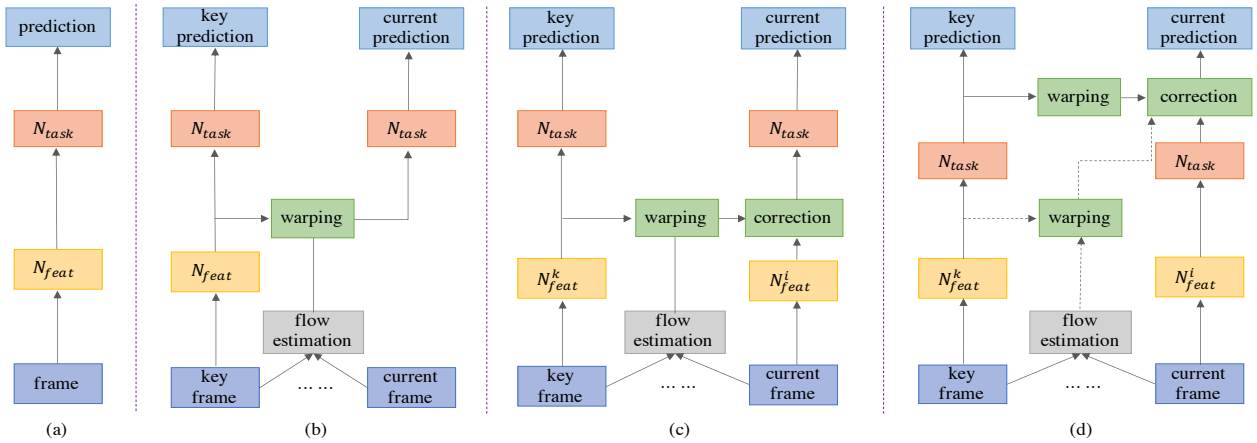

Fig. 2. Typical architectures in video semantic segmentation. (a) per-frame based video semantic segmentation. (b) feature warping-based video semantic segmentation. (c) feature warping and correction-based video semantic segmentation. (d) label warping and correction-based video semantic segmentation. N_{feat}, N_{task} represent the network for feature extraction and the task network for segmentation result, respectively. N^k_{feat}, N^i_{feat} represent the network for key-frame feature extraction and the network for non-keyframe feature extraction, respectively.

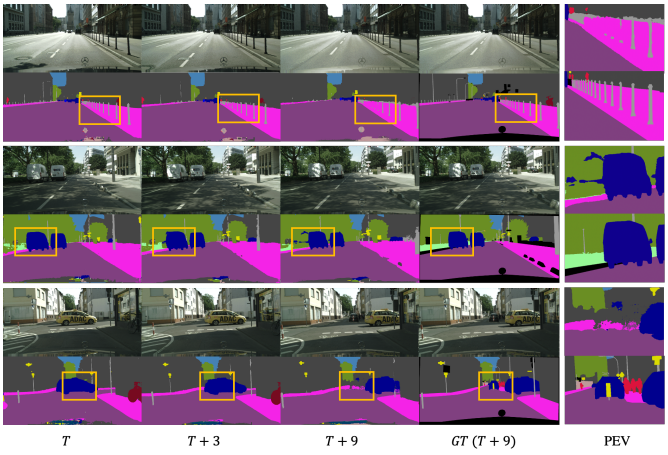

Fig. 3. Typical problems of feature warping methods caused by the inaccurate optical flow estimation in the regions of moving thin objects, moving boundaries, occlusions. "PEV" in the last column donates partial enlarged views of the images inside the yellow rectangles in $T+9$ and $GT(T+9)$.

assigned labels through a correction module [26]–[28].

In the feature correction process, a distortion map plays a key role in showing the distorted areas where the estimation of optical flow may be imprecise. Generally, authentic optical flow predictions on moving boundaries, thin objects, and occlusions are difficult, leading to the inaccurate warped features in these distorted regions. A discriminative distortion map can distinguish the distortion areas from other regions and merely guide the feature rectification on the distorted regions. For the distortion map estimation in previous works [22], [28], [29], Paul *et al.* [28] computed a probability map that represents the forward-backward inconsistencies of the optical flow as the distortion map. This distortion map is coarse since it only leverages the difference between the forward and backward propagated labels. Ding *et al.* [29] estimated two occlusion masks from a non-occluded flow branch and the segmentation maps inconsistency, and generated an error mask from the occlusion masks. This method can detect the photometric inconsistency between adjacent video frames, yet suffers from expensive computation. Zhuang *et al.* [22] proposed a separate distortion map network to predict the distorted areas of propagated features. However, the distortion map network is pre-trained before the video semantic segmentation network, and its parameters are fixed during the video segmentation training procedure, which decreases the flexibility and accuracy of the distortion map.

To improve the quality and save the computation of the distortion map, we present an efficient way to generate a trainable distortion map in a coarse-to-fine manner. Firstly, a coarse distortion map detecting the occlusions and rough moving boundaries is produced by the label inconsistency after warping. Next, an edge map that locates the boundaries and thin objects is learned from a lightweight edge & thin object perceiving module. Finally, the coarse distortion map is linearly combined with the edge map to generate a fine distortion map. In this way, the fine distortion map contains the information in both maps, and thus can distinguish the occlusions, moving boundaries, and thin objects from other regions. The aforementioned processes are extremely fast to compute, and the distortion map is updated and optimized during the video segmentation training process.

The proposed video semantic segmentation method is built upon the structure of Fig. 2 (c) - feature warping and correction. The architecture overview is illustrated in Fig. 4. For a given video sequence, the frames are divided into key and non-keyframes. A backbone network is employed to segment the keyframe and produce its feature map. Subsequently, the keyframe features are warped to the next consecutive non-keyframes based on the optical flow estimation. To obtain more accurate non-keyframe features, the core idea is to update and correct the warped features in unreliable regions while preserving the warped features for trustworthy regions. This is achieved via updated feature generation and distortion map-guided feature rectification. To be specific, a correction network CoNet is designed to extract the updated feature maps with higher attention on the distorted regions. Thereafter

a distortion map-guided feature rectification is developed to fuse the warped and the updated features and finally generate the non-keyframe prediction. The overall video segmentation procedure is summarized in Algorithm 1. We summarize the main contributions as follows:

1) We propose a distortion map-guided feature rectification method for efficient video semantic segmentation, in which a coarse-to-fine distortion map is generated to locate the regions with unreliable optical flow.

2) We propose an extremely lightweight Correction Network to focus on the feature extraction on distorted regions, together with an edge & thin object perceiving module to output an edge map to detect the boundaries and thin objects.

3) We define an edge-semantics loss to implement distorted region supervision with classes, strengthening the segmentation on distorted regions with correct label information.

4) We experimentally evaluate the effectiveness of the proposed method. The results on Cityscapes and CamVid demonstrate the improved segmentation accuracy and temporal consistency of our method compared with previous state-of-the-art methods.

II. Related Works

A. Video Cross-frame Relations

Compared to a single image, a video contains complex spatio-temporal dependence and tight cross-frame relations. Previous works can be summarized into three streams according to the way of modeling the cross-frame relations. The first one employs the motion clues between adjacent frames to encode the temporal consistency. Some works use a pretrained flow network to estimate the optical flow [13], [20]–[22], [26], [28]–[34], while some works utilize the block motion vectors presented in compressed video [14], [15], [23], [35]–[37]. The computation cost of extracting the motion vectors in the second option is cheap since the vectors already exist in a given compressed video. However, the motion vectors are coarse-grained, less precise than optical flow, and not trainable. Recently, convolutional neural network-based optical flow methods provide an effective way for the computation of flow field. Popular flow networks include FlowNet series [38], [39], PWC-Net [40], etc. The second line of works utilizes a 3D convolutional neural network (3D CNN) [8], [12], [41], [42] or a Recurrent neural network (RNN) [9], [43], [44] to learn the spatio-temporal dependencies directly. Moniruzzaman *et al.* [42] utilized 3D CNN to extract the feature maps from a video clip and learned discriminative spatial, temporal, and channel-wise features by feature representation generation. Qiu *et al.* [8] learned the temporal dependencies through 3D FCN on voxel level and employs Convolutional LSTM (ConvLSTM) on the sequential frames stream to exploit long-term temporal information. The last line of works aggregates representation of long-range temporal relations based on the attention mechanism [45]–[48]. Paul *et al.* [47] designed a memory module to aggregate semantic information from three past frames in the form of a feature maps buffer. Hu *et al.* [45] constructed an attention propagation module based on the non-local attention mechanism to reassemble features from previous frames.

B. Feature Propagation and Correction

Since the image content of consecutive frames in a video varies slowly, the high-level semantics evolve slightly across frames. Many works reuse the high-level features of a keyframe and propagate the keyframe features to the next consecutive non-keyframes. Li *et al.* [13] proposed deep feature flow to achieve fast video semantic segmentation, which is the first work to warp the previous frame features with bilinear interpolation under the guidance of the optical flow information. To further rectify the wrongly warped features due to the impreciseness of motion clues, the later works design the feature rectification module to update and correct the wrong warped features [15], [20]–[23], [32], [36]. Generally, the design of feature correction follows the idea of constructing another branch of network to extract the updated features of non-keyframes independently, then merge the features with the warped features through feature fusion. Jain *et al.* [21] and Gadde *et al.* [20] adopted simple ways to merge the updated features with the warped features, such as a fusion layer or linear combination. Feng *et al.* [15] proposed a context feature rectification module and a residual-guided attention module as correction modules to alleviate the non-rigid object deformation and error accumulation during feature warping. Xu *et al.* [32] and Feng *et al.* [36] divided the non-keyframe into several small frame regions and operate the frame regions independently. The frame region is processed by either a simple warping operation or a complex segmentation module according to the difference between a threshold and a score of the small frame region.

