Bootstrapping Objectness from Videos by Relaxed Common Fate and Visual Grouping Supplementary Materials

Long Lian[1]
[1]UC Berkeley
longlian@berkeley.edu

Zhirong Wu[2]
[2]Microsoft Research Asia
wuzhiron@microsoft.com

Stella X. Yu[1,3]
[3]University of Michigan
stellayu@umich.edu

1. Additional Visualizations and Discussions

We present additional visualizations on the three main datasets that we benchmark our method on [1,7,10,11]. We demonstrate high-quality segmentation in several challenging cases and discuss some limitations of our method with examples.

1.1. Visualizations of the Residual Pathway

As shown in Fig. 1, the introduction of the residual pathway allows our segmentation prediction to better fit the flow of deformable and articulated objects. In addition, it also relieves our segmentation module from strictly fitting the flow from 3D rotation and changing depth in a piecewise constant manner. By modeling relative motion in 2D flow, the residual pathway makes our method flexible and robust to objects with complex motion.

1.2. DAVIS2016, SegTrackv2, and FBMS59

We visualize our methods on DAVIS2016, SegTrackv2, and FBMS59 in Fig. 2, Fig. 3, and Fig. 4, respectively. Our method shows great robustness in challenging scenes where there is insufficient motion information, due to its ability to leverage both motion and appearance.

2. Additional Experiments

Unless otherwise stated, all the ablation experiments in this section include only stage 1, as the ablations in this section are not relevant to the appearance supervision. Results are without post-processing.

2.1. Ablation on Different Optical Flow Estimation Methods

As listed in Tab. 1, almost all recent UVOS works rely on a separate optical flow model pretrained on synthetic data. We use RAFT [15] flow by default, following previous works in UVOS. AMD trains [14] from scratch but achieves much lower mIoU.

Method	CIS	MG	EM	SIMO	Tok.Cut	GWM	OCLR	**RCF**
Flow Model	PWCNet	RAFT	RAFT	RAFT	RAFT	RAFT	RAFT	**RAFT**

Table 1. **Optical flow methods that each UVOS approach employs by default.** All methods in the table use pretrained weights for flow estimation. We utilize RAFT flow with pretrained weights from synthetic data, which is the common flow method among all the UVOS methods. Other than the methods listed in the table, AMD trains PWCNet [14] architecture from scratch but achieves much lower performance compared to RCF.

Method	ARFlow [8]	PWCNet [14]	GMFlow [18]	**RAFT** [15]
DAVIS16 $\mathcal{J}(\uparrow)$	70.3	74.8	76.6	**78.9**

Table 2. **Our method with different optical flow estimation methods.** We use pretrained optical flow on synthetic data for supervised optical flow methods.

To evaluate our method's robustness to optical flow estimation methods, we evaluate our method on PWCNet [14], GMFlow [18], and self-supervised ARFlow [8], in addition to RAFT [15].

As shown in Tab. 2, our method suffers from a mild drop with noisier optical flow. However, our performance is largely retained without tuning the hyperparameters when employing other optical flow methods. We believe the performance gap between different optical flow estimation methods will be reduced further with additional hyperparameter tuning on each flow estimation method.

2.2. Preventing Trivial Solutions for Residual Flow Prediction

There are two factors that prevent trivial solutions: **1)** Regularization with upper bound λ limits the residual prediction to only capturing small relative motion (10 pixels by default). **2)** The residual flow branch is initialized to be small, which favors the solution to be simple motion patterns.

As shown in Tab. 3, the results (mIoU on DAVIS16)

Upper bound λ	1	5	**10**	20	50	100	200	400
Ours Init	72.7	76.5	**78.9**	78.3	78.3	77.4	72.8	78.3
Default Init	72.7	76.0	78.1	78.5	73.5	73.4	73.3	1.0

Table 3. **Using a smaller initialization and upper bound is important for the residual flow pathway in our method.** Ours Init refers to an initialization scheme which is 10x smaller than PyTorch default init. Red color indicates collapses.

Weight Decay	10^{-6}	10^{-4}	10^{-2}
Motion-app. Alignment	-0.672	**-0.670**	-0.768
Subset 1 mIoU	77.2	**77.6**	75.7
Subset 2 mIoU	77.0	**80.5**	72.0
Subset 3 mIoU	**77.3**	76.8	76.2
Full val mIoU	77.2	**78.9**	74.8

Table 4. **Applying motion-appearance alignment provides the optimal weight decay without using labels.** In contrast, using subset mIoU misses the optimal value in one of the three runs. Higher metric values indicate higher segmentation quality for all metrics.

show that small residual initialization allows RCF to be insensitive to a large range of λ against performance degradation or collapses, even though setting λ too large will still cause instability in the form of large mIoU fluctuations. With small residual initialization, λ is relatively stable to tune.

