HODOR: High-level Object Descriptors for Object Re-segmentation in Video Learned from Static Images
Supplementary Material

Ali Athar[1] Jonathon Luiten[1,2] Alexander Hermans[1] Deva Ramanan[2] Bastian Leibe[1]

[1]RWTH Aachen University, Germany [2]Carnegie Mellon University, USA

{athar,luiten,hermans,leibe}@vision.rwth-aachen.de deva@cs.cmu.edu

I. Ablation: Temporal History for Inference

In Sec. 3.3 of the main text, we discussed how HODOR can effectively incorporate temporal history from past frames when predicting the object masks for a given frame. Fig. 1 plots the $\mathcal{J}\&\mathcal{F}$ score on DAVIS'17 val for different temporal history lengths. It can be seen that increasing the frame history from 1 to 4 frames yields an approximately linear performance improvement from 74.3 to 77.2. Thereafter, the $\mathcal{J}\&\mathcal{F}$ saturated at 7 frames at a $\mathcal{J}\&\mathcal{F}$ score of 77.5. Recall that because we only need the object/background descriptors for past frames rather than the full feature maps, the inference run-time is minimally affected by the temporal history length: increasing the temporal history from 1 to 10 frames only reduces the inference speed from 17.3 to 16.7 frame/s (reported speed is an average over 5 runs).

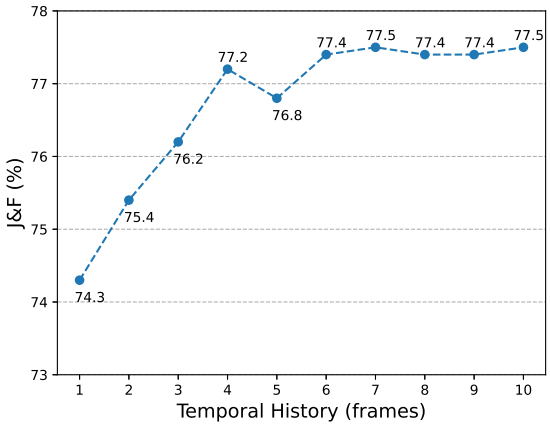

Figure 1. **Temporal History:** Performance on DAVIS'17 val for different temporal history lengths during inference.

II. Object Descriptors for Re-Identification

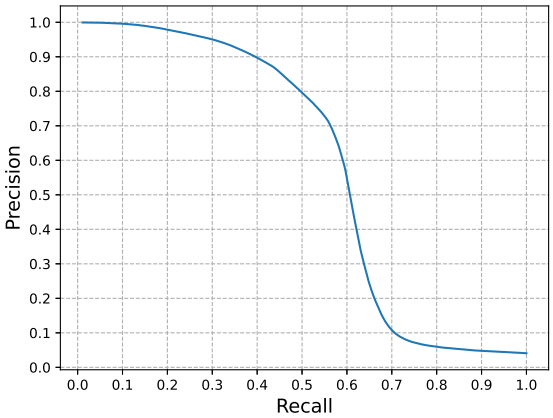

Figure 2. **Retrieval Task:** Precision-Recall curve for retrieval task on all object instances in DAVIS'17 val.

Our object descriptors are trained to encode an object's appearance so that it can be re-segmented, *i.e.* segmented in another video frame. Here we explore the applicability of these descriptors for a re-identification/retrieval task. For this, we consider the set of object descriptors for all frames for all video sequences in the DAVIS'17 validation set. For each descriptor, we calculate the Euclidean distance to all other descriptors, and then use these distances to retrieve other descriptors belonging to the same object instance. The resulting precision and recall is used to generate the precision-recall curve in Fig. 2 by averaging the retrieval scores across all descriptors.

Looking at the curve, we see that for each descriptor, $\sim$ 50% of the other descriptors belonging to the same object instance can be retrieved with a fairly high recall of $\sim$ 80%. Thereafter, the precision drops off sharply. Note, however, that this plot does not reflect the full quality of the object

descriptors for the Video Object Segmentation (VOS) task due to two main reasons:

1. This experiment disregards the image feature maps and directly compares the descriptors to one another. In the actual VOS use-case, we compute the dot-product between descriptors and image features to produce per-pixel logits which are then optimized to correctly segment the given object. In this experiment however, we directly compute the Euclidean distance between the descriptors themselves. Recall from Sec. 2 of the main text where we discussed that "Object-object Correspondence" based method use such re-identification techniques for associating objects over time. HODOR by contrast is an "Object-pixel Correspondence" based method.