C. Keyframes Scheduling Strategy

Video semantic segmentation methods rely heavily on keyframes scheduling strategies. Generally, the segmentation processes for keyframes and non-keyframes are different. The computation cost of keyframes is higher than that of non-keyframes. Two keyframe scheduling strategies commonly used in video semantic segmentation include fixed keyframe selection [29], [35], [45], [47], [48], and adaptive keyframe selection [32], [34], [36], [49], [50]. In fixed keyframe selection, we choose a keyframe for every fixed number of a frame interval. For instance, given a fixed interval of 5, there will be one keyframe out of 5 consecutive frames. This kind of strategy is super-efficient since there is no time consumption required for distinguishing the keyframes. However, it sacrifices flexibility and customizability to some extent. Adaptive keyframe selection can tackle this problem by adaptively adjusting the update period of the keyframes. Xu *et al.* [32] proposed an adaptive keyframe scheduling policy by introducing a decision network to determine whether to send the image to the segmentation path or the spatial warping path according to a metric called expected confidence score. Li *et al.* [49] leveraged the low-level features to predict a deviation and set the keyframe by comparing the deviation value with a threshold. The adaptive strategy is more flexible than the fixed one, but it demands additional computation which damages the efficient video semantic segmentation. In this work, we employ the simple fixed keyframe interval strategy.

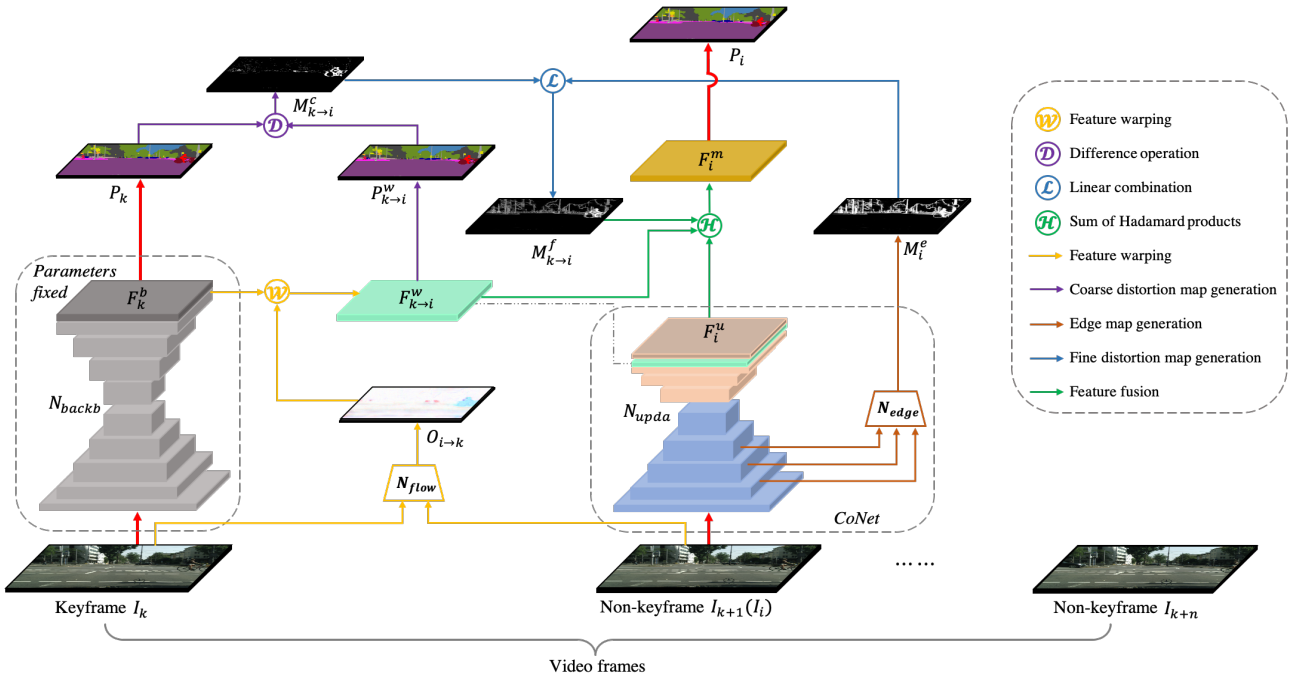

Fig. 4. An overview of the architecture of the proposed method, which is built upon the structure of Fig. 2 (c). Given a video, the video frames are divided into keyframes and non-keyframes. The segmentation prediction of the keyframe is achieved by the backbone network. The segmentation prediction of the non-keyframe is achieved by the proposed distortion map-guided feature rectification method.

Algorithm 1 Video semantic segmentation procedure

Input: Video frames $\{I_i\}$, keyframe interval n
Output: Frame segmentations $\{S_i\}$

for frame I_i in $\{I_i\}$ **do**
  if I_i is keyframe, **then**
    $k = i$
    $F^b_k = N_{backb}(I_k)$ ▷ keyframe features
    $P_k = N_{task}(F^b_k)$ ▷ keyframe prediction
    $S_i = \mathcal{F}_{up}(P_k)$ ▷ keyframe segmentation
  else
    $F^w_{k\to i} = Warp(F^b_k,\ N_{flow}(I_k,\ I_i))$ ▷ warping
    $F^u_i = N_{upda}\,(I_i,\ F^w_{k\to i})$ ▷ updated features
    $E^2_i, E^3_i, E^4_i = N_{upda-enc}\,(I_i)$ ▷ features from N_{upda} encoder
    $P^w_{k\to i} = N_{task}(F^w_{k\to i})$ ▷ warped prediction
    $M^c_{k\to i} = Diff(P^w_{k\to i}, P_k)$ ▷ coarse distortion map
    $M^e_i = N_{edge}\left(E^2_i,\ E^3_i, E^4_i\right)$ ▷ edge map
    $M^f_{k\to i} = w_f M^c_{k\to i} + (1 - w_f) M^e_i$ ▷ fine distortion map
    $F^m_i = \left(1 - M^f_{k\to i}\right) \otimes F^w_{k\to i} + M^f_{k\to i} \otimes F^u_i$ ▷ merged features
    $P_i = N_{task}(F^m_i)$ ▷ non-keyframe prediction
    $S_i = \mathcal{F}_{up}(P_i)$ ▷ non-keyframe segmentation
    $F^b_k \leftarrow F^w_{k\to i}$ ▷ iteration
    $P_k \leftarrow P_i$ ▷ iteration
    $I_k \leftarrow I_i$ ▷ iteration
  end if
end for

III. Methodology

A. Backbone Network

The keyframe $I_k \in \mathbb{R}^{W\times H\times 3}$ is fed into a backbone network N_{backb} to generate the feature maps of the keyframe F^b_k. Then, the keyframe prediction is obtained through a task network N_{task}. The processes can be written as $F^b_k = N_{backb}(I_k) \in \mathbb{R}^{W/4\times H/4\times C}$ and $P_k = N_{task}(F^b_k)$. The final segmentation map S_k is obtained by upsampling P_k to match the spatial dimensionality of the input frame given as $S_k = \mathcal{F}_{up}(P_k)$, where $\mathcal{F}_{up}$ is the bilinear upsampling. In this work, N_{task} only contains an $argmax$ operator to generate the prediction map from the features.

B. Feature Warping Representations

For the segmentation of non-keyframes, we follow the process of feature warping and correction. The "Simple" architecture from the FlowNet2 [39] is modified to obtain the optical flow field between $I_k \in \mathbb{R}^{W\times H\times 3}$ and $I_i \in \mathbb{R}^{W\times H\times 3}$. Take I_i as the next frame of the keyframe as an example, which means the index $i = k+1$. It is easy to extend to other intermediate frames (the frames between the keyframe and the next keyframe) by iteratively setting $k \leftarrow i$ and $i \leftarrow i+1$. The revised FlowNet2 N_{flow} concatenates the pairs of frames I_k and I_i as the input and computes a two-dimensional flow field $O_{i\to k} = N_{flow}(I_k,\ I_i) \in \mathbb{R}^{W/4\times H/4\times 2}$. The flow field is a quarter of the resolution of the inputs to keep the runtime small. The warped features $F^w_{k\to i}$ is spatially transformed from the keyframe features F^b_k with bilinear interpolation:

$$F^w_{k\to i} = Warp(F^b_k,\ N_{flow}(I_k,\ I_i)). \tag{1}$$

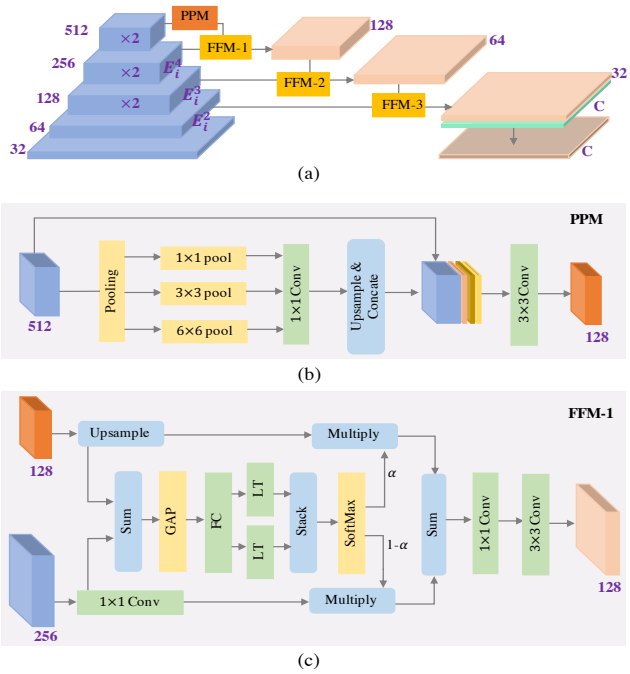

Fig. 5. The architecture of (a) the updating network N_{upda}, (b) PPM and (c) FFM-1. GAP: Global Average Pooling. FC: Fully Convolutional layer followed by a batch-normalization layer and a ReLU layer. LT: Linear Transformation. 1×1 or 3×3 Conv: 1×1 or 3×3 convolution layer followed by a batch-normalization layer and a ReLU layer. C is the total number of classes of interest, e.g., $C = 19$ in Cityscapes dataset and $C = 11$ in CamVid dataset.

