Supplemental Material for

Semantic Attention Flow Fields for Monocular Dynamic Scene Decomposition

Yiqing Liang Eliot Laidlaw Alexander Meyerowitz Srinath Sridhar James Tompkin
Brown University

This document contains a related work comparison table (Appendix A), details and rationale for our dataset construction (Appendix B), justification of design choices throughout the approach (Appendix D), and implementation details (Appendix E). Given the numerous figures and tables, many sections begin on a new page for easier reading.

Further, we include a supplemental website that highlights key findings and allows comparison between different methods and ablations for the different SAFF volumes. Please find it here: https://visual.cs.brown.edu/saff

A. Expanded Related Work Table

We include an expanded table of related work (Tab. 1). This includes additional work in 2D image object-centric learning (IODINE [7], MONET [2], Slot Attention [18]), 2D videos (SIMONe [9], SAVi [10]), a method that considers light field input (COLF [22]), and works that add semantic information to fields (PNF [14]) including from mask supervision (Object-NeRF [26]).

Table 1: An comparison of related work in scene decomposition shows the unstudied area of real-world dynamic 3D segmentation without explicit segmentation clues. We investigate whether saliency can provide similar clues for the monocular video case. From this table, the closest related method is N3F; however, they take user input to define their segmentation.
Learning: Large-scale training data:
T: Supervised task-specific data.
P: Generic features (e.g., ImageNet).
×: No features used.

	Dynamic (video)	Monocular	Real world	3D	No seg. clue	Learning	Adaptive # objects	Object-level
IODINE[7]	×	✓	×	×	✓	×	×	✓
MONET[2]	×	✓	×	×	✓	×	×	✓
Slot Attention[18]	×	✓	×	×	Mask	T	×	✓
SIMONe[9]	✓	✓	×	×	✓	×	×	✓
SAVi[10]	✓	✓	×	×	Mask	T	×	✓
SAVi++[5]	✓	✓	✓	×	Mask	T	×	✓
ObSuRF[23]	×	×	×	✓	✓	×	×	✓
uORF[28]	×	×	×	✓	✓	×	×	✓
COLF[22]	×	×	×	✓	✓	×	×	✓
PNF[14]	✓	✓	✓	✓	Mask	×	×	✓
Object-NeRF [26]	×	×	✓	✓	Mask	×	×	✓
DFF[11]	×	×	✓	✓	User	P	✓	✓
N3F[24]	✓	✓	✓	✓	User	P	✓	✓
ProposeReduce[17]	✓	✓	✓	×	✓	T	✓	✓
NSFF[15]	✓	✓	✓	✓	Mask	×	N/A	N/A
D^2NeRF[25]	✓	✓	✓	✓	✓	×	N/A	N/A
SAFF (this paper)	✓	✓	✓	✓	✓	P	✓	✓

B. Dynamic Scene Dataset (Masked) Creation

To perform experiments on segmentations, we manually annotate object masks for every view and time step in the NVIDIA Dynamic Scene Dataset[27] and in the DyCheck dataset [6]. Some object masks are visualized in Fig. 1.

One natural question is why we do not use existing unsupervised video segmentation benchmarks like DAVIS [3] for evaluation. When testing these videos, we found that there is little camera motion in most of these videos. This causes classic structure-from-motion approaches like COLMAP [21] to fail to estimate camera poses, thus we cannot optimize SAFF on these sequences. Concurrent tangential work attempts to improve this situation with better pose estimation [12, 16, 29].

Further, even if we did have poses, there could be no evaluation of the sequences in a 3D sense because the scenes were only ever captured with a single camera. While collecting ground truth 3D segmentation for dynamic casual videos is difficult, as discussed in the main paper, our approach allows evaluation at novel spacetime views as the scene was initially captured with 12 cameras. This gives a sense of the ability of the method to perform consistent 3D segmentation of the dynamic scene as captured by a simulated monocular camera view (main paper, Sec, 4, paragraph 'Data').

We additionally mask five sequences within the DyCheck dataset [6]. These are captured from a single smartphone RGB camera, and so do not have large disparity from frame to frame but may have large motion. Hold-out images can be taken by not processing some images in the videos, but these require less significant interpolation ability to render novel spacetime views.

Different Data Data in different modalities like infrared or depth data are beyond the scope of our work, but better depth information through, say, time of flight imaging would be a valuable addition to monocular reconstruction.