2.3. Applying Motion-appearance Alignment to Non-method Specific Hyperparameters

To explore the possibility of using our proposed label-free hyperparameter tuning method to tune hyperparameters that are non-method specific, we evaluate our metric on runs with three different weight decay values: 10^{-6} and 10^{-2} in addition to our default value of 10^{-4}. We choose this range of hyperparameter values since we observed that varying the weight decay by smaller amounts had a negligible impact on the final mIoU. As in other hyperparameter tuning experiments, we randomly sample 25% of the sequences from the validation set three times and evaluate the effect of using a smaller labeled validation subset for comparison. Shown in Tab. 4, while the mIoU values from the labeled validation subsets vary significantly between samplings, with one of the three runs missing the optimal value, our metric follows the full validation mIoU trend and selects the best hyperparameter values among the three.

3. Pseudo-code for Hyperparameter Tuning With Motion-appearance Alignment

We present the pseudo-code for hyperparameter tuning with motion-appearance alignment in Algorithm 1.

Algorithm 1 Pseudo-code for using motion-appearance alignment for hyperparameter tuning

Input: A set of frames $\{I\}$ with N frames
Input: A set of settings with different hyperparameters S
Output: A chosen optimal setting S^* according to motion-appearance-alignment
  for each setting S in $\{S\}$ **do**
    Train a model with setting S
    Obtain prediction masks $\{M\}$ with trained model
    for each frame-mask pair (I_i, M_i) in $\{I\}, \{M\}$ **do**
      Calculate affinity A from frozen ViT features:
      $A_{ij} = \mathbb{1}(\text{sim}(f_{\text{aux}}(I_t)_i, f_{\text{aux}}(I_t)_j) \geq 0.2)$
      Calculate cut between the predicted foreground and background $\text{Cut}(A, \mathbf{x})$:
      $\mathbf{x} \leftarrow \text{Flatten}(M_i)$
      $\text{Cut}(A, \mathbf{x}) = (1 - \mathbf{x})A\mathbf{x}$
      Calculate normalized cut between the predicted foreground and background $\text{NCut}(A, \mathbf{x})$:
      $\text{NCut}(A, \mathbf{x}) = \frac{\text{Cut}(A,\mathbf{x})}{\sum_{i=1}^{HW}(A\mathbf{x})_i} + \frac{\text{Cut}(A,\mathbf{x})}{\sum_{i=1}^{HW}(A(1-\mathbf{x}))_i}$
      Calculate the motion-appearance alignment for the current frame:
      $L_i \leftarrow -\text{NCut}(A, \mathbf{x})$
    end for
    $L_S \leftarrow \frac{1}{N}\sum_{i=1}^{N} L_i$
  end for
  $S^* = \arg\max_S L_S$

4. Additional Implementation Details

Our setting mostly follows previous works [3, 9]. Following the official implementation in [9], we treat the video frame pair $\{t, t+1\}$ as both a forward action from time t to time $t+1$ and a backward action from time $t+1$ and t, since they follow similar rules for visual grouping. Therefore, we use this to implement a symmetric loss that applies the loss function on both forward and backward. We then sum the forward loss and backward loss up to obtain the final loss. Note that this could be understood as a data augmentation technique that always supplies a pair in forward and backward to the training batch. However, since our ResNet shares weights for each image input, the feature for each input is reused by the forward and backward action. Furthermore, we use twice the number of output channels on the segmentation head than needed for single direction flow prediction to predict forward and backward flow in one forward run, due to better performance. Thus, the symmetric loss only adds marginal computation and is included in our implementation as well.

Furthermore, following [9], for DAVIS16, we use random crop augmentation during training to crop a square image from the original image. At test time, we directly input the original image, which is non-square. It is worth not-

ing that the augmentation makes the image size different for training and testing, but as ResNet [5] takes images of different sizes, this does not pose a problem empirically. In STv2 and FBMS59, the images have very different aspect ratios (some having a height lower than the width), and thus we resize the images to 480p as a preprocessing before the standard pipeline. We additionally use pixel-wise photometric transformation [4] for augmentation with the default hyperparameters for this augmentation.