C. Updated feature generation

We construct a lightweight Correction Network (CoNet), which provides additional feature information to revise the warped features in the distorted regions. It contains an updating network branch N_{upda} and an edge & thin object perceiving module branch N_{edge}. Details of the edge branch will be presented in the next subsection. The framework of N_{upda} is an encoder-decoder network designed for fast semantic feature extraction, which is illustrated in Fig. 5. The encoder of N_{upda} comprises 8 layers with 5 down-sampling operations. Thereafter, a spatial Pyramid Pooling Module (PPM) is employed to enlarge the receptive field and decrease the channel number. The decoder involves three consecutive Feature Fusion Modules (FFM), assigning soft channel attention to the features with diverse resolutions. The output resolution of the decoder is 1/4 of the original image resolution. Since the warped features $F_{k\to i}^w$ contains useful feature information from the keyframe, we also concatenate $F_{k\to i}^w$ to the decoder and obtain the updated features of the current frame F_i^u. The whole process can be written as:

$$F_i^u = N_{upda}\left(I_i,\ F_{k\to i}^w\right). \tag{2}$$

D. Distortion map-guided feature rectification

The warped features and the updated features have different properties, i.e., the warped features have stronger regional continuity, while the updated features on distorted regions are more accurate. To leverage the various information in both warped features and updated features, we learn a coarse-to-fine distortion map to distinguish the distorted regions in the warped features, and correct the unreliable warped features with the updated features. The distorted areas in this map are the combinations of occlusions, thin objects, and boundaries. Finally, the distortion map serves as weighted attention to guide the feature rectification. The whole process contains four steps:

*1) **Coarse distortion map generation**:* A coarse distortion map detecting the occlusion regions as well as rough moving boundaries is derived from comparing the keyframe prediction P_k with the warped prediction $P_{k\to i}^w$. $P_{k\to i}^w$ is obtained by feeding the warped features $F_{k\to i}^w$ to the task network N_{task} given as $P_{k\to i}^w = N_{task}(F_{k\to i}^w)$. For the pixel (x, y) where $P_{k\to i}^w(x, y)$ and $P_k(x, y)$ are different, the prediction after the warping process is inconsistent with the keyframe prediction. In consequence, this pixel position may locate the occlusions or moving boundaries. We use $Diff$ to donate the label difference between $P_{k\to i}^w$ and P_k, then the process to compute the coarse distortion map $M_{k\to i}^c$ can be represented as Eq. (3). The coarse distortion map is a binary map, which means the value in this map is either 0 or 1.

$$\begin{aligned} M_{k\to i}^c(x,y) &= Diff(P_{k\to i}^w(x,y), P_k(x,y)) \\ &= \begin{cases} 1; & if P_{k\to i}^w(x,y) \neq P_k(x,y) \\ 0; & otherwise \end{cases}. \end{aligned} \tag{3}$$

*2) **Edge map generation**:* The edge map showing the regions of edge and thin objects is generated from an edge & thin object perceiving module N_{edge}, which is modified from CE2P [51]. This module aims at learning the representation of boundaries and thin objects from the current non-keyframe. In the labeled classes, we define "pole", "pedestrian" and "bicyclist" as thin object classes. The inputs of the edge perceiving module are E_i^2, E_i^3, and E_i^4 generated from the second, third and fourth block of N_{upda} encoder (as shown in Fig. 5). Three 1×1 convolution layers and a shared 3×3 convolution are conducted to the inputs to generate three 2-channel score maps. Thereafter, a 1×1 convolution is performed to the upsampled and concatenated score maps to obtain a fused score map. Finally, a $Sigmoid$ layer is employed to the fused score map to generate the edge & thin object map, simplified as edge map M_i^e. The aforementioned processes can be written as $M_i^e = N_{edge}\left(E_i^2,\ E_i^3, E_i^4\right)$. The range of the edge map is [0, 1].

*3) **Fine distortion map generation**:* The fine distortion map $M_{k\to i}^f$ is obtained by a linear combination of coarse distortion map $M_{k\to i}^c$ and edge map M_i^e given as:

$$M_{k\to i}^f = w_f M_{k\to i}^c + (1 - w_f) M_i^e, \tag{4}$$

where $w_f \in [0, 1]$ is the weight to fuse the coarse distortion map $M_{k\to i}^c$ and the edge map M_i^e. We set w_f to 0.5 in the proposed model. The range of the fine distortion map $M_{k\to i}^f$ is [0, 1]. The larger the value in the fine distortion map, the more severe the distortion at that location. Fig. 6 presents an example to visualize these three types of maps. It can be observed that the fine distortion map highlights the regions of occlusions (behind the yellow car), moving boundaries,

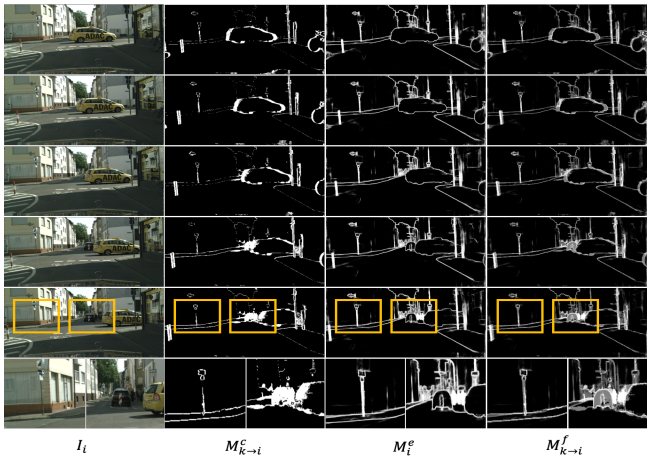

Fig. 6. An example to show different types of maps over time: coarse distortion map $M^c_{k\to i}$, edge map M^e_i and fine distortion map $M^f_{k\to i}$. The frame indexes of I_i are $k+1$, $k+3$, $k+5$, $k+7$ and $k+9$ from the top to the bottom, where k is the index of the keyframe. The last row shows the partial enlarged views of the images inside the yellow rectangles.

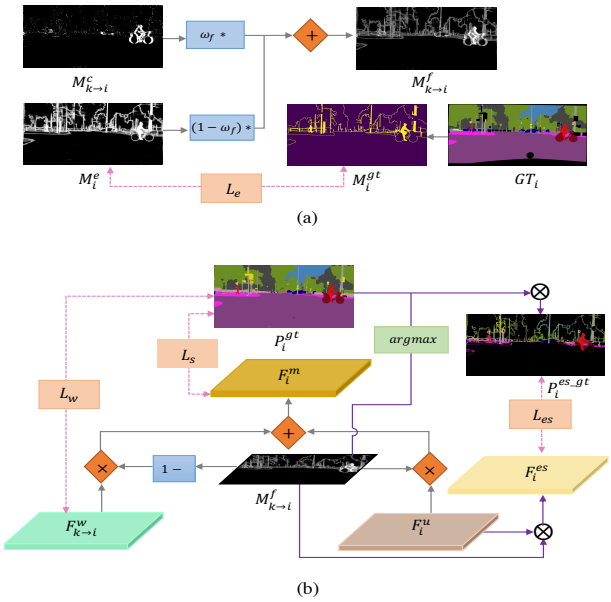

Fig. 7. Illustration of (a) linear combination of the coarse distortion map $M^c_{k\to i}$ and edge map M^e_i (b) distortion map-guided feature fusion of the warped features $F^w_{k\to i}$ and the updated features F^u_i by the sum of Hadamard products. The four loss items: warp loss L_w, semantic loss L_s, edge loss L_e, and edge-semantics loss L_{es} are displayed in the orange rectangles.

and thin objects (poles). The linear combination of the coarse distortion map $M^c_{k\to i}$ and edge map M^e_i to generate the fine distortion map is shown in Fig. 7 (a).

*4) **Feature fusion**:* The fine distortion map $M^f_{k\to i}$ is adopted as weighted attention to aggregate the warped features $F^w_{k\to i}$ and the updated features F^u_i. The merged feature F^m_i is computed as the sum of Hadamard products, which can be represented as:

$$F^m_i = \left(1 - M^f_{k\to i}\right) \otimes F^w_{k\to i} + M^f_{k\to i} \otimes F^u_i, \tag{5}$$

where $\otimes$ represents the Hadamard product. The merged features adaptively combine the information in the warped features and the updated features according to the values of $M^f_{k\to i}$. For a pixel $I(x,y)$, if $M^f_{k\to i}(x,y)$ is close to 1, it indicates this position has serious distortion occurred. The warped features are inaccurate, and the merged features $F^m_i(x,y)$ use the values in the updated features as an alternative. On the contrary, if $M^f_{k\to i}(x,y)$ is close to 0, it implies that the optical flow estimation in this region is trustworthy. The merged features $F^m_i(x,y)$ believe the feature warping process and keep the values in the warped features. The illustration of the feature fusion process is shown in Fig. 7 (b). Benefiting from the design of feature information compensation, the warped features for the authentic regions are preserved, while the warped features on the distorted regions are rectified.

The final segmentation map of the current non-keyframe S_i is obtained by feeding the merged features F^m_i to N_{task} given as $P_i = N_{task}(F^m_i)$, and upsampling P_i to match the spatial dimensionality of input frame given as $S_i = \mathcal{F}_{up}(P_i)$.