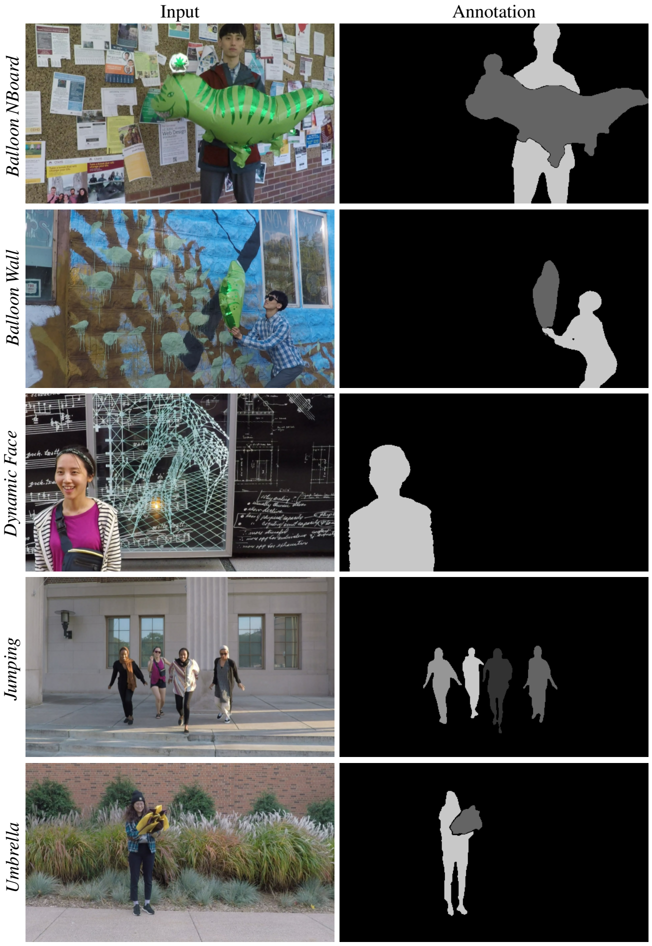

Figure 1: **Manually annotated object masks.** We store annotation masks as grayscale images. The background is assigned as pixel value 0, and foreground instances are each assigned a unique non-zero value.

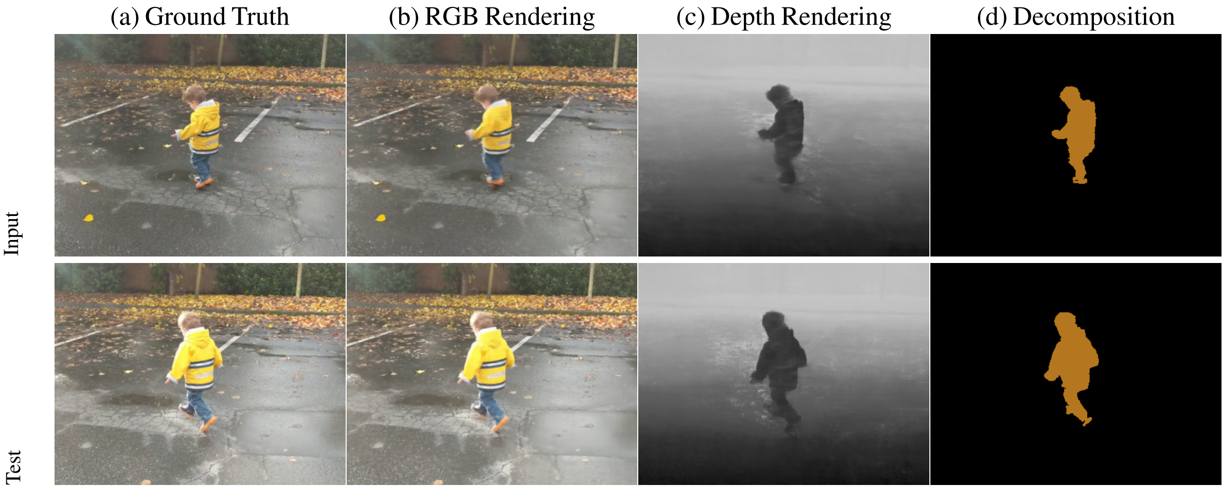

Figure 2: SAFF's rendering and decomposition result on the *kid-running* scene.

C. NSFF Extra Scene

Apart from Dynamic Scene Dataset (Masked), the NSFF [15] authors shared a *kid-running* scene to help explain their method and as a test example in their code release. Along with the smartphone sequences in DyCheck [6], this is also a 'true' monocular sequence captured with a single handheld camera. We demonstrate SAFF's decomposition result on this sequence to further demonstrate our semantic attention approach (Fig. 2).

D. Design Choices

D.1. Underlying Dynamic NeRF Approach

View synthesis and depth First, we evaluate whether RGB view synthesis performance is affected by adding more heads to the MLP. We find that it is not affected (Tab. 2). D^{2}NeRF's hyper-spacetime deformation has trouble reconstructing images on this dataset, producing distorted dynamic objects or failing to freeze time.

In Fig. 3, we show qualitatively that SAFF does not degrade view synthesis or depth quality compared to NSFF [15], while D^{2}NeRF struggles with our data.

Table 2: **SAFF does not degrade image quality.** Adding semantics and attention on the same backbone produces the same image quality as NSFF [15]. Metrics: L is LPIPS ($[0, 1]$, lower is better), S is SSIM ($[0, 1]$, higher is better), P is PSNR ($[0, \infty]$, higher is better).

	Input			Fix Cam 0			Fix Time 0		
	L ▼	S ▲	P ▲	L	S	P	L	S	P
D^{2}NeRF	0.115	0.790	23.91	0.228	0.565	18.04	0.344	0.309	13.85
NSFF w/o masks	0.070	0.805	23.92	0.100	0.762	21.68	0.302	0.386	14.92
SAFF (ours)	0.070	0.805	23.92	0.100	0.762	21.70	0.302	0.386	14.93

Why does D^{2}NeRF struggle? The main distinction is that D^{2}NeRF is a deformation-based method while NSFF and SAFF are flow-based methods. For D^{2}NeRF, the scene is reconstructed in a canonical space and deformed to render the results. D^{2}NeRF struggles with larger motion in the scene—in the NVIDIA dataset, it is notably more difficult to find temporal correspondence within because frames are spatially far apart (unlike other monocular datasets created from one video camera only). Given only a monocular video to describe a scene with large camera motion *and* large object motion, it appears difficult to faithfully reconstruct both the canonical space and the deformation. In comparison, D^{2}NeRF produces good RGB reconstructions on the Nerfies [20] dataset because both the camera and scene motion are smaller than in the NVIDIA Dynamic Scene Dataset.