As for the architecture, we found that simply taking the feature from the last ResNet stage provides insufficient detailed information for high-quality output. Instead of incorporating a more complicated segmentation head (*e.g.*, [2] in [3]), we chose to keep our architecture easy to implement by only changing the head in a simple fashion. Following the standard approach of multi-scale feature fusion, we resized and concatenated the feature from the first residual block and the last residual block in ResNet, which allows the feature to jointly capture high-level information and low-level details. Note that such fusion is only applied to the segmentation head, and residual prediction is simply bilinearly upsampled. Due to lower image resolution, no feature merging is performed for STv2 in stage 1. Following [3], we load self-supervised ImageNet pretrained weights learned without annotation, since the training video datasets are too small for learning generalizable feature (*e.g.*, DAVIS16/STv2/FBMS59 has only 3,455/976/13,860 frames), with DenseCL weights [12, 17] on ResNet50 for our method. This can be replaced by training on uncurated Youtube-VOS [19] with our training process, as in [9], so that one implementation can be used throughout training for simplicity in real-world applications.

In our training, we follow [9] and use a batch size of 16 (with two images in a pair, and thus 32 images processed in each forward pass). Stage 1 and stage 2 take around 200 and 40 epochs, respectively, for DAVIS16. We use a learning rate of 1×10^{-4} with Adam optimizer [6] and polynomial decay (factor of 0.9, min learning rate of 1×10^{-6}). We set weight decay to 1×10^{-4} for DAVIS and 1×10^{-6} for STv2 and FBMS59. Due to the fact that normalized cuts is slow to optimize, we split stage 2 into two sub-stages: one with the CRF followed by one with normalized cuts optimization, each of the stage has the same number of training steps. In the CRF substage in stage 2, we set $w_{\text{motion}} = 1$ and $w_{\text{app}} = 10$ to balance the two losses. However, we observe training instability if we supervise the network directly by its output refined by the CRF. Therefore, we apply exponential moving averaging (EMA) to the model weights and supervise the network by the output from the EMA model, with momentum $m = 0.999$. In the normalized cuts substage, we pre-generate the network's outputs and use the refinement as described in the methods section, which involves running CRF before and after normalized cuts refinement and multiplying the refined masks from the two CRF runs. This is equivalent to applying such refinement with EMA with $m = 1.0$. In this substage, we set $w_{\text{motion}} = 0.1$ and $w_{\text{app}} = 2.0$.

5. Per-sequence Results

We list our per-sequence results on DAVIS16 [11], STv2 [7], FBMS59 [1, 10] in Tab. 5, Tab. 6, and Tab. 7, respectively. The results are with post-processing.

6. Future Directions

As our method does not impose temporal consistency, it does not effectively leverage information redundancy from neighboring frames. Using such information could make our method more robust in dealing with frames that provide insufficient motion and appearance information. Temporal consistency measures, such as matching warped predictions, could be incorporated as an additional loss term or as post-processing, as in [20].

Furthermore, our method currently does not support segmenting multiple parts of the foreground or identifying each object instance. To address this, methods such as normalized cuts [13] could be used to split the foreground into several objects with motion and appearance input to provide signals to train the model. Another potential approach is to over-split the scene with many object channels and use other unsupervised methods such as FreeSOLO [16, 17] to obtain coarse segmentation proposals to merge the channels to form object instance segmentation.

Sequence	$\mathcal{J}$
blackswan	76.2
bmx-trees	78.3
breakdance	86.1
camel	92.7
car-roundabout	80.7
car-shadow	80.4
cows	88.0
dance-twirl	90.4
dog	91.7
drift-chicane	94.1
drift-straight	65.6
goat	81.6
horsejump-high	93.4
kite-surf	53.1
libby	96.6
motocross-jump	57.0
paragliding-launch	26.0
parkour	95.8
scooter-black	72.4
soapbox	86.1
Frame Avg	83.0

Table 5. **Per sequence Jaccard index $\mathcal{J}$ on DAVIS16 [11].**

Sequence	$\mathcal{J}$
bird of paradise	91.7
birdfall	60.4
bmx	76.6
cheetah	52.4
drift	86.3
frog	82.2
girl	80.6
hummingbird	67.6
monkey	82.5
monkeydog	55.5
parachute	93.2
penguin	66.2
soldier	79.8
worm	85.6
Frame Avg	79.6