E. Objective Function

The objective function for the proposed model is presented in Fig. 7. In particular, the objective function contains four kinds of losses: warp loss (L_w), semantic loss (L_s), edge loss (L_e) and edge-semantics loss (L_{es}), which is given by:

$$L = \lambda_w L_w + \lambda_s L_s + \lambda_e L_e + \lambda_{es} L_{es}, \tag{6}$$

where λ_w, λ_s, λ_e and λ_{es} are the weights to balance the warp loss, semantic loss, edge loss and edge-semantics loss. L_w, L_s and L_{es} are implemented with the cross-entropy loss function. L_e is implemented with weighted cross-entropy loss function. L_w works on the warped features $F^w_{k\to i}$ for improving the quality of optical flow. L_s works on the merged features F^m_i for enhancing the segmentation ability of the whole model. L_e works on the edge map M^e_i to improve the performance of segmenting edge and thin objects. L_{es} works on the edge-semantics features F^{es}_i, which is obtained by multiplying the updated features F^u_i with the fine distortion map $M^f_{k\to i}$ given as $F^{es}_i = F^u_i \otimes M^f_{k\to i}$.

The edge-semantics loss is to implement the distorted region supervision with classes on F^{es}_i and strengthen the segmentation of distorted regions with correct label information. The ground truth prediction of I_i is obtained by feeding the frame I_i to the backbone network directly. This pseudo label P^{gt}_i can help CoNet learn faster and preserve the temporal consistent prediction between I_k and I_i. The true ground truth GT_i is used to generate the ground truth for the edge map M^{gt}_i. The fine distortion map is binary classified through an $argmax$ operation and is multiplied with P^{gt}_i to generate the edge-semantics ground truth $P^{es\_gt}_i$ given as:

$$P^{es\_gt}_i = argmax(M^f_{k\to i}) \otimes P^{gt}_i. \tag{7}$$

IV. Experiment

A. Implementation Details

Datasets. We use Cityscapes [52] and CamVid [53] to evaluate the performance of the proposed method. Cityscapes is a large-scale dataset for complex urban scene understanding.

It consists of 30-frame snippets shot at 17 fps. The image spatial resolution is 1024 $\times$ 2048. This dataset is divided into train, val, and test subsets with 2975, 500, and 1525 video clips. Ground truth labels are only provided for the 20-th frame in each train or val video clip. Following similar protocols of previous works [13], [15], [32], [49], we train the model on the train split and report the results on the val split. CamVid is another urban street scene dataset for video semantic segmentation, but the data is much less than Cityscapes. Moreover, the video quality and the label quality are coarser than Cityscapes, making it a challenging dataset. CamVid contains 10 minutes of footage captured at 30 fps. The image spatial resolution is 960 $\times$ 720. Following the standard train-test split of [54], we divide this dataset into train and test subsets with 468 and 233 video clips. Ground truth labels are only provided for the 30-th frame in each train or test video clip.

Training samples. For video semantic segmentation, the labeled images of the video dataset are limited. Hence, we enrich the diversity of training samples through multiple sampling. In this work, a training sample to our model is a pair of images $(I_k,\ I_i)$, where I_k is randomly selected from the nearby video clip and I_i is the video frame with the segmentation ground truth. The interval between I_k and I_i is set to $interval = i - k \in [-4, 5]$.

Training details. The proposed method consists of four sub-networks which are N_{backb}, N_{flow}, N_{upda} and N_{edge}. The parameters of N_{backb} are fixed during training. N_{flow} is pretrained on the synthetic Flying Chairs dataset [38] and then jointly trained with the randomly initialized CoNet. The backbone network N_{backb} is opened to an arbitrary semantic segmentation network. In this work, we employ CSRNet [55] as the backbone network of the keyframe because of its good trade-off between segmentation accuracy and inference speed. Specifically, CSRNet achieves a mIoU score of 76.49% on Cityscapes with a speed of 55.0 (ms) per frame, and 73.68% on CamVid with a speed of 43.2 (ms) per frame. For each step of training, we employ the Adam [56] optimizer with the initial learning rate $1e-4$ in both dataset. The batch size is set to 16. We decay the learning rate with cosine annealing strategy [57] to the minimum value of $1e-6$ in the last epoch. We employ random horizontal flip and random crop the image to a fixed size for training. We adopt the cropped resolution of 512 $\times$ 1024 for Cityscapes and 480 $\times$ 720 for CamVid. The numbers of training epochs for both Cityscapes and CamVid are 100 epochs. We conducted the experiments based on PyTorch 1.5.0 framework with two NVIDIA GeForce GTX 1080 Ti under CUDA 10.2 and cuDNN 7.6.5. Codes and models are available at https://github.com/Mayy1994/EVSS.

B. Evaluation metrics

The evaluation metrics in this work are segmentation accuracy, inference speed, and temporal consistency. For the evaluation of segmentation accuracy, the standard mean interaction-over-union (mIoU) is adopted. We select an operational keyframe interval $k \in [1, 9]$ and compute the mIoU of different intervals. The keyframe interval determines the distance between the keyframe and the labeled frame. For instance, if the 20-th frame is the labeled frame and $k = 3$, then 17-th frame is set to the keyframe and this keyframe is segmented by the backbone network N_{backb}. Afterward, we propagate and rectify the features to the 18-th, 19-th, 20-th frames and compute the IoU between the 20-th frame and the ground truth. Similarly, if the keyframe interval k is set to 9, then the 11-th frame is chosen as the keyframe. For the measurement of inference speed, since the computation costs for the keyframes and the non-keyframes are unbalanced, the average segmentation time from the keyframe to the labeled frame is employed to evaluate inference speed.

We evaluate the temporal consistency of the proposed video semantic segmentation model by computing the mIoU score between the predicted segmentation map with the warped segmentation map, following the evaluation methodology in [58], [59]. For Cityscapes dataset, 100 video sequences randomly sampled from the validation set are employed to evaluate the temporal stability. For CamVid dataset, all the sequences in "Seq05VD" are used for temporal stability evaluation. The temporal consistency (TC) is computed as:

$$TC = \frac{1}{N}\sum_{s=1}^{N}\frac{\mathcal{P}_s^w \cap \mathcal{P}_s}{\mathcal{P}_s^w \cup \mathcal{P}_s}, \tag{8}$$

where $\mathcal{P}_s^w = \{P_{1\to 2}^w,\ \ldots, P_{T-1\to T}^w\}$ and $\mathcal{P}_s = \{P_2,\ \ldots, P_T\}$ are the set of the warped prediction and final prediction. T represents the total frames in a video sequence and N is the number of the video sequences evaluated.

C. Performance

To evaluate the performance of the proposed model on video semantic segmentation, we compare the proposed method with state-of-the-art video segmentation methods, including TDNet [45], LMANet [47], GSVNet [33], DFF [13], GRFP [26], Accel [21], DDWSF [30], BMC [23], BDNet [35], FSS [14], DAFC [22], TWNet [15], TapLab [36]. All the measurements of the proposed method are conducted under the original resolution of Cityscapes and CamVid.

*1) **Segmentation accuracy**:* The comparisons of the mIoU accuracy on Cityscapes and CamVid with recent video semantic segmentation methods under various keyframe intervals {3, 5, 9} are reported in the third column of Table I and Table II. The proposed method achieves 75.92, 75.38, and 73.71 mIoU on Cityscapes dataset with 3, 5, and 9 keyframe intervals. On CamVid dataset, the proposed method outperforms the previous methods by a large margin, attaining 73.51, 72.89, and 71.26 mIoU with 3, 5, and 9 keyframe intervals. The accuracy vs. inference speed plot on Cityscapes is shown in Fig. 1. We also illustrate the comparison of accuracy vs. keyframe interval (from 1 to 9) on Cityscapes dataset with several optical flow-based warping methods, including Accel series [21], TWNet [15], FSS [14], DFF [13] in Fig. 8. As shown in this figure, the proposed method gets obvious accuracy improvements across different keyframe intervals. As the keyframe interval increases, the segmentation accuracy shows a downward trend. Among these methods, the segmentation accuracy of DFF and FSS methods drops rapidly

TABLE I
COMPARISON OF ACCURACY AND INFERENCE TIME ON CITYSCAPES. TABLE ORDERED BY PUBLISHED YEAR. "-" INDICATES THAT THE CORRESPONDING RESULT IS NOT PROVIDED BY THE METHOD. NUMBERS IN **RED** AND **BLUE** REPRESENT THE BEST AND SECOND-BEST RESULTS.

Method	Platform	Accuracy (mIoU, %) ↑	Time (ms/frame) ↓	Time-norm (ms/frame) ↓
Keyframe Interval: 3				
FSS [14]	Tesla K80	72.5	294.1	232.3
TDNet [45]	Titan XP	**75**	21	23.5
DDWSF [30]	Titan XP	74.6	400	448
TapLab [36]	GTX 1080 Ti	72.5	14.9	**14.9**
TWNet [15]	GTX 1080 Ti	73.1	-	-
LMANet [47]	RTX 2080Ti	73.37	142.8	195.6
GSVNet [33]	GTX 1080Ti	72.6	8	**8**
Proposed	GTX 1080Ti	**75.92**	45.5	45.5
Keyframe Interval: 5				
DFF [13]	Tesla K40	68.7	250	110
GRFP [26]	Titan X	69.4	470	286.7
FSS [14]	Tesla K80	70.5	204	161.2
Accel [21]	Tesla K80	**74.2**	670	529.3
DDWSF [30]	Titan XP	74.1	340	380.8
BMC [23]	-	73.1	72.5	**72.5**
TWNet [15]	GTX 1080 Ti	72.8	-	-
BDNet [35]	Tesla K80	70.2	290	229.1
Proposed	GTX 1080 Ti	**75.38**	41.8	**41.8**
Keyframe Interval: 9				
FSS [14]	Tesla K80	66.9	144.9	114.5
DAFC [22]	GTX 1080 Ti	**72.99**	74.5	74.5
TWNet [15]	GTX 1080 Ti	72	-	-
TapLab [36]	GTX 1080 Ti	71.0	10.7	**10.7**
Proposed	GTX 1080 Ti	**73.71**	38.7	**38.7**

Note: Some methods report the performance on a particular keyframe interval, while some report the performance on several intervals.