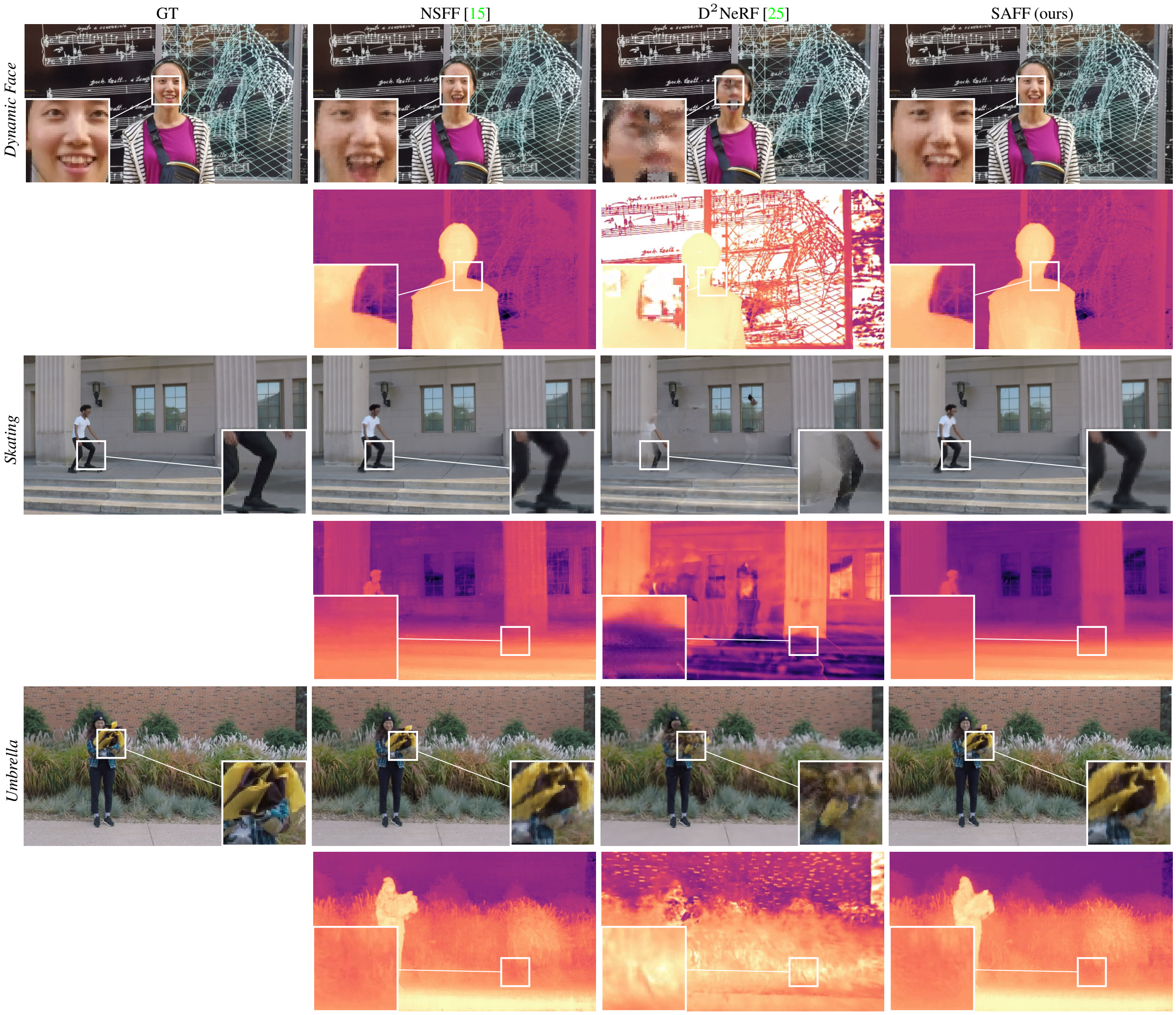

Figure 3: **SAFF does not degrade novel spacetime view synthesis or depth quality.** D^{2}NeRF struggles on the NVIDIA Dynamic Scene Dataset sequences because of the large motions between cameras.

D.2. 3D vs. 2D Projected vs. 4D Spacetime Clustering

In our saliency-aware clustering step, the elbow k-means method can take as input the per-pixel semantics features that have been sampled from 3D points in the volume (**in 3D**), or that have been rendered from the volume back to the 2D plane. Even though a volume is reconstructed, we find performance is somewhat worse when clustering in 3D than in projected 2D space (Tab. 3; and in main paper). With respect to evaluation, it is difficult to collect ground truth segmented 3D data for dynamic real world scenes (none exist to our knowledge); this remains future work.

So, why might clustering in 3D lead to worse segmentations? Given monocular input from narrow baselines and dynamic scenes, the reconstruction can be imprecise with noisy geometry. This is in contrast to dynamic scenes captured with multi-camera setups or static scenes captured with wide baselines. As semantics and saliency use the same estimated geometry as radiance, clustering in the 3D volume introduces inaccuracy in the decomposition result, thus reducing the performance quantitatively and qualitatively. We visualize the volume as a point cloud from sampled 3D points in Fig. 4. Although the volume looks natural from the training view, erroneous regions are visible when the camera pose is far away from the training view, especially on the dynamic objects.