2. For this experiment, we expect the network to learn descriptors which separate objects globally, *i.e*. across different video sequences. During training however, the network was only trained to distinguish between objects in the same image (or image sequence).

We hence conclude from this experiment that the object descriptors learned by our network can be used for re-identification tasks. However, the distribution of the descriptors for a given instance do not follow a unimodal distribution. This results in the sharp drop-off in recall seen in Fig. 2.

III. Visualizing Descriptor Feature Space

We attempt to visualize the object descriptors by projecting the 256-D object descriptors for all object instances in the DAVIS'17 validation to 2-D using t-SNE [4]. The resulting visualization is shown in Fig. 3 wherein the object crop for each descriptor is pasted at the projected 2-D coordinates. We can clearly see that descriptors for the same object instance are tightly clustered in a trajectory-like sequence. Though not visualized here, we observed that the trajectory-like shape usually corresponds to the frame index, which means that the descriptors tend to drift slightly over time.

We can also see a strong semantic trend in the descriptors. The lower-right portion of the image contains several of the '*car* objects, the top-right contains several *riders* (*i.e*. persons righting motorbikes, bicycles, horses). The center portions generally contains persons, and the lower-left portion of the image contains several animal classes *e.g*. *cow*, *dog*, *goat*. There are, however, noticeable exceptions. Note how there is a cluster of three fish on center-right, but the remaining two fish are very far away from them and each other.

Given that applications based on object embeddings often use a simple linear projection for further processing, we also visualize the object descriptors by projecting them to 2-D using Principal Component Analysis (PCA). The resulting illustration is given in Fig. 4. Here, in general, the descriptors are less distinguishable from each other, but the overall trend still holds true, *i.e*. descriptors for the same instance and similar semantic classes are generally located close to one another.

IV. Background Descriptors

In Sec. 3 of the main text, we explained how HODOR uses high-level descriptors to model the foreground objects and also the background. For the latter, all non-object pixels are combined into a background mask which is then split into 9 separate masks by dividing it into a 3×3 grid. One further minor architectural detail is that aside from the 9 background descriptors, we also predict an additional 'catch-all' background logit for each pixel. To do this, we apply a 3×3 convolution followed by a 1×1 convolution to the refined feature map $F^{4(L)}$ in the decoder (*cf*. Eq. 4 in the main text) to obtain a single-channel logit map. Then, before computing the softmax over the descriptors, we append the logit value for each pixel, representing another background descriptor. Formally speaking, Eq. 4 of the main text changes to the following:

$$\begin{aligned} F^{4(L)} &\longleftarrow \texttt{Conv}\left(F^4 + \texttt{upsample2}(F^{8(L)})\right) \\ M_{bc} &\longleftarrow \texttt{Conv}\left(\texttt{Conv}\left(F^{4(L)}\right)\right) \\ M' &\longleftarrow \texttt{Concatenate}\left(F^{4(L)} \cdot D, M_{bc}\right) \\ M &\longleftarrow \texttt{softmax}\left(\texttt{upsample4}\left(M'\right)\right) \end{aligned} \tag{1}$$

where M_{bc} is the background catch-all logit map and M' is an intermediate variable used to denote the concatenation of the dot-products $F^{4(L)} \cdot D$ and M_{bc}.

Note that these catch-all background logits are not propagated frame-by-frame when processing a video sequence. Without this technique, we obtain a $\mathcal{J}\&\mathcal{F}$ of 76.2 on DAVIS'17 val, which is 1.3 lower than the 77.5 reported in Table 1 of the main text.

Some example probability heatmaps for both the catch-all, as well as the 3×3 background grid can seen in Fig. 6 and 7. Note how the catch-all logits have high magnitudes mostly around object edges. The background descriptors sometimes associate themselves to an object-like region *e.g*. to the bush in the *horsejump-high* sequence (bottom-right descriptor), or to the black box/case in the *bike-packing* sequence (bottom-right descriptor). In general, we an also see a location bias based on the background mask patch which each descriptor was made to focus on by the encoder.

