A Semi-Self-Supervised Approach for Dense-Pattern Video Object Segmentation

Supplementary Material

Keyhan Najafian[1], Farhad Maleki[2], Lingling Jin[1], Ian Stavness[1]

[1]Department of Computer Science, University of Saskatchewan, Saskatoon, Saskatchewan, Canada

[2]Department of Computer Science, University of Calgary, Calgary, Alberta, Canada

{keyhan.najafian, lingling.jin, ian.stavness}@usask.ca, farhad.maleki1@ucalgary.ca

S7. Data Synthesis Procedure

We utilized video clips of wheat fields and background fields, which were fields without wheat. An annotated video clip $V_i = \{(x_{i_t}, y_{i_t})\}_{i_t=1}^{T_i}$, is defined as a series of T_i consecutive image frames, where x_{i_t} represents the ${i_t}^{th}$ frame of V_i, and y_{i_t} represents the pixel-level annotation (mask) for x_{i_t}, representing wheat heads in x_{i_t}.

We used three top-view wheat field videos, $\mathbb{V} = \{V_1, V_2, V_3\}$, and 28 background videos, $\mathbb{B} = \{B_i \mid 1 \leq i \leq 28\}$, featuring fields without wheat. All videos were captured with a 12-megapixel handheld camera. The background videos contained $118,259$ frames. We manually annotated seven randomly selected frames from the wheat field videos, $\mathbb{F} = \{F_i \mid 1 \leq i \leq 7\}$, for data synthesis. Using these frames and extracted wheat heads, we applied a cut-and-paste strategy to overlay wheat heads onto background frames, generating computationally annotated video clips, thereby reducing manual annotation effort.

We employed a group of background videos ($\mathbb{B}$), which were fields without wheat crops exhibiting various vegetation types and environmental conditions, used as the backgrounds of synthetic videos. Figure S7 illustrates the background video frames extraction process. This was achieved by randomly choosing video B_i from the background videos $\mathbb{B} = \{B_i \mid 1 \leq i \leq 28\}$. Then, τ consequent frames from B_i were selected to form a set $C_i = \{c_{i_t} \mid 1 \leq t \leq T_i - \tau\}$, where T_i is the B_i's length. Next, a random region of size 1024×1024 was chosen, and the crop defined by this region was applied to all the frames in B_i, resulting in a set of τ consecutive frames of size 1024×1024, denoted as C_i'. This set was subsequently utilized as the background frames for overlaying objects of interest in synthesizing a video clip forming τ frames.

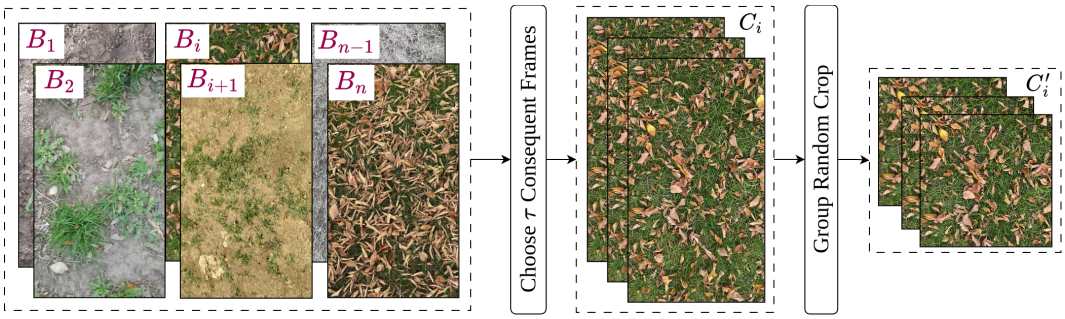

Figure S7. The procedure for extracting video frames from background video $B_i \in \mathbb{B}$.

We also extracted wheat heads from a small number of manually annotated frames depicting both earlier growth stages in green shades and harvestable-ready stages in yellow coloration, resulting in a set of real wheat heads ($\mathbb{H}$) consisting of yellow (mature) and green (mid-season) wheat heads. Note that, for wheat head extraction, we chose the frames from three distinct videos in $\mathbb{V}$ that had no intersection with the $\mathbb{W}$ dataset used in the pseudo-labeling phase of model training. We also used extracted wheat heads in $\mathbb{H}$ as cookie-cutters to extract regions with no wheat heads (fake wheat heads) from the original frames, denoted as $\overline{\mathbb{H}}$.