Given direct clustering in 3D suffers from the narrow-baseline issue, we introduce another variant (**in 4D spacetime**) that also clusters upon a 3D position for each input pixel using the recovered volume density (from the estimated depth) and the timestep. While not the same as a volumetric clustering with some dense sampling, this sparse alternative is a reasonable computational compromise as our scenes contain opaque objects without participating media. We concatenate the spacetime coordinates with the semantic features for each pixel, then empirically adjust their relative weights to increase foreground segmentation performance.

In principle, this could exploit the underlying geometry to provide better edges or more instance awareness to the method. However, in practice, this does not reliably happen (Tab. 3). Any error in alignment between the semantic information and the geometry causes the clustering to confuse elements, e.g., semantics for the same object at different depths over edge boundaries. As a consequence, semantically-different entities may not be correctly separated (*Truck*), or are missing parts (*Jumping*, Fig. 5). As such, we use only projected 2D semantic features as input to the clustering.

However, even though we use semantic-attention pyramids to increase the geometric resolution of the DINO-ViT features significantly, and even though volume integration increases these still (e.g., main paper, Figure 2), one might ask whether the optimization routine could also help further align the geometry and semantic features during volume integration. This brings us to the next subsection.

Table 3: **3D volume clustering produces worse 2D segmentations than projected 2D clustering.** This is because precise localization of 3D geometry is difficult from monocular inputs for dynamic scenes. 4D spacetime clustering also produces worse foregrounds. Metric: Adjusted Rand Index ($[-1, 1]$, higher is better).

	Input	Fix Cam 0	Fix Time 0
SAFF (ours)	**0.653**	**0.634**	**0.625**
in 3D	0.594	0.578	0.566
in 4D spacetime	0.482	0.464	0.452

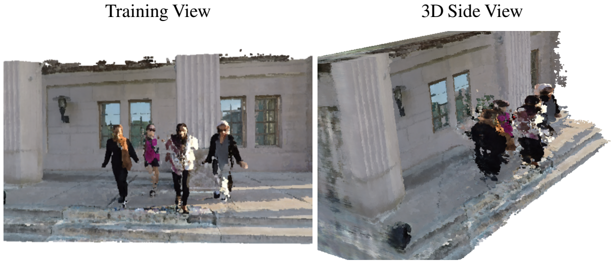

Figure 4: **3D samples.** Erroneous geometry reconstruction at regions invisible during training harms 3D clustering quality.

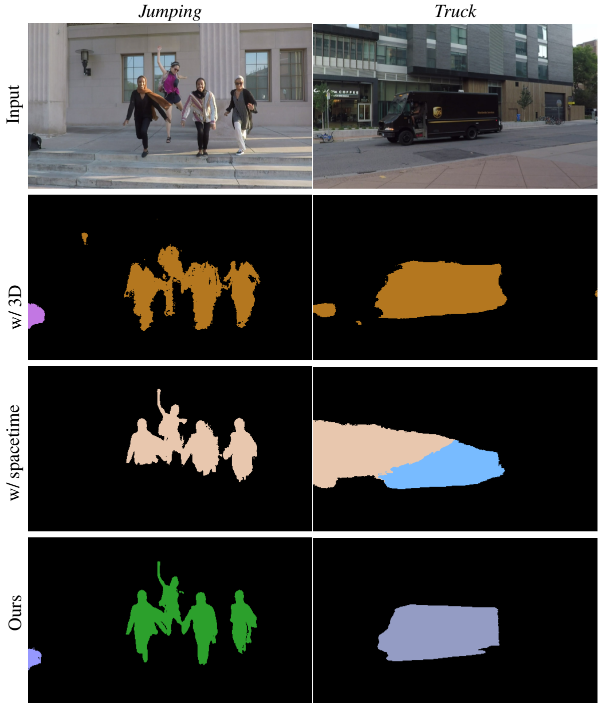

Figure 5: Clustering upon sampled 3D volume features or adding 4D spacetime features produces worse foregrounds qualitatively than just projected 2D feature clustering. This is because the narrow-baseline monocular input sometimes leads to noisy geometry estimation, and because the semantic edges must conform to the same geometry of the scene as the radiance. In 4D spacetime clustering, some clusters are confused or unnecessarily merged.

Table 4: **Decaying semantic and attention information produces overall foregrounds.** While geometric alignment improves, semantic meaning also shifts, which harms the ability of the model to correctly identify salient objects. Metric: Adjusted Rand Index ($[-1, 1]$, higher is better).

	Input	Fix Cam 0	Fix Time 0
SAFF (ours)	**0.653**	**0.634**	**0.625**
w/ decay	0.592	0.568	0.554

D.3. Decaying Semantics and Attention

One way to improve the alignment of depth edges and semantic and attention features is through decaying their reconstruction loss through training. This decay happens to the depth and optical flow priors, and for those channels of information the decay provides freedom to the optimization to refine the spacetime density once initialized with respect to the self-consistent multi-view and scene flow constraints. Intuitively, decaying the semantics and attention reconstructions would also provide more freedom to further optimize the spacetime density to minimize the self-consistent multi-view and scene flow constraints when the semantics disagreed.

However, in the main paper, we describe that semantics and attention are not priors—there is no self-consistency for semantics to constrain their values, thus, after decay the optimization is free for them to vary inconsistently and so for their reprojection to lose useful meaning. This could have unwanted consequences.

To investigate this design choice, we implement variant **w/ decay** in which we use the same decaying mechanism as depth and optical flow on $\mathcal{L}_{\hat{\mathbf{s}}}$ and $\mathcal{L}_{\hat{\mathbf{a}}}$. We also decay $\mathcal{L}_{\hat{\mathbf{s}}_{i\to j}}$ and $\mathcal{L}_{\hat{\mathbf{a}}_{i\to j}}$, because the semantics are not necessarily consistent with the spacetime geometry.

