Abstract—We have witnessed a growing interest in video salient object detection (VSOD) techniques in today’s computer vision applications. In contrast with temporal information (which is still considered a rather unstable source thus far), the spatial information is more stable and ubiquitous, thus it could influence our vision system more. As a result, the current main-stream VSOD approaches have inferred and obtained their saliency primarily from the spatial perspective, still treating temporal information as subordinate. Although the aforementioned methodology of focusing on the spatial aspect is effective in achieving a numeric performance gain, it still has two critical limitations. First, to ensure the dominance by the spatial information, its temporal counterpart remains inadequately used, though in some complex video scenes, the temporal information may represent the only reliable data source, which is critical to derive the correct VSOD. Second, both spatial and temporal saliency cues are often computed independently in advance and then integrated later on, while the interactions between them are omitted completely, resulting in saliency cues with limited quality. To combat these challenges, this paper advocates a novel spatiotemporal network, where the key innovation is the design of its temporal unit. Compared with other existing competitors (e.g., convLSTM), the proposed temporal unit exhibits an extremely lightweight design that does not degrade its strong ability to sense temporal information. Furthermore, it fully enables the computation of temporal saliency cues that interact with their spatial counterparts, ultimately boosting the overall VSOD performance and realizing its full potential towards mutual performance improvement for each. The proposed method is easy to implement yet still effective, achieving high-quality VSOD at 50 FPS in real-time applications.

Index Terms—Video salient object detection; lightweight temporal unit; fast temporal shuffle; multiscale spatiotemporal deep features

I. INTRODUCTION AND MOTIVATION

VSOD (video salient object detection) aims to extract the most visually distinctive objects in video sequences, and the existing approaches [3], [4], [5], [6] usually compute their VSODs mainly by fusing saliency cues estimated from either the spatial domain or temporal scale. Thus, the key to improving the SOTA (state-of-the-art) performance includes two aspects: 1) how to compute spatial and temporal saliency cues in a more efficient and more trustworthy way; and 2) how to fuse these two cues in a more complementary fashion. Currently, taking a single video frame as input, there exist various ISOD (image salient object detection) deep models [7], [8], [9] capable of producing high-quality spatial saliency. These off-the-shelf models can be applied to compute spatial saliency cues directly; thus, we intend to omit this beyond-the-scope content here. For the temporal saliency cues, the existing approaches [10], [11] are mainly developed on other motion sensing tools, which take the output of these tools (motion-related signals/features) as their input. At present, these tools can be categorized into three types: 1) optical flow (including FlowNet), 2) LSTM (including its variants), and 3) 3D convolution. Generally, if ranking these tools according to their motion sensing abilities, they would appear as listed above; however, this ranking is reversed when taking the efficiency as the main ranking criterion. Currently, the VSOD community widely adopts either optical flow or LSTM as the temporal sensing tool.

Let us now move to the fusion part. Compared with the static spatial domain, the temporal saliency cues usually vary from frame to frame. For example, compared with temporal information, spatial information is frequently more stable and effective in locating salient objects because the motions may become completely absent if salient objects remain static for a long period of time [10]. This fact motivates the current SOTA approaches [12], [2], [1] to concentrate more on the spatial saliency cues, leaving the temporal saliency cues as subordinates. Despite the strong performance, this methodology has two critical limitations. First, to maintain a bias towards the spatial saliency cues, the temporal information is not fully exploited. In some complex video scenes, however, the temporal information might be the only data available to derive the correct VSODs. Second, both spatial and temporal saliency cues are usually computed independently in advance and then combined later, causing the interactions between them to be partially or completely overlooked, resulting in low-quality saliency cues.

To facilitate a better understanding, we provide a pictorial demonstration in Fig. 1, which divides the SOTA VSOD approaches into three categories, with the middle two columns illustrating the two most representative architectures.

As shown in subfigure B (MGA [1]), the temporal saliency clues are directly derived via the optical flow; then, these clues, belonging to different temporal layers, serve as the
A demonstration of the structural differences between the proposed model and other SOTA methods. Notice that since this is a VSOD (video salient object detection) paper, we focus only on the deep-learning-based VSOD models here, omitting the video fixation detection approaches. Subfigure A demonstrates the early classic bistream structure. Subfigures B and C are the two currently most representative structures (e.g., B: [1] and C: [2]), which bias their spatiotemporal saliency fusion towards their spatial branches. Sub-Figure D is our novel structure, which is capable of making “full use” of the spatial information when computing temporal saliency cues, realizing the multiscale spatiotemporal saliency interaction; this is empowered by the novel temporal sensing unit, a simple unit based on 3D convolution.

As shown in subfigure C ([12], [2]), this type of architecture is clearly optical-flow-free; instead, it newly adopts the end-to-end convLSTM, a convolutional variant of the classic LSTM, to sense the temporal saliency cues. This architecture is mainly developed on a spatial encoder, followed by a convLSTM that takes the output of the spatial encoder as its input, of which the key rationale is to realize the spatiotemporal saliency fusion by filtering those spatial saliency cues with weak consistency over the temporal scale. Despite its high computational speed, this architecture has one critical limitation, i.e., its temporal subbranch cannot make full use of the multiscale spatial deep features. Consequently, with limited spatiotemporal interactions, the saliency cues obtained in either the spatial or temporal branch are far from adequate in general. Moreover, the usage of convLSTM may result in a bottleneck, because the convLSTM itself is neither lightweight nor strong enough to sense motion.

This paper aims to tackle all the aforementioned limitations via a completely new temporal unit, of which the major advantages include its extremely lightweight design and its ability to provide the multiscale spatiotemporal interaction.