Table 6. **Per sequence Jaccard index $\mathcal{J}$ on STv2 [7].**

Sequence	$\mathcal{J}$
camel01	88.3
cars1	86.4
cars10	38.2
cars4	70.3
cars5	79.3
cats01	88.2
cats03	82.0
cats06	59.7
dogs01	74.4
dogs02	91.6
farm01	82.6
giraffes01	65.9
goats01	89.8
horses02	86.2
horses04	88.6
horses05	71.6
lion01	84.9
marple12	79.3
marple2	73.7
marple4	87.8
marple6	50.8
marple7	32.1
marple9	38.4
people03	42.9
people1	86.1
people2	88.0
rabbits02	93.8
rabbits03	85.9
rabbits04	20.2
tennis	78.6
Frame Avg	72.4

Table 7. **Per sequence Jaccard index $\mathcal{J}$ on FBMS59 [1, 10].**

Figure 1. **Visualizations for both the piecewise constant and the residual pathways** show that the introduction of the residual pathway allows our segmentation prediction to better fit the flow of deformable and articulated objects. In addition, it also relieves our segmentation module from strictly fitting the flow from 3D rotation and changing depth in a piecewise constant manner. By modeling relative motion in 2D flow, the residual pathway makes our method flexible and robust to objects with complex motion.

Figure 2. **Additional visualizations on DAVIS16 [11].** Our method remains robust in scenes where there is insufficient motion information, in which cases our method leverages appearance cues to learn high-quality segmentation in (a) to (e). Our method accurately segments multiple foreground objects as foreground when they move together, which is consistent with human perception in (b). However, our method may exclude a portion of an object in (f), since the motion misses part of the front wheel of the bicycle and the structure is too small for appearance to capture.