Qualitatively, adding decay does better align the semantics and attention fields with the geometry, e.g., in *Skating*, the space between skater's legs are better segmented; however, the clustering performance degrades all over the image (Fig. 6), e.g., in *Skating*, now including a sconce and unwanted floor details (zoom in). This is also reflected quantitatively (Tab. 4).

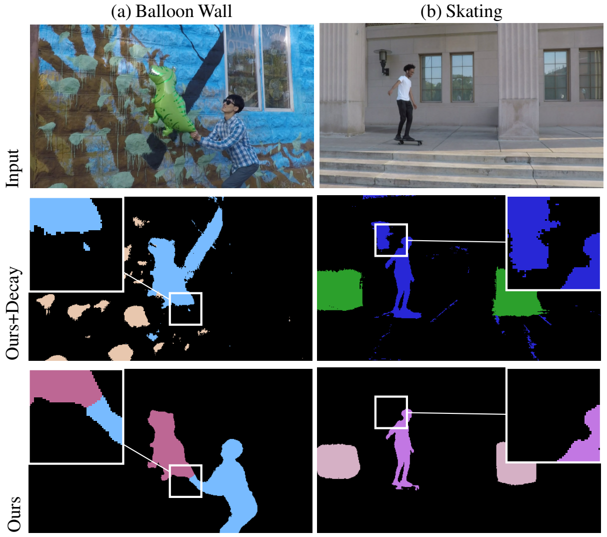

Figure 6: **Decaying semantics and attention leads to missing objects and unwanted objects.** With no self-consistent constraint, the optimization is more free to adjust the meaning of regions. While this can increase edge detail, it creates worse overall results.

D.4. Pyramid Construction

Figure 7 provides a diagrammatic example for how we construct our feature pyramids. We also provide algorithm pseudocode (Algorithm 1).

As mentioned in the main paper, given the pyramid layers, we begin conceptually from a weighted sum of three layers for semantics using $\lambda_{\hat{\mathbf{s}}} = \{1/3, 1/3, 1/3\}$ and with coarsest whole-image attention with $\lambda_{\hat{\mathbf{a}}} = \{1, 0, 0\}$. This already gives quantitatively better decomposition performance than without using a pyramid (**w/o pyr**; Tab. 5). However, the optimized semantic field does not correspond as well to scene geometry and is more influenced by error in the coarsest layer semantics and attention than our approach. For instance, the human is not identified as salient in *Balloon NBoard* (Fig. 8). Additionally, as the finer layer sliding windows do not typically contain the object of interest in boundary regions, the extracted features are incorrect. This causes unwanted clusters to appear around image edges, e.g., the gray cluster in the top right corner in *Umbrella* (Fig. 8).

To increase semantic and attention resolution, we increase $\lambda_{\hat{\mathbf{s}}}$'s dependency on finer layers to $\{4/9, 4/9, 1/9\}$. To deal with the boundary issues, we decrease the weight of fine layers towards the boundary back to $\{1/3, 1/3, 1/3\}$. The image boundary problem is resolved in *Umbrella* (Fig. 8). However, there is still a mismatch between the fidelity of the semantics and saliency (the head disappears in *Dynamic Face*), which is also reflected quantitatively (Tab. 5).

Thus, we use the same weight proportions and boundary reduction for both semantics and attention. This strikes a balance between correct edges from fine layers and whole object features from coarse layers, and mitigates the feature noise around image boundaries. This yields the best overall results both qualitatively on balance (Fig. 8) and quantitatively (Tab. 5).

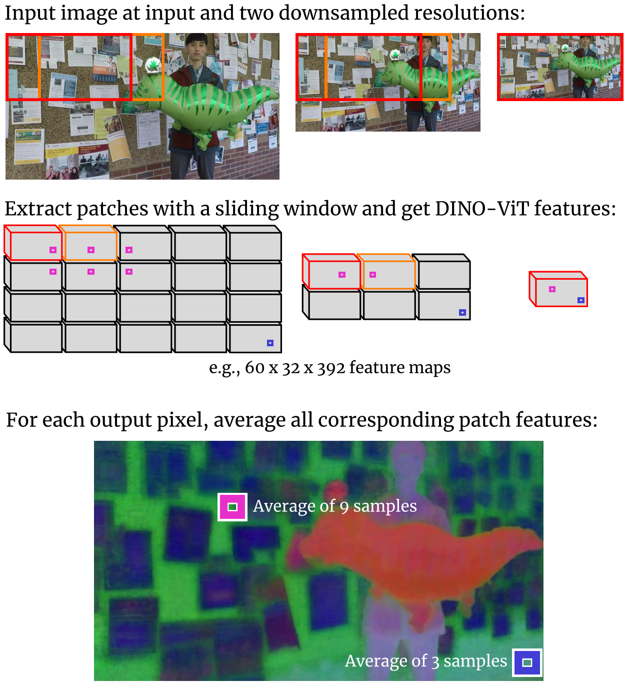
Input image at input and two downsampled resolutions:
Extract patches with a sliding window and get DINO-ViT features:
e.g., 60 x 32 x 392 feature maps
For each output pixel, average all corresponding patch features:
Average of 9 samples
Average of 3 samples

Figure 7: **Pyramid construction example.** This attempts to balance feature quality with computational cost by aggregating overlapping feature extraction blocks from different image resolutions.