The proposed temporal unit is essentially developed on a 3D convolution. Although the plain 3D convolution is very efficient, its ability to sense motion is rather weak compared with other tools (e.g., optical flow). Therefore, we propose a novel spatiotemporal network, in which the encoder is directly obtained from other off-the-shelf 2D ISOD deep models. As the key innovation, the decoder is implemented mainly using 3D convolutions. To improve the motion sensing ability of the 3D convolution while still being lightweight, a series of 3D convolutions (Sec. III-B), coupled with a temporal shuffle (Sec. IV-B) and fast padding operations (Sec. IV), are inserted into each decoder layer. In addition, to make full use of the spatial saliency cues provided by the encoder, each encoder layer is connected to its corresponding decoder layer. Thus, benefiting from the multiscale saliency cues provided by the encoder, the decoder’s problem domain can be reduced significantly. As a result, the temporal saliency cues can be derived more easily. Specifically, from the perspective of technical implementation, it is very challenging for the widely-used convLSTM to realize this multiscale spatiotemporal interaction. In contrast, this can be realized both naturally and easily using our new approach.

The proposed spatiotemporal network also enables the errors from the decoder (temporal) to be propagated to the encoder (spatial); thus, it can well avoid the performance bottleneck that exists in the current SOTA models [1], [12], [2], the overall performances of which are mainly determined by the spatial saliency cues. In sharp contrast, our model can improve the quality of the saliency cues by interacting its spatial branch with messages fed back from its temporal decoder in a multiscale manner. We demonstrate the major differences between the proposed model (Fig. 1-d) and other most representative competitors in Fig. 1 (a-c).

Compared with other approaches, the proposed spatiotemporal network with 3D convolutions has three prominent advantages:

- Instead of using the time consuming optical flow to acquire temporal information, we have devised a novel temporal unit, which is very fast and compatible with the end-to-end encoder-decoder structure, achieving high-quality VSOD at 50 FPS and thus being suitable for real-
time applications;

• The proposed novel temporal unit can be integrated into each decoder layer easily, making full use of the multiscale spatial saliency-aware deep features, ensuring the decoder’s advantage in computing temporal saliency cues;

• Because of the high-quality temporal saliency cues empowered by the new temporal unit, the overall quality of the spatial saliency cues can also be boosted mutually by their interactions with that of the decoder, further enhancing the overall VSOD performance.

II. RELATED WORKS

A. Handcrafted-Feature-Based Methods

Conventional methods estimate the temporal information mainly by computing the contrast over various handcrafted features. Liu et al. [14] computed a superpixelwise spatiotemporal contrast histogram to sense the temporal information. Similarly, Liu et al. [4] proposed the intraframe similarity matrices to perform bidirectional temporal propagation as the temporal information. Wang et al. [15] acquired the spatiotemporal feature by computing a robust geodesic measurement for locating spatial edges and motion boundaries. Wang et al. [3] adopted the gradient flow field to obtain intraframe boundary information as well as interframe motion information. Further, Chen et al. [16], [17] adopted the long-term information to enhance the spatiotemporal saliency consistency.

B. Deep-Learning-Based Methods

With the rapid development of deep learning techniques, the current SOTA deep models have widely adopted the bistream network structure, in which one of the streams computes the color saliency over the spatial domain and the other stream extracts the motion saliency over the temporal scale. Wang et al. [18] proposed the use of a 2D convolution network to sense the differences between two adjacent frames to sense temporal information, of which the rationale is quiet similar to that behind the deep-learning-based optical flow computation (e.g., FlowNet). Le et al. [19] further extended this idea by adopting the bistream structure, where one of the streams computes the superpixelwise spatial saliency cues and the other one computes the temporal saliency cues by applying the 3D convolution directly over multiple video frames. Though 3D convolution can provide some temporal saliency cues, its bilevel fusion scheme cannot avoid the critical limitation of 3D convolution, i.e., the boundaries of the salient objects tend to be obscured due to spatial misalignments when performing the 3D convolutions. To further improve the quality of the temporal saliency cues, Song et al. [12] adopted several dilated convLSTMs to extract multiscale spatiotemporal information and feed it to another bidirectional convLSTM to obtain the spatiotemporal information.

Visual fixations can also be used to facilitate the VSOD task. Wang et al. [20] and Fan et al. [2] adopted the human visual fixations to enable attention shifting between different salient objects. Li et al. [21] adopted the FlowNet-based optical flow to sense the temporal information, which was then used to enhance the saliency consistency over the temporal scale by using the newly designed gate unit. The major highlight of this gate unit is its ability to filter those less trustworthy temporal saliency cues. Further, Li et al. [1] developed a multitask network. As typical, it adopted the optical flow to sense temporal movements, of which the major difference is that it utilizes its temporal branch to complement the spatial branch, achieving a significant performance improvement. In [1], though the temporal branch is able to interact with the spatial branch, the opposite interaction is overlooked, and the color saliency obtained by the spatial branch can affect the temporal branch positively in practice. Further, Ren et al. [22] integrated the gate unit mentioned in [1] into the convLSTM-based end-to-end network.

Most recently, the long-term spatiotemporal information was revisited with consideration of VSOD. Chen et al. [5] developed a novel long-term patchwise alignment approach for estimating the long-term spatiotemporal constraint, which is used to facilitate salient object detection in short-term video contents. Similarly, Wang et al. [23] proposed the GCN (graph convolutional network)-based VSOD model, which is capable of sensing temporal saliency cues in the long-term in an end-to-end manner.

III. THE PROPOSED SPATIOTEMPORAL NETWORK

The classic UNet [13] follows the typical encoder-decoder network structure, which has a remarkable learning ability to conserve high-level semantic information embedded in the pre-trained backbone (i.e., the encoder) while making full use of its decoder to provide fine, tiny details. Thus, we choose it as the baseline network for the proposed spatiotemporal network with several critical modifications, which are introduced and explained in the following sections.

A. Combining 2D Convolutions with 3D Temporal Units

In the CV (computer vision) field, there exist two approaches for the UNet-based deep models to compute the spatial and temporal features: the 2D convolution-based UNet and the corresponding 3D version. In brief, the former is characterized by its basic computational unit, of which the 2D kernels are used for sensing spatial deep features in a single image/frame, while the temporal information is completely omitted. Since the 2D-based UNet can well preserve tiny spatial details, it has been widely used in many segmentation-related tasks, such as the salient object detection that we are interested in.