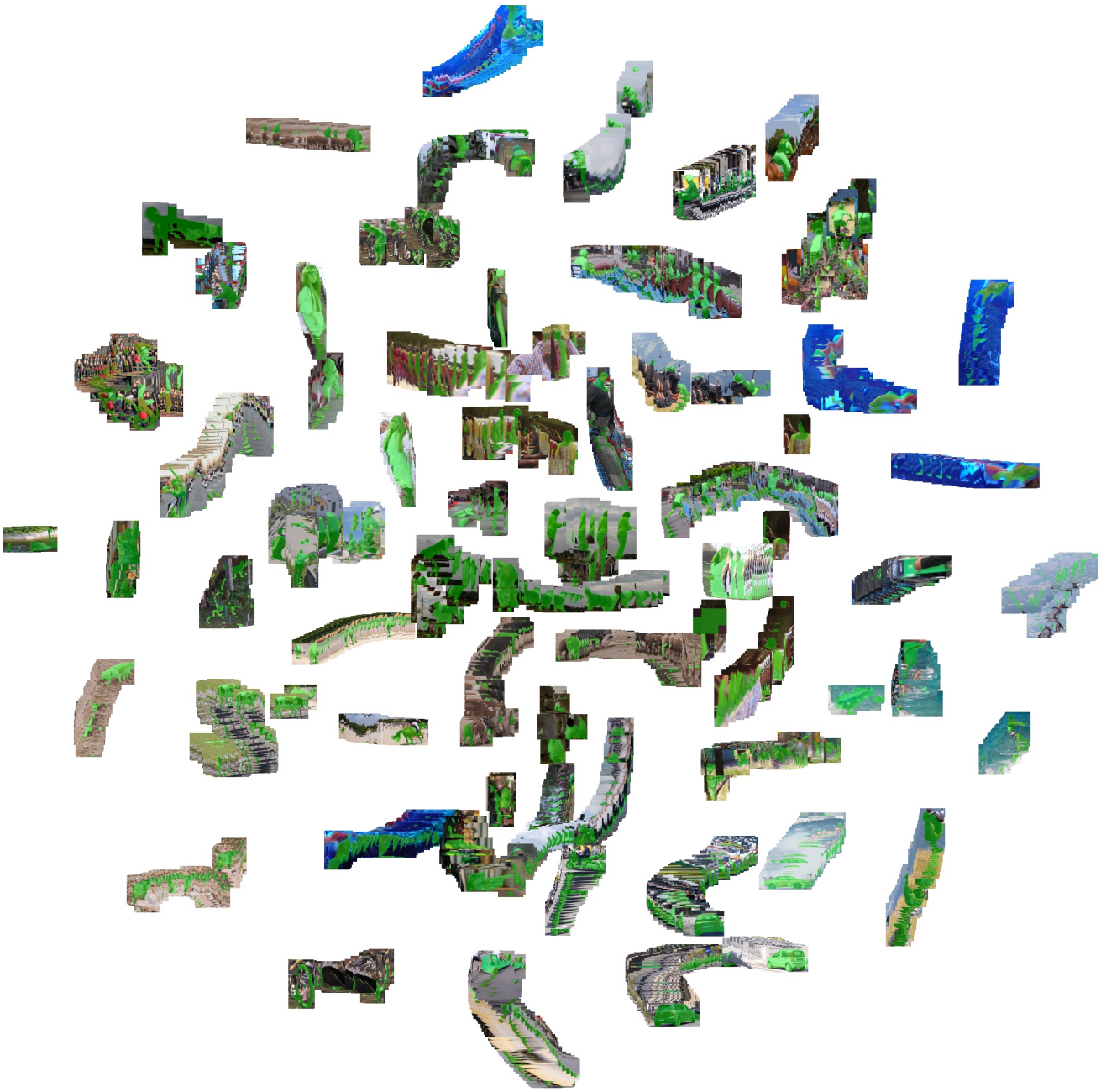

Figure 3. **Object Descriptor Visualization:** The descriptors for all object instances in DAVIS'17 val projected to 2-D using t-SNE.