Table 5: **Pyramid weighting choice.** Even though in our final algorithm the coarse layer has smaller weight, it balances high resolution edges from fine layers and whole object features from coarse layers while reducing geometry conflicts, and mitigate the feature noise around edges.
Metric: Adjusted Rand Index ($[-1, 1]$, higher is better).

	Input	Fix Cam 0	Fix Time 0
SAFF			
w/o pyr $\hat{\mathbf{s}}, \hat{\mathbf{a}}$	0.545	0.532	0.521
SAFF (ours)	**0.653**	**0.634**	**0.625**
w/ pyr $\lambda_{\hat{\mathbf{a}}} = \{1, 0, 0\}$	0.620	0.598	0.592
w/ pyr $\lambda_{\hat{\mathbf{a}}} = \{1, 0, 0\}$, $\lambda_{\hat{\mathbf{s}}} = \{1/3, 1/3, 1/3\}$	0.631	0.612	0.601

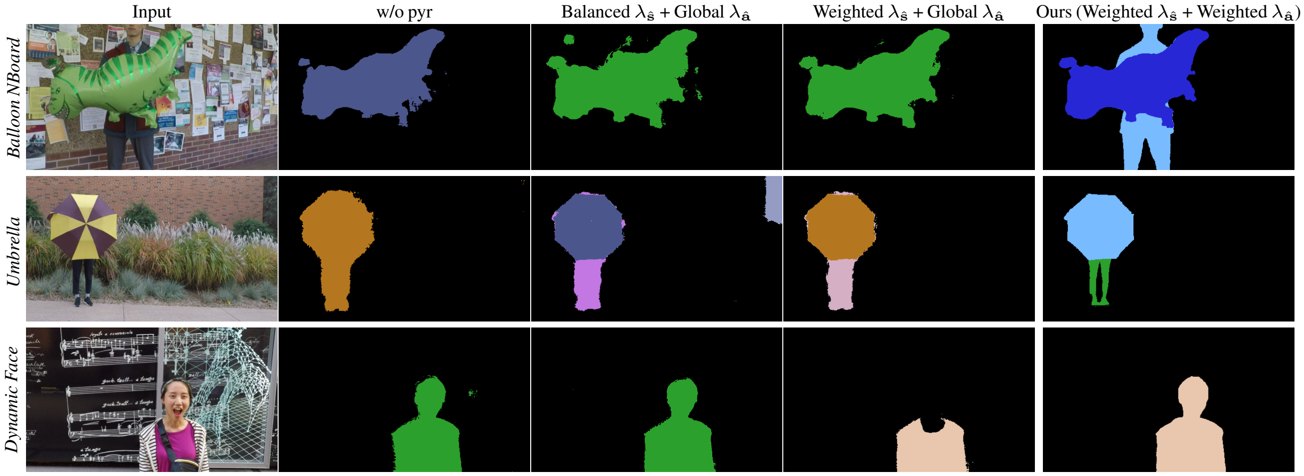
Input
w/o pyr
Balanced $\lambda_{\hat{\mathbf{s}}}$ + Global $\lambda_{\hat{\mathbf{a}}}$
Weighted $\lambda_{\hat{\mathbf{s}}}$ + Global $\lambda_{\hat{\mathbf{a}}}$
Ours (Weighted $\lambda_{\hat{\mathbf{s}}}$ + Weighted $\lambda_{\hat{\mathbf{a}}}$)
Balloon NBoard
Umbrella
Dynamic Face

Figure 8: **Pyramid construction varies final output quality after clustering.** Without the pyramid, key objects are missing or undersegmented. Our approach captures the objects with fewest errors compared to the variants.
Balanced $\lambda_{\hat{\mathbf{s}}}$ + Global $\lambda_{\hat{\mathbf{a}}}$: $\lambda_{\hat{\mathbf{s}}} = \{1/3, 1/3, 1/3\}$ with $\lambda_{\hat{\mathbf{a}}} = \{1, 0, 0\}$
Weighted $\lambda_{\hat{\mathbf{s}}}$ + Global $\lambda_{\hat{\mathbf{a}}}$: semantic attention pyramid with $\lambda_{\hat{\mathbf{a}}} = \{1, 0, 0\}$
Ours (Weighted $\lambda_{\hat{\mathbf{s}}}$ + Weighted $\lambda_{\hat{\mathbf{a}}}$): Our semantic attention pyramid

Algorithm 1 Pyramid Construction Algorithm (Example). Given an RGB image I of size $(H \times W \times 3)$ and a D-dimensional feature extractor $E : (A \times B \times 3) \mapsto (A/4 \times B/4 \times D)$, produce a processed feature map for the input image.

```
Input: $I \in R^{H \times W \times 3}, E \leftarrow dino\_vit8, H, W, D$
  $level0 \leftarrow I$
  $level1 \leftarrow$ DOWNSAMPLE$(I, 480, 256)$    ▷ Downsample image to 480 × 256.
  $level2 \leftarrow$ DOWNSAMPLE$(I, 240, 128)$
  $all\_patches\_in\_imagespace \leftarrow []$
  for $level \in [level0, level1, level2]$ do
      $x \leftarrow 0$
      while $x + 240 \leq W$ do
          $y \leftarrow 0$
          while $y + 128 \leq H$ do
              $patch \leftarrow level[y : y + 128, x : x + 240, :]$
              $feature\_patch \leftarrow$ UPSAMPLE$(E(patch), 240, 128)$
              $patch\_in\_imagespace \leftarrow$ NEW_NULL_ARRAY$(H, W, D)$    ▷ To hold features in their input image location.
              $patch\_in\_imagespace[y : y + 128, x : x + 240, :] \leftarrow feature\_patch$
              $all\_patches\_in\_imagespace$.APPEND$(patch\_in\_imagespace)$
              $y \leftarrow y + 64$
          end while
          $x \leftarrow x + 64$
      end while
  end for
  $output\_full \leftarrow$ NON_NULL_AVERAGE$(all\_patches\_in\_imagespace)$    ▷ Element-wise average of non-null values.
  $output \leftarrow$ PCA$(output\_full, 64)$
```