In sharp contrast, the latter is mainly developed on 3D convolutions, and the additional third dimension in the 3D kernel is used to perform a convolution over the temporal scale. Thus, the purely 3D convolution-based UNet can simultaneously sense both spatial and temporal information, and thanks to this spatiotemporal attribute, it has been widely used in various recognition tasks, such as for motion recognition. Despite the merit, the plain 3D convolution-based UNet has multiple critical drawbacks, making it unsuitable for the segmentation task. First, to strive for some specific learning objective, the 3D convolution usually needs to make a trade-off between the
Fig. 2: The network architecture overview of our spatiotemporal network. We follow the vanilla UNet [13] encoder-decoder structure (Sec. III). The major highlight of this network is its ability to allow a full interaction between spatial (encoder) and temporal (decoder) saliency cues in a multiscale manner, which the main-stream SOTA models based on convLSTM cannot achieve. It may be noted that this advantage is mainly realized by the newly proposed temporal unit, which consists of sequential 3D convolutions with a fast cyclic padding scheme (3D, Sec. IV-A) and temporal shuffle scheme (S, Sec. IV-B). These two modifications ensure that our temporal branch becomes stronger in sensing temporal changes than the ordinary 3D convolution. This novel temporal unit also takes the multiscale spatial deep features as its input; thus, it can sense temporal information much more easily. \{U,C,+\} respectively denote the feature upsample, feature concatenation, and feature elementwise summation.

spatial and the temporal; thus, when performing an end-to-end segmentation task, its final output often has blurry object boundaries. Second, compared with the 2D version, the 3D-based UNet is usually applied for tasks with more challenges; however, there are fewer data available for the 3D training, as in each training iteration, the 3D version needs to be fed multiple images/frames, while the 2D version requires a single image only. Additionally, in light of the fact that most of the 2D SOTA ISOD models can already produce high-quality spatial saliency cues, we propose combining 2D convolutions with 3D convolutions.

To be more specific, we take an off-the-shelf 2D ISOD model as the baseline (i.e., plain UNet trained with SOD data), in which we assign a novel temporal unit, as fully explained in Sec. IV, to each decoder layer, aiming to convert this 2D model to one that is capable of sensing spatiotemporal saliency cues. Next, we give a full investigation of the interaction between these two types of convolutions, an aspect that has long been omitted by other works due to the various limitations of their network designs.

### B. Multiscale Spatiotemporal Interaction

In the case of a single image, the high-level semantic information embedded in the encoder tends to decrease with the increase in the number of decoder layers; thus, the widely used scheme to solve this issue is to align each encoder layer with each decoder layer accordingly.

When dealing with video data, the proposed spatiotemporal network (Fig. 2) takes three video frames as its input each time. For each encoder layer, we represent its feature block as $F_{i,j}$, $i \in \{1, 2, 3\}$, where $j$ is the encoder layer index and $i$ indicates each of the three input frames.

As mentioned before, we respectively assign one temporal unit, marked by the yellow rectangle, to each decoder layer. Each temporal unit takes the spatial deep feature $F_{i,j}$ as its input, of which the key rationale is to conserve spatial saliency cues when computing the temporal saliency cues; thus, the output of each temporal unit is spatiotemporal-aware.

Thus, to make full use of the multiscale spatial deep features, the input of each temporal unit should also include the deep features derived in the preceding scale $(R)$. Therefore, taking the 3rd encoder layer for instance, we concatenate $F_{i,3}$ with $R_{i,2}$ in advance (see Fig. 2). Moreover, each temporal unit should also make full use of the spatial attentions, because these attentions can serve as the high-level localization information to enforce each decoder layer to focus on those spatially salient regions. We denote the real input of each temporal unit as $T_{i,j}$ (see Eq. 1):

$$T_{i,j} = Conv(U(A) \otimes Conv(R_{i,2} \otimes F_{i,3})),$$

(1)

where $Conv$ denotes the feature convolution operation, $\otimes$ is the feature concatenating operation and $U$ denotes the upsampling operation. The spatial attention maps $(A)$ are computed by applying the dilated convolutions with dilation factor $d = \{0, 2, 4, 6\}$ over the spatial deep features provided by the last encoder layer.

We denote the output of each temporal unit as $ST_{i,j}$ and, taking the 3rd decoder layer for instance, the computation of $ST$ can be formulated as

$$ST_{i,3} = TU(T_{i,3}),$$

(2)

where $TU$ denotes the temporal unit.

The spatiotemporal deep features ($ST$) then serve as another attention to enhance the original input of the current temporal unit (i.e., $T$), converting $T$ to a spatiotemporal-aware state while avoiding the spatial information loss during the carrying
out of 3D convolutions in a temporal unit. Thus, the final output of the 2nd decoder layer can be expressed as

\[ R_{i,2} = U \left( ST_{i,2} + \text{Conv} (\text{Conv}(F_{i,2} \otimes R_{i,1}) \otimes U(A)) \right). \] (3)

In this way, we have achieved the full interaction status between the spatial branch (i.e., the encoder layers of UNet) and the temporal branch (i.e., the decoder layers of UNet). In the next section, we provide full introductions and explanations regarding the proposed temporal unit.

IV. THE PROPOSED TEMPORAL UNIT

A. Fast 3D Convolution with a Strong Temporal Sensing Ability

It is well known that the conventional 2D convolution using a flat 2D kernel (e.g., \(3 \times 3\)) can sense only spatial information in a single image/frame, while, benefiting from the additional third dimension, the deep features computed by the 3D convolution with a cubic kernel (e.g., \(3 \times 3 \times 3\)) can be temporal-aware.