Figure S8 illustrates the process for synthesizing a manually annotated video clip. For τ consecutive background frames in C_i', a random number of fake and real wheat heads were chosen randomly from $\overline{\mathbb{H}}$ and $\mathbb{H}$, respectively. The fake wheat heads were overlaid on the first frame of C_i', followed by real wheat heads. To simulate the movement and deformation of the crop naturally caused by wind, we applied a sequence of spatial- and pixel-level transformations to each wheat head before

overlying them on consecutive frames in C'_i. These transformations were automatically generated for each wheat head to ensure consistent movement while showing varying degrees of deformations. Specifically, we defined two types of movement for each wheat head object: object-level and frame-level movement. In object-level movement, we updated the position and direction of each object individually. However, at the frame-level motion, all the objects' positions were adjusted according to a predefined motion behavior at the frame level.

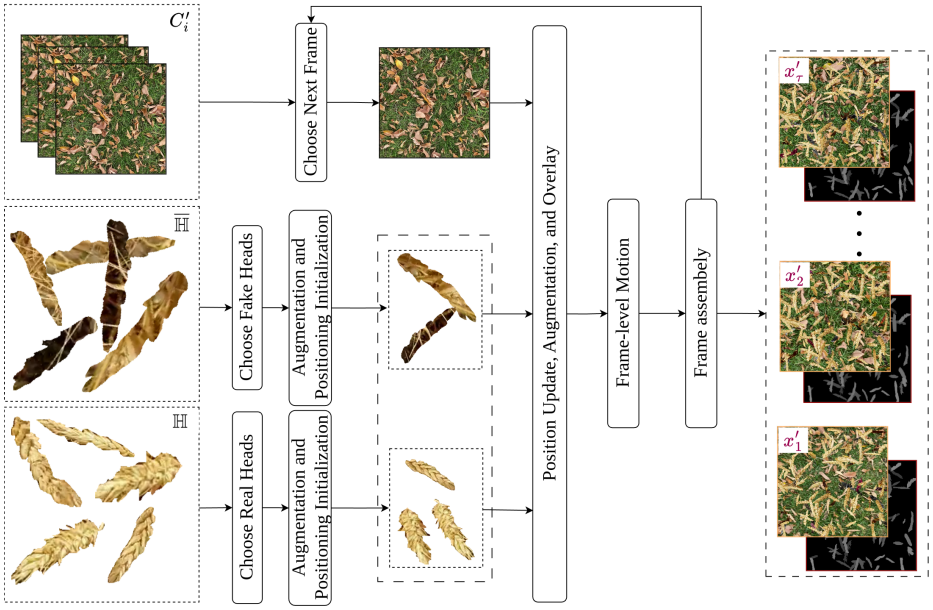

Figure S8. This diagram illustrates the process of synthesizing videos with dimensions of 1024×1024 and a length of τ. We generated a synthesized video V' by initializing a random selection of real and fake heads from $\mathbb{H}$ and $\overline{\mathbb{H}}$, thereby uniquely augmented and positioned by predefined parameters and overlaid on the first frame of τ frames in C'_i. Subsequent frames of V' were simulated based on the positions of wheat heads in the preceding frame, utilizing both object-level and frame-level motions. Objects that were not within the frame anymore (due to the object motions) were subsequently restored by incorporating additional real heads into the chosen object set and overlaid on the current frame before proceeding to the next frame. The frame masks were also generated simultaneously by applying the same object- and frame-level motions on the segmentation counters.

This process synthesized frames and their corresponding masks by applying special transformations and deformations to both the frames and masks, ensuring consistent annotation for the resulting video clip. To create the mask corresponding to each synthesized video frame, a 1024×1024 blank frame was allocated for each of the τ frames in C'_i. We kept track of the position of each real wheat head when overlaid on the background image and its movement and deformation to convert the corresponding region on the mask to 1. Note that the fake wheat heads were ignored. Table S3 provides a summary of the synthesized dataset, including statistical information on the synthesized videos and the raw data used in the synthesis process. This includes the number of background videos and the number of real wheat heads ($\mathbb{H}$).