E. Implementation Details

Semantics and Attention Following Amir *et al.* [1], for semantics we extract the 384-dim. 'key' facet from the 11th layer of DINO-ViT, and for saliency we extract the 1-dim. 'attention' facet from the 11th layer. For the pyramid, we use three levels. The coarsest level 0 is downsampled to 240 × 128. The finest level 2 is the input RGB size, with the mid level 1's size set between the two. For each level, we use DINO-ViT as a sliding window of size 240 × 128 to extract a (8×8 patch; stride 4) 60 × 32 feature map (i.e., level 0 has only one window position). For fast computation, we set window stride to 64. Once extracted, we upsample and place each feature map within level 2's frame. Then, for each pixel, we mean average all features from all windows that intersected it. Finally, to fit within GPU VRAM, we perform PCA on all images' normalized extracted pyramid $\hat{\mathbf{s}}$, $\hat{\mathbf{a}}$ features and keep the most important 64 dimensions.

Clustering Any clustering method has hyperparameters (or thresholds) for it to make decisions about cluster assignment. We use the same hyperparameters for all sequences.

We use the GPU via Faiss [8]. Again to fit within GPU VRAM, we uniformly sample every fifth point for elbow-k finding, then propagate cluster assignment to all points given k. We set the elbow-k threshold to 0.975 and the max cluster number to 25, with 10 trials per k. We cluster on direction and so normalize each pixel's vector.

Object extraction To extract an object from the volume, we sample 3D points along each input ray, then compare their semantics to to existing projected 2D semantic cluster centroids. We assign each 3D point to its closest centroid. Then, we set non-salient cluster label points to have zero density.

Post process All quantitative results are *without* this unless stated; only the main paper Table 3 line includes this. As appearance and geometry information is embedded in the volume, we want to refine the decomposition results by constraining them to align with rendered RGB and depth images. Specifically, we apply a CRF [13] with a pairwise Gaussian unary potential (θ_γ=3, w=15), a pairwise bilateral RGB potential (θ_γ=40, θ_β=13, w=10), and a pairwise bilateral depth potential (θ_γ=40, θ_β=13, w=20). Applying a CRF is similar to Amir et al. [1] on DINO-ViT in 2D, but our spacetime volume-integrated geometry provides a much stronger constraint on where the true edge is over time. The contribution of post processing is showed in Fig. 9 qualitatively. We see that small unimportant regions are removed, while still maintaining thin features and fine details. This trade-off is an application-level decision.

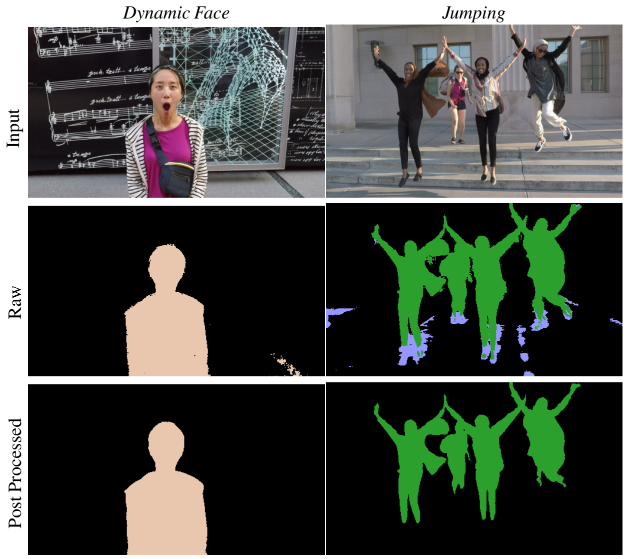

Figure 9: Post processing helps remove small noisy regions from the raw results while maintaining thin features.

Optimization routine and hyperparameters We optimize the combined loss with Adam optimizer, learning rate $5e^{-4}$, $\beta = (0.9, 0.999)$. We multiply prior losses for depth and optical flow by a decay rate. This rate starts at 1 and is divided by 10 at every 300,000th iteration. For SAFF, we set reprojection losses $\lambda_{\hat{\mathbf{s}}_{i\to j}} = 1.0$ and $\lambda_{\hat{\mathbf{a}}_{i\to j}} = 1.0$, and prior losses $\lambda_{\hat{\mathbf{s}}} = 0.04$ and $\lambda_{\hat{\mathbf{a}}} = 0.04$ [11]. For all other losses, we follow Li et al. [15]: $\lambda_{|\mathbf{f}|} = 0.1$, $\lambda_{\delta\mathbf{f}} = 0.1$, $\lambda_{\hat{w}} = 0.1$, $\lambda_{\hat{\mathbf{c}}_{i\to j}} = 1.0$, $\lambda_{\text{Cyc}} = 1.0$, $\lambda_{\hat{d}} = 0.04$, $\lambda_{\hat{\mathbf{p}}} = 0.02$, $\lambda_{\text{entropy}} = 0.001$.