The proposed temporal unit is developed on the basic 3D convolution; yet, we have made several modifications to make it more suitable for the VSOD task. In theory, the 3D convolution cannot only sense temporal information but also spatial information. However, in the view of temporal sensing ability, the 3D convolution is inferior to the existing temporal sensing tools (e.g., optical flow).

As mentioned in Eq. 2, the input data of each temporal unit consist of three parts: the multiscale spatial deep features \((T)\), the attention maps \((A)\), and the recurrent data from the precedent decoder layer \((R)\). We use \(T_{i,j}, i \in \{1, 2, 3\}\) to denote the input of the temporal unit in the \(j\)-th decoder layer. To compute the temporal saliency cues, we use the sliding window scheme to apply the 3D convolution \((\text{Conv3D})\) over \(T_i\); thus, the output \((ST_{i,j})\) of each temporal unit can be quickly computed via Eq. 4:

\[ ST_{i,j} = \begin{cases} \text{Conv3D}(pad_1, T_{1,j}, T_{2,j}) & \text{if } i = 1 \\ \text{Conv3D}(T_{1,j}, T_{2,j}, T_{3,j}) & \text{if } i = 2 \\ \text{Conv3D}(T_{2,j}, T_{3,j}, pad_2) & \text{if } i = 3 \end{cases}, \] (4)

where \(pad\) denotes the padding data.

However, regarding VSOD, temporal information often plays an important role in locating salient objects; this issue was well explained in Sec. I. Thus, we propose a simple scheme to further improve the temporal sensing ability of the proposed temporal unit. That is, instead of using a single 3D convolution each time, we advocate sequential 3D convolutions as the basic computational unit of the encoder. In our implementation, we empirically use three 3D convolutions each time, in which the 2nd 3D convolution takes the output of the 1st 3D convolution as its input, and so on. The spatiotemporal deep features of the 2nd 3D convolution can be updated by Eq. 5:

\[ ST_{i,j}^2 \leftarrow \begin{cases} \text{Conv3D}(pad_3, ST_{1,j}^1, ST_{2,j}^1) & \text{if } i = 1 \\ \text{Conv3D}(ST_{1,j}^1, ST_{2,j}^1, ST_{3,j}^1) & \text{if } i = 2 \\ \text{Conv3D}(ST_{2,j}^1, ST_{3,j}^1, pad_4) & \text{if } i = 3 \end{cases}. \] (5)

Though using multiple 3D convolutions can really enhance the temporal sensing ability of the encoder, this implementation may result in two other problems: 1) since the spatial information between different frames is generally misaligned when performing a 3D convolution, the direct usage of sequential 3D convolutions easily degenerates the corresponding deep features in terms of preserving tiny details, eventually blurring the object boundaries; 2) the widely used zero padding operation, an indispensable step in each 3D convolution, will cause the sequential 3D convolutions to be problematic because the zero padding scheme degenerates those deep features of video frames in either end.

To handle the first problem, we simply add each \(ST\) to the deep features computed by the 2D convolutions, e.g., \(ST_{1,j}^1 \leftarrow \{ST_{1,j}^1 + \text{Conv}(T_{1,j})\}, \ ST_{2,j}^2 \leftarrow \{ST_{2,j}^2 + \text{Conv}(ST_{1,j}^1)\}\).

To handle the second problem, we resort to using the novel cyclic padding scheme:

\[ \{pad_1, pad_2, pad_3, pad_4\} \leftarrow \{T_{3,j}, T_{1,j}, ST_{1,j}^1, ST_{3,j}^3\}. \] (6)

In fact, the cyclic padding is time consuming if we realize it by reorganizing the input data as in Eq. 6. To improve this process, we directly use the repeating operation on a GPU three times to expand the original \(\{T_1, T_2, T_3\}\) to \(\{T_1, T_2, T_3, T_1, T_2, T_3, T_1, T_2, T_3\}\); thus, the cyclic padding can be fulfilled by using a sliding window over these expanded features, and this implementation is almost five times faster than the conventional feature reorganization (see the example in Eq. 7).

\[ \text{repeat}\{T_1, T_2, T_3\}, 3 \rightarrow \{T_1, T_2, T_3, T_1, T_2, T_3, T_1, T_2, T_3, \ldots\}. \]

\[ ST_{1,j}^3 = \text{Conv3D}(\ldots) \] (7)

In essence, this fast cyclic padding scheme kills two birds with one stone:
It cyclically uses deep features of other frames as the input and output of three consecutive video frames, of which the input deep features (64 channels) pass through each other by performing the proposed fast temporal shuffle scheme.

1) It avoids the performance degradation induced by the conventional zero padding when using sequential 3D convolutions;
2) It cyclicly uses deep features of other frames as the padding data, enhancing the temporal ability naturally without additional computational costs. In other words, this padding scheme ensures that the proposed temporal unit is fast and strong in temporal sensing.

We present the complete data flow of our temporal unit in Fig. 3.

B. Fast Temporal Shuffle

The temporal information sensed by the 3D convolutions is mainly derived from the 3rd dimension of the adopted 3D kernels (e.g., “Spatial”;{3 × 3}×“Temporal”;{3}), which may lead the final spatiotemporal deep features to be biased towards the spatial domain.

To further improve this situation, we propose a fast temporal shuffle scheme, which is inspired by the ShuffleNet [24]. It enhances the spatial feature diversity by randomly scrambling the feature orders. Thus, the proposed temporal shuffle scheme further enhances the temporal sensing ability by swapping deep features between consecutive video frames. The motivation of swapping features across different frames is to enhance the ability of the precedent 3D convolutions, because simply using the proposed 3D fast convolution solely tends to degenerate the contribution that the temporal part should bring.

For example, as shown in Eq. 8, we swap the deep feature $f_a$ in frame #1 (i.e., $ST_1$) with the deep feature $f_b$ in frame #2 (i.e., $ST_2$), and we swap $f_c$ with $f_d$.

$$ST_1\{f_a, f_b, f_c, f_d\} ← ST_2\{f_b, f_a, f_d, f_c\}.$$  (8)

Specifically, the proposed temporal shuffle scheme can be fully implemented on a GPU, thus being an extremely simple and fast plug-in; see the pictorial demonstration in Fig. 4.