V. Failure Cases

Fig. 5 illustrates some failure cases for our method. In the first, fourth and fifth columns, we see that HODOR incorrectly merges two difference instances because they are visually similar and spatially close to each other. In column two, we see that the method struggles to accurately segment the strings of the parachute, and likewise in column three we see that the bicycle in not correctly segmented. Since HODOR mainly operates on feature maps at the $8\times$ downsampled level w.r.t the input resolution, it often struggles with small objects.

VI. Implementation Details

Input Image Dimensions. For training, the input image is resized in an aspect-ratio preserving manner such that the pixel area is $\sim$ 300,000 and the lower dimension is an integer multiple of 32. During inference, the images are resized to have lower dimension 512.

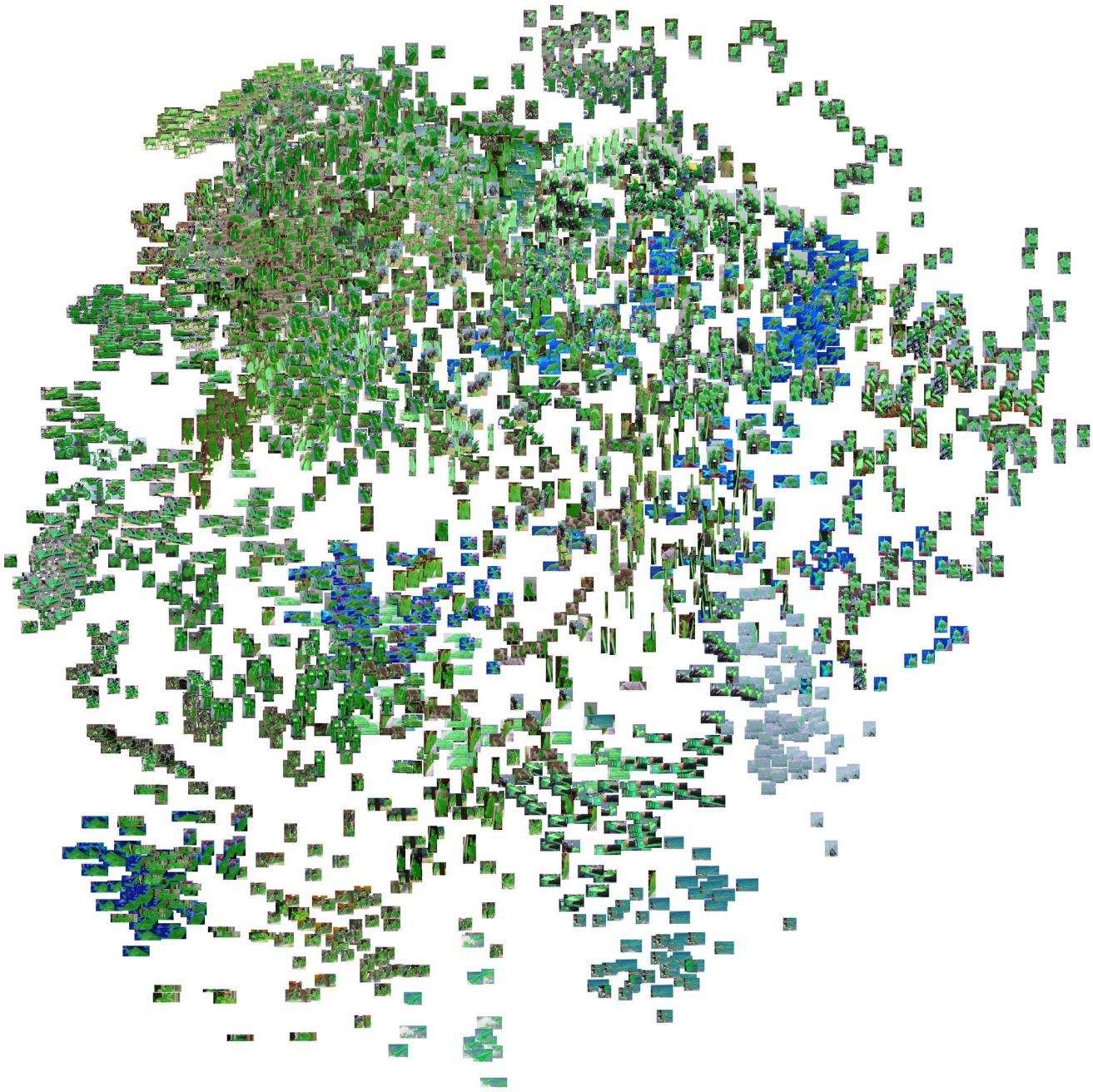

Figure 4. **Object Descriptor Visualization:** The descriptors for all object instances in DAVIS'17 val projected to 2-D using PCA.