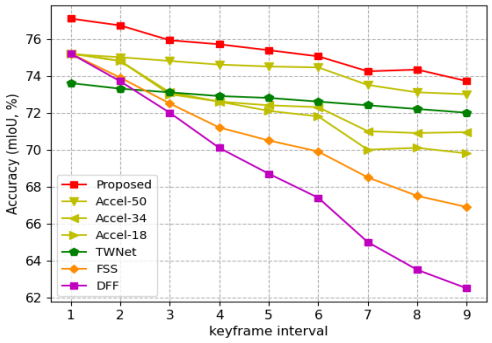

Fig. 8. Comparison of accuracy vs. keyframe interval on Cityscapes dataset with several feature warping (and correction) methods, including Accel series [21], TWNet [15], FSS [14], GRFP [26], and DFF [13].

as the keyframe increases. The reason is that DFF and FSS simply warp the features from keyframe to consecutive non-keyframes without any feature correction operations, resulting in serious accumulative warping errors and poor segmentation performance over time.

*2) **Inference speed**:* The comparisons of inference speed with recent methods on Cityscapes and CamVid are reported in the last two columns of Table I and Table II. Since the experimental environments of different methods vary a lot, we normalize the time value as the "Time-norm" based on the

TABLE II
COMPARISON OF ACCURACY AND INFERENCE TIME ON CAMVID. TABLE ORDERED BY PUBLISHED YEAR. "-" INDICATES THAT THE CORRESPONDING RESULT IS NOT PROVIDED BY THE METHOD. NUMBERS IN **RED** AND **BLUE** REPRESENT THE BEST AND SECOND-BEST RESULTS.

Method	Platform	Accuracy (mIoU, %) ↑	Time (ms/frame) ↓	Time-norm (ms/frame) ↓
Keyframe Interval: 3				
FSS [14]	Tesla K80	68.7	109.9	86.8
TDNet [45]	Titan XP	**72.6**	40	44.8
GSVNet [33]	GTX 1080Ti	64.8	4	**4**
Proposed	GTX 1080Ti	**73.51**	19.2	**19.2**
Keyframe Interval: 5				
DFF [13]	Tesla K40	66	102	**44.9**
FSS [14]	Tesla K80	**68.4**	76.3	60.3
Accel [21]	Tesla K80	67.7	239	188.8
Proposed	GTX 1080 Ti	**72.89**	18.7	**18.7**
Keyframe Interval: 9				
FSS [14]	Tesla K80	67	52.4	41.4
DAFC [22]	GTX 1080 Ti	**69.53**	25.5	**25.5**
Proposed	GTX 1080 Ti	**71.26**	18.1	**18.1**

Note: Some methods report the performance on a particular keyframe interval, while some report the performance on several intervals.

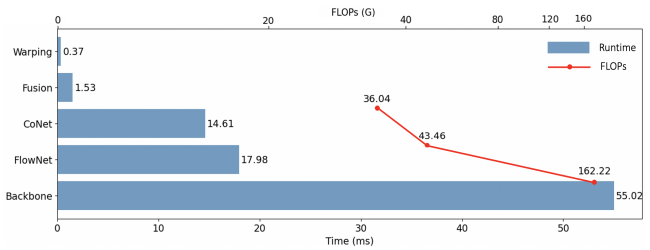

Fig. 9. Runtime and FLOPs comparison. Experiments carried on Cityscapes with spatial resolution 1024 × 2048.

GPU types for a fair comparison. Following the previous work [15], [60], the scaling factors are: 1 for GTX 1080 Ti, 0.44 for Tesla K40, 0.79 for Tesla K80, 0.61 For Titan X, 1.12 for Titan XP, and 1.37 for RTX 2080Ti. The inference time of the proposed method gradually decreases as the keyframe interval increases due to the time imbalance between segmenting keyframe and non-keyframes. The inference time to segment a non-keyframe is much less than segmenting a keyframe. The proposed method costs 45.5, 41.8, and 38.7 (ms) on Cityscapes dataset with 3, 5, and 9 keyframe intervals, respectively. On CamVid dataset, the proposed method spends 19.2, 18.7, and 18.1 (ms) with 3, 5, and 9 keyframe intervals, respectively. Compared with the GSVNet [33] and TapLab [36] which achieve fast inference time, the proposed method gains a decent trade-off between the accuracy and speed when the keyframe interval varies.

In addition to analyzing the overall inference time of video segmentation with various keyframe intervals, we also compute the average running time and floating point operations (FLOPs) of different components when segmenting video frames on Cityscapes dataset, as shown in Fig. 9. The experiment is conducted on a Ubuntu 18.04 LTS PC with Intel(R) Core(TM) i9-7900X CPU @ 3.30GHz CPU and a single GTX 1080 Ti GPU. In the proposed model,

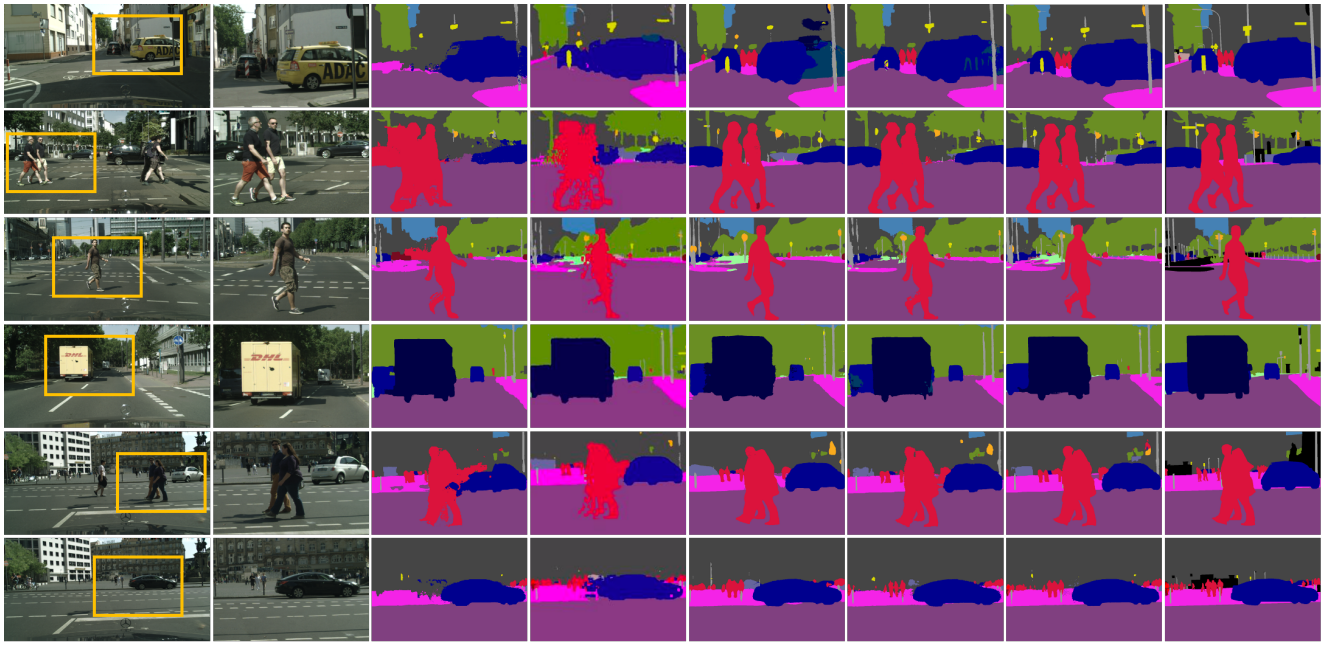

Fig. 10. Qualitative outputs on Cityscapes dataset. "PEV" donates partial enlarged views of the images inside the yellow rectangles. We compare the proposed method with state-of-the-art methods, including DFF [13], TapLab [36], GRFP [26], DAFC [22]. The images in Column 3-8 are the segmentation results inside the yellow rectangles with different methods.

TABLE III
COMPARISON OF TEMPORAL CONSISTENCY (%) ON CITYSCAPES AND CAMVID. TABLE ORDERED BY PUBLISHED YEAR. "-" INDICATES THAT THE CORRESPONDING RESULT IS NOT PROVIDED BY THE METHOD. NUMBERS IN **RED** REPRESENT THE BEST RESULTS.

Method	Backbone	Dataset	
		Cityscapes	CamVid
Per-frame inference			
MobileNetV2 [61]	MobileNetV2	68.4	76.8
PSPNet [62]	ResNet101	69.7	77.1
PSPNet18 [62]	ResNet18	68.5	-
SKD-MV2 [63]	MobileNetV2	68.2	-
SKD-R18 [63]	ResNet18	67.6	75.4
HRNet-w18 [64]	HRNet	69.1	-
PFI [58]	MobileNetV2	69.9	77.9
Multi-frame inference			
CC [65]	VGG16	71.2	-
DFF [13]	ResNet101	71.4	78.0
Accel [21]	ResNet101	70.3	76.2
DAFC [22]	DeepLabV3+	72.8	78.1
Proposed	ResNet18	**75.3**	**81.0**

we compute the runtime of (1) "Backbone" - the backbone network N_{backb} to segment the keyframe. (2) "FlowNet" - the optical flow network N_{flow} to compute the flow field between the keyframe and the next frame. (3) "Warping" - the feature warping operation $Warp$ to propagate the feature from the keyframe to the next frame. (4) "CoNet" - the updating network N_{upda} to compute the updated features and an edge & thin object perceiving module N_{edge} to generate the edge map. (5) "Fusion" - the feature fusion process to obtain the merged features and preform non-keyframe segmentation. For

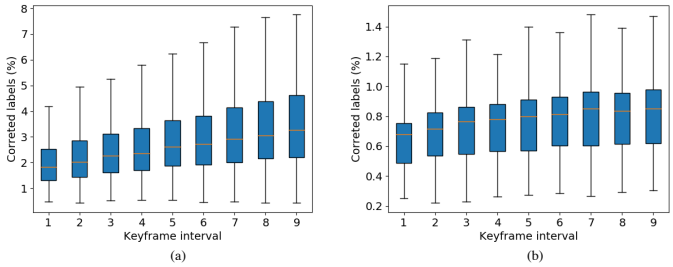

Fig. 11. The percentage of corrected labels with different keyframe intervals on (a) Cityscapes and (b) CamVid.