The technical details can be summarized as follows: first, we sequentially divide the original 192 × 1 deep features into 64 groups, with each group including 3 deep features; next, we reshape the original 192 × 1 deep features to $64 \times 3$ according to their group orders; and then, we transpose them to $3 \times 64$ and flatten them back to $192 \times 1$ deep features. In this way, we automatically insert the temporally neighbored spatial features into the current frame.

In our implementation, we repeat the above temporal shuffle twice, i.e., once for the output ($ST_1$) of the first 3D convolution and again for the output ($ST_2$) of the second 3D convolution. Thus, the complete dataflow of our temporal unit can be represented by Eq. 9:

$$ST ← Conv3D(S(Conv3D(S(Conv3D(T))))).$$  (9)

where $T$ and $ST$ respectively denote the input and output of the temporal unit and $S$ represents the temporal shuffle operation.

The novelty of the proposed temporal shuffle could be summarized as below. Compared with the ShuffleNet that swaps single-scale features in a single frame, our temporal shuffle is the first attempt to perform the fast swapping operation for features belonging to different frames. Thus, our approach is able to achieve stronger ability in sensing temporal information. Meanwhile, our temporal shuffle is able to serve as a plug-in in the proposed multi-scale spatiotemporal network. Consequently, the multi-scale information embedded in previous scales can also be implicitly involved into the current stage.

V. QUANTITATIVE EXPERIMENTS

A. Dataset Introduction

To evaluate the performance of our method, we conducted extensive quantitative evaluations over six widely used public benchmark datasets, including DAVIS-T [30], SegTrack-V2 [31], Visal [3], FBMS-T [32], VOS-T [33] and DAVSOD-T [2].

B. Evaluation Metrics

Following other SOTA works, we employed 3 widely used metrics in the quantitative comparisons and component evaluations: the F-measure [34]; the structure measure (S-measure) [35]; and the mean absolute error (MAE) [36].

C. Training Set Details

Since the VSOD task requires much more training data than the conventional ISOD (image salient object detection [37], [38]) task, previous works [1], [12], [2] have followed the stagewise training protocol as follows: pretrain the target VSOD model using image data first and then fine tune it using video data.

As shown in Table II, we summarized all the details regarding the training schemes adopted by the current SOTA methods. Our model is pretrained using 9.5K image data selected from the DUTOMRON (2.5K) [39], HKU-IS (3K) [40]...
Fig. 5: Qualitative comparisons with several of the most representative SOTA models. a: original video frame; b: human well-annotated VSOD ground truth; c: VSOD results of the proposed model; d-n: results respectively obtained via MGA [1], AGS [20], SSAV [2], CPD [25], PoolNet [26], EGNet [27], PDBM [12], MBNM [28], FGRN [21], DLVS [18] and STBP [29].

and MSRA10K (4K) [41]. Then, we fine-tune our model using 7.5K video data, including the widely used DAVIS-TR (2K) [30] and the recently proposed DAVSOD (5.5K) [2]. Notice that our training did not include any eye-fixation data provided in the DAVSOD set.

D. Stagewise Training Details

Our network training uses the stochastic gradient descent (SGD) with a momentum value of 0.9 and a weight decay of 5e-4, and we set the initial learning rate to 5e-3. All images/frames are resized to 256×256, and the batch size is initially assigned to 16. To avoid overfitting, we use a random horizontal flip to augment the image training set. In addition, we resample the video data to have different frame rates using intervals of {0,1,2,3,4,5,6} to augment the video training set. We first use the entire training set (all 17.5K data, including both images & videos) to pretrain the spatial branch, in which all temporal units are removed from the decoder. This training stage takes 33,000 epochs. Next, we train the whole spatiotemporal model (all temporal units being included) using the above training set (including both images & videos). Since the spatiotemporal network takes 3 frames as its input each time, each static image is copied three times to meet the input size requirement. This training stage takes 8,500 epochs. Specifically, in this stage, we decrease the batch size from 16 to 4 because the spatiotemporal training requires more GPU memory.

E. Quantitative & Qualitative Comparisons with Other SOTA Models

We compared the proposed model with 23 SOTA models, of which 20 models are VSOD models. These models include: MGA [1], AGS [20], SSAV [2], PDBM [12], MBNM [28], FGRN [21], DLVS [18], SCNN [42], SCOM [43], SFLR [17], SGSP [4], STBP [29], MSTM [44], GFVM [3], SAGM [15], MB+M [45], RWRV [46], SPVM [14], TIMP [47] and SIVM [48].

We also compared our model with the 3 most recent ISOD models: CPD [25], PoolNet [26] and EGNet [27].

1) Quantitative Comparisons: We employed three widely used metrics (F-max, S-measure and MAE) for these comparisons, and Table I shows the results. As can be seen, our model is typically in the top three for all tested datasets, achieving the best results on the SegTrack-V2 and VOS-T datasets.

It may be possible for our model to achieve more competitive results, e.g., we may achieve the best performances on the FBMS and DAVSOD datasets if we include the FBMS-T training set and the eye-fixation data provided in DAVSOD in our training set, as in MGA [1] and SSAV [2]. However, our key focus is to design a general VSOD model with extremely fast speed (the highest FPS); hence, it may not be very necessary to pursue the top performance on all tested datasets.