Dataset	Subset	Background Videos	Heads	Synthesized Videos	Video Clips
$\mathbb{S}_{train}$	Green Shaded	13	101	260	15600
	Yellow Shaded	15	251	600	36000

Table S3. Quantitative summary of the synthetic videos and their frames distributions in the $\mathbb{S}_{train}$ dataset.

S8. Proposed Model Architecture Components

This section presents the two key components of the proposed UNet-style architecture: the Residual Building Block (top dashed box) and the Spatiotemporal Attention Module (bottom dashed box).

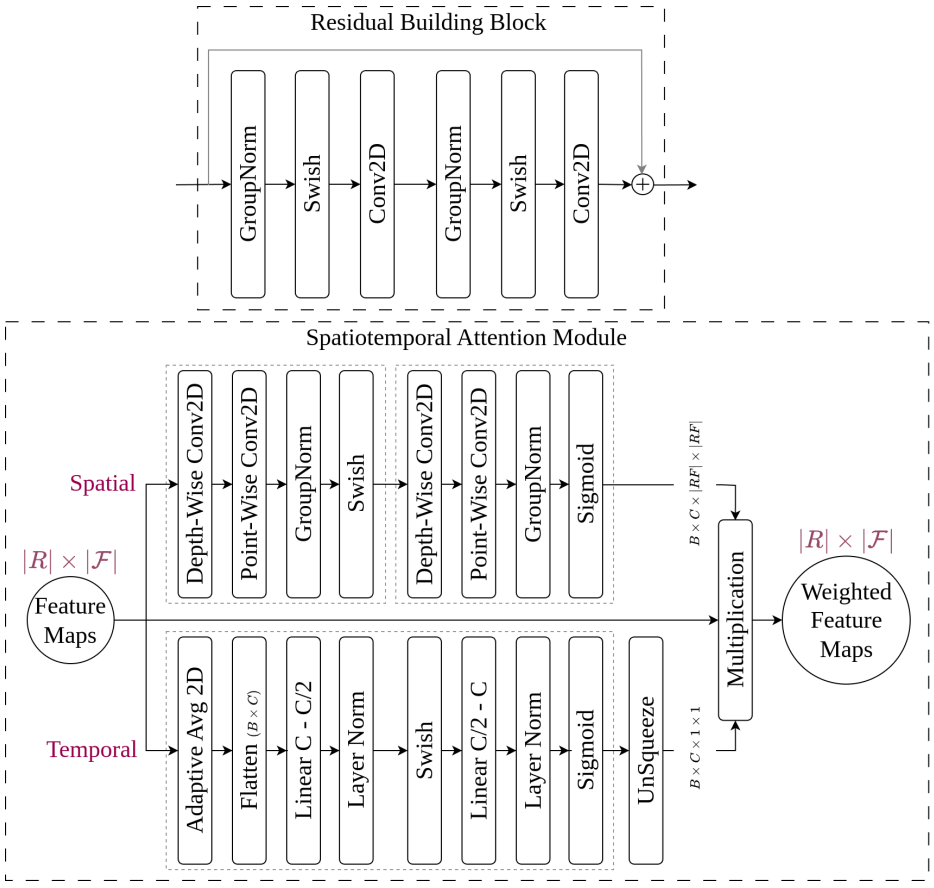

Figure S9. Top dashed-box: The residual building block represents the base component of the model architecture. Bottom dashed-box: The spatiotemporal attention module incorporates two processing streams for input feature maps: (1) Spatial Attention Stream (Top) captures and weights informative spatial regions within individual reference frames. Group normalization is employed to highlight the spatial importance of feature maps corresponding to each input frame. Additionally, depth-wise separable convolutions are used for the efficient processing of large-scale feature maps; (2) the Temporal Attention Stream (Bottom) focuses on capturing temporal dynamics and dependencies. It weights each time step across feature maps extracted from all consecutive frames.