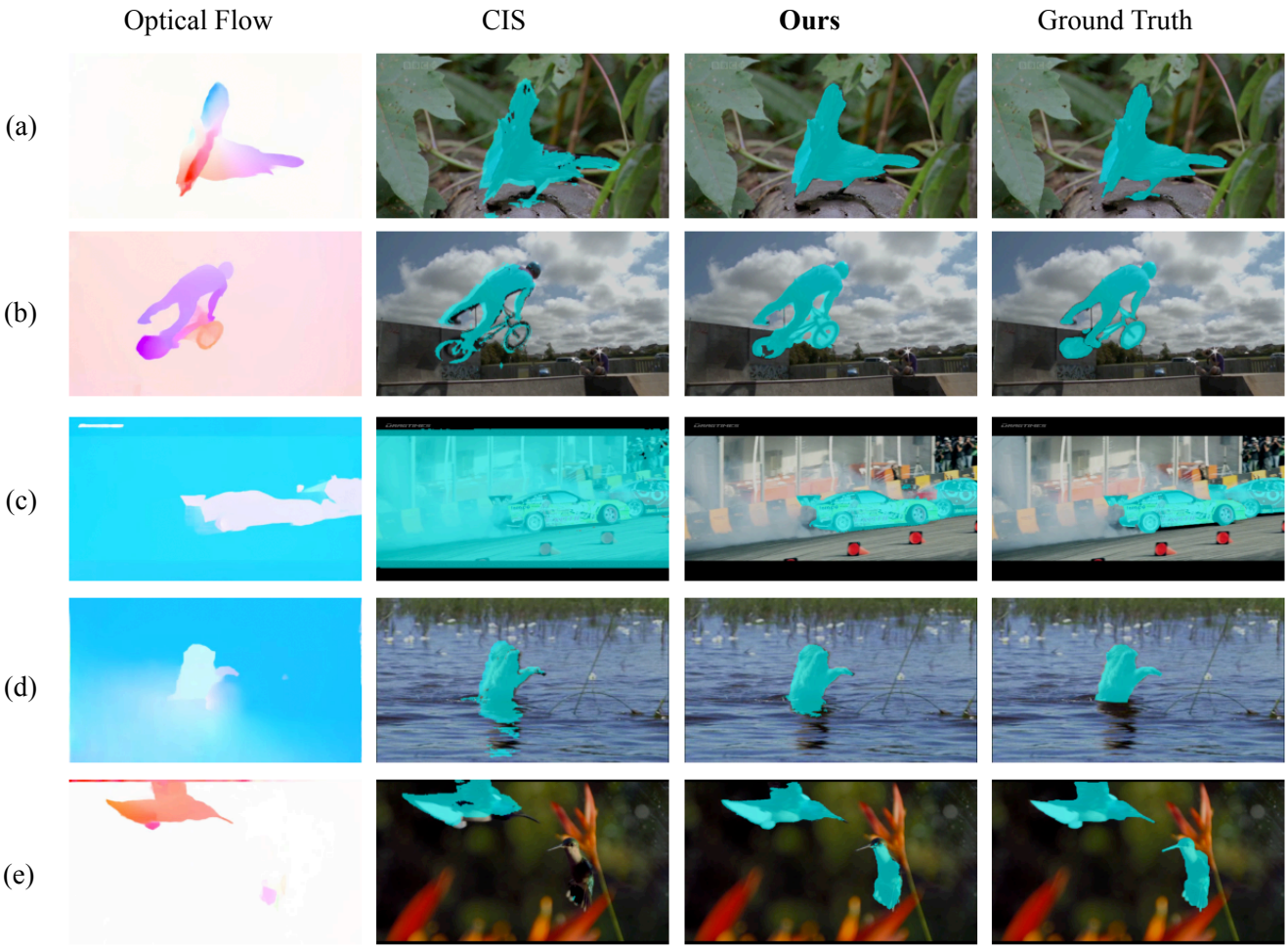

Figure 3. **Additional visualizations on STv2 [7].** Our method, with the residual flow, could model non-uniform 2D flow resulting from object rotation in 3D in (a), as long as the rotation flow falls within our upper bound constraint for the residual flow. Our method also captures multiple objects in a foreground group in (b), (c), and (e). Our method is robust to camera motion that leads to non-uniform background flow in (c) and misleading common motion (reflections) in (d). However, due to the relatively low image resolution, our method may miss some details of the object. For example, the legs of both animals in (f) and the wings of the bird in (g).

Figure 4. **Additional visualizations on FBMS59 [1, 10].** Our method is robust in scenes with complicated and distracting appearances in (a). Our method also works with fine details in (b) and (e). Our method accurately segments multiple foreground objects in (c) and (d). However, when multiple objects or object parts exist in one scene and exhibit different motion patterns, our method may be confused in (f) and (g).

References

[1] T Brox, J Malik, and P Ochs. Freiburg-berkeley motion segmentation dataset (fbms-59). In *European Conference on Computer Vision (ECCV)*, 2010. 1, 3, 4, 8

[2] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. 2021. 3

[3] Subhabrata Choudhury, Laurynas Karazija, Iro Laina, Andrea Vedaldi, and Christian Rupprecht. Guess what moves: Unsupervised video and image segmentation by anticipating motion. *arXiv preprint arXiv:2205.07844v1*, 2022. 2, 3

[4] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020. 3

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[7] Fuxin Li, Taeyoung Kim, Ahmad Humayun, David Tsai, and James M Rehg. Video segmentation by tracking many figure-ground segments. In *Proceedings of the IEEE international conference on computer vision*, pages 2192–2199, 2013. 1, 3, 4, 7

[8] Liang Liu, Jiangning Zhang, Ruifei He, Yong Liu, Yabiao Wang, Ying Tai, Donghao Luo, Chengjie Wang, Jilin Li, and Feiyue Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6489–6498, 2020. 1

[9] Runtao Liu, Zhirong Wu, Stella Yu, and Stephen Lin. The emergence of objectness: Learning zero-shot segmentation from videos. *Advances in Neural Information Processing Systems*, 34:13137–13152, 2021. 2, 3

[10] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *IEEE transactions on pattern analysis and machine intelligence*, 36(6):1187–1200, 2013. 1, 3, 4, 8

[11] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 724–732, 2016. 1, 3, 4, 6

[12] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 3

[13] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. 3

[14] D Sun, X Yang, MY Liu, and J Kautz. Pwc-net: Cnns for optical flow using pyramid. *Warping, and Cost Volume [J]*, 2017. 1

[15] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 1

[16] Xinlong Wang, Zhiding Yu, Shalini De Mello, Jan Kautz, Anima Anandkumar, Chunhua Shen, and Jose M Alvarez. Freesolo: Learning to segment objects without annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14176–14186, 2022. 3

[17] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3024–3033, 2021. 3

[18] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8121–8130, 2022. 1

[19] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 585–601, 2018. 3

[20] Yanchao Yang, Antonio Loquercio, Davide Scaramuzza, and Stefano Soatto. Unsupervised moving object detection via contextual information separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 879–888, 2019. 3