We also provided a brief qualitative comparison in Fig. 5, showing the advantage of our method in handling cluttered
### TABLE I: Quantitative comparisons between our model and other SOTA VSOD models using the F-max, S-measure, and MAE metrics. The top three results are respectively highlighted in red, green, and blue. “—” indicates that the model was trained on this set; “*” indicates that the model is a deep-learning-based one; and “***” indicates that the results are obtained from a deep-learning-based ISOD model.

| Dataset                | DA VIS-T [30] | FBMS-T [32] | VOS-T [33] | DAVSO-T [2] |
|------------------------|---------------|-------------|------------|-------------|
| MGA [1]†               | 0.892         | 0.924       | 0.952      | 0.899       |
| AGS [2]†               | 0.873         | 0.935       | 0.970      | 0.934       |
| SAAV [2]†              | 0.861         | 0.936       | 0.960      | 0.940       |
| CPD [25]**              | 0.778         | 0.859       | 0.831      | 0.816       |
| PoolNet [26]**          | 0.827         | 0.860       | 0.877      | 0.816       |
| EGNet [27]**            | 0.767         | 0.828       | 0.838      | 0.840       |
| TBBM [23]               | 0.790         | 0.838       | 0.834      | 0.820       |
| MBNM [25]**             | 0.819         | 0.874       | 0.856      | 0.831       |
| FGRN [21]†              | 0.783         | 0.838       | 0.834      | 0.824       |
| DLVS [18]†              | 0.714         | 0.783       | 0.847      | 0.762       |
| SCON [43]†              | 0.783         | 0.832       | 0.764      | 0.702       |
| SFLR [17]               | 0.727         | 0.790       | 0.779      | 0.690       |
| SGSP [4]                | 0.655         | 0.692       | 0.673      | 0.630       |
| STBP [29]               | 0.544         | 0.677       | 0.640      | 0.622       |
| MSTM [34]               | 0.429         | 0.583       | 0.526      | 0.574       |
| GFVM [3]                | 0.569         | 0.687       | 0.599      | 0.595       |
| SAGM [15]               | 0.515         | 0.676       | 0.634      | 0.688       |
| MBM [45]                | 0.497         | 0.597       | 0.554      | 0.692       |
| RWKV [46]               | 0.345         | 0.556       | 0.438      | 0.440       |
| SPVM [14]               | 0.390         | 0.592       | 0.618      | 0.700       |
| TIMP [47]               | 0.448         | 0.593       | 0.573      | 0.479       |
| STM [34]                | 0.450         | 0.557       | 0.581      | 0.522       |

### TABLE III: Experiment setting to perform fair comparison between our model and SSAV19.

| Data | SAAV19 [1] | OUR | Videos | DA VISOD(5.5K)+DAVIS(2K)=7.5K | DAVSO(5.5K) + DAVIS(2K)=7.5K |
|------|------------|-----|--------|--------------------------------|-----------------------------|
| Images | DUTORMON(3K) | DUTORMON(3K)+HKU-IS(4.5K)=9.5K | Fixation | - | - |
| Total | Videos(13K)+Images(5K)=18K | Videod(7.5K)+Images(9.5K)=17K |

### TABLE IV: Qualitative comparisons between our method and the SSAV using the experiment setting detailed in Table III.

| Data | SAAV19 [1] | OUR | Videos | DAVSO(5.5K)+DAVIS(2K)=7.5K | DAVSO(5.5K)+DVBMs(0.5K)=2.5K | DAVSO(5.5K)+DVBMs(0.5K)=2.5K |
|------|------------|-----|--------|--------------------------------|-----------------------------|-----------------------------|
| Images | DUTORMON(3K) | DUTORMON(3K)+HKU-IS(4.5K)=9.5K | Fixation | - | - |
| Total | Videos(13K)+Images(5K)=18K | Videod(7.5K)+Images(9.5K)=17K |

### TABLE II: Training set details of the current SOTA VSOD models: DA VIS [30], SegTrack-V2 [31], FBMS [32], DAVSO [2], MSRA10K [41], DUTORMON [39], PASCAL-S [49], HKU-IS [50], and DUTS [51]. “—” indicates that this dataset was not adopted.
TABLE V: Comparison of size, backbone, runtime, weight, toolbox and platform between our method and SSA V19 using the experiment setting detailed in Table III.

| Methods       | Size | Backbone | FPS | Weight(M) | Toolbox | Platform     |
|---------------|------|----------|-----|-----------|---------|--------------|
| SSA V19 [2]   | 473  | ResNet50 | 20  | 236.2     | Caffe   | GTXTITANX    |
| OUR           | 473  | ResNet50 | 70  | 124       | Pytorch | GTX1080Ti    |

increases to 70, which outperforms the SSA V19 significantly. Specifically, because a relatively large resolution could only cost additional GPU memory without increasing the parameter size (the convolutional operations are parallel performed by GPU threads), our higher FPS rate is mainly derived from our network implementation rather than the lower resolution.

2) Discussions: In practice, the human-eye-fixation data are extremely important for guiding the shifting process between different salient objects. Since the AGS [20] trained its model using a massive amount of eye-fixation data in the DAVSOD set, it achieved a superior performance to that of the DAVSOD-T set.

On the other hand, the MGA [1] also achieved the best performance on the FBMS-T set because it trained its model using the FBMS training set, which shares a similar distribution and semantic categories with the FBMS-T set. The MGA method also adopts the powerful FlowNet2.0 [52]—an off-the-shelf tool trained on massive additional video data—to compute its optical flow data, thus enabling the MGA to perform well on the DA VIS set.

Our model achieved the best performance on the SegTrack-V2 and VOS-T datasets, which are respectively dominated by both temporal movements and spatial appearances, thus making these two sets very challenging. Despite the challenge, our model still performed well on these sets, showing that the spatiotemporal interaction is really effective.

3) Efficiency Comparisons: We report the runtime comparisons and the net size comparisons in Table VI and Table VII, in which our model is evaluated on a machine with a GTX1080Ti GPU. As shown in Table VI, even being tested on an outdated GPU, our model achieved the highest FPS compared with all other models. Notice that several of these models were evaluated on more powerful GPUs, of which the corresponding FPS results were directly gathered from the original papers. Table VII demonstrates the network parameter size comparisons between our model and several of the most representative SOTA ones, in which our method is designed with the lightest network architecture. The main reasons for our model being capable of achieving the highest FPS are as follows: 1) its lightweight network design with the fewest learnable hidden parameters and 2) the redesign of the UNet encoder as a temporal branch without requiring additional computing resources.