The Residual Building Block serves as the fundamental unit of our architecture. It comprises two sequential series of Group Normalization, Swish activation, and Convolutional layers, designed to enhance feature representation while preserving gradient flow through residual connections. Moreover, the layer arrangement within the Residual Building Block defines a standardized ordering framework for all preceding and succeeding modules and components in the architecture.

We also enhance the skip connections with a Spatiotemporal Attention Module, which operates alongside the diffusion module and feature reduction modules. This module enables the aggregation of feature maps from multiple input reference frames into a unified feature representation before being passed to the decoder. The Spatiotemporal Attention Module consists of two parallel processing streams: Spatial Attention (top) and Temporal Attention (bottom).

The Spatial Attention Stream employs depth-wise separable convolutions, group normalization, and Swish activation to highlight informative spatial regions within individual frames. The Temporal Attention Stream captures dependencies across frames using adaptive average pooling, linear transformations, and layer normalization, followed by Swish or sigmoid activation to generate temporal attention weights. The final attention-enhanced feature maps are computed by element-wise multiplication of the original feature maps with the spatial and temporal attention outputs. This mechanism enables the model to effectively capture both spatial and temporal information, improving segmentation performance.

S9. The Beta Distribution Scheduler

The beta distribution is parameterized by two positive shape parameters, denoted by α and β.

$$f(x, \alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1} (1-x)^{\beta-1} \tag{1}$$

where $x \in [0, 1]$, and $B(\alpha, \beta)$ is the beta function, which is a normalization constant that ensures the total area under the curve equals 1. The beta function is also defined as:

$$B(\alpha, \beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1} dt \tag{2}$$

The mean and variance of the beta distribution are calculated as follows, which perfectly control the randomly generated time steps.

$$\mu = \frac{\alpha}{\alpha + \beta} \tag{3}$$

$$\sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{4}$$

We established a scheduler to determine the values of α and β, assigning distinct beta distribution functions to each level and

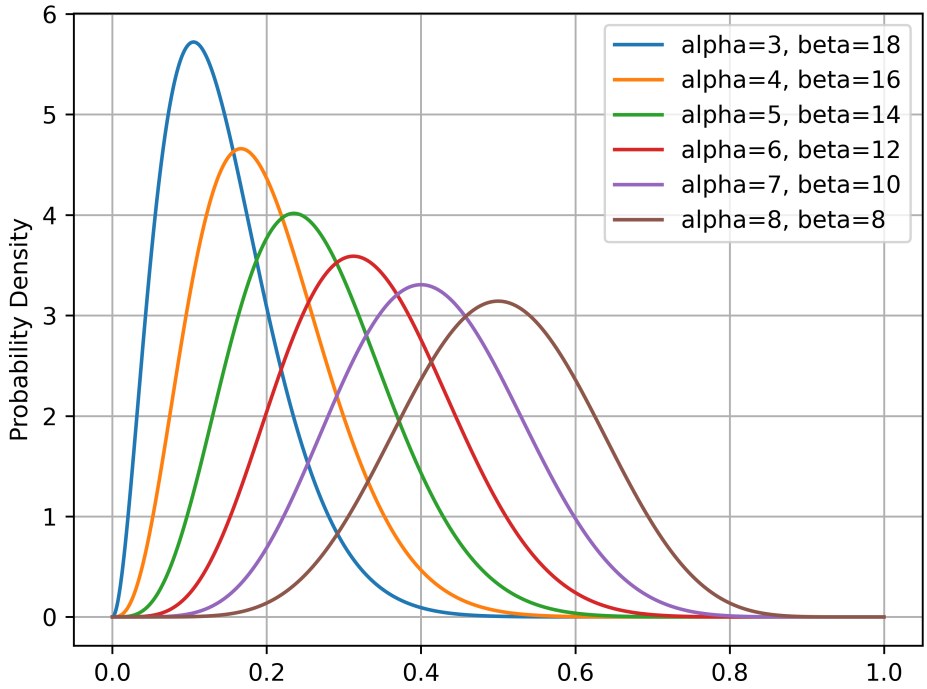

Figure S10. This figure displays the shapes of beta distributions generated by our scheduler for various combinations of α and β values. Each curve corresponds to a distinct beta distribution, illustrating the impact of parameter variation on the distribution's shape. The curves are arranged from left to right for a model with six skip connections, progressing bottom-up from the latent space to the highest resolution of the upsampling path.