SAFF, its ablations, and NSFF are optimized for 360k iterations. As D^{2}NeRF is a different architecture, we use the author's stated 100k iterations.

Computational cost. The code was developed on Ubuntu 20.10 in Python/PyTorch, and trained on NVIDIA GeForce RTX 3090, NVIDIA GeForce RTX A6000, and NVIDIA GeForce RTX 2080 TI GPUs. All operations assume access to only 1 GPU. The CUDA VRAM required for using a trained SAFF on 540 × 288 image size is 12 GB, while optimization requires 24 GB. The CPU RAM used to optimize, cluster, and render SAFF used 16 GB, while we use 36 GB to extract objects. Optimizing a SAFF for 360, 000 iterations takes 1 to 2 days, depending on hardware, and is similar in time to NSFF. Recent improvements have dramatically reduced this time for non-semantic-attention fields [19, 4], and we expect similar performance gains were these methods used as the underlying architecture. In terms of runtime, we preprocess the DINO-ViT features and so do not incur significant additional cost during optimization, and none during inference due to DINO-ViT. Per-frame rendering is 15% slower than NSFF due to the additional heads. Saliency-aware clustering takes a few seconds only.

The computational cost is currently expensive, but we see our work as a step towards integrating high and low level information for 4D semantic volume reconstruction. NeRF-based approaches are still some way from real-time performance, but there have been significant gains recently (e.g., Instant-NGP [19]). Building our approach upon a fast backbone like this would make our approach more practical. One additional note is that, versus supervised 2D segmentation networks that are typically trained to be feed forward and to make predictions quickly (e.g., ProposeReduce takes only a few seconds to process a short sequence), the output of our model is richer as a 4D reconstruction with time-varying correspondence.

With respect to scalability, longer sequences will take more time to process, and at some point the capacity of the MLPs will limit reconstruction detail. For scenes with more dynamic elements, many objects are not in principle a problem, but instances that spatially overlap cannot be separately determined due to the fact that DINO-ViT features are not instance-aware.

Network architecture We add two heads to the architecture of NSFF [15]. For the semantic head, we add a single linear layer (256 neurons) appended by a tanh layer, with output dimension 64 to match the size of the per-sequence PCA-reduced DINO-ViT features. For the saliency head, we add a single linear layer (256 neurons) appended by a sigmoid layer, with output dimension 1.

References

[1] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2021. 9

[2] Christopher P. Burgess, Loïc Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew M. Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *ArXiv*, abs/1901.11390, 2019. 1

[3] Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. The 2019 davis challenge on vos: Unsupervised multi-object segmentation. *arXiv:1905.00737*, 2019. 2

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *arXiv preprint arXiv:2203.09517*, 2022. 9

[5] Gamaleldin F. Elsayed, Aravindh Mahendran, Sjoerd van Steenkiste, Klaus Greff, Michael C. Mozer, and Thomas Kipf. SAVi++: Towards end-to-end object-centric learning from real-world videos. In *Advances in Neural Information Processing Systems*, 2022. 1

[6] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. In *NeurIPS*, 2022. 2, 3

[7] Klaus Greff, Raphael Lopez Kaufman, Rishabh Kabra, Nicholas Watters, Christopher P. Burgess, Daniel Zoran, Loïc Matthey, Matthew M. Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *ICML*, 2019. 1

[8] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. 9

[9] Rishabh Kabra, Daniel Zoran, Goker Erdogan, Loïc Matthey, Antonia Creswell, Matthew M. Botvinick, Alexander Lerchner, and Christopher P. Burgess. Simone: View-invariant, temporally-abstracted object representations via unsupervised video decomposition. *ArXiv*, abs/2106.03849, 2021. 1

[10] Thomas Kipf, Gamaleldin F. Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional Object-Centric Learning from Video. In *International Conference on Learning Representations (ICLR)*, 2022. 1

[11] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. In *Advances in Neural Information Processing Systems*, volume 35, 2022. 1, 9

[12] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 2

[13] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. 9

[14] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J. Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12871–12881, June 2022. 1

[15] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 3, 4, 9, 10

[16] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021. 2

[17] Huaijia Lin, Ruizheng Wu, Shu Liu, Jiangbo Lu, and Jiaya Jia. Video instance segmentation with a propose-reduce paradigm. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1719–1728, 2021. 1

[18] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020. 1

[19] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022. 9, 10

[20] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021. 3

[21] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[22] Cameron Smith, Hong-Xing Yu, Sergey Zakharov, Frédo Durand, Joshua B. Tenenbaum, Jiajun Wu, and Vincent Sitzmann. Unsupervised discovery and composition of object light fields. *ArXiv*, abs/2205.03923, 2022. 1

[23] Karl Stelzner, Kristian Kersting, and Adam R. Kosiorek. Decomposing 3d scenes into objects via unsupervised volume segmentation. *ArXiv*, abs/2104.01148, 2021. 1

[24] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representations. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022. 1

[25] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Öztireli. D2nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *ArXiv*, abs/2205.15838, 2022. 1, 4

[26] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *International Conference on Computer Vision (ICCV)*, October 2021. 1

[27] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[28] Hong-Xing Yu, Leonidas J. Guibas, and Jiajun Wu. Unsupervised discovery of object radiance fields. In *International Conference on Learning Representations*, 2022. 1

[29] Zhoutong Zhang, Forrester Cole, Zhengqi Li, Michael Rubinstein, Noah Snavely, and William T Freeman. Structure and motion from casual videos. In *European Conference on Computer Vision*, pages 20–37. Springer, 2022. 2