F. Component Evaluations

In Table VIII, we verify the effectiveness of each component used in the proposed model, in which the baseline represents the original UNet using the 2D convolution only. Qualitative comparisons between different components can also be found in Fig. 6.
However, we also noticed a slight performance degeneration when the number was larger than 4 (i.e., 3D_R5), which may be a result of the fact that the spatial misalignments become extremely worse after multiple 3D convolutions are carried out; thus, this side effect may outweigh the benefit obtained from the enhancement towards the temporal sensing ability. Therefore, according to the experimental results, we decided to use 3 sequential 3D convolutions each time. Notice that this ablation study also verified the advantage of the proposed sequential 3D convolutions against the plain 3D convolution.

We also conducted an additional experiment to verify the effectiveness of the proposed fast cyclic padding scheme, i.e., we compared the proposed fast cyclic padding scheme with the conventional zero padding scheme, the quantitative results of which can be seen in Table X. The new padding scheme can effectively boost the performance mainly on the SegTrack-V2 set, while the improvement becomes really marginal on the other sets. The main reason is that for all the sets tested, SegTrack-V2 is the only one with all its salient objects dominated by temporal information (e.g., the birdfall sequence), while the proposed cyclic padding aims for only one thing—to make the sequential 3D convolutions strong in sensing temporal information. Thus, the results reported in Table X are reasonable, verifying the effectiveness of the proposed padding scheme.

### 2) Effectiveness of the Proposed Fast Temporal Shuffle Scheme (Sec. IV-B):

As shown in Table VIII, the overall performance after integrating the fast temporal shuffle scheme (denoted by “+S”) into the temporal unit (i.e., the proposed sequential 3D convolutions with the fast padding scheme) can be further improved; see “+3D” vs. “+3D+S”.

This performance gain may be explained by taking into account the major highlight of the proposed fast temporal shuffle scheme. That is, the shuffle scheme is capable of making the current spatial deep features temporal-aware by integrating the temporally neighbored spatial deep features implicitly; thus, the overall performance can be boosted over those datasets dominated by temporal information, e.g., DAVIS-T and SegTrack-V2.

### TABLE VIII: Ablation study of our method on six datasets. “+S” represents the fast temporal shuffle module, “+3D” represents the fast 3D convolution module, “+MA” represents the multi-scale attention module, and “+SN” denotes the plain shuffle module adopted in the ShuffleNet.

| DataSets | DAVIS-T [30] | SegTrack-V2 [31] | Visal [3] |
|----------|--------------|------------------|-----------|
| Metric   | F-max S-measure MAE | F-max S-measure MAE | F-max S-measure MAE |
| +3D+S+MA | 0.865 0.892 0.025 | 0.890 0.891 0.017 | 0.952 0.952 0.013 |
| +3D+MA   | 0.861 0.890 0.026 | 0.851 0.883 0.018 | 0.944 0.950 0.016 |
| +3D+S    | 0.858 0.890 0.024 | 0.855 0.891 0.017 | 0.951 0.949 0.017 |
| +3D+SN   | 0.853 0.894 0.027 | 0.843 0.886 0.018 | 0.943 0.941 0.016 |
| +3D     | 0.855 0.895 0.027 | 0.841 0.884 0.017 | 0.942 0.943 0.016 |
| Baseline | 0.837 0.878 0.033 | 0.822 0.876 0.022 | 0.912 0.924 0.025 |

### TABLE IX: The ablation study on different numbers of the proposed sequential 3D convolutions (i.e., 3D_Ri, in which i ∈ {1, 3, 5} denotes different number of 3D convolutions). The best scores are labeled in red.

| Dataset | DAVIS-T [30] | SegTrack-V2 [31] |
|---------|--------------|------------------|
| Metric  | F-max S-measure MAE | F-max S-measure MAE |
| +3D     | 0.841 0.882 0.025 | 0.832 0.890 0.015 |
| Baseline | 0.937 0.941 0.016 | 0.943 0.941 0.016 |

### TABLE X: The padding scheme comparisons. “CyclicPadding” represents the proposed fast cyclic padding scheme, and “ZeroPadding” represents the conventional zero-padding scheme.

| Dataset | DAVIS-T [30] | SegTrack-V2 [31] |
|---------|--------------|------------------|
| Metric  | F-max S-measure MAE | F-max S-measure MAE |
| CyclicPadding | 0.865 0.892 0.023 | 0.860 0.891 0.017 |
| ZeroPadding | 0.857 0.889 0.024 | 0.848 0.890 0.018 |

### TABLE XI: The ablation study on different numbers of the proposed sequential 3D convolutions (i.e., 3D_Ri, in which i ∈ {1, 3, 5} denotes different number of 3D convolutions). The best scores are labeled in red.

| Dataset | DAVIS-T [30] | SegTrack-V2 [31] |
|---------|--------------|------------------|
| Metric  | F-max S-measure MAE | F-max S-measure MAE |
| +3D+MA | 0.855 0.895 0.027 | 0.841 0.884 0.016 |
| +3D+S | 0.851 0.851 0.060 | 0.650 0.745 0.086 |
| Baseline | 0.771 0.839 0.062 | 0.629 0.725 0.099 |

In addition, we have provided two quantitative evaluations regarding the proposed temporal shuffle, i.e., “3D+MA” and “3D+SN+MA”, where ‘SN’ denotes the plain shuffle scheme adopted in the ShuffleNet. As can be seen in Table VII, the performance of ‘3D+MA’ is worse than that of the ‘3D+SN+MA’. Meanwhile, we can notice that ‘3D+SN’ is clearly worse than ‘3D+S’, showing the effectiveness of the proposed temporal shuffle scheme.

### 3) Effectiveness of Using Multiscale Attention (Eq. 2):

As shown in Table VIII, the overall performance can be further improved by integrating the multiscale attention into each decoder layer.