thereby selecting diffusion time steps. This scheduling mechanism is governed by the following equations:

$$\alpha = \alpha_0 + l \tag{5}$$

$$\beta = \beta_0 - (\beta_c * l) \tag{6}$$

where α_0, β_0 and β_c represent initial and coefficient hyperparameters, and l is the upsampling level index. The index l begins at the last skip connection and progresses towards the first skip connection, signifying the final upsampling step. Figure S10 depicts the shapes of beta distributions generated using our scheduler across varying values of α and β.

S10. Input Diffusion Process

In this section, we present our strategy for diffusing input images in a patching style. Each reference frame in the mini-batch undergoes diffusion with a patching rate of $P_d = 0.5$, following the forward diffusion process with random time steps ranging from 0 to 1000. Figure S11 illustrates an example of an input image after applying this diffusion process.

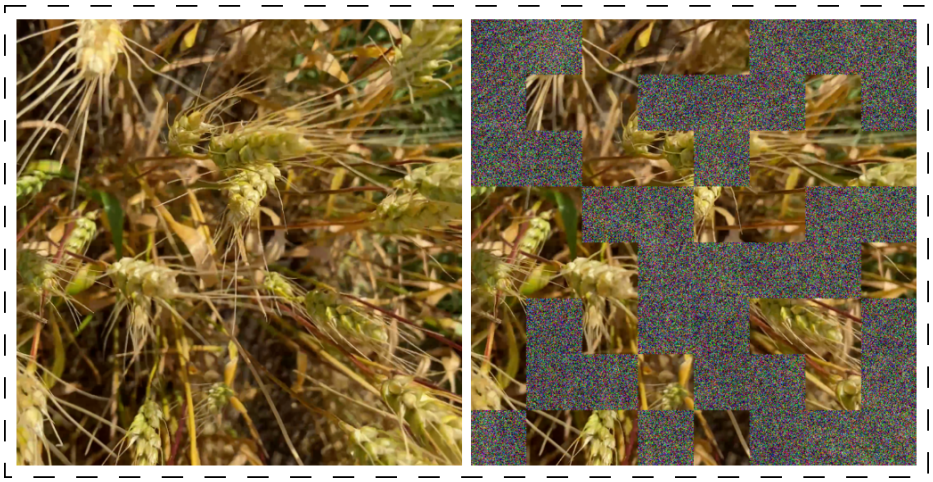

Figure S11. A depiction of an image frame alongside its diffused version, produced through the forward diffusion process.

S11. Ablation Studies and Experimental Insights

This section details the ablation studies performed to evaluate the performance of our proposed architecture across various configurations, both in the current setting and under alternative scenarios. Specifically, the $\mathbb{IM}_{\text{Synt}}$, $\mathbb{VM}_{\text{Synt}}$, and $\mathbb{XM}_{\text{Synt}}$ models were developed on individual frames (only for $\mathbb{IM}_{\text{Synt}}$) and video clips, from our $\mathbb{S}_{train}$ and Δ datasets. Similarly, the $\mathbb{IM}_{\text{Pseu}}$, $\mathbb{VM}_{\text{Pseu}}$, $\mathbb{XM}_{\text{Pseu}}$ were trained on individual frames (only for $\mathbb{IM}_{\text{Pseu}}$) and video clips from the $\mathbb{P}_{train}$ and $\mathbb{P}_{valid}$ datasets. All experiments are conducted under consistent configurations and settings, except for those settings that were intentionally modified, which are clearly explained in the following list:

1. **QAI**: This experiment evaluates the performance of frame-level models trained for image segmentation and reconstruction tasks, where the query frame is used as input in a conventional segmentation setting. The architecture includes only the encoder, decoder, and heads, excluding the skip-spatiotemporal attention modules of the proposed video model.
   - $\mathbb{IM}_{\textbf{Synt}}$: This model, evaluated without pretraining, shows moderate performance in comparison to other configurations, where a slight performance improvement is observed.
   - $\mathbb{IM}_{\textbf{Pseu}}$: Pretrained on $\mathbb{IM}_{\text{Synt}}$, this configuration demonstrates a noticeable performance improvement, outperforming the previous model by a significant margin.