Loss Function. To supervise the predicted masks, we use the sum of the cross-entropy loss and the DICE loss (both weighted by unity).

Learning Rate Schedule. When training on COCO [3], the learning rate is first warmed up from 0 to 10^{-4} over 10k iterations. Then at 100k iterations we apply step decay and reduce the learning rate to 10^{-5}. The training is then run for a further 150k iterations. For training on annotated video frames (both augmented frames and cyclic consistency), we fine-tune the network by loading weights from the COCO augmented sequence checkpoint, and then warm-up the learning rate from 0 to 10^{-5} over 10k iterations. The network then trains for a further 10k iterations with constant learning rate. Since there are only $\sim$ 3500 labeled image frames under this setting, the model tends to over-fit if trained longer.

Training Time. The main training on COCO for 250k iterations requires $\sim$ 2 days on 4 Nvidia 3090 GPUs. The

Figure 5. **Failure Examples:** From left to right: First three columns show the *camel*, *paragliding-launch* and *bmx-trees* sequences from the DAVIS. Last two columns show sequences *282651c6f7* and *c280d21988* from YouTube-VOS.

fine-tuning for 20k iterations requires less than 6 hrs.

Soft Attention-masking Scaling Factors. In Sec. 3.1 of the main text, we explained our novel attention-masking mechanism which applies an additive offset to the $\text{Key}^T \text{Query}$ matrix. The offset is the mask value scaled by a positive scalar α. We initialize α separately for each of the 8 attention heads as follows: [32, 32, 16, 16, 8, 8, 4, 4]. These are applied as learnable parameters which can be optimized by the network.

Image Augmentations. For the results reported in Table 2 of the main text, we trained on image sequences generated by applying random affine transformations to COCO images. We use the popular `imgaug` library [2] for this task. The range of values for each transformation type are as follows:

- Translation: $0 - 25\%$ w.r.t the dimension size.
- Rotation: $0 - 10\%$ in both directions.
- Shear: $0 - 10\%$ along both axes.
- Crop: $60 - 90\%$ of the image is retained.

Note that each image in the training image sequence is generated by applying the transformations to the original image, *i.e*. we do not apply sequential augmentation. Aside from these geometric augmentations, we also apply color augmentations as follows:

- Hue : $0 - 12\%$
- Saturation: $0 - 12\%$
- Contrast (linear): $0 - 5\%$.
- Brightness: $0 - 25\%$

Our color augmentation strategy is inspired from that used by Cheng *et al*. [1] for STCN.

VI.1. Qualitative Results Video

The zip file for our supplementary material contains a short video file with qualitative results for some video sequences from the DAVIS'17 val set. Additionally, there are a couple of video snippets from the popular Game of Thrones TV series where annotated the first frame mask ourselves for the 'Hodor' character who is the namesake for our method name.

References

[1] H. K. Cheng, Y.-W. Tai, and C.-K. Tang. Rethinking Space-Time Networks with Improved Memory Coverage for Efficient Video Object Segmentation. In *NeurIPS*, 2021. 5

[2] A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, C. Reinders, S. Yadav, J. Banerjee, G. Vecsei, A. Kraft, Z. Rui, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernández, F.-M. De Rainville, C.-H. Weng, A. Ayala-Acevedo, R. Meudec, M. Laporte, et al. imgaug. `https://github.com/aleju/imgaug`, 2020. Online; accessed 01-Feb-2020. 5

[3] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014. 4

[4] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008. 2

Figure 6. **Background Descriptor Visualization:** Each block shows the ground truth foreground object mask(s) (top left) and a total of 10 background probability heatmaps, corresponding 1 catch-all heatmap (mid left) and the full 3×3 background grid heatmaps (columns 2-4). Not the location bias in the 3×3 grid, where the grid-based background descriptor initialization sometimes causes the background descriptors to attach to a nearby object.