Cityscapes dataset, the backbone network spends 55 (ms) on segmenting the keyframe, which is the most expensive part. It is reasonable since the keyframe segmentation provides a high-quality feature reference to the following frames. The time of segmenting a non-keyframe is the runtime summation of FlowNet, Warping, CoNet, and Fusion. It takes nearly 35 (ms) on average for a non-keyframe segmentation. The inference time can be further decreased when a lighter flow network is employed. Ideally, the relationship between the overall per-frame segmentation time t and the keyframe interval k is $t = (55 + 35k)/(k + 1) = 35 + 20/(k + 1)$. The range of t is [55, 35) (ms). As the keyframe interval k increases, the average running time gets closer to 35 (ms), which is the time of segmenting a non-keyframe. The qualitative results of our framework on samples of Cityscapes are shown in Fig. 10. The proposed method can achieve more satisfactory results on occlusions, moving boundaries, and thin objects compared with other methods.

*3) **Temporal Consistency**:* The comparison of temporal consistency with previous methods on Cityscapes and CamVid is reported in Table III. The methods can be divided into per-frame inference methods, including MobileNetV2 [61], PSPNet [62], SKD [63], HRNet-w18 [64], PFI [58] and multi-frame inference methods, including CC [65], DFF [13], Accel [21], DAFC [22]. Per-frame inference methods are inferred on each frame independently, while multi-frame inference methods exploit feature warping to infer on multi frames. From Table III, per-frame inference methods produce lower temporal consistency results than multi-frame inference methods due to the abandonment of video temporal information. The proposed method outperforms the previous multi-frame inference methods by a large margin, showing the superiority in maintaining the temporal stability along with video frames.

To further quantitatively analyze the effect of the proposed feature rectification process, we also evaluate the percentage of corrected labels that are inconsistent with the labels from the warped features when increasing the keyframe interval. The percentage of corrected labels (CL) between the warped prediction $P^w_{i-1\to i}$ and the final prediction of the labeled frame P_i is computed as:

$$CL = \frac{1}{hw}\sum_{x,y} Diff(P^w_{i-1\to i}(x,y), P_i(x,y)), \qquad (9)$$

where $P^w_{i-1\to i}$, $P_i \in \mathbb{R}^{W/4\times H/4}$, $w = W/4$, and $h = H/4$. $Diff$ donates the label difference the same as the definition in Eq. (3). The average percentages of corrected labels across different keyframe intervals on Cityscapes and CamVid are shown in Fig. 11. We observe that the number of the corrected labels is directly proportional to the keyframe interval. It indicates that with the propagation of features, the model gradually distrusts the feature warping results and relies more on the updated features.

D. Visualization

To qualitatively evaluate the proposed method, we visualize the segmentation results of some samples from the Cityscapes dataset and CamVid dataset in Fig. 12. The keyframe interval is set to 9 for all samples. We extract keyframe features using the backbone network, and then propagate and rectify the features to the labeled frame I_i. The black-and-white maps shown in these two figures are coarse distortion maps, edge maps, and fine distortion maps. The last three colorful maps are the prediction maps from the warped features, the merged features, and ground truth. It can be observed that: (1) The coarse distortion map can show the regions of occlusions (see the person behind the truck in Row 2 of Fig. 12 (a), and the pedestrian behind the bicyclist in Row 1 of Fig. 12 (b)) and moving boundaries (see the moving bicyclist in Row 4 of Fig. 12 (a)). (2) The edge map precisely locates the boundaries as well as some thin objects. (3) The fine distortion map combines the information of coarse distortion map and edge map, which can represent the regions where the optical flow estimation may be incorrect. (4) The prediction from the merged features rectifies the wrong segmentation from the warped features on distorted regions. The final segmentation performances on the distorted regions like thin poles and traffic light (see Row 1 in Fig 12 (a), and Row 3/4 in Fig. 12 (b)), moving boundaries (see Row 3/4 in Fig. 12 (a), and Row 2 in Fig. 12 (b)), and occlusions (see Row 2 in Fig. 12 (a), and Row 1 in Fig. 12 (b)) are improved significantly.

E. Ablation Study

To demonstrate the effectiveness of the components in the proposed method, we define three settings to exclude some parts from the original architecture. The evaluation is performed on the original resolution on Cityscapes dataset. The quantitative results are reported in Table IV, and the experimental details are described as follows:

Network structure. In training details, we mentioned that the FlowNet is pretrained on the synthetic Flying Chairs dataset and then jointly trained with CoNet. To verify the effect of jointly training FlowNet on the entire model, we fix the FlowNet parameters during training. From Table IV, it is obvious that fixing parameters brings a significant accuracy drop, showing a decrease of 3.37% on Cityscapes and 2.73% on CamVid with 9 keyframe intervals. It proves that it is important to fine-tune the FlowNet on the video datasets. Besides, to analyze the effect of PPM, the ablation study isolates PPM while keeping other parts intact. Replacing the PPM with a simple 3×3 convolution only produces a slight decrease in accuracy (less than 1% on both datasets). We claim the reason is that the downsampling factor (64) is enough to obtain a large receptive field, leading to limited accuracy gain of adding PPM. In addition, we employ three consecutive FFM in CoNet to fuse the feature maps with various resolution by exploiting attention mechanism. To fully analyze the benefits bought by feature fusion modules, we remove some FFM and obtain lighter CoNet counterparts to compute the updated features. For Cityscapes dataset, the average running time of the original CoNet, CoNet without FFM-2, and CoNet without FFM-2 and FFM-3 is 14.61, 12.26, and 10.84 (ms), respectively. Removing the modules can decrease the inference time slightly, yet it causes varying degrees of accuracy degradation. For instance, when discarding the FFM-2 and FFM-3 simultaneously, the accuracy dropped by 1.21%.

The warped features $F^w_{k\to i}$ affects the segmentation of non-keyframes in two ways. Firstly, in CoNet, $F^w_{k\to i}$ is concatenated as a layer of N_{upda} to output the updated features. Secondly, in feature fusion process, $F^w_{k\to i}$ is adaptively combined with the updated features to generate the merged features. To validate the functionality of the warped features to the final prediction of non-keyframes, we block the effect of the warped features $F^w_{k\to i}$ on segmenting non-keyframes (including the feature concatenation and feature combination). On one hand, there is a consistent decline of around 1.2% on Cityscapes and around 2% on CamVid with different keyframe intervals when canceling the concatenation operation of $F^w_{k\to i}$ in N_{upda}. It proves the importance of the warped features in providing useful information for the generation of updated features. On the other hand, when discarding the feature concatenation and combination of $F^w_{k\to i}$ simultaneously, the experimental results show a consistent accuracy of 67.32% on Cityscapes and

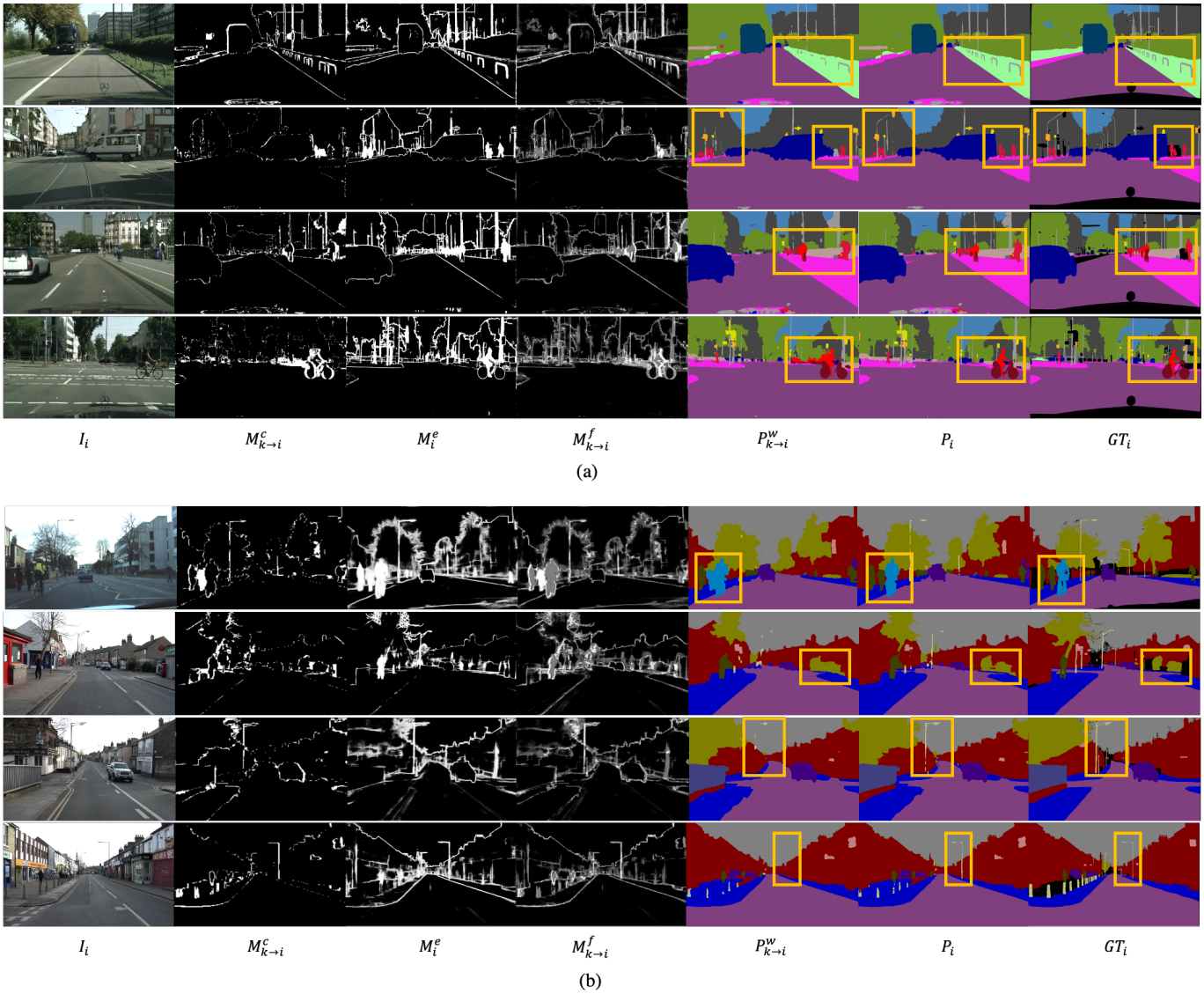

Fig. 12. Visualization of some samples on (a) Cityscapes and (b) CamVid dataset. The keyframe interval is set to 9. From left to right: labeled video frame, coarse distortion map, edge map, fine distortion map, prediction map from the warped features, prediction map after feature rectification, and ground truth.