2. **PPQAI**: This experiment evaluates the model's performance when fully integrated with all components for DVOS, starting from the pretrained frame-based models. This partial pretraining allows the video models to benefit from the skip modules, which are still untrained in this configuration. Asterisks denote the best-performing models among various configurations. The full architecture is trained for the DVOS task, with the reference frame as input, and predicts the query frame and mask as output.
   - $\mathbb{VM}^{*}_{\textbf{Synt}}$: Partially pretrained with $\mathbb{IM}_{\text{Synt}}$, this model performs moderately, achieving solid results.
   - $\mathbb{VM}^{*}_{\textbf{Pseu}}$: Pretrained with $\mathbb{VM}^{*}_{\text{Synt}}$, this model shows a significant improvement over all other models, achieving the best overall performance across all configurations.

3. **RRAI**: This experiment investigates the performance when frame-based models are trained with random reference frames as input. For each training sample, a single input frame is randomly selected from the set of reference frames, and the query frame and its mask are predicted in a frame-level frame/mask prediction training paradigm.
   - $\mathbb{IM}_{\textbf{Synt}}$: Without pretraining, this model demonstrates relatively poor performance compared to other configurations, indicating the need for better initialization.

- $\mathbb{IM}_{\textbf{Pseu}}$: Pretrained with $\mathbb{IM}_{\text{Synt}}$, this model shows a slight improvement, though its performance still remains relatively low when compared to other, better-pretrained models.

4. **PPRAI**: This experiment assesses the performance when the model is partially pretrained on the RRAI $\mathbb{IM}_{\text{Pseu}}$ model. The entire architecture is trained for the DVOS task, with the reference frame as input, and predicting the query frame and mask as output.
   - $\mathbb{VM}_{\textbf{Synt}}$: Pretrained with $\mathbb{IM}_{\text{Synt}}$, this configuration provides moderate results compared to the more optimized models in other settings.
   - $\mathbb{VM}_{\textbf{Pseu}}$: Pretrained with $\mathbb{VM}_{\text{Synt}}$, this model shows a noticeable performance improvement.

5. **FS**: This experiment evaluates DVOS models trained from scratch, without any pretraining.
   - $\mathbb{VM}_{\textbf{Synt}}$: Trained from scratch, this model shows poor performance relative to the other models that benefit from pretraining.
   - $\mathbb{VM}_{\textbf{Pseu}}$: Trained from scratch with $\mathbb{VM}_{\text{Synt}}$.

6. **NDNA**: This experiment tests the model's performance trained from scratch, without input and skips diffusion and without applying color augmentation.
   - $\mathbb{VM}_{\textbf{Synt}}$: Without these components, the model's performance is significantly reduced, resulting in notably lower scores.
   - $\mathbb{VM}_{\textbf{Pseu}}$: Despite the absence of diffusion and color augmentation, this model still shows an improvement, but its performance is relatively poorer compared to our best-performing models.

7. **RFI**: This experiment evaluates the model using a frame interval of 2. The first training stage on synthetic data is skipped due to the lack of natural motion in frame-per-second settings. Here, the best-performing model, $\mathbb{VM}^{*}_{\text{Synt}}$, is used as the pretraining stage for the second stage. The model is first trained and tested with a frame interval of 2 and later tested with a frame interval of 1, identified as **RFITI1**.
   - $\mathbb{VM}^{=}_{\textbf{Pseu}}$: Tested with a frame interval of 2, this configuration demonstrates a slight performance drop compared to the model tested with a frame interval of 1, as seen in **RFITI1**.

8. **RFITI1**: This experiment tests the model with a frame interval of 1.
   - $\mathbb{VM}^{=}_{\textbf{Pseu}}$: With a frame interval of 1, this model shows a clear improvement compared to the performance of **RFI** tested with a frame interval of 2, suggesting better performance with a shorter frame interval.