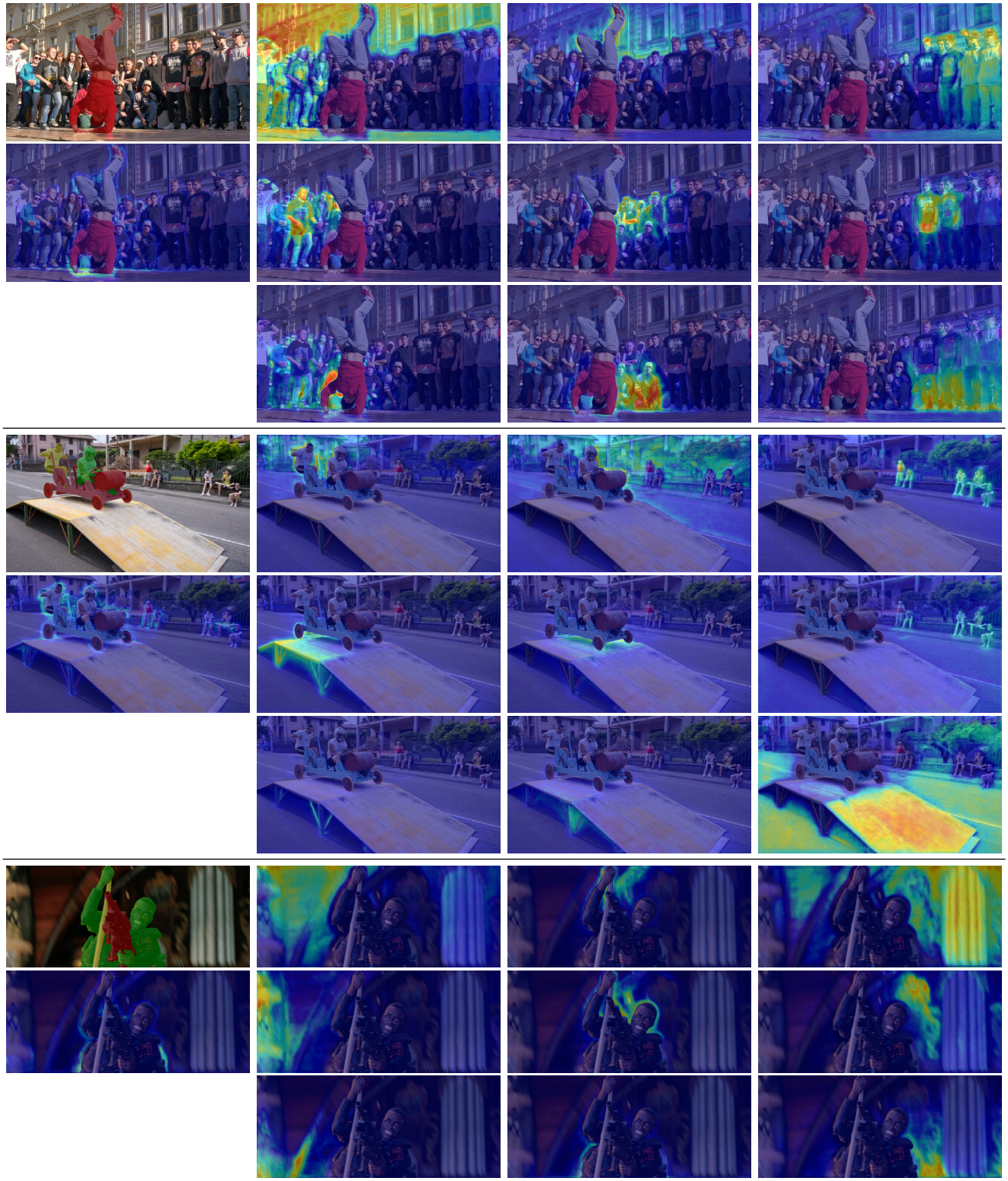

Figure 7. **Background Descriptor Visualization (continued):** Each block shows the ground truth foreground object mask(s) (top left) and a total of 10 background probability heatmaps, corresponding 1 catch-all heatmap (mid left) and the full 3×3 background grid heatmaps (columns 2-4). Not the location bias in the 3×3 grid, where the grid-based background descriptor initialization sometimes causes the background descriptors to attach to a nearby object.