62.46% on CamVid regardless of the keyframe interval. The reason is that without the feature concatenation and combination of $F^w_{k\to i}$, the segmentation of non-keyframes will lose the connection with the previous frames. The merged features F^m_i is obtained by multiplying the fine distortion map $M^f_{k\to i}$ with the updated features F^u_i directly. The multiplication of $M^f_{k\to i}$ does not affect the $argmax$ operator to generate the non-keyframe prediction map. Hence, essentially the non-keyframe prediction merely relies on N_{upda} which has limited layers and insufficient receptive fields. This explains why the accuracy degradation is so severe. The feature information compensation between the warped features and the updated features can produce more comprehensive merged features, and yield a promising non-keyframe segmentation.

Distortion map. In the feature fusion process, we combine the coarse distortion map and the edge map to generate the fine distortion map. In this ablation study, we carry out experiments by utilizing only one of the maps as the final distortion map to guide the feature rectification. As shown in Table IV, it causes a decrease in segmentation accuracy, especially when the keyframe interval is large. For instance, if we merely adopt the coarse distortion map, the accuracy declined by 0.97% on Cityscapes when the keyframe interval is set to 3, while the accuracy dropped by 2.1% with 9 keyframe intervals. When the keyframe interval is small, the propagation distance of feature warping is short, and the distorted areas in the warped features are relatively limited. A coarse distortion map or an edge map can also provide rough distorted regions. However, as the keyframe interval increases, the warped features contain more wrong values in moving boundaries, thin objects, and occlusions due to the imprecise optical flow estimation. Applying only one of the maps cannot detect sufficient different kinds of distorted regions in the warped features, leading to a rapid accuracy decline. In addition, the magnitude of decrease in accuracy by merely employing the edge map is smaller than that of using the coarse distortion map since the edge map

TABLE IV
ABLATION STUDY ON CITYSCAPES AND CAMVID WITH KEYFRAME INTERVALS {3,5,7,9}. NUMBERS IN **RED** REPRESENT THE BEST RESULTS.

Experimental Setting	Target	Accuracy (mIoU, %) with different keyframe intervals							
		Cityscapes				CamVid			
		3	5	7	9	3	5	7	9
FlowNet fixed	Model structure	74.58	73.15	71.47	70.34	71.86	70.75	69.49	68.53
w/o PPM	Model structure	75.87	75.18	73.91	73.58	73.15	72.33	71.19	70.48
w/o FFM-3	Model structure	75.31	74.94	74.01	73.34	73.06	72.16	71.00	70.45
w/o FFM-3 & FFM-2	Model structure	74.71	74.27	73.18	72.50	72.58	71.43	70.04	69.37
w/o concatenation of $F^w_{k\to i}$	Model structure	74.78	74.17	72.95	72.71	71.25	70.33	69.78	69.13
w/o the warped features $F^w_{k\to i}$	Model structure	67.32	67.32	67.32	67.32	62.46	62.46	62.46	62.46
w/ coarse distortion map $M^c_{k\to i}$ only	Distortion map	74.95	73.98	72.49	71.61	72.75	71.68	70.20	69.51
w/ edge map M^e_i only	Distortion map	75.80	75.02	73.67	72.80	73.18	72.40	71.18	70.57
$w_f = 0.3$ in Eq. (4)	Distortion map	75.79	75.19	74.00	73.64	73.27	72.55	71.45	71.01
$w_f = 0.8$ in Eq. (4)	Distortion map	75.57	74.82	73.55	73.14	73.08	72.08	70.88	70.27
w/o edge-semantics loss L_{es}	Objective function	75.61	74.95	73.77	73.40	73.20	72.43	71.29	70.68
w/o edge loss L_e	Objective function	75.68	75.06	73.84	73.39	73.14	72.36	71.25	70.72
w/o both L_{es} and L_e	Objective function	75.38	74.71	73.37	72.99	73.21	72.15	70.84	70.33
The proposed	Full model	**75.92**	**75.38**	**74.24**	**73.71**	**73.51**	**72.89**	**71.77**	**71.26**

contains more distorted regions information than the coarse distortion map.

In Equation (4), the hyper-parameter w_f is leveraged to balance the effect of the coarse distortion map and the edge map. w_f is set to 0.5 in the proposed method. To validate the robustness of w_f, we set the value to 0.3 and 0.8 to train the model. A higher w_f indicates that the fused distortion map relies more on the coarse distortion map and vise versa. The experiments show that assigning a smaller w_f to the coarse distortion map has similar performance on segmentation accuracy compared with $w_f = 0.5$, while setting a high w_f has caused a negative impact on segmentation accuracy. The accuracy declined by 0.07% and 0.57% over 9 keyframe intervals on Cityscapes with w_f set to 0.3 and 0.8, respectively, respectively. Since the coarse distortion map contains less distortion information than the edge map, counting on the coarse distortion map heavily may degrade the effectiveness of the final distortion map.

Objective function. We define an innovative edge-semantics loss to strengthen the supervision of distorted regions with classes. The edge loss is also introduced to supervise the learning of boundaries and thin objects. In the ablation study of objective loss, we separately drop the edge-semantics loss L_{es}, the edge loss L_e, or both the two loss items. As Table IV shows, if disabling either the edge-semantics loss or the edge loss, the accuracy decreases moderately. The downward trend on CamVid dataset is more obvious than that of Cityscapes. Moreover, dropping the loss items simultaneously aggravates the accuracy decline. The edge & thin object perceiving module classifies the boundaries into one class (0/1 classification). The edge loss only supervises the network to distinguish the edge from non-edge regions, thereby losing the information of boundary classes. The edge-semantics loss reinforces the distorted region segmentation with correct classes and recovers the important class information. Under the combined effect of boundary supervision and distorted region's class supervision, the entire network can achieve improved segmentation results on distorted regions.

V. CONCLUSION

In this paper, we present a distortion map-guided feature rectification framework for efficient video semantic segmentation. The method in this work is based on the idea of feature warping and correction. We obtain the temporal correlations between adjacent frames through an optical flow network, and then warp the features of the keyframe to the consecutive non-keyframes. The prediction from the warped features has the good property of temporal continuity. However, the segmentation of regions like occlusions, thin objects, and edges may be inaccurate due to the imprecise optical flow estimation in these regions. To correct the features in these distorted regions, we fuse the updated feature information output from a light correction network into the warped features. The feature fusion process is guided by a distortion map, which can detect the distorted regions with imprecise optical flow. Furthermore, to strengthen the segmentation ability of distorted regions with correct label information of CoNet, we introduced an edge loss and an edge-semantics loss to implement the edge supervision and the distorted region supervision with classes. Extensive experimental results on Cityscapes and CamVid demonstrate that the proposed method shows superiority in accuracy and temporal consistency over the existing methods.

REFERENCES

[1] A. Tao, K. Sapra, and B. Catanzaro, "Hierarchical multi-scale attention for semantic segmentation," *arXiv preprint arXiv:2005.10821*, 2020.
[2] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*. Springer, 2020, pp. 173–190.
[3] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, 2020.
[4] S. Borse, Y. Wang, Y. Zhang, and F. Porikli, "Inverseform: A loss function for structured boundary-aware segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5901–5911.
[5] 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," *arXiv preprint arXiv:2103.14030*, 2021.

[6] 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.

[7] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," *arXiv preprint arXiv:2102.12122*, 2021.

[8] Z. Qiu, T. Yao, and T. Mei, "Learning deep spatio-temporal dependence for semantic video segmentation," *IEEE Transactions on Multimedia*, vol. 20, no. 4, pp. 939–949, 2017.

[9] D. Li, T. Yao, L.-Y. Duan, T. Mei, and Y. Rui, "Unified spatio-temporal attention networks for action recognition in videos," *IEEE Transactions on Multimedia*, vol. 21, no. 2, pp. 416–428, 2018.

[10] S. Chandra, C. Couprie, and I. Kokkinos, "Deep spatio-temporal random fields for efficient video segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8915–8924.

[11] C.-C. Wong, Y. Gan, and C.-M. Vong, "Efficient outdoor video semantic segmentation using feedback-based fully convolution neural network," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5128–5136, 2019.

[12] N. Li, F. Chang, and C. Liu, "Spatial-temporal cascade autoencoder for video anomaly detection in crowded scenes," *IEEE Transactions on Multimedia*, vol. 23, pp. 203–215, 2020.

[13] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2349–2358.

[14] S. Jain and J. E. Gonzalez, "Fast semantic segmentation on video using block motion-based feature interpolation," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[15] J. Feng, S. Li, Y. Chen, F. Huang, J. Cui, and X. Li, "How to train your dragon: Tamed warping network for semantic video segmentation," *arXiv preprint arXiv:2005.01344*, 2020.