9. **RRFI**: This experiment investigates random intervals of 1 or 2 for the reference frames.
   - $\mathbb{VM}_{\textbf{Pseu}}$: Trained with reference frame intervals randomly chosen between 1 or 2, this model shows a modest performance drop compared to the best-performing model $\mathbb{VM}^{*}_{\text{Pseu}}$ from the PPQAI experiment.

10. **XMem Model**: This experiment compares our model with the XMem [3] model. We used XMem as our base model due to its strong performance in VOS. XMem takes the first-frame segmentation mask as input and effectively propagates object masks across subsequent frames using a dynamic memory mechanism. Its ability to balance segmentation accuracy and computational efficiency makes it well-suited for practical applications. Furthermore, XMem remains a state-of-the-art method on multiple VOS benchmarks, providing a strong foundation for evaluating our proposed approach. As discussed in the paper, this model manipulates the given first frame's mask and generates the segmentation mask for the query mask while mainly ignoring the inputted reference and query frames, achieving higher quantitative scores by generating identical masks to the ground truth. However, in the presence of noisy input masks, it fails to predict the objects of interest, which are examined accurately through visual inspection of its predicted masks (Video 1 in Figure S12).
    - $\mathbb{XM}_{\textbf{Synt}}$: Evaluated with XMem-s012 [3] pretraining, this model shows better performance compared to other configurations of our models.
    - $\mathbb{XM}_{\textbf{Pseu}}$: Pretrained with $\mathbb{XM}_{\text{Synt}}$, this configuration outperforms all previous configurations, setting the bar as the best-performing model.

Table S4. Ablation Study Results Across Different Model Configurations. This table presents the results of various ablation experiments conducted to evaluate the performance of the proposed model under different configurations. The experiments assess the impact of different model components, pretraining strategies, and training conditions on segmentation performance, measured by Dice and IoU metrics.

Experiment	Model	Pretrained On	Metric	Ψ	Γ	$\mathbb{P}_{test}$
Query as Input (QAI)	$\mathbb{IM}_{\text{Synt}}$	None	Dice	0.729	0.408	0.145
			IoU	0.580	0.278	0.0861
	$\mathbb{IM}_{\text{Pseu}}$	$\mathbb{IM}_{\text{Synt}}$	Dice	0.759	0.761	0.647
			IoU	0.621	0.619	0.513
Partially Pretrained on QAI (PPQAI)	$\mathbb{VM}^{*}_{\text{Synt}}$	$\mathbb{IM}_{\text{Pseu}}$	Dice	0.482	0.453	0.244
			IoU	0.335	0.307	0.150
	$\mathbb{VM}^{*}_{\text{Pseu}}$	$\mathbb{VM}^{*}_{\text{Synt}}$	Dice	**0.650**	**0.791**	**0.679**
			IoU	**0.493**	**0.657**	**0.542**
Random Reference as Input (RRAI)	$\mathbb{IM}_{\text{Synt}}$	None	Dice	0.304	0.169	0.036
			IoU	0.193	0.100	0.200
	$\mathbb{IM}_{\text{Pseu}}$	$\mathbb{IM}_{\text{Synt}}$	Dice	0.284	0.376	0.378
			IoU	0.178	0.248	0.254
Partially Pretrained on RAI (PPRAI)	$\mathbb{VM}_{\text{Synt}}$	$\mathbb{IM}_{\text{Pseu}}$	Dice	0.487	0.416	0.315
			IoU	0.337	0.277	0.204
	$\mathbb{VM}_{\text{Pseu}}$	$\mathbb{VM}_{\text{Synt}}$	Dice	0.611	0.773	0.672
			IoU	0.451	0.634	0.532
From Scratch (FS)	$\mathbb{VM}_{\text{Synt}}$	None	Dice	0.423	0.320	0.118
			IoU	0.290	0.212	0.069
	$\mathbb{VM}_{\text{Pseu}}$	$\mathbb{VM}_{\text{Synt}}$	Dice	0.647	0.782	0.670
			IoU	0.489	0.647	0.535
No Diffusion & No Color Augmentation (NDNA)	$\mathbb{VM}_{\text{Synt}}$	None	Dice	0.347	0.080	0.018
			IoU	0.232	0.046	0.010
	$\mathbb{VM}_{\text{Pseu}}$	$\mathbb{VM}_{\text{Synt}}$	Dice	0.664	0.578	0.675
			IoU	0.504	0.429	0.541
Reference Frame Interval of 2 (RFI)	$\mathbb{VM}^{=}_{\text{Pseu}}$	$\mathbb{VM}^{*}_{\text{Synt}}$	Dice	0.555	0.735	0.622
			IoU	0.398	0.586	0.476
RFI Tested on Interval of 1 (RFITI1)	$\mathbb{VM}^{=}_{\text{Pseu}}$	$\mathbb{VM}^{*}_{\text{Synt}}$	Dice	0.645	0.780	0.671
			IoU	0.484	0.641	0.532
Random Reference Frame Intervals of 1 or 2 (RRFI)	$\mathbb{VM}_{\text{Pseu}}$	$\mathbb{VM}^{*}_{\text{Synt}}$	Dice	0.533	0.741	0.623
			IoU	0.383	0.595	0.478
XMem Model [3]	$\mathbb{XM}_{\text{Synt}}$	XMem-s012 [3]	Dice	0.794	0.454	0.448
			IoU	0.668	0.314	0.302
	$\mathbb{XM}_{\text{Pseu}}$	$\mathbb{XM}_{\text{Synt}}$	Dice	**0.831**	**0.811**	**0.835**
			IoU	**0.716**	**0.690**	**0.726**