In fact, the attention mechanism was widely adopted by the most recent works (e.g., [2]); however, these works cannot make full use of this spatial attention in a multiscale manner because these models revised their decoders into some single-in-single-out modules (e.g., convLSTM). In sharp contrast, the
TABLE XI: Performance comparisons between different decoder layers. $DeCoder_i, i \in \{5, 4, 3, 2, 1\}$ represents the output of the $i$-th decoder layer; see more details in Fig. 2.

| Dataset | F-max | S-measure | MAE | F-max | S-measure | MAE |
|-----------------------|--------|------------|-----|--------|------------|-----|
| $DeCoder_5$            | 0.865  | 0.892      | 0.023 | 0.860  | 0.891      | 0.017 |
| $DeCoder_4$            | 0.863  | 0.891      | 0.022 | 0.857  | 0.891      | 0.016 |
| $DeCoder_3$            | 0.861  | 0.890      | 0.024 | 0.853  | 0.885      | 0.017 |
| $DeCoder_2$            | 0.839  | 0.874      | 0.026 | 0.827  | 0.863      | 0.020 |
| $DeCoder_1$            | 0.806  | 0.855      | 0.032 | 0.792  | 0.839      | 0.024 |

| Dataset | Metric | Dataset | Metric |
|-----------------------|--------|-----------------------|--------|
| F-max | S-measure | MAE | F-max | S-measure | MAE |
| $DeCoder_5$ | 0.952 | 0.952 | 0.013 | 0.856 | 0.872 | 0.038 |
| $DeCoder_4$ | 0.948 | 0.949 | 0.013 | 0.853 | 0.782 | 0.038 |
| $DeCoder_3$ | 0.942 | 0.945 | 0.015 | 0.848 | 0.868 | 0.040 |
| $DeCoder_2$ | 0.923 | 0.932 | 0.018 | 0.832 | 0.854 | 0.044 |
| $DeCoder_1$ | 0.899 | 0.915 | 0.024 | 0.807 | 0.833 | 0.050 |

| Dataset | Metric | Dataset | Metric |
|-----------------------|--------|-----------------------|--------|
| F-max | S-measure | MAE | F-max | S-measure | MAE |
| $DeCoder_5$ | 0.791 | 0.850 | 0.058 | 0.651 | 0.746 | 0.086 |
| $DeCoder_4$ | 0.790 | 0.849 | 0.058 | 0.651 | 0.746 | 0.086 |
| $DeCoder_3$ | 0.787 | 0.847 | 0.058 | 0.648 | 0.744 | 0.087 |
| $DeCoder_2$ | 0.769 | 0.836 | 0.059 | 0.639 | 0.736 | 0.087 |
| $DeCoder_1$ | 0.745 | 0.816 | 0.064 | 0.619 | 0.725 | 0.090 |

G. Why do we choose a small feature size for the spatial branch?

In SSAV [2] and PDBM [12], the interactions between their spatial and temporal branches are quite limited, which solely feed the last output of the spatial branch into the temporal branch. As a result, this “single-scale and one-way interaction” has one critical weakness: the final VSOD results are frequently associated with blurry object boundaries. Therefore, these two models need to use additional spatial attention to compensate for the loss in spatial details; thus, their dilated convolutions must use a relatively large size (60*60) to ensure the effectiveness of their attention models.

In sharp contrast, our model introduces “multiscale” spatial information into the temporal branch by using side-outputs from different spatial layers (Fig. 2), which ensures that the detection results are conserved with sharp object boundaries. In addition, the attention module adopted by our method aims to provide the “location information” for the temporal branch; thus, it is totally acceptable to design our spatial branch with a small feature size (8*8).

H. Limitations

Because our method takes only 3 consecutive video frames as its input each time, its sensing scope over the temporal scale is quite limited, leaving those regions undergoing a long static period undetected; see pictorial demonstrations in Fig. 7. In fact, this drawback is quite common in the VSOD community, and we believe it can be alleviated by introducing the long-term spatiotemporal information, which deserves some future investigations.

VI. CONCLUSIONS

In this paper, we proposed an extremely fast end-to-end VSOD model. The major highlights of this model can be summarized as follows.

First, we devised a lightweight temporal unit, which can be treated as a plug-in to be inserted into each decoder layer, enabling the original spatial decoder to sense temporal information.

Second, we provided a feasible way to enhance the temporal sensing ability of the proposed temporal unit, in which the key innovations include sequential 3D convolutions and a temporal shuffle scheme. In addition, we used the fast padding scheme to avoid the biasing limitation induced by the usage of sequential 3D convolutions; this padding scheme also further improves the temporal sensing ability.

Last, we conducted extensive quantitative comparisons and evaluations to show the advantages of the proposed model and verified the effectiveness of each of its main components. The quantitative comparisons indicated that the proposed spatiotemporal model outperforms all other SOTA models in both detection performance and speed.

Acknowledgments. This research was supported in part by the National Natural Science Foundation of China (No. 61802215 and No. 61806106), Natural Science Foundation of Shandong Province (No. ZR2019BF011 and ZR2019QF009) and National Science Foundation of the USA (No. IIS-1715985 and IIS-1812606).

REFERENCES

[1] H. Li, G. Chen, G. Li, and Y. Yu, “Motion guided attention for video salient object detection,” ICCV, 2019.

[2] D. Fan, W. Wang, M. Cheng, and J. Shen, “Shifting more attention to video salient object detection,” in CVPR, 2019, pp. 8554–8564.

[3] W. Wang, J. Shen, and L. Shao, “Consistent video saliency using local gradient flow optimization and global refinement,” IEEE TIP, vol. 24, no. 11, pp. 4185–4196, 2015.

[4] Z. Liu, J. Li, L. Ye, G. Sun, and L. Shen, “Saliency detection for unconstrained videos using superpixel-level graph and spatiotemporal propagation,” IEEE TCSVT, vol. 27, no. 12, pp. 2527–2542, 2016.