[16] Q. Xie, O. Remil, Y. Guo, M. Wang, M. Wei, and J. Wang, "Object detection and tracking under occlusion for object-level rgb-d video segmentation," *IEEE Transactions on Multimedia*, vol. 20, no. 3, pp. 580–592, 2017.

[17] J. Hur and S. Roth, "Iterative residual refinement for joint optical flow and occlusion estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5754–5763.

[18] J. Cheng, Y. Yuan, Y. Li, J. Wang, and S. Wang, "Learning to segment video object with accurate boundaries," *IEEE Transactions on Multimedia*, 2020.

[19] Q. Peng and Y.-M. Cheung, "Automatic video object segmentation based on visual and motion saliency," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3083–3094, 2019.

[20] R. Gadde, V. Jampani, and P. V. Gehler, "Semantic video cnns through representation warping," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4453–4462.

[21] S. Jain, X. Wang, and J. E. Gonzalez, "Accel: A corrective fusion network for efficient semantic segmentation on video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8866–8875.

[22] J. Zhuang, Z. Wang, and B. Wang, "Video semantic segmentation with distortion-aware feature correction," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[23] S. Tanujaya, T. Chu, J.-H. Liu, and W.-H. Peng, "Semantic segmentation on compressed video using block motion compensation and guided inpainting," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[24] V. Badrinarayanan, F. Galasso, and R. Cipolla, "Label propagation in video sequences," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 3265–3272.

[25] S. K. Mustikovela, M. Y. Yang, and C. Rother, "Can ground truth label propagation from video help semantic segmentation?" in *European Conference on Computer Vision*. Springer, 2016, pp. 804–820.

[26] D. Nilsson and C. Sminchisescu, "Semantic video segmentation by gated recurrent flow propagation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6819–6828.

[27] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro, "Improving semantic segmentation via video propagation and label relaxation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8856–8865.

[28] M. Paul, C. Mayer, L. V. Gool, and R. Timofte, "Efficient video semantic segmentation with labels propagation and refinement," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 2873–2882.

[29] M. Ding, Z. Wang, B. Zhou, J. Shi, Z. Lu, and P. Luo, "Every frame counts: joint learning of video segmentation and optical flow," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 10 713–10 720.

[30] M. Awan and J. Shin, "Semantic video segmentation with dense-and-dual warping spatial features," in *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, 2020, pp. 129–132.

[31] G. Guarino, T. Chateau, C. Teulière, and V. Antoine, "Temporal information integration for video semantic segmentation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8545–8551.

[32] Y.-S. Xu, T.-J. Fu, H.-K. Yang, and C.-Y. Lee, "Dynamic video segmentation network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6556–6565.

[33] S.-P. Lee, S.-C. Chen, and W.-H. Peng, "Gsvnet: Guided spatially-varying convolution for fast semantic segmentation on video," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2021, pp. 1–6.

[34] M. Awan and J. Shin, "Online keyframe selection scheme for semantic video segmentation," in *2020 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*. IEEE, 2020, pp. 1–5.

[35] H. Lu and Z. Deng, "A boundary-aware distillation network for compressed video semantic segmentation," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 5354–5359.

[36] J. Feng, S. Li, X. Li, F. Wu, Q. Tian, M.-H. Yang, and H. Ling, "Taplab: A fast framework for semantic video segmentation tapping into compressed-domain knowledge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[37] Z. Tan, B. Liu, Q. Chu, H. Zhong, Y. Wu, W. Li, and N. Yu, "Real time video object segmentation in compressed domain," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 1, pp. 175–188, 2020.

[38] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[39] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470.

[40] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943.

[41] H. Zhang, K. Jiang, Y. Zhang, Q. Li, C. Xia, and X. Chen, "Discriminative feature learning for video semantic segmentation," in *2014 International Conference on Virtual Reality and Visualization*. IEEE, 2014, pp. 321–326.

[42] M. Moniruzzaman, Z. Yin, Z. H. He, R. Qin, and M. Leu, "Human action recognition by discriminative feature pooling and video segmentation attention model," *IEEE Transactions on Multimedia*, 2021.

[43] M. Siam, S. Valipour, M. Jagersand, N. Ray, and S. Yogamani, "Convolutional gated recurrent networks for video semantic segmentation in automated driving," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–7.

[44] C. Sun, H. Song, X. Wu, Y. Jia, and J. Luo, "Exploiting informative video segments for temporal action localization," *IEEE Transactions on Multimedia*, 2021.

[45] P. Hu, F. Caba, O. Wang, Z. Lin, S. Sclaroff, and F. Perazzi, "Temporally distributed networks for fast video semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8818–8827.

[46] J. Wu, Z. Wen, S. Zhao, and K. Huang, "Video semantic segmentation via feature propagation with holistic attention," *Pattern Recognition*, vol. 104, p. 107268, 2020.

[47] M. Paul, M. Danelljan, L. Van Gool, and R. Timofte, "Local memory attention for fast video semantic segmentation," *arXiv preprint arXiv:2101.01715*, 2021.

[48] H. Wang, W. Wang, and J. Liu, "Temporal memory attention for video semantic segmentation," *arXiv preprint arXiv:2102.08643*, 2021.

[49] Y. Li, J. Shi, and D. Lin, "Low-latency video semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5997–6005.
[50] Y. Zhao, M. Dong, Y. Wang, D. Feng, Q. Lv, R. P. Dick, D. Li, T. Lu, N. Gu, and L. Shang, "A reinforcement-learning-based energy-efficient framework for multi-task video analytics pipeline," *arXiv preprint arXiv:2104.04443*, 2021.
[51] T. Ruan, T. Liu, Z. Huang, Y. Wei, S. Wei, and Y. Zhao, "Devil in the details: Towards accurate single and multiple human parsing," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 4814–4821.
[52] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.
[53] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *European conference on computer vision*. Springer, 2008, pp. 44–57.
[54] P. Sturgess, K. Alahari, L. Ladicky, and P. H. Torr, "Combining appearance and structure from motion features for road scene understanding," in *BMVC-British Machine Vision Conference*. BMVA, 2009.
[55] J. Xiong, L.-M. Po, W.-Y. Yu, C. Zhou, P. Xian, and W. Ou, "Csrnet: Cascaded selective resolution network for real-time semantic segmentation," *arXiv preprint arXiv:2106.04400*, 2021.
[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[57] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.
[58] Y. Liu, C. Shen, C. Yu, and J. Wang, "Efficient semantic video segmentation with per-frame inference," in *European Conference on Computer Vision*. Springer, 2020, pp. 352–368.
[59] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang, "Learning blind video temporal consistency," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 170–185.
[60] M. Orsic, I. Kreso, P. Bevandic, and S. Segvic, "In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 607–12 616.
[61] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.
[62] 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.
[63] Y. Liu, K. Chen, C. Liu, Z. Qin, Z. Luo, and J. Wang, "Structured knowledge distillation for semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2604–2613.
[64] K. Sun, Y. Zhao, B. Jiang, T. Cheng, B. Xiao, D. Liu, Y. Mu, X. Wang, W. Liu, and J. Wang, "High-resolution representations for labeling pixels and regions," *arXiv preprint arXiv:1904.04514*, 2019.
[65] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell, "Clockwork convnets for video semantic segmentation," in *European Conference on Computer Vision*. Springer, 2016, pp. 852–868.

Jingjing Xiong received the B.S. degree in Mechanical Design, Manufacturing and Automation from the Xiangtan University, Hunan, China, in 2015, and the M.S. degree in Artificial Intelligence and Pattern Recognition from Shenyang Institute of Automation, Chinese Academy of Sciences, Liaoning, China, in 2018. She is currently pursuing the Ph.D. degree in electrical engineering at City University of Hong Kong. Her research interests are in semantic segmentation, deep learning and computer vision.

Lai-Man Po (M'92-SM'09) received the B.S. and Ph.D. degrees in electronic engineering from the City University of Hong Kong, Hong Kong, in 1988 and 1991, respectively. He has been with the Department of Electronic Engineering, City University of Hong Kong, since 1991, where he is currently an Associate Professor of Department of Electrical Engineering. He has authored over 150 technical journal and conference papers. His research interests include image and video coding with an emphasis deep learning based computer vision algorithms.

Dr. Po is a member of the Technical Committee on Multimedia Systems and Applications and the IEEE Circuits and Systems Society. He was the Chairman of the IEEE Signal Processing Hong Kong Chapter in 2012 and 2013. He was an Associate Editor of HKIE Transactions in 2011 to 2013. He also served on the Organizing Committee, of the IEEE International Conference on Acoustics, Speech and Signal Processing in 2003, and the IEEE International Conference on Image Processing in 2010.

Wing-Yin Yu received the B.Eng. degree in Information Engineering from City University of Hong Kong, in 2019. He is currently pursuing the Ph.D. degree at Department of Electrical Engineering at City University of Hong Kong. His research interests are deep learning and computer vision.

Yuzhi Zhao (S'19) received the B.Eng. Degree in electronic information from Huazhong University of Science and Technology, Wuhan, China, in 2018. He is currently pursuing the Ph.D. degree with the Department of Electronic Engineering, City University of Hong Kong. His research interests include image processing, computer vision, deep learning and machine learning.

Kwok-Wai Cheung (M'10) received the BEng, M.Sc. and Ph.D. degrees from City University of Hong Kong in 1990, 1994 and 2001, all in Electronic Engineering. He worked in Hong Kong Telecom as engineer from 1990 to 1995. He was a research assistant at the Department of Electronic Engineering, City University of Hong Kong, from 1996 to 2002. He was an Assistant Professor at Chu Hai College of Higher Education, Hong Kong from 2002 to 2016. He has been with the School of Communication, the Hang Seng University of Hong Kong as an Associate Professor since 2016. Dr. Cheung's research interests are in the areas of image processing, machine learning and social computing.