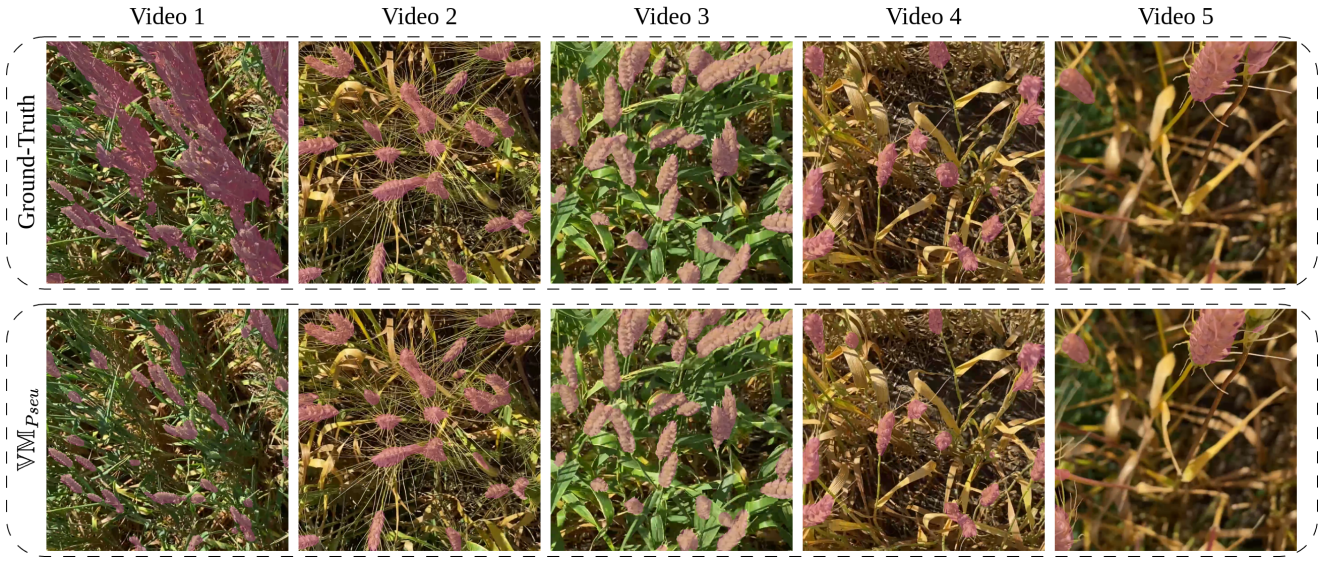

Figure S12. Segmentation prediction performance of $\mathbb{VM}^{*}_{\text{Pseu}}$ across various videos in $\mathbb{P}_{test}$.